Cluster Server 8.0 Administrator's Guide - Linux

VERITAS™

Last updated: 2021-12-21

Legal Notice

Copyright © 2021 Veritas Technologies LLC. All rights reserved.

Veritas and the Veritas Logo are trademarks or registered trademarks of Veritas Technologies LLC or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

This product may contain third-party software for which Veritas is required to provide attribution to the third-party (“Third-Party Programs”). Some of the Third-Party Programs are available under open source or free software licenses. The License Agreement accompanying the Software does not alter any rights or obligations you may have under those open source or free software licenses. Refer to the third-party legal notices document accompanying this Veritas product or available at:
https://www.veritas.com/about/legal/license-agreements

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Veritas Technologies LLC and its licensors, if any.

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. VERITAS TECHNOLOGIES LLC SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, et seq. "Commercial Computer Software and Commercial Computer Software Documentation," as applicable, and any successor regulations, whether delivered by Veritas as on premises or hosted services. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Veritas Technologies LLC
2625 Augustine Drive
Santa Clara, CA 95054
http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policies. For information about our support offerings and how to contact Technical Support, visit our website:
https://www.veritas.com/support

You can manage your Veritas account information at the following URL:
https://my.veritas.com

If you have questions regarding an existing support agreement, please email the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The latest documentation is available on the Veritas website:
https://sort.veritas.com/documents

Documentation feedback

Your feedback is important to us. Suggest improvements or report errors or omissions to the documentation. Include the document title, document version, chapter title, and section title of the text on which you are reporting. Send feedback to:
infoscaledocs@veritas.com

You can also see documentation information or ask a question on the Veritas community site:
http://www.veritas.com/community/

Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:
https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Section 1 Clustering concepts and terminology 24

Chapter 1 Introducing Cluster Server 25

About Cluster Server 25
How VCS detects failure 25
How VCS ensures application availability 26
About cluster control guidelines 27
Defined start, stop, and monitor procedures 27
Ability to restart the application in a known state 28
External data storage 28
Licensing and host name issues 29
About the physical components of VCS 29
About VCS nodes 29
About shared storage 30
About networking 30
Logical components of VCS 30
About resources and resource dependencies 31
Categories of resources 33
About resource types 33
About service groups 34
Types of service groups 34
About the ClusterService group 35
About the cluster UUID 35
About agents in VCS 36
About agent functions 37
About resource monitoring 38
Agent classifications 41
VCS agent framework 42
About cluster control, communications, and membership 42
About security services 45
Components for administering VCS 49
Putting the pieces together 51

Chapter 2 About cluster topologies 53

Basic failover configurations 53
Asymmetric or active / passive configuration 53
Symmetric or active / active configuration 54
About N-to-1 configuration 55
About advanced failover configurations 56
About the N + 1 configuration 56
About the N-to-N configuration 57
Cluster topologies and storage configurations 58
About basic shared storage cluster 58
About campus, or metropolitan, shared storage cluster 59
About shared nothing clusters 60
About replicated data clusters 60
About global clusters 61

Chapter 3 VCS configuration concepts 63

About configuring VCS 63
VCS configuration language 64
About the main.cf file 64
About the SystemList attribute 67
Initial configuration 67
Including multiple .cf files in main.cf 68
About the types.cf file 68
About VCS attributes 70
About attribute data types 70
About attribute dimensions 71
About attributes and cluster objects 71
Attribute scope across systems: global and local attributes 73
About attribute life: temporary attributes 73
Size limitations for VCS objects 73
VCS keywords and reserved words 74
About managing VCS modules 74
Disabling CmdServer 76
VCS environment variables 76
Defining VCS environment variables 80
Environment variables to start and stop VCS modules 81

Section 2 Administration - Putting VCS to work 85

Chapter 4 About the VCS user privilege model 86

About VCS user privileges and roles 86
VCS privilege levels 86
User roles in VCS 87
Hierarchy in VCS roles 88
User privileges for CLI commands 88
User privileges for cross-cluster operations 89
User privileges for clusters that run in secure mode 89
About the cluster-level user 90
How administrators assign roles to users 90
User privileges for OS user groups for clusters running in secure mode 90
VCS privileges for users with multiple roles 91

Chapter 5 Administering the cluster from the command line 93

About administering VCS from the command line 94
Symbols used in the VCS command syntax 94
How VCS identifies the local system 95
About specifying values preceded by a dash (-) 95
About the -modify option 95
Encrypting VCS passwords 96
Encrypting agent passwords 97
About installing a VCS license 100
Installing and updating license keys using vxlicinst 101
Setting or changing the product level for keyless licensing 101
Administering LLT 102
Displaying the cluster details and LLT version for LLT links 103
Adding and removing LLT links 103
Configuring aggregated interfaces under LLT 105
Configuring destination-based load balancing for LLT 107
Configuring heartbeat threads to improve cluster resiliency 107
Administering the AMF kernel driver 109
Starting VCS 110
Starting the VCS engine (HAD) and related processes 111
Starting VCS in a customized environment 112
Stopping VCS 113

Stopping VCS without evacuating service groups 115
Stopping the VCS engine and related processes 115
About stopping VCS without the -force option 116
About stopping VCS with options other than the -force option 116
About controlling the hastop behavior by using the EngineShutdown attribute 117
Additional considerations for stopping VCS 117
Logging on to VCS 118
Running high availability commands (HA) commands as non-root users on clusters in secure mode 119
About managing VCS configuration files 120
About multiple versions of .cf files 120
Verifying a configuration 120
Scheduling automatic backups for VCS configuration files 120
Saving a configuration 121
Setting the configuration to read or write 121
Displaying configuration files in the correct format 122
About managing VCS users from the command line 122
Adding a user 122
Assigning and removing user privileges 124
Modifying a user 125
Deleting a user 125
Displaying a user 126
About querying VCS 126
Querying service groups 127
Querying resources 128
Querying resource types 129
Querying agents 130
Querying systems 131
Querying clusters 131
Querying status 132
Querying log data files (LDFs) 133
Using conditional statements to query VCS objects 134
About administering service groups 135
Adding and deleting service groups 135
Modifying service group attributes 136
Bringing service groups online 138
Taking service groups offline 139
Switching service groups 139
Migrating service groups 140
Freezing and unfreezing service groups 141
Enabling and disabling service groups 141

Enabling and disabling priority based failover for a service group 142
Clearing faulted resources in a service group 143
Flushing service groups 144
Linking and unlinking service groups 145
Administering agents 146
About administering resources 147
About adding resources 147
Adding resources 147
Deleting resources 148
Adding, deleting, and modifying resource attributes 148
Defining attributes as local 150
Defining attributes as global 152
Enabling and disabling intelligent resource monitoring for agents manually 152
Enabling and disabling IMF for agents by using script 154
Linking and unlinking resources 159
Bringing resources online 160
Taking resources offline 160
Probing a resource 161
Clearing a resource 161
About administering resource types 162
Adding, deleting, and modifying resource types 162
Overriding resource type static attributes 163
About initializing resource type scheduling and priority attributes 164
Setting scheduling and priority attributes 164
Administering systems 165
About administering clusters 166
Configuring and unconfiguring the cluster UUID value 167
Retrieving version information 168
Adding and removing systems 169
Changing ports for VCS 170
Setting cluster attributes from the command line 171
About initializing cluster attributes in the configuration file 172
Enabling and disabling secure mode for the cluster 173
Migrating from secure mode to secure mode with FIPS 174
About cluster formation with different versions of the VCS engine 174
Using the -wait option in scripts that use VCS commands 176
Running HA fire drills 176
About administering simulated clusters from the command line 177

Chapter 6 Configuring applications and resources in VCS 179

Configuring resources and applications 179
VCS bundled agents for UNIX 180
About Storage agents 180
About Network agents 182
About File share agents 184
About Services and Application agents 185
About VCS infrastructure and support agents 186
About Testing agents 187
Configuring NFS service groups 188
About NFS 188
Configuring NFS service groups 189
Sample configurations 197
About configuring the RemoteGroup agent 218
About the ControlMode attribute 219
About the ReturnIntOffline attribute 220
Configuring a RemoteGroup resource 220
Service group behavior with the RemoteGroup agent 222
About configuring Samba service groups 223
Sample configuration for Samba in a failover configuration 223
Configuring the Coordination Point agent 225
About testing resource failover by using HA fire drills 225
About HA fire drills 225
About running an HA fire drill 226

Chapter 7 Predicting VCS behavior using VCS Simulator 227

About VCS Simulator 227
VCS Simulator ports 228
Administering VCS Simulator from the command line interface 229
Starting VCS Simulator from the command line interface 230
Administering simulated clusters from the command line 232

Section 3 VCS communication and operations 234

Chapter 8 About communications, membership, and data protection in the cluster 235

About cluster communications 235
About intra-system communications 236

About inter-system cluster communications 236
About cluster membership 240
Initial joining of systems to cluster membership 240
Ongoing cluster membership 244
About membership arbitration 245
About membership arbitration components 246
About server-based I/O fencing 253
About majority-based fencing 256
About making CP server highly available 257
About the CP server database 258
Recommended CP server configurations 258
About the CP server service group 261
About the CP server user types and privileges 263
About secure communication between the VCS cluster and CP server 263
About data protection 265
About SCSI-3 Persistent Reservation 265
About I/O fencing configuration files 266
Examples of VCS operation with I/O fencing 268
About the I/O fencing algorithm 268
Example: Two-system cluster where one system fails 269
Example: Four-system cluster where cluster interconnect fails 270
How I/O fencing works in different event scenarios 272
About cluster membership and data protection without I/O fencing 276
About jeopardy 277
About Daemon Down Node Alive (DDNA) 277
Examples of VCS operation without I/O fencing 278
Example: Four-system cluster without a low priority link 278
Example: Four-system cluster with low priority link 280
Summary of best practices for cluster communications 283

Chapter 9 Administering I/O fencing 285

About administering I/O fencing 285
About the vxfentsthdw utility 286
General guidelines for using the vxfentsthdw utility 286
About the vxfentsthdw command options 287
Testing the coordinator disk group using the -c option of vxfentsthdw 289
Performing non-destructive testing on the disks using the -r option 291

Testing the shared disks using the vxfentsthdw -m option 291
Testing the shared disks listed in a file using the vxfentsthdw -f option 293
Testing all the disks in a disk group using the vxfentsthdw -g option 293
Testing a disk with existing keys 294
Testing disks with the vxfentsthdw -o option 294
About the vxfenadm utility 295
About the I/O fencing registration key format 295
Displaying the I/O fencing registration keys 296
Verifying that the nodes see the same disk 299
About the vxfenclearpre utility 300
Removing preexisting keys 301
About the vxfenswap utility 303
Replacing I/O fencing coordinator disks when the cluster is online 304
Replacing the coordinator disk group in a cluster that is online 308
Adding disks from a recovered site to the coordinator disk group 312
Refreshing lost keys on coordinator disks 314
About administering the coordination point server 316
CP server operations (cpsadm) 316
Cloning a CP server 317
Adding and removing VCS cluster entries from the CP server database 319
Adding and removing a VCS cluster node from the CP server database 320
Adding or removing CP server users 320
Listing the CP server users 321
Listing the nodes in all the VCS clusters 321
Listing the membership of nodes in the VCS cluster 321
Preempting a node 321
Registering and unregistering a node 322
Enable and disable access for a user to a VCS cluster 322
Starting and stopping CP server outside VCS control 323
Checking the connectivity of CP servers 323
Adding and removing virtual IP addresses and ports for CP servers at run-time 323
Taking a CP server database snapshot 326
Replacing coordination points for server-based fencing in an online cluster 326

Refreshing registration keys on the coordination points for server-based fencing 328
About configuring a CP server to support IPv6 or dual stack 330
Deployment and migration scenarios for CP server 334
About migrating between disk-based and server-based fencing configurations 340
Migrating from disk-based to server-based fencing in an online cluster 340
Migrating from server-based to disk-based fencing in an online cluster 341
Migrating between fencing configurations using response files 341
Enabling or disabling the preferred fencing policy 347
About I/O fencing log files 349

Chapter 10 Controlling VCS behavior 350

VCS behavior on resource faults 350
Critical and non-critical resources 351
VCS behavior diagrams 351
About controlling VCS behavior at the service group level 353
About the AutoRestart attribute 353
About controlling failover on service group or system faults 354
About defining failover policies 355
About AdaptiveHA 356
About system zones 360
About sites 360
Load-based autostart 361
About freezing service groups 361
About controlling Clean behavior on resource faults 361
Clearing resources in the ADMIN_WAIT state 362
About controlling fault propagation 364
Customized behavior diagrams 364
About preventing concurrency violation 365
VCS behavior for resources that support the intentional offline functionality 369
VCS behavior when a service group is restarted 370
About controlling VCS behavior at the resource level 371
Resource type attributes that control resource behavior 371
How VCS handles resource faults 373
VCS behavior after a resource is declared faulted 377
VCS behavior when a resource is restarted 379
About disabling resources 380

Changing agent file paths and binaries 383
VCS behavior on loss of storage connectivity 384
Disk group configuration and VCS behavior 384
How VCS attributes control behavior on loss of storage connectivity 385
VCS behavior when a disk group is disabled 386
Recommendations to ensure application availability 387
Service group workload management 387
About enabling service group workload management 387
System capacity and service group load 387
System limits and service group prerequisites 389
About capacity and limits 389
Sample configurations depicting workload management 390
System and Service group definitions 390
Sample configuration: Basic four-node cluster 391
Sample configuration: Complex four-node cluster 395
Sample configuration: Server consolidation 399

Chapter 11 The role of service group dependencies 407

About service group dependencies 407
About dependency links 407
About dependency limitations 411
Service group dependency configurations 411
About failover parent / failover child 411
Frequently asked questions about group dependencies 425
About linking service groups 427
About linking multiple child service groups 427
Dependencies supported for multiple child service groups 427
Dependencies not supported for multiple child service groups ... 4 2 8
VCS behavior with service group dependencies 428
Online operations in group dependencies 428
Offline operations in group dependencies 429
Switch operations in group dependencies 429

Section 4 Administration - Beyond the basics 430

Chapter 12 VCS event notification 431

About VCS event notification 431
Event messages and severity levels 433
About persistent and replicated message queue 433

How HAD deletes messages 433
Components of VCS event notification 434
About the notifier process 434
About the hanotify utility 435
About VCS events and traps 436
Events and traps for clusters 436
Events and traps for agents 437
Events and traps for resources 437
Events and traps for systems 439
Events and traps for service groups 440
SNMP-specific files 441
Trap variables in VCS MIB 442
About monitoring aggregate events 445
How to detect service group failover 445
How to detect service group switch 445
About configuring notification 445

Chapter 13 VCS event triggers 447

About VCS event triggers 447
Using event triggers 448
Performing multiple actions using a trigger 448
List of event triggers 449
About the dumptunables trigger 449
About the globalcounter_not_updated trigger 449
About the injeopardy event trigger 450
About the loadwarning event trigger 450
About the nofailover event trigger 451
About the postoffline event trigger 452
About the postonline event trigger 452
About the preonline event trigger 452
About the resadminwait event trigger 453
About the resfault event trigger 454
About the resnotoff event trigger 455
About the resrestart event trigger 455
About the resstatechange event trigger 456
About the sysoffline event trigger 457
About the sysup trigger 458
About the sysjoin trigger 458
About the unable_to_restart_agent event trigger 458
About the unable_to_restart_had event trigger 459
About the violation event trigger 459

Chapter 14 Virtual Business Services 460

About Virtual Business Services 460
Features of Virtual Business Services 460
Sample virtual business service configuration 461
About choosing between VCS and VBS level dependencies 463

Section 5 Veritas High Availability Configuration wizard 464

Chapter 15 Introducing the Veritas High Availability Configuration wizard 465

About the Veritas High Availability Configuration wizard 465
Launching the Veritas High Availability Configuration wizard 466
Typical VCS cluster configuration in a VMware virtual environment 468
Typical VCS cluster configuration in a physical environment 470

Chapter 16 Administering application monitoring from the Veritas High Availability view 471

Administering application monitoring from the Veritas High Availability view 471
Understanding the Veritas High Availability view 472
To view the status of configured applications 475
To configure or unconfigure application monitoring 475
To start or stop applications 477
To suspend or resume application monitoring 478
To switch an application to another system 479
To add or remove a failover system 480
To clear Fault state 483
To resolve a held-up operation 484
To determine application state 484
To remove all monitoring configurations 484
To remove VCS cluster configurations 485
Administering application monitoring settings 485

Section 6 Cluster configurations for disaster recovery 487

Chapter 17 Connecting clusters–Creating global clusters 4 8 8

How VCS global clusters work 488
VCS global clusters: The building blocks 489
Visualization of remote cluster objects 490
About global service groups 490
About global cluster management 491
About serialization–The Authority attribute 492
About resiliency and "Right of way" 493
VCS agents to manage wide-area failover 493
About the Steward process: Split-brain in two-cluster global clusters 496
Secure communication in global clusters 497
Prerequisites for global clusters 498
Prerequisites for cluster setup 498
Prerequisites for application setup 498
Prerequisites for wide-area heartbeats 499
Prerequisites for ClusterService group 499
Prerequisites for replication setup 500
Prerequisites for clusters running in secure mode 500
About planning to set up global clusters 500
Setting up a global cluster 501
Configuring application and replication for global cluster setup 502
Configuring clusters for global cluster setup 503
Configuring service groups for global cluster setup 511
Configuring a service group as a global service group 515
About IPv6 support with global clusters 516
Prerequisites for configuring a global cluster to support IPv6 516
Migrating an InfoScale Availability cluster from IPv4 to IPv6 when Virtual IP (ClusterAddress) is configured 516
Migrating an InfoScale Availability cluster to IPv6 in a GCO deployment 518
About cluster faults 520
About the type of failure 521
Switching the service group back to the primary 521
About setting up a disaster recovery fire drill 522

About creating and configuring the fire drill service group manually ... 523
About configuring the fire drill service group using the Fire Drill Setup wizard ... 526
Verifying a successful fire drill ... 528
Scheduling a fire drill ... 528
Multi-tiered application support using the RemoteGroup agent in a global environment ... 528
Test scenario for a multi-tiered environment ... 530
About the main.cf file for cluster 1 ... 531
About the main.cf file for cluster 2 ... 532
About the main.cf file for cluster 3 ... 533
About the main.cf file for cluster 4 ... 534

Chapter 18 Administering global clusters from the command line ... 536

About administering global clusters from the command line ... 536
About global querying in a global cluster setup ... 537
Querying global cluster service groups ... 537
Querying resources across clusters ... 538
Querying systems ... 540
Querying clusters ... 540
Querying status ... 542
Querying heartbeats ... 542
Administering global service groups in a global cluster setup ... 544
Administering resources in a global cluster setup ... 546
Administering clusters in global cluster setup ... 546
Managing cluster alerts in a global cluster setup ... 547
Changing the cluster name in a global cluster setup ... 548
Removing a remote cluster from a global cluster setup ... 549
Administering heartbeats in a global cluster setup ... 549

Chapter 19 Setting up replicated data clusters ... 551

About replicated data clusters ... 551
How VCS replicated data clusters work ... 552
About setting up a replicated data cluster configuration ... 553
About typical replicated data cluster configuration ... 553
About setting up replication ... 554
Configuring the service groups ... 555
Configuring the service group dependencies ... 556
About migrating a service group ... 556
Switching the service group ... 557

About setting up a fire drill 557

Chapter 20 Setting up campus clusters 558

About campus cluster configuration 558
VCS campus cluster requirements 559
Typical VCS campus cluster setup 560
How VCS campus clusters work 561
About I/O fencing in campus clusters 565
About setting up a campus cluster configuration 566
Preparing to set up a campus cluster configuration 566
Configuring I/O fencing to prevent data corruption 567
Configuring VxVM disk groups for campus cluster configuration 567
Configuring VCS service group for campus clusters 569
Fire drill in campus clusters 569
About the DiskGroupSnap agent 570
About running a fire drill in a campus cluster 570
Configuring the fire drill service group 571
Running a successful fire drill in a campus cluster 571

Section 7 Troubleshooting and performance 573

Chapter 21 VCS performance considerations 574

How cluster components affect performance 574
How kernel components (GAB and LLT) affect performance 5 7 5
How the VCS engine (HAD) affects performance 575
How agents affect performance 576
How the VCS graphical user interfaces affect performance 577
How cluster operations affect performance 577
VCS performance consideration when booting a cluster system 578
VCS performance consideration when a resource comes online 579
VCS performance consideration when a resource goes offline 579
VCS performance consideration when a service group comes online 579
VCS performance consideration when a service group goes offline 580
VCS performance consideration when a resource fails 580

VCS performance consideration when a system fails 581
VCS performance consideration when a network link fails 582
VCS performance consideration when a system panics 582
VCS performance consideration when a service group switches over 585
VCS performance consideration when a service group fails over 585
About scheduling class and priority configuration 586
About priority ranges 586
Default scheduling classes and priorities 586
VCS agent statistics 587
Tracking monitor cycle times 588
VCS attributes enabling agent statistics 589
About VCS tunable parameters 590
About LLT tunable parameters 590
About GAB tunable parameters 600
About VXFEN tunable parameters 607
About AMF tunable parameters 610

Chapter 22 Troubleshooting and recovery for VCS 612

VCS message logging 613
Log unification of VCS agent's entry points 614
Enhancing First Failure Data Capture (FFDC) to troubleshoot VCS resource's unexpected behavior 614
GAB message logging 615
Enabling debug logs for agents 616
Enabling debug logs for IMF 617
Enabling debug logs for the VCS engine 617
Enabling debug logs for VxAT 618
About debug log tags usage 619
Gathering VCS information for support analysis 620
Gathering LLT and GAB information for support analysis 622
Gathering IMF information for support analysis 623
Message catalogs 623
Troubleshooting the VCS engine 625
HAD diagnostics 625
HAD restarts continuously 625
DNS configuration issues cause GAB to kill HAD 626
Seeding and I/O fencing 626
Preonline IP check 626
Troubleshooting Low Latency Transport (LLT) 627
LLT startup script displays errors 627

LLT detects cross links usage 627
LLT link status messages 628
Cluster does not form after installing and starting VCS or SFHA 630
Troubleshooting Group Membership Services/Atomic Broadcast (GAB) 631
Delay in port reopen 631
Node panics due to client process failure 631
Troubleshooting VCS startup 632
"VCS:10622 local configuration missing" 632
"VCS:10623 local configuration invalid" 632
"VCS:11032 registration failed. Exiting" 633
"Waiting for cluster membership." 633
Troubleshooting issues with systemd unit service files 633
If a unit service has failed and the corresponding module is still loaded, systemd cannot unload it and so its package cannot be removed 633
If a unit service is active and the corresponding process is stopped outside of systemd, the service cannot be started again using 'systemctl start' 635
If a unit service takes longer than the default timeout to stop or start the corresponding service, it goes into the Failed state 638
Troubleshooting Intelligent Monitoring Framework (IMF) 639
Troubleshooting service groups 642
VCS does not automatically start service group 642
System is not in RUNNING state 642
Service group not configured to run on the system 642
Service group not configured to autostart 642
Service group is frozen 642
Failover service group is online on another system 643
A critical resource faulted 643
Service group autodisabled 643
Service group is waiting for the resource to be brought online/taken offline 643
Service group is waiting for a dependency to be met. 644
Service group not fully probed. 644
Service group does not fail over to the forecasted system 645
Service group does not fail over to the BiggestAvailable system even if FailOverPolicy is set to BiggestAvailable 645
Restoring metering database from backup taken by VCS 646
Initialization of metering database fails 647

Error message appears during service group failover or switch 648
Troubleshooting resources 648
Service group brought online due to failover 648
Waiting for service group states 648
Waiting for child resources 648
Waiting for parent resources 648
Waiting for resource to respond 649
Agent not running 649
The Monitor entry point of the disk group agent returns ONLINE even if the disk group is disabled 649
Troubleshooting sites 650
Online propagate operation was initiated but service group failed to be online 650
VCS panics nodes in the preferred site during a network-split ... 6 5 0
Configuring of stretch site fails 650
Renaming a Site 651
Troubleshooting I/O fencing 651
Node is unable to join cluster while another node is being ejected 651
The vxfentsthdw utility fails when SCSI TEST UNIT READY command fails 651
Manually removing existing keys from SCSI-3 disks 652
System panics to prevent potential data corruption 653
Cluster ID on the I/O fencing key of coordinator disk does not match the local cluster's ID 654
Fencing startup reports preexisting split-brain 655
Registered keys are lost on the coordinator disks 657
Replacing defective disks when the cluster is offline 658
The vxfenswap utility exits if rcp or scp commands are not functional 660
Troubleshooting CP server 660
Troubleshooting server-based fencing on the VCS cluster nodes 662
Issues during online migration of coordination points 663
Troubleshooting notification 664
Notifier is configured but traps are not seen on SNMP console. 664
Troubleshooting and recovery for global clusters 664
Disaster declaration 665
Lost heartbeats and the inquiry mechanism 665
VCS alerts 666

Troubleshooting the steward process 668
Troubleshooting licensing 669
Validating license keys 669
Licensing error messages 670
Troubleshooting secure configurations 671
FIPS mode cannot be set 671
Broker does not start 671
AT initialization fails 672
Troubleshooting wizard-based configuration issues 672
Running the 'hastop -all' command detaches virtual disks 672
Troubleshooting issues with the Veritas High Availability view 672
Veritas High Availability view does not display the application monitoring status 673
Veritas High Availability view may freeze due to special characters in application display name 673
In the Veritas High Availability tab, the Add Failover System link is dimmed 674

Section 8 Appendixes 675

Appendix A VCS user privileges—administration matrices 676

About administration matrices 676
Administration matrices 676
Agent Operations (haagent) 677
Attribute Operations (haattr) 677
Cluster Operations (haclus, haconf) 677
Service group operations (hagrp) 678
Heartbeat operations (hahb) 679
Log operations (halog) 680
Resource operations (hares) 680
System operations (hasys) 681
Resource type operations (hatype) 682
User operations (hauser) 682

Appendix B VCS commands: Quick reference 684

About this quick reference for VCS commands 684
VCS command line reference 684

Appendix C Cluster and system states 688

Remote cluster states 688
Examples of cluster state transitions 689
System states 690
Examples of system state transitions 692

Appendix D VCS attributes 693

About attributes and their definitions 693
Resource attributes 694
Resource type attributes 703
Service group attributes 719
System attributes 746
Cluster attributes 760
Heartbeat attributes (for global clusters) 775
Remote cluster attributes 777
Site attributes 780

Appendix E Accessibility and VCS 782

About accessibility in VCS 782
Navigation and keyboard shortcuts 782
Navigation in the Java Console 783
Navigation in the Web console 783
Support for accessibility settings 783
Support for assistive technologies 783

Section 1

Clustering concepts and terminology

- Chapter 1. Introducing Cluster Server
- Chapter 2. About cluster topologies
- Chapter 3. VCS configuration concepts

Chapter 1

Introducing Cluster Server

This chapter includes the following topics:

- About Cluster Server
- About cluster control guidelines
- About the physical components of VCS
- Logical components of VCS
- Putting the pieces together

About Cluster Server

Cluster Server (VCS) connects multiple, independent systems into a management framework for increased availability. Each system, or node, runs its own operating system and cooperates at the software level to form a cluster. VCS links commodity hardware with intelligent software to provide application failover and control. When a node or a monitored application fails, other nodes can take predefined actions to take over and bring up services elsewhere in the cluster.

How VCS detects failure

VCS detects failure of an application by issuing specific commands, tests, or scripts to monitor the overall health of an application. VCS also determines the health of underlying resources by supporting the applications such as file systems and network interfaces.

VCS uses a redundant network heartbeat to differentiate between the loss of a system and the loss of communication between systems. VCS can also use SCSI3-based membership coordination and data protection for detecting failure on a node and on fencing.

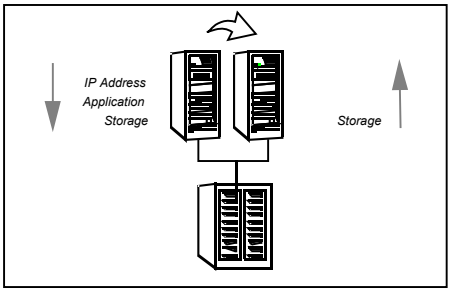
See "About cluster control, communications, and membership" on page 42.

How VCS ensures application availability

When VCS detects an application or node failure, VCS brings application services up on a different node in a cluster.

Figure 1-1 shows how VCS virtualizes IP addresses and system names, so client systems continue to access the application and are unaware of which server they use.

Figure 1-1 VCS virtualizes of IP addresses and system names to ensure application availability



For example, in a two-node cluster consisting of db-server1 and db-server2, a virtual address may be called db-server. Clients access db-server and are unaware of which physical server hosts the db-server.

About switchover and failover

Switchover and failover are the processes of bringing up application services on a different node in a cluster by VCS. The difference between the two processes is as follows:

Switchover	A switchover is an orderly shutdown of an application and its supporting resources on one server and a controlled startup on another server.
Failover	A failover is similar to a switchover, except the ordered shutdown of applications on the original node may not be possible due to failure of hardware or services, so the services are started on another node.

About cluster control guidelines

Most applications can be placed under cluster control provided the following guidelines are met:

- Defined start, stop, and monitor procedures
  See " Defined start, stop, and monitor procedures" on page 27.
- Ability to restart in a known state
  See " Ability to restart the application in a known state" on page 28.
- Ability to store required data on shared disks
  See " External data storage" on page 28.
- Adherence to license requirements and host name dependencies
  See " Licensing and host name issues" on page 29.

Defined start, stop, and monitor procedures

The following table describes the defined procedures for starting, stopping, and monitoring the application to be clustered:

Start procedure	The application must have a command to start it and all resources it may require. VCS brings up the required resources in a specific order, then brings up the application by using the defined start procedure. For example, to start an Oracle database, VCS must know which Oracle utility to call, such as sqlplus. VCS must also know the Oracle user, instance ID, Oracle home directory, and the pfile.
Stop procedure	An individual instance of the application must be capable of being stopped without affecting other instances. For example, You cannot kill all **httpd** processes on a Web server because it also stops other Web servers. If VCS cannot stop an application cleanly, it may call for a more forceful method, like a kill signal. After a forced stop, a clean-up procedure may be required for various process-specific and application-specific items that may be left behind. These items include shared memory segments or semaphores.

Monitor procedure

The application must have a monitor procedure that determines if the specified application instance is healthy. The application must allow individual monitoring of unique instances.

For example, the monitor procedure for a Web server connects to the specified server and verifies that it serves Web pages. In a database environment, the monitoring application can connect to the database server and perform SQL commands to verify read and write access to the database.

If a test closely matches what a user does, it is more successful in discovering problems. Balance the level of monitoring by ensuring that the application is up and by minimizing monitor overhead.

Ability to restart the application in a known state

When you take an application offline, the application must close out all tasks, store data properly on shared disk, and exit. Stateful servers must not keep that state of clients in memory. States should be written to shared storage to ensure proper failover.

Commercial databases such as Oracle, Sybase, or SQL Server are good examples of well-written, crash-tolerant applications. On any client SQL request, the client is responsible for holding the request until it receives acknowledgement from the server. When the server receives a request, it is placed in a special redo log file. The database confirms that the data is saved before it sends an acknowledgement to the client. After a server crashes, the database recovers to the last-known committed state by mounting the data tables and by applying the redo logs. This returns the database to the time of the crash. The client resubmits any outstanding client requests that are unacknowledged by the server, and all others are contained in the redo logs.

If an application cannot recover gracefully after a server crashes, it cannot run in a cluster environment. The takeover server cannot start up because of data corruption and other problems.

External data storage

The application must be capable of storing all required data and configuration information on shared disks. The exception to this rule is a true shared nothing cluster.

See “About shared nothing clusters” on page 60.

To meet this requirement, you may need specific setup options or soft links. For example, a product may only install in /usr/local. This limitation requires one of the

following options: linking /usr/local to a file system that is mounted from the shared storage device or mounting file system from the shared device on /usr/local.

The application must also store data to disk instead of maintaining it in memory. The takeover system must be capable of accessing all required information. This requirement precludes the use of anything inside a single system inaccessible by the peer. NVRAM accelerator boards and other disk caching mechanisms for performance are acceptable, but must be done on the external array and not on the local host.

Licensing and host name issues

The application must be capable of running on all servers that are designated as potential hosts. This requirement means strict adherence to license requirements and host name dependencies. A change of host names can lead to significant management issues when multiple systems have the same host name after an outage. To create custom scripts to modify a system host name on failover is not recommended. Veritas recommends that you configure applications and licenses to run properly on all hosts.

About the physical components of VCS

A VCS cluster comprises of systems that are connected with a dedicated communications infrastructure. VCS refers to a system that is part of a cluster as a node.

Each cluster has a unique cluster ID. Redundant cluster communication links connect systems in a cluster.

See “About VCS nodes” on page 29.

See “About shared storage” on page 30.

See “About networking” on page 30.

About VCS nodes

VCS nodes host the service groups (managed applications and their resources). Each system is connected to networking hardware, and usually to storage hardware also. The systems contain components to provide resilient management of the applications and to start and stop agents.

Nodes can be individual systems, or they can be created with domains or partitions on enterprise-class systems or on supported virtual machines. Individual cluster nodes each run their own operating system and possess their own boot device. Each node must run the same operating system within a single VCS cluster.

VCS supports clusters with up to 128 nodes. You can configure applications to run on specific nodes within the cluster.

About shared storage

Storage is a key resource of most applications services, and therefore most service groups. You can start a managed application on a system that has access to its associated data files. Therefore, a service group can only run on all systems in the cluster if the storage is shared across all systems. In many configurations, a storage area network (SAN) provides this requirement.

See " Cluster topologies and storage configurations" on page 58.

You can use I/O fencing technology for data protection. I/O fencing blocks access to shared storage from any system that is not a current and verified member of the cluster.

See "About the I/O fencing module" on page 45.

See "About the I/O fencing algorithm" on page 268.

About networking

Networking in the cluster is used for the following purposes:

- Communications between the cluster nodes and the customer systems.
- Communications between the cluster nodes.

See "About cluster control, communications, and membership" on page 42.

Logical components of VCS

VCS is comprised of several components that provide the infrastructure to cluster an application.

See "About resources and resource dependencies" on page 31.

See "Categories of resources" on page 33.

See "About resource types" on page 33.

See "About service groups" on page 34.

See "Types of service groups" on page 34.

See "About the ClusterService group" on page 35.

See "About the cluster UUID" on page 35.

See "About agents in VCS" on page 36.

See “About agent functions” on page 37.

See “ VCS agent framework” on page 42.

See “About cluster control, communications, and membership” on page 42.

See “About security services” on page 45.

See “ Components for administering VCS” on page 49.

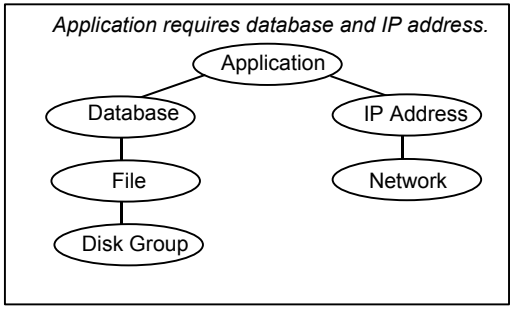
About resources and resource dependencies

Resources are hardware or software entities that make up the application. Disk groups and file systems, network interface cards (NIC), IP addresses, and applications are a few examples of resources.

Resource dependencies indicate resources that depend on each other because of application or operating system requirements. Resource dependencies are graphically depicted in a hierarchy, also called a tree, where the resources higher up (parent) depend on the resources lower down (child).

Figure 1-2 shows the hierarchy for a database application.

Figure 1-2 Sample resource dependency graph



Resource dependencies determine the order in which resources are brought online or taken offline. For example, you must import a disk group before volumes in the disk group start, and volumes must start before you mount file systems. Conversely, you must unmount file systems before volumes stop, and volumes must stop before you deport disk groups.

A parent is brought online after each child is brought online, and this continues up the tree, until finally the application starts. Conversely, to take a managed application offline, VCS stops resources by beginning at the top of the hierarchy. In this example, the application stops first, followed by the database application. Next the IP address and file systems stop concurrently. These resources do not have any resource dependency between them, and this continues down the tree.

Child resources must be brought online before parent resources are brought online. Parent resources must be taken offline before child resources are taken offline. If resources do not have parent-child interdependencies, they can be brought online or taken offline concurrently.

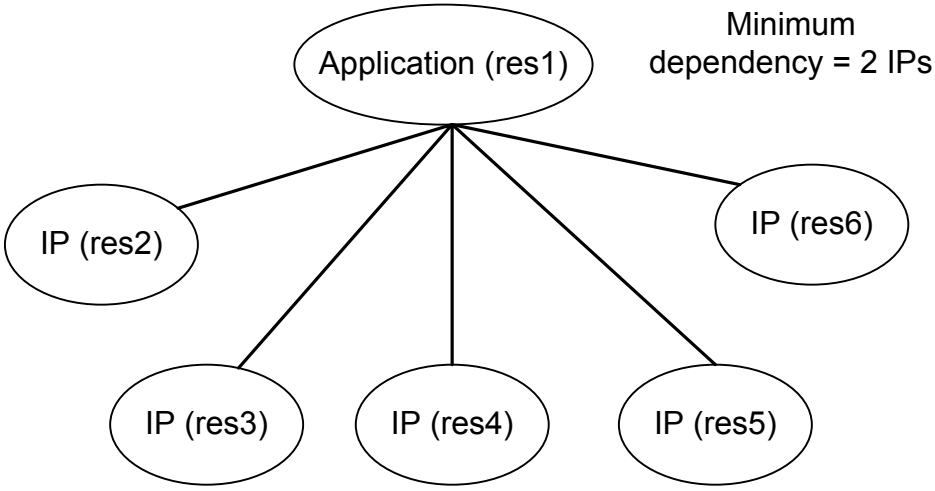
Atleast resource dependency

A new type of resource dependency has been introduced wherein a parent resource can depend on a set of child resources. The parent resource is brought online or remains online only if a minimum number of child resources in this resource set are online.

For example, if an application depends on five IPs and if this application has to be brought online or has to remain online, at least two IPs must be online. You can configure this dependency as shown in the figure below. See "Configuring atleast resource dependency" on page 160.

The system creates a set of child IP resources and the application resource will depend on this set. For this example, the assumption is that the application resource is res1 and the child IP resources are res2, res3, res4, res5, and res6.

Figure 1-3 Atleast resource dependency graph



If two or more IP resources come online, the application attempts to come online. If the number of online resources falls below the minimum requirement, (in this case: 2), resource fault is propagated up the resource dependency tree.

Note: Veritas InfoScale Operations Manager and Java GUI does not support atleast resource dependency and so the dependency is shown as normal resource dependency.

Categories of resources

Different types of resources require different levels of control.

Table 1-1 describes the three categories of VCS resources.

Table 1-1 Categories of VCS resources

VCS resources	VCS behavior
On-Off	VCS starts and stops On-Off resources as required. For example, VCS imports a disk group when required, and deports it when it is no longer needed.
On-Only	VCS starts On-Only resources, but does not stop them. For example, VCS requires NFS daemons to be running to export a file system. VCS starts the daemons if required, but does not stop them if the associated service group is taken offline.
Persistent	These resources cannot be brought online or taken offline. For example, a network interface card cannot be started or stopped, but it is required to configure an IP address. A Persistent resource has an operation value of None. VCS monitors Persistent resources to ensure their status and operation. Failure of a Persistent resource triggers a service group failover.

About resource types

VCS defines a resource type for each resource it manages. For example, you can configure the NIC resource type to manage network interface cards. Similarly, you can configure an IP address using the IP resource type.

VCS includes a set of predefined resources types. For each resource type, VCS has a corresponding agent, which provides the logic to control resources.

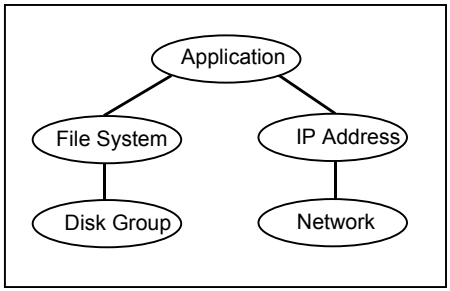
See “About agents in VCS” on page 36.

About service groups

A service group is a virtual container that contains all the hardware and software resources that are required to run the managed application. Service groups allow VCS to control all the hardware and software resources of the managed application as a single unit. When a failover occurs, resources do not fail over individually; the entire service group fails over. If more than one service group is on a system, a group can fail over without affecting the others.

Figure 1-4 shows a typical database service group.

Figure 1-4 Typical database service group



A single node can host any number of service groups, each providing a discrete service to networked clients. If the server crashes, all service groups on that node must be failed over elsewhere.

Service groups can be dependent on each other. For example, a managed application might be a finance application that is dependent on a database application. Because the managed application consists of all components that are required to provide the service, service group dependencies create more complex managed applications. When you use service group dependencies, the managed application is the entire dependency tree.

See "About service group dependencies" on page 407.

Types of service groups

VCS service groups fall in three main categories: failover, parallel, and hybrid.

About failover service groups

A failover service group runs on one system in the cluster at a time. Failover groups are used for most applications that do not support multiple systems to simultaneously access the application's data.

About parallel service groups

A parallel service group runs simultaneously on more than one system in the cluster. A parallel service group is more complex than a failover group. Parallel service groups are appropriate for applications that manage multiple application instances that run simultaneously without data corruption.

About hybrid service groups

A hybrid service group is for replicated data clusters and is a combination of the failover and parallel service groups. It behaves as a failover group within a system zone or site and a parallel group across system zones or site.

A hybrid service group cannot fail over across system zones. VCS allows a switch operation on a hybrid group only if both systems are within the same system zone or site. If no systems exist within a zone for failover, VCS calls the nofailover trigger on the lowest numbered node.

See "About service group dependencies" on page 407.

See "About the nofailover event trigger" on page 451.

About the ClusterService group

The ClusterService group is a special purpose service group, which contains resources that are required by VCS components.

The group contains resources for the following items:

- Notification
- Wide-area connector (WAC) process, which is used in global clusters

By default, the ClusterService group can fail over to any node despite restrictions such as the node being frozen. However, if you disable the AutoAddSystemToCSG attribute, you can control the nodes that are included in the SystemList. The ClusterService group is the first service group to come online and cannot be autodisabled. The ClusterService group comes online on the first node that transitions to the running state. The VCS engine discourages the action of taking the group offline manually.

About the cluster UUID

When you install VCS using the product installer, the installer generates a universally unique identifier (UUID) for the cluster. This value is the same across all the nodes in the cluster. The value is defined in the ClusterUUID or CID attribute.

See "Cluster attributes" on page 760.

If you do not use the product installer to install VCS, you must run the uuidconfig.pl utility to configure the UUID for the cluster.

See "Configuring and unconfiguring the cluster UUID value" on page 167.

About agents in VCS

Agents are multi-threaded processes that provide the logic to manage resources. VCS has one agent per resource type. The agent monitors all resources of that type; for example, a single IP agent manages all IP resources.

When the agent starts, it obtains the necessary configuration information from the VCS engine. It then periodically monitors the resources, and updates the VCS engine with the resource status. The agents that support Intelligent Monitoring Framework (IMF) also monitors the resources asynchronously. These agents register with the IMF notification module for resource state change notifications. Enabling IMF for process-based and mount-based agents can give you significant performance benefits in terms of system resource utilization and also aid faster failover of applications.

See "About resource monitoring" on page 38.

The action to bring a resource online or take it offline differs significantly for each resource type. For example, when you bring a disk group online, it requires importing the disk group. But, when you bring a database online, it requires that you start the database manager process and issue the appropriate startup commands.

VCS monitors resources when they are online and offline to ensure that they are not started on systems where they are not supposed to run. For this reason, VCS starts the agent for any resource that is configured to run on a system when the cluster is started. If no resources of a particular type are configured, the agent is not started. For example, if no Oracle resources exist in your configuration, the Oracle agent is not started on the system.

Certain agents can identify when an application has been intentionally shut down outside of VCS control. For agents that support this functionality, if an administrator intentionally shuts down an application outside of VCS control, VCS does not treat it as a fault. VCS sets the service group state as offline or partial, which depends on the state of other resources in the service group.

This feature allows administrators to stop applications that do not cause a failover. The feature is available for V51 agents. Agent versions are independent of VCS versions. For example, VCS 6.0 can run V40, V50, V51, and V52 agents for backward compatibility.

See "VCS behavior for resources that support the intentional offline functionality" on page 369.

About agent functions

Agents carry out specific functions on resources. The functions an agent performs are called entry points.

For details on agent functions, see the *Cluster Server Agent Developer's Guide*.

Table 1-2 describes the agent functions.

Table 1-2 Agent functions

Agent functions	Role
Online	Brings a specific resource ONLINE from an OFFLINE state.
Offline	Takes a resource from an ONLINE state to an OFFLINE state.
Monitor	Tests the status of a resource to determine if the resource is online or offline. The function runs at the following times: ■ During initial node startup, to probe and determine the status of all resources on the system. ■ After every online and offline operation. ■ Periodically, to verify that the resource remains in its correct state. Under normal circumstances, the monitor entry point is run every 60 seconds when a resource is online. The entry point is run every 300 seconds when a resource is expected to be offline. ■ When you probe a resource using the following command: `# hares -probe res_name -sys system_name`.
imf_init	Initializes the agent to interface with the IMF notification module. This function runs when the agent starts up.
imf_getnotification	Gets notification about resource state changes. This function runs after the agent initializes with the IMF notification module. This function continuously waits for notification and takes action on the resource upon notification.
imf_register	Registers or unregisters resource entities with the IMF notification module. For example, the function registers the PID for online monitoring of a process. This function runs for each resource after the resource goes into steady state (online or offline).

Table 1-2 Agent functions *(continued)*

Agent functions	Role
Clean	Cleans up after a resource fails to come online, fails to go offline, or fails to detect as ONLINE when resource is in an ONLINE state. The clean entry point is designed to clean up after an application fails. The function ensures that the host system is returned to a valid state. For example, the clean function may remove shared memory segments or IPC resources that are left behind by a database.
Action	Performs actions that can be completed in a short time and which are outside the scope of traditional activities such as online and offline. Some agents have predefined action scripts that you can run by invoking the action function.
Info	Retrieves specific information for an online resource. The retrieved information is stored in the resource attribute ResourceInfo. This function is invoked periodically by the agent framework when the resource type attribute InfoInterval is set to a non-zero value. The InfoInterval attribute indicates the period after which the info function must be invoked. For example, the Mount agent may use this function to indicate the space available on the file system. To see the updated information, you can invoke the info agent function explicitly from the command line interface by running the following command: `hares -refreshinfo res [-sys system] -clus cluster | -localclus`

About resource monitoring

VCS agents poll the resources periodically based on the monitor interval (in seconds) value that is defined in the MonitorInterval or in the OfflineMonitorInterval resource type attributes. After each monitor interval, VCS invokes the monitor agent function for that resource. For example, for process offline monitoring, the process agent's monitor agent function corresponding to each process resource scans the process table in each monitor interval to check whether the process has come online. For process online monitoring, the monitor agent function queries the operating system for the status of the process id that it is monitoring. In case of the mount agent, the monitor agent function corresponding to each mount resource checks if the block device is mounted on the mount point or not. In order to determine this, the monitor function does operations such as mount table scans or runs `statfs` equivalents.

With intelligent monitoring framework (IMF), VCS supports intelligent resource monitoring in addition to poll-based monitoring. IMF is an extension to the VCS

agent framework. You can enable or disable the intelligent monitoring functionality of the VCS agents that are IMF-aware. For a list of IMF-aware agents, see the *Cluster Server Bundled Agents Reference Guide*.

See "How intelligent resource monitoring works" on page 40.

See "Enabling and disabling intelligent resource monitoring for agents manually" on page 152.

See "Enabling and disabling IMF for agents by using script" on page 154.

Poll-based monitoring can consume a fairly large percentage of system resources such as CPU and memory on systems with a huge number of resources. This not only affects the performance of running applications, but also places a limit on how many resources an agent can monitor efficiently.

However, with IMF-based monitoring you can either eliminate poll-based monitoring completely or reduce its frequency. For example, for process offline and online monitoring, you can completely avoid the need for poll-based monitoring with IMF-based monitoring enabled for processes. Similarly for vxfs mounts, you can eliminate the poll-based monitoring with IMF monitoring enabled. Such reduction in monitor footprint will make more system resources available for other applications to consume.

Note: Intelligent Monitoring Framework for mounts is supported only for the VxFS, ext4, and XFS file system types.

With IMF-enabled agents, VCS will be able to effectively monitor larger number of resources.

Thus, intelligent monitoring has the following benefits over poll-based monitoring:

- Provides faster notification of resource state changes
- Reduces VCS system utilization due to reduced monitor function footprint
- Enables VCS to effectively monitor a large number of resources

Consider enabling IMF for an agent in the following cases:

- You have a large number of process resources or mount resources under VCS control.
- You have any of the agents that are IMF-aware.

For information about IMF-aware agents, see the following documentation:

- See the *Cluster Server Bundled Agents Reference Guide* for details on whether your bundled agent is IMF-aware.

- See the *Storage Foundation Cluster File System High Availability Installation Guide* for IMF-aware agents in CFS environments.

How intelligent resource monitoring works

When an IMF-aware agent starts up, the agent initializes with the IMF notification module. After the resource moves to a steady state, the agent registers the details that are required to monitor the resource with the IMF notification module. For example, the process agent registers the PIDs of the processes with the IMF notification module. The agent's imf_getnotification function waits for any resource state changes. When the IMF notification module notifies the imf_getnotification function about a resource state change, the agent framework runs the monitor agent function to ascertain the state of that resource. The agent notifies the state change to VCS which takes appropriate action.

A resource moves into a steady state when any two consecutive monitor agent functions report the state as ONLINE or as OFFLINE. The following are a few examples of how steady state is reached.

- When a resource is brought online, a monitor agent function is scheduled after the online agent function is complete. Assume that this monitor agent function reports the state as ONLINE. The next monitor agent function runs after a time interval specified by the MonitorInterval attribute. If this monitor agent function too reports the state as ONLINE, a steady state is achieved because two consecutive monitor agent functions reported the resource state as ONLINE. After the second monitor agent function reports the state as ONLINE, the registration command for IMF is scheduled. The resource is registered with the IMF notification module and the resource comes under IMF control.The default value of MonitorInterval is 60 seconds.

  A similar sequence of events applies for taking a resource offline.

- Assume that IMF is disabled for an agent type and you enable IMF for the agent type when the resource is ONLINE. The next monitor agent function occurs after a time interval specified by MonitorInterval. If this monitor agent function again reports the state as ONLINE, a steady state is achieved because two consecutive monitor agent functions reported the resource state as ONLINE.

  A similar sequence of events applies if the resource is OFFLINE initially and the next monitor agent function also reports the state as OFFLINE after you enable IMF for the agent type.

See “About the IMF notification module” on page 45.

About Open IMF

The Open IMF architecture builds further upon the IMF functionality by enabling you to get notifications about events that occur in user space.

The architecture uses an IMF daemon (IMFD) that collects notifications from the user space notification providers (USNPs) and passes the notifications to the AMF driver, which in turn passes these on to the appropriate agent. IMFD starts on the first registration with IMF by an agent that requires Open IMF.

The Open IMF architecture provides the following benefits:

- IMF can group events of different types under the same VCS resource and is the central notification provider for kernel space events and user space events.
- More agents can become IMF-aware by leveraging the notifications that are available only from user space.
- Agents can get notifications from IMF without having to interact with USNPs.

For example, Open IMF enables the AMF driver to get notifications from vxnotify, the notification provider for Veritas Volume Manager. The AMF driver passes these notifications on to the DiskGroup agent. For more information on the DiskGroup agent, see the *Cluster Server Bundled Agents Reference Guide*.

Agent classifications

The different kinds of agents that work with VCS include bundled agents, enterprise agents, and custom agents.

About bundled agents

Bundled agents are packaged with VCS. They include agents for Disk, Mount, IP, and various other resource types.

See the *Cluster Server Bundled Agents Reference Guide*.

About enterprise agents

Enterprise agents control third party applications. These include agents for Oracle, Sybase, and DB2.

See the following documentation for more information:

- *Cluster Server Agent for Oracle Installation and Configuration Guide*
- *Cluster Server Agent for Sybase Installation and Configuration Guide*
- *Cluster Server Agent for DB2 Installation and Configuration Guide*

About custom agents

Custom agents are agents that customers or Veritas consultants develop. Typically, agents are developed because the user requires control of an application that the current bundled or enterprise agents do not support.

See the *Cluster Server Agent Developer's Guide*.

VCS agent framework

The VCS agent framework is a set of common, predefined functions that are compiled into each agent. These functions include the ability to connect to the Veritas High Availability Engine (HAD) and to understand common configuration attributes. The agent framework frees the developer from developing functions for the cluster; the developer instead can focus on controlling a specific resource type.

VCS agent framework also includes IMF which enables asynchronous monitoring of resources and instantaneous state change notifications.

See "About resource monitoring" on page 38.

For more information on developing agents, see the *Cluster Server Agent Developer's Guide*.

About cluster control, communications, and membership

Cluster communications ensure that VCS is continuously aware of the status of each system's service groups and resources. They also enable VCS to recognize which systems are active members of the cluster, which have joined or left the cluster, and which have failed.

See "About the high availability daemon (HAD)" on page 42.

See "About the HostMonitor daemon" on page 43.

See "About Group Membership Services and Atomic Broadcast (GAB)" on page 44.

See "About Low Latency Transport (LLT)" on page 44.

See "About the I/O fencing module" on page 45.

See "About the IMF notification module" on page 45.

About the high availability daemon (HAD)

The VCS high availability daemon (HAD) runs on each system.

Also known as the Veritas High Availability Engine, HAD is responsible for the following functions:

- Builds the running cluster configuration from the configuration files
- Distributes the information when new nodes join the cluster
- Responds to operator input
- Takes corrective action when something fails.

The engine uses agents to monitor and manage resources. It collects information about resource states from the agents on the local system and forwards it to all cluster members.

The local engine also receives information from the other cluster members to update its view of the cluster. HAD operates as a replicated state machine (RSM). The engine that runs on each node has a completely synchronized view of the resource status on each node. Each instance of HAD follows the same code path for corrective action, as required.

The RSM is maintained through the use of a purpose-built communications package. The communications package consists of the protocols Low Latency Transport (LLT) and Group Membership Services and Atomic Broadcast (GAB).

See "About inter-system cluster communications" on page 236.

The hashadow process monitors HAD and restarts it when required.

About the HostMonitor daemon

VCS also starts HostMonitor daemon when the VCS engine comes up. The VCS engine creates a VCS resource VCShm of type HostMonitor and a VCShmg service group. The VCS engine does not add these objects to the main.cf file. Do not modify or delete these VCS components. VCS uses the HostMonitor daemon to monitor the resource utilization of CPU, Memory, and Swap. VCS reports to the engine log if the resources cross the threshold limits that are defined for the resources. The HostMonitor daemon also monitors the available capacity of CPU, memory, and Swap in absolute terms, enabling the VCS engine to make dynamic decisions about failing over an application to the biggest available or best suitable system.

VCS deletes user-defined VCS objects that use the HostMonitor object names. If you had defined the following objects in the main.cf file using the reserved words for the HostMonitor daemon, then VCS deletes these objects when the VCS engine starts:

- Any group that you defined as VCShmg along with all its resources.
- Any resource type that you defined as HostMonitor along with all the resources of such resource type.
- Any resource that you defined as VCShm.

See "VCS keywords and reserved words" on page 74.

You can control the behavior of the HostMonitor daemon using the Statistics attribute.

See "Cluster attributes" on page 760.

About Group Membership Services and Atomic Broadcast (GAB)

The Group Membership Services and Atomic Broadcast protocol (GAB) is responsible for the following cluster membership and cluster communications functions:

- Cluster Membership
  GAB maintains cluster membership by receiving input on the status of the heartbeat from each node by LLT. When a system no longer receives heartbeats from a peer, it marks the peer as DOWN and excludes the peer from the cluster. In VCS, memberships are sets of systems participating in the cluster.

  VCS has the following types of membership:

  - A regular membership includes systems that communicate with each other across more than one network channel.
  - A jeopardy membership includes systems that have only one private communication link.
  - A visible membership includes systems that have GAB running but the GAB client is no longer registered with GAB.
- Cluster Communications
  GAB's second function is reliable cluster communications. GAB provides guaranteed delivery of point-to-point and broadcast messages to all nodes. The Veritas High Availability Engine uses a private IOCTL (provided by GAB) to tell GAB that it is alive.

About Low Latency Transport (LLT)

VCS uses private network communications between cluster nodes for cluster maintenance. The Low Latency Transport functions as a high-performance, low-latency replacement for the IP stack, and is used for all cluster communications. Veritas recommends at least two independent networks between all cluster nodes.

For detailed information on the setting up networks, see *Cluster Server Configuration and Upgrade Guide*.

These networks provide the required redundancy in the communication path and enable VCS to differentiate between a network failure and a system failure.

LLT has the following two major functions:

- Traffic distribution
  LLT distributes (load balances) internode communication across all available private network links. This distribution means that all cluster communications are evenly distributed across all private network links (maximum eight) for

performance and fault resilience. If a link fails, traffic is redirected to the remaining links.

- Heartbeat
  LLT is responsible for sending and receiving heartbeat traffic over network links. The Group Membership Services function of GAB uses this heartbeat to determine cluster membership.

About the I/O fencing module

The I/O fencing module implements a quorum-type functionality to ensure that only one cluster survives a split of the private network. I/O fencing also provides the ability to perform SCSI-3 persistent reservations on failover. The shared disk groups offer complete protection against data corruption by nodes that are assumed to be excluded from cluster membership.

See "About the I/O fencing algorithm" on page 268.

About the IMF notification module

The notification module of Intelligent Monitoring Framework (IMF) is the Asynchronous Monitoring Framework (AMF).

AMF is a kernel driver which hooks into system calls and other kernel interfaces of the operating system to get notifications on various events such as:

- When a process starts or stops.
- When a block device gets mounted or unmounted from a mount point.

AMF also interacts with the Intelligent Monitoring Framework Daemon (IMFD) to get disk group related notifications. AMF relays these notifications to various VCS Agents that are enabled for intelligent monitoring.

See "About Open IMF" on page 40.

See "About resource monitoring" on page 38.

See "About cluster control, communications, and membership" on page 42.

About security services

InfoScale uses the VCS Authentication Service (VxAT) to provide secure communication between cluster nodes. VCS uses digital certificates for authentication and uses SSL to encrypt communication over the public network.

In the secure mode:

- VCS uses platform-based authentication.

- VCS does not store user passwords.
- VCS provides a single sign-on mechanism, so that the authenticated users do not need to sign in each time to connect to a cluster.
  All VCS users are system and domain users, and are configured using fully-qualified user names. For example, `administrator@vcsdomain`.

For secure communication, VCS components acquire credentials from the authentication broker that is configured on the local system. When a secure cluster is configured, a root and authentication broker is automatically deployed on each node. The acquired certificate is used during authentication, and is presented to clients for the SSL handshake.

VCS and its components specify the account name and the domain in the following format:

- **HAD Account**

  ```
  name = HAD
  domain = VCS_SERVICES@Cluster UUID
  ```

- **CmdServer**

  ```
  name = CMDSERVER
  domain = VCS_SERVICES@Cluster UUID
  ```

For instructions on how to set up Security Services while setting up the cluster, see the *Veritas InfoScale Installation Guide*.

See "Enabling and disabling secure mode for the cluster" on page 173.

Digital certification structure

The content in the digital certificates is organized as follows:

- User Information
  - Name
  - Domain Name
  - Domain Type
  - Groups to which the user belongs
    (This information is retrieved from the external domain.)
- Issuer
- Issuee
- Expires on

- Credential Info
  - Serial number of the credential
  - Whether this is a root credential
  - Whether this is a trusted credential
- Keys
  - Public Key

You can review this content to identify any issues that may occur with certificates. For example, if a certificate is expired, the entity to which it belongs may not be able to connect to the cluster. You can identify this issue by reviewing the contents of the certificate.

Sample certificate

```
Certificate:
  Data:
    Version: 3 (0x2)
      Serial Number: 0 (0x0)
    Signature Algorithm: sha256WithRSAEncryption
      Issuer: CN=vcsroot, \
      OU=root@101a489e-1580-11e9-b437-58e6671a649c, O=vx
      Validity
        Not Before: Jan 11 07:50:43 2019 GMT
        Not After : Jan  6 09:05:43 2039 GMT
      Subject: CN=vcsroot, \
      OU=root@101a489e-1580-11e9-b437-58e6671a649c, O=vx
      Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
          Public-Key: (2048 bit)
            Modulus:
              00:ce:1b:91:77:4e:41:16:a0:0f:3e:41:e3:b7:bd:
              c4:8a:4f:fb:5a:98:5a:b2:b9:f3:3c:c9:c6:cd:55:
              91:3e:c4:79:cc:d9:17:2f:06:ba:23:b8:98:99:92:
              22:90:9e:35:8f:bb:99:74:cd:79:27:58:36:54:14:
              79:5f:57:10:39:c0:dc:fb:98:7e:58:31:ed:6d:0d:
              2e:36:72:7a:93:55:b5:0e:25:c9:9c:0c:f2:39:a6:
              b9:0b:33:25:04:a9:97:cf:44:a6:d7:c5:5e:18:da:
              3f:ec:9b:eb:f2:e3:ee:7e:0d:3f:29:3f:80:b5:2b:
              58:23:e5:7c:ec:73:b6:24:86:70:64:54:74:9b:51:
              a3:f1:7a:52:61:e2:0c:61:e9:fd:a6:0b:c0:2f:45:
              87:95:cd:0a:d5:09:c6:b8:f9:4c:9d:ec:ed:3d:b1:
              af:5d:35:c6:0d:90:96:85:92:49:7c:e9:73:74:17:
```

```
            fb:35:e9:33:3c:3d:13:46:9b:03:cc:fd:ea:18:b5:
            70:f5:44:b2:e3:d9:9e:ba:4a:e7:e4:32:88:7e:0d:
            a2:c7:e2:cc:17:69:15:1f:53:cc:0c:79:6d:7e:75:
            71:74:1c:cb:32:87:0d:ad:f6:0b:8f:b5:bc:01:50:
            b2:7d:34:7b:dc:f9:b6:fd:29:91:11:41:4a:fc:75:
            03:a1
        Exponent: 65537 (0x10001)
X509v3 extensions:
    X509v3 Basic Constraints: critical
        CA:TRUE
        1.2.3.5:
            root
        1.2.3.6:
            00000017
        1.2.3.8:
```

Components for secure communication

SSL component

The VxAT server uses openssl 1.0.2w for SSL communication to provide enhanced secruity.

Third-party certificate for entity validation on SSL/TLS server

InfoScale supports using a third-party certificate for entity validation on the SSL/TLS server in VxAT on Linux hosts. With this support, you can generate a certificate for the SSL/TLS server by providing the encrypted passphrase to InfoScale. InfoScale then issues a certificate signing request, which is used to generate a certificate for the SSL/TLS server.

Note: Third-party certificates are not supported on other platforms.

Discontinuation of SSL/TLS Server support for TLSv1.0 and TLSv1.1

To reduce security vulnerabilities, the TLSv1.0 and TLSv1.1 protocols are not supported by default. However, you can enable these protocols by setting the value of the AllowTLSV1 attribute in `/var/VRTSvcs/vcsauth/data/VCSAUTHSERVER/root/.VRTSat/profile/VRTSatlocal.conf` to 1. Restart the`vcsauthserver` service for the changes to take effect.

Note: The discontinuation of this support is applicable only to a fresh installation of InfoScale, but not to upgrade scenarios.

Restriction after failed user login attempts

Access to VCS user accounts can be restricted after a predefined number of login attempts. InfoScale does not provide any specific policies or methods of its own to apply such restrictions.

You can configure the account lockout policy that is applicable to the directory service in your environment. Additionally, you may use the options provided by the authentication module. For details, refer to:

OpenLDAP password policies

Pluggable Authentication Modules (PAM) configuration options

Components for administering VCS

VCS provides several components to administer clusters.

Table 1-3 VCS components to administer clusters

VCS components	Description
Veritas InfoScale Operations Manager	A Web-based graphical user interface for monitoring and administering the cluster. Install the Veritas InfoScale Operations Manager on a management server outside the cluster to manage multiple clusters. See the Veritas InfoScale Operations Manager documentation for more information.
Cluster Manager (Java Console)	A cross-platform Java-based graphical user interface that provides complete administration capabilities for your cluster. The console runs on any system inside or outside the cluster, on any operating system that supports Java. You cannot use Java Console if the external communication port for VCS is not open. By default, the external communication port for VCS is 14141. **Note:** Veritas Technologies recommends Veritas InfoScale Operations Manager be used as the graphical user interface to monitor and administer the cluster. Cluster Manager (Java Console) may not support features released in VCS 6.0 and later versions.

Table 1-3 VCS components to administer clusters *(continued)*

VCS components	Description
VCS command line interface (CLI)	The VCS command-line interface provides a comprehensive set of commands for managing and administering the cluster. See "About administering VCS from the command line" on page 94.
Veritas High Availability Configuration wizard	In a physical or virtual environment, you can use the Veritas High Availability Configuration wizard to configure monitoring for generic applications. See the *Cluster Server Generic Application Agent Configuration Guide* for more information. In a VMware virtual environment, you can use the Veritas High Availability Configuration wizard to configure monitoring for the following applications: ■ Oracle ■ SAP WebAS ■ WebSphere MQ See the corresponding agent guide for more information.

About Veritas InfoScale Operations Manager

Veritas InfoScale Operations Manager provides a centralized management console for Veritas InfoScale products. You can use Veritas InfoScale Operations Manager to monitor, visualize, and manage storage resources and generate reports.

Veritas recommends using Veritas InfoScale Operations Manager to manage Storage Foundation and Cluster Server environments.

You can download Veritas InfoScale Operations Manager from:

https://www.veritas.com/content/support/en_US/downloads

Refer to the Veritas InfoScale Operations Manager documentation for installation, upgrade, and configuration instructions.

If you want to manage a single cluster using Cluster Manager (Java Console), a version is available for download from:

https://www.veritas.com/form/trialware/vcs-utilities

You cannot manage the new features of this release using the Java Console. Cluster Server Management Console is deprecated.

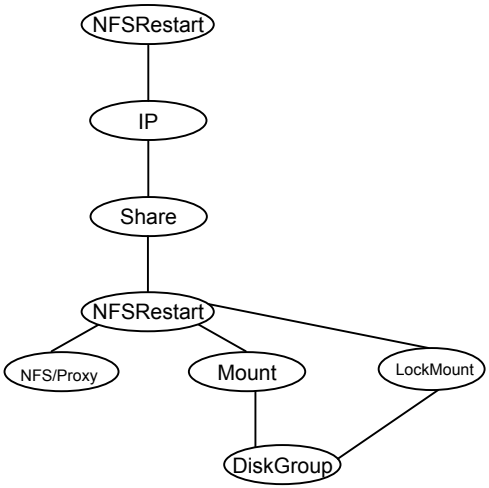
Putting the pieces together

In this example, a two-node cluster exports an NFS file system to clients. Both nodes are connected to shared storage, which enables them to access the directories being shared. A single service group, NFS_Group, fails over between System A and System B, as necessary.

The Veritas High Availability Engine (HAD) reads the configuration file, determines what agents are required to control the resources in the service group, and starts the agents. HAD uses resource dependencies to determine the order in which to bring the resources online. VCS issues online commands to the corresponding agents in the correct order.

Figure 1-5 shows a dependency graph for the sample NFS group.

Figure 1-5



VCS starts the agents for DiskGroup, Mount, Share, NFS, NIC, IP, and NFSRestart on all systems that are configured to run NFS_Group.

The resource dependencies are configured as follows:

- The /home file system (configured as a Mount resource), requires that the disk group (configured as a DiskGroup resource) is online before you mount.
- The lower NFSRestart resource requires that the file system is mounted and that the NFS daemons (NFS) are running.
- The NFS export of the home file system (Share) requires that the lower NFSRestart resource is up.

- The high availability IP address, nfs_IP, requires that the file system (Share) is shared and that the network interface (NIC) is up.
- The upper NFSRestart resource requires that the IP address is up.
- The NFS daemons and the disk group have no child dependencies, so they can start in parallel.
- The NIC resource is a persistent resource and does not require starting.

You can configure the service group to start automatically on either node in the preceding example. It then can move or fail over to the second node on command or automatically if the first node fails. On failover or relocation, to make the resources offline on the first node, VCS begins at the top of the graph. When it starts them on the second node, it begins at the bottom.

Chapter 2

About cluster topologies

This chapter includes the following topics:

- Basic failover configurations
- About advanced failover configurations
- Cluster topologies and storage configurations

Basic failover configurations

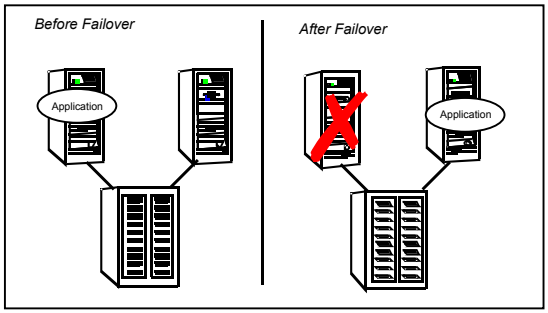
The basic failover configurations include asymmetric, symmetric, and N-to-1.

Asymmetric or active / passive configuration

In an asymmetric configuration, an application runs on a primary server. A dedicated redundant server is present to take over on any failure. The redundant server is not configured to perform any other functions.

Figure 2-1 shows failover within an asymmetric cluster configuration, where a database application is moved, or failed over, from the master to the redundant server.

Figure 2-1 Asymmetric failover



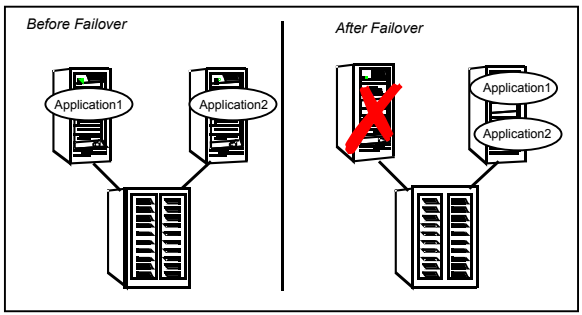
This configuration is the simplest and most reliable. The redundant server is on stand-by with full performance capability. If other applications are running, they present no compatibility issues.

Symmetric or active / active configuration

In a symmetric configuration, each server is configured to run a specific application or service and provide redundancy for its peer. In this example, each server runs one application service group. When a failure occurs, the surviving server hosts both application groups.

Figure 2-2 shows failover within a symmetric cluster configuration.

Figure 2-2 Symmetric failover



Symmetric configurations appear more efficient in terms of hardware utilization. In the asymmetric example, the redundant server requires only as much processor power as its peer. On failover, performance remains the same. In the symmetric example, the redundant server requires adequate processor power to run the existing application and the new application it takes over.

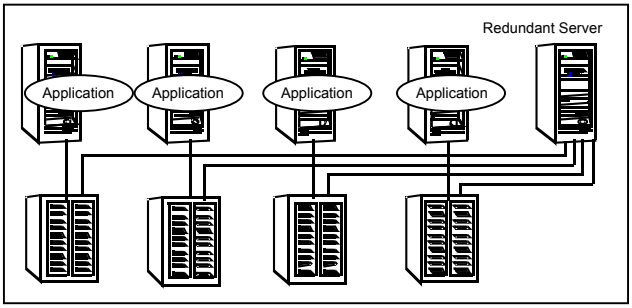
Further issues can arise in symmetric configurations when multiple applications that run on the same system do not co-exist properly. Some applications work well with multiple copies started on the same system, but others fail. Issues also can arise when two applications with different I/O and memory requirements run on the same system.

About N-to-1 configuration

An N-to-1 failover configuration reduces the cost of hardware redundancy and still provides a potential, dedicated spare. In an asymmetric configuration no performance penalty exists. No issues exist with multiple applications running on the same system; however, the drawback is the 100 percent redundancy cost at the server level.

Figure 2-3 shows an N to 1 failover configuration.

Figure 2-3 N-to-1 configuration

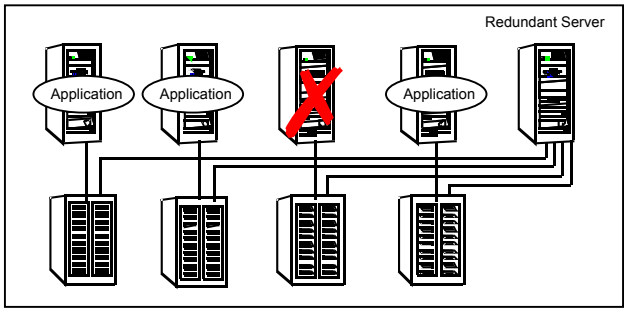


An N-to-1 configuration is based on the concept that multiple, simultaneous server failures are unlikely; therefore, a single redundant server can protect multiple active servers. When a server fails, its applications move to the redundant server. For example, in a 4-to-1 configuration, one server can protect four servers. This configuration reduces redundancy cost at the server level from 100 percent to 25 percent. In this configuration, a dedicated, redundant server is cabled to all storage and acts as a spare when a failure occurs.

The problem with this design is the issue of failback. When the failed server is repaired, you must manually fail back all services that are hosted on the failover server to the original server. The failback action frees the spare server and restores redundancy to the cluster.

Figure 2-4 shows an N to 1 failover requiring failback.

Figure 2-4 N-to-1 failover requiring failback



Most shortcomings of early N-to-1 cluster configurations are caused by the limitations of storage architecture. Typically, it is impossible to connect more than two hosts to a storage array without complex cabling schemes and their inherent reliability problems, or expensive arrays with multiple controller ports.

About advanced failover configurations

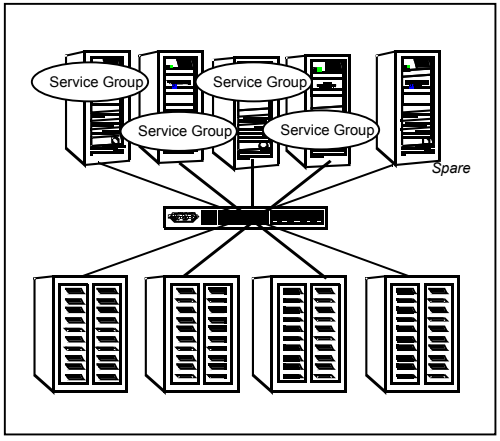
Advanced failover configuration for VCS include N + 1 and N-to-N configurations.

About the N + 1 configuration

With the capabilities introduced by storage area networks (SANs), you cannot only create larger clusters, you can also connect multiple servers to the same storage.

Figure 2-5 shows an N+1 cluster failover configuration.

Figure 2-5 N+1 configuration

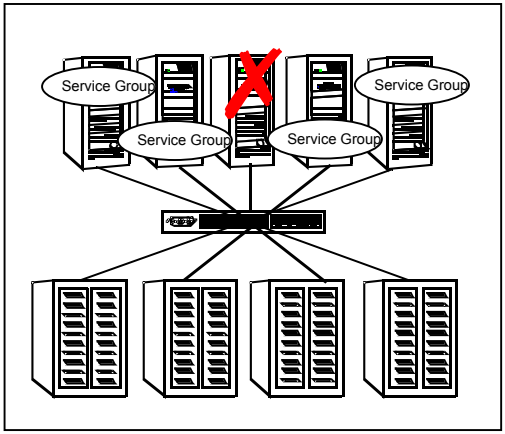


A dedicated, redundant server is no longer required in the configuration. Instead of N-to-1 configurations, you can use an N+1 configuration. In advanced N+1 configurations, an extra server in the cluster is spare capacity only.

When a server fails, the application service group restarts on the spare. After the server is repaired, it becomes the spare. This configuration eliminates the need for a second application failure to fail back the service group to the primary system. Any server can provide redundancy to any other server.

Figure 2-6 shows an N+1 cluster failover configuration requiring failback.

Figure 2-6 N+1 cluster failover configuration requiring failback

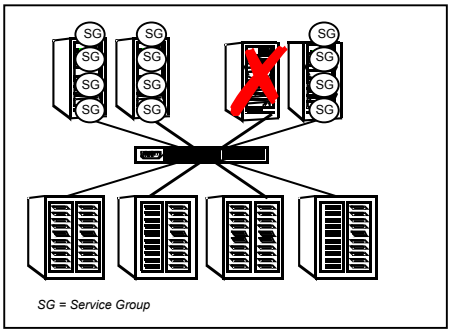


About the N-to-N configuration

An N-to-N configuration refers to multiple service groups that run on multiple servers, with each service group capable of being failed over to different servers. For example, consider a four-node cluster in which each node supports three critical database instances.

Figure 2-7 shows an N to N cluster failover configuration.

Figure 2-7 N-to-N configuration



If any node fails, each instance is started on a different node. this action ensures that no single node becomes overloaded. This configuration is a logical evolution of N + 1; it provides cluster standby capacity instead of a standby server.

N-to-N configurations require careful testing to ensure that all applications are compatible. You must specify a list of systems on which a service group is allowed to run in the event of a failure.

Cluster topologies and storage configurations

The commonly-used cluster topologies include the following:

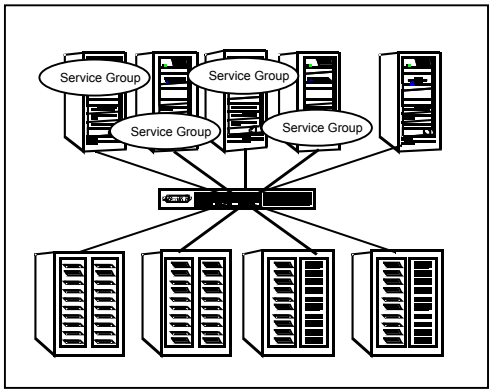
- Shared storage clusters
- Campus clusters
- Shared nothing clusters
- Replicated data clusters
- Global clusters

About basic shared storage cluster

In this configuration, a single cluster shares access to a storage device, typically over a SAN. You can only start an application on a node with access to the required storage. For example, in a multi-node cluster, any node that is designated to run a specific database instance must have access to the storage where the database's tablespaces, redo logs, and control files are stored. Such a shared disk architecture is also the easiest to implement and maintain. When a node or application fails, all data that is required to restart the application on another node is stored on the shared disk.

Figure 2-8 shows a shared disk architecture for a basic cluster.

Figure 2-8 Shared disk architecture for basic cluster

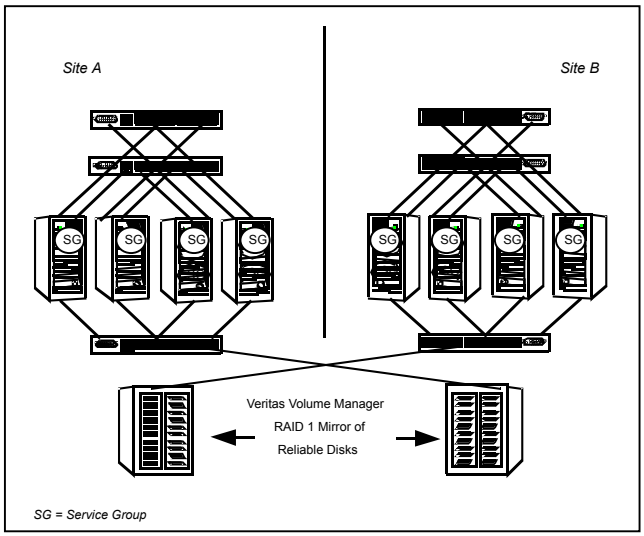


About campus, or metropolitan, shared storage cluster

In a campus environment, you use VCS and Veritas Volume Manager to create a cluster that spans multiple datacenters or buildings. Instead of a single storage array, data is mirrored between arrays by using Veritas Volume Manager. This configuration provides synchronized copies of data at both sites. This procedure is identical to mirroring between two arrays in a datacenter; only now it is spread over a distance.

Figure 2-9 shows a campus shared storage cluster.

Figure 2-9 Campus shared storage cluster



A campus cluster requires two independent network links for heartbeat, two storage arrays each providing highly available disks, and public network connectivity between buildings on same IP subnet. If the campus cluster setup resides on different subnets with one for each site, then use the VCS DNS agent to handle the network changes or issue the DNS changes manually.

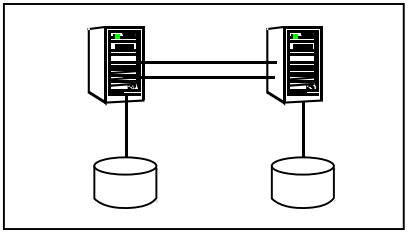
See " How VCS campus clusters work" on page 561.

About shared nothing clusters

Systems in shared nothing clusters do not share access to disks; they maintain separate copies of data. VCS shared nothing clusters typically have read-only data stored locally on both systems. For example, a pair of systems in a cluster that includes a critical Web server, which provides access to a backend database. The Web server runs on local disks and does not require data sharing at the Web server level.

Figure 2-10 shows a shared nothing cluster.

Figure 2-10 Shared nothing cluster

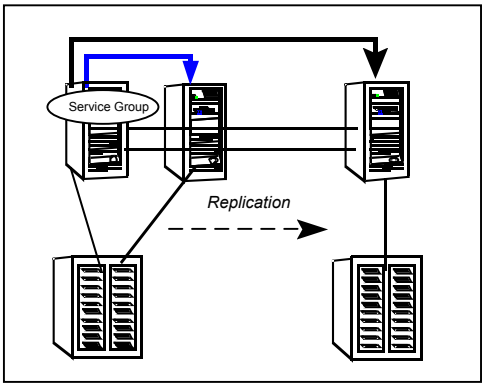


About replicated data clusters

In a replicated data cluster no shared disks exist. Instead, a data replication product synchronizes copies of data between nodes or sites. Replication can take place at the application, host, and storage levels. Application-level replication products, such as Oracle DataGuard, maintain consistent copies of data between systems at the SQL or database levels. Host-based replication products, such as Veritas Volume Replicator, maintain consistent storage at the logical volume level. Storage-based or array-based replication maintains consistent copies of data at the disk or RAID LUN level.

Figure 2-11 shows a hybrid shared storage and replicated data cluster, in which different failover priorities are assigned to nodes according to particular service groups.

Figure 2-11 Shared storage replicated data cluster



You can also configure replicated data clusters without the ability to fail over locally, but this configuration is not recommended.

See " How VCS replicated data clusters work" on page 552.

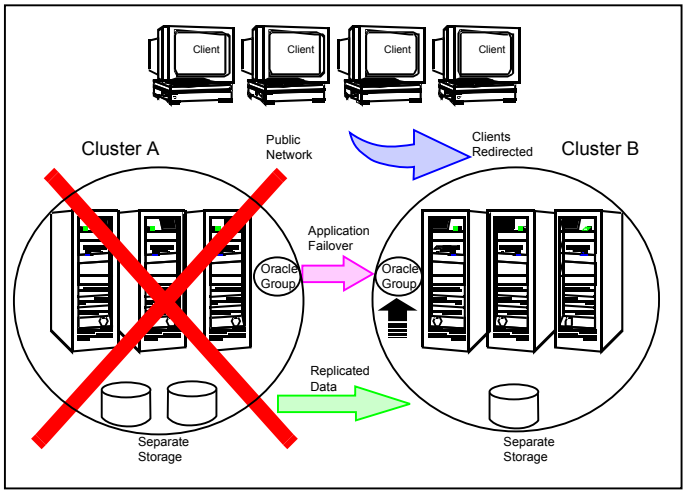
About global clusters

A global cluster links clusters at separate locations and enables wide-area failover and disaster recovery.

Local clustering provides local failover for each site or building. Campus and replicated cluster configurations offer protection against disasters that affect limited geographic regions. Large scale disasters such as major floods, hurricanes, and earthquakes can cause outages for an entire city or region. In such situations, you can ensure data availability by migrating applications to sites located considerable distances apart.

Figure 2-12 shows a global cluster configuration.

Figure 2-12 Global cluster



In a global cluster, if an application or a system fails, the application is migrated to another system within the same cluster. If the entire cluster fails, the application is migrated to a system in another cluster. Clustering on a global level also requires the replication of shared data to the remote site.

Chapter 3

VCS configuration concepts

This chapter includes the following topics:

- About configuring VCS
- VCS configuration language
- About the main.cf file
- About the types.cf file
- About VCS attributes
- VCS keywords and reserved words
- About managing VCS modules
- Disabling CmdServer
- VCS environment variables

About configuring VCS

When you configure VCS, you convey to the Veritas High Availability Engine the definitions of the cluster, service groups, resources, and dependencies among service groups and resources.

VCS uses the following two configuration files in a default configuration:

- main.cf
  Defines the cluster, including services groups and resources.
- types.cf
  Defines the resource types.

By default, both files reside in the following directory:

/etc/VRTSvcs/conf/config

Additional files that are similar to types.cf may be present if you enabled agents. OracleTypes.cf, by default, is located at /etc/VRTSagents/ha/conf/Oracle/.

In a VCS cluster, the first system to be brought online reads the configuration file and creates an internal (in-memory) representation of the configuration. Systems that are brought online after the first system derive their information from systems that are in the cluster.

You must stop the cluster if you need to modify the files manually. Changes made by editing the configuration files take effect when the cluster is restarted. The node where you made the changes should be the first node to be brought back online.

VCS configuration language

The VCS configuration language specifies the makeup of service groups and their associated entities, such as resource types, resources, and attributes. These specifications are expressed in configuration files, whose names contain the suffix .cf.

Several ways to generate configuration files are as follows:

- Use the Web-based Veritas InfoScale Operations Manager.
- Use Cluster Manager (Java Console).
- Use the command-line interface.
- If VCS is not running, use a text editor to create and modify the files.
- Use the VCS simulator on a Windows system to create the files.

About the main.cf file

The format of the main.cf file comprises include clauses and definitions for the cluster, systems, service groups, and resources. The main.cf file also includes service group and resource dependency clauses.

Table 3-1 describes some of the components of the main.cf file:

Table 3-1 Components of the main.cf file

Components of main.cf file	Description
Include clauses	Include clauses incorporate additional configuration files into main.cf. These additional files typically contain type definitions, including the types.cf file. Typically, custom agents add type definitions in their own files. `include "types.cf"` See "Including multiple .cf files in main.cf" on page 68.

Cluster definition	Defines the attributes of the cluster, the cluster name and the names of the cluster users.
	`cluster demo (`
	`UserNames = { admin = cDRpdxPmHzpS }`
	`)`
	See "Cluster attributes" on page 760.

System definition	Lists the systems designated as part of the cluster. The system names must match the name returned by the command `uname -a`.
	Each service group can be configured to run on a subset of systems defined in this section.
	`system Server1`
	`system Server2`
	See System attributes on page 746.

Service group definition	Service group definitions in main.cf comprise the attributes of a particular service group.
	`group NFS_group1 (`
	`SystemList = { Server1=0, Server2=1 }`
	`AutoStartList = { Server1 }`
	`)`
	See "Service group attributes" on page 719.
	See "About the SystemList attribute" on page 67.

Table 3-1 Components of the main.cf file *(continued)*

Components of main.cf file	Description

Resource definition	Defines each resource that is used in a particular service group. You can add resources in any order. The utility hacf arranges the resources alphabetically the first time the configuration file is run.
	`DiskGroup DG_shared1 (`
	`  DiskGroup = shared1`
	`  )`

Resource dependency clause	Defines a relationship between resources. A dependency is indicated by the keyword requires between two resource names.
	`IP_resource requires NIC_resource`
	In an atleast resource dependency, the parent resource depends on a set of child resources and a minimum number of resources from this resource set are required for the parent resource to be brought online or to remain online.
	`res1 requires atleast 2 from res2, res3,res4,res5,res6`
	The above dependency states that res1 can be brought online or can remain online only when 2 resources from the resource set (res2, res3, res4, res5, res6) are online. The minimum number of resources required by the parent resource should be greater than or equal to 1 and less than the total number of child resources in the resource set.
	See "About resources and resource dependencies" on page 31.

Service group dependency clause	To configure a service group dependency, place the keyword requires in the service group declaration of the main.cf file. Position the dependency clause before the resource dependency specifications and after the resource declarations.
	`requires group_x`
	`<dependency category>`
	`<dependency location>`
	`<dependency rigidity>`
	See "About service group dependencies" on page 407.

Note: Sample configurations for components of global clusters are listed separately.

See " VCS global clusters: The building blocks" on page 489.

About the SystemList attribute

The SystemList attribute designates all systems where a service group can come online. By default, the order of systems in the list defines the priority of systems that are used in a failover. For example, the following definition configures SystemA to be the first choice on failover, followed by SystemB, and then by SystemC.

```
SystemList = { SystemA, SystemB, SystemC }
```

You can assign system priority explicitly in the SystemList attribute by assigning numeric values to each system name. For example:

```
SystemList = { SystemA = 0, SystemB = 1, SystemC = 2 }
```

If you do not assign numeric priority values, VCS assigns a priority to the system without a number by adding 1 to the priority of the preceding system. For example, if the SystemList is defined as follows, VCS assigns the values SystemA = 0, SystemB = 2, SystemC = 3.

```
SystemList = { SystemA, SystemB = 2, SystemC }
```

Note that a duplicate numeric priority value may be assigned in some situations:

```
SystemList = { SystemA, SystemB=0, SystemC }
```

The numeric values assigned are SystemA = 0, SystemB = 0, SystemC = 1.

To avoid this situation, do not assign any numbers or assign different numbers to each system in SystemList.

Initial configuration

When VCS is installed, a basic main.cf configuration file is created with the cluster name, systems in the cluster, and a Cluster Manager user named *admin* with the password *password*.

The following is an example of the main.cf for cluster demo and systems SystemA and SystemB.

```
include "types.cf"
cluster demo (
UserNames = { admin = cDRpdxPmHzpS }
)
```

```
system SystemA (
)
system SystemB (
)
```

Including multiple .cf files in main.cf

You may choose include several configuration files in the main.cf file. For example:

```
include "applicationtypes.cf"
include "listofsystems.cf"
include "applicationgroup.cf"
```

If you include other .cf files in main.cf, the following considerations apply:

- Resource type definitions must appear before the definitions of any groups that use the resource types.
  In the following example, the applicationgroup.cf file includes the service group definition for an application. The service group includes resources whose resource types are defined in the file applicationtypes.cf. In this situation, the applicationtypes.cf file must appear first in the main.cf file.
  For example:

  ```
  include "applicationtypes.cf"
  include "applicationgroup.cf"
  ```

- If you define heartbeats outside of the main.cf file and include the heartbeat definition file, saving the main.cf file results in the heartbeat definitions getting added directly to the main.cf file.

About the types.cf file

The types.cf file describes standard resource types to the VCS engine; specifically, the data required to control a specific resource.

The types definition performs the following two important functions:

- Defines the type of values that may be set for each attribute.
  In the following DiskGroup example, the NumThreads and OnlineRetryLimit attributes are both classified as int, or integer. The DiskGroup, StartVolumes and StopVolumes attributes are defined as str, or strings.
  See “About attribute data types” on page 70.

- Defines the parameters that are passed to the VCS engine through the ArgList attribute. The line static str ArgList[] = { xxx, yyy, zzz } defines the order in which

parameters are passed to the agents for starting, stopping, and monitoring resources.

The following example illustrates a DiskGroup resource type definition for Linux:

```
type DiskGroup (
        static keylist SupportedActions = {
            "license.vfd", "disk.vfd", "udid.vfd",
            "verifyplex.vfd", campusplex, volinuse,
            checkudid, numdisks, joindg, splitdg,
            getvxvminfo }
        static int NumThreads = 1
        static int OnlineRetryLimit = 1
        static str ArgList[] = { DiskGroup,
            StartVolumes, StopVolumes, MonitorOnly,
            MonitorReservation, PanicSystemOnDGLoss,
            tempUseFence, DiskGroupType,
            UmountVolumes, Reservation }
        str DiskGroup
        boolean StartVolumes = 1
        boolean StopVolumes = 1
        boolean MonitorReservation = 0
        boolean PanicSystemOnDGLoss = 0
        temp str tempUseFence = INVALID
        str DiskGroupType = private
        int UmountVolumes
        str Reservation = ClusterDefault
)
```

For another example, review the following main.cf and types.cf files that represent an IP resource:

- The high-availability address is configured on the interface, which is defined by the Device attribute.
- The IP address is enclosed in double quotes because the string contains periods. See "About attribute data types" on page 70.
- The VCS engine passes the identical arguments to the IP agent for online, offline, clean, and monitor. It is up to the agent to use the arguments that it requires. All resource names must be unique in a VCS cluster.

main.cf for Linux:

```
IP nfs_ip1 (
    Device = eth0
    Address = "192.168.1.201"
```

```
    NetMask = "255.255.252.0"
)
```

types.cf for Linux:

```
type IP (
        static keylist RegList = { NetMask }
        static keylist SupportedActions = { "device.vfd", "route.vfd" }
        static str ArgList[] = { Device, Address, NetMask, PrefixLen,
          Options, IPOptions, IPRouteOptions }
        str Device
        str Address
        str NetMask
        int PrefixLen = 1000
        str Options
        str IPOptions
        str IPRouteOptions
)
```

About VCS attributes

VCS components are configured by using attributes. Attributes contain data about the cluster, systems, service groups, resources, resource types, agent, and heartbeats if you use global clusters. For example, the value of a service group's SystemList attribute specifies on which systems the group is configured and the priority of each system within the group. Each attribute has a definition and a value. Attributes also have default values assigned when a value is not specified.

About attribute data types

VCS supports the following data types for attributes:

String

A string is a sequence of characters that is enclosed by double quotes. A string can also contain double quotes, but the quotes must be immediately preceded by a backslash. A backslash is represented in a string as \\. Quotes are not required if a string begins with a letter, and contains only letters, numbers, dashes (-), and underscores (_).

For example, a string that defines a network interface such as eth0 does not require quotes since it contains only letters and numbers. However a string that defines an IP address contains periods and requires quotes- such as: "192.168.100.1".

Integer — Signed integer constants are a sequence of digits from 0 to 9. They may be preceded by a dash, and are interpreted in base 10. Integers cannot exceed the value of a 32-bit signed integer: 21471183247.

Boolean — A boolean is an integer, the possible values of which are 0 (false) and 1 (true).

About attribute dimensions

VCS attributes have the following dimensions:

Scalar — A scalar has only one value. This is the default dimension.

Vector — A vector is an ordered list of values. Each value is indexed by using a positive integer beginning with zero. Use a comma (,) or a semi-colon (;) to separate values. A set of brackets ([]) after the attribute name denotes that the dimension is a vector.

For example, an agent's ArgList is defined as:

```
static str ArgList[] = { RVG, DiskGroup }
```

Keylist — A keylist is an unordered list of strings, and each string is unique within the list. Use a comma (,) or a semi-colon (;) to separate values.

For example, to designate the list of systems on which a service group will be started with VCS (usually at system boot):

```
AutoStartList = {SystemA; SystemB; SystemC}
```

Association — An association is an unordered list of name-value pairs. Use a comma (,) or a semi-colon (;) to separate values.

A set of braces ({}) after the attribute name denotes that an attribute is an association.

For example, to associate the average time and timestamp values with an attribute:

```
str MonitorTimeStats{} = { Avg = "0", TS = "" }
```

About attributes and cluster objects

VCS has the following types of attributes, depending on the cluster object the attribute applies to:

Cluster attributes	Attributes that define the cluster. For example, ClusterName and ClusterAddress.
Service group attributes	Attributes that define a service group in the cluster. For example, Administrators and ClusterList.
System attributes	Attributes that define the system in the cluster. For example, Capacity and Limits.
Resource type attributes	Attributes that define the resource types in VCS. These resource type attributes can be further classified as: ■ Type-independent Attributes that all agents (or resource types) understand. Examples: RestartLimit and MonitorInterval; these can be set for any resource type. Typically, these attributes are set for all resources of a specific type. For example, setting MonitorInterval for the IP resource type affects all IP resources. ■ Type-dependent Attributes that apply to a particular resource type. These attributes appear in the type definition file (types.cf) for the agent. Example: The Address attribute applies only to the IP resource type. Attributes defined in the file types.cf apply to all resources of a particular resource type. Defining these attributes in the main.cf file overrides the values in the types.cf file for a specific resource. For example, if you set StartVolumes = 1 for the DiskGroup types.cf, it sets StartVolumes to True for all DiskGroup resources, by default. If you set the value in main.cf , it overrides the value on a per-resource basis. ■ Static These attributes apply for every resource of a particular type. These attributes are prefixed with the term static and are not included in the resource's argument list. You can override some static attributes and assign them resource-specific values. See "Overriding resource type static attributes" on page 163.
Resource attributes	Attributes that define a specific resource. Some of these attributes are type-independent. For example, you can configure the Critical attribute for any resource. Some resource attributes are type-dependent. For example, the Address attribute defines the IP address that is associated with the IP resource. These attributes are defined in the main.cf file.

Site attributes	Attributes that define a site. For example, Preference and SystemList.

Attribute scope across systems: global and local attributes

An attribute whose value applies to all systems is global in scope. An attribute whose value applies on a per-system basis is local in scope. The at operator (@) indicates the system to which a local value applies.

An example of local attributes can be found in the following resource type where IP addresses and routing options are assigned per machine.

```
MultiNICA mnic (
Device@sys1 = { eth0 = "166.98.16.103", eth1 = "166.98.16.103"
}
Device@sys2 = { eth0 = "166.98.16.104", eth2 = "166.98.16.104"
}
NetMask = "255.255.255.0"
RouteOptions@sys1 = "-net 192.100.201.0 192.100.13.7"
RouteOptions@sys2 = "-net 192.100.201.1 192.100.13.8"
)
```

About attribute life: temporary attributes

You can define temporary attributes in the types.cf file. The values of temporary attributes remain in memory as long as the VCS engine (HAD) is running. Values of temporary attributes are not available when HAD is restarted. These attribute values are not stored in the main.cf file.

You cannot convert temporary attributes to permanent attributes and vice-versa. When you save a configuration, VCS saves temporary attributes and their default values in the file types.cf.

The scope of these attributes can be local to a node or global across all nodes in the cluster. You can define local attributes even when the node is not part of a cluster.

You can define and modify these attributes only while VCS is running.

See “Adding, deleting, and modifying resource attributes” on page 148.

Size limitations for VCS objects

The following VCS objects are restricted to 1024 characters.

- Service group names

- Resource names
- Resource type names
- User names
- Attribute names

VCS passwords are restricted to 255 characters. You can enter a password of maximum 255 characters.

VCS keywords and reserved words

Following is a list of VCS keywords and reserved words. Note that they are case-sensitive.

action	firm	offline	set	system
after	global	online	Signaled	System
ArgListValues	group	MonitorOnly	site	temp
before	Group	Name	soft	type
boolean	hard	NameRule	start	Type
cluster	heartbeat	Path	Start	VCShm
Cluster	HostMonitor	Probed	state	VCShmg
condition	int	remote	State	
ConfidenceLevel	IState	remotecluster	static	
event	keylist	requires	stop	
false	local	resource	str	

About managing VCS modules

The initialization daemon of the operating system manages the start, stop, restart, and status of the AMF, GAB, LLT, VCS, and VxFEN modules. The service files and the script files for these modules are deployed at the appropriate location at the time of installation. Table 3-2 lists these files and their locations.

`systemd` is an initialization system that controls how services are started, stopped, or otherwise managed on RHEL 7 and SLES12 or later systems. The LSB files for the VCS stack have been replaced with the corresponding `systemd` unit service

files. In the older RHEL distributions, the init.d daemon managed the related services. In newer RHEL distributions, `systemd` manages them as unit service files.

The following VCS unit service files and startup scripts help provide `systemd` support for those services in Linux:

Table 3-2 Unit service files and script files for `systemd` support

Unit service file	Corresponding script file (*SourcePath*)
/usr/lib/systemd/system/amf.service	/opt/VRTSamf/bin/amf
/usr/lib/systemd/system/gab.service	/opt/VRTSgab/gab
/usr/lib/systemd/system/llt.service	/opt/VRTSllt/llt
/usr/lib/systemd/system/vcs.service	/opt/VRTSvcs/bin/vcs
/usr/lib/systemd/system/vxfen.service	/opt/VRTSvcs/vxfen/bin/vxfen

To start, stop, restart, or view the status of one of these services, use the following command:

```
systemctl [start | stop | restart | status] unitServiceFile
```

Note: The status option of the `systemctl` command displays only the status of the unit service file, like whether it is active, inactive, or failed.

To view the actual status information about the module or to view the HAD status, use:

```
serviceSourceScript status
```

For example:

```
/opt/VRTSvcs/bin/vcs status
```

To view the source path of any service, you can use the `systemctl` command as follows:

```
# systemctl show unitServiceFile -p SourcePath
```

For example:

```
# systemctl show vcs -p SourcePath
```

The `systemctl` command displays the source path as follows:

```
SourcePath=/opt/VRTSvcs/bin/vcs
```

Wherever `systemd` support is available, all the process in the VCS stack start in system.slice instead of user.slice.

Note: You can start or stop the `CmdServer` service independently of the VCS service by using the `systemctl` commands like you do for the other VCS modules.

Disabling CmdServer

By default, the `CmdServer` process runs as a daemon. It starts as soon as VCS starts, but InfoScale lets you disable the `CmdServer` daemon.

The STARTCMDSERVER environment variable lets you specify whether to disable `CmdServer`.

- The default value of STARTCMDSERVER is 1, which indicates that the `CmdServer` starts as a daemon process when HAD starts and then runs as a background process.
- If this value is set to 0, the VCS start script does not start `CmdServer` when the HAD starts.

This variable is present in the `/etc/sysconfig/vcs` file.

VCS environment variables

See " Defining VCS environment variables" on page 80.

See "Environment variables to start and stop VCS modules" on page 81.

Table 3-3 VCS environment variables

Environment Variable	Definition and Default Value
PERL5LIB	Root directory for Perl executables. (applicable only for Windows) Default: Install Drive:\Program Files\VERITAS\cluster server\lib\perl5.
VCS_CONF	Root directory for VCS configuration files. Default: /etc/VRTSvcs **Note:** You cannot modify this variable.
VCS_CONN_INIT_QUOTA	Maximum number of simultaneous connections accepted by the VCS engine (HAD) per client host that has not completed the handshake.

Table 3-3 VCS environment variables *(continued)*

Environment Variable	Definition and Default Value
VCS_CONN_HANDSHAKE_TIMEOUT	Timeout in seconds after which the VCS engine (HAD) closes the client connection that has not completed the handshake.
VCS_DEBUG_LOG_TAGS	Enables debug logs for the VCS engine, VCS agents, and HA commands. You must set VCS_DEBUG_LOG_TAGS before you start HAD or before you execute HA commands. See "Enabling debug logs for the VCS engine" on page 617. You can also export the variable from the `/opt/VRTSvcs/bin/vcsenv` file.
VCS_DOMAIN	The Security domain to which the VCS users belong. The VCS Authentication Service uses this environment variable to authenticate VCS users on a remote host. Default: Fully qualified host name of the remote host as defined in the VCS_HOST environment variable or in the .vcshost file.
VCS_DOMAINTYPE	The type of Security domain such as unixpwd, nt, nis, nisplus, ldap, or vx. The VCS Authentication Service uses this environment variable to authenticate VCS users on a remote host. Default: unixpwd
VCS_DIAG	Directory where VCS dumps HAD cores and FFDC data.
VCS_ENABLE_LDF	Designates whether or not log data files (LDFs) are generated. If set to 1, LDFs are generated. If set to 0, they are not.
VCS_HOME	Root directory for VCS executables. Default: /opt/VRTSvcs **Note:** You cannot modify this variable.
VCS_HOST	VCS node on which ha commands will be run.
VCS_GAB_PORT	GAB port to which VCS connects. Default: h

Table 3-3 VCS environment variables *(continued)*

Environment Variable	Definition and Default Value
VCS_GAB_TIMEOUT	Timeout in milliseconds for HAD to send heartbeats to GAB. Default: 30000 (denotes 30 seconds) Range: 30000 to 300000 (denotes 30 seconds to 300 seconds) If you set VCS_GAB_TIMEOUT to a value outside the range, the value is automatically reset to 30000 or 300000, depending on the proximity of the value to the lower limit or upper limit of the range. For example, the value is reset to 30000 if you specify 22000 and to 300000 if you specify 400000. Irrespective of the values set, VCS_GAB_TIMEOUT_SECS overrides VCS_GAB_TIMEOUT if both are specified. **Note:** If the specified timeout is exceeded, GAB kills HAD, and all active service groups on the system are disabled.
VCS_GAB_TIMEOUT_SECS	Timeout in seconds for HAD to send heartbeats to GAB under normal system load conditions. Default: 30 seconds Range: 30 seconds to 300 seconds If you set VCS_GAB_TIMEOUT_SECS to a value outside the range, the value is automatically reset to 30 or 300, depending on the proximity of the value to the lower limit or upper limit of the range. For example, the value is reset to 30 if you specify 22 and to 300 if you specify 400. Irrespective of the values set, VCS_GAB_TIMEOUT_SECS overrides VCS_GAB_TIMEOUT if both are specified. **Note:** If the specified timeout is exceeded, GAB kills HAD, and all active service groups on the system are disabled.

Table 3-3 VCS environment variables *(continued)*

Environment Variable	Definition and Default Value
VCS_GAB_PEAKLOAD_TIMEOUT_SECS	Timeout in seconds for HAD to send heartbeats to GAB under peak system load conditions. Default: 30 seconds Range: 30 seconds to 300 seconds To set the GAB tunables in adaptive mode, you must set VCS_GAB_PEAKLOAD_TIMEOUT_SECS to a value that exceeds VCS_GAB_TIMEOUT_SECS. If you set VCS_GAB_PEAKLOAD_TIMEOUT_SECS to a value that is lower than VCS_GAB_TIMEOUT_SECS, it is reset to VCS_GAB_TIMEOUT_SECS. **Note:** If the specified timeout is exceeded, GAB kills HAD, and all active service groups on the system are disabled.
VCS_GAB_RMTIMEOUT	Timeout in milliseconds for HAD to register with GAB. Default: 200000 (denotes 200 seconds) If you set VCS_GAB_RMTIMEOUT to a value less than 200000, the value is automatically reset to 200000. See "About GAB client registration monitoring" on page 584.
VCS_GAB_RMACTION	Controls the GAB behavior when VCS_GAB_RMTIMEOUT exceeds. You can set the value as follows: ■ panic—GAB panics the system ■ SYSLOG—GAB logs an appropriate message Default: SYSLOG See "About GAB client registration monitoring" on page 584.
VCS_HAD_RESTART_TIMEOUT	Set this variable to designate the amount of time the hashadow process waits (sleep time) before restarting HAD. Default: 0
VCS_LOG	Root directory for log files and temporary files. You must not set this variable to "" (empty string). Default: /var/VRTSvcs **Note:** If this variable is added or modified, you must reboot the system to apply the changes.

Table 3-3 VCS environment variables *(continued)*

Environment Variable	Definition and Default Value
VCS_SERVICE	Name of the configured VCS service. The VCS engine uses this variable to determine the external communication port for VCS. By default, the external communication port for VCS is 14141. The value for this environment variable is defined in the service file at the following location: /etc/services Default value: vcs If a new port number is not specified, the VCS engine starts with port 14141. To change the default port number, you must create a new entry in the service file at: /etc/services For example, if you want the external communication port for VCS to be set to 14555, then you must create the following entries in the services file: vcs 14555/tcp vcs 14555/udp **Note:** The cluster-level attribute OpenExternalCommunicationPort determines whether the port is open or not. See "Cluster attributes" on page 760.
VCS_TEMP_DIR	Directory in which temporary information required by, or generated by, hacf is stored. Default: /var/VRTSvcs This directory is created in the `tmp` directory under the following conditions: ■ The variable is not set. ■ The variable is set but the directory to which it is set does not exist. ■ The `hacf` utility cannot find the default location.

Defining VCS environment variables

Create the `/opt/VRTSvcs/bin/custom_vcsenv` file and define VCS environment variables in that file. These variables are set for VCS when the `hastart` command is run.

To set a variable, use the syntax appropriate for the shell in which VCS starts. Typically, VCS starts in `/bin/sh`. For example, define the variables as:

```
VCS_GAB_TIMEOUT = 35000;export VCS_GAB_TIMEOUT
```

Environment variables to start and stop VCS modules

The start and stop environment variables for AMF, LLT, GAB, VxFEN, and VCS engine define the default VCS behavior to start these modules during system restart or stop these modules during system shutdown.

Note: The startup and shutdown of AMF, LLT, GAB, VxFEN, and VCS engine are inter-dependent. For a clean startup or shutdown of VCS, you must either enable or disable the startup and shutdown modes for all these modules.

In a single-node cluster, you can disable the start and stop environment variables for LLT, GAB, and VxFEN if you have not configured these kernel modules.

Table 3-4 Start and stop environment variables for VCS

Environment variable	Definition and default value
AMF_START	Startup mode for the AMF driver. By default, the AMF driver is enabled to start up after a system reboot. This environment variable is defined in the following file: `/etc/sysconfig/amf` Default: 1
AMF_STOP	Shutdown mode for the AMF driver. By default, the AMF driver is enabled to stop during a system shutdown. This environment variable is defined in the following file: `/etc/sysconfig/amf` Default: 1
LLT_START	Startup mode for LLT. By default, LLT is enabled to start up after a system reboot. This environment variable is defined in the following file: `/etc/sysconfig/llt` Default: 1

Table 3-4 Start and stop environment variables for VCS *(continued)*

Environment variable	Definition and default value
LLT_STOP	Shutdown mode for LLT. By default, LLT is enabled to stop during a system shutdown. This environment variable is defined in the following file: `/etc/sysconfig/llt` Default: 1
GAB_START	Startup mode for GAB. By default, GAB is enabled to start up after a system reboot. This environment variable is defined in the following file: `/etc/sysconfig/gab` Default: 1
GAB_STOP	Shutdown mode for GAB. By default, GAB is enabled to stop during a system shutdown. This environment variable is defined in the following file: `/etc/sysconfig/gab` Default: 1
VXFEN_START	Startup mode for VxFEN. By default, VxFEN is enabled to start up after a system reboot. This environment variable is defined in the following file: `/etc/sysconfig/vxfen` Default: 1
VXFEN_STOP	Shutdown mode for VxFEN. By default, VxFEN is enabled to stop during a system shutdown. This environment variable is defined in the following file: `/etc/sysconfig/vxfen` Default: 1
VCS_START	Startup mode for VCS engine. By default, VCS engine is enabled to start up after a system reboot. This environment variable is defined in the following file: `/etc/sysconfig/vcs` Default: 1

Table 3-4 Start and stop environment variables for VCS *(continued)*

Environment variable	Definition and default value
VCS_STOP	Shutdown mode for VCS engine. By default, VCS engine is enabled to stop during a system shutdown. This environment variable is defined in the following file: `/etc/sysconfig/vcs` Default: 1

Table 3-4 Start and stop environment variables for VCS *(continued)*

Environment variable	Definition and default value
VCS_STOP_TIMEOUT	Time-out value in seconds for the stop operation of the VCS service. VCS uses this value during a system shutdown or restart operation to determine how long to wait for its stop operation to complete. After this duration has elapsed, VCS forcefully stops itself. If this value is set to 0 seconds, the stop operation does not time out. If an issue occurs when the resources are taken offline, HAD continues to be in the LEAVING state, thereby blocking the system shutdown or restart operation. Administrative intervention might be required to address such situations. Set this value to a positive integer to eliminate the need for manual intervention in case an operation is hung. After the duration specified that is in this variable has elapsed, VCS stops itself forcefully (`hastop -local -force`) and releases control of the application that is configured for HA. The operating system can then take the necessary action on the application components and continue with the shutdown or the restart operation. **Note:** If this value is set to anything other than a positive integer, VCS uses the default value (0, which indicates no time-out) instead. In `systemd` environments, the TimeoutStopSec attribute of the VCS service is set to 0 (infinity) to prevent the stop operation from timing out. This environment variable is defined in the `/etc/sysconfig/vcs` file. **Note:** Veritas recommends that you do not change the TimeoutStopSec attribute of the VCS service. If you want to configure a time-out value for the stop operation, use the VCS_STOP_TIMEOUT variable in the `/etc/sysconfig/vcs` file. **Warning:** Specifying a time-out value other than the default may have some adverse effects on the applications managed by VCS. For example, during a shutdown or a restart operation on a cluster node: Scenario 1, which may result in some unexpected behavior: If the value of VCS_STOP_TIMEOUT is too less, the VCS service stop operation times out before it can stop all the resources. This time-out may occur even when there is no issue with the cluster. Such an event may cause application-level issues in the cluster, because the application processes are no longer under the control of VCS. Scenario 2, which may result in some unexpected behavior: If a VCS agent fails to stop an application that it monitors, administrative intervention might be required. The VCS service stop operation times out and the necessary administrative intervention is not carried out. Default value: 0 seconds (indicates no time-out)

Section 2

Administration - Putting VCS to work

- Chapter 4. About the VCS user privilege model
- Chapter 5. Administering the cluster from the command line
- Chapter 6. Configuring applications and resources in VCS
- Chapter 7. Predicting VCS behavior using VCS Simulator

Chapter 4

About the VCS user privilege model

This chapter includes the following topics:

- About VCS user privileges and roles
- How administrators assign roles to users
- User privileges for OS user groups for clusters running in secure mode
- VCS privileges for users with multiple roles

About VCS user privileges and roles

Cluster operations are enabled or restricted depending on the privileges with which you log on. VCS has three privilege levels: Administrator, Operator, and Guest. VCS provides some predefined user roles; each role has specific privilege levels. For example, the role Guest has the fewest privileges and the role Cluster Administrator has the most privileges.

See "About administration matrices" on page 676.

VCS privilege levels

Table 4-1 describes the VCS privilege categories.

Table 4-1 VCS privileges

VCS privilege levels	Privilege description
Administrators	Can perform all operations, including configuration

Table 4-1 VCS privileges *(continued)*

VCS privilege levels	Privilege description
Operators	Can perform specific operations on a cluster or a service group.
Guests	Can view specified objects.

User roles in VCS

Table 4-2 lists the predefined VCS user roles, with a summary of their associated privileges.

Table 4-2 User role and privileges

User Role	Privileges
Cluster administrator	Cluster administrators are assigned full privileges. They can make configuration read-write, create and delete groups, set group dependencies, add and delete systems, and add, modify, and delete users. All group and resource operations are allowed. Users with Cluster administrator privileges can also change other users' privileges and passwords. To stop a cluster, cluster administrators require administrative privileges on the local system. **Note:** Cluster administrators can change their own and other users' passwords only after they change the configuration to read or write mode. Cluster administrators can create and delete resource types.
Cluster operator	Cluster operators can perform all cluster-level, group-level, and resource-level operations, and can modify the user's own password and bring service groups online. **Note:** Cluster operators can change their own passwords only if configuration is in read or write mode. Cluster administrators can change the configuration to the read or write mode. Users with this role can be assigned group administrator privileges for specific service groups.
Group administrator	Group administrators can perform all service group operations on specific groups, such as bring groups and resources online, take them offline, and create or delete resources. Additionally, users can establish resource dependencies and freeze or unfreeze service groups. Note that group administrators cannot create or delete service groups.

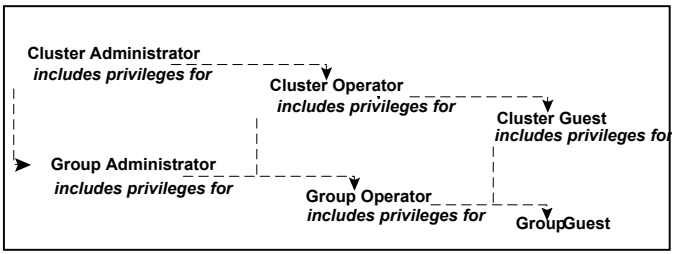
Table 4-2 User role and privileges *(continued)*

User Role	Privileges
Group operator	Group operators can bring service groups and resources online and take them offline. Users can also temporarily freeze or unfreeze service groups.
Cluster guest	Cluster guests have read-only access to the cluster, which means that they can view the configuration, but cannot change it. They can modify their own passwords only if the configuration is in read or write mode. They cannot add or update users. Additionally, users with this privilege can be assigned group administrator or group operator privileges for specific service groups. **Note:** By default, newly created users are assigned cluster guest permissions.
Group guest	Group guests have read-only access to the service group, which means that they can view the configuration, but cannot change it. The group guest role is available for clusters running in secure mode.

Hierarchy in VCS roles

Figure 4-1 shows the hierarchy in VCS and how the roles overlap with one another.

Figure 4-1 VCS roles



For example, cluster administrator includes privileges for group administrator, which includes privileges for group operator.

User privileges for CLI commands

Users logged with administrative or root privileges are granted privileges that exceed those of cluster administrator, such as the ability to start and stop a cluster.

If you do not have root privileges and the external communication port for VCS is not open, you cannot run CLI commands. If the port is open, VCS prompts for your VCS user name and password when you run *haxxx* commands.

You can use the `halogin` command to save the authentication information so that you do not have to enter your credentials every time you run a VCS command.

See "Logging on to VCS" on page 118.

See "Cluster attributes" on page 760.

User privileges for cross-cluster operations

A user or user group can perform a cross-cluster online or offline operation only if they have one of the following privileges on the remote cluster:

- Group administrator or group operator privileges for the group
- Cluster administrator or cluster operator privileges

A user or user group can perform a cross-cluster switch operation only if they have one of the following privileges on both the clusters:

- Group administrator or group operator privileges for the group
- Cluster administrator or cluster operator privileges

User privileges for clusters that run in secure mode

In secure mode, VCS assigns guest privileges to all native users.

When you assign privileges for clusters running in secure mode, you must specify fully-qualified user names, in the format username@domain.

When you assign privileges for clusters running in secure mode, you must specify user names in one of the following formats:

- To add a cluster-level user, specify only the user name.
- To add a node-level user, specify the user name in the username@FQDN format, where FQDN is the fully qualified domain name.

In a secure cluster, if the external communication port for VCS is not open, only root users logged on to the host system can run CLI commands.

See "About managing VCS users from the command line" on page 122.

See "Cluster attributes" on page 760.

You cannot assign or change passwords for users that use VCS when VCS runs in secure mode.

About the cluster-level user

You can add a cluster-level user by specifying the user name without the domain name. A cluster-level user can log in from any node in the cluster with the privilege that was assigned to the user. For example, if you add user1 to a cluster with four nodes (sys1, sys2, sys3, and sys4), user1 can access all four nodes with the privileges that were assigned at the cluster-level.

Adding users at the cluster-level offers the following advantages:

- Adding a cluster-level user requires only one entry per cluster. Adding a user at node level requires multiple entries, one for each node that the user is added to. To allow a user to operate from all four nodes in the cluster, you need four entries- user1@sys1, user1@sys2, user1@sys3, and user1@sys4, where sys1, sys2, sys3, and sys4 are fully qualified host names.
- Adding, updating, and deleting privileges at the cluster-level is easier and ensures uniformity of privileges for a user across nodes in a cluster.

How administrators assign roles to users

To assign a role to a user, an administrator performs the following tasks:

- Adds a user to the cluster, if the cluster is not running in secure mode.
- Assigns a role to the user.
- Assigns the user a set of objects appropriate for the role. For clusters that run in secure mode, you also can add a role to an operating system user group. See "User privileges for OS user groups for clusters running in secure mode" on page 90.

For example, an administrator may assign a user the group administrator role for specific service groups. Now, the user has privileges to perform operations on the specific service groups.

You can manage users and their privileges from the command line or from the graphical user interface.

See "About managing VCS users from the command line" on page 122.

User privileges for OS user groups for clusters running in secure mode

For clusters that run in secure mode, you can assign privileges to native users individually or at an operating system (OS) user group level.

For example, you may decide that all users that are part of the OS administrators group get administrative privileges to the cluster or to a specific service group. Assigning a VCS role to a user group assigns the same VCS privileges to all members of the user group, unless you specifically exclude individual users from those privileges.

When you add a user to an OS user group, the user inherits VCS privileges assigned to the user group.

Assigning VCS privileges to an OS user group involves adding the user group in one (or more) of the following attributes:

- AdministratorGroups—for a cluster or for a service group.
- OperatorGroups—for a cluster or for a service group.

For example, user Tom belongs to an OS user group: OSUserGroup1.

Table 4-3 shows how to assign VCS privileges. FQDN denotes the fully qualified domain name in these examples.

Table 4-3 To assign user privileges

To assign privileges	At an individual level, configure attribute	To the OS user group, configure attribute
Cluster administrator	cluster (Administrators = {tom@FQDN})	cluster (AdministratorGroups = {OSUserGroup1@FQDN})
Cluster operator	cluster (Operators = {tom@FQDN})	cluster (OperatorGroups = {OSUserGroup1@FQDN})
Cluster guest	Cluster (Guests = {tom@FQDN})	Not applicable
Group administrator	group *group_name* (Administrators = {tom@FQDN})	group *group_name* (AdministratorGroups = {OSUserGroup1@FQDN})
Group operator	group *group_name* (Operators = {tom@FQDN})	group *group_name* (OperatorGroups = {OSUserGroup1@FQDN})
Group guest	Cluster (Guests = {tom@FQDN})	Not applicable

VCS privileges for users with multiple roles

Table 4-4 describes how VCS assigns privileges to users with multiple roles. The scenarios describe user Tom who is part of two OS user groups: OSUserGroup1 and OSUserGroup2.

Table 4-4 VCS privileges for users with multiple roles

Situation and rule	Roles assigned in the VCS configuration	Privileges that VCS grants Tom
Situation: Multiple roles at an individual level. Rule: VCS grants highest privileges (or a union of all the privileges) to the user.	Tom: Cluster administrator Tom: Group operator	Cluster administrator.
Situation: Roles at an individual and OS user group level (secure clusters only). Rule: VCS gives precedence to the role granted at the individual level.	Tom: Group operator OSUserGroup1: Cluster administrator	Group operator
Situation: Different roles for different OS user groups (secure clusters only). Rule: VCS grants the highest privilege (or a union of all privileges of all user groups) to the user.	OSUserGroup1: Cluster administrators OSUserGroup2: Cluster operators	Cluster administrator
Situation: Roles at an individual and OS user group level (secure clusters only). Rule: VCS gives precedence to the role granted at the individual level. You can use this behavior to exclude specific users from inheriting VCS privileges assigned to their OS user groups.	OSUserGroup1: Cluster administrators OSUserGroup2: Cluster operators Tom: Group operator	Group operator

Chapter 5

Administering the cluster from the command line

This chapter includes the following topics:

- About administering VCS from the command line
- About installing a VCS license
- Administering LLT
- Administering the AMF kernel driver
- Starting VCS
- Stopping VCS
- Stopping VCS without evacuating service groups
- Stopping the VCS engine and related processes
- Logging on to VCS
- About managing VCS configuration files
- About managing VCS users from the command line
- About querying VCS
- About administering service groups
- Administering agents
- About administering resources
- About administering resource types

- Administering systems
- About administering clusters
- Using the -wait option in scripts that use VCS commands
- Running HA fire drills
- About administering simulated clusters from the command line

About administering VCS from the command line

Review the details on commonly used commands to administer VCS. For more information about specific commands or their options, see their usage information or the man pages associated with the commands.

You can enter most commands from any system in the cluster when VCS is running. The command to start VCS is typically initiated at system startup.

See "VCS command line reference" on page 684.

See "About administering I/O fencing" on page 285.

Symbols used in the VCS command syntax

The following table specifies the symbols used in the VCS commands. Do not use these symbols when you run the commands.

Table 5-1 Symbols used in the VCS commands

Symbols	Usage	Example
[]	Used for command options or arguments that are optional.	hasys -freeze [-persistent] [-evacuate] *system*
|	Used to specify that only one of the command options or arguments separated with | can be used at a time.	hagetcf [-s | -silent]
...	Used to specify that the argument can have several values.	hagrp -modify group *attribute* value … [-sys *system*]

Table 5-1 Symbols used in the VCS commands *(continued)*

Symbols	Usage	Example
{}	Used to specify that the command options or arguments enclosed within these braces must be kept together.	haatr -display {cluster | group | system | heartbeat | <restype>} haclus -modify *attribute* {*key value*}
<>	Used in the command help or usage output to specify that these variables must be replaced with the actual values.	haclus -help VCS INFO V-16-1-10601 Usage: haclus -add <cluster> <ip> haclus -delete <cluster>

How VCS identifies the local system

VCS checks the file `$VCS_CONF/conf/sysname`. If this file does not exist, the local system is identified by its node name. To view the system's node name, type:

```
# uname -n
```

The entries in this file must correspond to those in the files `/etc/llthosts` and `/etc/llttab`.

About specifying values preceded by a dash (-)

When you specify values in a command-line syntax, you must prefix values that begin with a dash (-) with a percentage sign (%). If a value begins with a percentage sign, you must prefix it with another percentage sign. (The initial percentage sign is stripped by the High Availability Daemon (HAD) and does not appear in the configuration file.)

About the -modify option

Most configuration changes are made by using the `-modify` options of the commands `haclus`, `hagrp`, `hares`, `hasys`, and `hatype`. Specifically, the `-modify` option of these commands changes the attribute values that are stored in the VCS configuration file. By default, all attributes are global, meaning that the value of the attribute is the same for all systems.

Note: VCS must be in read-write mode before you can change the configuration.

See "Setting the configuration to read or write" on page 121.

Encrypting VCS passwords

Use the `vcsencrypt` utility to encrypt passwords when you edit the VCS configuration file, `main.cf`, to add VCS users. The utility uses the standard AES-256 algorithm to encrypt the passwords.

Encrypting agent passwords by using security keys

Encrypting the agent password using security keys involves the following tasks:

- See "Generating a security key" on page 97.
- Encrypting the user passwords
- (Optional) See "Granting password encryption privileges to group administrators" on page 100.
- (Optional) See "Changing the security key" on page 100.

Encrypting the user passwords

Before you run the `vcsencrypt` command to add a user, you must run the following command:

```
# vcsencrypt -gensecinfo
```

Otherwise, the `vcsencrypt` command fails and logs an error, which states that the security key is not generated.

Note: Do not use the `vcsencrypt` utility when you enter passwords from the Java console.

Perform the following procedure on any cluster node.

To encrypt a password

1 Run the utility from the command line.

```
# vcsencrypt -vcs
```

2 The utility prompts you to enter the password twice. Enter the password and press return.

```
Enter Password:
 Enter Again:
```

The utility encrypts the entered password using the standard AES-256 algorithm and displays the encrypted password; for example:

```
0b:c3:0f:7f:42:52:c1:21:eb:7f:0d:59:28:56:e2:32:
ab:b7:f7:79:0a:4a:5a:34:be:c1:79:ca:27:d1:bc:58
```

3 The encrypted password and the IV value that is used for the encryption are stored as a key-value pair in the IV256List attribute in `main.cf`.

Encrypting agent passwords

To configure agents that require user passwords, you may need to edit the VCS configuration file, `main.cf`. To encrypt any such user passwords, use the `vcsencrypt` utility.

Note: Do not use the `vcsencrypt` utility when you enter passwords from the Java console.

Encrypting agent passwords by using security keys

Encrypting the agent password using security keys involves the following tasks:

- See "Generating a security key" on page 97.
- See "Encrypting the agent password" on page 98.
- (Optional) See "Granting password encryption privileges to group administrators" on page 100.
- (Optional) See "Changing the security key" on page 100.

Generating a security key

Use the `vcsencrypt` utility to generate a security key. This key is stored in the SecInfo256 cluster attribute.

Note: You must be a root user or must have administrative privileges to generate security keys.

To generate a security key

1. Log in on any one of the cluster nodes.
2. Make the VCS configuration writable.

   ```
   haconf -makerw
   ```

3. Run the `vcsencrypt` utility.

   ```
   vcsencrypt -gensecinfo
   ```

   The utility generates the security key and displays the following message:

   ```
   Generating SecInfo...please wait...
   SecInfo generated successfully.
   Trying to update its value in config file...
   SecInfo updated successfully.
   ```

4. Save the VCS configuration and make it read only.

   ```
   haconf -dump -makero
   ```

   Note: This step is crucial; if you do not perform this step, the VCS configuration cannot use the newly generated encryption key.

Encrypting the agent password

Perform the following procedure on any cluster node.

To encrypt the agent password

1. Run the encryption utility as follows:

   ```
   # vcsencrypt -agent -passwordValue
   ```

   If you do not provide the *passwordValue* argument, the utility prompts you for a password. Type the existing password and press return.

   The utility encrypts the entered password using the standard AES-256 algorithm and displays the encrypted password; for example:

   ```
   0b:c3:0f:7f:42:52:c1:21:eb:7f:0d:59:28:56:e2:32:ab:b7:f7:79:0a:4a:5a:34:be:c1:79:ca:27:d1:bc:58
   ```

   It also updates the cluster-level attribute IV256List, by adding a new entry that includes the encrypted password and the IV value that is used for the encryption as a key-value pair.

   For example:

   ```
   IV256List = {
   "fc:03:27:81:3b:71:47:83:38:33:02:bb:0c:d3:05:db:59:e1:52:44:a6:9b:9a:95:16:ce:c0:a4:94:d7:b0:80"
   =
   "e0:ca:04:b4:eb:4a:07:d3:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00",
   "af:d5:37:05:65:ca:fc:3e:db:13:72:f9:d0:4b:be:4f:3c:66:6e:58:b1:ec:0a:39:4d:74:85:fc:aa:3e:03:fd"
   =
   "83:a5:ad:54:07:d2:ba:61:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00"
   }
   ```

2. Modify the required agent attribute to assign the encrypted password.

   ```
   # hares -modify resourceName passwordAttributeName
   encryptedPasswordValue
   ```

To delete an entry from IV256List

1. Make the VCS configuration writable.

   ```
   # haconf -makerw
   ```

2. Run the following command:

   ```
   # vcsencrypt -delkey encryptedPasswordValue
   ```

   The *encryptedPasswordValue* is the key in the IV256List attribute.

3. Make the VCS configuration read-only.

   ```
   # haconf -makero
   ```

Granting password encryption privileges to group administrators

You may need to grant password encryption privileges to group administrators.

To grant password encryption privileges to group administrators

- Set the value of the cluster attribute SecInfoLevel to R+A:

```
haclus -modify SecInfoLevel R+A
```

To restrict password encryption privileges to superusers

- Set the value of the cluster attribute SecInfoLevel to R:

```
haclus -modify SecInfoLevel R
```

Changing the security key

If you change the security key, you must use the new security key to re-encrypt all the passwords that were encrypted earlier with the previous security key. Otherwise, the agent fails to decrypt the encrypted password, and may consequently fail to perform resource monitoring.

To change the security key

1. Save the VCS configuration and make it writeable.

```
haconf -makerw
```

2. Run the following command:

```
vcsencrypt -gensecinfo -force
```

3. Save the VCS configuration and make it read only.

```
haconf -dump -makero
```

About installing a VCS license

The Veritas InfoScale installer prompts you to select one of the following licensing methods:

- Install a license key for the product and features that you want to install. When you purchase a Veritas InfoScale product, you receive a License Key certificate. The certificate specifies the component keys and the type of license purchased.
- Continue to install without a license key. The installer prompts for the component options that you want to install, and then sets the required component level.

Within 60 days of choosing this option, you must install a valid license key corresponding to the license level entitled or continue with keyless licensing by managing the server or cluster with a management server.

Installing and updating license keys using vxlicinst

Use the `vxlicinstupgrade` command to install or update a valid product license key file for the product you have purchased.

You must have root privileges to use this utility. This utility must be run on each system in the cluster; the utility cannot install or update a license on remote nodes.

To install a new license

- Run the following command on each node in the cluster:

```
# cd /opt/VRTS/bin
./vxlicinstupgrade -k <key file path>
```

  Where, the *<key file path>* is the absolute path of the .slf license key file saved on the current node.

To update licensing information in a running cluster

1. Install the new component license on each node in the cluster using the vxlicinst utility.
2. Update system-level licensing information on all nodes in the cluster.

   You must run the `updatelic` command only after you add a license key using the `vxlicinst` command or after you set the component license level using the `vxkeyless` command.

```
# hasys -updatelic -all
```

   You must update licensing information on all nodes before proceeding to the next step.

3. Update cluster-level licensing information:

```
# haclus -updatelic
```

Setting or changing the product level for keyless licensing

Run the `vxkeyless` command to set the component level for the coponents you have purchased. This option also requires that you manage the server or cluster with a management server.

See the `vxkeyless`(1m) manual page.

To set or change the product level

1. View the current setting for the component level.

```
# vxkeyless [-v] display
```

2. View the possible settings for the component level.

```
# vxkeyless displayall
```

3. Set the desired component level.

```
# vxkeyless [-q] set prod_levels
```

Where *prod_levels* is a comma-separated list of keywords. Use the keywords returned by the `vxkeyless displayall` command.

If you want to remove keyless licensing and enter a key, you must clear the keyless licenses. Use the NONE keyword to clear all keys from the system.

Note that clearing the keys disables the components until you install a new key or set a new component level.

To clear the product license level

1. View the current setting for the component license level.

```
# vxkeyless [-v] display
```

2. If there are keyless licenses installed, remove all keyless licenses:

```
# vxkeyless [-q] set NONE
```

Administering LLT

You can use the LLT commands such as `lltdump` and `lltconfig` to administer the LLT links. See the corresponding LLT manual pages for more information on the commands.

See “About Low Latency Transport (LLT)” on page 44.

See “Displaying the cluster details and LLT version for LLT links” on page 103.

See “Adding and removing LLT links” on page 103.

See “Configuring aggregated interfaces under LLT” on page 105.

See “Configuring destination-based load balancing for LLT” on page 107.

See “Configuring heartbeat threads to improve cluster resiliency” on page 107.

Displaying the cluster details and LLT version for LLT links

You can use the `lltdump` command to display the LLT version for a specific LLT link. You can also display the cluster ID and node ID details.

See the `lltdump`(1M) manual page for more details.

To display the cluster details and LLT version for LLT links

◆ Run the following command to display the details:

```
# /opt/VRTSllt/lltdump -D -f link
```

For example, if eth2 is connected to sys1, then the command displays a list of all cluster IDs and node IDs present on the network link eth2.

```
# /opt/VRTSllt/lltdump -D -f eth2

lltdump : Configuration:

device : eth2

sap : 0xcafe
promisc sap : 0
promisc mac : 0
cidsnoop : 1
=== Listening for LLT packets ===
cid nid vmaj vmin
3456 1 5 0
3456 3 5 0
83 0 4 0
27 1 3 7
3456 2 5 0
```

Adding and removing LLT links

You can use the `lltconfig` command to add or remove LLT links when LLT is running.

See the `lltconfig`(1M) manual page for more details.

Note: When you add or remove LLT links, you need not shut down GAB or the high availability daemon, `had`. Your changes take effect immediately, but are lost on the next restart. For changes to persist, you must also update the `/etc/lltab` file.

To add LLT links

- Depending on the LLT link type, run the following command to add an LLT link:
  - For ether link type:

    ```
    # lltconfig -t devtag -d device
    [-b ether ] [-s SAP] [-m mtu]  [-I] [-Q]
    ```

  - For UDP link type:

    ```
    # lltconfig -t devtag -d device
    -b udp [-s port] [-m mtu]
    -I IPaddr -B bcast
    ```

  - For UDP6 link type:

    ```
    # lltconfig -t devtag -d device
    -b udp6 [-s port] [-m mtu]
    -I IPaddr [-B mcast]
    ```

  - For RDMA link type:

    ```
    # lltconfig -t devtag -d device
     -b rdma -s port [-m mtu]
     -I IPaddr -B bcast
    ```

  Where:

devtag	Tag to identify the link
device	Device name of the interface. For link type ether, you can specify the device name as an interface name. For example, eth0. Preferably, specify the device name as eth-macaddress. For example, eth- xx:xx:xx:xx:xx:xx. For link types udp and udp6, the device is the udp and udp6 device name respectively. For link type rdma, the device name is udp.
bcast	Broadcast address for the link type udp and rdma
mcast	Multicast address for the link type udp6
IPaddr	IP address for link types udp, udp6 and rdma
SAP	SAP to bind on the network links for link type ether

port	Port for link types udp, udp6 and rdma
mtu	Maximum transmission unit to send packets on network links

For example:

- For ether link type:

```
# lltconfig -t eth4 -d eth4 -s 0xcafe -m 1500
```

- For UDP link type:

```
# lltconfig -t link1 -d udp -b udp -s 50010
-I 192.168.1.1 -B 192.168.1.255
```

- For UDP6 link type:

```
# lltconfig -t link1 -d udp6 -b udp6 -s 50010 -I 2000::1
```

- For RDMA link:

```
# lltconfig -t link1 -d udp -b rdma -s 50010
 -I 192.168.1.1 -B 192.168.1.255
```

Note: If you want the addition of LLT links to be persistent after reboot, then you must edit the `/etc/lltab` with LLT entries.

To remove an LLT link

1. Run the following command to disable a network link that is configured under LLT.

```
# lltconfig -t devtag -L disable
```

2. Wait for the 16 seconds (LLT peerinact time).
3. Run the following command to remove the link.

```
# lltconfig -u devtag
```

Configuring aggregated interfaces under LLT

If you want to configure LLT to use aggregated interfaces after installing and configuring VCS, you can use one of the following approaches:

- Edit the /etc/llttab file

  This approach requires you to stop LLT. The aggregated interface configuration is persistent across reboots.

- Run the `lltconfig` command

  This approach lets you configure aggregated interfaces on the fly. However, the changes are not persistent across reboots.

To configure aggregated interfaces under LLT by editing the /etc/llttab file

1. If LLT is running, stop LLT after you stop the other dependent modules.

   ```
   # systemctl stop llt
   ```

   See "Stopping VCS" on page 113.

2. Add the following entry to the `/etc/llttab` file to configure an aggregated interface.

   ```
   link tag device_name systemid_range link_type sap mtu_size
   ```

tag	Tag to identify the link
device_name	Device name of the aggregated interface.
systemid_range	Range of systems for which the command is valid. If the link command is valid for all systems, specify a dash (-).
link_type	The link type must be ether.
sap	SAP to bind on the network links. Default is 0xcafe.
mtu_size	Maximum transmission unit to send packets on network links

3. Restart LLT for the changes to take effect. Restart the other dependent modules that you stopped in step 1.

   ```
   # systemctl start llt
   ```

   See "Starting VCS" on page 110.

To configure aggregated interfaces under LLT using the lltconfig command

- When LLT is running, use the following command to configure an aggregated interface:

```
lltconfig -t devtag -d device
[-b linktype ] [-s SAP] [-m mtu]
```

devtag	Tag to identify the link.
device	Device name of the aggregated interface.
link_type	The link type must be ether.
sap	SAP to bind on the network links. Default is 0xcafe.
mtu_size	Maximum transmission unit to send packets on network links.

See the `lltconfig`(1M) manual page for more details.

You need not reboot after you make this change. However, to make these changes persistent across reboot, you must update the `/etc/llttab` file.

See "To configure aggregated interfaces under LLT by editing the /etc/llttab file" on page 106.

Configuring destination-based load balancing for LLT

Destination-based load balancing for Low Latency Transport (LLT) is turned off by default. Veritas recommends destination-based load balancing when the cluster setup has more than two nodes and more active LLT ports.

See "About Low Latency Transport (LLT)" on page 238.

To configure destination-based load balancing for LLT

- Run the following command to configure destination-based load balancing:

```
lltconfig -F linkburst:0
```

Configuring heartbeat threads to improve cluster resiliency

If the operating system does not schedule LLT timer context on time, heartbeats from some of the cluster nodes may be lost, and those nodes may get fenced out. This situation typically occurs when the CPU load or the memory usage is high or, when the disk snapshot or the VM migration operations are in progress in virtualized environments. To make a cluster more resilient to such transient issues, InfoScale

creates per-CPU kernel threads for heartbeating by default. It does so in addition to the existing heartbeat mechanism that works in the context of the kernel timer.

This feature is primarily useful in virtualized environments, but it provides the same kind of resiliency in physical environments as well.

To disable or enable use of per-CPU heartbeat threads

1. During LLT configuration, set the `hbthread` tuneable in `/etc/llttab` as follows:
   - To disable per-CPU heartbeat threads, add the line:
     ```
     set-misc hbthread:0
     ```
   - To enable per-CPU heartbeat threads, add:
     ```
     set-misc hbthread:1
     ```
2. Alternatively, at runtime, set the value as follows:
   - To disable per-CPU heartbeat threads, run the command:
     ```
     # lltconfig -H hbthread:0
     ```
   - To enable per-CPU heartbeat threads, run:
     ```
     # lltconfig -H hbthread:1
     ```
3. Optionally, to view the current value of the tunable, run:
   ```
   # lltconfig -H query | grep hbthread
   ```
   Sample output:
   ```
   hbthread = 1
   ```

To further improve resiliency, you may optionally want to remove the IRQ lines associated with the LLT interfaces from the control of `irqbalance` and spread them across all the CPUs. This setting disabled by default.

To enable or disable spreading of interrupts for LLT interfaces

◆ During LLT configuration, set the IRQ balance value in `/etc/sysconfig/llt` as follows:
   - To enable the spreading of interrupts for LLT across all the CPUs, add the line:
     ```
     LLT_IRQBALANCE=1
     ```
   - To disable the spreading of interrupts, add:
     ```
     LLT_IRQBALANCE=0
     ```

Administering the AMF kernel driver

Review the following procedures to start, stop, or unload the AMF kernel driver.

See "About the IMF notification module" on page 45.

See "Environment variables to start and stop VCS modules" on page 81.

To start the AMF kernel driver

1. Set the value of the AMF_START variable to 1 in the following file, if the value is not already 1:

   ```
   # /etc/sysconfig/amf
   ```

2. Start the AMF kernel driver. Run the following command:

   ```
   # systemctl start amf
   ```

To stop the AMF kernel driver

1. Set the value of the AMF_STOP variable to 1 in the following file, if the value is not already 1:

   ```
   # /etc/sysconfig/amf
   ```

2. Stop the AMF kernel driver. Run the following command:

   ```
   # systemctl stop amf
   ```

To unload the AMF kernel driver

1. If agent downtime is not a concern, use the following steps to unload the AMF kernel driver:

   - Stop the agents that are registered with the AMF kernel driver.
     The `amfstat` command output lists the agents that are registered with AMF under the Registered Reapers section.
     See the `amfstat` manual page.
   - Stop the AMF kernel driver.
     See "To stop the AMF kernel driver" on page 109.
   - Start the agents.

2. If you want minimum downtime of the agents, use the following steps to unload the AMF kernel driver:

   - Run the following command to disable the AMF driver even if agents are still registered with it.

     ```
     # amfconfig -Uof
     ```

- Stop the AMF kernel driver.
  See "To stop the AMF kernel driver" on page 109.

Starting VCS

You can start VCS using one of the following approaches:

- Using the `installer -start` command
- Manually start VCS on each node

To start VCS

1. To start VCS using the installer program, perform the following steps on any node in the cluster:
   - Log in as root user.
   - Run the following command:

     ```
     # /opt/VRTS/install/installer -start
     ```

2. To start VCS manually, run the following commands on each node in the cluster:
   - Log in as root user.
   - Start LLT and GAB. Start I/O fencing if you have configured it. Skip this step if you want to start VCS on a single-node cluster.
     Optionally, you can start AMF if you want to enable intelligent monitoring.

     LLT — For systemd environments with supported Linux distributions:

     ```
     # systemctl start llt
     ```

     For other supported Linux distributions:

     ```
     # /etc/init.d/llt start
     ```

     GAB — For systemd environments with supported Linux distributions:

     ```
     # systemctl start gab
     ```

     For other supported Linux distributions:

     ```
     # /etc/init.d/gab start
     ```

     I/O fencing — For systemd environments with supported Linux distributions:

     ```
     # systemctl stop vxfen
     ```

     For other supported Linux distributions:

     ```
     # /etc/init.d/vxfen stop
     ```

AMF

For systemd environments with supported Linux distributions:

```
# systemctl start amf
```

For other supported Linux distributions:

```
# /etc/init.d/amf start
```

- Start the VCS engine.

  ```
  # hastart
  ```

  To start VCS in a single-node cluster, run the following command:

  ```
  # hastart -onenode
  ```

  See "Starting the VCS engine (HAD) and related processes" on page 111.

3 Verify that the VCS cluster is up and running.

```
# gabconfig -a
```

Make sure that port a and port h memberships exist in the output for all nodes in the cluster. If you configured I/O fencing, port b membership must also exist.

Starting the VCS engine (HAD) and related processes

The command to start VCS is invoked from the following file:

For systemd environments with supported Linux distributions:

```
systemctl start vcs
```

For other supported Linux distributions:

```
/etc/init.d/vcs
```

When VCS is started, it checks the state of its local configuration file and registers with GAB for cluster membership. If the local configuration is valid, and if no other system is running VCS, it builds its state from the local configuration file and enters the RUNNING state.

If the configuration on all nodes is invalid, the VCS engine waits for manual intervention, or for VCS to be started on a system that has a valid configuration.

See "System states" on page 690.

To start the VCS engine

◆ Run the following command:

 # hastart

To start the VCS engine when all systems are in the ADMIN_WAIT state

- Run the following command from any system in the cluster to force VCS to use the configuration file from the system specified by the variable system:

```
# hasys -force system
```

To start VCS on a single-node cluster

- Type the following command to start an instance of VCS that does not require GAB and LLT. Do not use this command on a multisystem cluster.

```
# hastart -onenode
```

Starting VCS in a customized environment

Depending on the needs of your environment, you may want to perform a series of tasks before you start the VCS engine. For example, before you run `hastart`, you may want to move certain files to different locations, or you may want to perform some clean up tasks.

InfoScale Availability lets you configure the following files to customize your VCS startup environment and to customize how the VCS engine is started.

To disable the custom scripts, remove the scripts from the `$VCS_HOME/bin` folder, or remove the execute permissions on those files.

pre_hastart

To define the custom code to be called before `hastart`, do one of the following:

- Create the `pre_hastart` script in the `$VCS_HOME/bin` folder.
- Alternatively, create a copy of the sample script file that is provided in the `$VCS_HOME/bin/sample_scripts/VRTSvcs` folder. Modify the script as appropriate and place the file in the `$VCS_HOME/bin` folder.

The sample script contains sample code and return value translations. Use it as reference and modify the values to suit the needs of your environment.

Ensure that the script has executable permissions and that its file size is greater than 0 bytes. If these conditions are not met, the script is not executed before `hastart`.

Ensure that the custom script follows the return value specifications that are mentioned in the sample script.

A non-zero return value causes the vcs service to go into Failed state.

Check the service logs for messages that are logged by the custom script.

custom_vcsenv

Create the `custom_vcsenv` file to define custom values for the VCS environment variables and place it in the `$VCS_HOME/bin` folder.

To set a variable, use the syntax that is appropriate for the shell in which VCS starts. For example, typically, VCS starts in `/bin/sh`, so you can define the variables as:

```
VCS_GAB_TIMEOUT = 35000;export VCS_GAB_TIMEOUT
```

These variables are set for VCS when the `hastart` command is run.

custom_had_start

The `hastart` script calls the custom script to start the `had` binary.

To define the custom code to be called before `hastart`, do one of the following

- Create the `custom_had_start` script in the `$VCS_HOME/bin` folder.
- Alternatively, create a copy of the sample script file that is provided in the `$VCS_HOME/bin/sample_scripts/VRTSvcs` folder. Modify the script as appropriate and place the file in the `$VCS_HOME/bin` folder.

The sample script contains sample code and return value translations. Use it as reference and modify the values to suit the needs of your environment.

Ensure that the script has executable permissions and that its file size is greater than 0 bytes. If these conditions are not met, HAD startup is performed in the default manner.

Ensure that the custom script follows the return value specifications that are mentioned in the sample script.

A non-zero return value causes the `vcs` service to go into Failed state.

Check the service logs for messages that are logged by the custom script.

Define appropriate log messages to determine whether the HAD daemon is started using the `$VCS_HOME/bin/custom_had_start` script.

Stopping VCS

You can stop VCS using one of the following approaches:

- Using the `installer -stop` command
- Manually stop VCS on each node

To stop VCS

1. To stop VCS using the installer program, perform the following steps on any node in the cluster:
   - Log in as the root user.
   - Run the following command:

     ```
     # /opt/VRTS/install/installer -stop
     ```

2. To stop VCS manually, run the following commands on each node in the cluster:
   - Log in as the root user.
   - Take the VCS service groups offline and verify that the groups are offline.

     ```
     # hagrp -offline service_group -sys system
     ```

     ```
     # hagrp -state service_group
     ```

   - Stop the VCS engine.

     ```
     # hastop -local
     ```

     See "Stopping the VCS engine and related processes" on page 115.
   - Verify that the VCS engine port h is closed.

     ```
     # gabconfig -a
     ```

   - Stop I/O fencing if you have configured it. Stop GAB and then LLT.

I/O fencing	For systemd environments with supported Linux distributions:
	`# systemctl stop vxfen`
	For other supported Linux distributions:
	`# /etc/init.d/vxfen stop`

GAB	For systemd environments with supported Linux distributions:
	`# systemctl stop gab`
	For other supported Linux distributions:
	`# /etc/init.d/gab stop`

LLT	For systemd environments with supported Linux distributions:
	`# systemctl stop llt`
	For other supported Linux distributions:
	`# /etc/init.d/llt stop`

Stopping VCS without evacuating service groups

By default, when VCS is stopped as part of a system restart operation, the active service groups on the node are migrated to another cluster node. In some cases, you may not want to evacuate the service groups during a system restart. For example, you may want to avoid administrative intervention during a manual shutdown. InfoScale lets you choose whether or not to evacuate service groups when VCS is stopped.

The NOEVACUATE environment variable lets you specify whether or not to evacuate service groups when a node is shut down or restarted.

- The default value of NOEVACUATE is 0, which specifies that the service groups should be evacuated when VCS is stopped.
  The service groups are evacuated by executing the following command, before the system is stopped:
  ```
  hastop -local -noautodisable
  ```
- If this value is set to 1, the VCS stop script does not evacuate the service groups.
  The service groups evacuation is skipped because the following command is executed, depending on the VCS time-out value:
  ```
  hastop -sysoffline -noautodisable
  ```

This variable is present in the `/etc/sysconfig/vcs` file.

Stopping the VCS engine and related processes

The `hastop` command stops the High Availability Daemon (HAD) and related processes. You can customize the behavior of the hastop command by configuring the EngineShutdown attribute for the cluster.

See "About controlling the hastop behavior by using the EngineShutdown attribute" on page 117.

The hastop command includes the following options:

```
hastop -all [-force]
hastop [-help]
```

```
hastop -local [-force | -evacuate | -noautodisable]
hastop -sys system ... [-force | -evacuate -noautodisable]
```

Table 5-2 shows the options for the `hastop` command.

Table 5-2 Options for the `hastop` command

Option	Description
`-all`	Stops HAD on all systems in the cluster and takes all service groups offline.
`-help`	Displays the command usage.
`-local`	Stops HAD on the system on which you typed the command
`-force`	Allows HAD to be stopped without taking service groups offline on the system. The value of the EngineShutdown attribute does not influence the behavior of the -force option.
`-evacuate`	When combined with `-local` or `-sys`, migrates the system's active service groups to another system in the cluster, before the system is stopped.
`-noautodisable`	Ensures the service groups that can run on the node where the `hastop` command was issued are not autodisabled. This option can be used with `-evacuate` but not with `-force`.
`-sys`	Stops HAD on the specified system.

About stopping VCS without the -force option

When VCS is stopped on a system without using the `-force` option, it enters the LEAVING state, and waits for all groups to go offline on the system. Use the output of the command `hasys -display` *`system`* to verify that the values of the SysState and the OnGrpCnt attributes are non-zero. VCS continues to wait for the service groups to go offline before it shuts down.

See "Troubleshooting resources" on page 648.

About stopping VCS with options other than the -force option

When VCS is stopped by options other than `-force` on a system with online service groups, the groups that run on the system are taken offline and remain offline. VCS indicates this by setting the attribute IntentOnline to 0. Use the option `-force` to enable service groups to continue being online while the VCS engine (HAD) is

brought down and restarted. The value of the IntentOnline attribute remains unchanged after the VCS engine restarts.

About controlling the hastop behavior by using the EngineShutdown attribute

Use the EngineShutdown attribute to define VCS behavior when a user runs the hastop command.

Note: VCS does not consider this attribute when the hastop is issued with the following options: -force or -local -evacuate -noautodisable.

Configure one of the following values for the attribute depending on the desired functionality for the hastop command:

Table 5-3 shows the engine shutdown values for the attribute.

Table 5-3 Engine shutdown values

EngineShutdown Value	Description
Enable	Process all hastop commands. This is the default behavior.
Disable	Reject all hastop commands.
DisableClusStop	Do not process the hastop -all command; process all other hastop commands.
PromptClusStop	Prompt for user confirmation before you run the `hastop -all` command; process all other hastop commands.
PromptLocal	Prompt for user confirmation before you run the `hastop -local` command; process all other hastop commands except `hastop -sys` command.
PromptAlways	Prompt for user confirmation before you run any `hastop` command.

Additional considerations for stopping VCS

Following are some additional considerations for stopping VCS:

- If you use the command `reboot`, behavior is controlled by the ShutdownTimeOut parameter. After HAD exits, if GAB exits within the time designated in the ShutdownTimeout attribute, the remaining systems recognize this as a reboot and fail over service groups from the departed system. For systems that run

several applications, consider increasing the value of the ShutdownTimeout attribute.

- If you stop VCS on a system by using the `hastop` command, it autodisables each service group that includes the system in their SystemList attribute. VCS does not initiate online of the servicegroup when in an autodisable state. (This does not apply to systems that are powered off)
- If you use the `-evacuate` option, evacuation occurs before VCS is brought down. But when there are dependencies between the service groups while `-evacuate` command is issued, VCS rejects the command

Logging on to VCS

VCS prompts for user name and password information when non-root users run *haxxx* commands. Use the `halogin` command to save the authentication information so that you do not have to enter your credentials every time you run a VCS command. Note that you may need specific privileges to run VCS commands.

When you run the `halogin` command, VCS stores encrypted authentication information in the user's home directory. For clusters that run in secure mode, the command also sets up a trust relationship and retrieves a certificate from an authentication broker.

If you run the command for different hosts, VCS stores authentication information for each host. After you run the command, VCS stores the information until you end the session.

For clusters that run in secure mode, you also can generate credentials for VCS to store the information for 24 hours or for eight years and thus configure VCS to not prompt for passwords when you run VCS commands as non-root users.

See "Running high availability commands (HA) commands as non-root users on clusters in secure mode" on page 119.

Root users do not need to run `halogin` when running VCS commands from the local host.

To log on to a cluster running in secure mode

1. Set the following environment variables:
   - VCS_DOMAIN—Name of the Security domain to which the user belongs.

- VCS_DOMAINTYPE—Type of VxSS domain: unixpwd, nt, ldap, nis, nisplus, or vx.

2. Define the node on which the VCS commands will be run. Set the VCS_HOST environment variable to the name of the node. To run commands in a remote cluster, you set the variable to the virtual IP address that was configured in the ClusterService group.
3. Log on to VCS:

```
# halogin vcsusername password
```

To log on to a cluster not running in secure mode

1. Define the node on which the VCS commands will be run. Set the VCS_HOST environment variable to the name of the node on which to run commands. To run commands in a remote cluster, you can set the variable to the virtual IP address that was configured in the ClusterService group.
2. Log on to VCS:

```
# halogin vcsusername password
```

To end a session for a host

◆ Run the following command:

```
# halogin -endsession hostname
```

To end all sessions

◆ Run the following command:

```
# halogin -endallsessions
```

After you end a session, VCS prompts you for credentials every time you run a VCS command.

Running high availability commands (HA) commands as non-root users on clusters in secure mode

Perform the following procedure to configure VCS to not prompt for passwords when a non-root user runs HA commands on a cluster which runs in secure mode.

To run HA commands as non-root users on clusters in secure mode

When a non-root user logs in, the logged in user context is used for HA commands, and no password is prompted. For example, if user *sam* has logged in to the host *@example.com*, and if VCS configuration specifies *sam@example.com* as the

group administrator for group G1, then the non-root user gets administrative privileges for group G1.

- Run any HA command. This step generates credentials for VCS to store the authentication information. For example, run the following command:

  ```
  # hasys -state
  ```

About managing VCS configuration files

This section describes how to verify, back up, and restore VCS configuration files.

See "About the main.cf file" on page 64.

See "About the types.cf file" on page 68.

About multiple versions of .cf files

When hacf creates a .cf file, it does not overwrite existing .cf files. A copy of the file remains in the directory, and its name includes a suffix of the date and time it was created, such as main.cf.03Dec2001.17.59.04. In addition, the previous version of any .cf file is saved with the suffix .previous; for example, main.cf.previous.

Verifying a configuration

Use hacf to verify (check syntax of) the main.cf and the type definition file, types.cf. VCS does not run if hacf detects errors in the configuration.

To verify a configuration

- Run the following command:

  ```
  # hacf -verify config_directory
  ```

  The variable *config_directory* refers to directories containing a main.cf file and any .cf files included in main.cf.

  No error message and a return value of zero indicates that the syntax is legal.

Scheduling automatic backups for VCS configuration files

Configure the BackupInterval attribute to instruct VCS to create a back up of the configuration periodically. VCS backs up the main.cf and types.cf files as main.cf.autobackup and types.cf.autobackup, respectively.

To start periodic backups of VCS configuration files

- Set the cluster-level attribute BackupInterval to a non-zero value.

  For example, to back up the configuration every 5 minutes, set BackupInterval to 5.

  Example:

```
# haclus -display | grep BackupInterval

BackupInterval         0

# haconf -makerw

# haclus -modify BackupInterval 5

# haconf -dump -makero
```

Saving a configuration

When you save a configuration, VCS renames the file main.cf.autobackup to main.cf. VCS also save your running configuration to the file main.cf.autobackup.

If have not configured the BackupInterval attribute, VCS saves the running configuration.

See "Scheduling automatic backups for VCS configuration files" on page 120.

To save a configuration

- Run the following command

```
# haconf -dump -makero
```

  The option -makero sets the configuration to read-only.

Setting the configuration to read or write

This topic describes how to set the configuration to read/write.

To set the mode to read or write

- Type the following command:

```
# haconf -makerw
```

Displaying configuration files in the correct format

When you manually edit VCS configuration files (for example, the main.cf or types.cf file), you create formatting issues that prevent the files from being parsed correctly.

To display the configuration files in the correct format

◆ Run the following commands to display the configuration files in the correct format:

```
# hacf -cftocmd config
# hacf -cmdtocf config
```

About managing VCS users from the command line

You can add, modify, and delete users on any system in the cluster, provided you have the privileges to do so.

If VCS is running in secure mode, specify user names, in one of the following formats:

- username@domain
- username

Specify only the user name to assign cluster-level privileges to a user. The user can access any node in the cluster. Privileges for a cluster-level user are the same across nodes in the cluster.

Specify the user name and the domain name to add a user on multiple nodes in the cluster. This option requires multiple entries for a user, one for each node.

You cannot assign or change passwords for users when VCS is running in secure mode.

The commands to add, modify, and delete a user must be executed only as root or administrator and only if the VCS configuration is in read/write mode.

See "Setting the configuration to read or write" on page 121.

Note: You must add users to the VCS configuration to monitor and administer VCS from the graphical user interface Cluster Manager.

Adding a user

Users in the category Cluster Guest cannot add users.

Before you run the `hauser` command to add a user in the non-secure, you must run the following command:

```
# vcsencrypt -gensecinfo
```

Otherwise, the `hauser` command fails and logs an error, which states that the security key is not generated.

To add a user

1. Set the configuration to read/write mode:

   ```
   # haconf -makerw
   ```

2. Add the user:

   ```
   # hauser -add user [-priv <Administrator|Operator> [-group
           service_groups]]
   ```

3. Enter a password when prompted.

   The encrypted password is added to the configuration; for example:

   ```
   UserNames = { root = "a4:97:e4:1e:00:1c:94:e4:cd:d2:4e:ad:
   23:78:90:11:f2:f2:aa:fa:ab:11:5b:06:cd:d2:4e:ad:23:78:90:11"}
   ```

   It also updates the cluster-level attribute IV256List, by adding a new entry that includes the encrypted password and the IV value that is used for the encryption as a key-value pair.

   For example:

   ```
   IV256List = {
   "fc:03:27:81:3b:71:47:83:38:33:02:bb:0c:d3:05:db:59:e1:52:44:a6:9b:9a:95:16:ce:c0:a4:94:d7:b0:80"
   =
   "e0:ca:04:b4:eb:4a:07:d3:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00",
   "af:d5:37:05:65:ca:fc:3e:db:13:72:f9:d0:4b:be:4f:3c:66:6e:58:b1:ec:0a:39:4d:74:85:fc:aa:3e:03:fd"
   =
   "83:a5:ad:54:07:d2:ba:61:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00"
   }
   ```

To add a user with cluster administrator access

- Type the following command:

  ```
  # hauser -add user -priv Administrator
  ```

To add a user with cluster operator access

◆ Type the following command:

```
# hauser -add user -priv Operator
```

To add a user with group administrator access

◆ Type the following command:

```
# hauser -add user -priv Administrator -group service_groups
```

To add a user with group operator access

◆ Type the following command:

```
# hauser -add user -priv Operator -group service_groups
```

To add a user on only one node with cluster administrator access

1 Set the configuration to read/write mode:

```
# haconf -makerw
```

2 Add the user:

```
# hauser -add user@node.domain -priv Administrator
```

For example,

```
# hauser -add user1@sys1.domain1.com -priv  Administrator
```

To add a user on only one node with group administrator access

◆ Type the following command:

```
# hauser -add user@node.domain -priv Administrator -group service_groups
```

Assigning and removing user privileges

The following procedure desribes how to assign and remove user privileges:

To assign privileges to an administrator or operator

◆ Type the following command:

```
hauser -addpriv user Adminstrator|Operator
        [-group service_groups]
```

To remove privileges from an administrator or operator

◆ Type the following command:

```
hauser -delpriv user Adminstrator|Operator
            [-group service_groups]
```

To assign privileges to an OS user group

◆ Type the following command:

```
hauser -addpriv usergroup AdminstratorGroup|OperatorGroup
        [-group service_groups]
```

To remove privileges from an OS user group

◆ Type the following command:

```
hauser -delpriv usergroup AdminstratorGroup|OperatorGroup
    [-group service_groups]
```

Modifying a user

Users of the Cluster Guest category cannot modify users.

You cannot modify a VCS user in clusters that run in secure mode.

To modify a user

1. Set the configuration to read or write mode:

   ```
   # haconf -makerw
   ```

2. Enter the following command to modify the user:

   ```
   # hauser -update user
   ```

3. Enter a new password when prompted.

Deleting a user

You can delete a user from the VCS configuration.

To delete a user

1. Set the configuration to read or write mode:

```
# haconf -makerw
```

2. For users with Administrator and Operator access, remove their privileges:

```
# hauser -delpriv user Adminstrator|Operator [-group
        service_groups]
```

3. Delete the user from the list of registered users:

```
# hauser -delete user
```

4. Reset the configuration to read-only:

```
# haconf -dump -makero
```

Displaying a user

This topic describes how to display a list of users and their privileges.

To display a list of users

- Type the following command:

```
# hauser -list
```

To display the privileges of all users

- Type the following command:

```
# hauser -display
```

To display the privileges of a specific user

- Type the following command:

```
# hauser -display user
```

About querying VCS

VCS enables you to query various cluster objects, including resources, service groups, systems, resource types, agents, clusters, and sites. You may execute commands from any system in the cluster. Commands to display information on

the VCS configuration or system states can be executed by all users: you do not need root privileges.

Querying service groups

This topic describes how to perform a query on service groups.

To display the state of a service group on a system

- Type the following command:

```
# hagrp -state [service_group] [-sys system]
```

To display the resources for a service group

- Type the following command:

```
# hagrp -resources service_group
```

To display a list of a service group's dependencies

- Type the following command:

```
# hagrp -dep [service_group]
```

To display a service group on a system

- Type the following command:

```
# hagrp -display [service_group] [-sys system]
```

If *service_group* is not specified, information regarding all service groups is displayed.

To display the attributes of a system

- Type the following command:

```
# hagrp -display [service_group]
[-attribute attribute] [-sys system]
```

Note that system names are case-sensitive.

To display the value of a service group attribute

- Type the following command:

```
# hagrp -value service_group attribute
```

To forecast the target system for failover

- Use the following command to view the target system to which a service group fails over during a fault:

  ```
  # hagrp -forecast service_group [-policy failoverpolicy value] [-verbose]
  ```

  The policy values can be any one of the following values: Priority, RoundRobin, Load or BiggestAvailable.

  Note: You cannot use the `-forecast` option when the service group state is in transition. For example, VCS rejects the command if the service group is in transition to an online state or to an offline state.

  The `-forecast` option is supported only for failover service groups. In case of offline failover service groups, VCS selects the target system based on the service group's failover policy.

  The BiggestAvailable policy is applicable only when the service group attribute Load is defined and cluster attribute Statistics is enabled.

  The actual service group FailOverPolicy can be configured as any policy, but the forecast is done as though FailOverPolicy is set to BiggestAvailable.

Querying resources

This topic describes how to perform a query on resources.

To display a resource's dependencies

- Enter the following command:

  ```
  # hares -dep resource
  ```

  Note: If you use this command to query atleast resource dependency, the dependency is displayed as a normal 1-to-1 dependency.

To display information about a resource

- Enter the following command:

  ```
  # hares -display resource
  ```

  If *resource* is not specified, information regarding all resources is displayed.

To display resources of a service group

- Enter the following command:

```
# hares -display -group service_group
```

To display resources of a resource type

- Enter the following command:

```
# hares -display -type resource_type
```

To display resources on a system

- Enter the following command:

```
# hares -display -sys system
```

To display the value of a specific resource attribute

- Enter the following command:

```
# hares -value resource attribute
```

To display atleast resource dependency

- Enter the following command:

```
# hares -dep -atleast resource
```

The following is an example of the command output:

```
#hares -dep -atleast
#Group  Parent  Child
g1      res1    res2,res3,res5,res6 min=2
g1      res7    res8
```

Querying resource types

This topic describes how to perform a query on resource types.

To display all resource types

- Type the following command:

```
# hatype -list
```

To display resources of a particular resource type

- Type the following command:

```
# hatype -resources resource_type
```

To display information about a resource type

- Type the following command:

```
# hatype -display resource_type
```

If *resource_type* is not specified, information regarding all types is displayed.

To display the value of a specific resource type attribute

- Type the following command:

```
# hatype -value resource_type attribute
```

Querying agents

Table 5-4 lists the run-time status for the agents that the `haagent -display` command displays.

Table 5-4 Run-time status for the agents

Run-time status	Definition
Faults	Indicates the number of agent faults within one hour of the time the fault began and the time the faults began.
Messages	Displays various messages regarding agent status.
Running	Indicates the agent is operating.
Started	Indicates the file is executed by the VCS engine (HAD).

To display the run-time status of an agent

- Type the following command:

```
# haagent -display [agent]
```

If *agent* is not specified, information regarding all agents appears.

To display the value of a specific agent attribute

- Type the following command:

```
# haagent -value agent attribute
```

Querying systems

This topic describes how to perform a query on systems.

To display a list of systems in the cluster

- Type the following command:

```
# hasys -list
```

To display information about each system

- Type the following command:

```
# hasys -display [system]
```

If you do not specify a system, the command displays attribute names and values for all systems.

To display the value of a specific system attribute

- Type the following command:

```
# hasys -value system attribute
```

To display system attributes related to resource utilization

- Use the following command to view the resource utilization of the following system attributes:
  - Capacity
  - HostAvailableForecast
  - HostUtilization

```
# hasys -util system
```

The `-util` option is applicable only if you set the cluster attribute Statistics to Enabled and define at least one key in the cluster attribute HostMeters.

The command also indicates if the HostUtilization, and HostAvailableForecast values are stale.

Querying clusters

This topic describes how to perform a query on clusters.

To display the value of a specific cluster attribute

◆ Type the following command:

```
# haclus -value attribute
```

To display information about the cluster

◆ Type the following command:

```
# haclus -display
```

Querying status

This topic describes how to perform a query on status of service groups in the cluster.

Note: Run the `hastatus` command with the `-summary` option to prevent an incessant output of online state transitions. If the command is used without the option, it will repeatedly display online state transitions until it is interrupted by the command CTRL+C.

To display the status of all service groups in the cluster, including resources

◆ Type the following command:

```
# hastatus
```

To display the status of a particular service group, including its resources

◆ Type the following command:

```
# hastatus [-sound] [-time] -group service_group
[-group service_group]...
```

If you do not specify a service group, the status of all service groups appears. The `-sound` option enables a bell to ring each time a resource faults.

The `-time` option prints the system time at which the status was received.

To display the status of service groups and resources on specific systems

◆ Type the following command:

```
# hastatus [-sound] [-time] -sys system_name
[-sys system_name]...
```

To display the status of specific resources

◆ Type the following command:

```
# hastatus [-sound] [-time] -resource resource_name
[-resource resource_name]...
```

To display the status of cluster faults, including faulted service groups, resources, systems, links, and agents

◆ Type the following command:

```
# hastatus -summary
```

Querying log data files (LDFs)

Log data files (LDFs) contain data regarding messages written to a corresponding English language file. Typically, for each English file there is a corresponding LDF.

To display the hamsg usage list

◆ Type the following command:

```
# hamsg -help
```

To display the list of LDFs available on the current system

◆ Type the following command:

```
# hamsg -list
```

To display general LDF data

◆ Type the following command:

```
# hamsg -info [-path path_name] LDF
```

The option `-path` specifies where `hamsg` looks for the specified LDF. If not specified, `hamsg` looks for files in the default directory:

/var/VRTSvcs/ldf

To display specific LDF data

◆ Type the following command:

```
# hamsg [-any] [-sev C|E|W|N|I]
[-otype VCS|RES|GRP|SYS|AGT]
```

[-oname *object_name*] [-cat *category*] [-msgid *message_ID*] [-path *path_name*] [-lang *language*] *LDF_file*

`-any` Specifies `hamsg` return messages that match any of the specified query options.

`-sev` Specifies `hamsg` return messages that match the specified message severity Critical, Error, Warning, Notice, or Information.

`-otype` Specifies `hamsg` return messages that match the specified object type

- VCS = general VCS messages
- RES = resource
- GRP = service group
- SYS = system
- AGT = agent

`-oname` Specifies `hamsg` return messages that match the specified object name.

`-cat` Specifies `hamsg` return messages that match the specified category. For example, the value 2 in the message id “V-16-2-13067”

`-msgid` Specifies `hamsg` return messages that match the specified message ID. For example, the value 13067 the message id “V-16-2-13067”'

`-path` Specifies where `hamsg` looks for the specified LDF. If not specified, `hamsg` looks for files in the default directory:

/var/VRTSvcs/ldf

`-lang` Specifies the language in which to display messages. For example, the value en specifies English and "ja" specifies Japanese.

Using conditional statements to query VCS objects

Some query commands include an option for conditional statements. Conditional statements take three forms:

`Attribute=Value` (the attribute equals the value)

`Attribute!=Value` (the attribute does not equal the value)

`Attribute=~Value` (the value is the prefix of the attribute, for example a query for the state of a resource = ~FAULTED returns all resources whose state begins with FAULTED.)

Multiple conditional statements can be used and imply `AND` logic.

You can only query attribute-value pairs that appear in the output of the command `hagrp -display`.

See "Querying service groups" on page 127.

To display the list of service groups whose values match a conditional statement

- Type the following command:

  ```
  # hagrp -list [conditional_statement]
  ```

  If no conditional statement is specified, all service groups in the cluster are listed.

To display a list of resources whose values match a conditional statement

- Type the following command:

  ```
  # hares -list [conditional_statement]
  ```

  If no conditional statement is specified, all resources in the cluster are listed.

To display a list of agents whose values match a conditional statement

- Type the following command:

  ```
  # haagent -list [conditional_statement]
  ```

  If no conditional statement is specified, all agents in the cluster are listed.

About administering service groups

Administration of service groups includes tasks such as adding, deleting, or modifying service groups.

Adding and deleting service groups

This topic describes how to add or delete a service group.

To add a service group to your cluster

- Type the following command:

  ```
  # hagrp -add service_group
  ```

  The variable *service_group* must be unique among all service groups defined in the cluster.

  This command initializes a service group that is ready to contain various resources. To employ the group properly, you must populate its SystemList attribute to define the systems on which the group may be brought online and taken offline. (A system list is an association of names and integers that represent priority values.)

To delete a service group

- Type the following command:

  ```
  # hagrp -delete service_group
  ```

  Note that you cannot delete a service group until all of its resources are deleted.

Modifying service group attributes

This topic describes how to modify service group attributes.

To modify a service group attribute

◆ Type the following command:

```
# hagrp -modify service_group attribute value [-sys system]
```

The variable *value* represents:

system_name1 priority1 system_name2 priority2

If the attribute that is being modified has local scope, you must specify the system on which to modify the attribute, except when modifying the attribute on the system from which you run the command.

For example, to populate the system list of service group groupx with Systems A and B, type:

```
# hagrp -modify groupx SystemList -add SystemA 1 SystemB 2
```

Similarly, to populate the AutoStartList attribute of a service group, type:

```
# hagrp -modify groupx AutoStartList SystemA SystemB
```

You may also define a service group as parallel. To set the Parallel attribute to 1, type the following command. (Note that the default for this attribute is 0, which designates the service group as a failover group.):

```
# hagrp -modify groupx Parallel 1
```

You cannot modify this attribute if resources have already been added to the service group.

You can modify the attributes SystemList, AutoStartList, and Parallel only by using the command `hagrp -modify`. You cannot modify attributes created by the system, such as the state of the service group.

Modifying the SystemList attribute

You use the `hagrp -modify` command to change a service group's existing system list, you can use the options `-modify`, `-add`, `-update`, `-delete`, or `-delete -keys`.

For example, suppose you originally defined the SystemList of service group groupx as SystemA and SystemB. Then after the cluster was brought up you added a new system to the list:

```
# hagrp -modify groupx SystemList -add SystemC 3
```

You must take the service group offline on the system that is being modified.

When you add a system to a service group's system list, the system must have been previously added to the cluster. When you use the command line, you can use the `hasys -add` command.

When you delete a system from a service group's system list, the service group must not be online on the system to be deleted.

If you attempt to change a service group's existing system list by using `hagrp -modify` without other options (such as `-add` or `-update`) the command fails.

Bringing service groups online

This topic describes how to bring service groups online.

To bring a service group online

◆ Type one of the following commands:

```
# hagrp -online service_group -sys system

# hagrp -online service_group -site [site_name]
```

To start a service group on a system and bring online only the resources already online on another system

◆ Type the following command:

```
# hagrp -online service_group -sys system
-checkpartial other_system
```

If the service group does not have resources online on the other system, the service group is brought online on the original system and the `checkpartial` option is ignored.

Note that the `checkpartial` option is used by the Preonline trigger during failover. When a service group that is configured with Preonline =1 fails over to another system (system 2), the only resources brought online on system 2 are those that were previously online on system 1 prior to failover.

To bring a service group and its associated child service groups online

◆ Type one of the following commands:

- `# hagrp -online -propagate service_group -sys system`
- `# hagrp -online -propagate service_group -any`
- `# hagrp -online -propagate service_group -site site`

Note: See the man pages associated with the `hagrp` command for more information about the `-propagate` option.

Taking service groups offline

This topic describes how to take service groups offline.

To take a service group offline

◆ Type one of the following commands:

```
# hagrp -offline service_group -sys system
# hagrp -offline service_group -site site_name
```

To take a service group offline only if all resources are probed on the system

◆ Type the following command:

```
# hagrp -offline [-ifprobed] service_group -sys system
```

To take a service group and its associated parent service groups offline

◆ Type one of the following commands:

- `# hagrp -offline -propagate service_group -sys system`
- `# hagrp -offline -propagate service_group -any`
- `# hagrp -offline -propagate service_group -site site_name`

Note: See the man pages associated with the `hagrp` command for more information about the `-propagate` option.

Switching service groups

The process of switching a service group involves taking it offline on its current system or site and bringing it online on another system or site.

To switch a service group from one system to another

◆ Type the following command:

```
# hagrp -switch service_group -to system
```

```
# hagrp -switch service_group -site site_name
```

A service group can be switched only if it is fully or partially online. The `-switch` option is not supported for switching hybrid service groups across system zones.

Switch parallel global groups across clusters by using the following command:

```
# hagrp -switch service_group -any -clus remote_cluster
```

VCS brings the parallel service group online on all possible nodes in the remote cluster.

Migrating service groups

VCS provides live migration capabilities for service groups that have resources to monitor virtual machines. The process of migrating a service group involves concurrently moving the service group from the source system to the target system with minimum downtime.

To migrate a service group from one system to another

◆ Type the following command:

```
# hagrp -migrate service_group -to system
```

A service group can be migrated only if it is fully online. The `-migrate` option is supported only for failover service groups and for resource types that have the SupportedOperations attribute set to migrate.

See "Resource type attributes" on page 703.

The service group must meet the following requirements regarding configuration:

- A single mandatory resource that can be migrated, having the SupportedOperations attribute set to migrate and the Operations attribute set to OnOff
- Other optional resources with Operations attribute set to None or OnOnly

The `-migrate` option is supported for the following configurations:

- Stand alone service groups
- Service groups having one or both of the following configurations:

- Parallel child service groups with online local soft or online local firm dependencies
- Parallel or failover parent service group with online global soft or online remote soft dependencies

Freezing and unfreezing service groups

Freeze a service group to prevent it from failing over to another system. This freezing process stops all online and offline procedures on the service group.

Note that if the service group is in ONLINE state and if you freeze the service group, then the group continues to remain in ONLINE state.

Unfreeze a frozen service group to perform online or offline operations on the service group.

To freeze a service group (disable online, offline, and failover operations)

◆ Type the following command:

```
# hagrp -freeze service_group [-persistent]
```

The option `-persistent` enables the freeze to be remembered when the cluster is rebooted.

To unfreeze a service group (reenable online, offline, and failover operations)

◆ Type the following command:

```
# hagrp -unfreeze service_group [-persistent]
```

Enabling and disabling service groups

Enable a service group before you bring it online. A service group that was manually disabled during a maintenance procedure on a system may need to be brought online after the procedure is completed.

Disable a service group to prevent it from coming online. This process temporarily stops VCS from monitoring a service group on a system that is undergoing maintenance operations

To enable a service group

◆ Type the following command:

```
# hagrp -enable service_group [-sys system]
```

A group can be brought online only if it is enabled.

To disable a service group

◆ Type the following command:

```
# hagrp -disable service_group [-sys system]
```

A group cannot be brought online or switched if it is disabled.

To enable all resources in a service group

◆ Type the following command:

```
# hagrp -enableresources service_group
```

To disable all resources in a service group

◆ Type the following command:

```
# hagrp -disableresources service_group
```

Agents do not monitor group resources if resources are disabled.

Enabling and disabling priority based failover for a service group

This topic describes how to enable and disable priority based failover for a service group. Even though the priority based failover gets configured for all the service groups in a cluster, you must enable the feature at the cluster level.

To enable priority based failover of your service group

- Run the following command on the cluster:

  ```
  # haclus -modify EnablePBF 1
  ```

  This attribute enables Priority based failover for high priority service group. During a failover, VCS checks the load requirement for the high-priority service group and evaluates the best available target that meets the load requirement. If none of the available systems meet the load requirement, then VCS checks for any lower priority groups that can be offlined to meet the load requirement.

VCS performs the following checks, before failing over the high-priority service group:

- Check the best target system that meets the load requirement of the high-priority service group.
- If no system meets the load requirement, create evacuation plan for each target system to see which system will have the minimum disruption.
- Evacuation plan consists of a list of low priority service groups that need to be taken offline to meet the load requirement. Disruption factor is calculated for the

system along with the evacuation plan. Disruption factor is based on the priority of the service groups that will be brought offline or evacuated. The following table shows the mapping of service group priority with the disruption factor:

Service group priority	Disruption factor
4	1
3	10
2	100
1	Will not be evacuated

- Choose the System with minimum disruption as the target system and execute evacuation plan. This initiates the offline of the low priority service groups. Plan is visible under attribute (Group::EvacList) . After all the groups in the evacuation plan are offline, initiate online of high priority service group.

To disable priority based failover of your service group:

- Run the following command on the service group:

```
# haclus -modify EnablePBF 0
```

Note: This attribute cannot be modified when evacuation of a group is in progress.

Clearing faulted resources in a service group

Clear a resource to remove a fault and make the resource available to go online.

To clear faulted, non-persistent resources in a service group

◆ Type the following command:

```
# hagrp -clear service_group [-sys system]
```

Clearing a resource initiates the online process previously blocked while waiting for the resource to become clear.

- If *system* is specified, all faulted, non-persistent resources are cleared from that system only.
- If *system* is not specified, the service group is cleared on all systems in the group's SystemList in which at least one non-persistent resource has faulted.

To clear resources in ADMIN_WAIT state in a service group

◆ Type the following command:

```
# hagrp -clearadminwait [-fault] service_group -sys system
```

See " Changing agent file paths and binaries" on page 383.

Flushing service groups

When a service group is brought online or taken offline, the resources within the group are brought online or taken offline. If the online operation or offline operation hangs on a particular resource, flush the service group to clear the WAITING TO GO ONLINE or WAITING TO GO OFFLINE states from its resources. Flushing a service group typically leaves the service group in a partial state. After you complete this process, resolve the issue with the particular resource (if necessary) and proceed with starting or stopping the service group.

Note: The flush operation does not halt the resource operations (such as online, offline, migrate, and clean) that are running. If a running operation succeeds after a flush command was fired, the resource state might change depending on the operation.

Use the command `hagrp -flush` to clear the internal state of VCS. The `hagrp -flush` command transitions resource state from 'waiting to go online' to 'not waiting'. You must use the `hagrp -flush -force` command to transition resource state from 'waiting to go offline' to 'not waiting'.

To flush a service group on a system

◆ Type the following command:

```
# hagrp -flush [-force]  group
-sys system [-clus cluster | -localclus]
```

To flush all service groups on a system

1 Save the following script as haflush at the location /opt/VRTSvcs/bin/

```
#!/bin/ksh
PATH=/opt/VRTSvcs/bin:$PATH; export PATH
if [ $# -ne 1 ]; then
    echo "usage: $0 <system name>"
    exit 1
fi

hagrp -list |
while read grp sys junk
do
    locsys="${sys##*:}"
    case "$locsys" in
    "$1")
        hagrp -flush "$grp" -sys "$locsys"
        ;;
    esac
done
```

2 Run the script.

```
# haflush systemname
```

Linking and unlinking service groups

This topic describes how to link service groups to create a dependency between them.

See "About service group dependencies" on page 407.

To link service groups

◆ Type the following command

```
# hagrp -link parent_group child_group
gd_category gd_location [gd_type]
```

parent_group	Name of the parent group
child_group	Name of the child group
gd_category	Category of group dependency (online/offline).
gd_location	The scope of dependency (local/global/remote/site).
gd_type	Type of group dependency (soft/firm/hard). Default is firm.

To unlink service groups

◆ Type the following command:

```
# hagrp -unlink parent_group child_group
```

Administering agents

Under normal conditions, VCS agents are started and stopped automatically.

To start an agent

◆ Run the following command:

```
haagent -start agent -sys system
```

To stop an agent

◆ Run the following command:

```
haagent -stop agent [-force] -sys system
```

The `-force` option stops the agent even if the resources for the agent are online. Use the -force option when you want to upgrade an agent without taking its resources offline.

About administering resources

Administration of resources includes tasks such as adding, deleting, modifying, linking, unlinking , probing, and clearing resources, bringing resources online, and taking them offline.

About adding resources

When you add a resource, all non-static attributes of the resource's type, plus their default values, are copied to the new resource.

Three attributes are also created by the system and added to the resource:

- Critical (default = 1). If the resource or any of its children faults while online, the entire service group is marked faulted and failover occurs.
  In an atleast resource set, the fault of a critical resource is tolerated until the minimum requirement associated with the resource set is met.
- AutoStart (default = 1). If the resource is set to AutoStart, it is brought online in response to a service group command. All resources designated as AutoStart=1 must be online for the service group to be considered online. (This attribute is unrelated to AutoStart attributes for service groups.)
- Enabled. If the resource is set to Enabled, the agent for the resource's type manages the resource. The default is 1 for resources defined in the configuration file main.cf, 0 for resources added on the command line.

Note: The addition of resources on the command line requires several steps, and the agent must be prevented from managing the resource until the steps are completed. For resources defined in the configuration file, the steps are completed before the agent is started.

Adding resources

This topic describes how to add resources to a service group or remove resources from a service group.

To add a resource

◆ Type the following command:

```
hares -add resource resource_type service_group
```

The resource name must be unique throughout the cluster. The resource type must be defined in the configuration language. The resource belongs to the group *service_group*.

Deleting resources

This topic describes how to delete resources from a service group.

To delete a resource

- Type the following command:

  ```
  hares -delete resource
  ```

  VCS does not delete online resources. However, you can enable deletion of online resources by changing the value of the DeleteOnlineResources attribute.

  See "Cluster attributes" on page 760.

  To delete a resource forcibly, use the -force option, which takes the resoure offline irrespective of the value of the DeleteOnlineResources attribute.

  ```
  hares -delete -force resource
  ```

Adding, deleting, and modifying resource attributes

Resource names must be unique throughout the cluster and you cannot modify resource attributes defined by the system, such as the resource state.

To modify a new resource

- Type the following command:

  ```
  hares -modify resource attribute value
  ```

  ```
  hares -modify resource attribute value
  [-sys system] [-wait [-time waittime]]
  ```

  The variable *value* depends on the type of attribute being created.

To set a new resource's Enabled attribute to 1

- Type the following command:

  ```
  hares -modify resourceA Enabled 1
  ```

  The agent managing the resource is started on a system when its Enabled attribute is set to 1 on that system. Specifically, the VCS engine begins to monitor the resource for faults. Agent monitoring is disabled if the Enabled attribute is reset to 0.

To add a resource attribute

◆ Type the following command:

```
haattr -add resource_type attribute
[value] [dimension] [default ...]
```

The variable *value* is a -string (default), -integer, or -boolean.

The variable *dimension* is -scalar (default), -keylist, -assoc, or -vector.

The variable *default* is the default value of the attribute and must be compatible with the *value* and *dimension*. Note that this may include more than one item, as indicated by ellipses (. . .).

To delete a resource attribute

◆ Type the following command:

```
haattr -delete resource_type attribute
```

To add a static resource attribute

◆ Type the following command:

```
haattr -add -static resource_type static_attribute
[value] [dimension] [default ...]
```

To delete a static resource attribute

◆ Type the following command:

```
haattr -delete -static resource_type static_attribute
```

To add a temporary resource attribute

◆ Type the following command:

```
haattr -add -temp resource_type attribute
[value] [dimension] [default ...]
```

To delete a temporary resource attribute

◆ Type the following command:

```
haattr -delete -temp resource_type attribute
```

To modify the default value of a resource attribute

◆ Type the following command:

```
haattr -default resource_type attribute new_value ...
```

The variable *new_value* refers to the attribute's new default value.

Defining attributes as local

Localizing an attribute means that the attribute has a per-system value for each system listed in the group's SystemList. These attributes are localized on a per-resource basis. For example, to localize the attribute *attribute_name* for *resource* only, type:

```
hares -local resource attribute_name
```

Note that global attributes cannot be modified with the `hares -local` command.

Table 5-5 lists the commands to be used to localize attributes depending on their dimension.

Table 5-5 Making VCS attributes local

Dimension	Task and Command
scalar	Replace a value: `-modify [object] attribute_name value [-sys system]`

vector	■ Replace list of values:
	`-modify [object] attribute_name value [-sys system]`
	■ Add list of values to existing list:
	`-modify [object] attribute_name -add value [-sys system]`
	■ Update list with user-supplied values:
	`-modify [object] attribute_name -update entry_value ... [-sys system]`
	■ Delete user-supplied values in list (if the list has multiple occurrences of the same value, then all the occurrences of the value is deleted):
	`-modify [object] attribute_name -delete <key> ... [-sys system]`
	■ Delete all values in list:
	`-modify [object] attribute_name -delete -keys [-sys system]`

Table 5-5 Making VCS attributes local *(continued)*

Dimension	Task and Command

keylist	■ Replace list of keys (duplicate keys not allowed):
	`-modify [object] attribute_name value ... [-sys system]`
	■ Add keys to list (duplicate keys not allowed):
	`-modify [object] attribute_name -add value ... [-sys system]`
	■ Delete user-supplied keys from list:
	`-modify [object] attribute_name -delete key ... [-sys system]`
	■ Delete all keys from list:
	`-modify [object] attribute_name -delete -keys [-sys system]`

association	■ Replace list of key-value pairs (duplicate keys not allowed):
	`-modify [object] attribute_name value ... [-sys system]`
	■ Add user-supplied list of key-value pairs to existing list (duplicate keys not allowed):
	`-modify [object] attribute_name -add value ...[-sys system]`
	■ Replace value of each key with user-supplied value:
	`-modify [object] attribute_name -update key value ... [-sys system]`
	■ Delete a key-value pair identified by user-supplied key:
	`-modify [object] attribute_name -delete key ... [-sys system]`
	■ Delete all key-value pairs from association:
	`-modify [object] attribute_name -delete -keys [-sys system]`
	Note: If multiple values are specified and if one is invalid, VCS returns an error for the invalid value, but continues to process the others. In the following example, if sysb is part of the attribute SystemList, but sysa is not, sysb is deleted and an error message is sent to the log regarding sysa.
	`hagrp -modify group1 SystemList -delete sysa sysb [-sys system]`

Defining attributes as global

Use the `hares -global` command to set a cluster-wide value for an attribute.

To set an attribute's value on all systems

◆ Type the following command:

```
# hares -global resource attribute
     value ... | key... | {key value}...
```

Enabling and disabling intelligent resource monitoring for agents manually

Review the following procedures to enable or disable intelligent resource monitoring manually. The intelligent resource monitoring feature is enabled by default. The IMF resource type attribute determines whether an IMF-aware agent must perform intelligent resource monitoring.

See "About resource monitoring" on page 38.

See "Resource type attributes" on page 703.

To enable intelligent resource monitoring

1. Make the VCS configuration writable.

```
# haconf -makerw
```

2. Run the following command to enable intelligent resource monitoring.

   - To enable intelligent monitoring of offline resources:

```
# hatype -modify resource_type IMF -update Mode 1
```

   - To enable intelligent monitoring of online resources:

```
# hatype -modify resource_type IMF -update Mode 2
```

   - To enable intelligent monitoring of both online and offline resources:

```
# hatype -modify resource_type IMF -update Mode 3
```

3. If required, change the values of the MonitorFreq key and the RegisterRetryLimit key of the IMF attribute.

   See the *Cluster Server Bundled Agents Reference Guide* for agent-specific recommendations to set these attributes.

4 Save the VCS configuration.

```
# haconf -dump -makero
```

5 Make sure that the AMF kernel driver is configured on all nodes in the cluster.

```
/opt/VRTSamf/bin/amf status
```

If the AMF kernel driver is configured, the output resembles:

```
AMF: Module loaded and configured
```

Configure the AMF driver if the command output returns that the AMF driver is not loaded or not configured.

See "Administering the AMF kernel driver" on page 109.

6 Restart the agent. Run the following commands on each node.

```
# haagent -stop agent_name -force -sys sys_name
# haagent -start agent_name -sys sys_name
```

To disable intelligent resource monitoring

1 Make the VCS configuration writable.

```
# haconf -makerw
```

2 To disable intelligent resource monitoring for all the resources of a certain type, run the following command:

```
# hatype -modify resource_type IMF -update Mode 0
```

3 To disable intelligent resource monitoring for a specific resource, run the following command:

```
# hares -override resource_name IMF
# hares -modify resource_name IMF -update Mode 0
```

4 Save the VCS configuration.

```
# haconf -dump -makero
```

Note: VCS provides haimfconfig script to enable or disable the IMF functionality for agents. You can use the script with VCS in running or stopped state. Use the script to enable or disable IMF for the IMF-aware bundled agents, enterprise agents, and custom agents.

See "Enabling and disabling IMF for agents by using script" on page 154.

Enabling and disabling IMF for agents by using script

VCS provides a script to enable and disable IMF for agents and for the AMF module. You can use the script when VCS is running or when VCS is stopped. Use the script to enable or disable IMF for the IMF-aware bundled agents, enterprise agents, and custom agents.

You must run the script once on each node of the cluster by using the same command. For example, if you enabled IMF for all IMF-aware agents on one node, you must repeat the action on the other cluster nodes too.

The following procedures describe how to enable or disable IMF for agents by using the script.

See "About resource monitoring" on page 38.

See "Resource type attributes" on page 703.

Enabling and disabling IMF for all IMF-aware agents

Enable or disable IMF for all IMF-aware agents as follows:

To enable IMF for all IMF-aware agents

- Run the following command:

```
haimfconfig  -enable
```

To disable IMF for all IMF-aware agents

- Run the following command:

```
haimfconfig  -disable
```

Note: The preceding `haimfconfig` commands cover all IMF-aware agents.

Enabling and disabling IMF for a set of agents

Enable or disable IMF for a set of agents as follows:

To enable IMF for a set of agents

◆ Run the following command:

```
haimfconfig  -enable -agent agent(s)
```

This command enables IMF for the specified agents. It also configures and loads the AMF module on the system if the module is not already loaded. If the agent is a custom agent, the command prompts you for the `Mode` and `MonitorFreq` values if Mode value is not configured properly.

- If VCS is running, this command might require to dump the configuration and to restart the agents. The command prompts whether you want to continue. If you choose **Yes**, it restarts the agents that are running and dumps the configuration. It also sets the `Mode` value (for all specified agents) and `MonitorFreq` value (for custom agents only) appropriately to enable IMF. For the agents that are not running, it changes the `Mode` value (for all specified agents) and `MonitorFreq` value (for all specified custom agents only) so as to enable IMF for the agents when they start.

  Note: The command prompts you whether you want to make the configuration changes persistent. If you choose **No**, the command exits. If you choose **Yes**, it enables IMF and dumps the configuration by using the `haconf -dump -makero` command.

- If VCS is not running, changes to the `Mode` value (for all specified agents) and `MonitorFreq` value (for all specified custom agents only) need to be made by modifying the VCS configuration files. Before the command makes any changes to configuration files, it prompts you for a confirmation. If you choose **Yes**, it modifies the VCS configuration files. IMF gets enabled for the specified agent when VCS starts.

Example

```
haimfconfig  -enable -agent Mount Application
```

The command enables IMF for the Mount agent and the Application agent.

To disable IMF for a set of agents

◆ Run the following command:

```
haimfconfig  -disable -agent agent(s)
```

This command disables IMF for specified agents by changing the `Mode` value to 0 for each agent and for all resources that had overridden the `Mode` values.

- If VCS is running, the command changes the `Mode` value of the agents and the overridden `Mode` values of all resources of these agents to 0.

  Note: The command prompts you whether you want to make the configuration changes persistent. If you choose **No**, the command exits. If you choose **Yes**, it enables IMF and dumps the configuration by using the `haconf -dump -makero` command.

- If VCS is not running, any change to the `Mode` value needs to be made by modifying the VCS configuration file. Before it makes any changes to configuration files, the command prompts you for a confirmation. If you choose **Yes**, it sets the `Mode` value to 0 in the configuration files.

Example

```
haimfconfig  -disable  -agent Mount Application
```

The command disables IMF for the Mount agent and Application agent.

Enabling and disabling AMF on a system

Enable or disable AMF as follows:

To enable AMF on a system

◆ Run the following command:

```
haimfconfig  -enable -amf
```

This command sets the value of `AMF_START` to 1 in the AMF configuration file. It also configures and loads the AMF module on the system.

To disable AMF on a system

◆ Run the following command:

```
haimfconfig  -disable -amf
```

This command unconfigures and unloads the AMF module on the system if AMF is configured and loaded. It also sets the value of `AMF_START` to 0 in the AMF configuration file.

Note: AMF is not directly unconfigured by this command if the agent is registered with AMF. The script prompts you if you want to disable AMF for all agents forcefully before it unconfigures AMF.

Viewing the configuration changes made by the script

View a summary of the steps performed by the command as follows:

To view the changes made when the script enables IMF for an agent

◆ Run the following command:

```
haimfconfig -validate  -enable -agent agent
```

To view the changes made when the script disables IMF for an agent

◆ Run the following command:

```
haimfconfig -validate  -disable -agent agent
```

Note: The script also generates the `$VCS_LOG/log/haimfconfig_A.log` file. This log contains information about all the commands that the script runs. It also logs the command outputs and the return values. The last ten logs are backed up.

Displaying the current IMF status of agents

To display current IMF status

◆ Run the following command:

```
haimfconfig -display
```

The status of agents can be one of the following:

- ENABLED: IMF is enabled for the agent.
- DISABLED: IMF is disabled for the agent.
- ENABLED|PARTIAL: IMF is partially enabled for some resources of the agent. The possible reasons are:
  - The agent has proper Mode value set at type level and improper mode value set at resource level.
  - The agent has proper Mode value set at resource level and improper mode value set at type level.

Examples:

If IMF is disabled for Mount agent (when Mode is set to 0) and enabled for rest of the installed IMF-aware agents:

```
haimfconfig -display
```

```
#Agent          STATUS

Application     ENABLED

Mount           DISABLED

Process         ENABLED

DiskGroup       ENABLED
```

If IMF is disabled for Mount agent (when VCS is running, agent is running and not registered with AMF module) and enabled for rest of the installed IMF-aware agents:

```
haimfconfig -display

#Agent          STATUS

Application     ENABLED

Mount           DISABLED

Process         ENABLED

DiskGroup       ENABLED
```

If IMF is disabled for all installed IMF-aware agents (when AMF module is not loaded):

```
haimfconfig -display

#Agent          STATUS

Application     DISABLED

Mount           DISABLED

Process         DISABLED

DiskGroup       DISABLED
```

If IMF is partially enabled for Mount agent (Mode is set to 3 at type level and to 0 at resource level for some resources) and enabled fully for rest of the installed IMF-aware agents.

```
haimfconfig -display

#Agent          STATUS

Application     ENABLED

Mount           ENABLED|PARTIAL
```

```
Process         ENABLED

DiskGroup       ENABLED
```

If IMF is partially enabled for Mount agent (Mode is set to 0 at type level and to 3 at resource level for some resources) and enabled fully for rest of the installed IMF-aware agents:

```
haimfconfig -display

#Agent          STATUS

Application     ENABLED

Mount           ENABLED|PARTIAL

Process         ENABLED

DiskGroup       ENABLED
```

Linking and unlinking resources

Link resources to specify a dependency between them. A resource can have an unlimited number of parents and children. When you link resources, the parent cannot be a resource whose Operations attribute is equal to None or OnOnly. Specifically, these are resources that cannot be brought online or taken offline by an agent (None), or can only be brought online by an agent (OnOnly).

Loop cycles are automatically prohibited by the VCS engine. You cannot specify a resource link between resources of different service groups.

To link resources

◆ Enter the following command:

```
hares -link parent_resource child_resource
```

The variable *parent_resource* depends on *child_resource* being online before going online itself. Conversely, *parent_resource* go offline before *child_resource* goes offline.

For example, a NIC resource must be available before an IP resource can go online, so for resources IP1 of type IP and NIC1 of type NIC, specify the dependency as:

```
hares -link IP1 NIC1
```

To unlink resources

- Enter the following command:

```
hares -unlink parent_resource child_resource
```

Configuring atleast resource dependency

In an atleast resource dependency, the parent resource depends on a set of child resources. The parent resource is brought online or can remain online only if the minimum number of child resources in this resource set is online.

Enter the following command to set atleast resource dependency. This command assumes that there are five child resources. If there are more child resources, enter the resource names without space.

```
hares -link parent_resource
child_res1,child_res2,child_res3,child_res4,child_res5 -min
minimum_resources
```

The following is an example:

```
hares -link res1 res2,res3,res4,res5,res6 -min 2
```

In the above example, the parent resource (res1) depends on the child resources (res2, res3, res4, res5, and res6). The parent resource can be brought online only when two or more resources come online and the parent resource can remain online only until two or more child resources are online.

Bringing resources online

This topic describes how to bring a resource online.

To bring a resource online

- Type the following command:

```
hares -online resource -sys system
```

Taking resources offline

This topic describes how to take a resource offline.

To take a resource offline

- Type the following command:

  ```
  hares -offline [-ignoreparent|parentprop] resource -sys system
  ```

  The option `-ignoreparent` enables a resource to be taken offline even if its parent resources in the service group are online. This option does not work if taking the resources offline violates the group dependency.

To take a resource and its parent resources offline

- Type the following command:

  ```
  hares -offline -parentprop resource -sys system
  ```

  The command stops all parent resources in order before taking the specific resource offline.

To take a resource offline and propagate the command to its children

- Type the following command:

  ```
  hares -offprop [-ignoreparent] resource -sys system
  ```

  As in the above command, the option `-ignoreparent` enables a resource to be taken offline even if its parent resources in the service group are online. This option does not work if taking the resources offline violates the group dependency.

Probing a resource

This topic describes how to probe a resource.

To prompt an agent to monitor a resource on a system

- Type the following command:

  ```
  hares -probe resource -sys system
  ```

  Though the command may return immediately, the monitoring process may not be completed by the time the command returns.

Clearing a resource

This topic describes how to clear a resource.

To clear a resource

- Type the following command:

  Initiate a state change from RESOURCE_FAULTED to RESOURCE_OFFLINE:

  ```
  hares -clear resource [-sys system]
  ```

  Clearing a resource initiates the online process previously blocked while waiting for the resource to become clear. If *system* is not specified, the fault is cleared on each system in the service group's SystemList attribute.

  See "Clearing faulted resources in a service group" on page 143.

  This command also clears the resource's parents. Persistent resources whose static attribute Operations is defined as None cannot be cleared with this command and must be physically attended to, such as replacing a raw disk. The agent then updates the status automatically.

About administering resource types

Administration of resource types includes the following activities:

Adding, deleting, and modifying resource types

After you create a resource type, use the `haattr` command to add its attributes. By default, resource type information is stored in the `types.cf` configuration file.

To add a resource type

- Type the following command:

  ```
  hatype -add resource_type
  ```

To delete a resource type

- Type the following command:

  ```
  hatype -delete resource_type
  ```

  You must delete all resources of the type before deleting the resource type.

To add or modify resource types in main.cf without shutting down VCS

- Type the following command:

  hatype -modify *resource_type* SourceFile "./*resource_type*.cf"

  The information regarding *resource_type* is stored in the file `config/resource_type.cf`.

  An include line for `resource_type.cf` is added to the `main.cf` file.

  Note: Make sure that the path to the SourceFile exists on all nodes before you run this command.

To set the value of static resource type attributes

- Type the following command for a scalar attribute:

  hatype -modify *resource_type attribute value*

  For more information, type:

  hatype -help -modify

Overriding resource type static attributes

You can override some resource type static attributes and assign them resource-specific values. When a static attribute is overriden and the configuration is saved, the main.cf file includes a line in the resource definition for the static attribute and its overriden value.

To override a type's static attribute

- Type the following command:

  hares -override *resource static_attribute*

To restore default settings to a type's static attribute

- Type the following command:

  hares -undo_override *resource static_attribute*

About initializing resource type scheduling and priority attributes

The following configuration shows how to initialize resource type scheduling and priority attributes through configuration files. The example shows attributes of a FileOnOff resource.

```
type FileOnOff (
    static str AgentClass = RT
    static str AgentPriority = 10
    static str ScriptClass = RT
    static str ScriptPriority = 40
    static str ArgList[] = { PathName }
    str PathName
)
```

Setting scheduling and priority attributes

This topic describes how to set scheduling and priority attributes.

Note: For attributes AgentClass and AgentPriority, changes are effective immediately. For ScriptClass and ScriptPriority, changes become effective for scripts fired after the execution of the `hatype` command.

To update the AgentClass

◆ Type the following command:

hatype -modify *resource_type* AgentClass *value*

For example, to set the AgentClass attribute of the FileOnOff resource to RealTime, type:

hatype -modify FileOnOff AgentClass "RT"

To update the AgentPriority

◆ Type the following command:

hatype -modify *resource_type* AgentPriority *value*

For example, to set the AgentPriority attribute of the FileOnOff resource to 10, type:

hatype -modify FileOnOff AgentPriority "10"

To update the ScriptClass

◆ Type the following command:

```
hatype -modify resource_type ScriptClass value
```

For example, to set the ScriptClass of the FileOnOff resource to RealTime, type:

```
hatype -modify FileOnOff ScriptClass "RT"
```

To update the ScriptPriority

◆ Type the following command:

```
hatype -modify resource_type ScriptPriority value
```

For example, to set the ScriptClass of the FileOnOff resource to 40, type:

```
hatype -modify FileOnOff ScriptPriority "40"
```

Administering systems

Administration of systems includes tasks such as modifying system attributes, freezing or unfreezing systems, and running commands.

To modify a system's attributes

◆ Type the following command:

```
hasys -modify modify_options
```

Some attributes are internal to VCS and cannot be modified.

See "About the -modify option" on page 95.

To display the value of a system's node ID as defined in the llttab file

◆ Type the following command to display the value of a system's node ID as defined in the following file:

/etc/llttab

```
hasys -nodeid [node_ID]
```

To freeze a system (prevent groups from being brought online or switched on the system)

- Type the following command:

```
hasys -freeze [-persistent] [-evacuate] system
```

-persistent	Enables the freeze to be "remembered" when the cluster is rebooted. Note that the cluster configuration must be in read/write mode and must be saved to disk (dumped) to enable the freeze to be remembered.
-evacuate	Fails over the system's active service groups to another system in the cluster before the freeze is enabled.

To unfreeze a frozen system (reenable online and switch of service groups)

- Type the following command:

```
hasys -unfreeze [-persistent] system
```

To run a command on any system in a cluster

- Type the following command:

```
hacli -cmd command [-sys | -server system(s)]
```

Issues a command to be executed on the specified system(s). VCS must be running on the systems.

The use of the hacli command requires setting HacliUserLevel to at least COMMANDROOT. By default, the HacliUserLevel setting is NONE.

If the users do not want the root user on system A to enjoy root privileges on another system B, HacliUserLevel should remain set to NONE (the default) on system B.

You can specify multiple systems separated by a single space as arguments to the option -sys. If no system is specified, command runs on all systems in cluster with VCS in a RUNNING state. The command argument must be entered within double quotes if command includes any delimiters or options.

About administering clusters

Administration of clusters includes the following activities:

Configuring and unconfiguring the cluster UUID value

When you install VCS using the installer, the installer generates the cluster UUID (Universally Unique ID) value. This value is the same across all the nodes in the cluster.

You can use the uuidconfig utility to display, copy, configure, and unconfigure the cluster UUID on the cluster nodes.

Make sure you have ssh or rsh communication set up between the systems. The utility uses ssh by default.

To display the cluster UUID value on the VCS nodes

- Run the following command to display the cluster UUID value:
  - For specific nodes:

    ```
    # /opt/VRTSvcs/bin/uuidconfig.pl [-rsh] -clus -display sys1
    [sys2 sys3...]
    ```

  - For all nodes that are specified in the `/etc/llthosts` file:

    ```
    # /opt/VRTSvcs/bin/uuidconfig.pl [-rsh] -clus -display
    -use_llthost
    ```

To configure cluster UUID on the VCS nodes

- Run the following command to configure the cluster UUID value:
  - For specific nodes:

    ```
    # /opt/VRTSvcs/bin/uuidconfig.pl [-rsh] -clus -configure
    sys1 [sys2 sys3...]
    ```

  - For all nodes that are specified in the `/etc/llthosts` file:

    ```
    # /opt/VRTSvcs/bin/uuidconfig.pl -clus -configure
    -use_llthost
    ```

    The utility configures the cluster UUID on the cluster nodes based on whether a cluster UUID exists on any of the VCS nodes:

  - If no cluster UUID exists or if the cluster UUID is different on the cluster nodes, then the utility does the following:
    - Generates a new cluster UUID using the `/opt/VRTSvcs/bin/osuuid`.
    - Creates the `/etc/vx/.uuids/clusuuid` file where the utility stores the cluster UUID.

- Configures the cluster UUID on all nodes in the cluster.

- If a cluster UUID exists and if the UUID is same on all the nodes, then the utility retains the UUID.
  Use the `-force` option to discard the existing cluster UUID and create new cluster UUID.
- If some nodes in the cluster have cluster UUID and if the UUID is the same, then the utility configures the existing UUID on the remaining nodes.

To unconfigure cluster UUID on the VCS nodes

◆ Run the following command to unconfigure the cluster UUID value:

- For specific nodes:

```
# /opt/VRTSvcs/bin/uuidconfig.pl [-rsh] -clus
-unconfigure sys1 [sys2 sys3...]
```

- For all nodes that are specified in the `/etc/llthosts` file:

```
# /opt/VRTSvcs/bin/uuidconfig.pl [-rsh] -clus
-unconfigure -use_llthost
```

The utility removes the `/etc/vx/.uuids/clusuuid` file from the nodes.

To copy the cluster UUID from one node to other nodes

◆ Run the following command to copy the cluster UUID value:

```
# /opt/VRTSvcs/bin/uuidconfig.pl [-rsh] -clus -copy
-from_sys sys -to_sys sys1 sys2 [sys3...]
```

The utility copies the cluster UUID from a system that is specified using the`-from_sys` option to all the systems that are specified using the `-to_sys` option.

Retrieving version information

This topic describes how to retrieve information about the version of VCS running on the system.

To retrieve information about the VCS version on the system

1 Run one of the following commands to retrieve information about the engine version, the join version, the build date, and the PSTAMP.

```
had -version
hastart -version
```

2 Run one of the following commands to retrieve information about the engine version.

```
had -v

hastart -v
```

Adding and removing systems

This topic provides an overview of tasks involved in adding and removing systems from a cluster.

For detailed instructions, see the *Cluster Server Installation Guide*.

To add a system to a cluster

1 Make sure that the system meets the hardware and software requirements for VCS.

2 Set up the private communication links from the new system.

3 Install VCS and required patches on the new system.

4 Add the VCS license key.

See "About installing a VCS license" on page 100.

5 Configure LLT and GAB to include the new system in the cluster membership.

6 Add the new system using the `hasys -add` command.

To remove a node from a cluster

1 Make a backup copy of the current configuration file, main.cf.

2 Switch or remove any VCS service groups from the node. The node cannot be removed as long as it runs service groups on which other service groups depend.

3 Stop VCS on the node.

```
hastop -sys systemname
```

4 Delete the system from the SystemList of all service groups.

```
hagrp -modify groupname SystemList -delete systemname
```

5 Delete the node from the cluster.

```
hasys -delete systemname
```

6 Remove the entries for the node from the `/etc/llthosts` file on each remaining node.

7 Change the node count entry from the `/etc/gabtab` file on each remaining node.

8 Unconfigure GAB and LLT on the node leaving the cluster.

9 Remove VCS and other RPMs from the node.

10 Remove GAB and LLT configuration files from the node.

Changing ports for VCS

You can change some of the default ports for VCS and its components.

The following is a list of changes that you can make and information concerning ports in a VCS environment:

- Changing VCS's default port.
  Add an entry for a VCS service name In `/etc/services` file, for example:

  ```
  vcs-app 3333/tcp # Veritas Cluster Server
  ```

  Where 3333 in the example is the port number where you want to run VCS. When the engine starts, it listens on the port that you configured above (3333) for the service. You need to modify the port to the /etc/services file on all the nodes of the cluster.

- You do not need to make changes for agents or HA commands. Agents and HA commands use locally present UDS sockets to connect to the engine, not TCP/IP connections.

- You do not need to make changes for HA commands that you execute to talk to a remotely running VCS engine (HAD), using the facilities that the VCS_HOST environment variable provides. You do not need to change these settings because the HA command queries the `/etc/services` file and connects to the appropriate port.

- For the Java Console GUI, you can specify the port number that you want the GUI to connect to while logging into the GUI. You have to specify the port number that is configured in the `/etc/services` file (for example 3333 above).

To change the default port

1. Stop VCS.
2. Add an entry service name vcs-app in `/etc/services`.

```
vcs-app 3333/tcp # Veritas Cluster Server
```

3. You need to modify the port to the `/etc/services` file on all the nodes of the cluster.
4. Restart VCS.
5. Check the port.

```
# netstat -an|grep 3333

*.3333 *.* 0 0 49152 0 LISTEN
*.3333 *.* 0 0 49152 0 LISTEN
```

6. Use the Java Console to connect to VCS through port 3333.

Setting cluster attributes from the command line

This topic describes how to set cluster attributes from the command line.

Note: For the attributes EngineClass and EnginePriority, changes are effective immediately. For ProcessClass and ProcessPriority changes become effective only for processes fired after the execution of the `haclus` command.

To modify a cluster attribute

◆ Type the following command

```
# haclus [-help [-modify]]
```

To update the EngineClass

◆ Type the following command:

```
# haclus -modify EngineClass value
```

For example, to set the EngineClass attribute to RealTime::

```
# haclus -modify EngineClass "RT"
```

To update the EnginePriority

◆ Type the following command:

```
# haclus -modify EnginePriority value
```

For example, to set the EnginePriority to 20::

```
# haclus -modify EnginePriority "20"
```

To update the ProcessClass

◆ Type the following command:

```
# haclus -modify ProcessClass value
```

For example, to set the ProcessClass to TimeSharing:

```
# haclus -modify ProcessClass "TS"
```

To update the ProcessPriority

◆ Type the following command:

```
# haclus -modify ProcessPriority value
```

For example, to set the ProcessPriority to 40:

```
# haclus -modify ProcessPriority "40"
```

About initializing cluster attributes in the configuration file

You may assign values for cluster attributes while you configure the cluster.

See “Cluster attributes” on page 760.

Review the following sample configuration:

```
cluster vcs-india (
    EngineClass = "RT"
```

```
EnginePriority = "20"
ProcessClass = "TS"
ProcessPriority = "40"
)
```

Enabling and disabling secure mode for the cluster

This topic describes how to enable and disable secure mode for your cluster.

To enable secure mode in a VCS cluster

1. `# hasys -state`

   The output must show the SysState value as RUNNING.

2. You can enable secure mode or secure mode with FIPS.

   To enable secure mode, start the installer program with the `-security` option.

   ```
   # /opt/VRTS/install/installer -security
   ```

   To enable secure mode with FIPS, start the installer program with the `-security -fips` option.

   ```
   # /opt/VRTS/install/installer -security -fips
   ```

   If you already have secure mode enabled and need to move to secure mode with FIPS, complete the steps in the following procedure.

   See "Migrating from secure mode to secure mode with FIPS" on page 174.

   The installer displays the directory where the logs are created.

3. Review the output as the installer verifies whether VCS configuration files exist.

   The installer also verifies that VCS is running on all systems in the cluster.

4. The installer checks whether the cluster is in secure mode or non-secure mode. If the cluster is in non-secure mode, the installer prompts whether you want to enable secure mode.

   ```
   Do you want to enable secure mode in this cluster? [y,n,q] (y) y
   ```

5. Review the output as the installer modifies the VCS configuration files to enable secure mode in the cluster, and restarts VCS.

To disable secure mode in a VCS cluster

1. Start the installer program with the `-security` option.

   ```
   # /opt/VRTS/install/installer -security
   ```

   To disable secure mode with FIPS, use the follwing command:

   ```
   # /opt/VRTS/install/installer -security -fips
   ```

   The installer displays the directory where the logs are created.

2. Review the output as the installer proceeds with a verification.

3. The installer checks whether the cluster is in secure mode or non-secure mode. If the cluster is in secure mode, the installer prompts whether you want to disable secure mode.

   ```
   Do you want to disable secure mode in this cluster? [y,n,q] (y) y
   ```

4. Review the output as the installer modifies the VCS configuration files to disable secure mode in the cluster, and restarts VCS.

Migrating from secure mode to secure mode with FIPS

To migrate from secure mode to secure mode with FIPS

1. Unconfigure security.

   ```
   # installer -security
   ```

2. Clear the credentials that exist in the following directories:

   - `/var/VRTSvcs/vcsauth`
   - `/var/VRTSat`
   - `/var/VRTSat_lhc`
   - `.VRTSat` from the home directories of all the VCS users

3. Configure security in FIPS mode.

   ```
   # installer -security -fips
   ```

About cluster formation with different versions of the VCS engine

The Cluster Server component (VCS) of InfoScale Availability requires all the nodes in a cluster to run the same version of the VCS engine, also known as the high

availability daemon or HAD. On each node, HAD operates as a replicated state machine (RSM), which means that all the nodes that form the cluster have a synchronized state of the cluster configuration. InfoScale supports a rolling upgrade of the nodes in a cluster but the process involves some downtime of the VCS engine. VCS upgrade is therefore a two-phased process: in the first phase, the installer upgrades the kernel filesets. In the second phase, the installer upgrades VCS and the VCS agent filesets.

Starting with version 7.4.2, InfoScale allows for the formation of clusters with nodes that run different versions of the VCS engine. It does so by providing a framework that lets you:

- Add nodes with newer versions of the VCS engine to an existing cluster
- Upgrade InfoScale on one node at a time without the need for a cluster-wide HA downtime
- Manage the simultaneous usage of multiple VCS engine versions by using the cluster protocol number

The concept of a cluster protocol number is introduced, which determines whether a node that runs a particular VCS version can become part of a cluster. An internal, system-level attribute, `SupportedProtocol`, is provided to store the supported cluster protocol number.

To identify the cluster protocol number that is supported by each node in a cluster, run the following commend:

```
# haclus -version -info
```

To identify the protocol number that is currently used by the cluster, run the following command

```
# haclus -value ProtocolNumber
```

When a cluster is formed with different versions of the VCS engine, the lowest supported cluster protocol number is used by all the nodes to maintain the RSM. The use of a supported cluster protocol number ensures consistent behaviour across cluster nodes during operations like startup, snapshot exchange, group failover, and so on. If the VCS engine is not compatible with the `SupportedProtocol` of the new nodes that are being added to the cluster, the cluster does not accept the new node and maintains the status quo of the RSM.

Note: This framework supports cluster formation with different versions of VCS only if the nodes run InfoScale 7.4.2 or later.

For a node to join an existing cluster, it should run the VCS engine with the minimum supported cluster protocol number.

Using the -wait option in scripts that use VCS commands

The `-wait` option is for use in the scripts that use VCS commands to wait till an attribute value changes to the specified value. The option blocks the VCS command until the value of the specified attribute is changed or until the specified timeout expires. Specify the timeout in seconds.

The option can be used only with changes to scalar attributes.

The `-wait` option is supported with the following commands:

haclus

```
haclus -wait attribute value
[-clus cluster] [-time timeout]
```

Use the `-clus` option in a global cluster environment.

hagrp

```
hagrp -wait group attribute value
[-clus cluster] [-sys system] [-time timeout]
```

Use the `-sys` option when the scope of the attribute is local.

Use the `-clus` option in a global cluster environment.

hares

```
hares -wait resource attribute value
[-clus cluster] [-sys system] [-time timeout]
```

Use the `-sys` option when the scope of the attribute is local.

Use the `-clus` option in a global cluster environment.

hasys

```
hasys -wait system attribute value
[-clus cluster] [-time timeout]
```

Use the `-clus` option in a global cluster environment.

See the man pages associated with these commands for more information.

Running HA fire drills

The service group must be online when you run the HA fire drill.

See “About testing resource failover by using HA fire drills” on page 225.

To run HA fire drill for a specific resource

- Type the following command.

  ```
  # hares -action <resname> <vfdaction>.vfd -sys <sysname>
  ```

  The command runs the infrastructure check and verifies whether the system <*sysname*> has the required infrastructure to host the resource <*resname*>, should a failover require the resource to come online on the system. For the variable <*sysname*>, specify the name of a system on which the resource is offline. The variable <*vfdaction*> specifies the Action defined for the agent. The "HA fire drill checks" for a resource type are defined in the SupportedActions attribute for that resource and can be identified with the .vfd suffix.

To run HA fire drill for a service group

- Type the following command.

  ```
  # havfd <grpname> -sys <sysname>
  ```

  The command runs the infrastructure check and verifies whether the system <*sysname*> has the required infrastructure to host resources in the service group <*grpname*> should a failover require the service group to come online on the system. For the variable <*sysname*>, specify the name of a system on which the group is offline

To fix detected errors

- Type the following command.

  ```
  # hares -action <resname> <vfdaction>.vfd -actionargs fix -sys
  <sysname>
  ```

  The variable <*vfdaction*> represents the check that reported errors for the system <*sysname*>. The "HA fire drill checks" for a resource type are defined in the SupportedActions attribute for that resource and can be identified with the .vfd suffix.

About administering simulated clusters from the command line

VCS Simulator is a tool to assist you in building and simulating cluster configurations. With VCS Simulator you can predict service group behavior during cluster or system faults, view state transitions, and designate and fine-tune various configuration parameters. This tool is especially useful when you evaluate complex, multi-node configurations. It is convenient in that you can design a specific configuration without test clusters or changes to existing configurations.

You can also fine-tune values for attributes that govern the rules of failover, such as Load and Capacity in a simulated environment. VCS Simulator enables you to simulate various configurations and provides the information that you need to make the right choices. It also enables simulating global clusters.

See "About VCS Simulator" on page 227.

Chapter 6

Configuring applications and resources in VCS

This chapter includes the following topics:

- Configuring resources and applications
- VCS bundled agents for UNIX
- Configuring NFS service groups
- About configuring the RemoteGroup agent
- About configuring Samba service groups
- Configuring the Coordination Point agent
- About testing resource failover by using HA fire drills

Configuring resources and applications

Configuring resources and applications in VCS involves the following tasks:

- Create a service group that comprises all resources that are required for the application.
  To configure resources, you can use any of the supported components that are used for administering VCS.
  See " Components for administering VCS" on page 49.
- Add required resources to the service group and configure them.
  For example, to configure a database in VCS, you must configure resources for the database and for the underlying shared storage and network resources.
  Use appropriate agents to configure resources.
  See "VCS bundled agents for UNIX" on page 180.

Configuring a resource involves defining values for its attributes.

See the *Cluster Server Bundled Agents Reference Guide* for a description of the agents provided by VCS.

The resources must be logically grouped in a service group. When a resource faults, the entire service group fails over to another node.

- Assign dependencies between resources. For example, an IP resource depends on a NIC resource.
- Bring the service group online to make the resources available.

VCS bundled agents for UNIX

Bundled agents are categorized according to the type of resources they make highly available.

See "About Storage agents" on page 180.

See "About Network agents" on page 182.

See "About File share agents" on page 184.

See "About Services and Application agents" on page 185.

See "About VCS infrastructure and support agents" on page 186.

See "About Testing agents" on page 187.

About Storage agents

Storage agents monitor shared storage and make shared storage highly available. Storage includes shared disks, disk groups, volumes, and mounts. The DiskGroup agent supports both IMF-based monitoring and traditional poll-based monitoring.

See the *Cluster Server Bundled Agents Reference Guide* for a detailed description of the following agents.

Table 6-1 Storage agents and their description

Agent	Description
DiskGroup	Brings Veritas Volume Manager (VxVM) disk groups online and offline, monitors them, and make them highly available. The DiskGroup agent supports both IMF-based monitoring and traditional poll-based monitoring. DiskGroup resources can depend on DiskReservation resources, if Dynamic Multi-pathing is not configured in Symantec Volume Manager.

Table 6-1 Storage agents and their description *(continued)*

Agent	Description
DiskGroupSnap	Brings resources online and offline and monitors disk groups used for fire drill testing. The DiskGroupSnap agent enables you to verify the configuration integrity and data integrity in a Campus Cluster environment with VxVM stretch mirroring. The service group that contains the DiskGroupSnap agent resource has an offline local dependency on the application's service group. This is to ensure that the fire drill service group and the application service group are not online at the same site.
DiskReservation	Enables you to reserve and monitor all SCSI disks or a percentage of disks for a system. Such reservations prevent disk data corruption by restricting other nodes from accessing and writing to the reserved disks by giving exclusive access to system for a shared disk.
Volume	Makes Veritas Volume Manager(VxVM) Volumes highly available and enables you to bring the volumes online and offline, and monitor them. Volume resources depend on DiskGroup resources.
VolumeSet	Brings Veritas Volume Manager (VxVM) volume sets online and offline, and monitors them. Use the VolumeSet agent to make a volume set highly available. VolumeSet resources depend on DiskGroup resources.
LVMLogicalVolume	Brings resource online and offline, and monitors Logical Volume Manager (LVM2) logical volumes. You can use this agent to make logical volumes highly available and to monitor them. LVMLogicalVolume resources depend on LVMVolumeGroup resources.
LVMVolumeGroup	Brings Logical Volume Manager (LVM2) volume groups online and offline, monitors them, and make them highly available. No fixed dependencies exist for the LVMVolumeGroup agent. When you create a volume group on disks with a single path, Symantec recommends that you use the DiskReservation agent.
Mount	Brings resources online and offline, monitors file system or NFS client mount points, and make them highly available. The Mount agent supports both IMF-based monitoring and traditional poll-based monitoring. The Mount agent can be used with the DiskGroup, LVMVolumeGroup, LVMLogicalVolume, VolumeSet, and Volume agents to provide storage to an application. See "About resource monitoring" on page 38.

Table 6-1 Storage agents and their description *(continued)*

Agent	Description
AWS EBSVol	The EBSVol agent monitors the Amazon EBS volumes. It also attaches or detaches the Amazon EBS volumes to and from the EC2 instances respectively. This agent uses AWS CLI commands to determine the state of the Amazon EBS volumes. The EBS volume resource does not depend on any other resources.
AzureDisk	The AzureDisk agent brings online, takes offline, and monitors the managed and unmanaged Azure data disks. It attaches the managed and unmanaged data disks to a virtual machine of the same or different resource group. The AzureDisk agent uses Azure Python SDK to determine whether the Azure data disks are attached to the Azure virtual machines or not.
GoogleDisk	The GoogleDisk agent brings the GCP data disks online, monitors their status, and takes them offline. The GoogleDisk agent uses GCP Python SDK to determine whether the data disks are attached to the virtual machines or not.

About Network agents

Network agents monitor network resources and make your IP addresses and computer names highly available. Network agents support both IPv4 and IPv6 addresses. However, you cannot use the two types of addresses concurrently.

Table 6-2 shows the Network agents and their description.

Table 6-2 Network agents and their description

Agent	Description
NIC	Monitors a configured NIC. If a network link fails or if a problem arises with the NIC, the resource is marked FAULTED. You can use the NIC agent to make a single IP address on a single adapter highly available and monitor it. No child dependencies exist for this resource. For the NIC agent, VCS supports Linux bonded interface.
IP	Manages the process of configuring a virtual IP address and its subnet mask on an interface. You can use the IP agent to monitor a single IP address on a single adapter. The interface must be enabled with a physical (or administrative) base IP address before you can assign it a virtual IP address. For the IP agent, VCS supports Linux bonded interface.

Table 6-2 Network agents and their description *(continued)*

Agent	Description
MultiNICA	Represents a set of network interfaces and provides failover capabilities between them. You can use the MultiNICA agent to make IP addresses on multiple adapter systems highly available and to monitor them. If a MultiNICA resource changes its active device, the MultiNICA agent handles the shifting of IP addresses. For the MultiNICA agent, VCS supports Linux bonded interface.
IPMultiNIC	Manages a virtual IP address that is configured as an alias on one interface of a MultiNICA resource. If the interface faults, the IPMultiNIC agent works with the MultiNICA resource to fail over to a backup NIC. The IPMultiNIC agent depends upon the MultiNICA agent to select the most preferred NIC on the system. For the IPMultiNIC agent, VCS supports Linux bonded interface.
DNS	Updates and monitors the mapping of host names to IP addresses and canonical names (CNAME). The DNS agent performs these tasks for a DNS zone when it fails over nodes across subnets (a wide-area failover). Use the DNS agent when the failover source and target nodes are on different subnets. The DNS agent updates the name server and allows clients to connect to the failed over instance of the application service.
AWSIP	Manages the networking resources in an Amazon Web Services (AWS) cloud environment. The agent uses AWS CLIs to create IP resources in an AWS cloud environment. The agent does the following: ■ Assigns and configures private IP address ■ Associates public IP address and configures private IP along with elastic IP ■ Manages route table for overlay IP for failing over across subnets
Route53	Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. The Route53 agent updates and monitors the host name to IP address mapping. The agent does the mapping for the AWS Route 53 domain when failing over nodes across subnets. Use the Route53 agent if the Resource Records need to be dynamically added and deleted from the Route 53 domain during failover. The agent updates the name server with the new resource record mappings while failing over and allows the clients to connect to the failed over instance of the application.

Table 6-2 Network agents and their description *(continued)*

Agent	Description
AzureIP	The AzureIP agent manages the networking resources in an Azure environment. The agent uses Azure Python APIs to associate IP resources in a Azure VM. The agent does the following: ■ Gets the NIC details, create the IP configuration and associate and disassociates private IP address. ■ Associates and disassociates Public IP address with Private IP address. ■ Manages route table entries of overlay IP for failing over across subnets.
AzureDNSZone	The AzureDNSZone agent monitors and updates the host name to resource record mapping. This agent does the mapping for the Azure DNS domain when failing over nodes across subnets or regions. AzureDNSZone agent provides DNS-based traffic routing and failover. Use this agent if the resource records need to be dynamically added and deleted from the domain during failover. This agent updates the new resource record mappings while failing over and allows the clients to connect to the failed over instance of the application.
GoogleIP	The GoogleIP agent manages the Private IP and Overlay IP resources in a Google Cloud Platform (GCP) environment. The agent fetches the NIC details, associates the private IP with the NIC, and disassociates the private IP from the NIC. It manages the route table entries of the overlay IP for failover across subnets.

About File share agents

File Service agents make shared directories and subdirectories highly available.

See the *Cluster Server Bundled Agents Reference Guide* for a detailed description of these agents.

Table 6-3 shows the File share agents and their description.

Table 6-3 File share agents and their description

Agent	Description
NFS	Manages NFS daemons which process requests from NFS clients. The NFS Agent manages the rpc.nfsd/nfsd daemon and the rpc.mountd daemon on the NFS server. If NFSv4 support is enabled, it also manages the rpc.idmapd/nfsmapid daemon. Additionally, the NFS Agent also manages NFS lock and status daemons.
NFSRestart	Provides NFS lock recovery in case of application failover or server crash. The NFSRestart Agent also prevents potential NFS ACK storms by closing all TCP connections of NFS servers with the client before the service group failover occurs.
Share	Shares, unshares, and monitors a single local resource for exporting an NFS file system that is mounted by remote systems. Share resources depend on NFS. In an NFS service group, the IP family of resources depends on Share resources.
SambaServer	Starts, stops, and monitors the smbd process as a daemon. You can use the SambaServer agent to make an smbd daemon highly available or to monitor it. The smbd daemon provides Samba share services. The SambaServer agent, with SambaShare and NetBIOS agents, allows a system running a UNIX or UNIX-like operating system to provide services using the Microsoft network protocol. It has no dependent resource.
SambaShare	Adds, removes, and monitors a share by modifying the specified Samba configuration file. You can use the SambaShare agent to make a Samba Share highly available or to monitor it. SambaShare resources depend on SambaServer, NetBios, and Mount resources.
NetBIOS	Starts, stops, and monitors the nmbd daemon. You can use the NetBIOS agent to make the nmbd daemon highly available or to monitor it. The nmbd process broadcasts the NetBIOS name, or the name by which the Samba server is known in the network. The NetBios resource depends on the IP or the IPMultiNIC resource.

About Services and Application agents

Services and Application agents make Web sites, applications, and processes highly available.

See the *Cluster Server Bundled Agents Reference Guide* for a detailed description of these agents.

Table 6-4 shows the Services and Applications agents and their description.

Table 6-4 Services and Application agents and their description

Agent	Description
Apache	Brings an Apache Server online, takes it offline, and monitors its processes. Use the Apache Web server agent with other agents to make an Apache Web server highly available. This type of resource depends on IP and Mount resources. The Apache agent can detect when an Apache Web server is brought down gracefully by an administrator. When Apache is brought down gracefully, the agent does not trigger a resource fault even though Apache is down. The Apache agent supports both IMF-based monitoring for online monitoring and traditional poll-based monitoring.
Application	Brings applications online, takes them offline, and monitors their status. Use the Application agent to specify different executables for the online, offline, and monitor routines for different programs. The executables must exist locally on each node. You can use the Application agent to provide high availability for applications that do not have bundled agents, enterprise agents, or custom agents. This type of resource can depend on IP, IPMultiNIC, and Mount resources. The Application agent supports both IMF-based monitoring and traditional poll-based monitoring. See "About resource monitoring" on page 38.
Process	Starts, stops, and monitors a process that you specify. Use the Process agent to make a process highly available. This type of resource can depend on IP, IPMultiNIC, and Mount resources. The Process agent supports both IMF-based monitoring and traditional poll-based monitoring. See "About resource monitoring" on page 38.
ProcessOnOnly	Starts and monitors a process that you specify. Use the agent to make a process highly available. No child dependencies exist for this resource.
KVMGuest	Monitors Linux Kernel-based Virtual Machines (KVM guests), brings them online, and takes them offline. You can use the KVMGuest agent to make KVM guests highly available and to monitor them.
AzureAuth	AzureAuth agent authenticates the Azure subscription using the Service Principal credentials. This agent is a persistent resource that monitors the validity of Service Principal credentials.

About VCS infrastructure and support agents

VCS infrastructure and support agents monitor Veritas components and VCS objects.

See the *Cluster Server Bundled Agents Reference Guide* for a detailed description of these agents.

Table 6-5 shows the VCS infrastructure and support agents and their description.

Table 6-5 VCS infrastructure and support agents and their description

Agent	Description
NotifierMngr	Starts, stops, and monitors a notifier process, making it highly available. The notifier process manages the reception of messages from VCS and the delivery of those messages to SNMP consoles and SMTP servers. However, for the NotifierMngr agent to work in dual stack mode, the SNMP server and SMTP server must have both IP versions (IPv4 and IPv6) enabled on it. The NotifierMngr resource can depend on the NIC resource.
Proxy	Mirrors the state of another resource on a local or remote system. It provides a method to specify and modify one resource and have its state reflected by its proxies. You can use the Proxy agent to replicate the status of a resource. For example, a service group that uses the NFS resource can use a Proxy resource. The Proxy resource can point to the NFS resource in a separate parallel service group.
Phantom	Enables VCS to determine the state of parallel service groups that do not include OnOff resources. No dependencies exist for the Phantom resource.
RemoteGroup	Establishes dependencies between applications that are configured on different VCS clusters. For example, if you configure an Apache resource in a local cluster and a MySQL resource in a remote cluster, the Apache resource depends on the MySQL resource. You can use the RemoteGroup agent to establish the dependency between two resources and to monitor or manage a service group that exists in a remote cluster.
CoordPoint	Monitors the I/O fencing coordination points.

About Testing agents

Testing agents provide high availability for program support resources that are useful for testing VCS functionality.

See the *Cluster Server Bundled Agents Reference Guide* for a detailed description of these agents.

Table 6-6 shows VCS Testing agents and their description.

Table 6-6 Testing agents and their description

Agent	Description
ElifNone	Monitors a file and checks for the file's absence. You can use the ElifNone agent to test service group behavior. No dependencies exist for the ElifNone resource.
FileNone	Monitors a file and checks for the file's existence. You can use the FileNone agent to test service group behavior. No dependencies exist for the FileNone resource.
FileOnOff	Creates, removes, and monitors files. You can use the FileOnOff agent to test service group behavior. No dependencies exist for the FileOnOff resource.
FileOnOnly	Creates and monitors files but does not remove files. You can use the FileOnOnly agent to test service group behavior. No dependencies exist for the FileOnOnly resource.

Configuring NFS service groups

This section describes the features of NFS and the methods of configuring NFS.

About NFS

Network File System (NFS) allows network users to access shared files stored on an NFS server. NFS lets users manipulate shared files transparently as if the files were on a local disk.

NFS terminology

Key terms used in NFS operations include:

NFS Server — The computer that makes the local file system accessible to users on the network.

NFS Client — The computer which accesses the file system that is made available by the NFS server.

rpc.mountd — A daemon that runs on NFS servers. It handles initial requests from NFS clients. NFS clients use the mount command to make requests.

rpc.nfsd/nfsd — A daemon that runs on NFS servers. It is formed of stateless kernel threads that handle most of the NFS requests (including NFS read/write requests) from NFS clients.

rpc.lockd/lockd	A daemon that runs on NFS servers and NFS clients. On the server side, it receives lock requests from the NFS client and passes the requests to the kernel-based nfsd. On the client side, it forwards the NFS lock requests from users to the rpc.lockd/lockd on the NFS server.
rpc.statd/statd	A daemon that runs on NFS servers and NFS clients. On the server side, it maintains the information about NFS clients that have locks and NFS clients that are trying for locks. If the NFS server recovers after a crash, rpc.statd/statd notifies all the NFS clients to reclaim locks that they had before the server crash. This process is called NFS lock recovery. On the client side, it maintains a list of servers for which users requested locks. It also processes notification from the server statd/rpc.statd. If the NFS client recovers after a crash, rpc.statd/statd notifies the NFS server to stop monitoring the client since the client restarted.
rpc.idmapd/nfsmapid	A userland daemon that maps the NFSv4 username and group to the local username and group of the system. This daemon is specific to NFSv4.
rpc.svcgssd	A userland daemon runs on NFS server. It provides rpcsec_gss security to the RPC daemons.
NFSv4	The latest version of NFS. It is a stateful protocol. NFXv4 requires only the rpc.nfsd/nfsd daemon to be running on the system. It does not require the associate daemons rpc.mountd, statd, and lockd.

About managing and configuring NFS

VCS uses agents to monitor and manage the entities related to NFS. These agents include NFS Agent, Share Agent, NFSRestart Agent, IP Agent, AWSIP Agent, Route53 Agent, AzureIP Agent, AzureDNSZone Agent, IPMultiNIC Agent, and IPMultiNICA Agent.

For information about these agents and their roles in providing NFS high availability, see the *Cluster Server Bundled Agents Reference Guide*.

See “About File share agents” on page 184.

Configuring NFS service groups

You can configure NFS with VCS in several ways. The following configurations are supported for NFS service groups supported by VCS.

- Configuring for a single NFS environment
  Use this configuration to export all the local directories from a single virtual IP address. In this configuration, the NFS resource is part of a failover service group and there is only one NFS related service group in the entire clustered environment. This configuration supports lock recovery and also handles potential NFS ACK storms. This configuration also supports NFSv4. See "Configuring for a single NFS environment" on page 190.
- Configuring for a multiple NFS environment
  Use this configuration to export the NFS shares from a multiple virtual IP addresses. You need to create different NFS share service groups, where each service group has one virtual IP address. Note that NFS is now a part of a different parallel service group. This configuration supports lock recovery and also prevents potential NFS ACK storms. This configuration also supports NFSv4. See "Configuring for a multiple NFS environment" on page 192.
- Configuring for multiple NFS environment with separate storage
  Use this configuration to put all the storage resources into a separate service group. The storage resources such as Mount and DiskGroup are part of different service group. In this configuration, the NFS share service group depends on the storage service group. SFCFSHA uses this configuration where the service group containing the storage resources is a parallel service group. See "Configuring NFS with separate storage" on page 194.
- Configuring NFS services in a parallel service group
  Use this configuration when you want only the NFS service to run. If you want any of the functionality provided by the NFSRestart agent, do not use this configuration. This configuration has some disadvantages because it does not support NFS lock recovery and it does not prevent potential NFS ACK storms. Veritas does not recommend this configuration. See "Configuring all NFS services in a parallel service group" on page 195.

Configuring for a single NFS environment

Use this configuration to export all the local directories from a single virtual IP address. In this configuration, the NFS resource is part of a failover service group and there is only one NFS related service group in the entire clustered environment. This configuration supports lock recovery and also handles potential NFS ACK storms. This configuration also supports NFSv4.

Creating the NFS exports service group

This service group contains the Share and IP resources for exports. The PathName attribute's value for the Share resource must be on shared storage and it must be visible to all nodes in the cluster.

To create the NFS exports service group

1. Create an NFS resource inside the service group.

   Note: You must set NFSLockFailover to 1 for NFSRestart resource if you intend to use NFSv4.

2. If you configure the backing store for the NFS exports using VxVM, create DiskGroup and Mount resources for the mount point that you want to export.

   If you configure the backing store for the NFS exports using LVM, configure the LVMVolumeGroup resource and Mount resource for the mount point that you want to export.

   Refer to Storage agents chapter in the *Cluster Server Bundled Agents Reference Guide* for details.

3. Create an NFSRestart resource. Set the Lower attribute of this NFSRestart resource to 1. Ensure that NFSRes attribute points to the NFS resource that is on the system.

   For NFS lock recovery, make sure that the NFSLockFailover attribute and the LocksPathName attribute have appropriate values. The NFSRestart resource depends on the Mount and NFS resources that you have configured for this service group.

   Note: The NFSRestart resource gets rid of preonline and postoffline triggers for NFS.

4. Create a Share resource. Set the PathName to the mount point that you want to export. In case of multiple shares, create multiple Share resources with different values for their PathName attributes. All the Share resources configured in the service group should have dependency on the NFSRestart resource with a value of 1 for the Lower attribute.

5. Create an IP resource. The value of the Address attribute for this IP resource is used to mount the NFS exports on the client systems. Make the IP resource depend on the Share resources that are configured in the service group.

6 Create a DNS resource if you want NFS lock recovery. The DNS resource depends on the IP resource. Refer to the sample configuration on how to configure the DNS resource.

7 Create an NFSRestart resource. Set the NFSRes attribute to the NFS resource (nfs) that is configured on the system. Set the Lower attribute of this NFSRestart resource to 0. Make the NFSRestart resource depend on the IP resource or the DNS resource (if you want to use NFS lock recovery.)

Note: Ensure that all attributes except the Lower attribute are identical for the two NFSRestart resources.

Configuring for a multiple NFS environment

Use this configuration to export the NFS shares from multiple virtual IP addresses. You need to create different NFS share service groups, where each service group has one virtual IP address. The following example has a single service group with a virtual IP. Note that NFS is now a part of a different parallel service group. This configuration supports lock recovery and also prevents potential NFS ACK storms. This configuration also supports NFSv4.

Creating the NFS service group for a multiple NFS environment

This service group contains an NFS resource. Depending on the service group's use, it can also contain a NIC resource and a Phantom resource.

To create the NFS service group

1 Configure a separate parallel service group (nfs_grp).

2 Set the value of the AutoStart and Parallel attributes to 1 for the service group.

3 The value for the AutoStartList attribute must contain the list of all the cluster nodes in the service group.

4 Configure an NFS resource (nfs) inside this service group. You can also put NIC resource in this service group to monitor a NIC.

Note: You must set NFSLockFailover to 1 for NFSRestart resource if you intend to use NFSv4.

---

5 You must create a Phantom resource in this service group to display the correct state of the service group.

Creating the NFS exports service group for a multiple NFS environment

This service group contains the Share and IP resources for exports. The value for the PathName attribute for the Share resource must be on shared storage and it must be visible to all nodes in the cluster.

To create the NFS exports service group

1. Create an NFS Proxy resource inside the service group. This Proxy resource points to the actual NFS resource that is configured on the system.
2. If you configure the backing store for the NFS exports with VxVM, create DiskGroup and Mount resources for the mount point that you want to export.

   If the backing store for the NFS exports is configured using LVM, configure the LVMVolumeGroup resource and Mount resources for the mount points that you want to export.

   Refer to Storage agents in the *Cluster ServerBundled Agents Reference Guide* for details.
3. Create an NFSRestart resource. Set the Lower attribute of this NFSRestart resource to 1. Ensure that NFSRes attribute points to the NFS resource configured on the system.

   For NFS lock recovery, make sure that the NFSLockFailover attribute and the LocksPathName attribute have appropriate values. The NFSRestart resource depends on the Mount resources that you have configured for this service group. The NFSRestart resource gets rid of preonline and postoffline triggers for NFS.
4. Create a Share resource. Set the PathName attribute to the mount point that you want to export. In case of multiple shares, create multiple Share resources with different values for their PathName attributes. All the Share resources that are configured in the service group need to have dependency on the NFSRestart resource that has a value of 1 for its Lower attribute.
5. Create an IP resource. The value of the Address attribute for this IP resource is used to mount the NFS exports on the client systems. Make the IP resource depend on the Share resources that are configured in the service group.
6. Create a DNS resource if you want NFS lock recovery. The DNS resource depends on the IP resource. Refer to the sample configuration on how to configure the DNS resource.
7. Create an NFSRestart resource. Set the NFSRes attribute to the NFS resource (nfs) that is configured on the system. Set the value of the Lower attribute for this NFSRestart resource to 0. Make the NFSRestart resource depend on the IP resource or the DNS resource to use NFS lock recovery.

Note: Ensure that all attributes except the Lower attribute are identical for the two NFSRestart resources.

Configuring NFS with separate storage

Use this configuration to put all the storage resources into a separate service group. The storage resources such as Mount and DiskGroup are part of different service group. In this configuration, the NFS share service group depends on the storage service group. SFCFSHA uses this configuration where the service group containing the storage resources is a parallel service group.

Creating the NFS service group

This service group contains an NFS resource. Depending on the service group's use, it can also contain a NIC resource and a Phantom resource.

To create the NFS service group

1. Configure a separate parallel service group (nfs_grp).
2. Set the value of the AutoStart and Parallel attributes to 1 for the service group.
3. The value for the AutoStartList must contain the list of all the cluster nodes in the service group.
4. Configure an NFS resource (nfs) inside this service group. You can also put NIC resource in this service group to monitor a NIC. You must create a Phantom resource in this service group to display the correct state of the service group.

Note: You must set NFSLockFailover to 1 for NFSRestart resource if you intend to use NFSv4.

Creating the NFS storage service group

This service group can contain a DiskGroup resource and a Mount resource; or an LVMVolumeGroup resource and a Mount resource.

To create the NFS storage service group

1. If you configure the backing store for the NFS exports with VxVM, create DiskGroup and Mount resources for the mount point that you want to export.
2. If the backing store for the NFS exports is configured using LVM, configure the LVMVolumeGroup resource plus the Mount resource for the mount point that you want to export.

Refer to Storage agents chapter in the *Cluster Server Bundled Agents Reference Guide* for details.

Creating the NFS exports service group

This service group contains the Share resource and IP resource for exports. The value for the PathName attribute for the Share resource must be on shared storage and it must be visible to all nodes in the cluster.

To create the NFS exports service group

1. Create an online local hard dependency between this service group and the storage service group.
2. Create an NFS Proxy resource inside the service group. This Proxy resource points to the actual NFS resource that is configured on the system.
3. Create an NFSRestart resource. Set the Lower attribute of this NFSRestart resource to 1. Ensure that NFSRes attribute points to the NFS resource that is configured on the system. For NFS lock recovery, make sure that the NFSLockFailover attribute and the LocksPathName attribute have appropriate values. The NFSRestart resource gets rid of preonline and postoffline triggers for NFS.
4. Create a Share resource. Set the value of the PathName attribute to the mount point that you want to export. In case of multiple shares, create multiple Share resources with different values for their PathName attributes. All the Share resources configured in the service group need to have dependency on the NFSRestart resource that has a value of 1 for the Lower attribute.
5. Create an IP resource. The value of the Address attribute for this IP resource is used to mount the NFS exports on the client systems. Make the IP resource depend on the Share resources that are configured in the service group.
6. Create a DNS resource if you want to use NFS lock recovery. The DNS resource depends on the IP resource. Refer to the sample configuration on how to configure the DNS resource.
7. Create an NFSRestart resource. Set the NFSRes attribute to the NFS resource (nfs) that is configured on the system. Set the Lower attribute of this NFSRestart resource to 0. To use lock recovery, make the NFSRestart resource depend on the IP resource or the DNS resource.

Note: Ensure that all attributes except the Lower attribute are identical for the two NFSRestart resources.

Configuring all NFS services in a parallel service group

Use this configuration when you want only the NFS service to run. If you want any of the functionality provided by the NFSRestart agent, do not use this configuration. This configuration has some disadvantages because it does not support NFS lock

recovery and it does not prevent potential NFS ACK storms. Veritas does not recommend this configuration.

Creating the NFS service group

This service group contains an NFS resource and an NFSRestart resource.

To create the NFS service group

1. Configure a separate parallel service group (nfs_grp).
2. Set the value of the AutoStart and Parallel attributes to 1 for the service group. The value for the AutoStartList must contain the list of all the cluster nodes in the service group.
3. Configure an NFS resource (nfs) inside this service group. You can also put NIC resource in this service group to monitor a NIC.
4. Configure an NFSRestart resource inside this service group. Set the value of Lower attribute to 2. The NFSRes attribute of this resource must point to the NFS resource (nfs) configured on the system. NFSLockFailover is not supported in this configuration.

Note:

- NFSLockFailover is not supported in this configuration.
- This configuration does not prevent potential NFS ACK storms.
- NFSv4 is not supported in this configuration.

Creating the NFS exports service group

This service group contains the Share and IP resources for exports. The value for the PathName attribute for the Share resource must be on shared storage and it must be visible to all nodes in the cluster.

To create the NFS exports service group

1. Create an NFS Proxy resource inside the service group. This Proxy resource points to the actual NFS resource that is configured on the system.
2. If you configure the backing store for the NFS exports with VxVM, create DiskGroup and Mount resources for the mount point that you want to export.

   If the backing store for the NFS exports is configured using LVM, configure the LVMVolumeGroup plus the Mount resources for the mount point that you want to export.

   Refer to Storage agents chapter of the *Cluster Server Bundled Agents Reference Guide* for details.

3. Create a Share resource. Set the PathName to the mount point that you want to export. In case of multiple shares, create multiple Share resources with different values for their PathName attributes.
4. Create an IP resource. The value of the Address attribute for this IP resource is used to mount the NFS exports on the client systems. Make the IP resource depend on the Share resources that are configured in the service group.

Sample configurations

The following are the sample configurations for some of the supported NFS configurations.

- See "Sample configuration for a single NFS environment without lock recovery" on page 197.
- See "Sample configuration for a single NFS environment with lock recovery" on page 199.
- See "Sample configuration for a single NFSv4 environment" on page 202.
- See "Sample configuration for a multiple NFSv4 environment" on page 204.
- See "Sample configuration for a multiple NFS environment without lock recovery" on page 207.
- See "Sample configuration for a multiple NFS environment with lock recovery" on page 210.
- See "Sample configuration for configuring NFS with separate storage" on page 213.
- See "Sample configuration when configuring all NFS services in a parallel service group" on page 215.

Sample configuration for a single NFS environment without lock recovery

```
include "types.cf"
cluster clus1 (
    UseFence = SCSI3
    )
system sys1 (
    )
system sys2 (
    )
group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
```

```
AutoStartList = { sys1 }
)

DiskGroup vcs_dg1 (
    DiskGroup = dg1
    StartVolumes = 0
    StopVolumes = 0
    )

IP ip_sys1 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    Address = "10.198.90.198"
    NetMask = "255.255.248.0"
    )

Mount vcs_dg1_r01_2 (
    MountPoint = "/testdir/VITA_dg1_r01_2"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
    FSType = vxfs
    FsckOpt = "-y"
    )

Mount vcs_dg1_r0_1 (
    MountPoint = "/testdir/VITA_dg1_r0_1"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
    FSType = vxfs
    FsckOpt = "-y"
    )

NFSRestart NFSRestart_sg11_L (
    NFSRes = nfs
    Lower = 1
    )
NFSRestart NFSRestart_sg11_U (
    NFSRes = nfs
    )

NIC nic_sg11_eth0 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    NetworkHosts = { "10.198.88.1" }
    )
```

```
NFS nfs (
    Nproc = 16
    )


Share share_dg1_r01_2 (
    PathName = "/testdir/VITA_dg1_r01_2"
    Options = rw
    )

Share share_dg1_r0_1 (
    PathName = "/testdir/VITA_dg1_r0_1"
    Options = rw
    )

Volume vol_dg1_r01_2 (
    Volume = dg1_r01_2
    DiskGroup = dg1
    )

Volume vol_dg1_r0_1 (
    Volume = dg1_r0_1
    DiskGroup = dg1
    )
NFSRestart_sg11_L requires nfs
NFSRestart_sg11_L requires vcs_dg1_r01_2
NFSRestart_sg11_L requires vcs_dg1_r0_1
NFSRestart_sg11_U requires ip_sys1
ip_sys1 requires nic_sg11_eth0
ip_sys1 requires share_dg1_r01_2
ip_sys1 requires share_dg1_r0_1
share_dg1_r01_2 requires NFSRestart_sg11_L
share_dg1_r0_1 requires NFSRestart_sg11_L
vcs_dg1_r01_2 requires vol_dg1_r01_2
vcs_dg1_r0_1 requires vol_dg1_r0_1
vol_dg1_r01_2 requires vcs_dg1
vol_dg1_r0_1 requires vcs_dg1
```

Sample configuration for a single NFS environment with lock recovery

```
include "types.cf"
```

```
cluster clus1 (
    UseFence = SCSI3
    )

system sys1 (
    )

system sys2 (
    )
group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
    AutoStartList = { sys1 }
    )

    DiskGroup vcs_dg1 (
        DiskGroup = dg1
        StartVolumes = 0
        StopVolumes = 0
        )

    IP ip_sys1 (
        Device @sys1 = eth0
        Device @sys2 = eth0
        Address = "10.198.90.198"
        NetMask = "255.255.248.0"
        )

    DNS dns_11 (
        Domain = "oradb.sym"
        TSIGKeyFile = "/Koradb.sym.+157+13021.private"
        StealthMasters = { "10.198.90.202" }
        ResRecord @sys1 = { sys1 = "10.198.90.198" }
        ResRecord @sys2 = { sys2 = "10.198.90.198" }
        CreatePTR = 1
        OffDelRR = 1
        )

    Mount vcs_dg1_r01_2 (
        MountPoint = "/testdir/VITA_dg1_r01_2"
        BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
        FSType = vxfs
        FsckOpt = "-y"
        )
```

```
Mount vcs_dg1_r0_1 (
    MountPoint = "/testdir/VITA_dg1_r0_1"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
    FSType = vxfs
    FsckOpt = "-y"
    )

NFS nfs (
    )

NFSRestart NFSRestart_sg11_L (
    NFSRes = nfs
    LocksPathName = "/testdir/VITA_dg1_r01_2"
    NFSLockFailover = 1
    Lower = 1
    )
NFSRestart NFSRestart_sg11_U (
    NFSRes = nfs
    LocksPathName = "/testdir/VITA_dg1_r01_2"
    NFSLockFailover = 1
    LockFailoverAddress="10.198.90.198"
    )

NIC nic_sg11_eth0 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    NetworkHosts = { "10.198.88.1" }
    )

Share share_dg1_r01_2 (
    PathName = "/testdir/VITA_dg1_r01_2"
    Options = rw
    )

Share share_dg1_r0_1 (
    PathName = "/testdir/VITA_dg1_r0_1"
    Options = rw
    )

Volume vol_dg1_r01_2 (
    Volume = dg1_r01_2
    DiskGroup = dg1
```

```
        )

    Volume vol_dg1_r0_1 (
        Volume = dg1_r0_1
        DiskGroup = dg1
        )

    NFSRestart_sg11_L requires nfs
    NFSRestart_sg11_L requires vcs_dg1_r01_2
    NFSRestart_sg11_L requires vcs_dg1_r0_1
    NFSRestart_sg11_U requires dns_11
    dns_11 requires ip_sys1
    ip_sys1 requires nic_sg11_eth0
    ip_sys1 requires share_dg1_r01_2
    ip_sys1 requires share_dg1_r0_1
    share_dg1_r01_2 requires NFSRestart_sg11_L
    share_dg1_r0_1 requires NFSRestart_sg11_L
    vcs_dg1_r01_2 requires vol_dg1_r01_2
    vcs_dg1_r0_1 requires vol_dg1_r0_1
    vol_dg1_r01_2 requires vcs_dg1
    vol_dg1_r0_1 requires vcs_dg1
```

Sample configuration for a single NFSv4 environment

```
include "types.cf"
cluster clus1 (
    UseFence = SCSI3
    )
system sys1 (
    )
system sys2 (
    )
group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
    AutoStartList = { sys1 }
    )
    DiskGroup vcs_dg1 (
        DiskGroup = dg1
        StartVolumes = 0
        StopVolumes = 0
        )
    IP ip_sys1 (
        Device @sys1 = eth0
```

```
        Device @sys2 = eth0
        Address = "10.198.90.198"
        NetMask = "255.255.248.0"
        )
    DNS dns_11 (
        Domain = "oradb.sym"
        TSIGKeyFile = "/Koradb.sym.+157+13021.private"
        StealthMasters = { "10.198.90.202" }
        ResRecord @sys1 = { sys1 = "10.198.90.198" }
        ResRecord @sys2 = { sys2 = "10.198.90.198" }
        CreatePTR = 1
        OffDelRR = 1
        )
    Mount vcs_dg1_r01_2 (
        MountPoint = "/testdir/VITA_dg1_r01_2"
        BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
        FSType = vxfs
        FsckOpt = "-y"
        )
    Mount vcs_dg1_r0_1 (
        MountPoint = "/testdir/VITA_dg1_r0_1"
        BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
        FSType = vxfs
        FsckOpt = "-y"
        )


NFS nfs (
        NFSv4Support = 1
        )


NFSRestart NFSRestart_sg11_L (
        NFSRes = nfs
        LocksPathName = "/testdir/VITA_dg1_r01_2"
        NFSLockFailover = 1
        Lower = 1
        )
    NFSRestart NFSRestart_sg11_U (
        NFSRes = nfs
        LocksPathName = "/testdir/VITA_dg1_r01_2"
        NFSLockFailover = 1
        )
    NIC nic_sg11_eth0 (
```

```
        Device @sys1 = eth0
        Device @sys2 = eth0
        NetworkHosts = { "10.198.88.1" }
        NetworkType = ether
        )
    Share share_dg1_r01_2 (
        PathName = "/testdir/VITA_dg1_r01_2"
        Options = rw
        )
    Share share_dg1_r0_1 (
        PathName = "/testdir/VITA_dg1_r0_1"
        Options = rw
        )
    Volume vol_dg1_r01_2 (
        Volume = dg1_r01_2
        DiskGroup = dg1
        )
    Volume vol_dg1_r0_1 (
        Volume = dg1_r0_1
        DiskGroup = dg1
        )
    NFSRestart_sg11_L requires nfs
    NFSRestart_sg11_L requires vcs_dg1_r01_2
    NFSRestart_sg11_L requires vcs_dg1_r0_1
    NFSRestart_sg11_U requires dns_11
    dns_11 requires ip_sys1
    ip_sys1 requires nic_sg11_eth0
    ip_sys1 requires share_dg1_r01_2
    ip_sys1 requires share_dg1_r0_1
    share_dg1_r01_2 requires NFSRestart_sg11_L
    share_dg1_r0_1 requires NFSRestart_sg11_L
    vcs_dg1_r01_2 requires vol_dg1_r01_2
    vcs_dg1_r0_1 requires vol_dg1_r0_1
    vol_dg1_r01_2 requires vcs_dg1
    vol_dg1_r0_1 requires vcs_dg1
```

Sample configuration for a multiple NFSv4 environment

```
include "types.cf"
cluster clus1 (
    UseFence = SCSI3
    )
```

```
system sys1 (
    )
system sys2 (
    )
group nfs_sg (
    SystemList = { sys1 = 0, sys2 = 1 }
    Parallel = 1
    AutoStartList = { sys1, sys2 }
    )

NFS n1 (
        Nproc = 6
        NFSv4Support = 1
        )

Phantom ph1 (
        )
group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
    AutoStartList = { sys1 }
    )
    DiskGroup vcs_dg1 (
        DiskGroup = dg1
        StartVolumes = 0
        StopVolumes = 0
        )
    DNS dns_11 (
        Domain = "oradb.sym"
        TSIGKeyFile = "/Koradb.sym.+157+13021.private"
        StealthMasters = { "10.198.90.202" }
        ResRecord @sys1 = { sys1 = "10.198.90.198" }
        ResRecord @sys2 = { sys2 = "10.198.90.198" }
        CreatePTR = 1
        OffDelRR = 1
        )
    IP ip_sys2 (
        Device @sys1 = eth0
        Device @sys2 = eth0
        Address = "10.198.90.198"
        NetMask = "255.255.248.0"
        )
    Mount vcs_dg1_r01_2 (
        MountPoint = "/testdir/VITA_dg1_r01_2"
        BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
```

```
    FSType = vxfs
    FsckOpt = "-y"
    )
Mount vcs_dg1_r0_1 (
    MountPoint = "/testdir/VITA_dg1_r0_1"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
    FSType = vxfs
    FsckOpt = "-y"
    )
NFSRestart NFSRestart_sg11_L (
    NFSRes = n1
    Lower = 1
    LocksPathName = "/testdir/VITA_dg1_r01_2"
    NFSLockFailover = 1
    )
NFSRestart NFSRestart_sg11_U (
    NFSRes = n1
    LocksPathName = "/testdir/VITA_dg1_r01_2"
    NFSLockFailover = 1
    )
NIC nic_sg11_eth0 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    NetworkHosts = { "10.198.88.1" }
    )
Proxy p11 (
    TargetResName = n1
    )
Share share_dg1_r01_2 (
    PathName = "/testdir/VITA_dg1_r01_2"
    Options = rw
    )
Share share_dg1_r0_1 (
    PathName = "/testdir/VITA_dg1_r0_1"
    Options = rw
    )
Volume vol_dg1_r01_2 (
    Volume = dg1_r01_2
    DiskGroup = dg1
    )
Volume vol_dg1_r0_1 (
    Volume = dg1_r0_1
    DiskGroup = dg1
```

```
    )
requires group nfs_sg online local firm
NFSRestart_sg11_L requires p11
NFSRestart_sg11_L requires vcs_dg1_r01_2
NFSRestart_sg11_L requires vcs_dg1_r0_1
NFSRestart_sg11_U requires dns_11
dns_11 requires ip_sys1
ip_sys1 requires nic_sg11_eth0
ip_sys1 requires share_dg1_r01_2
ip_sys1 requires share_dg1_r0_1
share_dg1_r01_2 requires NFSRestart_sg11_L
share_dg1_r0_1 requires NFSRestart_sg11_L
vcs_dg1_r01_2 requires vol_dg1_r01_2
vcs_dg1_r0_1 requires vol_dg1_r0_1
vol_dg1_r01_2 requires vcs_dg1
vol_dg1_r0_1 requires vcs_dg1
```

Sample configuration for a multiple NFS environment without lock recovery

```
include "types.cf"

cluster clus1 (
    UseFence = SCSI3
    )

system sys1 (
    )

system sys2 (
    )

group nfs_sg (
    SystemList = { sys1 = 0, sys2 = 1 }
    Parallel = 1
    AutoStartList = { sys1, sys2 }
    )


NFS n1 (
```

```
     Nproc = 6
     )


Phantom ph1 (
     )

group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
    AutoStartList = { sys1 }
    )

 DiskGroup vcs_dg1 (
     DiskGroup = dg1
     StartVolumes = 0
     StopVolumes = 0
     )

 IP ip_sys1 (
     Device @sys1 = eth0
     Device @sys2 = eth0
     Address = "10.198.90.198"
     NetMask = "255.255.248.0"
     )

 Mount vcs_dg1_r01_2 (
     MountPoint = "/testdir/VITA_dg1_r01_2"
     BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
     FSType = vxfs
     FsckOpt = "-y"
     )

 Mount vcs_dg1_r0_1 (
     MountPoint = "/testdir/VITA_dg1_r0_1"
     BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
     FSType = vxfs
     FsckOpt = "-y"
     )

 NFSRestart NFSRestart_sg11_L (
     NFSRes = n1
     Lower = 1
     )
```

```
NFSRestart NFSRestart_sg11_U (
    NFSRes = n1
    )

NIC nic_sg11_eth0 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    NetworkHosts = { "10.198.88.1" }
    )

Proxy p11 (
    TargetResName = n1
    )

Share share_dg1_r01_2 (
    PathName = "/testdir/VITA_dg1_r01_2"
    Options = rw
    )

Share share_dg1_r0_1 (
    PathName = "/testdir/VITA_dg1_r0_1"
    Options = rw
    )

Volume vol_dg1_r01_2 (
    Volume = dg1_r01_2
    DiskGroup = dg1
    )

Volume vol_dg1_r0_1 (
    Volume = dg1_r0_1
    DiskGroup = dg1
    )

requires group nfs_sg online local firm
NFSRestart_sg11_L requires p11
NFSRestart_sg11_L requires vcs_dg1_r01_2
NFSRestart_sg11_L requires vcs_dg1_r0_1
NFSRestart_sg11_U requires ip_sys1
ip_sys1 requires nic_sg11_eth0
ip_sys1 requires share_dg1_r01_2
ip_sys1 requires share_dg1_r0_1
share_dg1_r01_2 requires NFSRestart_sg11_L
```

```
share_dg1_r0_1 requires NFSRestart_sg11_L
vcs_dg1_r01_2 requires vol_dg1_r01_2
vcs_dg1_r0_1 requires vol_dg1_r0_1
vol_dg1_r01_2 requires vcs_dg1
vol_dg1_r0_1 requires vcs_dg1
```

Sample configuration for a multiple NFS environment with lock recovery

```
include "types.cf"

cluster clus1 (
    UseFence = SCSI3
    )

system sys1 (
)

system sys2 (
)

group nfs_sg (
    SystemList = { sys1 = 0, sys2 = 1 }
    Parallel = 1
    AutoStartList = { sys1, sys2 }
)


NFS n1 (
    Nproc = 6
    )


Phantom ph1 (
    )

group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
    AutoStartList = { sys1 }
    )

DiskGroup vcs_dg1 (
    DiskGroup = dg1
```

```
    StartVolumes = 0
    StopVolumes = 0
    )

DNS dns_11 (
    Domain = "oradb.sym"
    TSIGKeyFile = "/Koradb.sym.+157+13021.private"
    StealthMasters = { "10.198.90.202" }
    ResRecord @sys1 = { sys1 = "10.198.90.198" }
    ResRecord @sys2 = { sys2 = "10.198.90.198" }
    CreatePTR = 1
    OffDelRR = 1
    )

IP ip_sys1 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    Address = "10.198.90.198"
    NetMask = "255.255.248.0"
    )

Mount vcs_dg1_r01_2 (
    MountPoint = "/testdir/VITA_dg1_r01_2"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
    FSType = vxfs
    FsckOpt = "-y"
    )

Mount vcs_dg1_r0_1 (
    MountPoint = "/testdir/VITA_dg1_r0_1"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
    FSType = vxfs
    FsckOpt = "-y"
    )

NFSRestart NFSRestart_sg11_L (
    NFSRes = n1
    Lower = 1
    LocksPathName = "/testdir/VITA_dg1_r01_2"
    NFSLockFailover = 1
    )
NFSRestart NFSRestart_sg11_U (
    NFSRes = n1
```

```
    LocksPathName = "/testdir/VITA_dg1_r01_2"
    NFSLockFailover = 1
    )

NIC nic_sg11_eth0 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    NetworkHosts = { "10.198.88.1" }
    )

Proxy p11 (
    TargetResName = n1
    )

Share share_dg1_r01_2 (
    PathName = "/testdir/VITA_dg1_r01_2"
    Options = rw
    )

Share share_dg1_r0_1 (
    PathName = "/testdir/VITA_dg1_r0_1"
    Options = rw
    )

Volume vol_dg1_r01_2 (
    Volume = dg1_r01_2
    DiskGroup = dg1
    )

Volume vol_dg1_r0_1 (
    Volume = dg1_r0_1
    DiskGroup = dg1
    )

requires group nfs_sg online local firm
NFSRestart_sg11_L requires p11
NFSRestart_sg11_L requires vcs_dg1_r01_2
NFSRestart_sg11_L requires vcs_dg1_r0_1
NFSRestart_sg11_U requires dns_11
dns_11 requires ip_sys1
ip_sys1 requires nic_sg11_eth0
ip_sys1 requires share_dg1_r01_2
ip_sys1 requires share_dg1_r0_1
```

```
share_dg1_r01_2 requires NFSRestart_sg11_L
share_dg1_r0_1 requires NFSRestart_sg11_L
vcs_dg1_r01_2 requires vol_dg1_r01_2
vcs_dg1_r0_1 requires vol_dg1_r0_1
vol_dg1_r01_2 requires vcs_dg1
vol_dg1_r0_1 requires vcs_dg1
```

Sample configuration for configuring NFS with separate storage

```
include "types.cf"

    cluster clus1 (
         UseFence = SCSI3
       )

    system sys1 (
        )

    system sys2 (
        )

    group nfs_sg (
        SystemList = { sys1 = 0, sys2 = 1 }
        Parallel = 1
        AutoStartList = { sys1, sys2 }
        )

    NFS n1 (
        Nproc = 6
        )

    Phantom ph1 (
        )

    group sg11storage (
        SystemList = { sys1 = 0, sys2 = 1 }
        )

    DiskGroup vcs_dg1 (
        DiskGroup = dg1
        StartVolumes = 0
        StopVolumes = 0
        )
```

```
Mount vcs_dg1_r01_2 (
    MountPoint = "/testdir/VITA_dg1_r01_2"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
    FSType = vxfs
    FsckOpt = "-y"
    )

Mount vcs_dg1_r0_1 (
    MountPoint = "/testdir/VITA_dg1_r0_1"
    BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
    FSType = vxfs
    FsckOpt = "-y"
    )

Volume vol_dg1_r01_2 (
    Volume = dg1_r01_2
    DiskGroup = dg1
    )

Volume vol_dg1_r0_1 (
    Volume = dg1_r0_1
    DiskGroup = dg1
    )

vcs_dg1_r01_2 requires vol_dg1_r01_2
vcs_dg1_r0_1 requires vol_dg1_r0_1
vol_dg1_r01_2 requires vcs_dg1
vol_dg1_r0_1 requires vcs_dg1

group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
    AutoStartList = { sys1 }
    )

IP sys1 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    Address = "10.198.90.198"
    NetMask = "255.255.248.0"
    )

NFSRestart NFSRestart_sg11_L (
```

```
        NFSRes = n1
        Lower = 1
        )

    NFSRestart NFSRestart_sg11_U (
        NFSRes = n1
        )

    NIC nic_sg11_eth0 (
        Device @sys1 = eth0
        Device @sys2 = eth0
        NetworkHosts = { "10.198.88.1" }
        )

    Proxy p11 (
        TargetResName = n1
        )

    Share share_dg1_r01_2 (
        PathName = "/testdir/VITA_dg1_r01_2"
        Options = rw
        )
    Share share_dg1_r0_1 (
        PathName = "/testdir/VITA_dg1_r0_1"
        Options = rw
        )

    requires group sg11storage online local hard
    NFSRestart_sg11_L requires p11
    NFSRestart_sg11_U requires ip_sys1
    ip_sys1 requires nic_sg11_eth0
    ip_sys1 requires share_dg1_r01_2
    ip_sys1 requires share_dg1_r0_1
    share_dg1_r01_2 requires NFSRestart_sg11_L
    share_dg1_r0_1 requires NFSRestart_sg11_L
```

Sample configuration when configuring all NFS services in a parallel service group

```
include "types.cf"

    cluster clus1 (
        UseFence = SCSI3
```

```
    )

system sys1 (
    )

system sys2 (
    )

group nfs_sg (
    SystemList = { sys1 = 0, sys2 = 1 }
    Parallel = 1
    AutoStartList = { sys1, sys2 }
    )


NFS n1 (
    Nproc = 6
    )


NFSRestart nfsrestart (
    NFSRes = n1
    Lower = 2
    )

nfsrestart requires n1

group sg11 (
    SystemList = { sys1 = 0, sys2 = 1 }
    AutoStartList = { sys1 }
    )

IP ip_sys1 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    Address = "10.198.90.198"
    NetMask = "255.255.248.0"
    )

NIC nic_sg11_eth0 (
    Device @sys1 = eth0
    Device @sys2 = eth0
    NetworkHosts = { "10.198.88.1" }
    )
```

```
    Proxy p11 (
        TargetResName = n1
        )

    Share share_dg1_r01_2 (
        PathName = "/testdir/VITA_dg1_r01_2"
        Options = rw
        )

    Share share_dg1_r0_1 (
        PathName = "/testdir/VITA_dg1_r0_1"
        Options = rw
        )

    requires group sg11storage online local hard
    ip_sys1 requires nic_sg11_eth0
    ip_sys1 requires share_dg1_r01_2
    ip_sys1 requires share_dg1_r0_1
    share_dg1_r01_2 requires p11
    share_dg1_r0_1 requires p11

group sg11storage (
    SystemList = { sys1 = 0, sys2 = 1 }
    )

    DiskGroup vcs_dg1 (
        DiskGroup = dg1
        StartVolumes = 0
        StopVolumes = 0
        )

    Mount vcs_dg1_r01_2 (
        MountPoint = "/testdir/VITA_dg1_r01_2"
        BlockDevice = "/dev/vx/dsk/dg1/dg1_r01_2"
        FSType = vxfs
        FsckOpt = "-y"
        )

    Mount vcs_dg1_r0_1 (
        MountPoint = "/testdir/VITA_dg1_r0_1"
        BlockDevice = "/dev/vx/dsk/dg1/dg1_r0_1"
        FSType = vxfs
```

```
    FsckOpt = "-y"
    )

Volume vol_dg1_r01_2 (
    Volume = dg1_r01_2
    DiskGroup = dg1
    )

Volume vol_dg1_r0_1 (
    Volume = dg1_r0_1
    DiskGroup = dg1
    )

vcs_dg1_r01_2 requires vol_dg1_r01_2
vcs_dg1_r0_1 requires vol_dg1_r0_1
vol_dg1_r01_2 requires vcs_dg1
vol_dg1_r0_1 requires vcs_dg1
```

About configuring the RemoteGroup agent

The RemoteGroup agent monitors and manages service groups in a remote cluster. Use the RemoteGroup agent to establish dependencies between applications that are configured on different VCS clusters.

For example, you configure an Apache resource in a local cluster, and configure an Oracle resource in a remote cluster. In this example, the Apache resource in the local cluster depends on the Oracle resource in the remote cluster. You can use the RemoteGroup agent to establish this dependency between the service groups that contain the Apache resource and the Oracle resource, if you add the RemoteGroup agent as a resource in the Apache service group.

See the *Cluster Server Bundled Agents Reference Guide* for more information about the agent and its attributes.

Note: RemoteGroup agent configurations and Virtual Business Service configurations are mutually exclusive though both provide multi-tier application support.

For information about Virtual Business Services, see the *Virtual Business Service–Availability User's Guide*.

About the ControlMode attribute

In the ControlMode attribute, you can use these values, depending on your needs: OnOff, MonitorOnly, and OnlineOnly.

About the OnOff mode

Select the OnOff value of this attribute when you want the RemoteGroup resource to manage the remote service group completely.

In case of one-to-one mapping, set the value of the AutoFailOver attribute of the remote service group to 0. This avoids unnecessary onlining or offlining of the remote service group.

About the MonitorOnly mode

Select the MonitorOnly value of this attribute when you want to monitor the state of the remote service group. When you choose the MonitorOnly attribute, the RemoteGroup agent does not have control over the remote service group and cannot bring it online or take it offline.

The remote service group should be in an ONLINE state before you bring the RemoteGroup resource online.

Veritas recommends that the AutoFailOver attribute of the remote service group be set to 1.

About the OnlineOnly mode

Select the OnlineOnly value of this attribute when the remote service group takes a long time to come online or to go offline. When you use OnlineOnly for the ControlMode attribute, a switch or fail over of the local service group with VCSSysName set to ANY does not cause the remote service group to be taken offline and brought online.

Taking the RemoteGroup resource offline does not take the remote service group offline.

If you choose one-to-one mapping between the local nodes and remote nodes, then the value of the AutoFailOver attribute of the remote service group must be 0.

Note: When you set the value of ControlMode to OnlineOnly or to MonitorOnly, the recommend value of the VCSSysName attribute of the RemoteGroup resource is ANY. If you want one-to-one mapping between the local nodes and the remote nodes, then a switch or fail over of local service group is impossible. It is important to note that in both these configurations the RemoteGroup agent does not take the remote service group offline.

About the ReturnIntOffline attribute

The ReturnIntOffline attribute can take one of three values: RemotePartial, RemoteOffline, and RemoteFaulted.

These values are not mutually exclusive and can be used in combination with one another. You must set the IntentionalOffline attribute of RemoteGroup resource to 1 for the ReturnIntOffline attribute to work.

About the RemotePartial option

Select the RemotePartial value of this attribute when you want the RemoteGroup resource to return an IntentionalOffline when the remote service group is in an ONLINE | PARTIAL state.

About the RemoteOffline option

Select the RemoteOffline value of this attribute when you want the RemoteGroup resource to return an IntentionalOffline when the remote service group is in an OFFLINE state.

About the RemoteFaulted option

Select the RemoteFaulted value of this attribute when you want the RemoteGroup resource to return an IntentionalOffline when the remote service group is in an OFFLINE | FAULTED state.

Configuring a RemoteGroup resource

This topic describes how to configure a RemoteGroup resource.

In this example configuration, the following is true:

- VCS cluster (cluster1) provides high availability for Web services. Configure a VCS service group (ApacheGroup) with an agent to monitor the Web server (for example Apache) to monitor the Web services.

- VCS cluster (cluster2) provides high availability for the database required by the Web-services.

  Configure a VCS service group (OracleGroup) with a database agent (for example Oracle) to monitor the database.

The database resource must come online before the Web server comes online. You create this dependency using the RemoteGroup agent.

To configure the RemoteGroup agent

1. Add a RemoteGroup resource in the ApacheGroup service group (in cluster 1).
2. Link the resources such that the Web server resource depends on the RemoteGroup resource.
3. Configure the following RemoteGroup resource items to monitor or manage the service group that contains the database resource:
   - IpAddress:

     Set to the IP address or DNS name of a node in cluster2. You can also set this to a virtual IP address.
   - GroupName

     Set to OracleGroup.
   - ControlMode

     Set to OnOff.
   - Username

     Set to the name of a user who has administrative privileges for OracleGroup.
   - Password

     Encrypted password for the user mentioned in Username. Encrypt the password the `vcsencrypt` utility.

     See "Encrypting agent passwords" on page 97.
   - VCSSysName

     Set to local, per-node values.

     VCSSysName@local1—Set this value to remote1.

     VCSSysName@local2—Set this value to remote2.

     Note: If the remote cluster runs in secure mode, you must set the value for DomainType or BrokerIp attributes.

     ---
4. Set the value of the AutoFailOver attribute of the OracleGroup to 0.

Service group behavior with the RemoteGroup agent

Consider the following potential actions to better understand this solution.

Bringing the Apache service group online

Following are the dependencies to bring the Apache service group online:

- The Apache resource depends on the RemoteGroup resource.
- The RemoteGroup agent communicates to the remote cluster and authenticates the specified user.
- The RemoteGroup agent brings the database service group online in cluster2.
- The Apache resource comes online after the RemoteGroup resource is online.

Thus, you establish an application-level dependency across two different VCS clusters. The Apache resource does not go online unless the RemoteGroup goes online. The RemoteGroup resource does not go online unless the database service group goes online.

Unexpected offline of the database service group

Following are the sequence of actions when the database service group is unexpectedly brought offline:

- The RemoteGroup resource detects that the database group has gone OFFLINE or has FAULTED.
- The RemoteGroup resource goes into a FAULTED state.
- All the resources in the Apache service group are taken offline on the node.
- The Apache group fails over to another node.
- As part of the fail over, the Oracle service group goes online on another node in cluster2.

Taking the Apache service group offline

Following are the sequence of actions when the Apache service group is taken offline:

- All the resources dependant on the RemoteGroup resource are taken offline.
- The RemoteGroup agent tries to take the Oracle service group offline.
- Once the Oracle service group goes offline, the RemoteGroup goes offline.

Thus, the Web server is taken offline before the database goes offline.

Configuring RemoteGroup resources in parallel service groups

When a RemoteGroup resource is configured inside parallel service groups, it can come online on all the cluster nodes, including the offline nodes. Multiple instances of the RemoteGroup resource on cluster nodes can probe the state of a remote service group.

Note: The RemoteGroup resource automatically detects whether it is configured for a parallel service group or for a failover service group. No additional configuration is required to enable the RemoteGroup resource for parallel service groups.

A RemoteGroup resource in parallel service groups has the following characteristics:

- The RemoteGroup resource continues to monitor the remote service group even when the resource is offline.
- The RemoteGroup resource does not take the remote service group offline if the resource is online anywhere in the cluster.
- After an agent restarts, the RemoteGroup resource does not return offline if the resource is online on another cluster node.
- The RemoteGroup resource takes the remote service group offline if it is the only instance of RemoteGroup resource online in the cluster.
- An attempt to bring a RemoteGroup resource online has no effect if the same resource instance is online on another node in the cluster.

About configuring Samba service groups

You can configure Samba in parallel configurations or failover configurations with VCS.

Refer to the *Cluster Server Bundled Agents Reference Guide* for platform-specific details and examples of the attributes of the Samba family of agents.

Sample configuration for Samba in a failover configuration

The configuration contains a failover service group containing SambaServer, NetBios, IP, NIC, and SambaShare resources. You can configure the same NetBiosName and Interfaces attribute values for all nodes because the resource comes online only on one node at a time.

```
include "types.cf"
```

```
cluster clus1 (
)

system sys1(
)

system sys2(
)

group smbserver (
        SystemList = { sys1= 0, sys2= 1 }
        )

        IP ip (
                Device = eth0
                Address = "10.209.114.201"
                NetMask = "255.255.252.0"
                )

        NIC nic (
                Device = eth0
                NetworkHosts = { "10.209.74.43" }
                )

        NetBios nmb (
                SambaServerRes = smb
                NetBiosName = smb_vcs
                Interfaces = { "10.209.114.201" }
                )

        SambaServer smb (
                ConfFile = "/etc/samba/smb.conf"
                LockDir = "/var/run"
                SambaTopDir = "/usr"
                )

        SambaShare smb_share (
                SambaServerRes = smb
                ShareName = share1
                ShareOptions = "path = /samba_share/; public = yes;
                      writable = yes"
                )
```

```
ip requires nic
nmb requires smb
smb requires ip
smb_share requires nmb
```

Configuring the Coordination Point agent

To monitor I/O fencing coordination points, configure the Coordination Point agent in the VCS cluster where you have configured I/O fencing.

For server-based fencing, you can either use the `-fencing` option of the installer to configure the agent or you can manually configure the agent. For disk-based fencing, you must manually configure the agent.

See the *Cluster Server Bundled Agents Reference Guide* for more information on the agent.

See the *Cluster Server Installation Guide* for instructions to configure the agent.

About testing resource failover by using HA fire drills

Configuring high availability for a database or an application requires several infrastructure and configuration settings on multiple systems. However, cluster environments are subject to change after the initial setup. Administrators add disks, create new disk groups and volumes, add new cluster nodes, or new NICs to upgrade and maintain the infrastructure. Keeping the cluster configuration updated with the changing infrastructure is critical.

HA fire drills detect discrepancies between the VCS configuration and the underlying infrastructure on a node; discrepancies that might prevent a service group from going online on a specific node.

See the *Cluster Server Bundled Agents Reference Guide* for information on which agents support HA fire drills.

About HA fire drills

The HA fire drill (earlier known as virtual fire drill) feature uses the Action function associated with the agent. The Action functions of the supported agents are updated to support the HA fire drill functionality—running infrastructure checks and fixing specific errors.

The infrastructure check verifies the resources defined in the VCS configuration file (main.cf) have the required infrastructure to fail over on another node. For

example, an infrastructure check for the Mount resource verifies the existence of the mount directory defined in the MountPoint attribute for the resource.

You can run an infrastructure check only when the service group is online. The check verifies that the specified node is a viable failover target capable of hosting the service group.

The HA fire drill provides an option to fix specific errors detected during the infrastructure check.

About running an HA fire drill

You can run a HA fire drill from the command line or from Cluster Manager (Java Console).

See "Running HA fire drills" on page 176.

Chapter 7

Predicting VCS behavior using VCS Simulator

This chapter includes the following topics:

- About VCS Simulator
- VCS Simulator ports
- Administering VCS Simulator from the command line interface

About VCS Simulator

VCS Simulator enables you to simulate and test cluster configurations. Use VCS Simulator to view and modify service group and resource configurations and test failover behavior. VCS Simulator can be run on a stand-alone system and does not require any additional hardware.

VCS Simulator runs an identical version of the VCS High Availability Daemon (HAD) as in a cluster, ensuring that failover decisions are identical to those in an actual cluster.

You can test configurations from different operating systems using VCS Simulator. The VCS simulator can run only on Windows systems. However, it can simulate non-Windows operating systems on a Windows system. For example, you can run VCS Simulator on a Windows system and test VCS configurations for Windows, Linux, Solaris, HP-UX, and AIX clusters. VCS Simulator also enables creating and testing global clusters.

You can administer VCS Simulator from the Java Console or from the command line.

To download VCS Simulator, go to https://www.veritas.com/product/storage-management.

VCS Simulator ports

Table 7-1 lists the ports that VCS Simulator uses to connect to the various cluster configurations. You can modify cluster configurations to adhere to your network policies. Also, Veritas might change port assignments or add new ports based on the number of VCS Simulator configurations.

Table 7-1 VCS Simulator ports

Port	Usage
15552	SOL_ORA_SRDF_C1:simulatorport
15553	SOL_ORA_SRDF_C2:simulatorport
15554	SOL_ORACLE:simulatorport
15555	LIN_NFS:simulatorport
15556	HP_NFS:simulatorport
15557	AIX_NFS:simulatorport
15558	Consolidation:simulatorport
15559	SOL_NPLUS1:simulatorport
15572	AcmePrimarySite:simulatorport
15573	AcmeSecondarySite:simulatorport
15580	Win_Exch_2K7_primary:simulatorport
15581	Win_Exch_2K7_secondary:simulatorport
15582	WIN_NTAP_EXCH_CL1:simulatorport
15583	WIN_NTAP_EXCH_CL2:simulatorport
15611	WIN_SQL2K5_VVR_C1:simulatorport
15612	WIN_SQL2K5_VVR_C2:simulatorport
15613	WIN_SQL2K8_VVR_C1:simulatorport
15614	WIN_SQL2K8_VVR_C2:simulatorport
15615	WIN_E2K10_VVR_C1:simulatorport
15616	WIN_E2K10_VVR_C2:simulatorport

Table 7-2 lists the ports that the VCS Simulator uses for the wide area connector (WAC) process. Set the WAC port to -1 to disable WAC simulation.

Table 7-2 WAC ports

Port	Usage
15562	SOL_ORA_SRDF_C1:wacport
15563	SOL_ORA_SRDF_C2:wacport
15566	Win_Exch_2K7_primary:wacport
15567	Win_Exch_2K7_secondary:wacport
15570	WIN_NTAP_EXCH_CL1:wacport
15571	WIN_NTAP_EXCH_CL2:wacport
15582	AcmePrimarySite:wacport
15583	AcmeSecondarySite:wacport
15661	WIN_SQL2K5_VVR_C1:wacport
15662	WIN_SQL2K5_VVR_C2:wacport
15663	WIN_SQL2K8_VVR_C1:wacport
15664	WIN_SQL2K8_VVR_C2:wacport
15665	WIN_E2K10_VVR_C1:wacport
15666	WIN_E2K10_VVR_C2:wacport

Administering VCS Simulator from the command line interface

Start VCS Simulator on a Windows system before creating or administering simulated clusters.

Note: VCS Simulator treats clusters that are created from the command line and the Java Console separately. Hence, clusters that are created from the command line are not visible in the graphical interface. If you delete a cluster from the command line, you may see the cluster in the Java Console.

Starting VCS Simulator from the command line interface

This topic describes how to start VCS simulator from the command line:

To start VCS Simulator from the command line (Windows)

VCS Simulator installs platform-specific types.cf files at the path %*VCS_SIMULATOR_HOME*%\types\. The variable %*VCS_SIMULATOR_HOME*% represents the VCS Simulator installation directory, typically C:\Program Files\Veritas\VCS Simulator\.

Example: C:\DOS>set %VCS_SIMULATOR_HOME%=C:\Program Files\Veritas\VCS Simulator\

1. To simulate a cluster running a particular operating system, copy the types.cf. file for the operating system from the *types* directory to %*VCS_SIMULATOR_HOME*%\default_clus\conf\config\.

   For example, if the cluster to be simulated runs on the Linux platform, copy the file types.cf.linux.

2. Add custom type definitions to the file, if required, and rename the file to types.cf.

3. If you have a main.cf file to run in the simulated cluster, copy it to %*VCS_SIMULATOR_HOME*%\default_clus\conf\config\.

4. Start VCS Simulator:

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -start system_name
   ```

   The variable *system_name* represents a system name, as defined in the configuration file main.cf.

   This command starts VCS Simulator on port 14153.

5. Add systems to the configuration, if desired:

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -sys -add system_name
   ```

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -up system_name
   ```

6. Verify the state of each node in the cluster:

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -sys -state
   ```

   See "To simulate global clusters from the command line" on page 231.

To simulate global clusters from the command line

1. Install VCS Simulator in a directory (%VCS_SIMULATOR_HOME%) on your system.

   See the section Installing VCS Simulator in the *Cluster Server Installation Guide*.

2. Set up the clusters on your system. Run the following command to add a cluster:

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -setupclus new_clustername -simport
   port_no -wacport port_no
   ```

   Do not use default_clus as the cluster name when simulating a global cluster.

   VCS Simulator copies the sample configurations to the path %VCS_SIMULATOR_HOME%\clustername and creates a system named *clustername*_sys1.

   For example, to add cluster clus_a using ports 15555 and 15575, run the following command:

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -setupclus clus_a -simport 15555
   -wacport 15575
   ```

   Similarly, add the second cluster:

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -setupclus clus_b -simport 15556
   -wacport 15576
   ```

   To create multiple clusters without simulating a global cluster environment, specify -1 for the wacport.

3. Start the simulated clusters:

   ```
   %VCS_SIMULATOR_HOME%\bin> hasim -start clustername_sys1
   -clus clustername
   ```

4. Set the following environment variables to access VCS Simulator from the command line:

   - set %VCS_SIM_PORT%=*port_number*
   - set %VCS_SIM_WAC_PORT%=*wacport*

   Note that you must set these variables for each simulated cluster, otherwise VCS Simulator always connects default_clus, the default cluster.

   You can use the Java Console to link the clusters and to configure global service groups.

You can also edit the configuration file main.cf manually to create the global cluster configuration.

Administering simulated clusters from the command line

The functionality of VCS Simulator commands mimic that of standard ha commands.

Table 7-3 describes the VCS simulator commands:

Table 7-3 VCS simulator commands

Command	Description
`hasim -start` *`system_name`*	Starts VCS Simulator. The variable *system_name* represents the system that will transition from the LOCAL_BUILD state to the RUNNING state.
`hasim -setupclus` *`clustername`* `-simport port_no [-wacport port_no] [-sys systemname]`	Creates a simulated cluster and associates the specified ports with the cluster.
`hasim -deleteclus <clus>`	Deletes the specified cluster. Deleting the cluster removes all files and directories associated with the cluster. Before deleting a cluster, make sure the cluster is not configured as a global cluster.
`hasim -start` *`clustername_sys1`* `[-clus` *`clustername`*`] [-disablel10n]`	Starts VCS Simulator on the cluster specified by *clustername*. If you start VCS Simulator with the `-disablel10n` option, the simulated cluster does not accept localized values for attributes. Use this option when simulating a UNIX configuration on a Windows system to prevent potential corruption when importing the simulated configuration to a UNIX cluster.
`hasim -stop`	Stops the simulation process.
`hasim -poweroff` *`system_name`*	Gracefully shuts down the system.
`hasim -up` *`system_name`*	Brings the system up.
`hasim -fault` *`system_name resource_name`*	Faults the specified resource on the specified system.

Table 7-3 VCS simulator commands *(continued)*

Command	Description
`hasim -faultcluster` *`clustername`*	Simulates a cluster fault.
`hasim -clearcluster` *`clustername`*	Clears a simulated cluster fault.
`hasim -getsimconfig` *`cluster_name`*	Retrieves information about VCS Simulator ports.
`hasim -hb [..]`	Equivalent to standard `hahb` command.
`hasim -disablel10n`	Disables localized inputs for attribute values. Use this option when simulating UNIX configurations on Windows systems.
`hasim -clus [...]`	Equivalent to standard `haclus` command.
`hasim -sys [...]`	Equivalent to standard `hasys` command.
`hasim -grp [...]`	Equivalent to standard `hagrp` command.
`hasim -res [...]`	Equivalent to standard `hares` command.
`hasim -type [...]`	Equivalent to standard `hatype` command.
`hasim -conf [...]`	Equivalent to standard `haconf` command.
`hasim -attr [...]`	Equivalent to standard `haattr` command.

Section 3

VCS communication and operations

- Chapter 8. About communications, membership, and data protection in the cluster
- Chapter 9. Administering I/O fencing
- Chapter 10. Controlling VCS behavior
- Chapter 11. The role of service group dependencies

Chapter 8

About communications, membership, and data protection in the cluster

This chapter includes the following topics:

- About cluster communications
- About cluster membership
- About membership arbitration
- About data protection
- About I/O fencing configuration files
- Examples of VCS operation with I/O fencing
- About cluster membership and data protection without I/O fencing
- Examples of VCS operation without I/O fencing
- Summary of best practices for cluster communications

About cluster communications

VCS uses local communications on a system and system-to-system communications.

See “About intra-system communications” on page 236.

See “About inter-system cluster communications” on page 236.

About intra-system communications

Within a system, the VCS engine (VCS High Availability Daemon) uses a VCS-specific communication protocol known as Inter Process Messaging (IPM) to communicate with the GUI, the command line, and the agents.

Figure 8-1 shows basic communication on a single VCS system. Note that agents only communicate with High Availability Daemon (HAD) and never communicate with each other.

Figure 8-1 Basic communication on a single VCS system

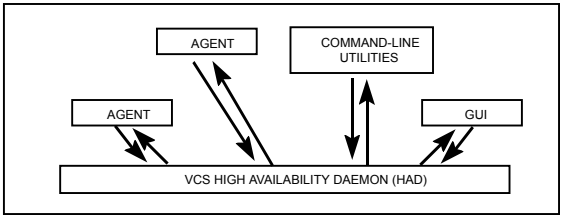
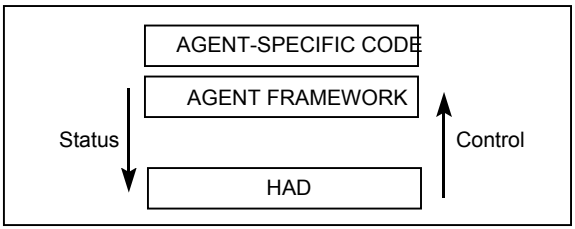


Figure 8-2 depicts communication from a single agent to HAD.

Figure 8-2 Communication from a single agent to HAD



The agent uses the agent framework, which is compiled into the agent itself. For each resource type configured in a cluster, an agent runs on each cluster system. The agent handles all resources of that type. The engine passes commands to the agent and the agent returns the status of command execution. For example, an agent is commanded to bring a resource online. The agent responds back with the success (or failure) of the operation. Once the resource is online, the agent communicates with the engine only if this status changes.

About inter-system cluster communications

VCS uses the cluster interconnect for network communications between cluster systems. Each system runs as an independent unit and shares information at the cluster level. On each system the VCS High Availability Daemon (HAD), which has the decision logic for the cluster, maintains a view of the cluster configuration. This

daemon operates as a replicated state machine, which means all systems in the cluster have a synchronized state of the cluster configuration. This is accomplished by the following:

- All systems run an identical version of HAD.
- HAD on each system maintains the state of its own resources, and sends all cluster information about the local system to all other machines in the cluster.
- HAD on each system receives information from the other cluster systems to update its own view of the cluster.
- Each system follows the same code path for actions on the cluster.

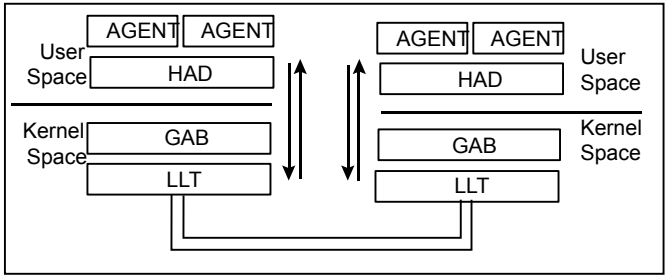
The replicated state machine communicates over a purpose-built communications package consisting of two components, Group Membership Services/Atomic Broadcast (GAB) and Low Latency Transport (LLT).

See "About Group Membership Services/Atomic Broadcast (GAB)" on page 237.

See "About Low Latency Transport (LLT)" on page 238.

Figure 8-3 illustrates the overall communications paths between two systems of the replicated state machine model.

Figure 8-3 Cluster communications with replicated state machine



About Group Membership Services/Atomic Broadcast (GAB)

The Group Membership Services/Atomic Broadcast protocol (GAB) is responsible for cluster membership and reliable cluster communications.

GAB has the following two major functions:

- Cluster membership
  GAB maintains cluster membership by receiving input on the status of the heartbeat from each system via LLT. When a system no longer receives heartbeats from a cluster peer, LLT passes the heartbeat loss notification to

GAB. GAB marks the peer as DOWN and excludes it from the cluster. In most configurations, membership arbitration is used to prevent network partitions.

- Cluster communications
  GAB's second function is reliable cluster communications. GAB provides ordered guaranteed delivery of messages to all cluster systems. The Atomic Broadcast functionality is used by HAD to ensure that all systems within the cluster receive all configuration change messages, or are rolled back to the previous state, much like a database atomic commit. While the communications function in GAB is known as Atomic Broadcast, no actual network broadcast traffic is generated. An Atomic Broadcast message is a series of point to point unicast messages from the sending system to each receiving system, with a corresponding acknowledgement from each receiving system.

About Low Latency Transport (LLT)

The Low Latency Transport protocol is used for all cluster communications as a high-performance, low-latency replacement for the IP stack.

LLT has the following two major functions:

- Traffic distribution
  LLT provides the communications backbone for GAB. LLT distributes (load balances) inter-system communication across all configured network links. This distribution ensures all cluster communications are evenly distributed across all network links for performance and fault resilience. If a link fails, traffic is redirected to the remaining links. A maximum of eight network links are supported.
- Heartbeat
  LLT is responsible for sending and receiving heartbeat traffic over each configured network link. The heartbeat traffic is point to point unicast. LLT uses ethernet broadcast to learn the address of the nodes in the cluster. All other cluster communications, including all status and configuration traffic is point to point unicast. The heartbeat is used by the Group Membership Services to determine cluster membership.

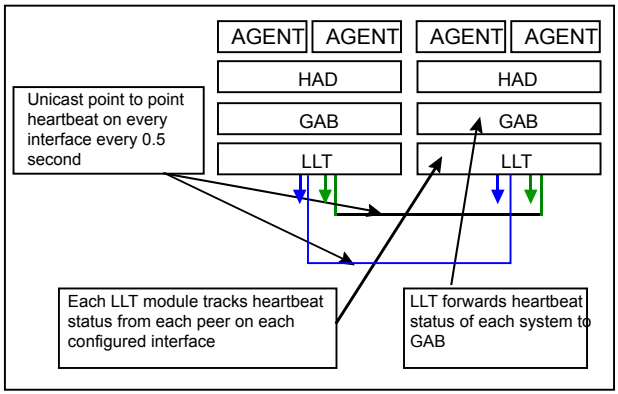
The heartbeat signal is defined as follows:

- LLT on each system in the cluster sends heartbeat packets out on all configured LLT interfaces every half second.
- LLT on each system tracks the heartbeat status from each peer on each configured LLT interface.
- LLT on each system forwards the heartbeat status of each system in the cluster to the local Group Membership Services function of GAB.

- GAB receives the status of heartbeat from all cluster systems from LLT and makes membership determination based on this information.

Figure 8-4 shows heartbeat in the cluster.

Figure 8-4 Heartbeat in the cluster



LLT can be configured to designate specific cluster interconnect links as either high priority or low priority. High priority links are used for cluster communications to GAB as well as heartbeat signals. Low priority links, during normal operation, are used for heartbeat and link state maintenance only, and the frequency of heartbeats is reduced to 50% of normal to reduce network overhead.

If there is a failure of all configured high priority links, LLT will switch all cluster communications traffic to the first available low priority link. Communication traffic will revert back to the high priority links as soon as they become available.

While not required, best practice recommends to configure at least one low priority link, and to configure two high priority links on dedicated cluster interconnects to provide redundancy in the communications path. Low priority links are typically configured on the public or administrative network.

If you use different media speed for the private NICs, Veritas recommends that you configure the NICs with lesser speed as low-priority links to enhance LLT performance. With this setting, LLT does active-passive load balancing across the private links. At the time of configuration and failover, LLT automatically chooses the link with high-priority as the active link and uses the low-priority links only when a high-priority link fails.

LLT sends packets on all the configured links in weighted round-robin manner. LLT uses the linkburst parameter which represents the number of back-to-back packets that LLT sends on a link before the next link is chosen. In addition to the default weighted round-robin based load balancing, LLT also provides destination-based load balancing. LLT implements destination-based load balancing where the LLT

link is chosen based on the destination node id and the port. With destination-based load balancing, LLT sends all the packets of a particular destination on a link. However, a potential problem with the destination-based load balancing approach is that LLT may not fully utilize the available links if the ports have dissimilar traffic. Veritas recommends destination-based load balancing when the setup has more than two cluster nodes and more active LLT ports. You must manually configure destination-based load balancing for your cluster to set up the port to LLT link mapping.

See "Configuring destination-based load balancing for LLT" on page 107.

LLT on startup sends broadcast packets with LLT node id and cluster id information onto the LAN to discover any node in the network that has same node id and cluster id pair. Each node in the network replies to this broadcast message with its cluster id, node id, and node name.

LLT on the original node does not start and gives appropriate error in the following cases:

- LLT on any other node in the same network is running with the same node id and cluster id pair that it owns.
- LLT on the original node receives response from a node that does not have a node name entry in the `/etc/llthosts` file.

About cluster membership

The current members of the cluster are the systems that are actively participating in the cluster. It is critical for Cluster Server engine (HAD) to accurately determine current cluster membership in order to take corrective action on system failure and maintain overall cluster topology.

A change in cluster membership is one of the starting points of the logic to determine if HAD needs to perform any fault handling in the cluster.

There are two aspects to cluster membership, initial joining of the cluster and how membership is determined once the cluster is up and running.

Initial joining of systems to cluster membership

When the cluster initially boots, LLT determines which systems are sending heartbeat signals, and passes that information to GAB. GAB uses this information in the process of seeding the cluster membership.

Seeding a new cluster

Seeding ensures a new cluster will start with an accurate membership count of the number of systems in the cluster. This prevents the possibility of one cluster splitting into multiple subclusters upon initial startup.

A new cluster can be automatically seeded as follows:

- When the cluster initially boots, all systems in the cluster are unseeded.
- GAB checks the number of systems that have been declared to be members of the cluster in the `/etc/gabtab` file.

  The number of systems declared in the cluster is denoted as follows:

  ```
  /sbin/gabconfig -c -nN
  ```

  where the variable # is replaced with the number of systems in the cluster.

  Note: Veritas recommends that you replace # with the exact number of nodes in the cluster.

- When GAB on each system detects that the correct number of systems are running, based on the number declared in `/etc/gabtab` and input from LLT, it will seed.
- If you have I/O fencing enabled in your cluster and if you have set the GAB auto-seeding feature through I/O fencing, GAB automatically seeds the cluster even when some cluster nodes are unavailable.

  See "Seeding a cluster using the GAB auto-seed parameter through I/O fencing" on page 241.
- HAD will start on each seeded system. HAD will only run on a system that has seeded.

  HAD can provide the HA functionality only when GAB has seeded.

See "Manual seeding of a cluster" on page 242.

Seeding a cluster using the GAB auto-seed parameter through I/O fencing

If some of the nodes are not up and running in a cluster, then GAB port does not come up to avoid any risks of preexisting split-brain. In such cases, you can manually seed GAB using the command `gabconfig -x` to bring the GAB port up.

See "Manual seeding of a cluster" on page 242.

However, if you have enabled I/O fencing in the cluster, then I/O fencing can handle any preexisting split-brain in the cluster. You can configure I/O fencing in such a way for GAB to automatically seed the cluster. The behavior is as follows:

- If a number of nodes in a cluster are not up, GAB port (port a) still comes up in all the member-nodes in the cluster.
- If the coordination points do not have keys from any non-member nodes, I/O fencing (GAB port b) also comes up.

This feature is disabled by default. You must configure the `autoseed_gab_timeout` parameter in the `/etc/vxfenmode` file to enable the automatic seeding feature of GAB.

See "About I/O fencing configuration files" on page 266.

To enable GAB auto-seeding parameter of I/O fencing

◆ Set the value of the `autoseed_gab_timeout` parameter in the `/etc/vxfenmode` file to 0 to turn on the feature.

  To delay the GAB auto-seed feature, you can also set a value greater than zero. GAB uses this value to delay auto-seed of the cluster for the given number of seconds.

To disable GAB auto-seeding parameter of I/O fencing

◆ Set the value of the `autoseed_gab_timeout` parameter in the `/etc/vxfenmode` file to -1 to turn off the feature.

  You can also remove the line from the `/etc/vxfenmode` file.

Manual seeding of a cluster

Seeding the cluster manually is appropriate when the number of cluster systems declared in `/etc/gabtab` is more than the number of systems that will join the cluster.

This can occur if a system is down for maintenance when the cluster comes up.

Warning: It is not recommended to seed the cluster manually unless the administrator is aware of the risks and implications of the command.

Note:

If you have I/O fencing enabled in your cluster, you can set the GAB auto-seeding feature through I/O fencing so that GAB automatically seeds the cluster even when some cluster nodes are unavailable.

See "Seeding a cluster using the GAB auto-seed parameter through I/O fencing" on page 241.

Before manually seeding the cluster, check that systems that will join the cluster are able to send and receive heartbeats to each other. Confirm there is no possibility of a network partition condition in the cluster.

Before manually seeding the cluster, do the following:

- Check that systems that will join the cluster are able to send and receive heartbeats to each other.
- Confirm there is no possibility of a network partition condition in the cluster.

To manually seed the cluster, type the following command:

```
/sbin/gabconfig -x
```

Note there is no declaration of the number of systems in the cluster with a manual seed. This command will seed all systems in communication with the system where the command is run.

So, make sure not to run this command in more than one node in the cluster.

See "Seeding and I/O fencing" on page 626.

Stale key detection to avoid a false preexisting split brain condition

When a cluster starts, all the cluster nodes register their unique keys on the coordination point servers (CP servers). The key is unique to the cluster and the node, and is based on the LLT cluster ID and the LLT node ID. When the network connectivity between the cluster nodes and the CP servers is lost, as the nodes go offline, some stale keys may be left on one or more CP servers. The presence of a key on a CP server indicates that the corresponding node is alive. Thus, after the network connectivity is restored, the presence of a stale key of a node that freshly comes online creates a false preexisting split brain in the cluster. When a false preexisting split brain is detected, the node fails to start, and manual intervention is required to bring the cluster online.

Cluster Server provides a reliable method to identify a false preexisting split brain condition that is caused by the presence of such stale keys. The CP server

functionality detects such stale keys and provide this information to the `vxfen` module. The `vxfen` module performs its arbitration and brings the cluster online.

This feature is available only on Linux when the following criteria are met:

- `vxfen_mode` is set to `customized`.
- Majority of the coordination points are servers.
- `detect_false_pesb` attribute is set to `1`.

To enable stale key detection in a server-based fencing configuration

1. Stop VCS on all nodes.

   ```
   # hastop -all
   ```

2. Make sure that the port `h` is closed on all the nodes. Run the following command on each node to verify that the port `h` is closed:

   ```
   # gabconfig -a
   ```

   Port `h` must not appear in the output.

3. Stop I/O fencing on all nodes. Run the following command on each node:

   ```
   # /etc/init.d/vxfen stop
   ```

4. If you have any applications that run outside of VCS control that have access to the shared storage, shut down all the other nodes in the cluster that have access to the shared storage. This prevents data corruption.

5. Open the `/etc/vxfenmode` file, add `detect_false_pesb=1` at the appropriate location, and save and close the file.

6. Start the fencing module on all the nodes.

   ```
   # /etc/init.d/vxfen start
   ```

7. Start VCS on all nodes.

   ```
   # hastart
   ```

See "About I/O fencing configuration files" on page 266.

Ongoing cluster membership

Once the cluster is up and running, a system remains an active member of the cluster as long as peer systems receive a heartbeat signal from that system over the cluster interconnect. A change in cluster membership is determined as follows:

- When LLT on a system no longer receives heartbeat messages from a system on any of the configured LLT interfaces for a predefined time (peerinact), LLT informs GAB of the heartbeat loss from that specific system.

This predefined time is 16 seconds by default, but can be configured.
You can set this predefined time with the `set-timer peerinact` command.
See the `lltab` manual page.

Note: When you configure InfoScale cluster in an Azure environment and enable I/O fencing, you must increase the `llt-peerinact` time to 120 seconds. Each time a cluster node reboots, the fencing module checks whether a network partition has occurred and performs I/O fencing if required. When you reboot a cluster node using the Azure portal, the reboot takes longer than the fencing module takes to complete its operations. As a result, the node that reboots or any other node may panic. To avoid this situation, you must increase the `llt-peerinact` time.

- When LLT informs GAB of a heartbeat loss, the systems that are remaining in the cluster coordinate to agree which systems are still actively participating in the cluster and which are not. This happens during a time period known as GAB Stable Timeout (5 seconds).
  VCS has specific error handling that takes effect in the case where the systems do not agree.
- GAB marks the system as DOWN, excludes the system from the cluster membership, and delivers the membership change to the fencing module.
- The fencing module performs membership arbitration to ensure that there is not a split brain situation and only one functional cohesive cluster continues to run.

The fencing module is turned on by default.

Review the details on actions that occur if the fencing module has been deactivated:

See "About cluster membership and data protection without I/O fencing" on page 276.

About membership arbitration

Membership arbitration is necessary on a perceived membership change because systems may falsely appear to be down. When LLT on a system no longer receives heartbeat messages from another system on any configured LLT interface, GAB marks the system as DOWN. However, if the cluster interconnect network failed, a system can appear to be failed when it actually is not. In most environments when this happens, it is caused by an insufficient cluster interconnect network infrastructure, usually one that routes all communication links through a single point of failure.

If all the cluster interconnect links fail, it is possible for one cluster to separate into two subclusters, each of which does not know about the other subcluster. The two

subclusters could each carry out recovery actions for the departed systems. This condition is termed split brain.

In a split brain condition, two systems could try to import the same storage and cause data corruption, have an IP address up in two places, or mistakenly run an application in two places at once.

Membership arbitration guarantees against such split brain conditions.

About membership arbitration components

The components of membership arbitration are the fencing module and the coordination points.

See "About the fencing module" on page 246.

See "About coordination points" on page 246.

About the fencing module

Each system in the cluster runs a kernel module called vxfen, or the fencing module. This module is responsible for ensuring valid and current cluster membership during a membership change through the process of membership arbitration.

vxfen performs the following actions:

- Registers with the coordination points during normal operation
- Races for control of the coordination points during membership changes

About coordination points

Coordination points provide a lock mechanism to determine which nodes get to fence off data drives from other nodes. A node must eject a peer from the coordination points before it can fence the peer from the data drives. VCS prevents split-brain when vxfen races for control of the coordination points and the winner partition fences the ejected nodes from accessing the data disks.

Note: Typically, a fencing configuration for a cluster must have three coordination points. Veritas Technologies also supports server-based fencing with a single CP server as its only coordination point with a caveat that this CP server becomes a single point of failure.

The coordination points can either be disks or servers or both.

- Coordinator disks

Disks that act as coordination points are called coordinator disks. Coordinator disks are three standard disks or LUNs set aside for I/O fencing during cluster reconfiguration. Coordinator disks do not serve any other storage purpose in the VCS configuration.

You can configure coordinator disks to use Veritas Volume Manager's Dynamic Multi-pathing (DMP) feature. Dynamic Multi-pathing (DMP) allows coordinator disks to take advantage of the path failover and the dynamic adding and removal capabilities of DMP. So, you can configure I/O fencing to use DMP devices. I/O fencing uses SCSI-3 disk policy that is dmp-based on the disk device that you use.

Note: The dmp disk policy for I/O fencing supports both single and multiple hardware paths from a node to the coordinator disks. If few coordinator disks have multiple hardware paths and few have a single hardware path, then we support only the dmp disk policy. For new installations, Veritas Technologies only supports dmp disk policy for IO fencing even for a single hardware path.

See the *Storage Foundation Administrator's Guide*.

- Coordination point servers

  The coordination point server (CP server) is a software solution which runs on a remote system or cluster. CP server provides arbitration functionality by allowing the SFHA cluster nodes to perform the following tasks:

  - Self-register to become a member of an active VCS cluster (registered with CP server) with access to the data drives
  - Check which other nodes are registered as members of this active VCS cluster
  - Self-unregister from this active VCS cluster
  - Forcefully unregister other nodes (preempt) as members of this active VCS cluster

  In short, the CP server functions as another arbitration mechanism that integrates within the existing I/O fencing module.

Note: With the CP server, the fencing arbitration logic still remains on the VCS cluster.

Multiple VCS clusters running different operating systems can simultaneously access the CP server. TCP/IP based communication is used between the CP server and the VCS clusters.

About preferred fencing

The I/O fencing driver uses coordination points to prevent split-brain in a VCS cluster. By default, the fencing driver favors the subcluster with maximum number of nodes during the race for coordination points. With the preferred fencing feature, you can specify how the fencing driver must determine the surviving subcluster.

You can configure the preferred fencing policy using the cluster-level attribute PreferredFencingPolicy for the following:

- Enable system-based preferred fencing policy to give preference to high capacity systems.
- Enable group-based preferred fencing policy to give preference to service groups for high priority applications.
- Enable site-based preferred fencing policy to give preference to sites with higher priority.
- Disable preferred fencing policy to use the default node count-based race policy.

See "How preferred fencing works" on page 251.

See "Enabling or disabling the preferred fencing policy" on page 347.

How the fencing module starts up

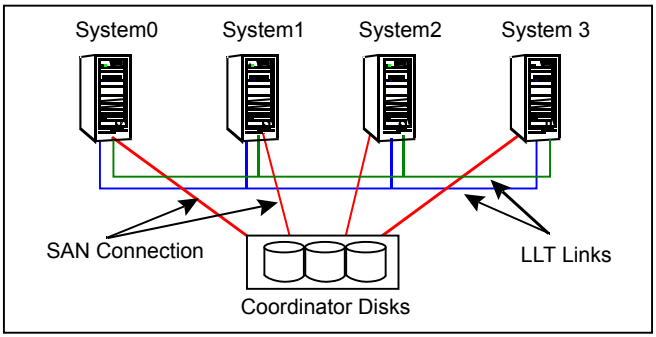
The fencing module starts up as follows:

- The coordinator disks are placed in a disk group.
  This allows the fencing startup script to use Veritas Volume Manager (VxVM) commands to easily determine which disks are coordinator disks, and what paths exist to those disks. This disk group is never imported, and is not used for any other purpose.
- The fencing start up script on each system uses VxVM commands to populate the file /etc/vxfentab with the paths available to the coordinator disks.
  See "About I/O fencing configuration files" on page 266.
- When the fencing driver is started, it reads the physical disk names from the /etc/vxfentab file. Using these physical disk names, it determines the serial numbers of the coordinator disks and builds an in-memory list of the drives.
- The fencing driver verifies that the systems that are already running in the cluster see the same coordinator disks.
  The fencing driver examines GAB port B for membership information. If no other system is up and running, it is the first system up and is considered to have the correct coordinator disk configuration. When a new member joins, it requests the coordinator disks configuration. The system with the lowest LLT ID will respond with a list of the coordinator disk serial numbers. If there is a match,

the new member joins the cluster. If there is not a match, vxfen enters an error state and the new member is not allowed to join. This process ensures all systems communicate with the same coordinator disks.

- The fencing driver determines if a possible preexisting split brain condition exists. This is done by verifying that any system that has keys on the coordinator disks can also be seen in the current GAB membership. If this verification fails, the fencing driver prints a warning to the console and system log and does not start.
- If all verifications pass, the fencing driver on each system registers keys with each coordinator disk.

Figure 8-5 Topology of coordinator disks in the cluster



How membership arbitration works

Upon startup of the cluster, all systems register a unique key on the coordinator disks. The key is unique to the cluster and the node, and is based on the LLT cluster ID and the LLT system ID.

See "About the I/O fencing registration key format" on page 295.

When there is a perceived change in membership, membership arbitration works as follows:

- GAB marks the system as DOWN, excludes the system from the cluster membership, and delivers the membership change—the list of departed systems—to the fencing module.
- The system with the lowest LLT system ID in the cluster races for control of the coordinator disks
  - In the most common case, where departed systems are truly down or faulted, this race has only one contestant.
  - In a split brain scenario, where two or more subclusters have formed, the race for the coordinator disks is performed by the system with the lowest

LLT system ID of that subcluster. This system that races on behalf of all the other systems in its subcluster is called the RACER node and the other systems in the subcluster are called the SPECTATOR nodes.

- During the I/O fencing race, if the RACER node panics or if it cannot reach the coordination points, then the VxFEN RACER node re-election feature allows an alternate node in the subcluster that has the next lowest node ID to take over as the RACER node.

  The racer re-election works as follows:

  - In the event of an unexpected panic of the RACER node, the VxFEN driver initiates a racer re-election.
  - If the RACER node is unable to reach a majority of coordination points, then the VxFEN module sends a RELAY_RACE message to the other nodes in the subcluster. The VxFEN module then re-elects the next lowest node ID as the new RACER.
  - With successive re-elections if no more nodes are available to be re-elected as the RACER node, then all the nodes in the subcluster will panic.

- The race consists of executing a preempt and abort command for each key of each system that appears to no longer be in the GAB membership.

  The preempt and abort command allows only a registered system with a valid key to eject the key of another system. This ensures that even when multiple systems attempt to eject other, each race will have only one winner. The first system to issue a preempt and abort command will win and eject the key of the other system. When the second system issues a preempt and abort command, it cannot perform the key eject because it is no longer a registered system with a valid key.

  If the value of the cluster-level attribute PreferredFencingPolicy is System, Group, or Site then at the time of a race, the VxFEN Racer node adds up the weights for all nodes in the local subcluster and in the leaving subcluster. If the leaving partition has a higher sum (of node weights) then the racer for this partition will delay the race for the coordination point. This effectively gives a preference to the more critical subcluster to win the race. If the value of the cluster-level attribute PreferredFencingPolicy is Disabled, then the delay will be calculated, based on the sums of node counts.

  See “About preferred fencing” on page 248.

- If the preempt and abort command returns success, that system has won the race for that coordinator disk.

  Each system will repeat this race to all the coordinator disks. The race is won by, and control is attained by, the system that ejects the other system’s registration keys from a majority of the coordinator disks.

- On the system that wins the race, the vxfen module informs all the systems that it was racing on behalf of that it won the race, and that subcluster is still valid.
- On the system(s) that do not win the race, the vxfen module will trigger a system panic. The other systems in this subcluster will note the panic, determine they lost control of the coordinator disks, and also panic and restart.
- Upon restart, the systems will attempt to seed into the cluster.
  - If the systems that restart can exchange heartbeat with the number of cluster systems declared in /etc/gabtab, they will automatically seed and continue to join the cluster. Their keys will be replaced on the coordinator disks. This case will only happen if the original reason for the membership change has cleared during the restart.
  - If the systems that restart cannot exchange heartbeat with the number of cluster systems declared in /etc/gabtab, they will not automatically seed, and HAD will not start. This is a possible split brain condition, and requires administrative intervention.
  - If you have I/O fencing enabled in your cluster and if you have set the GAB auto-seeding feature through I/O fencing, GAB automatically seeds the cluster even when some cluster nodes are unavailable.
    See "Seeding a cluster using the GAB auto-seed parameter through I/O fencing" on page 241.

Note: Forcing a manual seed at this point will allow the cluster to seed. However, when the fencing module checks the GAB membership against the systems that have keys on the coordinator disks, a mismatch will occur. vxfen will detect a possible split brain condition, print a warning, and will not start. In turn, HAD will not start. Administrative intervention is required.

See "Manual seeding of a cluster" on page 242.

How preferred fencing works

The I/O fencing driver uses coordination points to prevent split-brain in a VCS cluster. At the time of a network partition, the fencing driver in each subcluster races for the coordination points. The subcluster that grabs the majority of coordination points survives whereas the fencing driver causes a system panic on nodes from all other subclusters. By default, the fencing driver favors the subcluster with maximum number of nodes during the race for coordination points.

This default racing preference does not take into account the application groups that are online on any nodes or the system capacity in any subcluster. For example, consider a two-node cluster where you configured an application on one node and

the other node is a standby-node. If there is a network partition and the standby-node wins the race, the node where the application runs panics and VCS has to bring the application online on the standby-node. This behavior causes disruption and takes time for the application to fail over to the surviving node and then to start up again.

The preferred fencing feature lets you specify how the fencing driver must determine the surviving subcluster. The preferred fencing solution makes use of a fencing parameter called node weight. VCS calculates the node weight based on online applications, system capacity, and site preference details that you provide using specific VCS attributes, and passes to the fencing driver to influence the result of race for coordination points. At the time of a race, the racer node adds up the weights for all nodes in the local subcluster and in the leaving subcluster. If the leaving subcluster has a higher sum (of node weights) then the racer for this subcluster delays the race for the coordination points. Thus, the subcluster that has critical systems or critical applications wins the race.

The preferred fencing feature uses the cluster-level attribute PreferredFencingPolicy that takes the following race policy values:

- Disabled (default): Preferred fencing is disabled.
  When the PreferredFencingPolicy attribute value is set as Disabled, VCS sets the count based race policy and resets the value of node weight as 0.
- System: Based on the capacity of the systems in a subcluster.
  If one system is more powerful than others in terms of architecture, number of CPUs, or memory, this system is given preference in the fencing race.
  When the PreferredFencingPolicy attribute value is set as System, VCS calculates node weight based on the system-level attribute FencingWeight.
  See System attributes on page 746.
- Group: Based on the higher priority applications in a subcluster.
  The fencing driver takes into account the service groups that are online on the nodes in any subcluster.
  In the event of a network partition, I/O fencing determines whether the VCS engine is running on all the nodes that participate in the race for coordination points. If VCS engine is running on all the nodes, the node with higher priority service groups is given preference during the fencing race.
  However, if the VCS engine instance on a node with higher priority service groups is killed for some reason, I/O fencing resets the preferred fencing node weight for that node to zero. I/O fencing does not prefer that node for membership arbitration. Instead, I/O fencing prefers a node that has an instance of VCS engine running on it even if the node has lesser priority service groups.
  Without synchronization between VCS engine and I/O fencing, a node with high priority service groups but without VCS engine running on it may win the race.

Such a situation means that the service groups on the loser node cannot failover to the surviving node.

When the PreferredFencingPolicy attribute value is set as Group, VCS calculates node weight based on the group-level attribute Priority for those service groups that are active.

See "Service group attributes" on page 719.

- Site: Based on the priority assigned to sites in a subcluster.

  The Site policy is available only if you set the cluster attribute SiteAware to 1. VCS sets higher weights to the nodes in a higher priority site and lesser weights to the nodes in a lower priority site. The site with highest cumulative node weight becomes the preferred site. In a network partition between sites, VCS prefers the subcluster with nodes from the preferred site in the race for coordination points.

  See "Site attributes" on page 780.

See "Enabling or disabling the preferred fencing policy" on page 347.

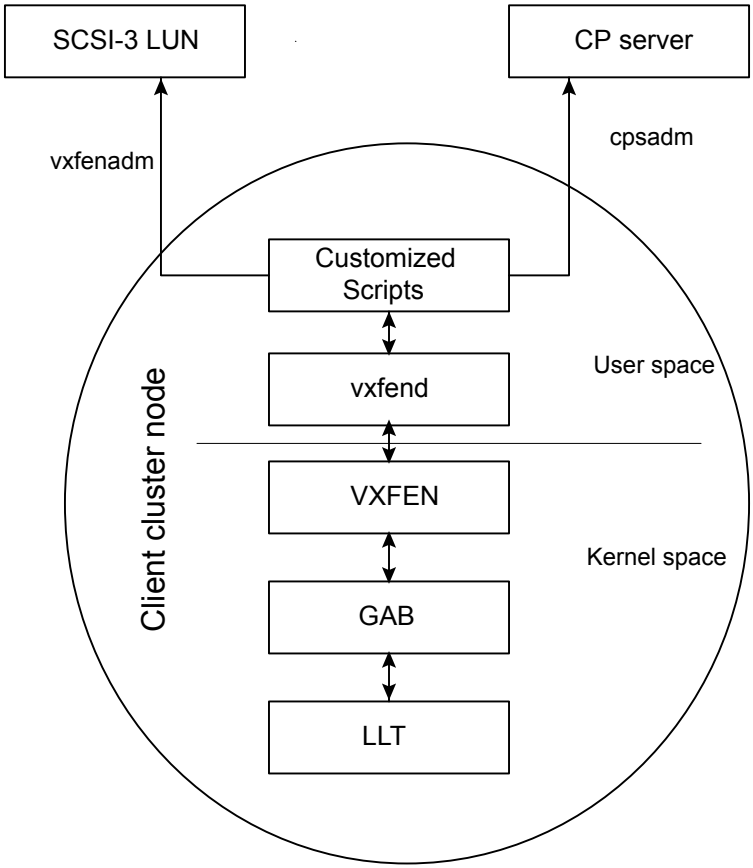
About server-based I/O fencing

In a disk-based I/O fencing implementation, the vxfen driver handles various SCSI-3 PR based arbitration operations completely within the driver. I/O fencing also provides a framework referred to as customized fencing wherein arbitration operations are implemented in custom scripts. The vxfen driver invokes the custom scripts.

The CP server-based coordination point uses a customized fencing framework. Note that SCSI-3 PR based fencing arbitration can also be enabled using customized fencing framework. This allows the user to specify a combination of SCSI-3 LUNs and CP servers as coordination points using customized fencing. Customized fencing can be enabled by specifying vxfen_mode=customized and vxfen_mechanism=cps in the `/etc/vxfenmode` file.

Figure 8-6 displays a schematic of the customized fencing options.

Figure 8-6 Customized fencing



A user level daemon vxfend interacts with the vxfen driver, which in turn interacts with GAB to get the node membership update. Upon receiving membership updates, vxfend invokes various scripts to race for the coordination point and fence off data disks. The vxfend daemon manages various fencing agents. The customized fencing scripts are located in the `/opt/VRTSvcs/vxfen/bin/customized/cps` directory.

The scripts that are involved include the following:

- generate_snapshot.sh : Retrieves the SCSI ID's of the coordinator disks and/or UUID ID's of the CP servers
  CP server uses the UUID stored in `/etc/VRTScps/db/current/cps_uuid`.
  See "About the cluster UUID" on page 35.
- join_local_node.sh: Registers the keys with the coordinator disks or CP servers

- race_for_coordination_point.sh: Races to determine a winner after cluster reconfiguration
- unjoin_local_node.sh: Removes the keys that are registered in join_local_node.sh
- fence_data_disks.sh: Fences the data disks from access by the losing nodes.
- local_info.sh: Lists local node's configuration parameters and coordination points, which are used by the vxfen driver.
- validate_pesb_join.sh:
  - Determines whether the preexisting split brain is a true condition or a false alarm due to the presence of stale keys on the coordination points.
  - Clears the coordination points if the preexisting split brain turns out to be a false alarm.

I/O fencing enhancements provided by CP server

CP server configurations enhance disk-based I/O fencing by providing the following new capabilities:

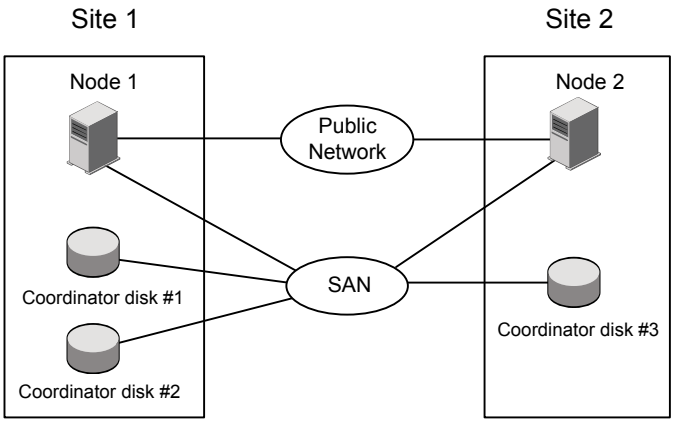
- CP server configurations are scalable, and a configuration with three CP servers can provide I/O fencing for multiple VCS clusters. Since a single CP server configuration can serve a large number of VCS clusters, the cost of multiple VCS cluster deployments can be significantly reduced.
- Appropriately situated CP servers can eliminate any coordinator disk location bias in the I/O fencing process. For example, this location bias may occur where, due to logistical restrictions, two of the three coordinator disks are located at a single site, and the cost of setting up a third coordinator disk location is prohibitive.

  See Figure 8-7 on page 256.

  In such a configuration, if the site with two coordinator disks is inaccessible, the other site does not survive due to a lack of a majority of coordination points. I/O fencing would require extension of the SAN to the third site which may not be a suitable solution. An alternative is to place a CP server at a remote site as the third coordination point.

Note: The CP server provides an alternative arbitration mechanism without having to depend on SCSI-3 compliant coordinator disks. Data disk fencing in Cluster Volume Manager (CVM) will still require SCSI-3 I/O fencing.

Figure 8-7 Skewed placement of coordinator disks at Site 1



About majority-based fencing

The majority-based fencing mode is a fencing mechanism that does not need coordination points. You do not need shared disks or free nodes to be used as coordination points or servers.

The fencing mechanism provides an arbitration method that is reliable and does not require any extra hardware setup., such as CP Servers or shared SCSI3 disks. In a split brain scenario, arbitration is done based on `majority` number of nodes among the sub-clusters.

Majority-based fencing: If N is defined as the total number of nodes (current membership) in the cluster, then majority is equal to N/2 + 1.

Leader node: The node with the lowest node ID in the cluster (before the split brain happened) is called the leader node. This plays a role in case of a tie.

How majority-based I/O fencing works

When a network partition happens, one node in each sub-cluster is elected as the racer node, while the other nodes are designated as spectator nodes. As majority-based fencing does not use coordination points, sub-clusters do not engage in an actual race to decide the winner after a split brain scenario. The sub-cluster with the majority number of nodes survives while nodes in the rest of the sub-clusters are taken offline.

The following algorithm is used to decide the winner sub-cluster:

1. The node with the lowest node ID in the current gab membership, before the network partition, is designated as the leader node in the fencing race.
2. When a network partition occurs, each racer sub-cluster computes the number of nodes in its partition and compares it with that of the leaving partition.
3. If a racer finds that its partition does not have majority, it sends a LOST_RACE message to all the nodes in its partition including itself and all the nodes panic. On the other hand, if the racer finds that it does have majority, it sends a WON_RACE message to all the nodes. Thus, the partition with majority nodes survives.

Deciding cluster majority for majority-based I/O fencing mechanism

Considerations to decide cluster majority in the event of a network partition:

1. Odd number of cluster nodes in the current membership: One sub-cluster gets majority upon a network split.
2. Even number of cluster nodes in the current membership: In case of an even network split, both the sub-clusters have equal number of nodes. The partition with the leader node is treated as majority and that partition survives.

   In case of an uneven network split, such that one sub-cluster has more number of nodes than other sub-clusters, the majority sub-cluster gets majority and survives.

About making CP server highly available

If you want to configure a multi-node CP server cluster, install and configure SFHA on the CP server nodes. Otherwise, install and configure VCS on the single node.

In both the configurations, VCS provides local start and stop of the CP server process, taking care of dependencies such as NIC, IP address, and so on. Moreover, VCS also serves to restart the CP server process in case the process faults.

VCS can use multiple network paths to access a CP server. If a network path fails, CP server does not require a restart and continues to listen on one of the other available virtual IP addresses.

To make the CP server process highly available, you must perform the following tasks:

- Install and configure SFHA on the CP server systems.
- Configure the CP server process as a failover service group.

- Configure disk-based I/O fencing to protect the shared CP server database.

Note: Veritas recommends that you do not run any other applications on the single node or SFHA cluster that is used to host CP server.

A single CP server can serve up to 128 VCS clusters. A common set of CP servers can also serve up to 128 VCS clusters.

About the CP server database

CP server requires a database for storing the registration keys of the VCS cluster nodes. CP server uses a SQLite database for its operations. By default, the database is located at /etc/VRTScps/db.

For a single node VCS cluster hosting a CP server, the database can be placed on a local file system. For an SFHA cluster hosting a CP server, the database must be placed on a shared file system. The file system must be shared among all nodes that are part of the SFHA cluster.

In an SFHA cluster hosting the CP server, the shared database is protected by setting up SCSI-3 PR based I/O fencing. SCSI-3 PR based I/O fencing protects against split-brain scenarios.

Warning: The CP server database must not be edited directly and should only be accessed using `cpsadm`(1M). Manipulating the database manually may lead to undesirable results including system panics.

Recommended CP server configurations

Following are the recommended CP server configurations:

- Multiple application clusters use three CP servers as their coordination points
  See Figure 8-8 on page 259.
- Multiple application clusters use a single CP server and single or multiple pairs of coordinator disks (two) as their coordination points
  See Figure 8-9 on page 260.
- Multiple application clusters use a single CP server as their coordination point
  This single coordination point fencing configuration must use a highly available CP server that is configured on an SFHA cluster as its coordination point.
  See Figure 8-10 on page 260.

Warning: In a single CP server fencing configuration, arbitration facility is not available during a failover of the CP server in the SFHA cluster. So, if a network partition occurs on any application cluster during the CP server failover, the application cluster is brought down.

Although the recommended CP server configurations use three coordination points, you can use more than three coordination points for I/O fencing. Ensure that the total number of coordination points you use is an odd number. In a configuration where multiple application clusters share a common set of CP server coordination points, the application cluster as well as the CP server use a Universally Unique Identifier (UUID) to uniquely identify an application cluster.

Figure 8-8 displays a configuration using three CP servers that are connected to multiple application clusters.

Figure 8-8 Three CP servers connecting to multiple application clusters

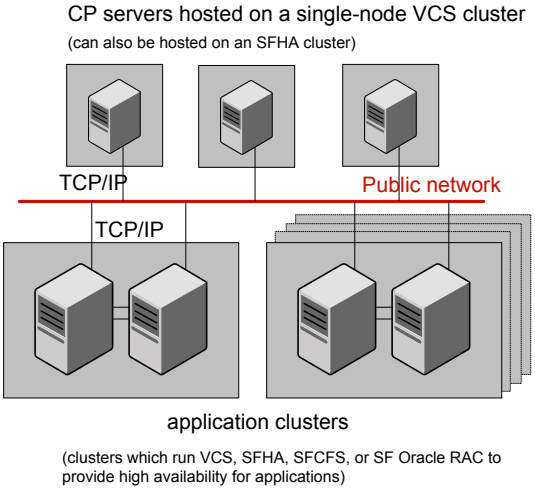


Figure 8-9 displays a configuration using a single CP server that is connected to multiple application clusters with each application cluster also using two coordinator disks.

Figure 8-9 Single CP server with two coordinator disks for each application cluster

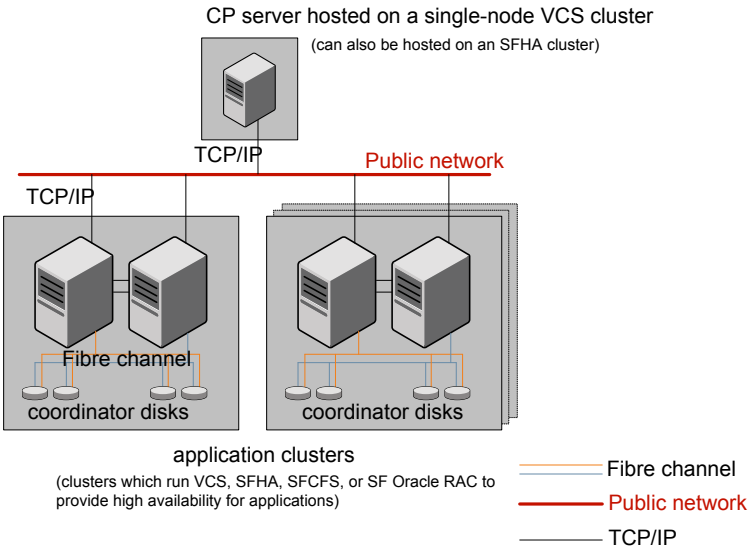
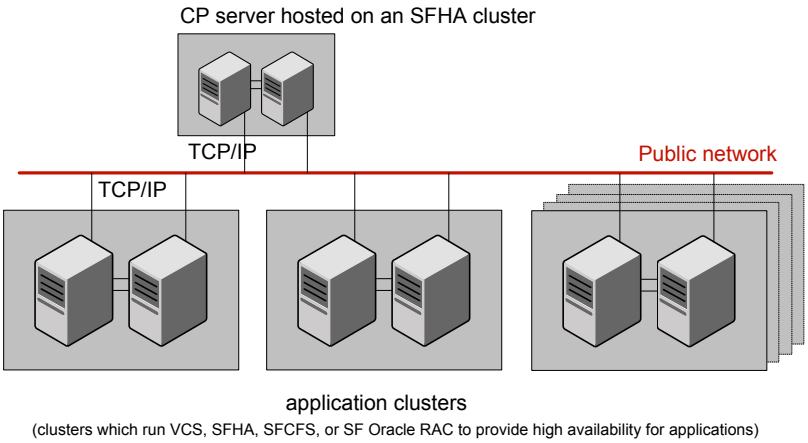


Figure 8-10 displays a configuration using a single CP server that is connected to multiple application clusters.

Figure 8-10 Single CP server connecting to multiple application clusters



About the CP server service group

The CP server service group (CPSSG) is a VCS service group. An I/O fencing configuration using CP server requires the CP server service group.

You can configure the CP server service group failover policies using the Quorum agent. CP server uses the Quorum agent to monitor the virtual IP addresses of the CP server.

See "About the Quorum agent for CP server" on page 262.

Figure 8-11 displays a schematic of the CPSSG group and its dependencies when CP server is hosted on a single node VCS cluster.

Figure 8-11 CPSSG group when CP server hosted on a single node VCS cluster

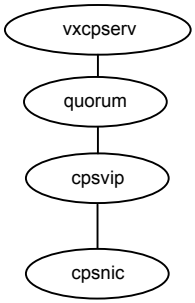
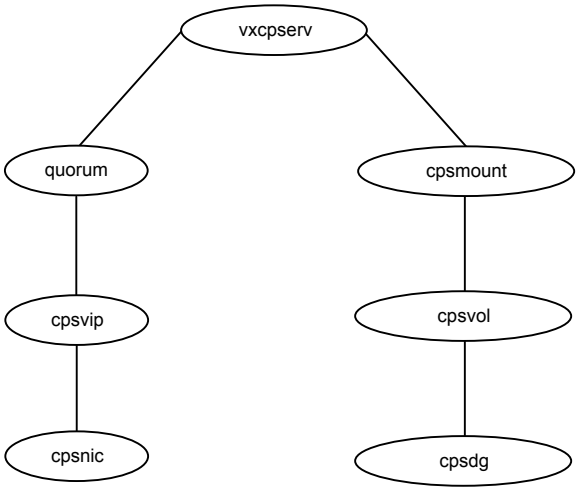


Figure 8-12 displays a schematic of the CPSSG group and its dependencies when the CP server is hosted on an SFHA cluster.

Figure 8-12 CPSSG group when the CP server is hosted on an SFHA cluster



About the Quorum agent for CP server

Coordination point server (CP server) uses the Quorum agent to monitor the virtual IP addresses of the CP server. This agent is installed as part of the VRTScps RPM. The agent's functions include online, monitor, offline, and clean.

You can configure the CP server service group failover policies using the Quorum agent.

The attributes of the Quorum agent are as follows:

- QuorumResources
  This attribute defines the IP resources that the agent must monitor.
- Quorum
  The value of the Quorum attribute determines the minimum number of virtual IP addresses that the CP server requires to come up. The default value is 1.
- State
  This attribute is used to display the log message at info level in the Quorum agent log file when Quorum reaches its threshold value. This attribute is for internal use of VCS and not editable. The log file can be found at, `/var/VRTSvcs/log/Quorum_A.log`.

The resource type definition for the Quorum agent is as follows:

```
type Quorum (
        static str ArgList[] = { QuorumResources, Quorum, State }
        str QuorumResources[]
```

```
        int Quorum = 1
)
```

About the CP server user types and privileges

The CP server supports the following user types, each with a different access level privilege:

- CP server administrator (admin)
- CP server operator

Different access level privileges permit the user to issue different commands. If a user is neither a CP server admin nor a CP server operator user, then the user has guest status and can issue limited commands.

The user types and their access level privileges are assigned to individual users during VCS cluster configuration for fencing. During the installation process, you are prompted for a user name, password, and access level privilege (CP server admin or CP server operator).

To administer and operate a CP server, there must be at least one CP server admin.

A root user on a CP server is given all the administrator privileges, and these administrator privileges can be used to perform all the CP server specific operations.

About secure communication between the VCS cluster and CP server

In a data center, TCP/IP communication between the VCS cluster (application cluster) and CP server must be made secure. The security of the communication channel involves encryption, authentication, and authorization.

The CP server node or cluster needs to confirm the authenticity of the VCS cluster nodes that communicate with it as a coordination point and only accept requests from known VCS cluster nodes. Requests from unknown clients are rejected as non-authenticated. Similarly, the fencing framework in VCS cluster must confirm that authentic users are conducting fencing operations with the CP server.

The secure mode of communication between CP server and VCS cluster is HTTPS communication.

HTTPS communication: The SSL infrastructure uses the client cluster certificates and CP server certificates to ensure that communication is secure. The HTTPS mode does not use the broker mechanism to create the authentication server credentials.

How secure communication works between the CP servers and the VCS clusters using the HTTPS protocol

HTTPS is HTTP communication over SSL/TLS (Secure Sockets Layer/Transport Layer Security). In HTTPS, the communication between client and server is secure using the Public Key Infrastructure (PKI). HTTPS is an industry standard protocol, which is widely used over the Internet for secure communication. Data encrypted using private key of an entity, can only be decrypted by using its public key. A common trusted entity, such as, the Certification Authority (CA) confirms the identities of the client and server by signing their certificates. In a CP server deployment, both the server and the clients have their own private keys, individual certificates signed by the common CA, and CA's certificate. CP server uses the SSL implementation from OpenSSL to implement HTTPS for secure communication.

CP server and VCS cluster (application cluster) node communication involve the following entities:

- vxcpserv for the CP server
- cpsadm for the VCS cluster node

Communication flow between CP server and VCS cluster nodes with security configured on them is as follows:

- Initial setup:
  Identities of CP server and VCS cluster nodes are configured on respective nodes by the VCS installer.

  Note: For secure communication using HTTPS, you do not need to establish trust between the CP server and the application cluster.

  The signed client certificate is used to establish the identity of the client. Once the CP server authenticates the client, the client can issue the operational commands that are limited to its own cluster.
- Getting credentials from authentication broker:
  The `cpsadm` command tries to get the existing credentials that are present on the local node. The installer generates these credentials during fencing configuration.
  The vxcpserv process tries to get the existing credentials that are present on the local node. The installer generates these credentials when it enables security.
- Communication between CP server and VCS cluster nodes:
  After the CP server establishes its credential and is up, it becomes ready to receive data from the clients. After the `cpsadm` command obtains its credentials and authenticates CP server credentials, cpsadm connects to the CP server. Data is passed over to the CP server.

- Validation:
  On receiving data from a particular VCS cluster node, vxcpserv validates its credentials. If validation fails, then the connection request data is rejected.

About data protection

Membership arbitration by itself is inadequate for complete data protection because it assumes that all systems will either participate in the arbitration or are already down.

Rare situations can arise which must also be protected against. Some examples are:

- A system hang causes the kernel to stop processing for a period of time.
- The system resources were so busy that the heartbeat signal was not sent.
- A break and resume function is supported by the hardware and executed. Dropping the system to a system controller level with a break command can result in the heartbeat signal timeout.

In these types of situations, the systems are not actually down, and may return to the cluster after cluster membership has been recalculated. This could result in data corruption as a system could potentially write to disk before it determines it should no longer be in the cluster.

Combining membership arbitration with data protection of the shared storage eliminates all of the above possibilities for data corruption.

Data protection fences off (removes access to) the shared data storage from any system that is not a current and verified member of the cluster. Access is blocked by the use of SCSI-3 persistent reservations.

About SCSI-3 Persistent Reservation

SCSI-3 Persistent Reservation (SCSI-3 PR) supports device access from multiple systems, or from multiple paths from a single system. At the same time it blocks access to the device from other systems, or other paths.

VCS logic determines when to online a service group on a particular system. If the service group contains a disk group, the disk group is imported as part of the service group being brought online. When using SCSI-3 PR, importing the disk group puts registration and reservation on the data disks. Only the system that has imported the storage with SCSI-3 reservation can write to the shared storage. This prevents a system that did not participate in membership arbitration from corrupting the shared storage.

SCSI-3 PR ensures persistent reservations across SCSI bus resets.

Note: Use of SCSI 3 PR protects against all elements in the IT environment that might be trying to write illegally to storage, not only VCS related elements.

Membership arbitration combined with data protection is termed I/O fencing.

About I/O fencing configuration files

Table 8-1 lists the I/O fencing configuration files.

Table 8-1 I/O fencing configuration files

File	Description
/etc/sysconfig/vxfen	This file stores the start and stop environment variables for I/O fencing: ■ VXFEN_START—Defines the startup behavior for the I/O fencing module after a system reboot. Valid values include: 1—Indicates that I/O fencing is enabled to start up. 0—Indicates that I/O fencing is disabled to start up. ■ VXFEN_STOP—Defines the shutdown behavior for the I/O fencing module during a system shutdown. Valid values include: 1—Indicates that I/O fencing is enabled to shut down. 0—Indicates that I/O fencing is disabled to shut down. The installer sets the value of these variables to 1 at the end of VCS configuration. If you manually configured VCS, you must make sure to set the values of these environment variables to 1.
/etc/vxfendg	This file includes the coordinator disk group information. This file is not applicable for server-based fencing and majority-based fencing.

Table 8-1 I/O fencing configuration files *(continued)*

File	Description

/etc/vxfenmode	This file contains the following parameters:
	■ vxfen_mode
	  ■ scsi3—For disk-based fencing.
	  ■ customized—For server-based fencing.
	  ■ disabled—To run the I/O fencing driver but not do any fencing operations.
	  ■ majority— For fencing without the use of coordination points.
	■ vxfen_mechanism
	This parameter is applicable only for server-based fencing. Set the value as cps.
	■ scsi3_disk_policy
	  ■ dmp—Configure the vxfen module to use DMP devices
	  The disk policy is dmp by default. If you use iSCSI devices, you must set the disk policy as dmp.
	Note: You must use the same SCSI-3 disk policy on all the nodes.
	■ List of coordination points
	This list is required only for server-based fencing configuration.
	Coordination points in server-based fencing can include coordinator disks, CP servers, or both. If you use coordinator disks, you must create a coordinator disk group containing the individual coordinator disks.
	Refer to the sample file /etc/vxfen.d/vxfenmode_cps for more information on how to specify the coordination points and multiple IP addresses for each CP server.
	■ single_cp
	This parameter is applicable for server-based fencing which uses a single highly available CP server as its coordination point. Also applicable for when you use a coordinator disk group with single disk.
	■ autoseed_gab_timeout
	This parameter enables GAB automatic seeding of the cluster even when some cluster nodes are unavailable.
	This feature is applicable for I/O fencing in SCSI3 and customized mode.
	0—Turns the GAB auto-seed feature on. Any value greater than 0 indicates the number of seconds that GAB must delay before it automatically seeds the cluster.
	-1—Turns the GAB auto-seed feature off. This setting is the default.
	■ detect_false_pesb
	0—Disables stale key detection.
	1—Enables stale key detection to determine whether a preexisting split brain is a true condition or a false alarm.
	Default: 0
	Note: This parameter is considered only when `vxfen_mode=customized`.

Table 8-1 I/O fencing configuration files *(continued)*

File	Description

/etc/vxfentab	When I/O fencing starts, the vxfen startup script creates this /etc/vxfentab file on each node. The startup script uses the contents of the /etc/vxfendg and /etc/vxfenmode files. Any time a system is rebooted, the fencing driver reinitializes the vxfentab file with the current list of all the coordinator points.
	Note: The /etc/vxfentab file is a generated file; do not modify this file.
	For disk-based I/O fencing, the /etc/vxfentab file on each node contains a list of all paths to each coordinator disk along with its unique disk identifier. A space separates the path and the unique disk identifier. An example of the /etc/vxfentab file in a disk-based fencing configuration on one node resembles as follows:
	■ DMP disk:
	`/dev/vx/rdmp/sdx3 HITACHI%5F1724-100%20%20FAStT%5FDISKS%5F6 00A0B8000215A5D000006804E795D0A3`
	`/dev/vx/rdmp/sdy3 HITACHI%5F1724-100%20%20FAStT%5FDISKS%5F6 00A0B8000215A5D000006814E795D0B3`
	`/dev/vx/rdmp/sdz3 HITACHI%5F1724-100%20%20FAStT%5FDISKS%5F6 00A0B8000215A5D000006824E795D0C3`
	For server-based fencing, the /etc/vxfentab file also includes the security settings information.
	For server-based fencing with single CP server, the /etc/vxfentab file also includes the single_cp settings information.
	This file is not applicable for majority-based fencing.

Examples of VCS operation with I/O fencing

This topic describes the general logic employed by the I/O fencing module along with some specific example scenarios. Note that this topic does not apply to majority-based I/O fencing.

See “About the I/O fencing algorithm” on page 268.

See “Example: Two-system cluster where one system fails” on page 269.

See “Example: Four-system cluster where cluster interconnect fails” on page 270.

About the I/O fencing algorithm

To ensure the most appropriate behavior is followed in both common and rare corner case events, the fencing algorithm works as follows:

- The fencing module is designed to never have systems in more than one subcluster remain current and valid members of the cluster. In all cases, either one subcluster will survive, or in very rare cases, no systems will.
- The system with the lowest LLT ID in any subcluster of the original cluster races for control of the coordinator disks on behalf of the other systems in that subcluster.
- If a system wins the race for the first coordinator disk, that system is given priority to win the race for the other coordinator disks.

  Any system that loses a race will delay a short period of time before racing for the next disk. Under normal circumstances, the winner of the race to the first coordinator disk will win all disks.

  This ensures a clear winner when multiple systems race for the coordinator disk, preventing the case where three or more systems each win the race for one coordinator disk.
- If the cluster splits such that one of the subclusters has at least 51% of the members of the previous stable membership, that subcluster is given priority to win the race.

  The system in the smaller subcluster(s) delay a short period before beginning the race.

  If you use the preferred fencing feature, then the subcluster with the lesser total weight will delay.

  See "About preferred fencing" on page 248.

  This ensures that as many systems as possible will remain running in the cluster.
- If the vxfen module discovers on startup that the system that has control of the coordinator disks is not in the current GAB membership, an error message indicating a possible split brain condition is printed to the console.

  The administrator must clear this condition manually with the `vxfenclearpre` utility.

Example: Two-system cluster where one system fails

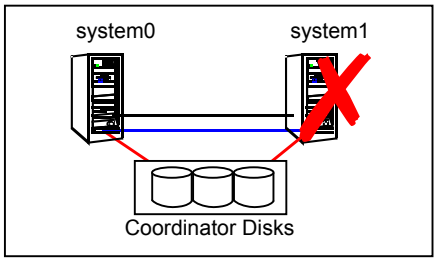
In this example, System1 fails, and System0 carries out the I/O fencing operation as follows:

- The GAB module on System0 determines System1 has failed due to loss of heartbeat signal reported from LLT.
- GAB passes the membership change to the fencing module on each system in the cluster.

  The only system that is still running is System0

- System0 gains control of the coordinator disks by ejecting the key registered by System1 from each coordinator disk.
  The ejection takes place one by one, in the order of the coordinator disk's serial number.
- When the fencing module on System0 successfully controls the coordinator disks, HAD carries out any associated policy connected with the membership change.
- System1 is blocked access to the shared storage, if this shared storage was configured in a service group that was now taken over by System0 and imported.

Figure 8-13 I/O Fencing example with system failure

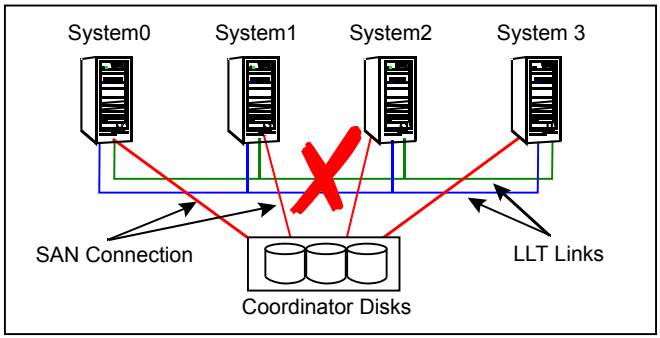


Example: Four-system cluster where cluster interconnect fails

In this example, the cluster interconnect fails in such a way as to split the cluster from one four-system cluster to two-system clusters. The cluster performs membership arbitration to ensure that only one subcluster remains.

Due to loss of heartbeat, System0 and System1 both believe System2 and System3 are down. System2 and System3 both believe System0 and System1 are down.

Figure 8-14 Four-system cluster where cluster interconnect fails



The progression of I/O fencing operations are as follows:

- LLT on each of the four systems no longer receives heartbeat messages from the systems on the other side of the interconnect failure on any of the configured LLT interfaces for the peer inactive timeout configured time.
- LLT on each system passes to GAB that it has noticed a membership change. Specifically:
  - LLT on System0 passes to GAB that it no longer sees System2 and System3
  - LLT on System1 passes to GAB that it no longer sees System2 and System3
  - LLT on System2 passes to GAB that it no longer sees System0 and System1
  - LLT on System3 passes to GAB that it no longer sees System0 and System1
- After LLT informs GAB of a heartbeat loss, the systems that are remaining do a "GAB Stable Timeout" (5 seconds). In this example:
  - System0 and System1 agree that both of them do not see System2 and System3
  - System2 and System3 agree that both of them do not see System0 and System1
- GAB marks the system as DOWN, and excludes the system from the cluster membership. In this example:
  - GAB on System0 and System1 mark System2 and System3 as DOWN and excludes them from cluster membership.
  - GAB on System2 and System3 mark System0 and System1 as DOWN and excludes them from cluster membership.
- GAB on each of the four systems passes the membership change to the vxfen driver for membership arbitration. Each subcluster races for control of the coordinator disks. In this example:
  - System0 has the lower LLT ID, and races on behalf of itself and System1.
  - System2 has the lower LLT ID, and races on behalf of itself and System3.
- GAB on each of the four systems also passes the membership change to HAD. HAD waits for the result of the membership arbitration from the fencing module before taking any further action.
- If System0 is not able to reach a majority of the coordination points, then the VxFEN driver will initiate a racer re-election from System0 to System1 and System1 will initiate the race for the coordination points.
- Assume System0 wins the race for the coordinator disks, and ejects the registration keys of System2 and System3 off the disks. The result is as follows:

  - System0 wins the race for the coordinator disk. The fencing module on System0 sends a WON_RACE to all other fencing modules in the current cluster, in this case System0 and System1. On receiving a WON_RACE, the fencing module on each system in turn communicates success to HAD. System0 and System1 remain valid and current members of the cluster.
  - If System0 dies before it sends a WON_RACE to System1, then VxFEN will initiate a racer re-election from System0 to System1 and System1 will initiate the race for the coordination points.
    System1 on winning a majority of the coordination points remains valid and current member of the cluster and the fencing module on System1 in turn communicates success to HAD.
  - System2 loses the race for control of the coordinator disks and the fencing module on System 2 sends a LOST_RACE message. The fencing module on System2 calls a kernel panic and the system restarts.
  - System3 sees another membership change from the kernel panic of System2. Because that was the system that was racing for control of the coordinator disks in this subcluster, System3 also panics.
- HAD carries out any associated policy or recovery actions based on the membership change.
- System2 and System3 are blocked access to the shared storage (if the shared storage was part of a service group that is now taken over by System0 or System 1).
- To rejoin System2 and System3 to the cluster, the administrator must do the following:
  - Shut down System2 and System3
  - Fix the cluster interconnect links
  - Restart System2 and System3

How I/O fencing works in different event scenarios

Table 8-2 describes how I/O fencing works to prevent data corruption in different failure event scenarios. For each event, review the corrective operator actions.

Table 8-2 I/O fencing scenarios

Event	Node A: What happens?	Node B: What happens?	Operator action
Both private networks fail.	Node A races for majority of coordination points. If Node A wins race for coordination points, Node A ejects Node B from the shared disks and continues.	Node B races for majority of coordination points. If Node B loses the race for the coordination points, Node B panics and removes itself from the cluster.	When Node B is ejected from cluster, repair the private networks before attempting to bring Node B back.
Both private networks function again after event above.	Node A continues to work.	Node B has crashed. It cannot start the database since it is unable to write to the data disks.	Restart Node B after private networks are restored.
One private network fails.	Node A prints message about an IOFENCE on the console but continues.	Node B prints message about an IOFENCE on the console but continues.	Repair private network. After network is repaired, both nodes automatically use it.
Node A hangs.	Node A is extremely busy for some reason or is in the kernel debugger. When Node A is no longer hung or in the kernel debugger, any queued writes to the data disks fail because Node A is ejected. When Node A receives message from GAB about being ejected, it panics and removes itself from the cluster.	Node B loses heartbeats with Node A, and races for a majority of coordination points. Node B wins race for coordination points and ejects Node A from shared data disks.	Repair or debug the node that hangs and reboot the node to rejoin the cluster.

Table 8-2 I/O fencing scenarios *(continued)*

Event	Node A: What happens?	Node B: What happens?	Operator action
Nodes A and B and private networks lose power. Coordination points and data disks retain power. Power returns to nodes and they restart, but private networks still have no power.	Node A restarts and I/O fencing driver (vxfen) detects Node B is registered with coordination points. The driver does not see Node B listed as member of cluster because private networks are down. This causes the I/O fencing device driver to prevent Node A from joining the cluster. Node A console displays: `Potentially a preexisting split brain. Dropping out of the cluster. Refer to the user documentation for steps required to clear preexisting split brain.`	Node B restarts and I/O fencing driver (vxfen) detects Node A is registered with coordination points. The driver does not see Node A listed as member of cluster because private networks are down. This causes the I/O fencing device driver to prevent Node B from joining the cluster. Node B console displays: `Potentially a preexisting split brain. Dropping out of the cluster. Refer to the user documentation for steps required to clear preexisting split brain.`	Resolve preexisting split-brain condition. See "Fencing startup reports preexisting split-brain" on page 655.

Table 8-2 I/O fencing scenarios *(continued)*

Event	Node A: What happens?	Node B: What happens?	Operator action
Node A crashes while Node B is down. Node B comes up and Node A is still down.	Node A is crashed.	Node B restarts and detects Node A is registered with the coordination points. The driver does not see Node A listed as member of the cluster. The I/O fencing device driver prints message on console: `Potentially a preexisting split brain. Dropping out of the cluster. Refer to the user documentation for steps required to clear preexisting split brain.`	Resolve preexisting split-brain condition. See "Fencing startup reports preexisting split-brain" on page 655.
The disk array containing two of the three coordination points is powered off. No node leaves the cluster membership	Node A continues to operate as long as no nodes leave the cluster.	Node B continues to operate as long as no nodes leave the cluster.	Power on the failed disk array so that subsequent network partition does not cause cluster shutdown, or replace coordination points. See "Replacing I/O fencing coordinator disks when the cluster is online" on page 304.

Table 8-2 I/O fencing scenarios *(continued)*

Event	Node A: What happens?	Node B: What happens?	Operator action
The disk array containing two of the three coordination points is powered off. Node B gracefully leaves the cluster and the disk array is still powered off. Leaving gracefully implies a clean shutdown so that vxfen is properly unconfigured.	Node A continues to operate in the cluster.	Node B has left the cluster.	Power on the failed disk array so that subsequent network partition does not cause cluster shutdown, or replace coordination points. See "Replacing I/O fencing coordinator disks when the cluster is online" on page 304.
The disk array containing two of the three coordination points is powered off. Node B abruptly crashes or a network partition occurs between node A and node B, and the disk array is still powered off.	Node A races for a majority of coordination points. Node A fails because only one of the three coordination points is available. Node A panics and removes itself from the cluster.	Node B has left cluster due to crash or network partition.	Power on the failed disk array and restart I/O fencing driver to enable Node A to register with all coordination points, or replace coordination points. See "Replacing defective disks when the cluster is offline" on page 658.

About cluster membership and data protection without I/O fencing

Proper seeding of the cluster and the use of low priority heartbeat cluster interconnect links are best practices with or without the use of I/O fencing. Best practice also recommends multiple cluster interconnect links between systems in the cluster. This allows GAB to differentiate between:

- A loss of all heartbeat links simultaneously, which is interpreted as a system failure. In this case, depending on failover configuration, HAD may attempt to restart the services that were running on that system on another system.
- A loss of all heartbeat links over time, which is interpreted as an interconnect failure. In this case, the assumption is made that there is a high probability that

the system is not down, and HAD does not attempt to restart the services on another system.

In order for this differentiation to have meaning, it is important to ensure the cluster interconnect links do not have a single point of failure, such as a network switch or Ethernet card.

About jeopardy

In all cases, when LLT on a system no longer receives heartbeat messages from another system on any of the configured LLT interfaces, GAB reports a change in membership.

When a system has only one interconnect link remaining to the cluster, GAB can no longer reliably discriminate between loss of a system and loss of the network. The reliability of the system's membership is considered at risk. A special membership category takes effect in this situation, called a jeopardy membership. This provides the best possible split-brain protection without membership arbitration and SCSI-3 capable devices.

When a system is placed in jeopardy membership status, two actions occur if the system loses the last interconnect link:

- VCS places service groups running on the system in autodisabled state. A service group in autodisabled state may failover on a resource or group fault, but cannot fail over on a system fault until the autodisabled flag is manually cleared by the administrator.
- VCS operates the system as a single system cluster. Other systems in the cluster are partitioned off in a separate cluster membership.

About Daemon Down Node Alive (DDNA)

Daemon Down Node Alive (DDNA) is a condition in which the VCS high availability daemon (HAD) on a node fails, but the node is running. When HAD fails, the hashadow process tries to bring HAD up again. If the hashadow process succeeds in bringing HAD up, the system leaves the DDNA membership and joins the regular membership.

In a DDNA condition, VCS does not have information about the state of service groups on the node. So, VCS places all service groups that were online on the affected node in the autodisabled state. The service groups that were online on the node cannot fail over.

Manual intervention is required to enable failover of autodisabled service groups. The administrator must release the resources running on the affected node, clear resource faults, and bring the service groups online on another node.

You can use the GAB registration monitoring feature to detect DDNA conditions.

See "About GAB client registration monitoring" on page 584.

Examples of VCS operation without I/O fencing

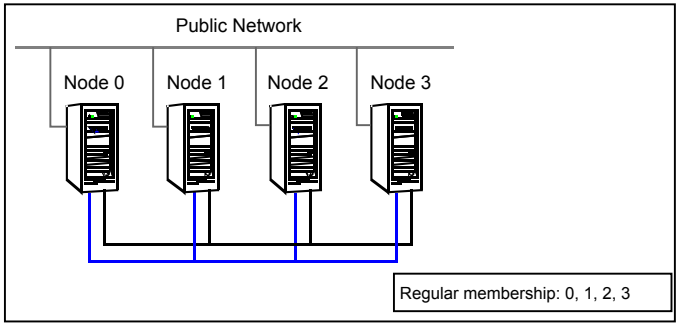
The following scenarios describe events, and how VCS responds, in a cluster without I/O fencing.

See "Example: Four-system cluster without a low priority link" on page 278.

See "Example: Four-system cluster with low priority link" on page 280.

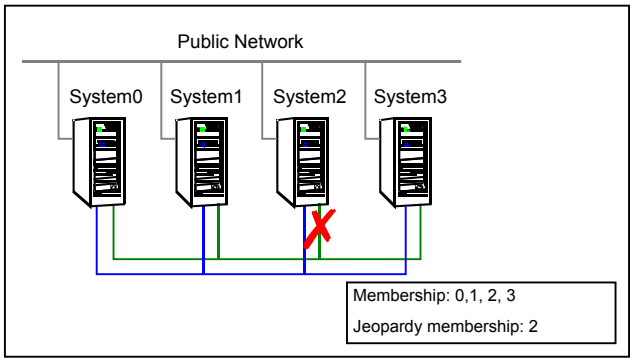
Example: Four-system cluster without a low priority link

Consider a four-system cluster that has two private cluster interconnect heartbeat links. The cluster does not have any low priority link.



Cluster interconnect link failure

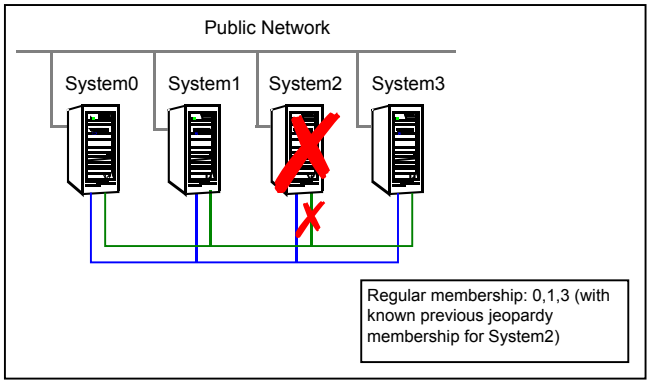
In this example, a link to System2 fails, leaving System2 with only one cluster interconnect link remaining.



The cluster is reconfigured. Systems 0, 1, and 3 are in the regular membership and System2 in a jeopardy membership. Service groups on System2 are autodisabled. All normal cluster operations continue, including normal failover of service groups due to resource fault.

Cluster interconnect link failure followed by system failure

In this example, the link to System2 fails, and System2 is put in the jeopardy membership. Subsequently, System2 fails due to a power fault.

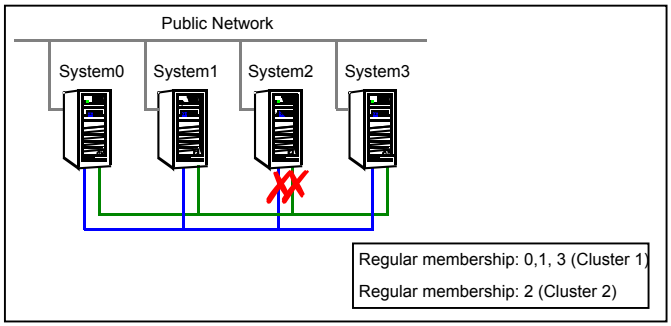


Systems 0, 1, and 3 recognize that System2 has faulted. The cluster is reformed. Systems 0, 1, and 3 are in a regular membership. When System2 went into jeopardy membership, service groups running on System2 were autodisabled. Even though the system is now completely failed, no other system can assume ownership of these service groups unless the system administrator manually clears the AutoDisabled flag on the service groups that were running on System2.

However, after the flag is cleared, these service groups can be manually brought online on other systems in the cluster.

All high priority cluster interconnect links fail

In this example, all high priority links to System2 fail. This can occur two ways:



- Both links to System2 fail at the same time

  System2 was never in jeopardy membership. Without a low priority link, the cluster splits into two subclusters, where System0, 1 and 3 are in one subcluster, and System2 is in another. This is a split brain scenario.

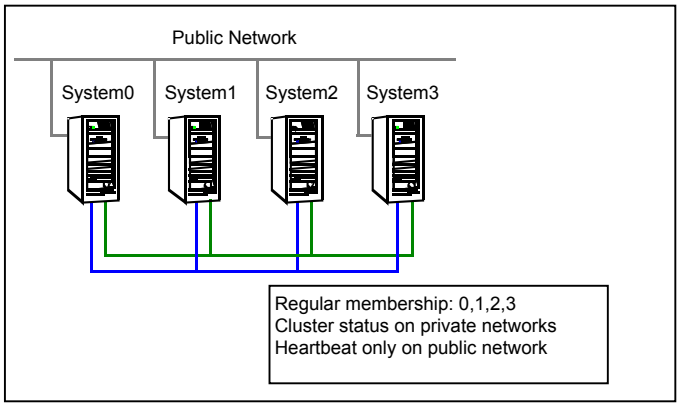
- Both links to System2 fail at different times

  System2 was in a jeopardy membership when the second link failed, and therefore the service groups that were online on System2 were autodisabled. No other system can online these service groups without administrator intervention.

  Systems 0, 1 and 3 form a mini-cluster. System2 forms another single- system mini-cluster. All service groups that were present on systems 0, 1 and 3 are autodisabled on System2.

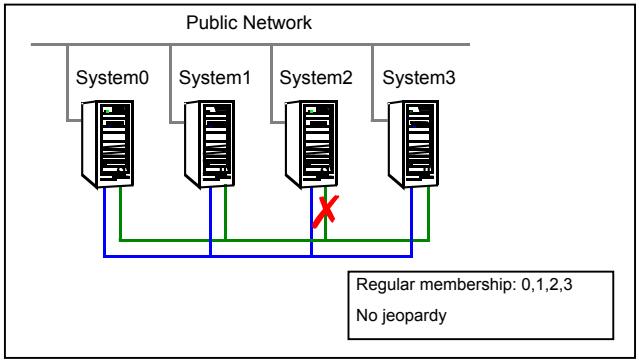
Example: Four-system cluster with low priority link

Consider a four-system cluster that has two private cluster interconnect heartbeat links, and one public low priority link.



Cluster interconnect link failure

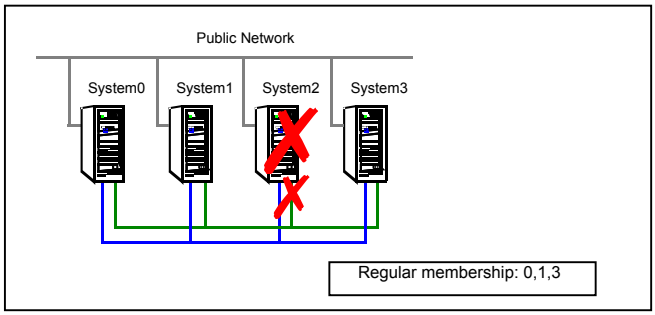
In this example, a link to System2 fails, leaving System2 with one cluster interconnect link and the low priority link remaining.



Other systems send all cluster status traffic to System2 over the remaining private link and use both private links for traffic between themselves. The low priority link continues carrying the heartbeat signal only. No jeopardy condition is in effect because two links remain to determine system failure.

Cluster interconnect link failure followed by system failure

In this example, the link to System2 fails. Because there is a low priority heartbeat link, System2 is not put in the jeopardy membership. Subsequently, System2 fails due to a power fault.

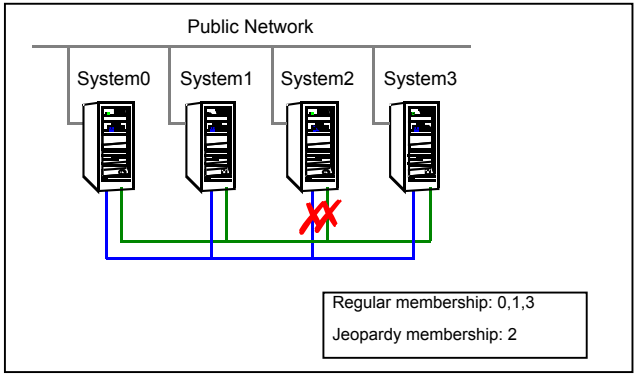


Systems 0, 1, and 3 recognize that System2 has faulted. The cluster is reformed. Systems 0, 1, and 3 are in a regular membership. The service groups on System2 that are configured for failover on system fault are attempted to be brought online on another target system, if one exists.

All high priority cluster interconnect links fail

In this example, both high priority cluster interconnect links to System2 fail, leaving System2 with only the low priority link remaining.

Cluster status communication is now routed over the low priority link to System2. System2 is placed in a jeopardy membership. The service groups on System2 are autodisabled, and the service group attribute AutoFailOver is set to 0, meaning the service group will not fail over on a system fault.



When a cluster interconnect link is re-established, all cluster status communications revert back to the cluster interconnect and the low priority link returns to sending heartbeat signal only. At this point, System2 is placed back in regular cluster membership.

Summary of best practices for cluster communications

Veritas Technologies recommends the following best practices for cluster communications to best support proper cluster membership and data protection:

- Properly seed the cluster by requiring all systems, and not just a subset of systems, to be present in the GAB membership before the cluster will automatically seed.
  If every system is not present, manual intervention by the administrator must eliminate the possibility of a split brain condition before manually seeding the cluster.
- Configure multiple independent communication network links between cluster systems.
  Networks should not have a single point of failure, such as a shared switch or Ethernet card.
- Low-priority LLT links in clusters with or without I/O fencing is recommended. In clusters without I/O fencing, this is critical.

  Note: An exception to this is if the cluster uses fencing along with Cluster File Systems (CFS) or Oracle Real Application Clusters (RAC).

  The reason for this is that low priority links are usually shared public network links. In the case where the main cluster interconnects fail, and the low priority link was the only remaining link, large amounts of data would be moved to the low priority link. This would potentially slow down the public network to unacceptable performance. Without a low priority link configured, membership arbitration would go into effect in this case, and some systems may be taken down, but the remaining systems would continue to run without impact to the public network.
  It is not recommended to have a cluster with CFS or RAC without I/O fencing configured.
- Disable the console-abort sequence
  Most UNIX systems provide a console-abort sequence that enables the administrator to halt and continue the processor. Continuing operations after the processor has stopped may corrupt data and is therefore unsupported by VCS.
  When a system is halted with the abort sequence, it stops producing heartbeats. The other systems in the cluster consider the system failed and take over its services. If the system is later enabled with another console sequence, it

continues writing to shared storage as before, even though its applications have been restarted on other systems.

Veritas Technologies recommends disabling the console-abort sequence or creating an alias to force the `go` command to perform a restart on systems not running I/O fencing.

- Veritas Technologies recommends at least three coordination points to configure I/O fencing. You can use coordinator disks, CP servers, or a combination of both.

  Select the smallest possible LUNs for use as coordinator disks. No more than three coordinator disks are needed in any configuration.

- Do not reconnect the cluster interconnect after a network partition without shutting down one side of the split cluster.

  A common example of this happens during testing, where the administrator may disconnect the cluster interconnect and create a network partition. Depending on when the interconnect cables are reconnected, unexpected behavior can occur.

Chapter 9

Administering I/O fencing

This chapter includes the following topics:

- About administering I/O fencing
- About the vxfentsthdw utility
- About the vxfenadm utility
- About the vxfenclearpre utility
- About the vxfenswap utility
- About administering the coordination point server
- About migrating between disk-based and server-based fencing configurations
- Enabling or disabling the preferred fencing policy
- About I/O fencing log files

About administering I/O fencing

The I/O fencing feature provides the following utilities that are available through the `VRTSvxfen` RPM:

vxfentsthdw	Tests SCSI-3 functionality of the disks for I/O fencing See "About the vxfentsthdw utility" on page 286.
vxfenconfig	Configures and unconfigures I/O fencing Lists the coordination points used by the vxfen driver.

vxfenadm	Displays information on I/O fencing operations and manages SCSI-3 disk registrations and reservations for I/O fencing See "About the vxfenadm utility" on page 295.
vxfenclearpre	Removes SCSI-3 registrations and reservations from disks See "About the vxfenclearpre utility" on page 300.
vxfenswap	Replaces coordination points without stopping I/O fencing See "About the vxfenswap utility" on page 303.
vxfendisk	Generates the list of paths of disks in the disk group. This utility requires that Veritas Volume Manager (VxVM) is installed and configured.

The I/O fencing commands reside in the /opt/VRTS/bin folder. Make sure you added this folder path to the PATH environment variable.

Refer to the corresponding manual page for more information on the commands.

About the vxfentsthdw utility

You can use the `vxfentsthdw` utility to verify that shared storage arrays to be used for data support SCSI-3 persistent reservations and I/O fencing. During the I/O fencing configuration, the testing utility is used to test a single disk. The utility has other options that may be more suitable for testing storage devices in other configurations. You also need to test coordinator disk groups.

See the *Cluster Server Installation Guide* to set up I/O fencing.

The utility, which you can run from one system in the cluster, tests the storage used for data by setting and verifying SCSI-3 registrations on the disk or disks you specify, setting and verifying persistent reservations on the disks, writing data to the disks and reading it, and removing the registrations from the disks.

Refer also to the `vxfentsthdw`(1M) manual page.

General guidelines for using the vxfentsthdw utility

Review the following guidelines to use the `vxfentsthdw` utility:

- The utility requires two systems connected to the shared storage.

Caution: The tests overwrite and destroy data on the disks, unless you use the `-r` option.

- The two nodes must have SSH (default) or rsh communication. If you use rsh, launch the `vxfentsthdw` utility with the `-n` option.

  After completing the testing process, you can remove permissions for communication and restore public network connections.
- To ensure both systems are connected to the same disk during the testing, you can use the `vxfenadm -i diskpath` command to verify a disk's serial number.

  See "Verifying that the nodes see the same disk" on page 299.
- For disk arrays with many disks, use the `-m` option to sample a few disks before creating a disk group and using the `-g` option to test them all.
- The utility indicates a disk can be used for I/O fencing with a message resembling:

  ```
  The disk /dev/sdx is ready to be configured for
  I/O Fencing on node sys1
  ```

  If the utility does not show a message stating a disk is ready, verification has failed.
- The `-o` option overrides disk size-related errors and the utility proceeds with other tests, however, the disk may not setup correctly as the size may be smaller than the supported size. The supported disk size for data disks is 256 MB and for coordinator disks is 128 MB.
- If the disk you intend to test has existing SCSI-3 registration keys, the test issues a warning before proceeding.

About the vxfentsthdw command options

Table 9-1 describes the methods that the utility provides to test storage devices.

Table 9-1 vxfentsthdw options

vxfentsthdw option	Description	When to use
-n	Utility uses rsh for communication.	Use when rsh is used for communication.
-r	Non-destructive testing. Testing of the disks for SCSI-3 persistent reservations occurs in a non-destructive way; that is, there is only testing for reads, not writes. Can be used with `-m`, `-f`, or `-g` options.	Use during non-destructive testing. See "Performing non-destructive testing on the disks using the -r option" on page 291.

Table 9-1 vxfentsthdw options *(continued)*

vxfentsthdw option	Description	When to use
-t	Testing of the return value of SCSI TEST UNIT (TUR) command under SCSI-3 reservations. A warning is printed on failure of TUR testing.	When you want to perform TUR testing.
-d	Use Dynamic Multi-Pathing (DMP) devices. Can be used with `-c` or `-g` options.	By default, the `vxfentsthdw` script picks up the DMP paths for disks in the disk group. If you want the script to use the raw paths for disks in the disk group, use the `-w` option.
-w	Use raw devices. Can be used with `-c` or `-g` options.	With the `-w` option, the `vxfentsthdw` script picks the operating system paths for disks in the disk group. By default, the script uses the `-d` option to pick up the DMP paths for disks in the disk group.
-c	Utility tests the coordinator disk group prompting for systems and devices, and reporting success or failure.	For testing disks in coordinator disk group. See "Testing the coordinator disk group using the -c option of vxfentsthdw" on page 289.
-m	Utility runs manually, in interactive mode, prompting for systems and devices, and reporting success or failure. Can be used with `-r` and `-t` options. `-m` is the default option.	For testing a few disks or for sampling disks in larger arrays. See "Testing the shared disks using the vxfentsthdw -m option" on page 291.
-f *filename*	Utility tests system and device combinations listed in a text file. Can be used with `-r` and `-t` options.	For testing several disks. See "Testing the shared disks listed in a file using the vxfentsthdw -f option" on page 293.

Table 9-1 vxfentsthdw options *(continued)*

vxfentsthdw option	Description	When to use
-g *disk_group*	Utility tests all disk devices in a specified disk group. Can be used with `-r` and `-t` options.	For testing many disks and arrays of disks. Disk groups may be temporarily created for testing purposes and destroyed (ungrouped) after testing. See "Testing all the disks in a disk group using the vxfentsthdw -g option" on page 293.
-o	Utility overrrides disk size-related errors.	For testing SCSI-3 Reservation compliance of disks, but, overrides disk size-related errors. See "Testing disks with the vxfentsthdw `-o` option" on page 294.

Testing the coordinator disk group using the -c option of vxfentsthdw

Use the `vxfentsthdw` utility to verify disks are configured to support I/O fencing. In this procedure, the `vxfentsthdw` utility tests the three disks, one disk at a time from each node.

The procedure in this section uses the following disks for example:

- From the node sys1, the disks are seen as `/dev/sdg`, `/dev/sdh`, and `/dev/sdi`.
- From the node sys2, the same disks are seen as `/dev/sdx`, `/dev/sdy`, and `/dev/sdz`.

Note: To test the coordinator disk group using the `vxfentsthdw` utility, the utility requires that the coordinator disk group, vxfencoorddg, be accessible from two nodes.

To test the coordinator disk group using vxfentsthdw -c

1. Use the `vxfentsthdw` command with the `-c` option. For example:

```
# vxfentsthdw -c vxfencoorddg
```

2. Enter the nodes you are using to test the coordinator disks:

```
Enter the first node of the cluster: sys1
Enter the second node of the cluster: sys2
```

3. Review the output of the testing process for both nodes for all disks in the coordinator disk group. Each disk should display output that resembles:

```
ALL tests on the disk /dev/sdg have PASSED.
The disk is now ready to be configured for I/O Fencing on node
sys1 as a COORDINATOR DISK.

ALL tests on the disk /dev/sdx have PASSED.
The disk is now ready to be configured for I/O Fencing on node
sys2 as a COORDINATOR DISK.
```

4. After you test all disks in the disk group, the vxfencoorddg disk group is ready for use.

Removing and replacing a failed disk

If a disk in the coordinator disk group fails verification, remove the failed disk or LUN from the vxfencoorddg disk group, replace it with another, and retest the disk group.

To remove and replace a failed disk

1. Use the vxdiskadm utility to remove the failed disk from the disk group.

   Refer to the *Storage Foundation Administrator's Guide*.

2. Add a new disk to the node, initialize it, and add it to the coordinator disk group.

   See the *Cluster Server Installation Guide* for instructions to initialize disks for I/O fencing and to set up coordinator disk groups.

   If necessary, start the disk group.

   See the *Storage Foundation Administrator's Guide* for instructions to start the disk group.

3. Retest the disk group.

   See "Testing the coordinator disk group using the -c option of vxfentsthdw" on page 289.

Performing non-destructive testing on the disks using the -r option

You can perform non-destructive testing on the disk devices when you want to preserve the data.

To perform non-destructive testing on disks

- To test disk devices containing data you want to preserve, you can use the `-r` option with the `-m`, `-f`, or `-g` options.

  For example, to use the -m option and the `-r` option, you can run the utility as follows:

  ```
  # vxfentsthdw -rm
  ```

  When invoked with the `-r` option, the utility does not use tests that write to the disks. Therefore, it does not test the disks for all of the usual conditions of use.

Testing the shared disks using the vxfentsthdw -m option

Review the procedure to test the shared disks. By default, the utility uses the `-m` option.

This procedure uses the /dev/sdx disk in the steps.

If the utility does not show a message stating a disk is ready, verification has failed. Failure of verification can be the result of an improperly configured disk array. It can also be caused by a bad disk.

If the failure is due to a bad disk, remove and replace it. The `vxfentsthdw` utility indicates a disk can be used for I/O fencing with a message resembling:

```
The disk /dev/sdx is ready to be configured for
I/O Fencing on node sys1
```

Note: For A/P arrays, run the `vxfentsthdw` command only on active enabled paths.

To test disks using the vxfentsthdw script

1. Make sure system-to-system communication is functioning properly.
2. From one node, start the utility.

   ```
   # vxfentsthdw [-n]
   ```

3. After reviewing the overview and warning that the tests overwrite data on the disks, confirm to continue the process and enter the node names.

   ```
   ******** WARNING!!!!!!!! ********
   THIS UTILITY WILL DESTROY THE DATA ON THE DISK!!

   Do you still want to continue : [y/n] (default: n) y
   Enter the first node of the cluster: sys1
   Enter the second node of the cluster: sys2
   ```

4. Enter the names of the disks you are checking. For each node, the disk may be known by the same name:

   ```
   Enter the disk name to be checked for SCSI-3 PGR on node
   sys1 in the format:
           for dmp: /dev/vx/rdmp/<disk_name>
           for raw: /dev/<disk_name>
   Make sure it's the same disk as seen by nodes sys1 and sys2
   /dev/sdr

   Enter the disk name to be checked for SCSI-3 PGR on node
   sys2 in the format:
           for dmp: /dev/vx/rdmp/<disk_name>
           for raw: /dev/<disk_name>
   Make sure it's the same disk as seen by nodes sys1 and sys2
   /dev/sdr
   ```

   If the serial numbers of the disks are not identical, then the test terminates.

5 Review the output as the utility performs the checks and report its activities.

6 If a disk is ready for I/O fencing on each node, the utility reports success:

```
ALL tests on the disk /dev/sdx have PASSED
The disk is now ready to be configured for I/O Fencing on node
sys1
...
Removing test keys and temporary files, if any ...
.
.
```

7 Run the vxfentsthdw utility for each disk you intend to verify.

Testing the shared disks listed in a file using the vxfentsthdw -f option

Use the -f option to test disks that are listed in a text file. Review the following example procedure.

To test the shared disks listed in a file

1 Create a text file `disks_test` to test two disks shared by systems `sys1` and `sys2` that might resemble:

```
sys1 /dev/sdz sys2 /dev/sdy
sys1 /dev/sdu sys2 /dev/sdw
```

where the first disk is listed in the first line and is seen by `sys1` as `/dev/sdz` and by `sys2` as `/dev/sdy`. The other disk, in the second line, is seen as `/dev/sdu` from `sys1` and `/dev/sdw` from `sys2`. Typically, the list of disks could be extensive.

2 To test the disks, enter the following command:

```
# vxfentsthdw -f disks_test
```

The utility reports the test results one disk at a time, just as for the `-m` option.

Testing all the disks in a disk group using the vxfentsthdw -g option

Use the -g option to test all disks within a disk group. For example, you create a temporary disk group consisting of all disks in a disk array and test the group.

Note: Do not import the test disk group as shared; that is, do not use the `-s` option with the `vxdg import` command.

After testing, destroy the disk group and put the disks into disk groups as you need.

To test all the disks in a disk group

1. Create a disk group for the disks that you want to test.
2. Enter the following command to test the disk group test_disks_dg:

   ```
   # vxfentsthdw -g test_disks_dg
   ```

   The utility reports the test results one disk at a time.

Testing a disk with existing keys

If the utility detects that a coordinator disk has existing keys, you see a message that resembles:

```
There are Veritas I/O fencing keys on the disk. Please make sure
that I/O fencing is shut down on all nodes of the cluster before
continuing.

         ******** WARNING!!!!!!!! ********

THIS SCRIPT CAN ONLY BE USED IF THERE ARE NO OTHER ACTIVE NODES
IN THE CLUSTER!  VERIFY ALL OTHER NODES ARE POWERED OFF OR
INCAPABLE OF ACCESSING SHARED STORAGE.

If this is not the case, data corruption will result.

Do you still want to continue : [y/n] (default: n) y
```

The utility prompts you with a warning before proceeding. You may continue as long as I/O fencing is not yet configured.

Testing disks with the vxfentsthdw -o option

Use the -o option to check for SCSI-3 reservation or registration compliance but not to test the disk size requirements.

Note that you can override a disk size-related error by using the -o flag. The -o flag lets the utility continue with tests related to SCSI-3 reservations, registrations, and other tests. However, the disks may not set up correctly as the disk size may be below the supported size for coordinator or data disks.

You can use the -o option with all the available vxfentsthdw options when you encounter size-related errors.

About the vxfenadm utility

Administrators can use the `vxfenadm` command to troubleshoot and test fencing configurations.

The command's options for use by administrators are as follows:

-s	read the keys on a disk and display the keys in numeric, character, and node format **Note:** The `-g` and `-G` options are deprecated. Use the `-s` option.
-i	read SCSI inquiry information from device
-m	register with disks
-n	make a reservation with disks
-p	remove registrations made by other systems
-r	read reservations
-x	remove registrations

Refer to the `vxfenadm`(1M) manual page for a complete list of the command options.

About the I/O fencing registration key format

The keys that the vxfen driver registers on the data disks and the coordinator disks consist of eight bytes. The key format is different for the coordinator disks and data disks.

The key format of the coordinator disks is as follows:

Byte	0	1	2	3	4	5	6	7
Value	V	F	cID 0x	cID 0x	cID 0x	cID 0x	nID 0x	nID 0x

where:

- VF is the unique identifier that carves out a namespace for the keys (consumes two bytes)
- cID 0x is the LLT cluster ID in hexadecimal (consumes four bytes)
- nID 0x is the LLT node ID in hexadecimal (consumes two bytes)

The vxfen driver uses this key format in both sybase mode of I/O fencing.

The key format of the data disks that are configured as failover disk groups under VCS is as follows:

Byte	0	1	2	3	4	5	6	7
Value	A+nID	V	C	S				

where nID is the LLT node ID

For example: If the node ID is 1, then the first byte has the value as B (‘A’ + 1 = B).

The key format of the data disks configured as parallel disk groups under Cluster Volume Manager (CVM) is as follows:

Byte	0	1	2	3	4	5	6	7
Value	A+nID	P	G	R	DGcount	DGcount	DGcount	DGcount

where DGcount is the count of disk groups in the configuration (consumes four bytes).

By default, CVM uses a unique fencing key for each disk group. However, some arrays have a restriction on the total number of unique keys that can be registered. In such cases, you can use the `same_key_for_alldgs` tunable parameter to change the default behavior. The default value of the parameter is `off`. If your configuration hits the storage array limit on total number of unique keys, you can change the value to `on` using the `vxdefault` command as follows:

```
# vxdefault set same_key_for_alldgs on
# vxdefault list
KEYWORD              CURRENT-VALUE   DEFAULT-VALUE
...
same_key_for_alldgs  on              off
...
```

If the tunable is changed to `on`, all subsequent keys that the CVM generates on disk group imports or creates have '0000' as their last four bytes (DGcount is 0). You must deport and re-import all the disk groups that are already imported for the changed value of the `same_key_for_alldgs` tunable to take effect.

Displaying the I/O fencing registration keys

You can display the keys that are currently assigned to the disks using the `vxfenadm` command.

The variables such as *disk_7*, *disk_8*, and *disk_9* in the following procedure represent the disk names in your setup.

To display the I/O fencing registration keys

1 To display the key for the disks, run the following command:

```
# vxfenadm -s disk_name
```

For example:

- To display the key for the coordinator disk `/dev/sdx` from the system with node ID 1, enter the following command:

```
# vxfenadm -s /dev/sdx
key[1]:
  [Numeric Format]:  86,70,68,69,69,68,48,48
  [Character Format]: VFDEED00
* [Node Format]: Cluster ID: 57069 Node ID: 0 Node Name: sys1
```

The `-s` option of `vxfenadm` displays all eight bytes of a key value in three formats. In the numeric format,

  - The first two bytes, represent the identifier VF, contains the ASCII value 86, 70.
  - The next four bytes contain the ASCII value of the cluster ID 57069 encoded in hex (0xDEED) which are 68, 69, 69, 68.
  - The remaining bytes contain the ASCII value of the node ID 0 (0x00) which are 48, 48. Node ID 1 would be 01 and node ID 10 would be 0A.

An asterisk before the Node Format indicates that the `vxfenadm` command is run from the node of a cluster where LLT is configured and is running.

- To display the keys on a CVM parallel disk group:

```
# vxfenadm -s /dev/vx/rdmp/disk_7

Reading SCSI Registration Keys...

Device Name: /dev/vx/rdmp/disk_7
Total Number Of Keys: 1
key[0]:
  [Numeric Format]:  66,80,71,82,48,48,48,49
  [Character Format]: BPGR0001
  [Node Format]: Cluster ID: unknown  Node ID: 1 Node Name: sys2
```

- To display the keys on a Cluster Server (VCS) failover disk group:

```
# vxfenadm -s /dev/vx/rdmp/disk_8

Reading SCSI Registration Keys...

Device Name: /dev/vx/rdmp/disk_8
Total Number Of Keys: 1
key[0]:
  [Numeric Format]:  65,86,67,83,0,0,0,0
  [Character Format]: AVCS
  [Node Format]: Cluster ID: unknown  Node ID: 0 Node Name: sys1
```

2 To display the keys that are registered in all the disks specified in a disk file:

```
# vxfenadm -s all -f disk_filename
```

For example:

To display all the keys on coordinator disks:

```
# vxfenadm -s all -f /etc/vxfentab

Device Name: /dev/vx/rdmp/disk_9
Total Number Of Keys: 2
key[0]:
[Numeric Format]:  86,70,70,68,57,52,48,49
[Character Format]: VFFD9401
* [Node Format]: Cluster ID: 64916 Node ID: 1   Node Name: sys2
key[1]:
[Numeric Format]:  86,70,70,68,57,52,48,48
[Character Format]: VFFD9400
* [Node Format]: Cluster ID: 64916 Node ID: 0   Node Name: sys1
```

You can verify the cluster ID using the `lltstat -C` command, and the node ID using the `lltstat -N` command. For example:

```
# lltstat -C
57069
```

If the disk has keys that do not belong to a specific cluster, then the `vxfenadm` command cannot look up the node name for the node ID, and hence prints the node name as unknown. For example:

```
Device Name: /dev/vx/rdmp/disk_7
Total Number Of Keys: 1
key[0]:
```

```
[Numeric Format]:  86,70,45,45,45,45,48,49
[Character Format]: VF----01
[Node Format]: Cluster ID: unknown  Node ID: 1  Node Name: sys2
```

For disks with arbitrary format of keys, the `vxfenadm` command prints all the fields as unknown. For example:

```
[Numeric Format]:  65,66,67,68,49,50,51,45
[Character Format]: ABCD123-
[Node Format]: Cluster ID: unknown Node ID: unknown
       Node Name: unknown
```

Verifying that the nodes see the same disk

To confirm whether a disk (or LUN) supports SCSI-3 persistent reservations, two nodes must simultaneously have access to the same disks. Because a shared disk is likely to have a different name on each node, check the serial number to verify the identity of the disk. Use the `vxfenadm` command with the `-i` option to verify that the same serial number for the LUN is returned on all paths to the LUN.

For example, an EMC disk is accessible by the `/dev/sdr` path on node A and the `/dev/sdt` path on node B.

To verify that the nodes see the same disks

1. Verify the connection of the shared storage for data to two of the nodes on which you installed Cluster Server.
2. From node A, enter the following command:

```
# vxfenadm -i /dev/sdr

Vendor id       : EMC
Product id      : SYMMETRIX
Revision        : 5567
Serial Number   : 42031000a
```

The same serial number information should appear when you enter the equivalent command on node B using the `/dev/sdt` path.

On a disk from another manufacturer, Hitachi Data Systems, the output is different and may resemble:

```
# vxfenadm -i /dev/sdt

Vendor id       : HITACHI
Product id      : OPEN-3
Revision        : 0117
Serial Number   : 0401EB6F0002
```

Refer to the `vxfenadm`(1M) manual page for more information.

About the vxfenclearpre utility

You can use the vxfenclearpre utility to remove SCSI-3 registrations and reservations on the disks as well as coordination point servers.

See "Removing preexisting keys" on page 301.

This utility now supports server-based fencing. You can use the vxfenclearpre utility to clear registrations from coordination point servers (CP servers) for the current cluster. The local node from where you run the utility must have the UUID of the current cluster at the /etc/vx/.uuids directory in the clusuuid file. If the UUID file for that cluster is not available on the local node, the utility does not clear registrations from the coordination point servers.

Note: You can use the utility to remove the registration keys and the registrations (reservations) from the set of coordinator disks for any cluster you specify in the command, but you can only clear registrations of your current cluster from the CP servers. Also, you may experience delays while clearing registrations on the coordination point servers because the utility tries to establish a network connection with the IP addresses used by the coordination point servers. The delay may occur because of a network issue or if the IP address is not reachable or is incorrect.

For any issues you encounter with the vxfenclearpre utility, you can you can refer to the log file at, `/var/VRTSvcs/log/vxfen/vxfen.log` file.

See "Issues during fencing startup on VCS cluster nodes set up for server-based fencing" on page 662.

Removing preexisting keys

If you encountered a split-brain condition, use the vxfenclearpre utility to remove CP Servers, SCSI-3 registrations, and reservations on the coordinator disks, Coordination Point servers, as well as on the data disks in all shared disk groups.

You can also use this procedure to remove the registration and reservation keys of another node or other nodes on shared disks or CP server.

To clear keys after split-brain

1. Stop VCS on all nodes.

   ```
   # hastop -all
   ```

2. Make sure that the port h is closed on all the nodes. Run the following command on each node to verify that the port h is closed:

   ```
   # gabconfig -a
   ```

   Port h must not appear in the output.

3. Stop I/O fencing on all nodes. Enter the following command on each node:

   ```
   # systemctl stop vxfen
   ```

4. If you have any applications that run outside of VCS control that have access to the shared storage, then shut down all other nodes in the cluster that have access to the shared storage. This prevents data corruption.

5. Start the vxfenclearpre script:

   ```
   # /opt/VRTSvcs/vxfen/bin/vxfenclearpre
   ```

6 Read the script's introduction and warning. Then, you can choose to let the script run.

```
Do you still want to continue: [y/n] (default : n) y
```

In some cases, informational messages resembling the following may appear on the console of one of the nodes in the cluster when a node is ejected from a disk/LUN. You can ignore these informational messages.

```
<date> <system name> scsi: WARNING: /sbus@3,0/lpfs@0,0/
sd@0,1(sd91):
<date> <system name> Error for Command: <undecoded
cmd 0x5f> Error Level: Informational
<date> <system name> scsi: Requested Block: 0 Error Block 0
<date> <system name> scsi: Vendor: <vendor> Serial Number:
0400759B006E
<date> <system name> scsi: Sense Key: Unit Attention
<date> <system name> scsi: ASC: 0x2a (<vendor unique code
0x2a>), ASCQ: 0x4, FRU: 0x0
```

The script cleans up the disks and displays the following status messages.

```
Cleaning up the coordinator disks...

Cleared keys from n out of n disks,
where n is the total number of disks.

Successfully removed SCSI-3 persistent registrations
from the coordinator disks.

Cleaning up the Coordination Point Servers...

...................
[10.209.80.194]:50001: Cleared all registrations
[10.209.75.118]:443: Cleared all registrations

Successfully removed registrations from the Coordination Point Servers.

Cleaning up the data disks for all shared disk groups ...

Successfully removed SCSI-3 persistent registration and
reservations from the shared data disks.

See the log file /var/VRTSvcs/log/vxfen/vxfen.log
```

```
You can retry starting fencing module. In order to restart the whole
product, you might want to reboot the system.
```

7 Start the fencing module on all the nodes.

```
# systemctl start vxfen
```

8 Start VCS on all nodes.

```
# hastart
```

About the vxfenswap utility

The vxfenswap utility allows you to add, remove, and replace coordinator points in a cluster that is online. The utility verifies that the serial number of the new disks are identical on all the nodes and the new disks can support I/O fencing.

This utility supports both disk-based and server-based fencing.

The utility uses SSH, RSH, or hacli for communication between nodes in the cluster. Before you execute the utility, ensure that communication between nodes is set up in one these communication protocols.

Refer to the `vxfenswap`(1M) manual page.

See the *Cluster Server Installation Guide* for details on the I/O fencing requirements.

You can replace the coordinator disks without stopping I/O fencing in the following cases:

- The disk becomes defective or inoperable and you want to switch to a new disk group.
  See "Replacing I/O fencing coordinator disks when the cluster is online" on page 304.
  See "Replacing the coordinator disk group in a cluster that is online" on page 308.
  If you want to replace the coordinator disks when the cluster is offline, you cannot use the vxfenswap utility. You must manually perform the steps that the utility does to replace the coordinator disks.
  See "Replacing defective disks when the cluster is offline" on page 658.
- New disks are available to act as coordinator disks.
  See "Adding disks from a recovered site to the coordinator disk group" on page 312.
- The keys that are registered on the coordinator disks are lost.

In such a case, the cluster might panic when a network partition occurs. You can replace the coordinator disks with the same disks using the vxfenswap command. During the disk replacement, the missing keys register again without any risk of data corruption.

See "Refreshing lost keys on coordinator disks" on page 314.

In server-based fencing configuration, you can use the vxfenswap utility to perform the following tasks:

- Perform a planned replacement of customized coordination points (CP servers or SCSI-3 disks).

  See "Replacing coordination points for server-based fencing in an online cluster" on page 326.

- Refresh the I/O fencing keys that are registered on the coordination points.

  See "Refreshing registration keys on the coordination points for server-based fencing" on page 328.

You can also use the vxfenswap utility to migrate between the disk-based and the server-based fencing without incurring application downtime in the VCS cluster.

See "Migrating from disk-based to server-based fencing in an online cluster" on page 340.

See "Migrating from server-based to disk-based fencing in an online cluster" on page 341.

If the vxfenswap operation is unsuccessful, then you can use the `-a cancel` of the `vxfenswap` command to manually roll back the changes that the vxfenswap utility does.

- For disk-based fencing, use the `vxfenswap -g diskgroup -a cancel` command to cancel the vxfenswap operation.

  You must run this command if a node fails during the process of disk replacement, or if you aborted the disk replacement.

- For server-based fencing, use the `vxfenswap -a cancel` command to cancel the vxfenswap operation.

Replacing I/O fencing coordinator disks when the cluster is online

Review the procedures to add, remove, or replace one or more coordinator disks in a cluster that is operational.

Warning: The cluster might panic if any node leaves the cluster membership before the vxfenswap script replaces the set of coordinator disks.

To replace a disk in a coordinator disk group when the cluster is online

1. Make sure system-to-system communication is functioning properly.
2. Determine the value of the FaultTolerance attribute.

   ```
   # hares -display coordpoint -attribute FaultTolerance -localclus
   ```

3. Estimate the number of coordination points you plan to use as part of the fencing configuration.
4. Set the value of the FaultTolerance attribute to 0.

   Note: It is necessary to set the value to 0 because later in the procedure you need to reset the value of this attribute to a value that is lower than the number of coordination points. This ensures that the Coordpoint Agent does not fault.

5. Check the existing value of the LevelTwoMonitorFreq attribute.

   ```
   #hares -display coordpoint -attribute LevelTwoMonitorFreq -localclus
   ```

   Note: Make a note of the attribute value before you proceed to the next step. After migration, when you re-enable the attribute you want to set it to the same value.

   You can also run the `hares -display coordpoint` to find out whether the LevelTwoMonitorFreq value is set.

6. Disable level two monitoring of CoordPoint agent.

   ```
   # hares -modify coordpoint LevelTwoMonitorFreq 0
   ```

7 Make sure that the cluster is online.

```
# vxfenadm -d
```

```
I/O Fencing Cluster Information:
================================
Fencing Protocol Version: 201
Fencing Mode: SCSI3
Fencing SCSI3 Disk Policy: dmp
Cluster Members:
  * 0 (sys1)
  1 (sys2)
RFSM State Information:
  node 0 in state 8 (running)
  node 1 in state 8 (running)
```

8 Import the coordinator disk group.

The file `/etc/vxfendg` includes the name of the disk group (typically, vxfencoorddg) that contains the coordinator disks, so use the command:

```
# vxdg -tfC import `cat /etc/vxfendg`
```

where:

-t specifies that the disk group is imported only until the node restarts.

-f specifies that the import is to be done forcibly, which is necessary if one or more disks is not accessible.

-C specifies that any import locks are removed.

9 If your setup uses `VRTSvxvm` *version*, then skip to step 10. You need not set `coordinator=off` to add or remove disks. For other VxVM versions, perform this step:

Where *`version`* is the specific release version.

Turn off the coordinator attribute value for the coordinator disk group.

```
# vxdg -g vxfencoorddg set coordinator=off
```

10 To remove disks from the coordinator disk group, use the VxVM disk administrator utility `vxdiskadm`.

11 Perform the following steps to add new disks to the coordinator disk group:

- Add new disks to the node.
- Initialize the new disks as VxVM disks.

- Check the disks for I/O fencing compliance.
- Add the new disks to the coordinator disk group and set the coordinator attribute value as "on" for the coordinator disk group.

See the *Cluster Server Installation Guide* for detailed instructions.

Note that though the disk group content changes, the I/O fencing remains in the same state.

12 From one node, start the vxfenswap utility. You must specify the disk group to the utility.

The utility performs the following tasks:

- Backs up the existing `/etc/vxfentab` file.
- Creates a test file `/etc/vxfentab.test` for the disk group that is modified on each node.
- Reads the disk group you specified in the vxfenswap command and adds the disk group to the `/etc/vxfentab.test` file on each node.
- Verifies that the serial number of the new disks are identical on all the nodes. The script terminates if the check fails.
- Verifies that the new disks can support I/O fencing on each node.

13 If the disk verification passes, the utility reports success and asks if you want to commit the new set of coordinator disks.

14 Confirm whether you want to clear the keys on the coordination points and proceed with the vxfenswap operation.

```
Do you want to clear the keys on the coordination points
and proceed with the vxfenswap operation? [y/n] (default: n) y
```

15 Review the message that the utility displays and confirm that you want to commit the new set of coordinator disks. Else skip to step 16.

```
Do you wish to commit this change? [y/n] (default: n) y
```

If the utility successfully commits, the utility moves the `/etc/vxfentab.test` file to the `/etc/vxfentab` file.

16 If you do not want to commit the new set of coordinator disks, answer n.

The vxfenswap utility rolls back the disk replacement operation.

17 If coordinator flag was set to off in step 9, then set it on.

```
# vxdg -g vxfencoorddg set coordinator=on
```

18 Deport the diskgroup.

```
# vxdg deport vxfencoorddg
```

19 Re-enable the LevelTwoMonitorFreq attribute of the CoordPoint agent.You may want to use the value that was set before disabling the attribute.

```
# hares -modify coordpoint LevelTwoMonitorFreq Frequencyvalue
```

where *Frequencyvalue* is the value of the attribute.

20 Set the FaultTolerance attribute to a value that is lower than 50% of the total number of coordination points.

For example, if there are four (4) coordination points in your configuration, then the attribute value must be lower than two (2).If you set it to a higher value than two (2) the CoordPoint agent faults.

Replacing the coordinator disk group in a cluster that is online

You can also replace the coordinator disk group using the vxfenswap utility. The following example replaces the coordinator disk group vxfencoorddg with a new disk group vxfendg.

To replace the coordinator disk group

1 Make sure system-to-system communication is functioning properly.

2 Determine the value of the FaultTolerance attribute.

```
# hares -display coordpoint -attribute FaultTolerance -localclus
```

3 Estimate the number of coordination points you plan to use as part of the fencing configuration.

4 Set the value of the FaultTolerance attribute to 0.

Note: It is necessary to set the value to 0 because later in the procedure you need to reset the value of this attribute to a value that is lower than the number of coordination points. This ensures that the Coordpoint Agent does not fault.

---

5 Check the existing value of the LevelTwoMonitorFreq attribute.

```
# hares -display coordpoint -attribute LevelTwoMonitorFreq -localclus
```

Note: Make a note of the attribute value before you proceed to the next step. After migration, when you re-enable the attribute you want to set it to the same value.

6 Disable level two monitoring of CoordPoint agent.

```
# haconf -makerw

# hares -modify coordpoint LevelTwoMonitorFreq 0

# haconf -dump -makero
```

7 Make sure that the cluster is online.

```
# vxfenadm -d


I/O Fencing Cluster Information:
================================
Fencing Protocol Version: 201
Fencing Mode: SCSI3
Fencing SCSI3 Disk Policy: dmp
Cluster Members:
  * 0 (sys1)
  1 (sys2)
RFSM State Information:
  node 0 in state 8 (running)
  node 1 in state 8 (running)
```

8 Find the name of the current coordinator disk group (typically vxfencoorddg) that is in the `/etc/vxfendg` file.

```
# cat /etc/vxfendg
vxfencoorddg
```

9 Find the alternative disk groups available to replace the current coordinator disk group.

```
# vxdisk -o alldgs list

DEVICE    TYPE     DISK  GROUP    STATUS
sda    auto:cdsdisk    -  (vxfendg)    online
sdb    auto:cdsdisk    -  (vxfendg)    online
sdc    auto:cdsdisk    -  (vxfendg)    online
sdx    auto:cdsdisk    -  (vxfencoorddg)    online
sdy    auto:cdsdisk    -  (vxfencoorddg)    online
sdz    auto:cdsdisk    -  (vxfencoorddg)    online
```

10 Validate the new disk group for I/O fencing compliance. Run the following command:

```
# vxfentsthdw -c vxfendg
```

See "Testing the coordinator disk group using the -c option of vxfentsthdw" on page 289.

11 If the new disk group is not already deported, run the following command to deport the disk group:

```
# vxdg deport vxfendg
```

12 Perform one of the following:

- Create the `/etc/vxfenmode.test` file with new fencing mode and disk policy information.
- Edit the existing the `/etc/vxfenmode` with new fencing mode and disk policy information and remove any preexisting `/etc/vxfenmode.test` file.

Note that the format of the /etc/vxfenmode.test file and the /etc/vxfenmode file is the same.

See the *Cluster Server Installation Guide* for more information.

13 From any node, start the vxfenswap utility. For example, if vxfendg is the new disk group that you want to use as the coordinator disk group:

```
# vxfenswap -g vxfendg [-n]
```

The utility performs the following tasks:

- Backs up the existing `/etc/vxfentab` file.

- Creates a test file `/etc/vxfentab.test` for the disk group that is modified on each node.
- Reads the disk group you specified in the vxfenswap command and adds the disk group to the `/etc/vxfentab.test` file on each node.
- Verifies that the serial number of the new disks are identical on all the nodes. The script terminates if the check fails.
- Verifies that the new disk group can support I/O fencing on each node.

14 If the disk verification passes, the utility reports success and asks if you want to replace the coordinator disk group.

15 Confirm whether you want to clear the keys on the coordination points and proceed with the vxfenswap operation.

```
Do you want to clear the keys on the coordination points
and proceed with the vxfenswap operation? [y/n] (default: n) y
```

16 Review the message that the utility displays and confirm that you want to replace the coordinator disk group. Else skip to step 21.

```
Do you wish to commit this change? [y/n] (default: n) y
```

If the utility successfully commits, the utility moves the `/etc/vxfentab.test` file to the `/etc/vxfentab` file.

The utility also updates the `/etc/vxfendg` file with this new disk group.

17 Import the new disk group if it is not already imported before you set the coordinator flag "on".

```
# vxdg -t import vxfendg
```

18 Set the coordinator attribute value as "on" for the new coordinator disk group.

```
# vxdg -g vxfendg set  coordinator=on
```

Set the coordinator attribute value as "off" for the old disk group.

```
# vxdg -g vxfencoorddg set  coordinator=off
```

19 Deport the new disk group.

```
#  vxdg deport vxfendg
```

20 Verify that the coordinator disk group has changed.

```
# cat /etc/vxfendg
vxfendg
```

The swap operation for the coordinator disk group is complete now.

21 If you do not want to replace the coordinator disk group, answer n at the prompt.

The vxfenswap utility rolls back any changes to the coordinator disk group.

22 Re-enable the LevelTwoMonitorFreq attribute of the CoordPoint agent. You may want to use the value that was set before disabling the attribute.

```
# haconf -makerw

# hares -modify coordpoint LevelTwoMonitorFreq Frequencyvalue

# haconf -dump -makero
```

where *Frequencyvalue* is the value of the attribute.

23 Set the FaultTolerance attribute to a value that is lower than 50% of the total number of coordination points.

For example, if there are four (4) coordination points in your configuration, then the attribute value must be lower than two (2).If you set it to a higher value than two (2) the CoordPoint agent faults.

Adding disks from a recovered site to the coordinator disk group

In a campus cluster environment, consider a case where the primary site goes down and the secondary site comes online with a limited set of disks. When the primary site restores, the primary site's disks are also available to act as coordinator disks. You can use the vxfenswap utility to add these disks to the coordinator disk group.

See “About I/O fencing in campus clusters” on page 565.

To add new disks from a recovered site to the coordinator disk group

1. Make sure system-to-system communication is functioning properly.
2. Make sure that the cluster is online.

```
# vxfenadm -d


I/O Fencing Cluster Information:
================================
Fencing Protocol Version: 201
Fencing Mode: SCSI3
Fencing SCSI3 Disk Policy: dmp
Cluster Members:
  * 0 (sys1)
  1 (sys2)
RFSM State Information:
  node 0 in state 8 (running)
  node 1 in state 8 (running)
```

3. Verify the name of the coordinator disk group.

```
# cat /etc/vxfendg
vxfencoorddg
```

4. Run the following command:

```
# vxdisk -o alldgs list

DEVICE    TYPE    DISK  GROUP    STATUS
sdx    auto:cdsdisk    -  (vxfencoorddg)    online
sdy    auto    -  -    offline
sdz    auto    -  -    offline
```

5. Verify the number of disks used in the coordinator disk group.

```
# vxfenconfig -l
I/O Fencing Configuration Information:
======================================
Count                : 1
Disk List
Disk Name        Major  Minor  Serial Number     Policy

/dev/vx/rdmp/sdx       32  48  R450 00013154 0312     dmp
```

6 When the primary site comes online, start the vxfenswap utility on any node in the cluster:

```
# vxfenswap -g vxfencoorddg [-n]
```

7 Verify the count of the coordinator disks.

```
# vxfenconfig -l
I/O Fencing Configuration Information:
======================================
Single Disk Flag      : 0
Count                 : 3
Disk List
Disk Name        Major  Minor  Serial Number     Policy

/dev/vx/rdmp/sdx       32  48  R450 00013154 0312     dmp
/dev/vx/rdmp/sdy       32  32  R450 00013154 0313     dmp
/dev/vx/rdmp/sdz       32  16  R450 00013154 0314     dmp
```

Refreshing lost keys on coordinator disks

If the coordinator disks lose the keys that are registered, the cluster might panic when a network partition occurs.

You can use the vxfenswap utility to replace the coordinator disks with the same disks. The vxfenswap utility registers the missing keys during the disk replacement.

To refresh lost keys on coordinator disks

1. Make sure system-to-system communication is functioning properly.
2. Make sure that the cluster is online.

```
# vxfenadm -d


I/O Fencing Cluster Information:
================================
Fencing Protocol Version: 201
Fencing Mode: SCSI3
Fencing SCSI3 Disk Policy: dmp
Cluster Members:
  * 0 (sys1)
  1 (sys2)
RFSM State Information:
  node 0 in state 8 (running)
  node 1 in state 8 (running)
```

3. Run the following command to view the coordinator disks that do not have keys:

```
# vxfenadm -s all -f /etc/vxfentab


Device Name: /dev/vx/rdmp/sdx
Total Number of Keys: 0
No keys...
...
```

4. Copy the `/etc/vxfenmode` file to the `/etc/vxfenmode.test` file.

   This ensures that the configuration details of both the files are the same.

5 On any node, run the following command to start the vxfenswap utility:

```
# vxfenswap -g vxfencoorddg [-n]
```

6 Verify that the keys are atomically placed on the coordinator disks.

```
# vxfenadm -s all -f /etc/vxfentab


Device Name: /dev/vx/rdmp/sdx
Total Number of Keys: 4
...
```

About administering the coordination point server

This section describes how to perform administrative and maintenance tasks on the coordination point server (CP server).

For more information about the `cpsadm` command and the associated command options, see the `cpsadm`(1M) manual page.

CP server operations (cpsadm)

Table 9-2 lists coordination point server (CP server) operations and required privileges.

Table 9-2 User privileges for CP server operations

CP server operations	CP server Operator	CP server Admin
add_cluster	–	✓
rm_clus	–	✓
add_node	✓	✓
rm_node	✓	✓
add_user	–	✓
rm_user	–	✓
add_clus_to_user	–	✓
rm_clus_from_user	–	✓
reg_node	✓	✓

Table 9-2 User privileges for CP server operations *(continued)*

CP server operations	CP server Operator	CP server Admin
unreg_node	✓	✓
preempt_node	✓	✓
list_membership	✓	✓
list_nodes	✓	✓
list_users	✓	✓
halt_cps	–	✓
db_snapshot	–	✓
ping_cps	✓	✓
client_preupgrade	✓	✓
server_preupgrade	✓	✓
list_protocols	✓	✓
list_version	✓	✓
list_ports	–	✓
add_port	–	✓
rm_port	–	✓

Cloning a CP server

Cloning a CP server greatly reduces the time and effort of assigning a new CP server to your cluster. Since the cloning replicates the existing CP server, you are saved from running the `vxfenswap` utilities on each node connected to the CP server. Therefore, you can perform the CP server maintenance without much hassle.

In this procedure, the following terminology is used to refer to the existing and cloned CP servers:

- cps1: Indicates the existing CP server
- cps2: Indicates the clone of cps1

Prerequisite: Before cloning a CP server, make sure that the existing CP server cps1 is up and running and fencing is configured on at least one client of cps1 in order to verify that the client can talk to cps2 .

To clone the CP server (cps1):

1. Install and configure Cluster Server (VCS) on the system where you want to clone cps1.

   Refer the InfoScale Installation Guide for procedure on installing the Cluster Server

2. Copy the cps1 database and UUID to the system where you want to clone cps1. You can copy the database and UUID from `/etc/VRTScps/db`.

3. Copy CP server configuration file of cps1 from `/etc/vxcps.conf` to the target system for cps2.

4. Copy the CP server and fencing certificates and keys on cps1 from `/var/VRTScps/security` and `/var/VRTSvxfen/security` directories respectively to the target system for cps2.

5. Copy main.cf of cps1 to the target system for cps2 at `/etc/VRTSvcs/conf/config/main.cf`.

6. On the target system for cps2, stop VCS and perform the following in main.cf:

   - Replace all the instances of system name with cps2 system name.
   - Change the device attribute under NIC and IP resources to cps2 values.

7. Copy `/opt/VRTScps/bin/QuorumTypes.cf` from cps1 to the target cps2 system.

8. Offline CP server service group if it is online on cps1.

   ```
   # /opt/VRTSvcs/bin/hagrp  -offline CPSSG -sys <cps1>
   ```

9. Start VCS on the target system for cps2. The CP server service group must come online on target system for cps2 and the system becomes a clone of cps1.

   ```
   # /opt/VRTSvcs/bin/hastart
   ```

10 Run the following command on cps2 to verify if the clone was successful and is running well.

```
# cpsadm -s <server_vip> -a ping_cps
CPS INFO V-97-1400-458 CP server successfully pinged
```

11 Verify if the Co-ordination Point agent service group is Online on the client clusters.

```
# /opt/VRTSvcs/bin/hares -state
Resource           Attribute  System   Value
RES_phantom_vxfen  State      system1  ONLINE
coordpoint         State      system1  ONLINE

# /opt/VRTSvcs/bin/hagrp -state
Group           Attribute      System   Value
vxfen           State          system1  |ONLINE|
```

Note: The cloned cps2 system must be in the same subnet of cps1. Make sure that the port (default 443) used for CP server configuration is free for cps2).

Adding and removing VCS cluster entries from the CP server database

- To add a VCS cluster to the CP server database
  Type the following command:

  ```
  # cpsadm -s cp_server -a add_clus -c cluster_name -u uuid
  ```

- To remove a VCS cluster from the CP server database
  Type the following command:

  ```
  # cpsadm -s cp_server -a rm_clus -u uuid
  ```

cp_server	The CP server's virtual IP address or virtual hostname.
cluster_name	The VCS cluster name.
uuid	The UUID (Universally Unique ID) of the VCS cluster.

Adding and removing a VCS cluster node from the CP server database

- To add a VCS cluster node from the CP server database
  Type the following command:

  ```
  # cpsadm -s cp_server -a add_node -u uuid -n nodeid
  -h host
  ```

- To remove a VCS cluster node from the CP server database
  Type the following command:

  ```
  # cpsadm -s cp_server -a rm_node -u uuid -n nodeid
  ```

cp_server	The CP server's virtual IP address or virtual hostname.
uuid	The UUID (Universally Unique ID) of the VCS cluster.
nodeid	The node id of the VCS cluster node.
host	Hostname

Adding or removing CP server users

- To add a user
  Type the following command:

  ```
  # cpsadm -s cp_server -a add_user -e user_name -f user_role
  -g domain_type -u uuid
  ```

- To remove a user
  Type the following command:

  ```
  # cpsadm -s cp_server -a rm_user -e user_name -g domain_type
  ```

cp_server	The CP server's virtual IP address or virtual hostname.
user_name	The user to be added to the CP server configuration.
user_role	The user role, either cps_admin or cps_operator.
domain_type	The domain type, for example vx, unixpwd, nis, etc.
uuid	The UUID (Universally Unique ID) of the VCS cluster.

Listing the CP server users

To list the CP server users

Type the following command:

```
# cpsadm -s cp_server -a list_users
```

Listing the nodes in all the VCS clusters

To list the nodes in all the VCS cluster

Type the following command:

```
# cpsadm -s cp_server -a list_nodes
```

Listing the membership of nodes in the VCS cluster

To list the membership of nodes in VCS cluster

Type the following command:

```
# cpsadm -s cp_server -a list_membership -c cluster_name
```

cp_server	The CP server's virtual IP address or virtual hostname.
cluster_name	The VCS cluster name.

Preempting a node

Use the following command to preempt a node.

To preempt a node

- Type the following command:

```
# cpsadm -s cp_server -a preempt_node -u uuid -n nodeid
-v victim_node id
```

cp_server	The CP server's virtual IP address or virtual hostname.
uuid	The UUID (Universally Unique ID) of the VCS cluster.
nodeid	The node id of the VCS cluster node.
victim_node id	Node id of one or more victim nodes.

Registering and unregistering a node

- To register a node
  Type the following command:

  ```
  # cpsadm -s cp_server -a reg_node -u uuid -n nodeid
  ```

- To unregister a node
  Type the following command:

  ```
  # cpsadm -s cp_server -a unreg_node -u uuid -n nodeid
  ```

cp_server	The CP server's virtual IP address or virtual hostname.
uuid	The UUID (Universally Unique ID) of the VCS cluster.
nodeid	The nodeid of the VCS cluster node.

Enable and disable access for a user to a VCS cluster

- To enable access for a user to a VCS cluster
  Type the following command:

  ```
  # cpsadm -s cp_server -a add_clus_to_user -e user
  -f user_role -g domain_type -u uuid
  ```

- To disable access for a user to a VCS cluster
  Type the following command:

  ```
  # cpsadm -s cp_server -a rm_clus_from_user -e user_name
  -f user_role -g domain_type -u uuid
  ```

cp_server	The CP server's virtual IP address or virtual hostname.
user_name	The user name to be added to the CP server.
user_role	The user role, either cps_admin or cps_operator.
domain_type	The domain type, for example vx, unixpwd, nis, etc.
uuid	The UUID (Universally Unique ID) of the VCS cluster

Starting and stopping CP server outside VCS control

You can start or stop coordination point server (CP server) outside VCS control.

To start CP server outside VCS control

1. Run the `vxcpserv` binary directly:

   ```
   # /opt/VRTScps/bin/vxcpserv
   ```

   If the command is successful, the command immediately returns without any message.

2. Verify the log file `/var/VRTScps/log/cpserver_A.log` to confirm the state of the CP server.

To stop CP server outside VCS control

1. Run the following command:

   ```
   # cpsadm -s cp_server -a halt_cps
   ```

   The variable *cp_server* represents the CP server's virtual IP address or virtual host name and *port_number* represents the port number on which the CP server is listening.

2. Verify the log file `/var/VRTScps/log/cpserver_A.log` to confirm that the CP server received the halt message and has shut down.

Checking the connectivity of CP servers

To check the connectivity of a CP server

Type the following command:

```
# cpsadm -s cp_server -a ping_cps
```

Adding and removing virtual IP addresses and ports for CP servers at run-time

The procedure of adding and removing virtual IP addresses and ports for CP servers at run-time is only applicable for communication over Veritas Product Authentication Services (AT) and for non-secure communication. It does not apply for communication over HTTPS.

You can use more than one virtual IP address for coordination point server (CP server) communication. You can assign port numbers to each of the virtual IP addresses.

You can use the `cpsadm` command if you want to add or remove virtual IP addresses and ports after your initial CP server setup. However, these virtual IP addresses and ports that you add or remove does not change the `vxcps.conf` file. So, these changes do not persist across CP server restarts.

See the `cpsadm`(1m) manual page for more details.

To add and remove virtual IP addresses and ports for CP servers at run-time

1. To list all the ports that the CP server is configured to listen on, run the following command:

   ```
   # cpsadm -s cp_server -a list_ports
   ```

   If the CP server has not been able to successfully listen on a given port at least once, then the Connect History in the output shows never. If the IP addresses are down when the vxcpserv process starts, vxcpserv binds to the IP addresses when the addresses come up later. For example:

   ```
   # cpsadm -s 127.0.0.1 -a list_ports

   IP Address                      Connect History
   [10.209.79.60]:14250            once
   [10.209.79.61]:56789            once
   [10.209.78.252]:14250           never
   [192.10.10.32]:14250            once
   ```

   CP server does not actively monitor port health. If the CP server successfully listens on any IP:port at least once, then the Connect History for that IP:port shows once even if the port goes down later during CP server's lifetime. You can obtain the latest status of the IP address from the corresponding IP resource state that is configured under VCS.

2. To add a new port (IP:port) for the CP server without restarting the CP server, run the following command:

   ```
   # cpsadm -s cp_server -a add_port
   -i ip_address -r port_number
   ```

   For example:

   ```
   # cpsadm -s 127.0.0.1 -a add_port -i 10.209.78.52 -r 14250
   Port [10.209.78.52]:14250 successfully added.
   ```

3 To stop the CP server from listening on a port (IP:port) without restarting the CP server, run the following command:

```
# cpsadm -s cp_server -a rm_port
-i ip_address -r port_number
```

For example:

```
# cpsadm -s 10.209.78.52 -a rm_port -i 10.209.78.252
No port specified. Assuming default port i.e 14250
Port [10.209.78.252]:14250 successfully removed.
```

Taking a CP server database snapshot

To take a CP server database snapshot

Type the following command:

```
# cpsadm -s cp_server -a db_snapshot
```

The CP server database snapshot is stored at
`/etc/VRTScps/db/cpsdbsnap.DATE.TIME`

Where, *DATE* is the snapshot creation date, and *TIME* is the snapshot creation time.

Replacing coordination points for server-based fencing in an online cluster

Use the following procedure to perform a planned replacement of customized coordination points (CP servers or SCSI-3 disks) without incurring application downtime on an online VCS cluster.

Note: If multiple clusters share the same CP server, you must perform this replacement procedure in each cluster.

You can use the vxfenswap utility to replace coordination points when fencing is running in customized mode in an online cluster, with vxfen_mechanism=cps. The utility also supports migration from server-based fencing (vxfen_mode=customized) to disk-based fencing (vxfen_mode=scsi3) and vice versa in an online cluster.

However, if the VCS cluster has fencing disabled (vxfen_mode=disabled), then you must take the cluster offline to configure disk-based or server-based fencing.

See “Deployment and migration scenarios for CP server” on page 334.

You can cancel the coordination point replacement operation at any time using the `vxfenswap -a cancel` command.

See "About the vxfenswap utility" on page 303.

To replace coordination points for an online cluster

1. Ensure that the VCS cluster nodes and users have been added to the new CP server(s). Run the following commands:

   ```
   # cpsadm -s cpserver -a list_nodes
   # cpsadm -s cpserver -a list_users
   ```

   If the VCS cluster nodes are not present here, prepare the new CP server(s) for use by the VCS cluster.

   See the *Cluster Server Installation Guide* for instructions.

2. Ensure that fencing is running on the cluster using the old set of coordination points and in customized mode.

   For example, enter the following command:

   ```
   # vxfenadm -d
   ```

   The command returns:

   ```
   I/O Fencing Cluster Information:
   ================================
   Fencing Protocol Version: <version>
   Fencing Mode: Customized
   Cluster Members:
   * 0 (sys1)
   1 (sys2)
   RFSM State Information:
   node 0 in state 8 (running)
   node 1 in state 8 (running)
   ```

3. Create a new `/etc/vxfenmode.test` file on each VCS cluster node with the fencing configuration changes such as the CP server information.

   Review and if necessary, update the vxfenmode parameters for security, the coordination points, and if applicable to your configuration, vxfendg.

   Refer to the text information within the `vxfenmode` file for additional information about these parameters and their new possible values.

4 From one of the nodes of the cluster, run the `vxfenswap` utility.

The `vxfenswap` utility requires secure ssh connection to all the cluster nodes. Use –n to use rsh instead of default ssh. Use –p <*protocol*>, where <*protocol*> can be ssh, rsh, or hacli.

```
# vxfenswap [-n | -p <protocol>]
```

5 Review the message that the utility displays and confirm whether you want to commit the change.

- If you do not want to commit the new fencing configuration changes, press Enter or answer n at the prompt.

  ```
  Do you wish to commit this change? [y/n] (default: n) n
  ```

  The `vxfenswap` utility rolls back the migration operation.

- If you want to commit the new fencing configuration changes, answer y at the prompt.

  ```
  Do you wish to commit this change? [y/n] (default: n) y
  ```

  If the utility successfully completes the operation, the utility moves the`/etc/vxfenmode.test` file to the `/etc/vxfenmode` file.

6 Confirm the successful execution of the `vxfenswap` utility by checking the coordination points currently used by the vxfen driver.

For example, run the following command:

```
# vxfenconfig -l
```

Refreshing registration keys on the coordination points for server-based fencing

Replacing keys on a coordination point (CP server) when the VCS cluster is online involves refreshing that coordination point's registrations. You can perform a planned refresh of registrations on a CP server without incurring application downtime on the VCS cluster. You must refresh registrations on a CP server if the CP server agent issues an alert on the loss of such registrations on the CP server database.

The following procedure describes how to refresh the coordination point registrations.

To refresh the registration keys on the coordination points for server-based fencing

1. Ensure that the VCS cluster nodes and users have been added to the new CP server(s). Run the following commands:

   ```
   # cpsadm -s cp_server -a list_nodes
   ```

   ```
   # cpsadm -s cp_server -a list_users
   ```

   If the VCS cluster nodes are not present here, prepare the new CP server(s) for use by the VCS cluster.

   See the *Cluster Server Installation Guide* for instructions.

2. Ensure that fencing is running on the cluster in customized mode using the coordination points mentioned in the `/etc/vxfenmode` file.

   If the /etc/vxfenmode.test file exists, ensure that the information in it and the /etc/vxfenmode file are the same. Otherwise, vxfenswap utility uses information listed in /etc/vxfenmode.test file.

   For example, enter the following command:

   ```
   # vxfenadm -d
   ```

   ```
   ================================
   Fencing Protocol Version: 201
   Fencing Mode: CUSTOMIZED
   Cluster Members:
   * 0 (sys1)
   1 (sys2)
   RFSM State Information:
   node 0 in state 8 (running)
   node 1 in state 8 (running)
   ```

3. List the coordination points currently used by I/O fencing :

   ```
   # vxfenconfig -l
   ```

4. Copy the `/etc/vxfenmode` file to the `/etc/vxfenmode.test` file.

   This ensures that the configuration details of both the files are the same.

5 Run the `vxfenswap` utility from one of the nodes of the cluster.

The `vxfenswap` utility requires secure ssh connection to all the cluster nodes. Use `-n` to use rsh instead of default ssh.

For example:

```
# vxfenswap [-n]
```

The command returns:

```
VERITAS vxfenswap version <version> <platform>
The logfile generated for vxfenswap is
/var/VRTSvcs/log/vxfen/vxfenswap.log.
19156
Please Wait...
VXFEN vxfenconfig NOTICE Driver will use customized fencing
- mechanism cps
Validation of coordination points change has succeeded on
all nodes.
You may commit the changes now.
WARNING: This may cause the whole cluster to panic
if a node leaves membership before the change is complete.
```

6 You are then prompted to commit the change. Enter **y** for yes.

The command returns a confirmation of successful coordination point replacement.

7 Confirm the successful execution of the vxfenswap utility. If CP agent is configured, it should report ONLINE as it succeeds to find the registrations on coordination points. The registrations on the CP server and coordinator disks can be viewed using the `cpsadm` and `vxfenadm` utilities respectively.

Note that a running online coordination point refreshment operation can be canceled at any time using the command:

```
# vxfenswap -a cancel
```

About configuring a CP server to support IPv6 or dual stack

You may want to configure a CP server to support IPv6 or dual-stack configurations in the following scenarios:

- When configuring a new CP server to support pure IPv6
  See "Configuring a new CP server to support pure IPv6" on page 331.

- When configuring an existing CP server to support IPv6 or dual stack
  See "Configuring an existing CP server to support IPv6 or dual stack" on page 331.

Configuring a new CP server to support pure IPv6

Configure the CP server manually.

For instructions, see the *Cluster Server Configuration and Upgrade Guide*.

While performing the following tasks, ensure that you specify IPv6 addresses instead of IPv4 addresses:

- Manually generating the key and the server certificates for the CP server
- Manually generating the client key and certificates on the client nodes
- Creating the `/etc/vxcps.conf` configuration file

Configuring an existing CP server to support IPv6 or dual stack

Perform the following steps to manually configure the CP server in HTTPS-based mode to support IPv6 or dual stack:

1. Modify the `/etc/vxcps.conf` file to include the IPv6 address of the CP server.

   If the CP server should support pure IPv6 communication, remove the existing IPv4 entries and add the new IPv6 addresses in the configuration file. If the CP server should support IPv6 and IPv4 communications, add the IPv6 addresses along with the existing IPv4 addresses.

2. Generate the server certificate for the CP server to facilitate communication over the IPv6 channel along with IPv4.

   Perform these tasks sequentially:

   - If not already present, create an OpenSSL configuration file (https_ssl_cert.conf) to add the new DNS.
   - Edit the `https_ssl_cert.conf` file to add DNS entries for the IPv4 and the IPv6 addresses.
     To support communication over pure IPv6 networks, remove the existing IPv4 entries from the file.
     For example:

     ```
     [req]
     distinguished_name = req_distinguished_name
     req_extensions = v3_req

     [req_distinguished_name]
     ```

```
countryName = Country Name (2 letter code. eg, US)
countryName_default = US
localityName = Locality Name (eg, city)
organizationalUnitName = Organizational Unit Name (eg, section)
commonName = Common Name (eg, YOUR name)
commonName_max = 64
emailAddress = Email Address
emailAddress_max = 40

[v3_req]
keyUsage = keyEncipherment, dataEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
DNS.1 = cpsone.company.com
DNS.2 = ipv6Address
DNS.3 = ipv4Address
```

- Recreate the server certificates by reusing the CA certificate, the server key, the newly created `https_ssl_cert.conf` file, and the cluster UUID.

Note: The CA certificate and the server key are already present on the setup. Get the cluster UUID from the `/etc/vx/.uuids/clusuuid` file.

```
# /opt/VRTSperl/non-perl-libs/bin/openssl req
  -new -key /var/VRTScps/security/keys/server_private.key
  -config https_ssl_cert.conf -subj
  '/C=US/L=city/OU=section/CN={<UUID>}'
  -out /var/VRTScps/security/certs/server.csr

# /opt/VRTSperl/non-perl-libs/bin/openssl x509 -req -days 100
  -in /var/VRTScps/security/certs/server.csr
  -CA /var/VRTScps/security/certs/ca.crt
  -CAkey /var/VRTScps/security/keys/ca.key
  -set_serial 01 -extensions v3_req
  -extfile https_ssl_cert.conf
  -out /var/VRTScps/security/certs/server.crt
```

3. On the CP server, create a copy of the existing client certificates with the IPv4 addresses and rename the copy to include the IPv6 addresses in the certificate name.

For example, if the IPv6 address is `2002::2` and the hostname is `xyz`:

```
# cp /var/VRTSvxfen/security/certs/ca_xyz.crt
  /var/VRTSvxfen/security/certs/ca_2002\:\:2.crt

# cp /var/VRTSvxfen/security/certs/client_xyz.crt
  /var/VRTSvxfen/security/certs/client_2002\:\:2.crt
```

4. On each client node, create a copy of the existing client certificates with the IPv4 addresses and rename the copy to include IPv6 addresses in the certificate name.

   For example, if the IPv4 address is `10.209.81.122` and IPv6 address is `2002::2`:

```
# cp /var/VRTSvxfen/security/certs/ca_10.209.81.122.crt
  /var/VRTSvxfen/security/certs/ca_2002\:\:2.crt

# cp /var/VRTSvxfen/security/certs/client_10.209.81.122.crt
  /var/VRTSvxfen/security/certs/client_2002\:\:2.crt
```

5. Stop VCS on the CP server.

```
# hastop -local
```

6. Update the `main.cf` file to include the newly added IPv6 resources, the quorum resource, and the dependencies for the newly added IPv6 resources.

```
IP cpsvip2
(
Critical = 0
Device @cps1 = eth1
Address = "ipv6Address"
PrefixLen = 64
)
NIC cpsnic2 (
Critical = 0
Device @cps1 = eth1
NetworkHosts @cps1 = {ipv6AddressOfNetworkHost}
           )

Quorum quorum (
                QuorumResources = { cpsvip1, cpsvip2 }
                )
```

```
cpsvip1 requires cpsnic1
cpsvip2 requires cpsnic2
vxcpserv requires quorum
```

7. Restart the CP server.
   - To start VCS in a single-node cluster, run `# hastart -onenode`.
   - To start VCS in an SFHA cluster, run `# hastart`.
8. Perform the following tasks sequentially on each client node:
   - Create the `/etc/vxfenmode.test` file with the new IPv6 address of the CP server.
   - From any client node, start the `vxfenswap` utility.
   - Verify that fencing is running successfully on each node using the `vxfenadm -d` command.

Deployment and migration scenarios for CP server

Table 9-3 describes the supported deployment and migration scenarios, and the procedures you must perform on the VCS cluster and the CP server.

Table 9-3 CP server deployment and migration scenarios

Scenario	CP server	VCS cluster	Action required
Setup of CP server for a VCS cluster for the first time	New CP server	New VCS cluster using CP server as coordination point	On the designated CP server, perform the following tasks: **1** Prepare to configure the new CP server. **2** Configure the new CP server. **3** Prepare the new CP server for use by the VCS cluster. On the VCS cluster nodes, configure server-based I/O fencing. See the *Cluster Server Configuration and Upgrade Guide* for the procedures.

Table 9-3 CP server deployment and migration scenarios *(continued)*

Scenario	CP server	VCS cluster	Action required
Add a new VCS cluster to an existing and operational CP server	Existing and operational CP server	New VCS cluster	On the VCS cluster nodes, configure server-based I/O fencing. See the *Cluster Server Configuration and Upgrade Guide* for the procedures. **Note:** If the existing CP server only supports IPv4, and the new cluster that you want to add has nodes configured to use IPv6 address for communication with the CP server, you must migrate the existing CP server to IPv6 or dual stack configuration. See "Configuring an existing CP server to support IPv6 or dual stack" on page 331.
Replace the coordination point from an existing CP server to a new CP server	New CP server	Existing VCS cluster using CP server as coordination point	On the designated CP server, perform the following tasks: **1** Prepare to configure the new CP server. **2** Configure the new CP server. **3** Prepare the new CP server for use by the VCS cluster. See the *Cluster Server Configuration and Upgrade Guide* for the procedures. On a node in the VCS cluster, run the `vxfenswap` command to move to replace the CP server: See "Replacing coordination points for server-based fencing in an online cluster" on page 326.
Replace the coordination point from an existing CP server to an operational CP server coordination point	Operational CP server	Existing VCS cluster using CP server as coordination point	On a node in the VCS cluster, run the `vxfenswap` command to move to replace the CP server: See "Replacing coordination points for server-based fencing in an online cluster" on page 326.

Table 9-3 CP server deployment and migration scenarios *(continued)*

Scenario	CP server	VCS cluster	Action required

Enabling fencing in a VCS cluster with a new CP server coordination point	New CP server	Existing VCS cluster with fencing configured in disabled mode	**Note:** Migrating from fencing in disabled mode to customized mode incurs application downtime on the VCS cluster.
			
			On the designated CP server, perform the following tasks:
			1 Prepare to configure the new CP server.
			2 Configure the new CP server
			3 Prepare the new CP server for use by the VCS cluster
			See the *Cluster Server Configuration and Upgrade Guide* for the procedures.
			
			On the VCS cluster nodes, perform the following:
			1 Stop all applications, VCS, and fencing on the VCS cluster.
			2 To stop VCS, use the following command (to be run on all the VCS cluster nodes):
			`# hastop -local`
			3 Stop fencing using the following command:
			For systemd environments with supported Linux distributions:
			`# systemctl stop vxfen`
			For other supported Linux distributions:
			`# /etc/init.d/vxfen stop`
			4 Reconfigure I/O fencing on the VCS cluster.
			See the *Cluster Server Configuration and Upgrade Guide* for the procedures.

Table 9-3 CP server deployment and migration scenarios *(continued)*

Scenario	CP server	VCS cluster	Action required

Enabling fencing in a VCS cluster with an operational CP server coordination point	Operational CP server	Existing VCS cluster with fencing configured in disabled mode	**Note:** Migrating from fencing in disabled mode to customized mode incurs application downtime.
			
			On the designated CP server, prepare to configure the new CP server.
			
			See the *Cluster Server Configuration and Upgrade Guide* for this procedure.
			
			On the VCS cluster nodes, perform the following tasks:
			
			1 Stop all applications, VCS, and fencing on the VCS cluster.
			
			2 To stop VCS, use the following command (to be run on all the VCS cluster nodes):
			
			`# hastop -local`
			
			3 Stop fencing using the following command:
			
			For systemd environments with supported Linux distributions:
			
			`# systemctl stop vxfen`
			
			For other supported Linux distributions:
			
			`# /etc/init.d/vxfen stop`
			
			4 Reconfigure fencing on the VCS cluster.
			
			See the *Cluster Server Configuration and Upgrade Guide* for the procedures.

Table 9-3 CP server deployment and migration scenarios *(continued)*

Scenario	CP server	VCS cluster	Action required

Enabling fencing in a VCS cluster with a new CP server coordination point	New CP server	Existing VCS cluster with fencing configured in scsi3 mode	On the designated CP server, perform the following tasks:
			1 Prepare to configure the new CP server.
			2 Configure the new CP server
			3 Prepare the new CP server for use by the VCS cluster
			See the *Cluster Server Configuration and Upgrade Guide* for the procedures.
			Based on whether the cluster is online or offline, perform the following procedures:
			For a cluster that is online, perform the following task on the VCS cluster:
			◆ Run the `vxfenswap` command to migrate from disk-based fencing to the server-based fencing.
			See "Migrating from disk-based to server-based fencing in an online cluster" on page 340.
			For a cluster that is offline, perform the following tasks on the VCS cluster:
			1 Stop all applications, VCS, and fencing on the VCS cluster.
			2 To stop VCS, use the following command (to be run on all the VCS cluster nodes):
			`# hastop -local`
			3 Stop fencing using the following command:
			For systemd environments with supported Linux distributions:
			`# systemctl stop vxfen`
			For other supported Linux distributions:
			`# /etc/init.d/vxfen stop`
			4 Reconfigure I/O fencing on the VCS cluster.
			See the *Cluster Server Configuration and Upgrade Guide* for the procedures.

Table 9-3 CP server deployment and migration scenarios *(continued)*

Scenario	CP server	VCS cluster	Action required

Enabling fencing in a VCS cluster with an operational CP server coordination point	Operational CP server	Existing VCS cluster with fencing configured in disabled mode	Based on whether the cluster is online or offline, perform the following procedures:
			
			For a cluster that is online, perform the following task on the VCS cluster:
			
			◆ Run the `vxfenswap` command to migrate from disk-based fencing to the server-based fencing.
			
			See "Migrating from disk-based to server-based fencing in an online cluster" on page 340.
			
			For a cluster that is offline, perform the following tasks on the VCS cluster:
			
			1 Stop all applications, VCS, and fencing on the VCS cluster.
			
			2 To stop VCS, use the following command (to be run on all the VCS cluster nodes):
			
			`# hastop -local`
			
			3 Stop fencing using the following command:
			
			For systemd environments with supported Linux distributions:
			
			`# systemctl stop vxfen`
			
			For other supported Linux distributions:
			
			`# /etc/init.d/vxfen stop`
			
			4 Reconfigure fencing on the VCS cluster.
			
			See the *Cluster Server Configuration and Upgrade Guide* for the procedures.

Refreshing registrations of VCS cluster nodes on coordination points (CP servers/ coordinator disks) without incurring application downtime	Operational CP server	Existing VCS cluster using the CP server as coordination point	On the VCS cluster run the `vxfenswap` command to refresh the keys on the CP server: See "Refreshing registration keys on the coordination points for server-based fencing" on page 328.

About migrating between disk-based and server-based fencing configurations

You can migrate between fencing configurations without incurring application downtime in the VCS clusters.

You can migrate from disk-based fencing to server-based fencing in the following cases:

- You want to leverage the benefits of server-based fencing.
- You want to replace faulty coordinator disks with coordination point servers (CP servers).

See "Migrating from disk-based to server-based fencing in an online cluster" on page 340.

Similarly, you can migrate from server-based fencing to disk-based fencing when you want to perform maintenance tasks on the CP server systems.

See "Migrating from server-based to disk-based fencing in an online cluster" on page 341.

Migrating from disk-based to server-based fencing in an online cluster

You can either use the installer or manually migrate from disk-based fencing to server-based fencing without incurring application downtime in the VCS clusters.

See "About migrating between disk-based and server-based fencing configurations" on page 340.

You can also use response files to migrate between fencing configurations.

See "Migrating between fencing configurations using response files" on page 341.

Warning: The cluster might panic if any node leaves the cluster membership before the coordination points migration operation completes.

This section covers the following procedures:

Migrating using the script-based installer

Migrating manually

Migrating from server-based to disk-based fencing in an online cluster

You can either use the installer or manually migrate from server-based fencing to disk-based fencing without incurring application downtime in the VCS clusters.

See "About migrating between disk-based and server-based fencing configurations" on page 340.

You can also use response files to migrate between fencing configurations.

See "Migrating between fencing configurations using response files" on page 341.

Warning: The cluster might panic if any node leaves the cluster membership before the coordination points migration operation completes.

This section covers the following procedures:

Migrating using the script-based installer

Migrating manually

Migrating between fencing configurations using response files

Typically, you can use the response file that the installer generates after you migrate between I/O fencing configurations. Edit these response files to perform an automated fencing reconfiguration in the VCS cluster.

To configure I/O fencing using response files

1. Make sure that VCS is configured.
2. Make sure system-to-system communication is functioning properly.

3 Make sure that the VCS cluster is online and uses either disk-based or server-based fencing.

```
# vxfenadm -d
```

For example, if VCS cluster uses disk-based fencing:

```
I/O Fencing Cluster Information:
================================
Fencing Protocol Version: 201
Fencing Mode: SCSI3
Fencing SCSI3 Disk Policy: dmp
Cluster Members:
  * 0 (sys1)
  1 (sys2)
RFSM State Information:
  node 0 in state 8 (running)
  node 1 in state 8 (running)
```

For example, if the VCS cluster uses server-based fencing:

```
I/O Fencing Cluster Information:
================================
Fencing Protocol Version: 201
Fencing Mode: Customized
Fencing Mechanism: cps
Cluster Members:
  * 0 (sys1)
  1 (sys2)
RFSM State Information:
  node 0 in state 8 (running)
  node 1 in state 8 (running)
```

4 Copy the response file to one of the cluster systems where you want to configure I/O fencing.

Review the sample files to reconfigure I/O fencing.

See "Sample response file to migrate from disk-based to server-based fencing" on page 343.

See "Sample response file to migrate from server-based fencing to disk-based fencing" on page 344.

See "Sample response file to migrate from single CP server-based fencing to server-based fencing" on page 344.

5 Edit the values of the response file variables as necessary.

See "Response file variables to migrate between fencing configurations" on page 344.

6 Start the I/O fencing reconfiguration from the system to which you copied the response file. For example:

```
# /opt/VRTS/install/installer<version> -responsefile /tmp/
\ response_file
```

Where <*version*> is the specific release version, and /tmp/*response_file* is the response file's full path name.

Sample response file to migrate from disk-based to server-based fencing

The following is a sample response file to migrate from disk-based fencing with three coordinator disks to server-based fencing with one CP server and two coordinator disks:

```
$CFG{disks_to_remove}=[ qw(emc_clariion0_62) ];
$CFG{fencing_cps}=[ qw(10.198.89.251)];
$CFG{fencing_cps_ports}{"10.198.89.204"}=14250;
$CFG{fencing_cps_ports}{"10.198.89.251"}=14250;
$CFG{fencing_cps_vips}{"10.198.89.251"}=[ qw(10.198.89.251 10.198.89.204) ];
$CFG{fencing_ncp}=1;
$CFG{fencing_option}=4;
$CFG{opt}{configure}=1;
$CFG{opt}{fencing}=1;
$CFG{prod}="VCS60";
$CFG{systems}=[ qw(sys1 sys2) ];
$CFG{vcs_clusterid}=22462;
$CFG{vcs_clustername}="clus1";
```

Sample response file to migrate from server-based fencing to disk-based fencing

The following is a sample response file to migrate from server-based fencing with one CP server and two coordinator disks to disk-based fencing with three coordinator disks:

```
$CFG{fencing_disks}=[ qw(emc_clariion0_66) ];
$CFG{fencing_mode}="scsi3";
$CFG{fencing_ncp}=1;
$CFG{fencing_ndisks}=1;
$CFG{fencing_option}=4;
$CFG{opt}{configure}=1;
$CFG{opt}{fencing}=1;
$CFG{prod}="VCS60";
$CFG{servers_to_remove}=[ qw([10.198.89.251]:14250) ];
$CFG{systems}=[ qw(sys1 sys2) ];
$CFG{vcs_clusterid}=42076;
$CFG{vcs_clustername}="clus1";
```

Sample response file to migrate from single CP server-based fencing to server-based fencing

The following is a sample response file to migrate from single CP server-based fencing to server-based fencing with one CP server and two coordinator disks:

```
$CFG{fencing_disks}=[ qw(emc_clariion0_62 emc_clariion0_65) ];
$CFG{fencing_dgname}="fendg";
$CFG{fencing_scsi3_disk_policy}="dmp";
$CFG{fencing_ncp}=2;
$CFG{fencing_ndisks}=2;
$CFG{fencing_option}=4;
$CFG{opt}{configure}=1;
$CFG{opt}{fencing}=1;
$CFG{prod}="VCS60";
$CFG{systems}=[ qw(sys1 sys2) ];
$CFG{vcs_clusterid}=42076;
$CFG{vcs_clustername}="clus1";
```

Response file variables to migrate between fencing configurations

Table 9-4 lists the response file variables that specify the required information to migrate between fencing configurations for VCS.

Table 9-4 Response file variables specific to migrate between fencing configurations

Variable	List or Scalar	Description
CFG {fencing_option}	Scalar	Specifies the I/O fencing configuration mode. ■ 1—Coordination Point Server-based I/O fencing ■ 2—Coordinator disk-based I/O fencing ■ 3—Disabled mode ■ 4—Fencing migration when the cluster is online (Required)
CFG {fencing_reusedisk}	Scalar	If you migrate to disk-based fencing or to server-based fencing that uses coordinator disks, specifies whether to use free disks or disks that already belong to a disk group. ■ 0—Use free disks as coordinator disks ■ 1—Use disks that already belong to a disk group as coordinator disks (before configuring these as coordinator disks, installer removes the disks from the disk group that the disks belonged to.) (Required if your fencing configuration uses coordinator disks)
CFG {fencing_ncp}	Scalar	Specifies the number of new coordination points to be added. (Required)
CFG {fencing_ndisks}	Scalar	Specifies the number of disks in the coordination points to be added. (Required if your fencing configuration uses coordinator disks)

Table 9-4 Response file variables specific to migrate between fencing configurations *(continued)*

Variable	List or Scalar	Description
CFG {fencing_disks}	List	Specifies the disks in the coordination points to be added. (Required if your fencing configuration uses coordinator disks)
CFG {fencing_dgname}	Scalar	Specifies the disk group that the coordinator disks are in. (Required if your fencing configuration uses coordinator disks)
CFG {fencing_scsi3_disk_policy}	Scalar	Specifies the disk policy that the disks must use. (Required if your fencing configuration uses coordinator disks)
CFG {fencing_cps}	List	Specifies the CP servers in the coordination points to be added. (Required for server-based fencing)
CFG {fencing_cps_vips}{$vip1}	List	Specifies the virtual IP addresses or the fully qualified host names of the new CP server. (Required for server-based fencing)
CFG {fencing_cps_ports}{$vip}	Scalar	Specifies the port that the virtual IP of the new CP server must listen on. If you do not specify, the default value is 14250. (Optional)
CFG {servers_to_remove}	List	Specifies the CP servers in the coordination points to be removed.
CFG {disks_to_remove}	List	Specifies the disks in the coordination points to be removed
CFG{donotreconfigurevcs}	Scalar	Defines if you need to re-configure VCS. (Optional)

Table 9-4 Response file variables specific to migrate between fencing configurations *(continued)*

Variable	List or Scalar	Description
CFG{donotreconfigurefencing}	Scalar	defines if you need to re-configure fencing. (Optional)

Enabling or disabling the preferred fencing policy

You can enable or disable the preferred fencing feature for your I/O fencing configuration.

You can enable preferred fencing to use system-based race policy, group-based race policy, or site-based policy. If you disable preferred fencing, the I/O fencing configuration uses the default count-based race policy.

Preferred fencing is not applicable to majority-based I/O fencing.

See "About preferred fencing" on page 248.

See "How preferred fencing works" on page 251.

To enable preferred fencing for the I/O fencing configuration

1. Make sure that the cluster is running with I/O fencing set up.

```
# vxfenadm -d
```

2. Make sure that the cluster-level attribute UseFence has the value set to SCSI3.

```
# haclus -value UseFence
```

3. To enable system-based race policy, perform the following steps:
   - Make the VCS configuration writable.

```
# haconf -makerw
```

   - Set the value of the cluster-level attribute PreferredFencingPolicy as System.

```
# haclus -modify PreferredFencingPolicy System
```

   - Set the value of the system-level attribute FencingWeight for each node in the cluster.

For example, in a two-node cluster, where you want to assign sys1 five times more weight compared to sys2, run the following commands:

```
# hasys -modify sys1 FencingWeight 50
# hasys -modify sys2 FencingWeight 10
```

- Save the VCS configuration.

```
# haconf -dump -makero
```

- Verify fencing node weights using:

```
# vxfenconfig -a
```

4 To enable group-based race policy, perform the following steps:

- Make the VCS configuration writable.

```
# haconf -makerw
```

- Set the value of the cluster-level attribute PreferredFencingPolicy as Group.

```
# haclus -modify PreferredFencingPolicy Group
```

- Set the value of the group-level attribute Priority for each service group. For example, run the following command:

```
# hagrp -modify service_group Priority 1
```

Make sure that you assign a parent service group an equal or lower priority than its child service group. In case the parent and the child service groups are hosted in different subclusters, then the subcluster that hosts the child service group gets higher preference.

- Save the VCS configuration.

```
# haconf -dump -makero
```

5 To enable site-based race policy, perform the following steps:

- Make the VCS configuration writable.

```
# haconf -makerw
```

- Set the value of the cluster-level attribute PreferredFencingPolicy as Site.

```
# haclus -modify PreferredFencingPolicy Site
```

- Set the value of the site-level attribute Preference for each site.

  ```
  For example,
  # hasite -modify Pune Preference 2
  ```

- Save the VCS configuration.

  ```
  # haconf -dump -makero
  ```

6 To view the fencing node weights that are currently set in the fencing driver, run the following command:

```
# vxfenconfig -a
```

To disable preferred fencing for the I/O fencing configuration

1 Make sure that the cluster is running with I/O fencing set up.

```
# vxfenadm -d
```

2 Make sure that the cluster-level attribute UseFence has the value set to SCSI3.

```
# haclus -value UseFence
```

3 To disable preferred fencing and use the default race policy, set the value of the cluster-level attribute PreferredFencingPolicy as Disabled.

```
# haconf -makerw
# haclus -modify PreferredFencingPolicy Disabled
# haconf -dump -makero
```

About I/O fencing log files

Refer to the appropriate log files for information related to a specific utility. The log file location for each utility or command is as follows:

- vxfen start and stop logs: `/var/VRTSvcs/log/vxfen/vxfen.log`
- vxfenclearpre utility: `/var/VRTSvcs/log/vxfen/vxfen.log`
- vxfenctl utility: `/var/VRTSvcs/log/vxfen/vxfenctl.log*`
- vxfenswap utility: `/var/VRTSvcs/log/vxfen/vxfenswap.log*`
- vxfentsthdw utility: `/var/VRTSvcs/log/vxfen/vxfentsthdw.log*`

The asterisk ,*, represents a number that differs for each invocation of the command.

Chapter 10

Controlling VCS behavior

This chapter includes the following topics:

- VCS behavior on resource faults
- About controlling VCS behavior at the service group level
- About controlling VCS behavior at the resource level
- Changing agent file paths and binaries
- VCS behavior on loss of storage connectivity
- Service group workload management
- Sample configurations depicting workload management

VCS behavior on resource faults

VCS considers a resource faulted in the following situations:

- When the resource state changes unexpectedly. For example, an online resource going offline.
- When a required state change does not occur. For example, a resource failing to go online or offline when commanded to do so.

In many situations, VCS agents take predefined actions to correct the issue before reporting resource failure to the engine. For example, the agent may try to bring a resource online several times before declaring a fault.

When a resource faults, VCS takes automated actions to clean up the faulted resource. The Clean function makes sure the resource is completely shut down before bringing it online on another node. This prevents concurrency violations.

When a resource faults, VCS takes all resources dependent on the faulted resource offline. The fault is thus propagated in the service group

Critical and non-critical resources

The Critical attribute for a resource defines whether a service group fails over when the resource faults. If a resource is configured as non-critical (by setting the Critical attribute to 0) and no resources depending on the failed resource are critical, the service group will not failover. VCS takes the failed resource offline and updates the group's status to PARTIAL. The attribute also determines whether a service group tries to come online on another node if, during the group's online process, a resource fails to come online.

In atleast resource dependency, criticality is overridden. If the minimum criteria of atleast resource dependency is met:

- Service group does not failover even if a critical resource faults.
- Fault is not propagated up in the dependency tree even if a non-critical resource faults.
- Service group remains in ONLINE state in spite of resource faults.

VCS behavior diagrams

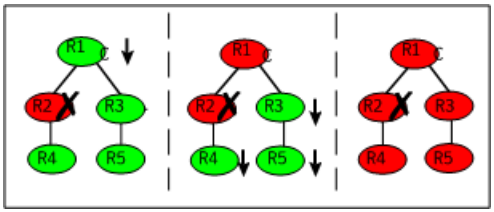
Figure 10-1 displays the symbols used for resource configuration and color codes.

Figure 10-1 Symbols for resource configuration/actions and color codes

Example scenario 1: Resource with critical parent faults

Figure 10-2 shows an example of a service group with five resources, of which resource R1 is configured as a critical resource.

Figure 10-2 Scenario 1: Resource with critical parent faults

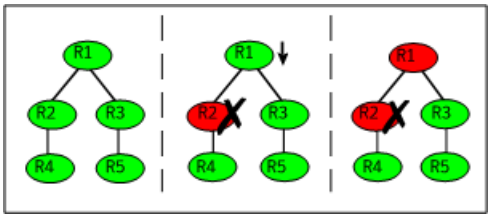


When resource R2 faults, the fault is propagated up the dependency tree to resource R1. When the critical resource R1 goes offline, VCS must fault the service group and fail it over elsewhere in the cluster. VCS takes other resources in the service group offline in the order of their dependencies. After taking resources R3, R4, and R5 offline, VCS fails over the service group to another node.

Example scenario 2: Resource with non-critical parent faults

Figure 10-3 shows an example of a service group that does not have any critical resources.

Figure 10-3 Scenario 2: Resource with non-critical parent faults

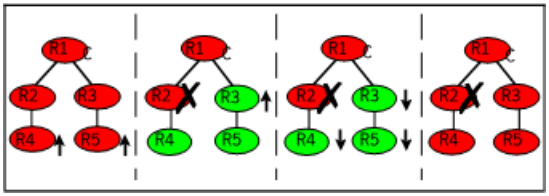


When resource R2 faults, the engine propagates the failure up the dependency tree. Neither resource R1 nor resource R2 are critical, so the fault does not result in the tree going offline or in service group failover.

Example scenario 3: Resource with critical parent fails to come online

Figure 10-4 shows an example where a command is issued to bring the service group online and resource R2 fails to come online.

Figure 10-4 Scenario 3: Resource with critical parent fails to come online

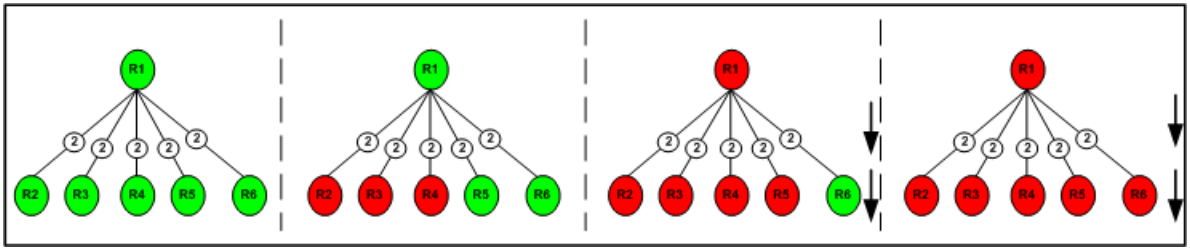


VCS calls the Clean function for resource R2 and propagates the fault up the dependency tree. Resource R1 is set to critical, so the service group is taken offline and failed over to another node in the cluster.

Example scenario 4: Resource with atleast resource dependency faults

Figure 10-5 shows an example where the parent resource (R1) depends on five child resources (R2, R3, R4, R5, and R6). Out of these five resources, at least two resources must be online for the parent resource to remain online.

Figure 10-5 Example scenario 4: Resource with atleast resource dependency faults



VCS tolerates the fault of R2, R3, and R4 as the minimum criteria of 2 is met. When R5 faults, only R6 is online and the number of online child resources goes below the minimum criteria of 2 and the fault is propagated up the dependency tree to the parent resource R1 and service group is taken offline.

About controlling VCS behavior at the service group level

You can configure service group attributes to modify VCS behavior in response to resource faults.

About the AutoRestart attribute

If a persistent resource on a service group (GROUP_1) faults, VCS fails the service group over to another system if the following conditions are met:

- The AutoFailOver attribute is set.
- Another system in the cluster exists to which GROUP_1 can fail over.

If neither of these conditions is met, GROUP_1 remains offline and faulted, even after the faulted resource becomes online.

Setting the AutoRestart attribute enables a service group to be brought back online without manual intervention. If no failover targets are available, setting the AutoRestart attribute enables VCS to bring the group back online on the first available system after the group's faulted resource came online on that system.

For example, NIC is a persistent resource. In some cases, when a system boots and VCS starts, VCS probes all resources on the system. When VCS probes the NIC resource, the resource may not be online because the networking is not up and fully operational. In such situations, VCS marks the NIC resource as faulted, and does not bring the service group online. However, when the NIC resource becomes online and if AutoRestart is enabled, the service group is brought online.

About controlling failover on service group or system faults

The AutoFailOver attribute configures service group behavior in response to service group and system faults.

The possible values include 0, 1, and 2. You can set the value of this attribute as 2 if you have enabled the HA/DR license and if the service group is a non-hybrid service group.

Table 10-1 shows the possible values for the attribute AutoFailover.

Table 10-1 Possible values of the AutoFailover attribute and their description

AutoFailover attribute value	Description
0	VCS does not fail over the service group when a system or service group faults. If a fault occurs in a service group, the group is taken offline, depending on whether any of its resources are configured as critical. If a system faults, the service group is not failed over to another system.
1	VCS automatically fails over the service group when a system or a service group faults, provided a suitable node exists for failover. The service group attributes SystemZones and FailOverPolicy impact the failover behavior of the service group. For global clusters, the failover decision is also based on the ClusterFailOverPolicy. See “Service group attributes” on page 719.
2	VCS automatically fails over the service group only if another suitable node exists in the same system zone or sites. If a suitable node does not exist in the same system zone or sites, VCS brings the service group offline, and generates an alert for administrator’s intervention. You can manually bring the group online using the `hagrp -online` command. **Note:** If SystemZones attribute is not defined, the failover behavior is similar to AutoFailOver=1.

About defining failover policies

The service group attribute FailOverPolicy governs how VCS calculates the target system for failover.

Table 10-2 shows the possible values for the attribute FailoverPolicy.

Table 10-2 Possible values of the FailOverPolicy attribute and their description

FailOverPolicy attribute value	Description
Priority	VCS selects the system with the lowest priority as the failover target. The Priority failover policy is ideal for simple two-node clusters or small clusters with few service groups. Priority is set in the SystemList attribute implicitly by ordering, such as SystemList = {SystemA, SystemB} or explicitly, such as SystemList = {SystemA=0, SystemB=1}. Priority is the default behavior.
RoundRobin	VCS selects the system running the fewest service groups as the failover target. This policy is ideal for large clusters running many service groups with similar server load characteristics (for example, similar databases or applications)
Load	The Load failover policy comprises the following components: System capacity and service group load, represented by the attributes Capacity and Load respectively. See System capacity and service group load on page 387. System limits and service group prerequisites, represented by the attributes Limits and Prerequisites, respectively. You cannot set the service group attribute FailOverPolicy to Load if the cluster attribute Statistics is enabled. See System limits and service group prerequisites on page 389.

Table 10-2 Possible values of the FailOverPolicy attribute and their description *(continued)*

FailOverPolicy attribute value	Description
BiggestAvailable	VCS selects a system based on the forecasted available capacity for all systems in the SystemList. The system with the highest forecasted available capacity is selected. This policy can be set only if the cluster attribute Statistics is set to Enabled. The service group attribute Load is defined in terms of CPU, Memory, and Swap in absolute units. The unit can be of the following values: ■ CPU, MHz, or GHz for CPU ■ GB or MB for Memory ■ GB or MB for Swap See "Cluster attributes" on page 760.

About AdaptiveHA

AdaptiveHA enables VCS to dynamically select the biggest available target system to fail over an application when you set the service group attribute FailOverPolicy to BiggestAvailable. VCS monitors and forecasts the available capacity of systems in terms of CPU, memory, and swap to select the biggest available system.

Enabling AdaptiveHA for a service group

AdaptiveHA enables VCS to make dynamic decisions about failing over an application to the biggest available system.

To enable AdaptiveHA for a service group

1. Ensure cluster attribute Statistics is set to Enabled. You can check by using the following command,

   ```
   # haclus -display|grep Statistics
   ```

   You cannot edit this value at run time.

2. Set the Load of the service group in terms of CPU, Memory, or Swap in absolute units. The unit can be of any following values:

   - CPU, MHz, or GHz for CPU
   - GB or MB for Memory
   - GB or MB for Swap

Define at least one key for the Load attribute.

3 Check the default value of the MeterWeight attribute at the cluster level. If the attribute does not meet the service group's meter weight requirement, then set the MeterWeight at the service group level.

4 Set or edit the values for the following service group attributes in main.cf:

- Modify the values of Load and MeterWeight as decided in the preceding steps.
- Set the value of FailOverPolicy to BiggestAvailable.
  You can set or edit the service group level attributes Load, MeterWeight, and FailOverPolicy during run time.

After you complete the above steps, AdaptiveHA is enabled. The service group follows the BiggestAvailable policy during a failover.

The following table provides information on various attributes and the values they can take to enable AdaptiveHA:

Table 10-3

Use-case	Attribute values to be set
To turn on host metering	Set the cluster attribute Statistics to MeterHostOnly.
To turn off host metering	Set the cluster attribute Statistics to Disabled.
To turn on host metering and forecasting	Set the cluster attribute Statistics to Enabled.
To enable hagrp -forecast CLI option	Set the cluster attribute Statistics to Enabled and also set the service group attribute Load based on your application's CPU, Mem or Swap usage.
To check the meters supported for any host or cluster node	Verify the value of cluster attribute HostAvailableMeters.
To enable host metering, forecast, and policy decisions using forecast	Perform the following actions: ■ Set the cluster attribute Statistics to Enabled ■ Set the service group attribute FailOverPolicy to BiggestAvailable. ■ Set service group attribute Load based on your application's CPU, Mem or Swap usage. ■ Optionally, set the service group attribute MeterWeight.

Table 10-3 *(continued)*

Use-case	Attribute values to be set
To change metering or forecast frequency	Set the MeterInterval and ForecastCycle keys in the cluster attribute MeterControl accordingly.
To check the available capacity and its forecast	Use the following commands to check values for available capacity and its forecast: ■ `# hasys -value system_name AvailableCapacity` ■ `# hasys -value system_name HostAvailableForecast`
To check if the metering and forecast is up-to-date	The metering value is up-to-date when the difference between GlobalCounter and AvailableGC is less than or equal to 24. The forecasting value is up-to-date when the difference between GlobalCounter and AvailableGC is less than or equal to 72.

See "Cluster attributes" on page 760.

See "Service group attributes" on page 719.

See System attributes on page 746.

Considerations for setting FailOverPolicy to BiggestAvailable

Parent or child service group linked with local dependency, and FailOverPolicy set to BiggestAvailable should have the same MeterWeight values. If the MeterWeight is not same for these groups, then the engine refers to the cluster attribute MeterWeight.

Note: The MeterWeight settings enable VCS to decide the target system for a child service group based on the perferred system for the parent service group.

See "Enabling AdaptiveHA for a service group" on page 356.

Limitations on AdaptiveHA

AdaptiveHA enables VCS to make dynamic decisions about failing over an application to the biggest available system. Due to the overcommitting behavior of

the virtualization technologies such as KVM, RHEV, and VMware, a user can create multiple guest system in a host system with system resources (Memory, CPU or SWAP) values higher than the actual values available in the host. Due to this behavior VCS engine considers the guest system with the overcommitted resources as highly available for failing over an application.

For example, on a host machine with in-built physical memory of 4 GB, if you create two guest systems each with 4 GB and 8 GB of memory respectively, VCS engine selects the guest system with 8GB of overcommitted memory value for failing over an application irrespective of the available physical memory in the host.

Manually upgrading the VCS configuration file to the latest version

The VCS configuration file (main.cf) gets automatically upgraded when you upgrade from older versions of VCS using the installer.

If you chose to upgrade VCS manually, ensure that you update the VCS configuration file to include the following changes:

- The Capacity attribute of system and Load attribute of service group are changed from scalar integer to integer-association (multidimensional) type attributes.
  For example, if the System has Capacity attribute defined and service group has Load attribute set in the main.cf file as:

  ```
  Group Gx (
   ....
  Load = { Units = 20 }
  )
  System N1 (
  ....
  Capacity = 30
  )
  ```

  To update the main.cf file, update the Capacity values for Load and System attributes as follows:

  ```
  Group Gx (
   ....
  Load = { Units = 20 }
  )
  System N1 (
  ....
  Capacity = { Units = 30 }
  )
  ```

- If the cluster attribute HostMonLogLvl is defined in the main.cf file, then replace it with Statistics and make the appropriate change from the following:
  - Replace HostMonLogLvl = ALL with Statistics = MeterHostOnly.
  - Replace HostMonLogLvl = AgentOnly with Statistics = MeterHostOnly.

    Note: For HostMonLogLvl value of AgentOnly, no exact equivalent value for Statistics exists. However, when you set Statistics to MeterHostOnly, VCS logs host utilization messages in the engine logs and in the HostMonitor agent logs.

  - Replace HostMonLogLvl = Disabled with Statistics = Disabled.

See "About the main.cf file" on page 64.

About system zones

The SystemZones attribute enables you to create a subset of systems to use in an initial failover decision. This feature allows fine-tuning of application failover decisions, and yet retains the flexibility to fail over anywhere in the cluster.

If the attribute is configured, a service group tries to stay within its zone before choosing a host in another zone. For example, in a three-tier application infrastructure with Web, application, and database servers, you could create two system zones: one each for the application and the database. In the event of a failover, a service group in the application zone will try to fail over to another node within the zone. If no nodes are available in the application zone, the group will fail over to the database zone, based on the configured load and limits.

In this configuration, excess capacity and limits on the database backend are kept in reserve to handle the larger load of a database failover. The application servers handle the load of service groups in the application zone. During a cascading failure, the excess capacity in the cluster is available to all service groups.

About sites

The SiteAware attribute enables you to create sites to use in an initial failover decision in campus clusters. A service group can failover to another site even though a failover target is available within the same site. If the SiteAware attribute is set to 1, you cannot configure SystemZones. You can define site dependencies to restrict dependent applications to fail over within the same site. If the SiteAware attribute is configured and set to 2, then the service group will failover within the same site.

For example, in a campus cluster with two sites, siteA and siteB, you can define a site dependency among service groups in a three-tier application infrastructure. The infrastructure consists of Web, application, and database to restrict the service group failover within same site.

See " How VCS campus clusters work" on page 561.

Load-based autostart

VCS provides a method to determine where a service group comes online when the cluster starts. Setting the AutoStartPolicy to Load instructs the VCS engine, HAD, to determine the best system on which to start the groups. VCS places service groups in an AutoStart queue for load-based startup as soon as the groups probe all running systems. VCS creates a subset of systems that meet all prerequisites and then chooses the system with the highest AvailableCapacity.

Set AutoStartPolicy = Load and configure the SystemZones attribute to establish a list of preferred systems on which to initially run a group.

About freezing service groups

Freezing a service group prevents VCS from taking any action when the service group or a system faults. Freezing a service group prevents dependent resources from going offline when a resource faults. It also prevents the Clean function from being called on a resource fault.

You can freeze a service group when performing operations on its resources from outside VCS control. This prevents VCS from taking actions on resources while your operations are on. For example, freeze a database group when using database controls to stop and start a database.

About controlling Clean behavior on resource faults

The ManageFaults attribute specifies whether VCS calls the Clean function when a resource faults. ManageFaults can be configured at both resource level and service group level. The values of this attribute at the resource level supersede those at the service group level.

You can configure the ManageFaults attribute with the following possible values:

- If the ManageFaults attribute is set to ALL, VCS calls the Clean function when a resource faults.
- If the ManageFaults attribute is set to NONE, VCS takes no action on a resource fault; it "hangs the service group until administrative action can be taken. VCS marks the resource state as ADMIN_WAIT and does not fail over the service group until the resource fault is removed and the ADMIN_WAIT state is cleared.

VCS calls the resadminwait trigger when a resource enters the ADMIN_WAIT state due to a resource fault if the ManageFaults attribute is set to NONE. You can customize this trigger to provide notification about the fault.

When ManageFaults is set to NONE and one of the following events occur, the resource enters the ADMIN_WAIT state:

Table 10-4 lists the possible events and the subsequent state of the resource when the ManageFaults attribute is set to NONE.

Table 10-4 Possible events when the ManageFaults attribute is set to NONE

Event	Resource state
The offline function did not complete within the expected time.	ONLINE|ADMIN_WAIT
The offline function was ineffective.	ONLINE|ADMIN_WAIT
The online function did not complete within the expected time.	OFFLINE|ADMIN_WAIT
The online function was ineffective.	OFFLINE|ADMIN_WAIT
The resource was taken offline unexpectedly.	ONLINE|ADMIN_WAIT
For the online resource the monitor function consistently failed to complete within the expected time.	ONLINE|MONITOR_TIMEDOUT|ADMIN_WAIT

Clearing resources in the ADMIN_WAIT state

When VCS sets a resource in the ADMIN_WAIT state, it invokes the resadminwait trigger according to the reason the resource entered the state.

See “About the resadminwait event trigger” on page 453.

To clear a resource

1. Take the necessary actions outside VCS to bring all resources into the required state.
2. Verify that resources are in the required state by issuing the command:

   ```
   hagrp -clearadminwait group -sys system
   ```

   This command clears the ADMIN_WAIT state for all resources. If VCS continues to detect resources that are not in the required state, it resets the resources to the ADMIN_WAIT state.
3. If resources continue in the ADMIN_WAIT state, repeat step 1 and step 2, or issue the following command to stop VCS from setting the resource to the ADMIN_WAIT state:

   ```
   hagrp -clearadminwait -fault group -sys system
   ```

   This command has the following results:

   - If the resadminwait trigger was called for the reasons 0 or 1, the resource state is set as ONLINE|UNABLE_TO_OFFLINE.
     - 0 = The offline function did not complete within the expected time.
     - 1 = The offline function was ineffective.
   - If the resadminwait trigger was called for reasons 2, 3, or 4, the resource state is set as FAULTED. Note that when resources are set as FAULTED for these reasons, the clean function is not called. Verify that resources in ADMIN-WAIT are in clean, OFFLINE state prior to invoking this command.
     - 2 = The online function did not complete within the expected time.
     - 3 = The online function was ineffective.
     - 4 = The resource was taken offline unexpectedly.

   When a service group has a resource in the ADMIN_WAIT state, the following service group operations cannot be performed on the resource: online, offline, switch, and flush. Also, you cannot use the hastop command when resources are in the ADMIN_WAIT state. When this occurs, you must issue the hastop command with -force option only.

About controlling fault propagation

The FaultPropagation attribute defines whether a resource fault is propagated up the resource dependency tree. It also defines whether a resource fault causes a service group failover.

You can configure the FaultPropagation attribute with the following possible values:

- If the FaultPropagation attribute is set to 1 (default), a resource fault is propagated up the dependency tree. If a resource in the path is critical, the service group is taken offline and failed over, provided the AutoFailOver attribute is set to 1.
- If the FaultPropagation is set to 0, resource faults are contained at the resource level. VCS does not take the dependency tree offline, thus preventing failover. If the resources in the service group remain online, the service group remains in the PARTIAL|FAULTED state. If all resources are offline or faulted, the service group remains in the OFFLINE| FAULTED state.

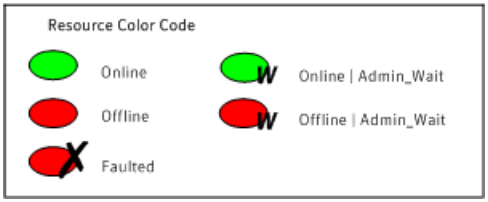
When a resource faults, VCS fires the resfault trigger and sends an SNMP trap. The trigger is called on the system where the resource faulted and includes the name of the faulted resource.

Customized behavior diagrams

This topic depicts how the ManageFaults and FaultPropagation attributes change VCS behavior when handling resource faults.

Figure 10-6 depicts the legends or resource color code.

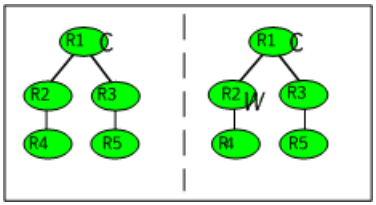
Figure 10-6 Legends and resource color code



Example scenario: Resource with a critical parent and ManageFaults=NONE

Figure 10-7 shows an example of a service group that has five resources. The ManageFaults attribute for the group of resource R2 is set to NONE.

Figure 10-7 Scenario: Resource with a critical parent and ManageFaults=NONE

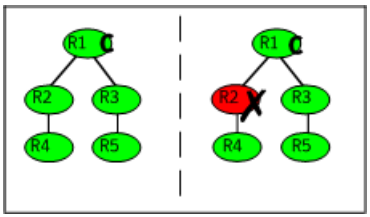


If resource R2 fails, the resource is marked as ONLINE|ADMIN_WAIT. The Clean function is not called for the resource. VCS does not take any other resource offline.

Example scenario: Resource with a critical parent and FaultPropagation=0

Figure 10-8 ahows an example where the FaultPropagation attribute is set to 0.

Figure 10-8 Scenario: Resource with a critical parent and FaultPropagation=0



When resource R2 faults, the Clean function is called and the resource is marked as faulted. The fault is not propagated up the tree, and the group is not taken offline.

About preventing concurrency violation

If a failover service group comes online on more than one node, the chances of data corruption increase. You can configure failover service groups that contain application resources for prevention of concurrency violation (PCV) by setting the group's ProPCV attribute to 1.

Note: You cannot set the ProPCV attribute for parallel service groups and for hybrid service groups.

You can set the ProPCV attribute when the service group is inactive on all the nodes or when the group is active (ONLINE, PARTIAL, or STARTING) on one node in the

cluster. You cannot set the ProPCV attribute if the service group is already online on multiple nodes in the cluster. See "Service group attributes" on page 719.

If ProPCV is set to 1, you cannot bring online processes that are listed in the MonitorProcesses attribute or the StartProgram attribute of the application resource on any other node in the cluster. If you try to start a process that is listed in the MonitorProcesses attribute or StartProgram attribute on any other node, that process is killed before it starts. Therefore, the service group does not get into concurrency violation.

Enabling or preventing resources to start outside VCS control

When a resource is brought online on one node in a cluster, the resource must not be allowed to come online outside VCS control on any other node in the cluster. By ensuring that the resource cannot be online on more than one node, you can prevent data corruption and ensure high availability through VCS.

The ProPCV attribute of the service group containing application resource determines whether or not to allow the processes for the application resource to start outside VCS control. The application type resource must be registered with IMF for offline monitoring or online monitoring. ProPCV applies only to the processes that are specified in the MonitorProcesses attribute or the StartProgram attribute of the application type resource. See the *Cluster Server Bundled Agents Reference Guide* for information about the propcv action agent function and also for information on when the application resource can be registered with IMF for offline monitoring.

Note: Currently, ProPCV works for application type resources only.

In situations where the propcv action agent function times out, you can use the amfregister command to manually mark a resource as one of the following:

- A resource that is allowed to be brought online outside VCS control.
- A resource that is prevented from being brought online outside VCS control. Such a ProPCV-enabled resource cannot be online on more than one node in the cluster.

To allow a resource to be started outside VCS control

◆ Type the following command:

```
amfregister -r reapername -g resourcename -P a
```

Example: `amfregister -r VCSApplicationAgent -g app1 -P a`

The application resource app1 is allowed to start if you invoke it outside VCS control or from the command line.

To prevent a resource from being started outside VCS control

◆ Type the following command:

```
amfregister -r reapername -g resourcename -P p
```

Example: `amfregister -r VCSApplicationAgent -g app1 -P p`

The application resource app1 is prevented from starting if you invoke it outside VCS control or from the command line.

In the preceding examples,

- *reapername* is the agent whose name is displayed under the Registered Reapers section of `amfstat` output. For application resources, reapername is VCSApplicationAgent.
- Option `r` indicates the name of the reaper or agent as displayed under the Registered Reapers section of the `amfstat` command's output. For application resources, reaper name is VCSApplicationAgent.
- *resourcename* is the resource name.
- Option `g` indicates the name of the resource. In the preceding example, the application resource is app1.
- Option `P` indicates whether to allow or prevent a resource from starting up outside VCS control.
  - Argument `a` denotes that the resource can be started outside VCS control.
  - Argument `p` denotes that the resource is prevented from starting outside VCS control.

Limitations of ProPCV

The following limitations apply:

- ProPCV feature is supported only when the Mode value for the IMF attribute of the Application type resource is set to 3 on all nodes in the cluster.
- The ProPCV feature does not protect against concurrency in the following cases:

  - When you modify the IMFRegList attribute for the resource type.
  - When you modify any value that is part of the IMFRegList attribute for the resource type.

- If you configure the application type resource for ProPCV, consider the following:
  - If you run the process with changed order of arguments, the ProPCV feature does not prevent the execution of the process.
    For example, a single command can be run in multiple ways:

    ```
    /usr/bin/tar -c -f a.tar
    ```

    ```
    /usr/bin/tar -f a.tar -c
    ```

    The ProPCV feature works only if you run the process the same way as it is configured in the resource configuration.
  - If there are multiple ways or commands to start a process, ProPCV prevents the startup of the process only if the process is started in the way specified in the resource configuration.

- You can bring processes online outside VCS control on another node when a failover service group is auto-disabled.

  Examples are:

  - When you use the `hastop -local` command or the `hastop -local -force` command on a node.
  - When a node is detected as FAULTED after its ShutdownTimeout value has elapsed because HAD exited.

  In such situations, you can bring processes online outside VCS control on a node even if the failover service group is online on another node on which VCS engine is not running.

- Before you set ProPCV to 1 for a service group, you must ensure that none of the processes specified in the MonitorProcesses attribute or the StartProgram attribute of the application resource of the group are running on any node where the resource is offline. If an application resource lists two processes in its MonitorProcesses attribute, both processes need to be offline on all nodes in the cluster. If a node has only one process running and you set ProPCV to 1 for the group, you can still start the second process on another node because the Application agent cannot perform selective offline monitoring or online monitoring of individual processes for an application resource

- If a ProPCV-enabled service group has some application resources and some non-application type resources (that cannot be configured for ProPCV), the group can still get into concurrency violation for the non-application type

resources. You can bring the non-application type resources online outside VCS control on a node when the service group is active on another node. In such cases, the concurrency violation trigger is invoked.

- When ProPCV is enabled for a group, the AMF driver prevents certain processes from starting based on the process offline registrations with the AMF driver. If a process starts whose pathname and arguments match with the registered event, and if the prevent action is set for this registered event, that process is prevented from starting. Apart from that, if the arguments match, and even if only the basename of the starting process matches with the basename of the pathname of the registered event, AMF driver prevents that process from starting
- Even with ProPCV enabled, the AMF driver can prevent only those processes from starting whose pathname and arguments match with the events registered with the AMF driver. If the same process is started in some other manner (for example, with a totally different pathname), AMF driver does not prevent the process from starting. This behavior is in line with how AMF driver works for process offline monitoring.

VCS behavior for resources that support the intentional offline functionality

Certain agents can identify when an application has been intentionally shut down outside of VCS control.

For agents that support this functionality, if an administrator intentionally shuts down an application outside of VCS control, VCS does not treat it as a fault. VCS sets the service group state as offline or partial, depending on the state of other resources in the service group.

This feature allows administrators to stop applications without causing a failover. The feature is available for V51 agents.

About the IntentionalOffline attribute

To configure a resource to recognize an intentional offline of configured application, set the IntentionalOffline attribute to 1. Set the attribute to its default value of 0 to disable this functionality. IntentionalOffline is Type level attribute and not a resource level attribute.

You can configure the IntentionalOffline attribute with the following possible values:

- If you set the attribute to 1: When the application is intentionally stopped outside of VCS control, the resource enters an OFFLINE state. This attribute does not affect VCS behavior on application failure. VCS continues to fault resources if managed corresponding applications fail.

- If you set the attribute to 0: When the application is intentionally stopped outside of VCS control, the resource enters a FAULTED state.

About the ExternalStateChange attribute

Use the ExternalStateChange attribute to control service group behavior when a configured application is intentionally started or stopped outside of VCS control.

The attribute defines how VCS handles service group state when resources are intentionally brought online or taken offline outside of VCS control.

You can configure the ExternalStateChange attribute with the values listed in Table 10-5.

Table 10-5 ExternalStateChange attribute values

Attribute value	Service group behavior
OnlineGroup	If the configured application is started outside of VCS control, VCS brings the corresponding service group online. If you attempt to start the application on a frozen node or service group, VCS brings the corresponding service group online once the node or the service group is unfrozen.
OfflineGroup	If the configured application is stopped outside of VCS control, VCS takes the corresponding service group offline.
OfflineHold	If a configured application is stopped outside of VCS control, the agent sets the state of the corresponding VCS resource as offline. In this case, the agents does not take any parent resource or the service group offline. **Note:** The agent sets offline state only for the resources that support detection of an intentional offline, and are registered as V51 or later agents.

OfflineHold and OfflineGroup are mutually exclusive.

VCS behavior when a service group is restarted

The VCS engine behavior is determined by certain service group-level attributes when service groups are restarted.

VCS behavior when non-persistent resources restart

The OnlineRetryLimit and OnlineRetryInterval attributes determine the VCS behavior when the VCS engine attempts to bring a faulted non-persistent resource online.

The OnlineRetryLimit attribute allows a service group to be brought online again on the same system if a non-persistent resource in the service group faults. If, for some reason, the service group cannot be restarted, the VCS engine repeatedly tries to bring the service group online till the number of attempts that are specified by OnlineRetryLimit expires.

However, if the OnlineRetryInterval attribute is set to a non-zero value, the service group that has been successfully restarted faults again; it should be failed over. The service group should not be retried on the same system, even if the attribute OnlineRetryLimit is non-zero. This prevents a group from continuously faulting and restarting on the same system. The interval is measured in seconds.

VCS behavior when persistent resources transition from faulted to online

The AutoRestart attribute determines the VCS behavior in the following scenarios:

- A service group cannot be automatically started because of a faulted persistent resource
- A service group is unable to failover

Later, when a persistent resource transitions from FAULTED to ONLINE, the VCS engine attempts to bring the service group online if the AutoRestart attribute is set to 1 or 2.

If AutoRestart is set to 1, the VCS engine restarts the service group. If AutoRestart is set to 2, the VCS engine clears the faults on all faulted non-persistent resources in the service group before it restarts the service group on the same system

About controlling VCS behavior at the resource level

You can control VCS behavior at the resource level. Note that a resource is not considered faulted until the agent framework declares the fault to the VCS engine.

Certain attributes affect how the VCS agent framework reacts to problems with individual resources before informing the fault to the VCS engine.

Resource type attributes that control resource behavior

The following attributes affect how the VCS agent framework reacts to problems with individual resources before informing the fault to the VCS engine.

About the RestartLimit attribute

The RestartLimit attribute defines whether VCS attempts to restart a failed resource before it informs the engine of the fault.

If the RestartLimit attribute is set to a non-zero value, the agent attempts to restart the resource before it declares the resource as Faulted. When the agent framework restarts a failed resource, it first calls the Clean function and then the Online function. However, if the ManageFaults attribute is set to NONE, the agent does not call the Clean function and does not retry the Online function. If you configure the ManageFaults attribute at the resource level, it supersedes the values that are configured at the service group level.

About the OnlineRetryLimit attribute

The OnlineRetryLimit attribute specifies the number of times the Online function is retried if the initial attempt to bring a resource online is unsuccessful.

When OnlineRetryLimit is set to a non-zero value, the agent first calls the Clean function and then reruns the Online function. However, if the ManageFaults attribute is set to NONE, the agent does not call the Clean function and does not retry the Online function. If you configure the ManageFaults attribute at the resource level, it supersedes the values that are configured at the service group level.

About the ConfInterval attribute

The ConfInterval attribute defines how long a resource must remain online without encountering problems before previous problem counters are cleared. The attribute controls when VCS clears the RestartCount, ToleranceCount and CurrentMonitorTimeoutCount values.

About the ToleranceLimit attribute

The ToleranceLimit attribute defines the number of times the monitor routine should return an offline status before declaring a resource offline. This attribute is typically used when a resource is busy and appears to be offline. Setting the attribute to a non-zero value instructs VCS to allow multiple failing monitor cycles with the expectation that the resource will eventually respond. Setting a non-zero ToleranceLimit also extends the time required to respond to an actual fault.

About the FaultOnMonitorTimeouts attribute

The FaultOnMonitorTimeouts attribute defines whether VCS interprets a Monitor function timeout as a resource fault.

If the attribute is set to 0, VCS does not treat Monitor timeouts as a resource faults. If the attribute is set to 1, VCS interprets the timeout as a resource fault and the agent calls the Clean function to shut the resource down.

By default, the FaultOnMonitorTimeouts attribute is set to 4. This means that the Monitor function must time out four times in a row before the resource is marked faulted. The first monitor time out timer and the counter of time outs are reset after one hour of the first monitor time out.

How VCS handles resource faults

This section describes the process VCS uses to determine the course of action when a resource faults.

VCS behavior when an online resource faults

In the following example, a resource in an online state is reported as being offline without being commanded by the agent to go offline.

VCS goes through the following steps when an online resource faults:

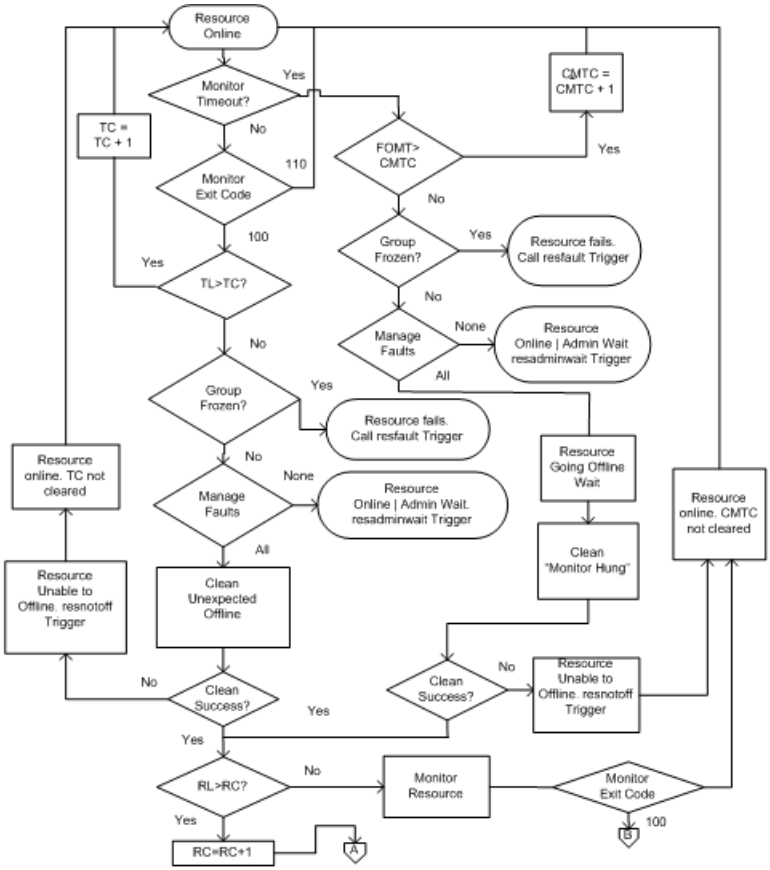
- VCS first verifies the Monitor routine completes successfully in the required time. If it does, VCS examines the exit code returned by the Monitor routine. If the Monitor routine does not complete in the required time, VCS looks at the FaultOnMonitorTimeouts (FOMT) attribute.
- If FOMT=0, the resource will not fault when the Monitor routine times out. VCS considers the resource online and monitors the resource periodically, depending on the monitor interval.
  If FOMT=1 or more, VCS compares the CurrentMonitorTimeoutCount (CMTC) with the FOMT value. If the monitor timeout count is not used up, CMTC is incremented and VCS monitors the resource in the next cycle.
- If FOMT= CMTC, this means that the available monitor timeout count is exhausted and VCS must now take corrective action. VCS checks the Frozen attribute for the service group. If the service group is frozen, VCS declares the resource faulted and calls the resfault trigger. No further action is taken.
- If the service group is not frozen, VCS checks the ManageFaults attribute at the resource level. VCS marks the resource as ONLINE|ADMIN_WAIT and fires the resadminwait trigger if its group-level value is NONE or if its resource-level value is IGNORE. If ManageFaults is set to ACT at the resource level or ALL at the group level, VCS invokes the Clean function with the reason Monitor Hung.

Note: The resource-level ManageFaults value supersedes the corresponding service group-level value. VCS honors the service group-level ManageFaults value only when the corresponding resource-level value is blank ("").

- If the Clean function is successful (that is, Clean exit code = 0), VCS examines the value of the RestartLimit attribute. If Clean fails (exit code = 1), the resource remains online with the state UNABLE TO OFFLINE. VCS fires the resnotoff trigger and monitors the resource again.
- If the Monitor routine does not time out, it returns the status of the resource as being online or offline.
- If the ToleranceLimit (TL) attribute is set to a non-zero value, the Monitor cycle returns offline (exit code = 100) for a number of times specified by the ToleranceLimit and increments the ToleranceCount (TC). When the ToleranceCount equals the ToleranceLimit (TC = TL), the agent declares the resource as faulted.
- If the Monitor routine returns online (exit code = 110) during a monitor cycle, the agent takes no further action. The ToleranceCount attribute is reset to 0 when the resource is online for a period of time specified by the ConfInterval attribute.

  If the resource is detected as being offline a number of times specified by the ToleranceLimit before the ToleranceCount is reset (TC = TL), the resource is considered faulted.
- After the agent determines the resource is not online, VCS checks the Frozen attribute for the service group. If the service group is frozen, VCS declares the resource faulted and calls the resfault trigger. No further action is taken.
- If the service group is not frozen, VCS checks the ManageFaults attribute. If ManageFaults=NONE, VCS marks the resource state as ONLINE|ADMIN_WAIT and calls the resadminwait trigger. If ManageFaults=ALL, VCS calls the Clean function with the CleanReason set to Unexpected Offline.
- If the Clean function fails (exit code = 1) the resource remains online with the state UNABLE TO OFFLINE. VCS fires the resnotoff trigger and monitors the resource again. The resource enters a cycle of alternating Monitor and Clean functions until the Clean function succeeds or a user intervenes.
- If the Clean function is successful, VCS examines the value of the RestartLimit (RL) attribute. If the attribute is set to a non-zero value, VCS increments the RestartCount (RC) attribute and invokes the Online function. This continues till the value of the RestartLimit equals that of the RestartCount. At this point, VCS attempts to monitor the resource.

- If the Monitor returns an online status, VCS considers the resource online and resumes periodic monitoring. If the monitor returns an offline status, the resource is faulted and VCS takes actions based on the service group configuration.



VCS behavior when a resource fails to come online

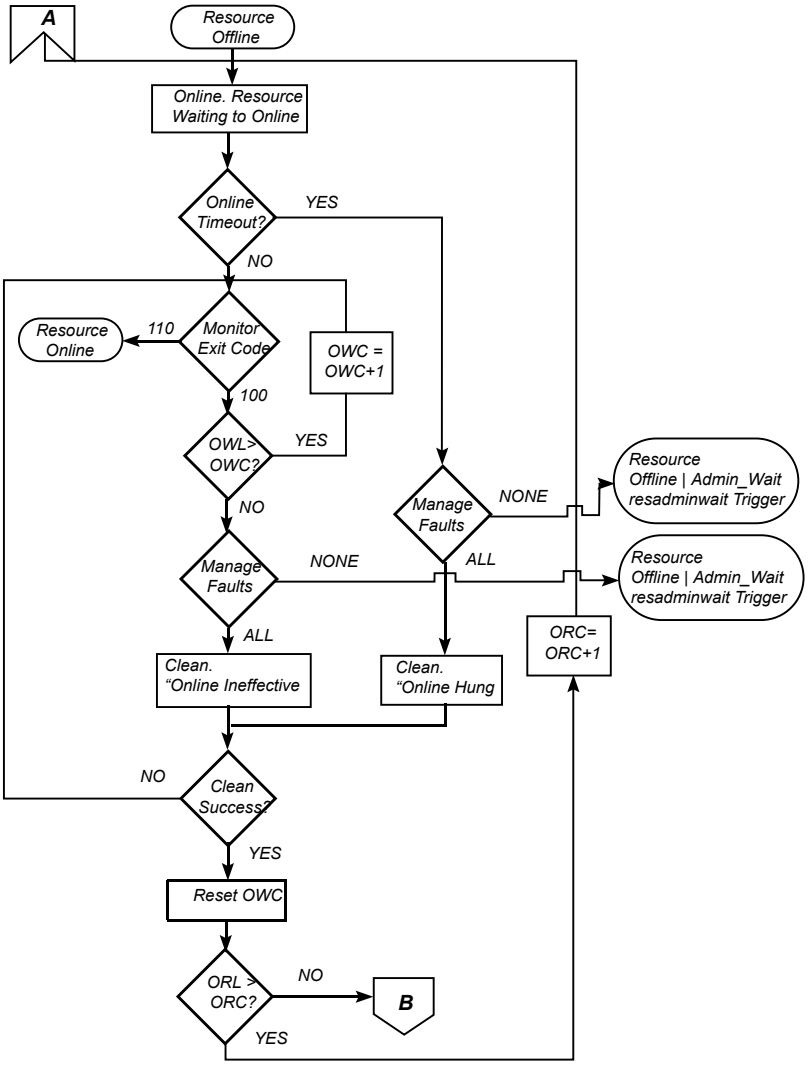
In the following example, the agent framework invokes the Online function for an offline resource. The resource state changes to WAITING TO ONLINE.

VCS goes through the following steps when a resource fails to come online:

- If the Online function times out, VCS examines the value of the ManageFaults attribute first at the resource level and then at the service group level. If ManageFaults is defined at the resource level, VCS overrides the corresponding

values that are specified at the service group level. VCS takes action based on the ManageFaults values that are specified at the resource level.

- If ManageFaults is set to IGNORE at the resource level, the resource state changes to OFFLINE|ADMIN_WAIT. The resource-level value overrides the service group-level value.
- If ManageFaults is set to ACT at the resource level, VCS calls the Clean function with the CleanReason set to Online Hung.
- If resource-level ManageFaults is set to "" or blank, VCS checks the corresponding service group-level value, and proceeds as follows:
  - If ManageFaults is set to NONE, the resource state changes to OFFLINE|ADMIN_WAIT.
  - If ManageFaults is set to ALL, VCS calls the Clean function with the CleanReason set to Online Hung.
- If ManageFaults is set to NONE, the resource state changes to OFFLINE|ADMIN_WAIT.
  If ManageFaults is set to ALL, VCS calls the Clean function with the CleanReason set to Online Hung.
- If the Online function does not time out, VCS invokes the Monitor function. The Monitor routine returns an exit code of 110 if the resource is online. Otherwise, the Monitor routine returns an exit code of 100.
- VCS examines the value of the OnlineWaitLimit (OWL) attribute. This attribute defines how many monitor cycles can return an offline status before the agent framework declares the resource faulted. Each successive Monitor cycle increments the OnlineWaitCount (OWC) attribute. When OWL= OWC (or if OWL= 0), VCS determines the resource has faulted.
- VCS then examines the value of the ManageFaults attribute. If the ManageFaults is set to NONE, the resource state changes to OFFLINE|ADMIN_WAIT.
  If the ManageFaults is set to ALL, VCS calls the Clean function with the CleanReason set to Online Ineffective.
- If the Clean function is not successful (exit code = 1), the agent monitors the resource. It determines the resource is offline, and calls the Clean function with the Clean Reason set to Online Ineffective. This cycle continues till the Clean function is successful, after which VCS resets the OnlineWaitCount value.
- If the OnlineRetryLimit (ORL) is set to a non-zero value, VCS increments the OnlineRetryCount (ORC) and invokes the Online function. This starts the cycle all over again. If ORL = ORC, or if ORL = 0, VCS assumes that the Online operation has failed and declares the resource as faulted.



VCS behavior after a resource is declared faulted

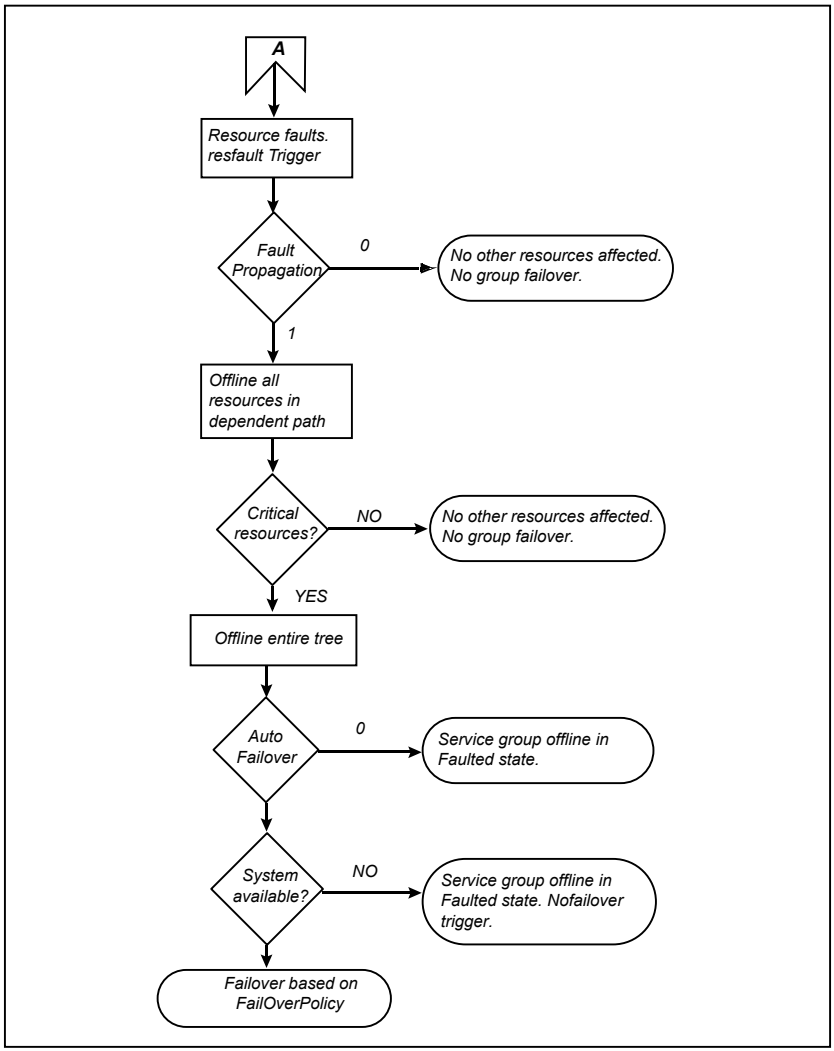
After a resource is declared faulted, VCS fires the resfault trigger and examines the value of the FaultPropagation attribute.

VCS goes through the following steps after a resource is declared faulted:

- If FaultPropagation is set to 0, VCS does not take other resources offline, and changes the group state to OFFLINE|FAULTED or PARTIAL|FAULTED. The service group does not fail over.

If FaultPropagation is set to 1, VCS takes all resources in the dependent path of the faulted resource offline, up to the top of the tree.

- VCS then examines if any resource in the dependent path is critical. If no resources are critical, the service group is left in its OFFLINE|FAULTED or PARTIAL|FAULTED state. If a resource in the path is critical, VCS takes the all resources in the service group offline in preparation of a failover.
- If the AutoFailOver attribute is set to 0, the service group is not failed over; it remains in a faulted state. If AutoFailOver is set to 1, VCS examines if any systems in the service group's SystemList are possible candidates for failover. If no suitable systems exist, the group remains faulted and VCS calls the nofailover trigger. If eligible systems are available, VCS examines the FailOverPolicy to determine the most suitable system to which to fail over the service group.
- If FailOverPolicy is set to Load, a NoFailover situation may occur because of restrictions placed on service groups and systems by Service Group Workload Management.



VCS behavior when a resource is restarted

The behavior of VCS agents is determined by certain resource-level attributes when resources are restarted.

VCS behavior when online agent function fails to bring a resource online

You can use the OnlineRetryLimit attribute to specify the number of retries that the agent should perform to bring the resource online.

VCS behavior when a resource goes offline

The RestartLimit attribute can be used to specify the number of retries that an agent should perform to bring a resource online before the VCS engine declares the resource as FAULTED.

The ToleranceLimit attribute defines the number of times the monitor agent function can return unexpected OFFLINE before it declares the resource as FAULTED. A large ToleranceLimit value delays detection of a genuinely faulted resource.

For example, assume that RestartLimit is set to 2 and ToleranceLimit is set to 3 for a resource and that the resource state is ONLINE. If the next monitor detects the resource state as OFFLINE, the agent waits for another monitor cycle instead of triggering a restart. The agent waits a maximum of 3 (ToleranceLimit) monitor cycles before it triggers a restart.

The RestartLimit and ToleranceLimit attributes determine the VCS behavior in the following scenarios:

- The RestartLimit attribute is considered when the resource state is ONLINE and next monitor cycle detects it as OFFLINE and ToleranceLimit is reached.
- The OnlineRetryLimit attribute is considered when the resource is to be brought online.

About disabling resources

Disabling a resource means that the resource is no longer monitored by a VCS agent, and that the resource cannot be brought online or taken offline. The agent starts monitoring the resource after the resource is enabled. The resource attribute Enabled determines whether a resource is enabled or disabled. A persistent resource can be disabled when all its parents are offline. A non-persistent resource can be disabled when the resource is in an OFFLINE state.

When to disable a resource

Typically, resources are disabled when one or more resources in the service group encounter problems and disabling the resource is required to keep the service group online or to bring it online.

Note: Disabling a resource is not an option when the entire service group requires disabling. In that case, set the service group attribute Enabled to 0.

Use the following command to disable the resource when VCS is running:

```
# hares -modify resource_name Enabled 0
```

To have the resource disabled initially when VCS is started, set the resource's Enabled attribute to 0 in main.cf.

Limitations of disabling resources

When VCS is running, there are certain prerequisites to be met before the resource is disabled successfully.

- An online non-persistent resource cannot be disabled. It must be in a clean OFFLINE state. (The state must be OFFLINE and IState must be NOT WAITING.)
- If it is a persistent resource and the state is ONLINE on some of the systems, all dependent resources (parents) must be in clean OFFLINE state. (The state must be OFFLINE and IState must be NOT WAITING)

Therefore, before disabling the resource you may be required to take it offline (if it is non-persistent) and take other resources offline in the service group.

Additional considerations for disabling resources

Following are the additional considerations for disabling resources:

- When a group containing disabled resources is brought online, the online transaction is not propagated to the disabled resources. Children of the disabled resource are brought online by VCS only if they are required by another enabled resource.
- You can bring children of disabled resources online if necessary.
- When a group containing disabled resources is taken offline, the offline transaction is propagated to the disabled resources.
- If a service group is part of an `hagrp -online -propagate` operation or an `hagrp -offline -propagate` operation and a resource in the service group is disabled, the resource might not complete its online operation or offline operation. In this case, PolicyIntention of the service group is set to 0.
  In an `hagrp online -propagate` operation, if the child service groups cannot be brought online, the parent service groups also cannot be brought online. Therefore, when the PolicyIntention value of 1 for the child service group is cleared, the PolicyIntention value of all its parent service groups in dependency

tree is also cleared. When the PolicyIntention value of 2 for the parent service group is cleared, the PolicyIntention value of all its child service groups in dependency tree is also cleared.

This section shows how a service group containing disabled resources is brought online.

Figure 10-9 shows Resource_3 is disabled. When the service group is brought online, the only resources brought online by VCS are Resource_1 and Resource_2 (Resource_2 is brought online first) because VCS recognizes Resource_3 is disabled. In accordance with online logic, the transaction is not propagated to the disabled resource.

Figure 10-9 Scenario: Transaction not propagated to the disabled resource (Resource_3)

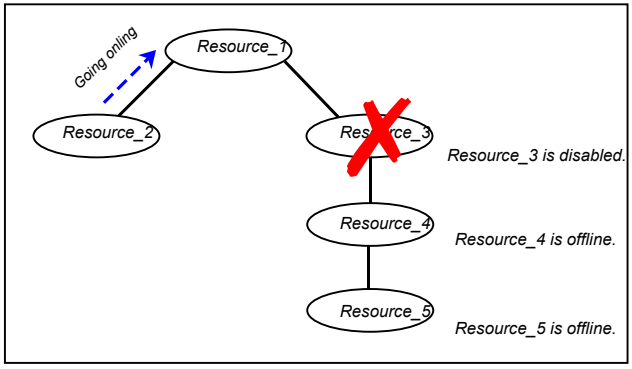
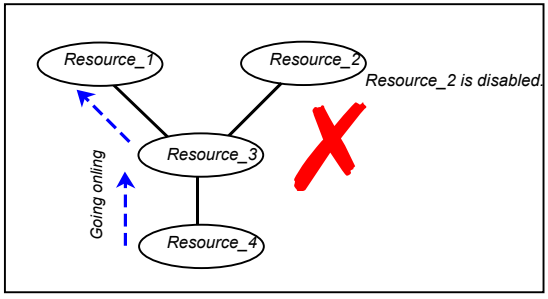


Figure 10-10, shows that Resource_2 is disabled. When the service group is brought online, resources 1, 3, 4 are also brought online (Resource_4 is brought online first). Note Resource_3, the child of the disabled resource, is brought online because Resource_1 is enabled and is dependent on it.

Figure 10-10 Scenario: Child of the disabled resource (Resource_3) is brought online



How disabled resources affect group states

When a service group is brought online containing non-persistent, disabled resources whose AutoStart attributes are set to 1, the group state is PARTIAL, even though enabled resources with Autostart=1 are online. This is because the disabled resource is considered for the group state.

To have the group in the ONLINE state when enabled resources with AutoStart set to 1 are in ONLINE state, set the AutoStart attribute to 0 for the disabled, non-persistent resources.

Changing agent file paths and binaries

By default, VCS runs agent binaries from the path $VCS_HOME/bin/*AgentName*/*AgentNameAgent*. For example, /opt/VRTSvcs/bin/FileOnOff/FileOnOffAgent.

You can instruct VCS to run a different set of agent binaries or scripts by specifying values for the following attributes.

- AgentFile:
  Specify a value for this attribute if the name of the agent binary is not the same as that of the resource type.
  For example, if the resource type is MyApplication and the agent binary is called MyApp, set the AgentFile attribute to MyApp. For a script-base agent, you could configure AgentFile as /opt/VRTSvcs/bin/ScriptAgent.
- AgentDirectory:
  Specify a value for this attribute if the agent is not installed at the default location.
  When you specify the agent directory, VCS looks for the agent file (*AgentNameAgent*) in the agent directory. If the agent file name does not conform to the *AgentNameAgent* convention, configure the AgentFile attribute.
  For example, if the MyApplication agent is installed at /opt/VRTSvcs/bin/CustomAgents/MyApplication, specify this path as the attribute value. If the agent file is not named MyApplicationAgent, configure the AgentFile attribute.
  If you do not set these attributes and the agent is not available at its default location, VCS looks for the agent at the /opt/VRTSagents/ha/bin/*AgentName*/*AgentNameAgent*.

To change the path of an agent

- Before configuring a resource for the agent, set the AgentFile and AgentDirectory attributes of the agent's resource type.

```
hatype -modify resource_type AgentFile \
       "binary_name"
hatype -modify resource_type AgentDirectory \
       "complete_path_to_agent_binary"
```

VCS behavior on loss of storage connectivity

When a node loses connectivity to shared storage, input-output operations (I/O) to volumes return errors and the disk group gets disabled. In this situation, VCS must fail the service groups over to another node. This failover is to ensure that applications have access to shared storage. The failover involves deporting disk groups from one node and importing them to another node. However, pending I/Os must complete before the disabled disk group can be deported.

Pending I/Os cannot complete without storage connectivity. When VCS is not configured with I/O fencing and the PanicSystemOnDGLoss attribute of DiskGroup is not configured to panic the system, VCS assumes data is being read from or written to disks and does not declare the DiskGroup resource as offline. This behavior prevents potential data corruption that may be caused by the disk group being imported on two hosts. However, this also means that service groups remain online on a node that does not have storage connectivity and the service groups cannot be failed over unless an administrator intervenes. This affects application availability.

Some Fibre Channel (FC) drivers have a configurable parameter called failover, which defines the number of seconds for which the driver retries I/O commands before returning an error. If you set the failover parameter to 0, the FC driver retries I/O infinitely and does not return an error even when storage connectivity is lost. This also causes the Monitor function for the DiskGroup to time out and prevents failover of the service group unless an administrator intervenes.

Disk group configuration and VCS behavior

Table 10-6 describes how the disk group state and the failover attribute define VCS behavior when a node loses connectivity to shared storage.

Table 10-6 Disk group state and the failover attribute define VCS behavior

Case	DiskGroup state	Failover attribute	VCS behavior on loss of storage connectivity
1	Enabled	N seconds	Fails over service groups to another node.
2	Disabled	N seconds	DiskGroup resource remains online. No failover.
3	Enabled	0	DiskGroup resource remains in monitor timed out state. No failover.
4	Disabled	0	DiskGroup resource remains online. No failover.

How VCS attributes control behavior on loss of storage connectivity

You can configure VCS attributes to ensure that a node panics on losing connectivity to shared storage. The panic causes service groups to fail over to another node.

A system reboot or shutdown could leave the system in a hung state because the operating system cannot dump the buffer cache to the disk. The panic operation ensures that VCS does not wait for I/Os to complete before triggering the failover mechanism, thereby ensuring application availability. However, you might have to perform a file system check when you restart the node.

The following attributes of the Diskgroup agent define VCS behavior on loss of storage connectivity:—

PanicSystemOnDGLoss — Defines whether the agent panics the system when storage connectivity is lost or Monitor times out.

FaultOnMonitorTimeouts

Specifies the number of consecutive monitor timeouts after which VCS calls the Clean function to mark the resource as FAULTED or restarts the resource.

If you set the attribute to 0, VCS does not treat a Monitor timeout as a resource fault. By default, the attribute is set to 4. This means that the Monitor function must time out four times in a row before VCS marks the resource faulted.

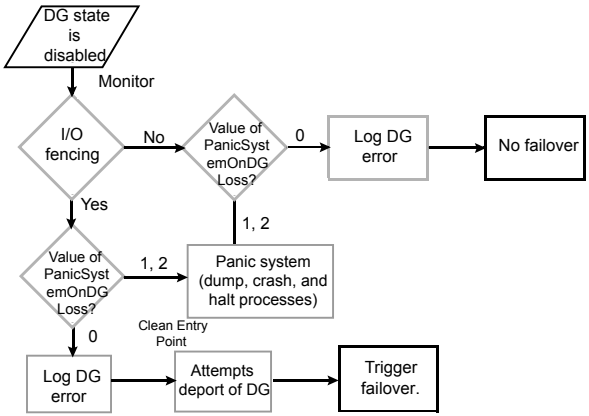
When the Monitor function for the DiskGroup agent times out, (case 3 in the preceding table), the FaultOnMonitorTimeouts attribute defines when VCS interprets the resource as faulted and invokes the Clean function. If the CleanReason is "monitor hung", the system panics.

For details about these attributes, see the *Cluster Server Bundled Agents Reference Guide*.

VCS behavior when a disk group is disabled

Figure 10-11 describes VCS behavior for a disabled diskgroup.

Figure 10-11 VCS behavior when a disk group is disabled



Recommendations to ensure application availability

Veritas makes the following recommendations to ensure application availability and data integrity when a node loses connectivity to shared storage:

- Do not set the failover attribute for the FC driver to 0.
  However, if you do set the failover attribute to 0, set the FaultOnMonitorTimeouts value for the DiskGroup resource type to a finite value.
- If you use I/O fencing, set the PanicSystemOnDGLoss attribute for the DiskGroup resource to appropriate value as per requirement. This ensures that the system panics when it loses connectivity to shared storage or monitor program timeout for FaultOnMonitorTimeouts value, and causes applications to fail over to another node. The failover ensures application availability, while I/O fencing ensures data integrity.

Service group workload management

Workload management is a load-balancing mechanism that determines which system hosts an application during startup, or after an application or server fault.

Service Group Workload Management provides tools for making intelligent decisions about startup and failover locations, based on system capacity and resource availability.

About enabling service group workload management

The service group attribute FailOverPolicy governs how VCS calculates the target system for failover. Set FailOverPolicy to Load to enable service group workload management.

See " About controlling VCS behavior at the resource level" on page 371.

System capacity and service group load

The Load and Capacity construct allows the administrator to define a fixed amount of resources a server provides (Capacity), and a fixed amount of resources a specific service group is expected to utilize (Load).

The system attribute Capacity sets a fixed load-handling capacity for servers. Define this attribute based on system requirements.

The service group attribute Load sets a fixed demand for service groups. Define this attribute based on application requirements.

When a service group is brought online, its load is subtracted from the system's capacity to determine available capacity. VCS maintains this info in the attribute AvailableCapacity.

When a failover occurs, VCS determines which system has the highest available capacity and starts the service group on that system. During a failover involving multiple service groups, VCS makes failover decisions serially to facilitate a proper load-based choice.

System capacity is a soft restriction; in some situations, value of the Capacity attribute could be less than zero. During some operations, including cascading failures, the value of the AvailableCapacity attribute could be negative.

Static load versus dynamic load

Dynamic load is an integral component of the Service Group Workload Management framework. Typically, HAD sets remaining capacity with the function:

AvailableCapacity = Capacity - (sum of Load values of all online service groups)

If the DynamicLoad attribute is defined, its value overrides the calculated Load values with the function:

AvailableCapacity = Capacity - DynamicLoad

This enables better control of system loading values than estimated service group loading (static load). However, this requires setting up and maintaining a load estimation package outside VCS. It also requires modifying the configuration file main.cf manually.

Note that the DynamicLoad (specified with hasys -load) is subtracted from the Capacity as an integer and not a percentage value. For example, if a system's capacity is 200 and the load estimation package determines the server is 80 percent loaded, it must inform VCS that the DynamicLoad value is 160 (not 80).

About overload warning

Overload warning provides the notification component of the Load policy. When a server sustains the preset load level (set by the attribute LoadWarningLevel) for a preset time (set by the attribute LoadTimeThreshold), VCS invokes the loadwarning trigger.

See "Using event triggers" on page 448.

See System attributes on page 746.

The loadwarning trigger is a user-defined script or application designed to carry out specific actions. It is invoked once, when system load exceeds the LoadWarningLevel for the LoadTimeThreshold. It is not invoked again until the

LoadTimeCounter, which determines how many seconds system load has been above LoadWarningLevel, is reset.

System limits and service group prerequisites

Limits is a system attribute and designates which resources are available on a system, including shared memory segments and semaphores.

Prerequisites is a service group attribute and helps manage application requirements. For example, a database may require three shared memory segments and 10 semaphores. VCS Load policy determines which systems meet the application criteria and then selects the least-loaded system.

If the prerequisites defined for a service group are not met on a system, the service group cannot be brought online on the system.

When configuring these attributes, define the service group's prerequisites first, then the corresponding system limits. Each system can have a different limit and there is no cap on the number of group prerequisites and system limits. Service group prerequisites and system limits can appear in any order.

You can also use these attributes to configure the cluster as N-to-1 or N-to-N. For example, to ensure that only one service group can be online on a system at a time, add the following entries to the definition of each group and system:

```
Prerequisites = { GroupWeight = 1 }
Limits = { GroupWeight = 1 }
```

System limits and group prerequisites work independently of FailOverPolicy. Prerequisites determine the eligible systems on which a service group can be started. When a list of systems is created, HAD then follows the configured FailOverPolicy.

About capacity and limits

When selecting a node as a failover target, VCS selects the system that meets the service group's prerequisites and has the highest available capacity. If multiple systems meet the prerequisites and have the same available capacity, VCS selects the system appearing lexically first in the SystemList.

Systems having an available capacity of less than the percentage set by the LoadWarningLevel attribute, and those remaining at that load for longer than the time specified by the LoadTimeThreshold attribute invoke the loadwarning trigger.

Sample configurations depicting workload management

This topic lists some sample configurations that use the concepts.

System and Service group definitions

The main.cf in this example shows various Service Group Workload Management attributes in a system definition and a service group definition.

See "About attributes and their definitions" on page 693.

```
include "types.cf"
cluster SGWM-demo (
)

system LargeServer1 (
    Capacity = 200
    Limits = { ShrMemSeg=20, Semaphores=10, Processors=12 }
    LoadWarningLevel = 90
    LoadTimeThreshold = 600
    )
system LargeServer2 (
    Capacity = 200
    Limits = { ShrMemSeg=20, Semaphores=10, Processors=12 }
    LoadWarningLevel=70
    LoadTimeThreshold=300
)

system MedServer1 (
    Capacity = 100
    Limits = { ShrMemSeg=10, Semaphores=5, Processors=6 }
)

system MedServer2 (
    Capacity = 100
    Limits = { ShrMemSeg=10, Semaphores=5, Processors=6 }
)

group G1 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   MedServer1 = 2 , MedServer2 = 3  }
    SystemZones = { LargeServer1=0, LargeServer2=0,
```

```
                    MedServer1=1, MedServer2=1 }
    AutoStartPolicy = Load
    AutoStartList = { MedServer1, MedServer2 }
    FailOverPolicy = Load
    Load = { Units = 100 }
    Prerequisites = { ShrMemSeg=10, Semaphores=5, Processors=6 }
    )
```

Sample configuration: Basic four-node cluster

Following is the sample configuration for a basic four-node cluster:

```
include "types.cf"
cluster SGWM-demo

system Server1 (
    Capacity = 100
    )

system Server2 (
    Capacity = 100
    )

system Server3 (
    Capacity = 100
    )

system Server4 (
    Capacity = 100
    )

group G1 (
    SystemList = { Server1 = 0, Server2 = 1,
                   Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1 = 0, Server2 = 1,
                   Server3 = 2, Server4 = 3 }
    FailOverPolicy = Load
    Load = { CPU = 1, Mem = 2000 }
    )

group G2 (
    SystemList = { Server1 = 0, Server2 = 1,
```

```
                    Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1, Server2, Server3, Server4 }
    FailOverPolicy = Load
    Load = { Units = 40 }
    )

group G3 (
    SystemList = { Server1 = 0, Server2 = 1,
                    Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1, Server2, Server3, Server4 }
    FailOverPolicy = Load
    Load = { Units =  30 }
    )

group G4 (
    SystemList = { Server1 = 0, Server2 = 1,
                    Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1, Server2, Server3, Server4 }
    FailOverPolicy = Load
    Load = { Units = 10 }
    )

group G5 (
    SystemList = { Server1 = 0, Server2 = 1,
                    Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1, Server2, Server3, Server4 }
    FailOverPolicy = Load
    Load = { Units = 50 }
    )

group G6 (
    SystemList = { Server1 = 0, Server2 = 1,
                    Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1, Server2, Server3, Server4 }
    FailOverPolicy = Load
    Load = { Units = 30 }
    )
```

```
group G7 (
    SystemList = { Server1 = 0, Server2 = 1,
                   Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1, Server2, Server3, Server4 }
    FailOverPolicy = Load
    Load = { Units = 20 }
    )

group G8 (
    SystemList = { Server1 = 0, Server2 = 1,
                   Server3 = 2, Server4 = 3 }
    AutoStartPolicy = Load
    AutoStartList = { Server1, Server2, Server3, Server4 }
    FailOverPolicy = Load
    Load = { Units = 40 }
    )
```

See "About AutoStart operation" on page 393.

About AutoStart operation

In this configuration, assume that groups probe in the same order they are described, G1 through G8. Group G1 chooses the system with the highest AvailableCapacity value. All systems have the same available capacity, so G1 starts on Server1 because this server is lexically first. Groups G2 through G4 follow on Server2 through Server4.

Table 10-7 shows the Autostart cluster configuration for a basic four-node cluster with the initial four service groups online.

Table 10-7 Autostart cluster configuration for a basic four-node cluster

Server	Available capacity	Online groups
Server1	80	G1
Server2	60	G2
Server3	70	G3
Server4	90	G4

As the next groups come online, group G5 starts on Server4 because this server has the highest AvailableCapacity value. Group G6 then starts on Server1 with

AvailableCapacity of 80. Group G7 comes online on Server3 with AvailableCapacity of 70 and G8 comes online on Server2 with AvailableCapacity of 60.

Table 10-8 shows the Autostart cluster configuration for a basic four-node cluster with the other service groups online.

Table 10-8 Autostart cluster configuration for a basic four-node clusterwith the other service groups online

Server	Available capacity	Online groups
Server1	50	G1 and G6
Server2	20	G2 and G8
Server3	50	G3 and G7
Server4	40	G4 and G5

In this configuration, Server2 fires the loadwarning trigger after 600 seconds because it is at the default LoadWarningLevel of 80 percent.

About the failure scenario

In the first failure scenario, Server4 fails. Group G4 chooses Server1 because Server1 and Server3 have AvailableCapacity of 50 and Server1 is lexically first. Group G5 then comes online on Server3. Serializing the failover choice allows complete load-based control and adds less than one second to the total failover time.

Table 10-9 shows the cluster configuration following the first failure for a basic four-node cluster.

Table 10-9 Cluster configuration following the first failure

Server	Available capacity	Online groups
Server1	40	G1, G6, and G4
Server2	20	G2 and G8
Server3	0	G3, G7, and G5

In this configuration, Server3 fires the loadwarning trigger to notify that the server is overloaded. An administrator can then switch group G7 to Server1 to balance the load across groups G1 and G3. When Server4 is repaired, it rejoins the cluster with an AvailableCapacity value of 100, making it the most eligible target for a failover group.

About the cascading failure scenario

If Server3 fails before Server4 can be repaired, group G3 chooses Server1, group G5 chooses Server2, and group G7 chooses Server1. This results in the following configuration:

Table 10-10 shows a cascading failure scenario for a basic four node cluster.

Table 10-10 Cascading failure scenario for a basic four node cluster

Server	Available capacity	Online groups
Server1	-10	G1, G6, G4, G3, and G7
Server2	-30	G2, G8, and G5

Server1 fires the loadwarning trigger to notify that it is overloaded.

Sample configuration: Complex four-node cluster

The cluster in this example has two large enterprise servers (LargeServer1 and LargeServer2) and two medium-sized servers (MedServer1 and MedServer2). It has four service groups, G1 through G4, with various loads and prerequisites. Groups G1 and G2 are database applications with specific shared memory and semaphore requirements. Groups G3 and G4 are middle-tier applications with no specific memory or semaphore requirements.

```
include "types.cf"
cluster SGWM-demo (
)

system LargeServer1 (
Capacity = 200
Limits = { ShrMemSeg=20, Semaphores=10, Processors=12 }
LoadWarningLevel = 90
LoadTimeThreshold = 600
)

system LargeServer2 (
Capacity = 200
Limits = { ShrMemSeg=20, Semaphores=10, Processors=12 }
LoadWarningLevel=70
LoadTimeThreshold=300
)

system MedServer1 (
```

```
Capacity = 100
Limits = { ShrMemSeg=10, Semaphores=5, Processors=6 }
)

system MedServer2 (
Capacity = 100
Limits = { ShrMemSeg=10, Semaphores=5, Processors=6 }
)

group G1 (
SystemList = { LargeServer1 = 0, LargeServer2 = 1,
               MedServer1 = 2, MedServer2 = 3 }
SystemZones = { LargeServer1=0, LargeServer2=0, MedServer1=1,
        MedServer2=1 }
AutoStartPolicy = Load
AutoStartList = { LargeServer1, LargeServer2 }
FailOverPolicy = Load
Load = { Units = 100 }
Prerequisites = { ShrMemSeg=10, Semaphores=5, Processors=6 }
)

group G2 (
SystemList = { LargeServer1 = 0, LargeServer2 = 1,
               MedServer1 = 2, MedServer2 = 3 }
SystemZones = { LargeServer1=0, LargeServer2=0, MedServer1=1,
        MedServer2=1 }
AutoStartPolicy = Load
AutoStartList = { LargeServer1, LargeServer2 }
FailOverPolicy = Load
Load = { Units =  100 }
Prerequisites = { ShrMemSeg=10, Semaphores=5, Processors=6 }
)

group G3 (
SystemList = { LargeServer1 = 0, LargeServer2 = 1,
               MedServer1 = 2, MedServer2 = 3 }
SystemZones = { LargeServer1=0, LargeServer2=0, MedServer1=1,
        MedServer2=1 }
AutoStartPolicy = Load
AutoStartList = { MedServer1, MedServer2 }
FailOverPolicy = Load
Load = { Units = 30 }
)
```

```
group G4 (
SystemList = { LargeServer1 = 0, LargeServer2 = 1,
               MedServer1 = 2, MedServer2 = 3 }
SystemZones = { LargeServer1=0, LargeServer2=0, MedServer1=1,
        MedServer2=1 }
AutoStartPolicy = Load
AutoStartList = { MedServer1, MedServer2 }
FailOverPolicy = Load
Load = { Units = 20 }
)
```

About the AutoStart operation

In this configuration, the AutoStart sequence resembles:

G1—LargeServer1

G2—LargeServer2

G3—MedServer1

G4—MedServer2

All groups begin a probe sequence when the cluster starts. Groups G1 and G2 have an AutoStartList of LargeServer1 and LargeServer2. When these groups probe, they are queued to go online on one of these servers, based on highest AvailableCapacity value. If G1 probes first, it chooses LargeServer1 because LargeServer1 and LargeServer2 both have an AvailableCapacity of 200, but LargeServer1 is lexically first. Groups G3 and G4 use the same algorithm to determine their servers.

About the normal operation

Table 10-11 shows the cluster configuration for a normal operation for a complex four-node cluster.

Table 10-11 Normal operation cluster configuration for a complex four-node cluster

Server	Available capacity	Current limits	Online groups
LargeServer1	100	ShrMemSeg=10 Semaphores=5 Processors=6	G1

Table 10-11 Normal operation cluster configuration for a complex four-node cluster *(continued)*

Server	Available capacity	Current limits	Online groups
LargeServer2	100	ShrMemSeg=10 Semaphores=5 Processors=6	G2
MedServer1	70	ShrMemSeg=10 Semaphores=5 Processors=6	G3
MedServer2	80	ShrMemSeg=10 Semaphores=5 Processors=6	G4

About the failure scenario

In this scenario, if LargeServer2 fails, VCS scans all available systems in group G2's SystemList that are in the same SystemZone and creates a subset of systems that meet the group's prerequisites. In this case, LargeServer1 meets all required Limits. Group G2 is brought online on LargeServer1. This results in the following configuration:

Table 10-12 shows a failure scenario cluster configuration for a complex four-node cluster.

Table 10-12 Failure scenario cluster configuration for a complex four-node cluster

Server	Available capacity	Current limits	Online groups
LargeServer1	0	ShrMemSeg=0 Semaphores=0 Processors=0	G1, G2
MedServer1	70	ShrMemSeg=10 Semaphores=5 Processors=6	G3

Table 10-12 Failure scenario cluster configuration for a complex four-node cluster *(continued)*

Server	Available capacity	Current limits	Online groups
MedServer2	80	ShrMemSeg=10 Semaphores=5 Processors=6	G4

After 10 minutes (LoadTimeThreshold = 600) VCS fires the loadwarning trigger on LargeServer1 because the LoadWarningLevel exceeds 90 percent.

About the cascading failure scenario

In this scenario, another system failure can be tolerated because each system has sufficient Limits to accommodate the service group running on its peer. If MedServer1 fails, its groups can fail over to MedServer2.

If LargeServer1 fails, the failover of the two groups running on it is serialized. The first group lexically, G1, chooses MedServer2 because the server meets the required Limits and has AvailableCapacity value. Group G2 chooses MedServer1 because it is the only remaining system that meets the required Limits.

Sample configuration: Server consolidation

The following configuration has a complex eight-node cluster running multiple applications and large databases. The database servers, LargeServer1, LargeServer2, and LargeServer3, are enterprise systems. The middle-tier servers running multiple applications are MedServer1, MedServer2, MedServer3, MedServer4, and MedServer5.

In this configuration, the database zone (system zone 0) can handle a maximum of two failures. Each server has Limits to support a maximum of three database service groups. The application zone has excess capacity built into each server.

The servers running the application groups specify Limits to support one database, even though the application groups do not run prerequisites. This allows a database to fail over across system zones and run on the least-loaded server in the application zone.

```
include "types.cf"
cluster SGWM-demo (
)

system LargeServer1 (
```

```
    Capacity = 200
    Limits = { ShrMemSeg=15, Semaphores=30, Processors=18 }
    LoadWarningLevel = 80
    LoadTimeThreshold = 900
    )

system LargeServer2 (
    Capacity = 200
    Limits = { ShrMemSeg=15, Semaphores=30, Processors=18 }
    LoadWarningLevel=80
    LoadTimeThreshold=900
    )

system LargeServer3 (
    Capacity = 200
    Limits = { ShrMemSeg=15, Semaphores=30, Processors=18 }
    LoadWarningLevel=80
    LoadTimeThreshold=900
    )

system MedServer1 (
    Capacity = 100
    Limits = { ShrMemSeg=5, Semaphores=10, Processors=6 }
    )

system MedServer2 (
    Capacity = 100
    Limits = { ShrMemSeg=5, Semaphores=10, Processors=6 }
    )

system MedServer3 (
    Capacity = 100
    Limits = { ShrMemSeg=5, Semaphores=10, Processors=6 }
    )

system MedServer4 (
    Capacity = 100
    Limits = { ShrMemSeg=5, Semaphores=10, Processors=6 }
    )

system MedServer5 (
    Capacity = 100
    Limits = { ShrMemSeg=5, Semaphores=10, Processors=6 }
```

```
    )

group Database1 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
                   MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
    AutoStartList = { LargeServer1, LargeServer2, LargeServer3 }
    FailOverPolicy = Load
    Load = { Units = 100 }
    Prerequisites = { ShrMemSeg=5, Semaphores=10, Processors=6 }
    )

group Database2 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
                   MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
    AutoStartList = { LargeServer1, LargeServer2, LargeServer3 }
    FailOverPolicy = Load
    Load = { Units = 100 }
    Prerequisites = { ShrMemSeg=5, Semaphores=10, Processors=6 }
    )

group Database3 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
                   MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
```

```
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
    AutoStartList = { LargeServer1, LargeServer2, LargeServer3 }
    FailOverPolicy = Load
    Load = { Units = 100 }
    Prerequisites = { ShrMemSeg=5, Semaphores=10, Processors=6 }
    )

group Application1 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
                   MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
        AutoStartList = { MedServer1, MedServer2, MedServer3,
MedServer4,
            MedServer5 }
    FailOverPolicy = Load
    Load = { Units = 50 }
    )

group Application2 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
                   MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
    AutoStartList = { MedServer1, MedServer2, MedServer3,
MedServer4,
            MedServer5 }
    FailOverPolicy = Load
```

```
    Load = { Units = 50 }
    )

group Application3 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
                   MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
    AutoStartList = { MedServer1, MedServer2, MedServer3,
MedServer4,
            MedServer5 }
    FailOverPolicy = Load
    Load = { Units = 50 }
    )

group Application4 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
                   MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
    AutoStartList = { MedServer1, MedServer2, MedServer3,
MedServer4,
            MedServer5 }
    FailOverPolicy = Load
    Load = { Units = 50 }
    )

group Application5 (
    SystemList = { LargeServer1 = 0, LargeServer2 = 1,
                   LargeServer3 = 2, MedServer1 = 3,
                   MedServer2 = 4, MedServer3 = 5,
```

```
                    MedServer4 = 6, MedServer5 = 6 }
    SystemZones = { LargeServer1=0, LargeServer2=0,
LargeServer3=0,
            MedServer1=1, MedServer2=1, MedServer3=1,
MedServer4=1,
            MedServer5=1 }
    AutoStartPolicy = Load
    AutoStartList = { MedServer1, MedServer2, MedServer3,
MedServer4,
            MedServer5 }
    FailOverPolicy = Load
    Load = { Units = 50 }
    )
```

About the AutoStart operation

Based on the preceding main.cf example, the AutoStart sequence resembles:

Database1	LargeServer1
Database2	LargeServer2
Database3	LargeServer3
Application1	MedServer1
Application2	MedServer2
Application3	MedServer3
Application4	MedServer4
Application5	MedServer5

About the normal operation

Table 10-13 shows the normal operation cluster configuration for a complex eight-node cluster running multiple applications and large databases.

Table 10-13 Normal operation cluster configuration for a complex eight-node cluster running multiple applications and large databases

Server	Available capacity	Current limits	Online groups
LargeServer1	100	ShrMemSeg=10 Semaphores=20 Processors=12	Database1
LargeServer2	100	ShrMemSeg=10 Semaphores=20 Processors=12	Database2
LargeServer3	100	ShrMemSeg=10 Semaphores=20 Processors=12	Database3
MedServer1	50	ShrMemSeg=5 Semaphores=10 Processors=6	Application1
MedServer2	50	ShrMemSeg=5 Semaphores=10 Processors=6	Application2
MedServer3	50	ShrMemSeg=5 Semaphores=10 Processors=6	Application3
MedServer4	50	ShrMemSeg=5 Semaphores=10 Processors=6	Application4
MedServer5	50	ShrMemSeg=5 Semaphores=10 Processors=6	Application5

About the failure scenario

In the following example, LargeServer3 fails. VCS scans all available systems in the SystemList for the Database3 group for systems in the same SystemZone and

identifies systems that meet the group's prerequisites. In this case, LargeServer1 and LargeServer2 meet the required Limits. Database3 is brought online on LargeServer1. This results in the following configuration:

Table 10-14 shows the failure scenario for a complex eight-node cluster running multiple applications and large databases.

Table 10-14 Failure scenario for a complex eight-node cluster running multiple applications and large databases

Server	Available capacity	Current limits	Online groups
LargeServer1	0	ShrMemSeg=5 Semaphores=10 Processors=6	Database1 Database3
LargeServer2	100	ShrMemSeg=10 Semaphores=20 Processors=12	Database2

In this scenario, further failure of either system can be tolerated because each has sufficient Limits available to accommodate the additional service group.

About the cascading failure scenario

If the performance of a database is unacceptable with two database groups running on a single server, the SystemZones policy can help expedite performance. Failing over a database group into the application zone has the effect of resetting the group's preferred zone. For example, in the above scenario Database3 was moved to LargeServer1. The administrator could reconfigure the application zone to move two application groups to a single system. The database application can then be switched to the empty application server (MedServer1–MedServer5), which would put Database3 in Zone1 (application zone). If a failure occurs in Database3, the group selects the least-loaded server in the application zone for failover.

Chapter 11

The role of service group dependencies

This chapter includes the following topics:

- About service group dependencies
- Service group dependency configurations
- Frequently asked questions about group dependencies
- About linking service groups
- About linking multiple child service groups
- VCS behavior with service group dependencies

About service group dependencies

Service groups can be dependent on each other. The dependent group is the parent and the other group is the child. For example a finance application (parent) may require that the database application (child) is online before it comes online. While service group dependencies offer more features to manage application service groups, they create more complex failover configurations.

A service group may function both as a parent and a child. In VCS, a dependency tree may be up to five levels deep.

About dependency links

The dependency relationship between a parent and a child is called a link. The link is characterized by the dependency category, the location of the service groups, and the rigidity of dependency.

- A dependency may be online, or offline.
- A dependency may be local, global, remote, or site.
- A dependency may be soft, firm, or hard with respect to the rigidity of the constraints between parent and child service group.

You can customize the behavior of service groups by choosing the right combination of the dependency category, location, and rigidity

Dependency categories

Dependency categories determine the relationship of the parent group with the state of the child group.

Table 11-1 shows dependency categories and relationships between parent and child service groups.

Table 11-1 Dependency categories

Dependency	Relationship between parent and child service groups
Online group dependency	The parent group must wait for the child group to be brought online before it can start. For example, to configure a database application and a database service as two separate groups, specify the database application as the parent, and the database service as the child. Online group dependency supports various location-based and rigidity-based combinations.
Offline group dependency	The parent group can be started only if the child group is offline and vice versa. This behavior prevents conflicting applications from running on the same system. For example, configure a test application as the parent and the production application as the child. The parent and child applications can be configured on the same system or on different systems. Offline group dependency supports only offline-local dependency.

Dependency location

The relative location of the parent and child service groups determines whether the dependency between them is a local, global, remote or site.

Table 11-2 shows the dependency locations for local, global, remote and site dependencies.

Table 11-2 Dependency location

Dependency	Relative location of the parent and child service groups
Local dependency	The parent group depends on the child group being online or offline on the same system.
Global dependency	An instance of the parent group depends on one or more instances of the child group being online on any system in the cluster.
Remote dependency	An instance of the parent group depends on one or more instances of the child group being online on any system in the cluster other than the system on which the parent is online.
Site dependency	An instance of the parent group depends on one or more instances of the child group being online on any system in the same site.

Dependency rigidity

The type of dependency defines the rigidity of the link between parent and child groups. A soft dependency means minimum constraints, whereas a hard dependency means maximum constraints.

Table 11-3 shows dependency rigidity and associated constraints.

Table 11-3 Dependency rigidity

Dependency rigidity	Constraints between parent and child service groups
Soft dependency	Specifies the minimum constraints while bringing parent and child groups online. The only constraint is that the child group must be online before the parent group is brought online. For example, in an online local soft dependency, an instance of the child group must be online on the same system before the parent group can come online. Soft dependency provides the following flexibility: ■ If the child group faults, VCS does not immediately take the parent offline. If the child group cannot fail over, the parent remains online. ■ When both groups are online, either group, child or parent, may be taken offline while the other remains online. ■ If the parent group faults, the child group remains online. ■ When the link is created, the child group need not be online if the parent is online. However, when both groups are online, their online state must not conflict with the type of link.

Table 11-3 Dependency rigidity *(continued)*

Dependency rigidity	Constraints between parent and child service groups
Firm dependency	Imposes more constraints when VCS brings the parent or child groups online or takes them offline. In addition to the constraint that the child group must be online before the parent group is brought online, the constraints include: ■ If the child group faults, the parent is taken offline. If the parent is frozen at the time of the fault, the parent remains in its original state. If the child cannot fail over to another system, the parent remains offline. ■ If the parent group faults, the child group may remain online. ■ The child group cannot be taken offline if the parent group is online. The parent group can be taken offline while the child is online. ■ When the link is created, the parent group must be offline. However, if both groups are online, their online state must not conflict with the type of link.
Hard dependency	Imposes the maximum constraints when VCS brings the parent of child service groups online or takes them offline. For example: ■ If a child group faults, the parent is taken offline before the child group is taken offline. If the child group fails over, the parent fails over to another system (or the same system for a local dependency). If the child group cannot fail over, the parent group remains offline. ■ If the parent faults, the child is taken offline. If the child fails over, the parent fails over. If the child group cannot fail over, the parent group remains offline. **Note:** When the child faults, if the parent group is frozen, the parent remains online. The faulted child does not fail over. The following restrictions apply when configuring a hard dependency: ■ Only online local hard dependencies are supported. ■ Only a single-level, parent-child relationship can be configured as a hard dependency. ■ A child group can have only one online hard parent group. Likewise, a parent group can have only one online hard child group. ■ Bringing the child group online does not automatically bring the parent online. ■ Taking the parent group offline does not automatically take the child offline. ■ Bringing the parent online is prohibited if the child is offline. ■ Hard dependencies are allowed only at the bottommost level of dependency tree (leaf level).

About dependency limitations

Following are some service group dependency limitations:

- A group dependency tree may be at most five levels deep.
- You cannot link two service groups whose current states violate the relationship.
  For example, all link requests are accepted if all instances of parent group are offline.
  All link requests are rejected if parent group is online and child group is offline, except in offline dependencies and in soft dependencies.
  All online global link requests, online remote link requests, and online site link requests to link two parallel groups are rejected.
  All online local link requests to link a parallel parent group to a failover child group are rejected.
- Linking service groups using site dependencies:
  - If the service groups to be linked are online on different sites, you cannot use site dependency to link them.
  - All link requests to link parallel or hybrid parent groups to a failover or hybrid child service group are rejected.
  - If two service groups are already linked using a local, site, remote, or global dependency, you must unlink the existing dependency and use site dependency. However, you can configure site dependency with other online dependencies in multiple child or multiple parent configurations.

Service group dependency configurations

In the following tables, the term instance applies to parallel groups only. If a parallel group is online on three systems, for example, an instance of the group is online on each system. For failover groups, only one instance of a group is online at any time. The default dependency type is Firm.

See "Service group attributes" on page 719.

About failover parent / failover child

Table 11-4 shows service group dependencies for failover parent / failover child.

Table 11-4 Service group dependency configurations: Failover parent / Failover child

Link	Failover parent depends on ...	Failover parent is online If ...	If failover child faults, then ...	If failover parent faults, then ...
online local soft	Failover Child online on same system.	Child is online on same system.	Parent stays online. If Child fails over to another system, Parent migrates to the same system. If Child cannot fail over, Parent remains online.	Child stays online.
online local firm	Failover Child online on same system.	Child is online on same system.	Parent taken offline. If Child fails over to another system, Parent migrates to the same system. If Child cannot fail over, Parent remains offline.	Child stays online.
online local hard	Failover Child online on same system.	Child is online on same system.	Parents taken offline before Child is taken offline. If Child fails over to another system, Parent migrates to the same system. If Child cannot fail over, Parent remains offline.	Child taken offline. If Child fails over, Parent migrates to the same system. If Child cannot fail over, Parent remains offline.

Table 11-4 Service group dependency configurations: Failover parent / Failover child *(continued)*

Link	Failover parent depends on ...	Failover parent is online If ...	If failover child faults, then ...	If failover parent faults, then ...
online global soft	Failover Child online somewhere in the cluster.	Child is online somewhere in the cluster.	Parent stays online. If Child fails over to another system, Parent remains online. If Child cannot fail over, Parent remains online.	Child stays online. Parent fails over to any available system. If no failover target system is available, Parent remains offline.
online global firm	Failover Child online somewhere in the cluster.	Child is online somewhere in the cluster.	Parent taken offline after Child is taken offline. If Child fails over to another system, Parent is brought online on any system. If Child cannot fail over, Parent remains offline.	Child stays online. Parent fails over to any available system. If no failover target system is available, Parent remains offline.

Table 11-4 Service group dependency configurations: Failover parent / Failover child *(continued)*

Link	Failover parent depends on ...	Failover parent is online If ...	If failover child faults, then ...	If failover parent faults, then ...
online remote soft	Failover Child online on another system in the cluster.	Child is online on another system in the cluster.	If Child fails over to the system on which Parent was online, Parent migrates to another system. If Child fails over to another system, Parent continues to run on original system. If Child cannot fail over, Parent remains online.	Child stays online. Parent fails over to a system where Child is not online. If the only system available is where Child is online, Parent is not brought online. If no failover target system is available, Child remains online.
online remote firm	Failover Child online on another system in the cluster.	Child is online on another system in the cluster.	If Child fails over to the system on which Parent was online, Parent switches to another system. If Child fails over to another system, Parent restarts on original system. If Child cannot fail over, VCS takes the parent offline.	Parent fails over to a system where Child is not online. If the only system available is where Child is online, Parent is not brought online. If no failover target system is available, Child remains online.

Table 11-4 Service group dependency configurations: Failover parent / Failover child *(continued)*

Link	Failover parent depends on ...	Failover parent is online If ...	If failover child faults, then ...	If failover parent faults, then ...
online site soft	Failover Child online on same site.	Child is online in the same site.	Parent stays online. If another Child instance is online or Child fails over to a system within the same site, Parent stays online. If Child fails over to a system in another site, parent migrates to another site where Child is online and depends on Child instance(s) in that site. If Child cannot fail over, Parent remains online.	Child remains online. Parent fails over to another system in the same site maintaining dependency on Child instances in the same site. If Parent cannot failover to a system within the same site, then Parent fails over to a system in another site where at least one instance of child is online in the same site.

Table 11-4 Service group dependency configurations: Failover parent / Failover child *(continued)*

Link	Failover parent depends on ...	Failover parent is online If ...	If failover child faults, then ...	If failover parent faults, then ...
online site firm	Failover Child online in the same site	Child is online in the same site.	Parent taken offline. If another instance of child is online in the same site or child fails over to another system in the same site, Parent migrates to a system in the same site. If no Child instance is online or Child cannot fail over, Parent remains offline.	Child remains online. Parent fails over to another system in same site maintaining dependence on Child instances in the same site. If Parent cannot failover to a system within same site, then Parent fails over to a system in another site where at least one instance of child is online.

Table 11-4 Service group dependency configurations: Failover parent / Failover child *(continued)*

Link	Failover parent depends on ...	Failover parent is online If ...	If failover child faults, then ...	If failover parent faults, then ...
offline local	Failover Child offline on the same system	Child is offline on the same system.	If Child fails over to the system on which parent in not running, parent continues running. If child fails over to system on which parent is running, parent switches to another system, if available. If no failover target system is available for Child to fail over to, Parent continues running.	Parent fails over to system on which Child is not online. If no failover target system is available, Child remains online.

About failover parent / parallel child

With a failover parent and parallel child, no hard dependencies are supported.

Table 11-5 shows service group dependency configurations for Failover parent / Parallel child.

Table 11-5 Service group dependency configurations: Failover parent / Parallel child

Link	Failover parent depends on ...	Failover parent is online if ...	If parallel child faults on a system, then ...	If failover parent faults, then ...
online local soft	Instance of parallel Child group on same system.	Instance of Child is online on same system.	If Child instance fails over to another system, the Parent also fails over to the same system. If Child instance cannot failover to another system, Parent remains online.	Parent fails over to other system and depends on Child instance there. Child Instance remains online where the Parent faulted.
online local firm	Instance of parallel Child group on same system.	Instance of Child is online on same system.	Parent is taken offline. Parent fails over to other system and depends on Child instance there.	Parent fails over to other system and depends on Child instance there. Child Instance remains online where Parent faulted.
online global soft	All instances of parallel Child group online in the cluster.	At least one instance of Child group is online somewhere in the cluster.	Parent remains online if Child faults on any system. If Child cannot fail over to another system, Parent remains online.	Parent fails over to another system, maintaining dependence on all Child instances.

Table 11-5 Service group dependency configurations: Failover parent / Parallel child *(continued)*

Link	Failover parent depends on ...	Failover parent is online if ...	If parallel child faults on a system, then ...	If failover parent faults, then ...
online global firm	One or more instances of parallel Child group remaining online	An instance of Child group is online somewhere in the cluster.	Parent is taken offline. If another Child instance is online or Child fails over, Parent fails over to another system or the same system. If no Child instance is online or Child cannot fail over, Parent remains offline.	Parent fails over to another system, maintaining dependence on all Child instances.
online remote soft	One or more instances parallel Child group remaining online on other systems.	One or more instances of Child group are online on other systems.	Parent remains online. If Child fails over to the system on which Parent is online, Parent fails over to another system.	Parent fails over to another system, maintaining dependence on the Child instances.

Table 11-5 Service group dependency configurations: Failover parent / Parallel child *(continued)*

Link	Failover parent depends on ...	Failover parent is online if ...	If parallel child faults on a system, then ...	If failover parent faults, then ...
online remote firm	All instances parallel Child group remaining online on other systems.	All instances of Child group are online on other systems.	Parent is taken offline. If Child fails over to the system on which Parent is online, Parent fails over to another system. If Child fails over to another system, Parent is brought online on its original system.	Parent fails over to another system, maintaining dependence on all Child instances.
online site soft	One or more instances of parallel Child group in the same site.	At least one instance of Child is online in same site.	Parent stays online if child is online on any system in the same site. If Child fails over to a system in another site, Parent stays in the same site. If Child cannot fail over, Parent remains online.	Child stays online. Parents fails over to a system with Child online in the same site. If the parent group cannot failover, child group remains online

Table 11-5 Service group dependency configurations: Failover parent / Parallel child *(continued)*

Link	Failover parent depends on ...	Failover parent is online if ...	If parallel child faults on a system, then ...	If failover parent faults, then ...
online site firm	One or more instances of parallel Child group in the same site.	At least one instance of Child is online in same site.	Parent stays online if any instance of child is online in the same site. If Child fails over to another system, Parent migrates to a system in the same site. If Child cannot fail over, Parent remains offline.	Child stays online. Parents fails over to a system with Child online in the same site. If the parent group cannot failover, child group remains online.
offline local	Parallel Child offline on same system.	No instance of Child is online on same system.	Parent remains online if Child fails over to another system. If Child fails over to the system on which Parent is online, Parent fails over.	Child remains online and parent fails over to another system where child is not online. If the parent group cannot failover, child group remains offline.

About parallel parent / failover child

Table 11-6 shows service group dependencies for parallel parent / failover child.

Online local dependencies between parallel parent groups and failover child groups are not supported.

Table 11-6 Service group dependency configurations: Parallel parent / Failover child

Link	Parallel parent instances depend on ...	Parallel parent instances are online if ...	If failover child faults on a system, then ...	If parallel parent faults, then ...
online global soft	Failover Child group online somewhere in the cluster.	Failover Child is online somewhere in the cluster.	Parent remains online.	Child remains online
online global firm	Failover Child group somewhere in the cluster.	Failover Child is online somewhere in the cluster.	All instances of Parent taken offline. After Child fails over, Parent instances are failed over or restarted on the same systems.	Child stays online.
online remote soft	Failover Child group on another system.	Failover Child is online on another system.	If Child fails over to system on which Parent is online, Parent fails over to other systems. If Child fails over to another system, Parent remains online.	Child remains online. Parent tries to fail over to another system where child is not online.

Table 11-6 Service group dependency configurations: Parallel parent / Failover child *(continued)*

Link	Parallel parent instances depend on ...	Parallel parent instances are online if ...	If failover child faults on a system, then ...	If parallel parent faults, then ...
online remote firm	Failover Child group on another system.	Failover Child is online on another system.	All instances of Parent taken offline. If Child fails over to system on which Parent was online, Parent fails over to other systems. If Child fails over to another system, Parent brought online on same systems.	Child remains online. Parent tries to fail over to another system where child is not online.
offline local	Failover Child offline on same system.	Failover Child is not online on same system.	Parent remains online if Child fails over to another system. Child fails over to another system. If Parent is online on that system, Parent is brought offline. Parent fails over to any other system.	Child remains online.

About parallel parent / parallel child

Global and remote dependencies between parallel parent groups and parallel child groups are not supported.

Table 11-7 shows service group dependency configurations for parallel parent / parallel child.

Table 11-7 Service group dependency configurations: Parallel parent / Parallel child

Link	Parallel parent depends on ...	Parallel parent is online If ...	If parallel child faults, then ...	If parallel parent faults, then ...
online local soft	Parallel Child instance online on same system.	Parallel Child instance is online on same system.	If Child fails over to another system, Parent migrates to the same system as the Child. If Child cannot fail over, Parent remains online.	Child instance stays online. Parent instance can fail over only to system where Child instance is running and other instance of Parent is not running.
online local firm	Parallel Child instance online on same system.	Parallel Child instance is online on same system.	Parent taken offline. If Child fails over to another system, VCS brings an instance of the Parent online on the same system as Child. If Child cannot fail over, Parent remains offline.	Child stays online. Parent instance can fail over only to system where Child instance is running and other instance of Parent is not running.

Table 11-7 Service group dependency configurations: Parallel parent / Parallel child *(continued)*

Link	Parallel parent depends on ...	Parallel parent is online If ...	If parallel child faults, then ...	If parallel parent faults, then ...
offline local	Parallel Child offline on same system.	No instance of Child is online on same system.	Parent remains online if Child fails over to another system. Parent goes offline if Child fails over to a system where Parent is online. Parent fails over to another system where Child is not online.	Child remains online. Parent fails over to a system where Child is not online.

Frequently asked questions about group dependencies

Table 11-8 lists some commonly asked questions about group dependencies.

Table 11-8 Frequently asked questions about group dependencies

Dependency	Frequently asked questions
Online local	Can child group be taken offline when parent group is online? Soft=Yes Firm=No Hard = No. Can parent group be switched while child group is online? Soft=No Firm=No Hard = No. Can child group be switched while parent group is online? Soft=No Firm=No Hard = No.

Table 11-8 Frequently asked questions about group dependencies *(continued)*

Dependency	Frequently asked questions
Online global	Can child group be taken offline when parent group is online? Soft=Yes Firm=No. Can parent group be switched while child group is running? Soft=Yes Firm=Yes. Can child group be switched while parent group is running? Soft=Yes Firm=No
Online remote	Can child group be taken offline when parent group is online? Soft=Yes Firm=No. Can parent group be switched while child group is running? Soft=Yes, but not to system on which child is running. Firm=Yes, but not to system on which child is running. Can child group be switched while parent group is running? Soft=Yes Firm=No, but not to system on which parent is running.
Offline local	Can parent group be brought online when child group is offline? Yes. Can child group be taken offline when parent group is online? Yes. Can parent group be switched while the child group is running? Yes, but not to system on which child is running. Can child group be switched while the parent group is running? Yes, but not to system on which parent is running.
Online site	Can child group be taken offline when parent group is online? Soft=Yes Firm=No. Can parent group be switched to a system in the same site while child group is running? Soft=Yes Firm=Yes. Can child group be switched while parent group is running? Soft=Yes Firm=No

About linking service groups

Note that a configuration may require that a certain service group be running before another service group can be brought online. For example, a group containing resources of a database service must be running before the database application is brought online.

Use the following command to link service groups from the command line

```
hagrp -link parent_group child_group gd_category gd_location [gd_type]
```

parent_group	Name of the parent group
child_group	Name of the child group
gd_category	category of group dependency (online/offline).
gd_location	the scope of dependency (local/global/remote/site).
gd_type	type of group dependency (soft/firm/hard). Default is firm

About linking multiple child service groups

VCS lets you configure multiple child service groups for a single parent service group to depend on. For example, an application might depend on two or more databases.

Dependencies supported for multiple child service groups

Multiple child service groups can be linked to a parent service group using soft and firm type of dependencies. Hard dependencies and offline local dependencies are not supported

The supported dependency relationships between a parent service group and multiple child service groups include:

- Soft dependency
  All child groups can be online local soft.
- Firm dependency
  All child groups can be online local firm.
- A combination of soft and firm dependencies
  Some child groups can be online local soft and other child groups can be online local firm.

- A combination of local, global, remote, and site dependencies
  One child group can be online local soft, another child group can be online remote soft, and yet another child group can be online global firm.

Dependencies not supported for multiple child service groups

Multiple child service groups *cannot* be linked to a parent service group using offline local and online local hard dependencies. Moreover, these dependencies *cannot* be used with other dependency types.

A few examples of dependency relationships that are *not* supported between a parent service group and multiple child service groups:

- A configuration in which multiple child service groups are offline local.
- A configuration in which multiple child service groups are online local hard.
- Any combination of offline local and other dependency types
  You cannot have a combination of offline local and online local soft or a combination of offline local and online local hard.
- Any combination of online local hard and other dependency types
  You cannot have a combination of online local hard and online local soft.

VCS behavior with service group dependencies

VCS enables or restricts service group operations to honor service group dependencies. VCS rejects operations if the operation violates a group dependency.

Online operations in group dependencies

Typically, bringing a child group online manually is never rejected, except under the following circumstances:

- For online local dependencies, if parent is online, a child group online is rejected for any system other than the system where parent is online.
- For online remote dependencies, if parent is online, a child group online is rejected for the system where parent is online.
- For offline local dependencies, if parent is online, a child group online is rejected for the system where parent is online.

The following examples describe situations where bringing a parallel child group online is accepted:

- For a parallel child group linked online local with failover/parallel parent, multiple instances of child group online are acceptable.

- For a parallel child group linked online remote with failover parent, multiple instances of child group online are acceptable, as long as child group does not go online on the system where parent is online.
- For a parallel child group linked offline local with failover/parallel parent, multiple instances of child group online are acceptable, as long as child group does not go online on the system where parent is online.

Offline operations in group dependencies

VCS rejects offline operations if the procedure violates existing group dependencies. Typically, firm dependencies are more restrictive to taking child group offline while parent group is online. Rules for manual offline include:

- Parent group offline is never rejected.
- For all soft dependencies, child group can go offline regardless of the state of parent group.
- For all firm dependencies, if parent group is online, child group offline is rejected.
- For the online local hard dependency, if parent group is online, child group offline is rejected.

Switch operations in group dependencies

Switching a service group implies manually taking a service group offline on one system, and manually bringing it back online on another system. VCS rejects manual switch if the group does not comply with the rules for offline or online operations.

Section 4

Administration - Beyond the basics

- Chapter 12. VCS event notification
- Chapter 13. VCS event triggers
- Chapter 14. Virtual Business Services

Chapter 12

VCS event notification

This chapter includes the following topics:

- About VCS event notification
- Components of VCS event notification
- About VCS events and traps
- About monitoring aggregate events
- About configuring notification

About VCS event notification

VCS provides a method for notifying important events such as resource or system faults to administrators or designated recipients. VCS includes a notifier component, which consists of the notifier process and the hanotify utility.

VCS supports SNMP consoles that can use an SNMP V2 MIB.

The notifier process performs the following tasks:

- Receives notifications from HAD
- Formats the notification
- Generates an SNMP (V2) trap or sends an email to the designated recipient, or does both.

If you have configured owners for resources, groups, or for the cluster, VCS also notifies owners of the events that affect their resources. A resource owner is notified of resource-related events, a group owner of group-related events, and so on.

You can also configure persons other than owners as recipients of notifications about events of a resource, resource type, service group, system, or cluster. The registered recipients get notifications for the events that have a severity level that

is equal to or greater than the level specified. For example, if you configure recipients for notifications and specify the severity level as Warning, VCS notifies the recipients about events with the severity levels Warning, Error, and SevereError but not about events with the severity level Information.

See "About attributes and their definitions" on page 693.

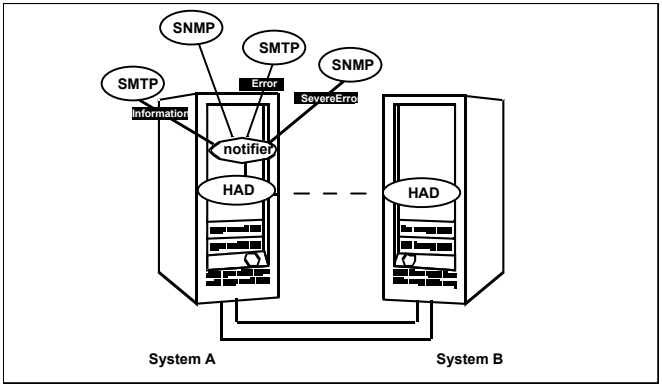
Figure 12-1 shows the severity levels of VCS events.

Table 12-1 VCS event severity levels

Severity level	Denotes
SevereError	Critical errors that can lead to data loss or corruption; SevereError is the highest severity level.
Error	Faults
Warning	Deviations from normal behavior
Information	Important events that exhibit normal behavior; Information is the lowest severity level.

Note: Severity levels are case-sensitive.

Figure 12-1 VCS event notification: Severity levels



SNMP traps are forwarded to the SNMP console. Typically, traps are predefined for events such as service group or resource faults. You can use the hanotify utility to send additional traps.

Event messages and severity levels

When the VCS engine starts up, it queues all messages of severity Information and higher for later processing.

When notifier connects, it communicates to HAD the lowest severity threshold level currently defined for the SNMP option or for the SMTP option.

If notifier is started from the command line without specifying a severity level for the SNMP console or SMTP recipients, notifier communicates the default severity level Warning to HAD. If notifier is configured under VCS control, severity must be specified.

See the description of the NotifierMngr agent in the *Cluster Server Bundled Agents Reference Guide*.

For example, if the following severities are specified for notifier:

- Warning for email recipient 1
- Error for email recipient 2
- SevereError for SNMP console

Notifier communicates the minimum severity, Warning, to HAD, which then queues all messages labelled severity level Warning and greater.

Notifier ensures that recipients receive only the messages they are designated to receive according to the specified severity level. However, until notifier communicates the specifications to HAD, HAD stores all messages, because it does not know the severity the user has specified. This behavior prevents messages from being lost between the time HAD stores them and notifier communicates the specifications to HAD.

About persistent and replicated message queue

VCS includes a sophisticated mechanism for maintaining event messages, which ensures that messages are not lost. On each node, VCS queues messages to be sent to the notifier process. This queue is persistent as long as VCS is running and the contents of this queue remain the same on each node. If the notifier service group fails, notifier is failed over to another node in the cluster. Because the message queue is consistent across nodes, notifier can resume message delivery from where it left off even after failover.

How HAD deletes messages

The VCS engine, HAD, stores messages to be sent to notifier. After every 180 seconds, HAD tries to send all the pending notifications to notifier. When HAD receives an acknowledgement from notifier that a message is delivered to at least

one of the recipients, it deletes the message from its queue. For example, if two SNMP consoles and two email recipients are designated, notifier sends an acknowledgement to HAD even if the message reached only one of the four recipients. If HAD does not get acknowledgement for some messages, it keeps on sending these notifications to notifier after every 180 seconds till it gets an acknowledgement of delivery from notifier. An error message is printed to the log file when a delivery error occurs.

HAD deletes messages under the following conditions too:

- The message has been in the queue for time (in seconds) specified in MessageExpiryInterval attribute (default value: one hour) and notifier is unable to deliver the message to the recipient.
- The message queue is full and to make room for the latest message, the earliest message is deleted.

Components of VCS event notification

This topic describes the notifier process and the hanotify utility.

About the notifier process

The notifier process configures how messages are received from VCS and how they are delivered to SNMP consoles and SMTP servers. Using notifier, you can specify notification based on the severity level of the events generating the messages. You can also specify the size of the VCS message queue, which is 30 by default. You can change this value by modifying the MessageQueue attribute.

See the *Cluster Server Bundled Agents Reference Guide* for more information about this attribute.

When notifier is started from the command line, VCS does not control the notifier process. For best results, use the NotifierMngr agent that is bundled with VCS. Configure notifier as part of a highly available service group, which can then be monitored, brought online, and taken offline.

For information about the agent, see the *Cluster Server Bundled Agents Reference Guide*.

Note that notifier must be configured in a failover group, not parallel, because only one instance of notifier runs in the entire cluster. Also note that notifier does not respond to SNMP `get` or `set` requests; notifier is a trap generator only.

Notifier enables you to specify configurations for the SNMP manager and SMTP server, including DNS resolvable hostnames, ports, community IDs, and recipients'

email addresses. You can specify more than one manager or server, and the severity level of messages that are sent to each.

Note: If you start the notifier outside of VCS control, use the absolute path of the notifier in the command. VCS cannot monitor the notifier process if it is started outside of VCS control using a relative path.

Example of notifier command

Following is an example of a `notifier` command:

```
 /opt/VRTSvcs/bin/notifier -s m=north -s
m=south,p=2000,l=Error,c=your_company
        -t m=north,e="abc@your_company.com",l=SevereError
```

In this example, notifier:

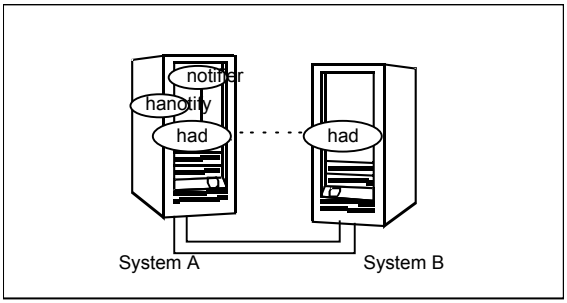
- Sends all level SNMP traps to north at the default SNMP port and community value public.
- Sends Warning traps to north.
- Sends Error and SevereError traps to south at port 2000 and community value your_company.
- Sends SevereError email messages to north as SMTP server at default port and to email recipient abc@your_company.com.

About the hanotify utility

The hanotify utility enables you to construct user-defined messages. The utility forwards messages to HAD, which stores them in its internal message queue. Along with other messages, user-defined messages are also forwarded to the notifier process for delivery to email recipients, SNMP consoles, or both.

Figure 12-2 shows the hanotify utility.

Figure 12-2 hanotify utility



Example of hanotify command

Following is an example of `hanotify` command:

```
hanotify -i 1.3.6.1.4.1.1302.3.8.10.2.8.0.10 -l Warning -n
agentres -T 7 -t "custom agent" -o 4 -S sys1 -L mv -p
  sys2 -P mv -c MyAgent -C 7 -O johndoe -m "Custom message"
```

In this example, the number 1.3.6.1.4.1.1302.3.8.10.2.8.0.10 is the OID (Object Identifier) for the message being sent. Because it is a user-defined message, HAD has no way of knowing the OID associated with the SNMP trap corresponding to this message. Users must provide the OID.

The message severity level is set to Warning. The affected systems are sys1 and sys2. Running this command generates a custom notification with the message "Custom message" for the resource *agentres*.

About VCS events and traps

This topic lists the events that generate traps, email notification, or both. Note that SevereError indicates the highest severity level, Information the lowest. Traps specific to global clusters are ranked from Critical, the highest severity, to Normal, the lowest.

Events and traps for clusters

Table 12-2 shows events and traps for clusters.

Table 12-2 Events and traps for clusters

Event	Severity level	Description
Cluster has faulted.	Error	The cluster is down because of a fault.
Heartbeat is down. (Global Cluster Option)	Error	The connector on the local cluster lost its heartbeat connection to the remote cluster.
Remote cluster is in RUNNING state. (Global Cluster Option)	Information	Local cluster has complete snapshot of the remote cluster, indicating the remote cluster is in the RUNNING state.
Heartbeat is "alive." (Global Cluster Option)	Information	The heartbeat between clusters is healthy.
User has logged on to VCS.	Information	A user log on has been recognized because a user logged on by Cluster Manager, or because a `haxxx` command was invoked.

Events and traps for agents

Table 12-3 depicts events and traps for agents.

Table 12-3 Events and traps for agents

Event	Severity level	Description
Agent is faulted.	Warning	The agent has faulted on one node in the cluster.
Agent is restarting	Information	VCS is restarting the agent.

Events and traps for resources

Table 12-4 depicts events and traps for resources.

Table 12-4 Events and traps for resources

Event	Severity level	Description
Resource state is unknown.	Warning	VCS cannot identify the state of the resource.
Resource monitoring has timed out.	Warning	Monitoring mechanism for the resource has timed out.
Resource is not going offline.	Warning	VCS cannot take the resource offline.
Health of cluster resource declined.	Warning	Used by agents to give additional information on the state of a resource. A decline in the health of the resource was identified during monitoring.
Resource went online by itself.	Warning (not for first probe)	The resource was brought online on its own.
Resource has faulted.	Error	The resource has faulted on one node in the cluster.
Resource is being restarted by agent.	Information	The agent is restarting the resource.
The health of cluster resource improved.	Information	Used by agents to give extra information about state of resource. An improvement in the health of the resource was identified during monitoring.

Table 12-4 Events and traps for resources *(continued)*

Event	Severity level	Description
Resource monitor time has changed.	Warning	This trap is generated when statistical analysis for the time taken by the monitor function of an agent is enabled for the agent. See " VCS agent statistics" on page 587. This trap is generated when the agent framework detects a sudden change in the time taken to run the monitor function for a resource. The trap information contains details of: ■ The change in time required to run the monitor function ■ The actual times that were compared to deduce this change.
Resource is in ADMIN_WAIT state.	Error	The resource is in the admin_wait state. See " About controlling Clean behavior on resource faults" on page 361.

Events and traps for systems

Table 12-5 depicts events and traps for systems.

Table 12-5 Events and traps for systems

Event	Severity level	Description
VCS is being restarted by `hashadow`.	Warning	The hashadow process is restarting the VCS engine.
VCS is in jeopardy.	Warning	One node running VCS is in jeopardy.

Table 12-5 Events and traps for systems *(continued)*

Event	Severity level	Description
VCS is up on the first node in the cluster.	Information	VCS is up on the first node.
VCS has faulted.	SevereError	VCS is down because of a fault.
A node running VCS has joined cluster.	Information	The cluster has a new node that runs VCS.
VCS has exited manually.	Information	VCS has exited gracefully from one node on which it was previously running.
CPU usage exceeded threshold on the system.	Warning	The system's CPU usage exceeded the Warning threshold level set in the CPUThreshold attribute.
Swap usage exceeded threshold on the system.	Warning	The system's swap usage exceeded the Warning threshold level set in the SwapThreshold attribute.
Memory usage exceeded threshold on the system.	Warning	The system's Memory usage exceeded the Warning threshold level set in the MemThresholdLevel attribute.

Events and traps for service groups

Table 12-6 depicts events and traps for service groups.

Table 12-6 Events and traps for service groups

Event	Severity level	Description
Service group has faulted.	Error	The service group is offline because of a fault.

Table 12-6 Events and traps for service groups *(continued)*

Event	Severity level	Description
Service group concurrency violation.	SevereError	A failover service group has become online on more than one node in the cluster.
Service group has faulted and cannot be failed over anywhere.	SevereError	Specified service group faulted on all nodes where group could be brought online. There are no nodes to which the group can fail over.
Service group is online	Information	The service group is online.
Service group is offline.	Information	The service group is offline.
Service group is autodisabled.	Information	VCS has autodisabled the specified group because one node exited the cluster.
Service group is restarting.	Information	The service group is restarting.
Service group is being switched.	Information	VCS is taking the service group offline on one node and bringing it online on another.
Service group restarting in response to persistent resource going online.	Information	The service group is restarting because a persistent resource recovered from a fault.
The global service group is online/partial on multiple clusters. (Global Cluster Option)	SevereError	A concurrency violation occurred for the global service group.
Attributes for global service groups are mismatched. (Global Cluster Option)	Error	The attributes ClusterList, AutoFailOver, and Parallel are mismatched for the same global service group on different clusters.

SNMP-specific files

VCS includes two SNMP-specific files: vcs.mib and vcs_trapd, which are created in:

/etc/VRTSvcs/snmp

The file vcs.mib is the textual MIB for built-in traps that are supported by VCS. Load this MIB into your SNMP console to add it to the list of recognized traps.

The file vcs_trapd is specific to the HP OpenView Network Node Manager (NNM) SNMP console. The file includes sample events configured for the built-in SNMP traps supported by VCS. To merge these events with those configured for SNMP traps:

```
xnmevents -merge vcs_trapd
```

When you merge events, the SNMP traps sent by VCS by way of notifier are displayed in the HP OpenView NNM SNMP console.

Note: For more information on xnmevents, see the HP OpenView documentation.

Trap variables in VCS MIB

Traps sent by VCS are reversible to SNMPv2 after an SNMPv2 to SNMPv1 conversion.

For reversible translations between SNMPv1 and SNMPv2 trap PDUs, the second-last ID of the SNMP trap OID must be zero. This ensures that once you make a forward translation (SNMPv2 trap to SNMPv1; RFC 2576 Section 3.2), the reverse translation (SNMPv1 trap to SNMPv2 trap; RFC 2576 Section 3.1) is accurate.

The VCS notifier follows this guideline by using OIDs with second-last ID as zero, enabling reversible translations.

About severityId

This variable indicates the severity of the trap being sent.

Table 12-7 shows the values that the variable severityId can take.

Table 12-7 Possible values of the variable severityId

Severity level and description	Value in trap PDU
Information Important events exhibiting normal behavior	0
Warning Deviation from normal behavior	1

Table 12-7 Possible values of the variable severityId *(continued)*

Severity level and description	Value in trap PDU
Error A fault	2
Severe Error Critical error that can lead to data loss or corruption	3

EntityType and entitySubType

These variables specify additional information about the entity.

Table 12-8 shows the variables entityType and entitySubType.

Table 12-8 Variables entityType and entitySubType

Entity type	Entity sub-type
Resource	String. For example, disk.
Group	String The type of the group (failover or parallel)
System	String. For example, RHEL 5.0.
Heartbeat	String Type of the heartbeat
VCS	String
GCO	String
Agent name	String The agent name

About entityState

This variable describes the state of the entity.

Table 12-9 shows the the various states.

Table 12-9 Possible states

Entity	States
VCS states	■ User has logged into VCS ■ Cluster has faulted ■ Cluster is in RUNNING state
Agent states	■ Agent is restarting ■ Agent has faulted
Resources states	■ Resource state is unknown ■ Resource monitoring has timed out ■ Resource is not going offline ■ Resource is being restarted by agent ■ Resource went online by itself ■ Resource has faulted ■ Resource is in admin wait state ■ Resource monitor time has changed
Service group states	■ Service group is online ■ Service group is offline ■ Service group is auto disabled ■ Service group has faulted ■ Service group has faulted and cannot be failed over anywhere ■ Service group is restarting ■ Service group is being switched ■ Service group concurrency violation ■ Service group is restarting in response to persistent resource going online ■ Service group attribute value does not match corresponding remote group attribute value ■ Global group concurrency violation
System states	■ VCS is up on the first node in the Cluster ■ VCS is being restarted by hashadow ■ VCS is in jeopardy ■ VCS has faulted ■ A node running VCS has joined cluster ■ VCS has exited manually ■ CPU usage exceeded the threshold on the system ■ Memory usage exceeded the threshold on the system ■ Swap usage exceeded the threshold on the system

Table 12-9 Possible states *(continued)*

Entity	States
GCO heartbeat states	■ Cluster has lost heartbeat with remote cluster ■ Heartbeat with remote cluster is alive

About monitoring aggregate events

This topic describes how you can detect aggregate events by monitoring individual notifications.

How to detect service group failover

VCS does not send any explicit traps when a failover occurs in response to a service group fault. When a service group faults, VCS generates the following notifications if the AutoFailOver attribute for the service group is set to 1:

- Service Group Fault for the node on which the service group was online and faulted
- Service Group Offline for the node on which the service group faulted
- Service Group Online for the node to which the service group failed over

How to detect service group switch

When a service group is switched, VCS sends a notification of severity Information to indicate the following events:

- Service group is being switched.
- Service group is offline for the node from which the service group is switched.
- Service group is online for the node to which the service group was switched. This notification is sent after VCS completes the service group switch operation.

Note: You must configure appropriate severity for the notifier to receive these notifications. To receive VCS notifications, the minimum acceptable severity level is Information.

About configuring notification

Configuring notification involves creating a resource for the Notifier Manager (NotifierMgr) agent in the ClusterService group.

See the *Cluster Server Bundled Agents Reference Guide* for more information about the agent.

VCS provides several methods for configuring notification:

- Manually editing the main.cf file.
- Using the Notifier wizard.

Chapter 13

VCS event triggers

This chapter includes the following topics:

- About VCS event triggers
- Using event triggers
- List of event triggers

About VCS event triggers

Triggers let you invoke user-defined scripts for specified events in a cluster.

VCS determines if the event is enabled and invokes the `hatrigger` script. The script is located at:

$VCS_HOME/bin/hatrigger

VCS also passes the name of the event trigger and associated parameters. For example, when a service group comes online on a system, VCS invokes the following command:

```
hatrigger -postonline system service_group
```

VCS does not wait for the trigger to complete execution. VCS calls the trigger and continues normal operation.

VCS invokes event triggers on the system where the event occurred, with the following exceptions:

- VCS invokes the sysoffline and nofailover event triggers on the lowest-numbered system in the RUNNING state.
- VCS invokes the violation event trigger on all systems on which the service group was brought partially or fully online.

By default, the `hatrigger` script invokes the trigger script(s) from the default path `$VCS_HOME/bin/triggers`. You can customize the trigger path by using the TriggerPath attribute.

See "Resource attributes" on page 694.

See "Service group attributes" on page 719.

The same path is used on all nodes in the cluster. The trigger path must exist on all the cluster nodes. On each cluster node, the trigger scripts must be installed in the trigger path.

Using event triggers

VCS provides a sample Perl script for each event trigger at the following location:

$VCS_HOME/bin/sample_triggers/VRTSvcs

Customize the scripts according to your requirements: you may choose to write your own Perl scripts.

To use an event trigger

1. Use the sample scripts to write your own custom actions for the trigger.
2. Move the modified trigger script to the following path on each node:

   $VCS_HOME/bin/triggers
3. Configure other attributes that may be required to enable the trigger. See the usage information for the trigger for more information.

Performing multiple actions using a trigger

To perform multiple actions by using a trigger, specify the actions in the following format:

T*NumService*

Example: T1clear_env, T2update_env, T5backup

VCS executes the scripts in the ascending order of T*num*. The actions must be placed inside a directory whose name is the trigger name. For example, to perform multiple actions for the postonline trigger, place the scripts in a directory named postonline.

Note: As a good practice, ensure that one script does not affect the functioning of another script. If script2 takes the output of script1 as an input, script2 must be capable of handling any exceptions that arise out of the behavior of script1.

Warning: Do not install customized trigger scripts in the $VCS_HOME/bin/sample_triggers/VRTSvcs directory or in the $VCS_HOME/bin/internal_triggers directory. If you install customized triggers in these directories, you might face issues while upgrading VCS.

List of event triggers

The information in the following sections describes the various event triggers, including their usage, parameters, and location.

About the dumptunables trigger

The following table describes the dumptunables event trigger:

Description The dumptunables trigger is invoked when HAD goes into the RUNNING state. When this trigger is invoked, it uses the HAD environment variables that it inherited, and other environment variables to process the event. Depending on the value of the *to_log* parameter, the trigger then redirects the environment variables to either stdout or the engine log.

This trigger is not invoked when HAD is restarted by hashadow.

This event trigger is internal and non-configurable.

Usage `-dumptunables triggertype system to_log`

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the name of the system on which the trigger is invoked.

to_log—represents whether the output is redirected to engine log (*to_log*=1) or stdout (*to_log*=0).

About the globalcounter_not_updated trigger

The following table describes the globalcounter_not_updated event trigger:

Description On the system having lowest NodeId in the cluster, VCS periodically broadcasts an update of GlobalCounter. If a node does not receive the broadcast for an interval greater than CounterMissTolerance, it invokes the globalcounter_not_updated trigger if CounterMissAction is set to Trigger. This event is considered critical since it indicates a problem with underlying cluster communications or cluster interconnects. Use this trigger to notify administrators of the critical events.

Usage `-globalcounter_not_updated` *`triggertype system_name global_counter`*

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the system which did not receive the update of GlobalCounter.

global_counter—represents the value of GlobalCounter.

About the injeopardy event trigger

The following table describes the injeopardy event trigger:

Description Invoked when a system is in jeopardy. Specifically, this trigger is invoked when a system has only one remaining link to the cluster, and that link is a network link (LLT). This event is considered critical because if the system loses the remaining network link, VCS does not fail over the service groups that were online on the system. Use this trigger to notify the administrator of the critical event. The administrator can then take appropriate action to ensure that the system has at least two links to the cluster.

This event trigger is non-configurable.

Usage `-injeopardy` *`triggertype system system_state`*

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the name of the system.

system_state—represents the value of the State attribute.

About the loadwarning event trigger

The following table describes the loadwarning event trigger:

Description

Invoked when a system becomes overloaded because the load of the system's online groups exceeds the system's LoadWarningLevel attribute for an interval exceeding the LoadTimeThreshold attribute.

For example, assume that the Capacity is 150, the LoadWarningLevel is 80, and the LoadTimeThreshold is 300. Also, the sum of the Load attribute for all online groups on the system is 135. Because the LoadWarningLevel is 80, safe load is 0.80*150=120. The trigger is invoked if the system load stays at 135 for more than 300 seconds because the actual load is above the limit of 120 specified by LoadWarningLevel.

Use this trigger to notify the administrator of the critical event. The administrator can then switch some service groups to another system, ensuring that no one system is overloaded.

This event trigger is non-configurable.

Usage

`-loadwarning` *`triggertype system available_capacity`*

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the name of the system.

available_capacity—represents the system's AvailableCapacity attribute. (AvailableCapacity=Capacity-sum of Load for system's online groups.)

About the nofailover event trigger

The following table describes the nofailover event trigger:

Description

Called from the lowest-numbered system in RUNNING state when a service group cannot fail over.

This event trigger is non-configurable.

Usage

`-nofailover` *`triggertype system service_group`*

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the name of the last system on which an attempt was made to bring the service group online.

service_group—represents the name of the service group.

About the postoffline event trigger

The following table describes the postoffline event trigger:

Description	This event trigger is invoked on the system where the group went offline from a partial or fully online state. This trigger is invoked when the group faults, or is taken offline manually. This event trigger is non-configurable.
Usage	`-postoffline` *`triggertype system service_group`* *triggertype*—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1). For this trigger, *triggertype*=0. *system*—represents the name of the system. *service_group*—represents the name of the service group that went offline.

About the postonline event trigger

The following table describes the postonline event trigger:

Description	This event trigger is invoked on the system where the group went online from an offline state. This event trigger is non-configurable.
Usage	`-postonline` *`triggertype system service_group`* *triggertype*—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1). For this trigger, *triggertype*=0. *system*—represents the name of the system. *service_group*—represents the name of the service group that went online.

About the preonline event trigger

The following table describes the preonline event trigger:

Description	Indicates when the HAD should call a user-defined script before bringing a service group online in response to the `hagrp -online` command or a fault. If the trigger does not exist, VCS continues to bring the group online. If the script returns 0 without an exit code, VCS runs the `hagrp -online -nopre` command, with the `-checkpartial` option if appropriate. If you do want to bring the group online, define the trigger to take no action. This event trigger is configurable.
Usage	`-preonline` *`triggertype system service_group whyonlining [system_where_group_faulted]`* *triggertype*—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1). For this trigger, *triggertype*=0. *system*—represents the name of the system. *service_group*—represents the name of the service group on which the `hagrp` command was issued or the fault occurred. *whyonlining*—represents three values: FAULT: Indicates that the group was brought online in response to a group failover. MANUAL: Indicates that group was brought online or switched manually on the system that is represented by the variable *system*. SYSFAULT: Indicates that the group was brought online in response to a sytem fault. *system_where_group_faulted*—represents the name of the system on which the group has faulted or switched. This variable is optional and set when the engine invokes the trigger during a failover or switch.
To enable the trigger	Set the PreOnline attribute in the service group definition to 1. You can set a local (per-system) value for the attribute to control behavior on each node in the cluster.
To disable the trigger	Set the PreOnline attribute in the service group definition to 0. You can set a local (per-system) value for the attribute to control behavior on each node in the cluster.

About the resadminwait event trigger

The following table describes the resadminwait event trigger:

Description

Invoked when a resource enters ADMIN_WAIT state.

When VCS sets a resource in the ADMIN_WAIT state, it invokes the resadminwait trigger according to the reason the resource entered the state.

See "Clearing resources in the ADMIN_WAIT state" on page 362.

This event trigger is non-configurable.

Usage

`-resadminwait system resource adminwait_reason`

system—represents the name of the system.

resource—represents the name of the faulted resource.

adminwait_reason—represents the reason the resource entered the ADMIN_WAIT state. Values range from 0-5:

0 = The offline function did not complete within the expected time.

1 = The offline function was ineffective.

2 = The online function did not complete within the expected time.

3 = The online function was ineffective.

4 = The resource was taken offline unexpectedly.

5 = The monitor function consistently failed to complete within the expected time.

About the resfault event trigger

The following table describes the resfault event trigger:

Description

Invoked on the system where a resource has faulted. Note that when a resource is faulted, resources within the upward path of the faulted resource are also brought down.

This event trigger is configurable.

To configure this trigger, you must define the following:

TriggerResFault: Set the attribute to 1 to invoke the trigger when a resource faults.

Usage

`-resfault triggertype system resource previous_state`

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the name of the system.

resource—represents the name of the faulted resource.

previous_state—represents the resource's previous state.

To enable the trigger

To invoke the trigger when a resource faults, set the TriggerResFault attribute to 1.

About the resnotoff event trigger

The following table describes the resnotoff event trigger:

Description

Invoked on the system if a resource in a service group does not go offline even after issuing the offline command to the resource.

This event trigger is configurable.

To configure this trigger, you must define the following:

Resource Name Define resources for which to invoke this trigger by entering their names in the following line in the script: `@resources = ("resource1", "resource2");`

If any of these resources do not go offline, the trigger is invoked with that resource name and system name as arguments to the script.

Usage

`-resnotoff triggertype system resource`

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the system on which the resource is not going offline.

resource—represents the name of the resource.

About the resrestart event trigger

The following table describes the resrestart event trigger.

Description

This trigger is invoked when a resource is restarted by an agent because resource faulted and RestartLimit was greater than 0.

Usage

`-resrestart triggertype system resource`

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the name of the system.

resource—represents the name of the resource.

To enable the trigger

This event trigger is not enabled by default. You must enable resrestart by setting the attribute TriggerResRestart to 1 in the main.cf file, or by issuing the command:

```
hagrp -modify service_group TriggerResRestart 1
```

However, the attribute is configurable at the resource level. To enable resrestart for a particular resource, you can set the attribute TriggerResRestart to 1 in the main.cf file or issue the command:

```
hares -modify resource TriggerResRestart 1
```

About the resstatechange event trigger

The following table describes the resstatechange event trigger:

Description

This trigger is invoked under the following conditions:

Resource goes from OFFLINE to ONLINE.

Resource goes from ONLINE to OFFLINE.

Resource goes from ONLINE to FAULTED.

Resource goes from FAULTED to OFFLINE. (When fault is cleared on non-persistent resource.)

Resource goes from FAULTED to ONLINE. (When faulted persistent resource goes online or faulted non-persistent resource is brought online outside VCS control.)

Resource is restarted by an agent because resource faulted and RestartLimit was greater than 0.

Warning: In later releases, you cannot use resstatechange to indicate restarting of a resource. Instead, use resrestart. See "About the resrestart event trigger" on page 455.

This event trigger is configurable.

Usage

`-resstatechange triggertype system resource previous_state new_state`

triggertype—represents whether trigger is custom (*triggertype*=0) or internal (*triggertype*=1).

For this trigger, *triggertype*=0.

system—represents the name of the system.

resource—represents the name of the resource.

previous_state—represents the resource's previous state.

new_state—represents the resource's new state.

To enable the trigger

This event trigger is not enabled by default. You must enable resstatechange by setting the attribute TriggerResStateChange to 1 in the main.cf file, or by issuing the command:

```
hagrp -modify service_group TriggerResStateChange 1
```

Use the resstatechange trigger carefully. For example, enabling this trigger for a service group with 100 resources means that 100 hatrigger processes and 100 resstatechange processes are fired each time the group is brought online or taken offline. Also, this is not a "wait-mode trigger. Specifically, VCS invokes the trigger and does not wait for trigger to return to continue operation.

However, the attribute is configurable at the resource level. To enable resstatechange for a particular resource, you can set the attribute TriggerResStateChange to 1 in the main.cf file or issue the command:

```
hares -modify resource TriggerResStateChange 1
```

About the sysoffline event trigger

The following table describes the sysoffline event trigger:

Description

Called from the lowest-numbered system in RUNNING state when a system is in jeopardy state or leaves the cluster.

This event trigger is non-configurable.

Usage

`-sysoffline system system_state`

system—represents the name of the system.

system_state—represents the value of the State attribute.

See "System states" on page 690.

About the sysup trigger

The following table describes the sysup event trigger:

Description	The sysup trigger is invoked when the first node joins the cluster.
Usage	`-sysup` *`triggertype system systemstate`* For this trigger, *triggertype* = 0. *system*— represents the system name. *systemstate*—represents the state of the system.

About the sysjoin trigger

The following table describes the sysjoin event trigger:

Description	The sysjoin trigger is invoked when a peer node joins the cluster.
Usage	`-sysjoin` *`triggertype system systemstate`* For this trigger, *triggertype* = 0. *system*—represents the system name. *systemstate*—represents the state of the system.

About the unable_to_restart_agent event trigger

The following table describes the unable_to_restart_agent event trigger:

Description	This trigger is invoked when an agent faults more than a predetermined number of times with in an hour. When this occurs, VCS gives up trying to restart the agent. VCS invokes this trigger on the node where the agent faults. You can use this trigger to notify the administrators that an agent has faulted, and that VCS is unable to restart the agent. The administrator can then take corrective action.
Usage	`-unable_to_restart_agent` *`system resource_type`* *system*—represents the name of the system. *resource_type*—represents the resource type associated with the agent.
To disable the trigger	Remove the files associated with the trigger from the *$VCS_HOME*/bin/triggers directory.

About the unable_to_restart_had event trigger

The following table describes the unable_to_restart_had event trigger:

Description	This event trigger is invoked by hashadow when hashadow cannot restart HAD on a system. If HAD fails to restart after six attempts, hashadow invokes the trigger on the system. The default behavior of the trigger is to reboot the system. However, service groups previously running on the system are autodisabled when hashadow fails to restart HAD. Before these service groups can be brought online elsewhere in the cluster, you must autoenable them on the system. To do so, customize the unable_to_restart_had trigger to remotely execute the following command from any node in the cluster where VCS is running: `hagrp -autoenable service_group -sys system` For example, if hashadow fails to restart HAD on *system1*, and if *group1* and *group2* were online on that system, a trigger customized in this manner would autoenable *group1* and *group2* on *system1* before rebooting. Autoenabling *group1* and *group2* on *system1* enables these two service groups to come online on another system when the trigger reboots *system1*. This event trigger is non-configurable.
Usage	`-unable_to_restart_had` This trigger has no arguments.

About the violation event trigger

The following table describes the violation event trigger:

Description	This trigger is invoked only on the system that caused the concurrency violation. Specifically, it takes the service group offline on the system where the trigger was invoked. Note that this trigger applies to failover groups only. The default trigger takes the service group offline on the system that caused the concurrency violation. This event trigger is internal and non-configurable.
Usage	`-violation system service_group` *system*—represents the name of the system. *service_group*—represents the name of the service group that was fully or partially online.

Chapter 14

Virtual Business Services

This chapter includes the following topics:

- About Virtual Business Services
- Features of Virtual Business Services
- Sample virtual business service configuration
- About choosing between VCS and VBS level dependencies

About Virtual Business Services

The Virtual Business Services feature provides visualization, orchestration, and reduced frequency and duration of service disruptions for multi-tier business applications running on heterogeneous operating systems and virtualization technologies. A virtual business service represents the multi-tier application as a consolidated entity that helps you manage operations for a business service. It builds on the high availability and disaster recovery provided for the individual tiers by Veritas InfoScale products such as Cluster Server.

Application components that are managed by Cluster Server or Microsoft Failover Clustering can be actively managed through a virtual business service.

You can use the Veritas InfoScale Operations Manager Management Server console to create, configure, and manage virtual business services.

Features of Virtual Business Services

The following VBS operations are supported:

- Start Virtual Business Services from the Veritas InfoScale Operations Manager console: When a virtual business service starts, its associated service groups are brought online.

- Stop Virtual Business Services from the Veritas InfoScale Operations Manager console: When a virtual business service stops, its associated service groups are taken offline.
- Applications that are under the control of ApplicationHAcan be part of a virtual business service. ApplicationHAenables starting, stopping, and monitoring of an application within a virtual machine. If applications are hosted on VMware virtual machines, you can configure the virtual machines to automatically start or stop when you start or stop the virtual business service, provided the vCenter for those virtual machines has been configured in Veritas InfoScale Operations Manager.
- Define dependencies between service groups within a virtual business service: The dependencies define the order in which service groups are brought online and taken offline. Setting the correct order of service group dependency is critical to achieve business continuity and high availability. You can define the dependency types to control how a tier reacts to high availability events in the underlying tier. The configured reaction could be execution of a predefined policy or a custom script.
- Manage the virtual business service from Veritas InfoScale Operations Manager or from the clusters participating in the virtual business service.
- Recover the entire virtual business service to a remote site when a disaster occurs.

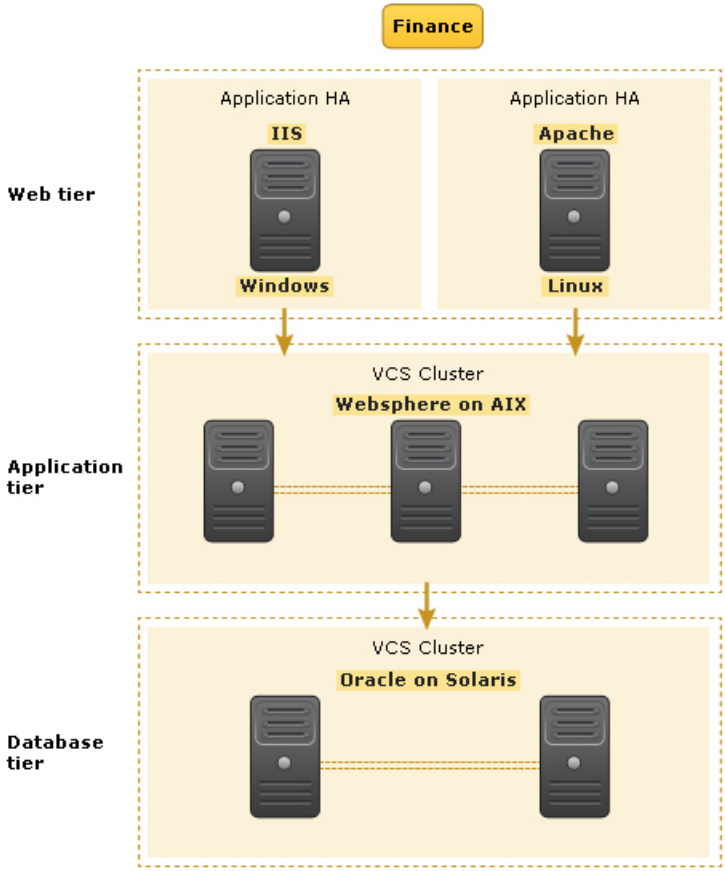
Sample virtual business service configuration

This section provides a sample virtual business service configuration comprising a multi-tier application. Figure 14-1 shows a Finance application that is dependent on components that run on three different operating systems and on three different clusters.

- Databases such as Oracle running on Solaris operating systems form the database tier.
- Middleware applications such as WebSphere running on AIX operating systems form the middle tier.
- Web applications such as Apache and IIS running on Windows and Linux virtual machines form the Web tier.

  Each tier can have its own high availability mechanism. For example, you can use Cluster Server for the databases and middleware applications for the Web servers.

Figure 14-1 Sample virtual business service configuration



Each time you start the Finance business application, typically you need to bring the components online in the following order – Oracle database, WebSphere, Apache and IIS. In addition, you must bring the virtual machines online before you start the Web tier. To stop the Finance application, you must take the components offline in the reverse order. From the business perspective, the Finance service is unavailable if any of the tiers becomes unavailable.

When you configure the Finance application as a virtual business service, you can specify that the Oracle database must start first, followed by WebSphere and the Web servers. The reverse order automatically applies when you stop the virtual business service. When you start or stop the virtual business service, the components of the service are started or stopped in the defined order.

For more information about Virtual Business Services, refer to the *Virtual Business Service–Availability User's Guide*.

About choosing between VCS and VBS level dependencies

While a VBS typically has service groups across clusters, you can also have two or more service groups from a single cluster participate in a given VBS. In this case, you can choose to configure either VCS dependency or VBS dependency between these intra-cluster service groups. Veritas recommends use of VCS dependency in this case. However, based on your requirement about the type of fault policy desired, you can choose appropriately. For information about VBS fault policies, refer to the *Virtual Business Service–Availability User's Guide*.

See "Service group dependency configurations" on page 411.

Section 5

Veritas High Availability Configuration wizard

- Chapter 15. Introducing the Veritas High Availability Configuration wizard
- Chapter 16. Administering application monitoring from the Veritas High Availability view

Chapter 15

Introducing the Veritas High Availability Configuration wizard

This chapter includes the following topics:

- About the Veritas High Availability Configuration wizard
- Launching the Veritas High Availability Configuration wizard
- Typical VCS cluster configuration in a VMware virtual environment
- Typical VCS cluster configuration in a physical environment

About the Veritas High Availability Configuration wizard

The Veritas High Availability Configuration wizard can be used to configure applications that use active-passive storage. Active-passive storage can access only one node at a time. Apart from configuring application availability, the wizard also sets up other components (for example: networking, storage) required for successful application monitoring.

Before you configure an application for monitoring, you must configure a cluster. You can use the VCS Cluster Configuration wizard to configure a cluster. Refer to the *VCS Installation Guide* for the steps to configure a cluster.

You can use the Veritas High Availability Configuration wizard to configure the applications listed in Table 15-1.

Table 15-1 Supported applications

Application	Platform/Virtualization technology
Generic application	■ AIX ■ Linux ■ Solaris
Oracle	VMware on Linux
SAP WebAS	VMware on Linux
WebSphere MQ	VMware on Linux

See "Launching the Veritas High Availability Configuration wizard" on page 466.

For information on the procedure to configure a particular application, refer to the application-specific agent guide.

Launching the Veritas High Availability Configuration wizard

In a physical Linux environment, VMware virtual environment, logical domain, or LPAR, you can launch the Veritas High Availability Configuration wizard using:

- Client: To launch the wizard from the Client
- A browser window: To launch the wizard from a browser window

In addition, in a Linux environment, you can also launch the wizard using:

- haappwizard utility: To launch the wizard using the haappwizard utility

In addition, in a VMware virtual environment, you can also launch the wizard using:

- VMware vSphere Client: To launch the wizard from the VMware vSphere Client

To launch the wizard from the Client

1. Log in to the Management Server console.
2. In the home page, click the **Availability** icon from the list of perspectives.
3. In the **Data Center** tree under the **Manage** pane, locate the cluster.
4. Navigate to the **Systems** object under the cluster.
5. Locate the system on which the application is running or application prerequisites are met.
6. Right-click on the system, and then click **Configure Application**.

To launch the wizard from a browser window

1. Open a browser window and enter the following URL:

   `https://system_name_or_IP:5634/vcs/admin/application_health.html`

   system_name_or_IP is the system name or IP address of the system on which you want to configure application monitoring.

2. In the Authentication dialog box, enter the username and password of the user who has administrative privileges.

3. Depending on your setup, use one of the following options to launch the wizard:

   - If you have not configured a cluster, click the **Configure a VCS Cluster** link.
   - If you have already configured a cluster, click **Actions > Configure Application for High Availability** or the **Configure Application for High Availability** link.
   - If you have already configured a cluster and configured an application for monitoring, click **Actions > Configure Application for High Availability**.

Note: In the Storage Foundation and High Availability (SFHA) 6.2 release and later, the `haappwizard` utility has been deprecated.

To launch the wizard using the haappwizard utility

The haappwizard utility allows you to launch the Veritas High Availability Configuration wizard. The utility is part of the product package and is installed in the /opt/VRTSvcs/bin directory.

◆ Enter the following command to launch the Veritas High Availability Configuration wizard:

```
happwizard [-hostname host_name] [-browser browser_name] [-help]
```

The following table describes the options of the happwizard utility:

Table 15-2 Options of the happwizard utility

–hostname	Allows you to specify the host name or IP address of the system from which you want to launch the Veritas High Availability Configuration wizard. If you do not specify a host name or IP address, the Veritas High Availability Configuration wizard is launched on the local host.

Table 15-2 Options of the happwizard utility *(continued)*

-browser	Allows you to specify the browser name. The supported browsers are Internet Explorer and Firefox. For example, enter `iexplore` for Internet Explorer and `firefox` for Firefox. **Note:** The value is case-insensitive.
-help	Displays the command usage.

To launch the wizard from the VMware vSphere Client

1. Launch the VMware vSphere Client and connect to the VMware vCenter Server that hosts the virtual machine.
2. From the vSphere Client's Inventory view in the left pane, select the virtual machine where you want to configure application monitoring.
3. Skip this step if you have already configured single sign-on during guest installation.

   Select the Veritas High Availability tab and in the Veritas High Availability View page, specify the credentials of a user account that has administrative privileges on the virtual machine and click **Configure**.

   The Veritas High Availability console sets up a permanent authentication for the user account on that virtual machine.
4. Depending on your setup, use one of the following options to launch the wizard:
   - If you have not configured a cluster, click the **Configure a VCS Cluster** link.
   - If you have already configured a cluster, click **Actions > Configure Application for High Availability** or the **Configure Application for High Availability** link.
   - If you have already configured a cluster and configured an application for monitoring and to configure another application for monitoring, click **Actions > Configure Application for High Availability**.

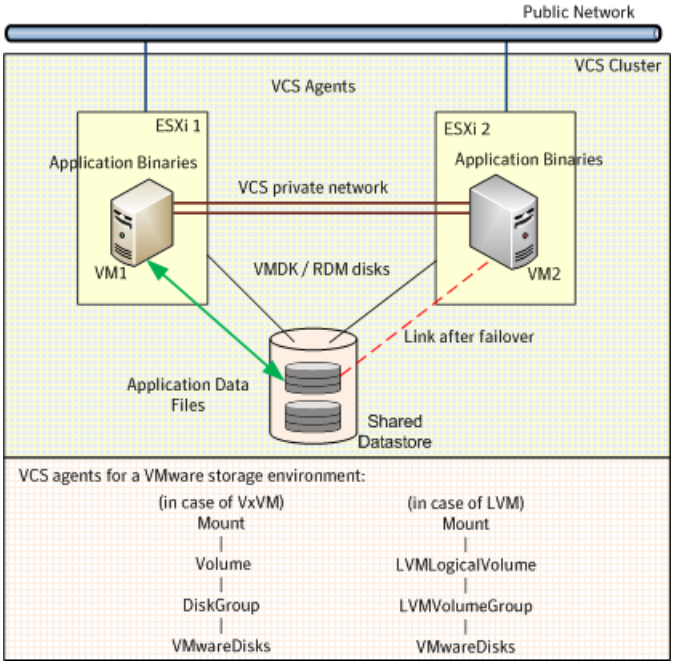
Typical VCS cluster configuration in a VMware virtual environment

A typical VCS cluster configuration in a VMware virtual environment involves two or more virtual machines. The virtual machine on which the application is active, accesses a non-shared VMware VMDK, or RDM disk that resides on a VMware datastore.

The virtual machines involved in the VCS cluster configuration may belong to a single ESX/ESXi host or could reside on separate ESX/ESXi hosts. If the virtual machines reside on separate ESX/ESXi hosts, the datastore on which the VMware VMDK or RDM disks (on which the application data is stored) reside must be accessible to each of these ESX/ESXi hosts.

The application binaries are installed on the virtual machines and the data files are installed on the non-shared VMDK or RDM disk. The VCS agents monitor the application components and services, and the storage and network components that the application uses.

Figure 15-1 Typical generic applications cluster configuration in a VMware virtual environment



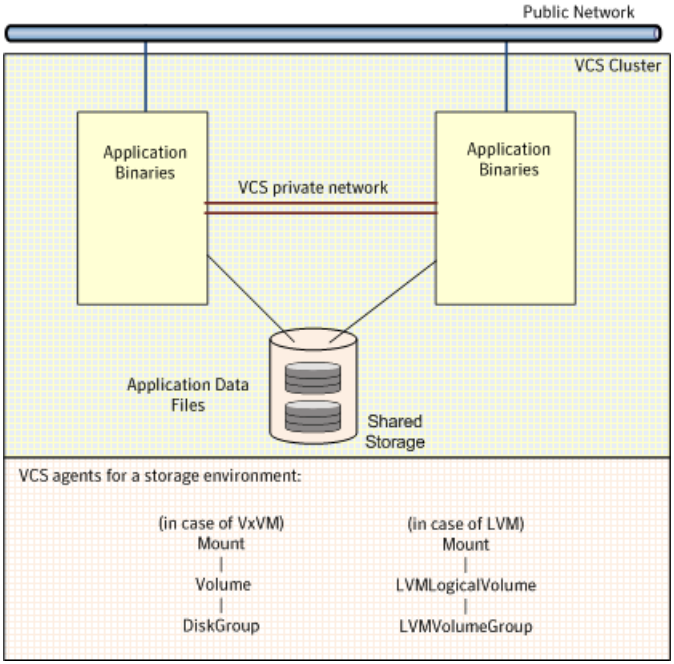
During a failover, the VCS storage agents move the VMware disks to the new system. The VCS network agents bring the network components online and the application specific agents then start application services on the new system.

Typical VCS cluster configuration in a physical environment

A typical VCS cluster configuration in a physical environment involves two or more physical machines.

The application binaries are installed on a local or shared storage and the data files should be installed on a shared storage. The VCS agents monitor the application components and services, and the storage and network components that the application uses.

Figure 15-2 Typical generic applications cluster configuration in a physical environment



During a fault, the VCS storage agents fail over the storage to a new system. The VCS network agents bring the network components online and the application-specific agents then start application services on the new system.

Chapter 16

Administering application monitoring from the Veritas High Availability view

This chapter includes the following topics:

- Administering application monitoring from the Veritas High Availability view
- Administering application monitoring settings

Administering application monitoring from the Veritas High Availability view

The Veritas High Availability view is a Web-based graphic user interface that you can use to administer application availability with Cluster Server (VCS).

This interface supports physical and virtual systems running Linux, Solaris, and AIX operating systems. The interface supports virtual environments such as IBM PowerVM, Oracle VM Server for SPARC, Red Hat Enterprise Virtualization (RHEV), Kernel-based Virtual Machine (KVM), and VMware.

You can launch the Veritas High Availability view from an Internet browser, by using the following URL:

https://*SystemNameorIP*:5634/vcs/admin/application_health.html?priv=ADMIN
where *SystemNameorIP* is the system name or the IP address of the physical or virtual system from where you want to access the tab.

In a VMware virtual environment, you can embed the Veritas High Availability view as a tab in the vSphere Client menu (both the desktop and Web versions).

You can perform tasks such as start, stop, or fail over an application, without the need for specialized training in VCS commands, operating systems, or virtualization technologies. You can also perform advanced tasks such as VCS cluster configuration and application (high availability) configuration from this view, by using simple wizard-based steps.

Use the Veritas High Availability view to perform the following tasks:

- To configure a VCS cluster
- To add a system to a VCS cluster
- To configure and unconfigure application monitoring
- To unconfigure the VCS cluster
- To start and stop configured applications
- To add and remove failover systems
- To enter and exit maintenance mode
- To switch an application
- To determine the state of an application (components)
- To clear Fault state
- To resolve a held-up operation
- To modify application monitoring settings
- To view application dependency
- To view component dependency

Understanding the Veritas High Availability view

The Veritas High Availability view displays the consolidated health information for applications running in a Cluster Server (VCS) cluster. The cluster may include one or more systems.

The Veritas High Availability tab displays application information for the entire VCS cluster, not just the local system.

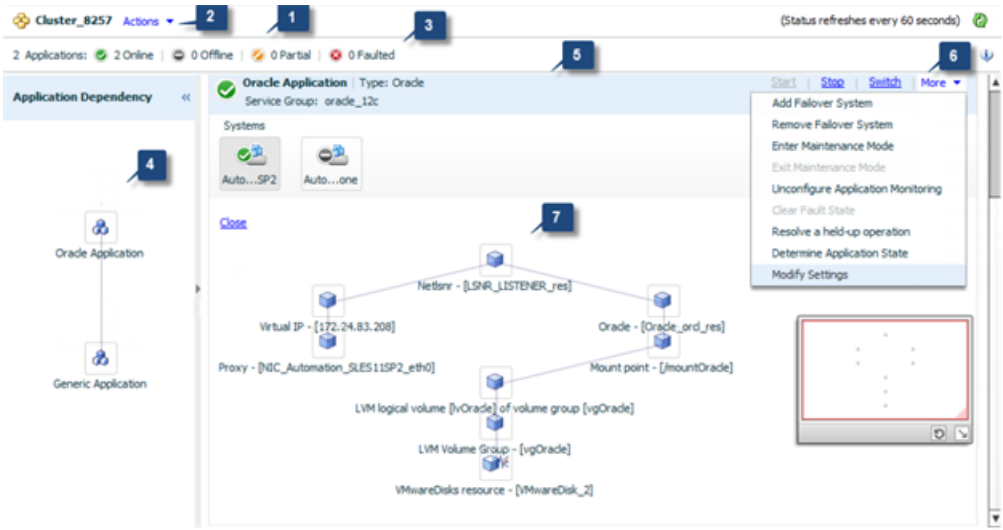
Note: If the local system is not part of any VCS cluster, then the Veritas Application High Availability view displays only the following link: **Configure a VCS cluster**.

If you are yet to configure an application for monitoring in a cluster of which the local system is a member, then the Veritas Application High Availability view displays only the following link: **Configure an application for high availability**.

The Veritas High Availability view uses icons, color coding, dependency graphs, and tool tips to report the detailed status of an application.

The Veritas High Availability view displays complex applications, in terms of multiple interdependent instances of that application. Such instances represent component groups (also known as service groups) of that application. Each service group in turn includes several critical components (resources) of the application.

The following figure shows the Veritas High Availability view, with one instance of Oracle Database and one instance of a generic application configured for high availability in a two-node VCS cluster:



1. Title bar

2. Actions menu

3. Aggregate Status Bar

4. Application dependency graph

5. Application table

6. Application-specific task menu

7. Component dependency graph

The Veritas High Availability view includes the following parts:

- Title bar: Displays the name of the VCS cluster, the Actions menu, the Refresh icon, the Alert icon. Note that the Alert icon appears only if the Veritas High Availability view fails to display a system, or displays stale data.
- Actions menu: Includes a drop-down list of operations that you can perform with effect across the cluster. These include: Configuring a cluster, Configuring an application for high availability; Unconfigure all applications; and Unconfigure VCS cluster.
- Aggregate status bar: Displays a summary of applications running in the cluster. This includes the total number of applications, and a breakdown of the number of applications in Online, Offline, Partial, and Faulted states.
- Application dependency graph: Illustrates the order in which the applications or application instances, must start or stop.

  If an application must start first for another application to successfully start, the earlier application appears at a lower level in the graph. A line connects the two applications to indicate the dependency. If no such dependency exists, all applications appear in a single horizontal line.
- Application table: Displays a list of all applications configured in the VCS cluster that is associated with the local system.

  Each application is listed in a separate row. Each row displays the systems where the application is configured for monitoring.

  The title bar of each row displays the following entities to identify the application or application instance (service group):
  - Display name of the application (for example, Payroll application)
  - Type of application (for example, Custom)
  - Service group name
- Application-specific task menu: Appears in each application-specific row of the application table. The menu includes application-specific tasks such as Start, Stop, Switch, and a dropdown list of more tasks. The More dropdown list includes tasks such as Add a failover system, and Remove a failover system.
- Component dependency graph: Illustrates the order in which application components (resources) must start or stop for the related application or application instance to respectively start or stop. The component dependency graph by default does not appear in the application table. To view the component dependency graph for an application, you must click a system on which the application is running.

  The track pad, at the right-bottom corner helps you navigate through component dependency graphs.

  If you do not want to view the component dependency graph, in the top left corner of the application row, click **Close**.

To view the status of configured applications

In the application dependency graph, click the application for which you want to view the status. If the appropriate row is not already visible, the application table automatically scrolls to the appropriate row. The row displays the state of the application for each configured failover system in the cluster for that application.

If you click any system in the row, a component dependency graph appears. The graph uses symbols, color code, and tool tips to display the health of each application component. Roll the mouse over a system or component to see its health details.

The health of each application/application component on the selected system is displayed in terms of the following states:

Table 16-1 Application states

State	Description
Online	Indicates that the configured application or application components are running on the system. If the application is offline on at least one other failover system, an alert appears next to the application name.
Offline	Indicates that the configured application or its components are not running on the system.
Partial	Indicates that either the application or its components are being started on the system or VCS was unable to start one or more of the configured components If the application is offline on at least one other failover system, an alert appears next to the application name.
Faulted	Indicates that the configured application or its components have unexpectedly stopped running.

To configure or unconfigure application monitoring

Use the Veritas High Availability view to configure or un-configure an application for monitoring in a cluster under Cluster Server (VCS) control.

The view provides you with specific links to perform the following configuration tasks:

- Configure a VCS cluster:
  If the local system is not part of a VCS cluster, the Veritas High Availability view appears blank, except for the link **Configure a VCS cluster**.

Before you can configure application monitoring, you must click the **Configure a VCS cluster** link to launch the VCS Cluster Configuration wizard. The wizard helps you configure a VCS cluster, where the local system is by default a cluster system.

- Configure the first application for monitoring in a VCS cluster:
  If you have not configured any application for monitoring in the cluster, the Veritas High Availability view appears blank except for the link **Configure an application for high availability**.
  Click the link to launch the Veritas High Availability Configuration Wizard. Use the wizard to configure application monitoring.
  Also, in applications where the Veritas High Availability Configuration Wizard is supported, for detailed wizard-based steps, see the application-specific VCS agent configuration guide. For custom applications, see the *Cluster Server Generic Application Agent Configuration Guide*.
- Add a system to the VCS cluster:
  Click **Actions** > **Add a System to VCS Cluster** to add a system to a VCS cluster where the local system is a cluster member. Adding a system to the cluster does not automatically add the system as a failover system for any configured application. To add a system as a failover system, see (add a cross-reference to 'To add or remove a failover system'.
- Configure an application or add an application to the existing application monitoring configuration:
  Click **Actions** > **Configure an application for high availability** to launch the Veritas High Availability Application Monitoring Configuration Wizard. Use the wizard to configure application monitoring.
- Unconfigure monitoring of an application:
  In the appropriate row of the application table, click **More** > **Unconfigure Application Monitoring** to delete the application monitoring configuration from the VCS.
  Note that this step does not remove VCS from the system or the cluster, this step only removes the monitoring configuration for that application.
  Also, to unconfigure monitoring for an application, you can perform one of the following procedures: unconfigure monitoring of all applications, or unconfigure VCS cluster.
- Unconfigure monitoring of all applications:
  Click **Actions** > **Unconfigure all applications**. This step deletes the monitoring configuration for all applications configured in the cluster.
- Unconfigure VCS cluster:
  Click **Actions** > **Unconfigure VCS cluster**. This step stops the VCS cluster, removes VCS cluster configuration, and unconfigures application monitoring.

To start or stop applications

Use the following options on the Veritas High Availability view to control the status of the configured application and the associated components or component groups (application instances).

Note that the **Start** and **Stop** links are dimmed in the following cases:

- If you have not configured any associated components or component groups (resources or service groups) for monitoring
- If the application is in maintenance mode
- If no system exists in the cluster, where the application is not already started or stopped as required.

To start an application

1. In the appropriate row of the application table, click **Start**.
2. If the application (service group) is of the failover type, on the Start Application panel, click **Any system**. VCS uses pre-defined policies to decide the system where to start the application.

   If the application (service group) is of the parallel type, on the Start Application panel, click **All systems**. VCS starts the application on all required systems, where the service group is configured.

   Note: Based on service group type, either the Any system or the All Systems link automatically appears.

   To learn more about policies, and parallel and failover service groups, see the *Cluster Server Administrator's Guide*.

   If you want to specify the system where you want to start the application, click **User selected system**, and then click the appropriate system.
3. If the application that you want to start requires other applications or component groups (service groups) to start in a specific order, then check the **Start the dependent components in order** check box, and then click **OK**.

To stop an application

1. In the appropriate row of the application table, click **Stop**.
2. If the application (service group) is of the failover type, in the Stop Application Panel, click **Any system**. VCS selects the appropriate system to stop the application.

   If the application (service group) is of the parallel type, in the Stop Application Panel click **All systems**. VCS stops the application on all configured systems.

   Note: Based on service group type, either the Any system or the All Systems link automatically appears.

   To learn more about parallel and failover service groups, see the *Cluster Server Administrator's Guide*.

   If you want to specify the system where you want to stop the application, click **User selected system**, and then click the appropriate system.
3. If the application that you want to stop requires other applications or component groups (service groups) to stop in a specific order, then check the **Stop the dependent components in order** check box, and then click **OK**.

To suspend or resume application monitoring

After configuring application monitoring you may want to perform routine maintenance tasks on those applications. These tasks may or may not involve stopping the application but may temporarily affect the state of the applications and its dependent components. If there is any change to the application status, Cluster Server (VCS) may try to restore the application state. This may potentially affect the maintenance tasks that you intend to perform on those applications.

The **Enter Maintenance Mode** link is automatically dimmed if the application is already in maintenance mode. Conversely, if the application is not in maintenance mode, the **Exit Maintenance Mode** link is dimmed.

The Veritas High Availability tab provides the following options:

To enter maintenance mode

1. In the appropriate row, click **More**> **Enter Maintenance Mode**.

   During the time the monitoring is suspended, Veritas high availability solutions do not monitor the state of the application and its dependent components. The Veritas High Availability view does not display the current status of the application. If there is any failure in the application or its components, VCS takes no action.

2. While in maintenance mode, if a virtual machine restarts, if you want application monitoring to remain in maintenance mode, then in the Enter Maintenance Mode panel, check the **Suspend the application availability even after reboot** check box, and then click **OK** to enter maintenance mode.

To exit the maintenance mode

1. In the appropriate row, click **More**> **Exit Maintenance Mode**, and then click **OK** to exit maintenance mode.

2. Click the Refresh icon in the top right corner of the Veritas High Availability view, to confirm that the application is no longer in maintenance mode.

To switch an application to another system

If you want to gracefully stop an application on one system and start it on another system in the same cluster, you must use the Switch link. You can switch the application only to a system where it is not running.

Note that the Switch link is dimmed in the following cases:

- If you have not configured any application components for monitoring
- If you have not specified any failover system for the selected application
- If the application is in maintenance mode
- If no system exists in the cluster, where the application can be switched
- If the application is not in online/partial state on even a single system in the cluster

To switch an application

1. In the appropriate row of the application table, click **Switch**.
2. If you want VCS to decide to which system the application must switch, based on policies, then in the Switch Application panel, click **Any system**, and then click **OK**.

   To learn more about policies, see the *Cluster Server Administrator's Guide*.

   If you want to specify the system where you want to switch the application, click **User selected system**, and then click the appropriate system, and then click **OK**.

   VCS stops the application on the system where the application is running, and starts it on the system you specified.

To add or remove a failover system

Each row in the application table displays the status of an application on systems that are part of a VCS cluster. The displayed system/s either form a single-system Cluster Server (VCS) cluster with application restart configured as a high-availability measure, or a multi-system VCS cluster with application failover configured. In the displayed cluster, you can add a new system as a failover system for the configured application.

The system must fulfill the following conditions:

- The system is not part of any other VCS cluster.
- The system has at least two network adapters.
- The host name of the system must be resolvable through the DNS server or, locally, using `/etc/hosts` file entries.
- The required ports are not blocked by a firewall.
- The application is installed identically on all the systems, including the proposed new system.

To add a failover system, perform the following steps:

Note: The following procedure describes generic steps to add a failover system. The wizard automatically populates values for initially configured systems in some fields. These values are not editable.

To add a failover system

1. In the appropriate row of the application table, click **More** > **Add Failover System**.
2. Review the instructions on the welcome page of the Veritas High Availability Configuration Wizard, and click **Next**.
3. If you want to add a system from the Cluster systems list to the Application failover targets list, on the Configuration Inputs panel, select the system in the Cluster systems list. Use the Edit icon to specify an administrative user account on the system. You can then move the required system from the Cluster system list to the Application failover targets list. Use the up and down arrow keys to set the order of systems in which VCS agent must failover applications.

   If you want to specify a failover system that is not an existing cluster node, on the Configuration Inputs panel, click **Add System**, and in the Add System dialog box, specify the following details:

System Name or IP address	Specify the name or IP address of the system that you want to add to the VCS cluster.
User name	Specify the user name with administrative privileges on the system. If you want to specify the same user account on all systems that you want to add, check the **Use the specified user account on all systems** box.
Password	Specify the password for the account you specified.
Use the specified user account on all systems	Click this check box to use the specified user credentials on all the cluster systems.

   The wizard validates the details, and the system then appears in the Application failover target list.
4. If you are adding a failover system from the existing VCS cluster, the Network Details panel does not appear.

   If you are adding a new failover system to the existing cluster, on the Network Details panel, review the networking parameters used by existing failover systems. Appropriately modify the following parameters for the new failover system.

   Note: The wizard automatically populates the networking protocol (UDP or Ethernet) used by the existing failover systems for Low Latency Transport communication. You cannot modify these settings.

- To configure links over ethernet, select the adapter for each network communication link. You must select a different network adapter for each communication link.
- To configure links over UDP, specify the required details for each communication link.

Network Adapter	Select a network adapter for the communication links. You must select a different network adapter for each communication link. Veritas recommends that one of the network adapters must be a public adapter and the VCS cluster communication link using this adapter is assigned a low priority. **Note:** Do not select the teamed network adapter or the independently listed adapters that are a part of teamed NIC.
IP Address	Select the IP address to be used for cluster communication over the specified UDP port.
Port	Specify a unique port number for each link. You can use ports in the range 49152 to 65535. The specified port for a link is used for all the cluster systems on that link.
Subnet mask	Displays the subnet mask to which the specified IP belongs.

5 If a virtual IP is not configured as part of your application monitoring configuration, the Virtual Network Details page is not displayed. Else, on the Virtual Network Details panel, review the following networking parameters that the failover system must use, and specify the NIC:

Virtual IP address	Specifies a unique virtual IP address.
Subnet mask	Specifies the subnet mask to which the IP address belongs.
NIC	For each newly added system, specify the network adaptor that must host the specified virtual IP.

6 On the Configuration Summary panel, review the VCS cluster configuration summary, and then click **Next** to proceed with the configuration.

7 On the Implementation panel, the wizard adds the specified system to the VCS cluster, if it is not already a part. It then adds the system to the list of failover targets. The wizard displays a progress report of each task.

- If the wizard displays an error, click **View Logs** to review the error description, troubleshoot the error, and re-run the wizard from the Veritas High Availability view.
- Click **Next**.

8 On the Finish panel, click **Finish**. This completes the procedure for adding a failover system. You can view the system in the appropriate row of the application table.

Similarly you can also remove a system from the list of application failover targets.

Note: You cannot remove a failover system if an application is online or partially online on the system.

To remove a failover system

1 In the appropriate row of the application table, click **More** > **Remove Failover System**.

2 On the Remove Failover System panel, click the system that you want to remove from the monitoring configuration, and then click **OK**.

Note: This procedure only removes the system from the list of failover target systems, not from the VCS cluster. To remove a system from the cluster, use VCS commands. For details, see the *VCS Administrator's Guide*.

To clear Fault state

When you fix an application fault on a system, you must further clear the application Faulted state on that system. Unless you clear the Faulted state, VCS cannot failover the application on that system.

You can use the Veritas High Availability view to clear this faulted state at the level of a configured application component (resource).

The Clear Fault link is automatically dimmed if there is no faulted system in the cluster.

To clear Fault state

1 In the appropriate row of the application table, click **More** > **Clear Fault state**.

2 In the Clear Fault State panel, click the system where you want to clear the Faulted status of a component, and then click **OK**.

To resolve a held-up operation

When you try to start or stop an application, in some cases, the start or stop operation may get held-up mid course. This may be due to VCS detecting an incorrect internal state of an application component. You can resolve this issue by using the resolve a held-up operation link. When you click the link, VCS appropriately resets the internal state of any held-up application component. This process prepares the ground for you to retry the original start or stop operation, or initiate another operation.

To resolve a held-up operation

1. In the appropriate row of the application table, click **More** > **Resolve a held-up operation**.
2. In the Resolve a held-up operation panel, click the system where you want to resolve the held-up operation, and then click **OK**.

To determine application state

The Veritas High Availability view displays the consolidated health information of all applications configured for monitoring in a VCS cluster. The application health information is automatically refreshed every 60 seconds.

If you do not want to wait for the automatic refresh, you can instantaneously determine the state of an application by performing the following steps:

To determine application state

1. In the appropriate row of the Application table, click **More** > **Determine Application State**.
2. In the Determine Application State panel, select a system and then click **OK**.

Note: You can also select multiple systems, and then click **OK**.

To remove all monitoring configurations

To discontinue all existing application monitoring in a VCS cluster, perform the following step:

- On the Veritas High Availability view, in the Title bar, click **Actions** > **Unconfigure all applications**. When a confirmation message appears, click **OK**.

To remove VCS cluster configurations

If you want to create a different VCS cluster, say with new systems, a different LLT protocol, or secure communication mode, you may want to remove existing VCS cluster configurations. To remove VCS cluster configurations, perform the following steps:

Note: The following steps deletes all cluster configurations, (including networking and storage configurations), as well as application-monitoring configurations.

- On the Title bar of the Veritas High Availability view, click **Actions** >**Unconfigure VCS cluster**.
- In the Unconfigure VCS Cluster panel, review the Cluster Name and Cluster ID, and specify the User name and Password of the Cluster administrator, and then click **OK**.

Administering application monitoring settings

The Veritas High Availability tab lets you define and modify settings that control application monitoring with Cluster Server(VCS). You can define the settings on a per application basis. The settings apply to all systems in a VCS cluster where the application is configured for monitoring.

The following settings are available:

- App.StartStopTimeout:
  This setting defines the number of seconds that VCS must wait for the application to start or stop, after initiating the operation.
  When you click the Start Application or Stop Application links in the Veritas High Availability tab, VCS initiates an application start or stop, respectively. If the application does not respond within the stipulated time, the tab displays an error. A delay in the application response does not indicate that the application or its dependent component has faulted. Parameters such as workload, system performance, and network bandwidth may affect the application response. VCS continues to wait for the application response even after the timeout interval elapses.
  If the application fails to start or stop, VCS takes the necessary action depending on the other configured remedial actions.
  You can set a AppStartStopTimeout value between 0 and 600 seconds. The default value is 30 seconds.
- App.RestartAttempts:

This setting defines the number of times that VCS must try to restart a failed application. If an application fails to start within the specified number of attempts, VCS fails over the application to a configured failover system. You can set a AppRestartAttempts value between 1 and 6. The default value is 1.

- App.DisplayName:
  This setting lets you specify an easy-to-use display name for a configured application. For example, Payroll application. VCS may internally use a different service group name to uniquely identify the same application. However, the service group name, for example OraSG2, may not be intuitive to understand, or easy to parse while navigating the application table. Moreover, once configured, you cannot edit the service group name. In contrast, VCS allows you to modify the display name as required. Note that the Veritas High Availability tab displays both the display name and the service group name associated with the configured application.

To modify the application monitoring configuration settings

1. Launch the vSphere Client and from the inventory pane on the left, select the virtual machine that is a part of the VCS cluster where you have configured application monitoring.
2. Click **Veritas High Availability**, to view the Veritas High Availability tab.
3. In the appropriate row of the application table, click **More** > **Modify Settings**.
4. In the Modify Settings panel, click to highlight the setting that you want to modify.
5. In the Value column, enter the appropriate value, and then click **OK**.

Section 6

Cluster configurations for disaster recovery

- Chapter 17. Connecting clusters–Creating global clusters
- Chapter 18. Administering global clusters from the command line
- Chapter 19. Setting up replicated data clusters
- Chapter 20. Setting up campus clusters

Chapter 17

Connecting clusters–Creating global clusters

This chapter includes the following topics:

- How VCS global clusters work
- VCS global clusters: The building blocks
- Prerequisites for global clusters
- About planning to set up global clusters
- Setting up a global cluster
- About IPv6 support with global clusters
- About cluster faults
- About setting up a disaster recovery fire drill
- Multi-tiered application support using the RemoteGroup agent in a global environment
- Test scenario for a multi-tiered environment

How VCS global clusters work

Local clustering provides local failover for each site or building. But, these configurations do not provide protection against large-scale disasters such as major

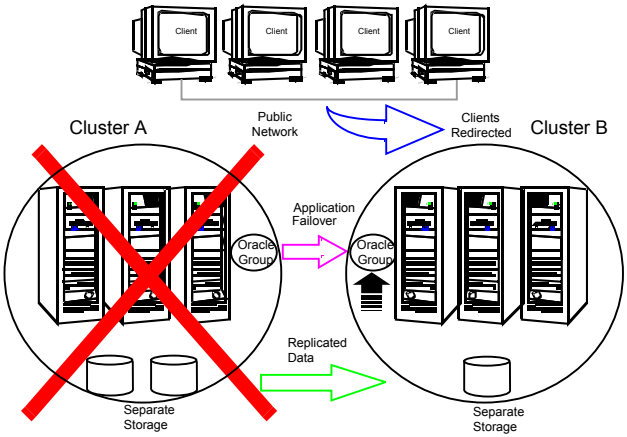
floods, hurricanes, and earthquakes that cause outages for an entire city or region. The entire cluster could be affected by an outage.

In such situations, VCS global clusters ensure data availability by migrating applications to remote clusters located considerable distances apart.

Let us take the example of an Oracle database configured in a VCS global cluster. Oracle is installed and configured in both clusters. Oracle data is located on shared disks within each cluster and is replicated across clusters to ensure data concurrency. The Oracle service group is online on a system in cluster A and is configured to fail over globally, on clusters A and B.

Figure 17-1 shows a sample global cluster setup.

Figure 17-1 Sample global cluster setup



VCS continuously monitors and communicates events between clusters. Inter-cluster communication ensures that the global cluster is aware of the state of the service groups that are configured in the global cluster at all times.

In the event of a system or application failure, VCS fails over the Oracle service group to another system in the same cluster. If the entire cluster fails, VCS fails over the service group to the remote cluster, which is part of the global cluster. VCS also redirects clients once the application is online on the new location.

VCS global clusters: The building blocks

VCS extends clustering concepts to wide-area high availability and disaster recovery with the following:

- Remote cluster objects

See " Visualization of remote cluster objects" on page 490.

- Global service groups
  See "About global service groups" on page 490.
- Global cluster management
  See "About global cluster management" on page 491.
- Serialization
  See "About serialization–The Authority attribute" on page 492.
- Resiliency and right of way
  See "About resiliency and "Right of way"" on page 493.
- VCS agents to manage wide-area failover
  See " VCS agents to manage wide-area failover" on page 493.
- Split-brain in two-cluster global clusters
  See "About the Steward process: Split-brain in two-cluster global clusters" on page 496.
- Secure communication
  See " Secure communication in global clusters" on page 497.

Visualization of remote cluster objects

VCS enables you to visualize remote cluster objects using any of the supported components that are used to administer VCS such as VCS CLI and Veritas InfoScale Operations Manager

See " Components for administering VCS" on page 49.

You can define remote clusters in your configuration file, `main.cf`. The Remote Cluster Configuration wizard provides an easy interface to do so. The wizard updates the `main.cf` files of all connected clusters with the required configuration changes.

About global service groups

A global service group is a regular VCS group with additional properties to enable wide-area failover. The global service group attribute ClusterList defines the list of clusters to which the group can fail over. The service group must be configured on all participating clusters and must have the same name on each cluster. The Global Group Configuration Wizard provides an easy interface to configure global groups.

VCS agents manage the replication during cross-cluster failover.

See " VCS agents to manage wide-area failover" on page 493.

About global cluster management

VCS enables you to perform operations (online, offline, switch) on global service groups from any system in any cluster. You must log on with adequate privileges for cluster operations.

See "User privileges for cross-cluster operations" on page 89.

You can bring service groups online or switch them to any system in any cluster. If you do not specify a target system, VCS uses the FailOverPolicy to determine the system.

See "About defining failover policies" on page 355.

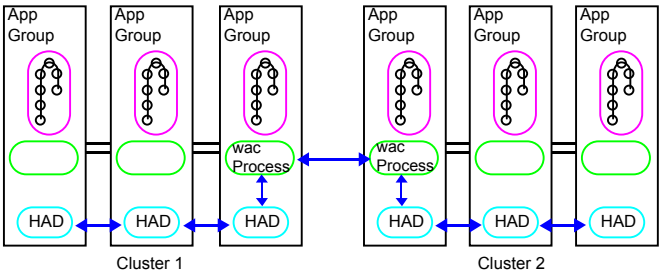
Management of remote cluster objects is aided by inter-cluster communication enabled by the wide-area connector (wac) process.

About the wide-area connector process

The wide-area connector (wac) is a failover Application resource that ensures communication between clusters.

Figure 17-2 is an illustration of the wide-area connector process.

Figure 17-2 Wide-area connector (wac) process



The wac process runs on one system in each cluster and connects with peers in remote clusters. It receives and transmits information about the status of the cluster, service groups, and systems. This communication enables VCS to create a consolidated view of the status of all the clusters configured as part of the global cluster. The process also manages wide-area heartbeating to determine the health of remote clusters. The process also transmits commands between clusters and returns the result to the originating cluster.

VCS provides the option of securing the communication between the wide-area connectors.

See " Secure communication in global clusters" on page 497.

About the wide-area heartbeat agent

The wide-area heartbeat agent manages the inter-cluster heartbeat. Heartbeats are used to monitor the health of remote clusters. VCS wide-area hearbeat agents include Icmp and IcmpS. While other VCS resource agents report their status to VCS engine, heartbeat agents report their status directly to the WAC process. The heartbeat name must be the same as the heartbeat type name. You can add only one heartbeat of a specific heartbeat type.

See "Sample configuration for the wide-area heartbeat agent" on page 492.

You can create custom wide-area heartbeat agents. For example, the VCS replication agent for SRDF includes a custom heartbeat agent for Symmetrix arrays.

You can add heartbeats using the `hahb -add heartbeatname` command and change the default values of the heartbeat agents using the `hahb -modify` command.

See "Administering heartbeats in a global cluster setup" on page 549.

See "Heartbeat attributes (for global clusters)" on page 775.

Sample configuration for the wide-area heartbeat agent

Following is a sample configuration for the wide-area heartbeat agent:

```
Heartbeat Icmp (
    ClusterList = {priclus
    Arguments @Cpriclus =
{"10.209.134.1"
    )
```

About serialization–The Authority attribute

VCS ensures that global service group operations are conducted serially to avoid timing problems and to ensure smooth performance. The Authority attribute prevents a service group from coming online in multiple clusters at the same time. Authority is a persistent service group attribute and it designates which cluster has the right to bring a global service group online. The attribute cannot be modified at runtime.

If two administrators simultaneously try to bring a service group online in a two-cluster global group, one command is honored, and the other is rejected based on the value of the Authority attribute.

The attribute prevents bringing a service group online in a cluster that does not have the authority to do so. If the cluster holding authority is down, you can enforce a takeover by using the command `hagrp -online -force service_group`. This

command enables you to fail over an application to another cluster when a disaster occurs.

Note: A cluster assuming authority for a group does not guarantee the group will be brought online on the cluster. The attribute merely specifies the right to attempt bringing the service group online in the cluster. The presence of Authority does not override group settings like frozen, autodisabled, non-probed, and so on, that prevent service groups from going online.

You must seed authority if it is not held on any cluster.

Offline operations on global groups can originate from any cluster and do not require a change of authority to do so, because taking a group offline does not necessarily indicate an intention to perform a cross-cluster failover.

About the Authority and AutoStart attributes

The attributes Authority and AutoStart work together to avoid potential concurrency violations in multi-cluster configurations.

If the AutoStartList attribute is set, and if a group's Authority attribute is set to 1, the VCS engine waits for the wac process to connect to the peer. If the connection fails, it means the peer is down and the AutoStart process proceeds. If the connection succeeds, HAD waits for the remote snapshot. If the peer is holding the authority for the group and the remote group is online (because of takeover), the local cluster does not bring the group online and relinquishes authority.

If the Authority attribute is set to 0, AutoStart is not invoked.

About resiliency and "Right of way"

VCS global clusters maintain resiliency using the wide-area connector process and the ClusterService group. The wide-area connector process runs as long as there is at least one surviving node in a cluster.

The wide-area connector, its alias, and notifier are components of the ClusterService group.

VCS agents to manage wide-area failover

VCS agents now manage external objects that are part of wide-area failover. These objects include replication, DNS updates, and so on. These agents provide a robust framework for specifying attributes and restarts, and can be brought online upon fail over.

DNS agent	The DNS agent updates the canonical name-mapping in the domain name server after a wide-area failover. See the *Cluster Server Bundled Agents Reference Guide* for more information.

VCS agents for Volume Replicator

You can use the following VCS agents for Volume Replicator in a VCS global cluster setup:

- RVG agent
  The RVG agent manages the Replicated Volume Group (RVG). Specifically, it brings the RVG online, monitors read-write access to the RVG, and takes the RVG offline. Use this agent when using Volume Replicator for replication.
- RVGPrimary agent
  The RVGPrimary agent attempts to migrate or take over a Secondary site to a Primary site following an application failover. The agent has no actions associated with the offline and monitor routines.
- RVGShared agent
  The RVGShared agent monitors the RVG in a shared environment. This is a parallel resource. The RVGShared agent enables you to configure parallel applications to use an RVG in a cluster. The RVGShared agent monitors the RVG in a shared disk group environment.
- RVGSharedPri agent
  The RVGSharedPri agent enables migration and takeover of a Volume Replicator replicated data set in parallel groups in a VCS environment. Bringing a resource of type RVGSharedPri online causes the RVG on the local host to become a primary if it is not already.
- RVGLogowner agent
  The RVGLogowner agent assigns and unassigns a node as the logowner in the CVM cluster; this is a failover resource. The RVGLogowner agent assigns or unassigns a node as a logowner in the cluster. In a shared disk group environment, currently only the cvm master node should be assigned the logowner role.
- RVGSnapshot agent
  The RVGSnapshot agent, used in fire drill service groups, takes space-optimized snapshots so that applications can be mounted at secondary sites during a fire drill operation. See the *Veritas InfoScale™ Replication Administrator's Guide* for more information.

In a CVM environment, the RVGShared agent, RVGSharedPri agent, RVGLogOwner agent, and RVGSnapshot agent are supported. For more information, see the *Cluster Server Bundled Agents Reference Guide*.

VCS agents for third-party replication technologies	VCS provides agents for other third-party array-based or application-based replication solutions. These agents are available in the High Availability Agent Pack software. See the agent pack documentation for a list of replication technologies that VCS supports.

About the Steward process: Split-brain in two-cluster global clusters

Failure of all heartbeats between any two clusters in a global cluster indicates one of the following:

- The remote cluster is faulted.
- All communication links between the two clusters are broken.

In global clusters with three or more clusters, VCS queries the connected clusters to confirm that the remote cluster is truly down. This mechanism is called inquiry.

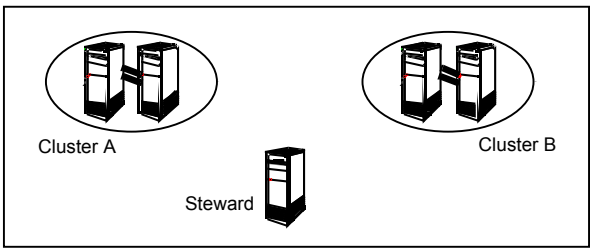
In a two-cluster setup, VCS uses the Steward process to minimize chances of a wide-area split-brain. The process runs as a standalone binary on a system outside of the global cluster configuration.

To configure redundancy for the Steward process, you can configure Steward in one of the following ways:

- Configure high availability of a single Steward process:
  You can configure the Steward process in a two-node cluster. In case of a failure, the Steward process fails over to the other node in the cluster.
- Configure multiple Stewards:
  You can configure multiple Stewards. Each Steward can be configured at a different site. If the communication links between one of the Stewards and one of the clusters are lost, the Steward process at the other site can respond to the inquiry.

Figure 17-3 depicts the Steward process to minimize chances of a split brain within a two-cluster setup.

Figure 17-3 Steward process: Split-brain in two-cluster global clusters



When all communication links between any two clusters are lost, each cluster contacts the Steward with an inquiry message. The Steward sends an ICMP ping to the cluster in question and responds with a negative inquiry if the cluster is running or with positive inquiry if the cluster is down. In case of multiple stewards, the cluster sends the inquiry to all the Stewards simultaneously. If at least one Steward responds with a negative inquiry, VCS assumes that the other cluster is running and does not need any corrective action.

The Steward can also be used in configurations with more than two clusters. VCS provides the option of securing communication between the Steward process and the wide-area connectors.

See " Secure communication in global clusters" on page 497.

In non-secure configurations, you can configure the steward process on a platform that is different to that of the global cluster nodes. Secure configurations have not been tested for running the steward process on a different platform.

For example, you can run the steward process on a Windows system for a global cluster running on Linux systems. However, the VCS release for Linux contains the steward binary for Linux only. You must copy the steward binary for Windows from the VCS installation directory on a Windows cluster, typically `C:\Program Files\VERITAS\Cluster Server`.

A Steward is effective only if there are independent paths from each cluster to the host that runs the Steward. If there is only one path between the two clusters, you must prevent split-brain by confirming manually via telephone or some messaging system with administrators at the remote site if a failure has occurred. By default, VCS global clusters fail over an application across cluster boundaries with administrator confirmation. You can configure automatic failover by setting the ClusterFailOverPolicy attribute to Auto.

The default port for the steward is 14156.

Secure communication in global clusters

In global clusters, VCS provides the option of making the following types of communication secure:

- Communication between the wide-area connectors.
- Communication between the wide-area connectors and the Steward process.

For secure authentication, the wide-area connector process gets a security context as an account in the local authentication broker on each cluster node.

The WAC account belongs to the same domain as HAD and Command Server and is specified as:

```
name = WAC
domain = VCS_SERVICES@cluster_uuid
```

See "Cluster attributes" on page 760.

You must configure the wide-area connector process in all clusters to run in secure mode. If the wide-area connector process runs in secure mode, you must run the Steward in secure mode.

See "Prerequisites for clusters running in secure mode" on page 500.

Prerequisites for global clusters

This topic describes the prerequisites for configuring global clusters.

Prerequisites for cluster setup

You must have at least two clusters to set up a global cluster. Every cluster must have the required licenses. A cluster can be part of only one global cluster. VCS supports a maximum of four clusters participating in a global cluster.

Clusters must be running on the same platform; the operating system versions can be different. Clusters must be using the same VCS version.

Cluster names must be unique within each global cluster; system and resource names need not be unique across clusters. Service group names need not be unique across clusters; however, global service groups must have identical names.

Every cluster must have a valid virtual IP address, which is tied to the cluster. Define this IP address in the cluster's ClusterAddress attribute. This address is normally configured as part of the initial VCS installation.

The global cluster operations require that the port for Wide-Area Connector (WAC) process (default is 14155) are open across firewalls.

All clusters in a global cluster must use either IPv4 or IPv6 addresses. VCS does not support configuring clusters that use different IP versions in a global cluster.

For remote cluster operations, you must configure a VCS user with the same name and privileges in each cluster.

See "User privileges for cross-cluster operations" on page 89.

Prerequisites for application setup

Applications to be configured as global groups must be configured to represent each other in their respective clusters. All application groups in a global group must have the same name in each cluster. The individual resources of the groups can

be different. For example, one group might have a MultiNIC resource or more Mount-type resources. Client systems redirected to the remote cluster in case of a wide-area failover must be presented with the same application they saw in the primary cluster.

However, the resources that make up a global group must represent the same application from the point of the client as its peer global group in the other cluster. Clients redirected to a remote cluster should not be aware that a cross-cluster failover occurred, except for some downtime while the administrator initiates or confirms the failover.

Prerequisites for wide-area heartbeats

There must be at least one wide-area heartbeat going from each cluster to every other cluster. VCS starts communicating with a cluster only after the heartbeat reports that the cluster is alive. VCS uses the ICMP ping by default, the infrastructure for which is bundled with the product. VCS configures the Icmp heartbeat if you use Cluster Manager (Java Console) to set up your global cluster. Other heartbeats must be configured manually.

Although multiple heartbeats can be configured but one heartbeat is sufficient to monitor the health of the remote site. Because Icmp & IcmpS heartbeats use IP network to check the health of the remote site. Even one heartbeat is not a single point of failure if the network is sufficiently redundant. Adding multiple heartbeats will not be useful if they have a single point of failure.

If you have a separate connection for the replication of data between the two sites, then that can be used to reduce single point of failure. Currently, Veritas only ships heartbeat agent for symmetric arrays.

Prerequisites for ClusterService group

The ClusterService group must be configured with the Application (for the wide-area connector), NIC, and IP resources. The service group may contain additional resources for Authentication Service or notification if these components are configured. The ClusterService group is configured automatically when VCS is installed or upgraded.

If you entered a license that includes VCS global cluster support during the VCS install or upgrade, the installer provides you an option to automatically configure a resource wac of type Application in the ClusterService group. The installer also configures the wide-area connector process.

You can run the Global Cluster Option (GCO) configuration wizard later to configure the WAC process and to update the ClusterService group with an Application resource for WAC.

Prerequisites for replication setup

VCS global clusters are used for disaster recovery, so you must set up real-time data replication between clusters. For supported replication technologies, you can use a VCS agent to manage the replication. These agents are available in the High Availability Agent Pack software.

See the High Availability agent pack documentation for a list of replication technologies that VCS supports.

Prerequisites for clusters running in secure mode

If you plan to configure secure communication among clusters in a global cluster, the following prerequisites must be met:

- You must configure the wide area connector processes in both clusters to run in the secure mode.
  When you configure security using the installer, the installer creates an AT account for the wide-area connector also.
- Both clusters must run in secure mode.
  - You can configure security by using the `installer -security` command.
- Both the clusters must share a trust relationship. You can set up a trust relationship by using the `installer -securitytrust` command.

For details, refer to the *Veritas InfoScale Installation Guide*.

See " Prerequisites for global clusters" on page 498.

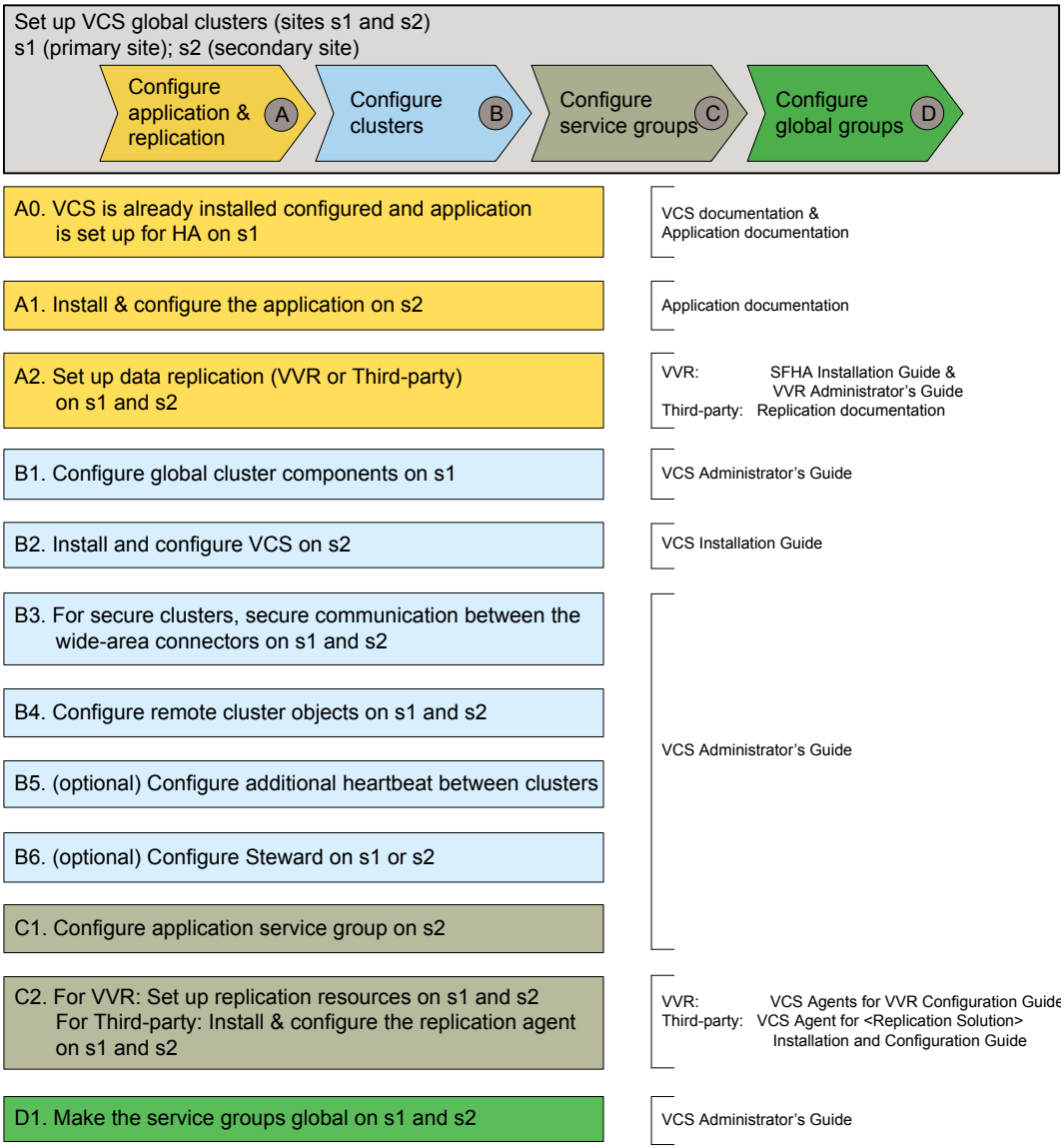
About planning to set up global clusters

Before you set up the global cluster, make sure you completed the following:

- Review the concepts.
  See " VCS global clusters: The building blocks" on page 489.
- Plan the configuration and verify that you have the required physical infrastructure.
  See " Prerequisites for global clusters" on page 498.
- Verify that you have the required software to install VCS and a supported replication technology. If you use a third-party replication technology, then verify that you have the appropriate replication agent software.

Figure 17-4 depicts the workflow to set up a global cluster where VCS cluster is already configured at site s1, and has application set up for high availability.

Figure 17-4 High-level workflow to set up VCS global clusters



Setting up a global cluster

This procedure assumes that you have configured a VCS cluster at the primary site and have set up application for high availability.

In this example, a single-instance Oracle database is configured as a VCS service group (appgroup) on a two-node cluster.

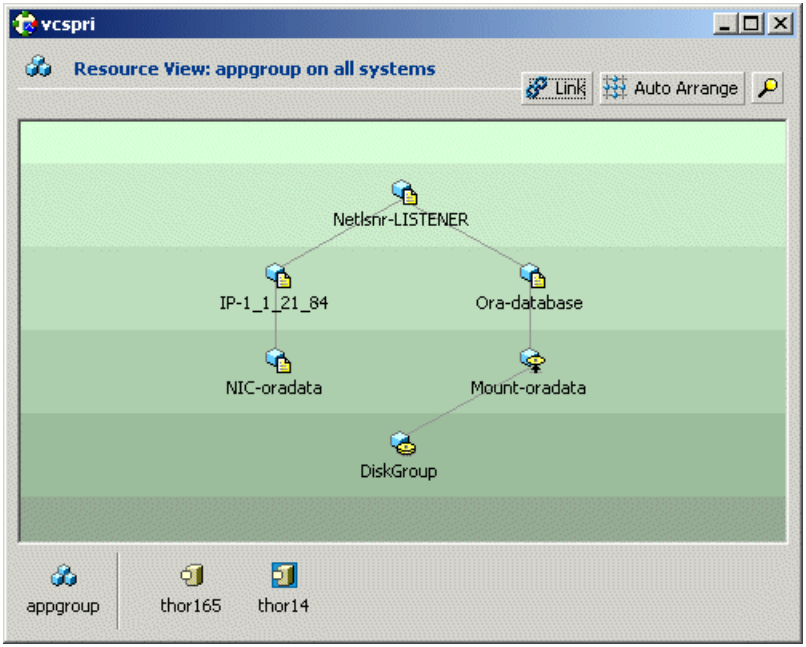


Table 17-1 lists the high-level tasks to set up VCS global clusters.

Table 17-1 Task map to set up VCS global clusters

Task	Reference
Task A	See "Configuring application and replication for global cluster setup" on page 502.
Task B	See "Configuring clusters for global cluster setup" on page 503.
Task C	See "Configuring service groups for global cluster setup" on page 511.
Task D	See "Configuring a service group as a global service group" on page 515.

Configuring application and replication for global cluster setup

Perform the following steps to configure application at the secondary site. This procedure assumes that you have already set up application for high availability at the primary site.

To configure application and replication

1. At the secondary site, install and configure the application that you want to make highly available.

   See the corresponding application documentation for instructions.

2. At each site, set up data replication using a replication technology that VCS supports:

   - Volume Replicator
     See the *Veritas InfoScale 8.0 Installation Guide* and the *Veritas InfoScale 8.0 Replication Administrator's Guide* for instructions.
   - A supported third-party replication technology
     See the corresponding replication documentation for instructions. See the *High Availability Agent Pack Getting Started Guide* for a list of third-party replication technologies that VCS supports.

Configuring clusters for global cluster setup

Perform the following steps to configure the clusters for disaster recovery:

- Configure global cluster components at the primary site
  See "Configuring global cluster components at the primary site" on page 503.
- Install and configure VCS at the secondary site
  See "Installing and configuring VCS at the secondary site" on page 505.
- Secure communication between the wide-area connectors
  See "Securing communication between the wide-area connectors" on page 505.
- Configure remote cluster objects
  See "Configuring remote cluster objects" on page 507.
- Configure additional heartbeat links (optional)
  See "Configuring additional heartbeat links (optional)" on page 507.
- Configure the Steward process (optional)
  See "Configuring the Steward process (optional)" on page 507.

Configuring global cluster components at the primary site

Perform the following steps to configure global cluster components at the primary site.

If you have already completed these steps during the VCS cluster configuration at the primary site, then proceed to the next task to set up a VCS cluster at the secondary site.

See "Installing and configuring VCS at the secondary site" on page 505.

Run the GCO Configuration wizard to create or update the ClusterService group. The wizard verifies your configuration and validates it for a global cluster setup.

See "About installing a VCS license" on page 100.

To configure global cluster components at the primary site

1. Start the GCO Configuration wizard.

   ```
   # gcoconfig
   ```

   The wizard discovers the NIC devices on the local system and prompts you to enter the device to be used for the global cluster.

2. Specify the name of the device and press Enter.

3. If you do not have NIC resources in your configuration, the wizard asks you whether the specified NIC will be the public NIC used by all systems.

   Enter **y** if it is the public NIC; otherwise enter **n**. If you entered n, the wizard prompts you to enter the names of NICs on all systems.

4. Enter the virtual IP to be used for the global cluster.

   You must use either IPv4 or IPv6 address. VCS does not support configuring clusters that use different Internet Protocol versions in a global cluster.

5. If you do not have IP resources in your configuration, the wizard does the following:

   - For IPv4 address:
     The wizard prompts you for the netmask associated with the virtual IP. The wizard detects the netmask; you can accept the suggested value or enter another value.
   - For IPv6 address:
     The wizard prompts you for the prefix associated with the virtual IP.

6. The wizard starts running commands to create or update the ClusterService group. Various messages indicate the status of these commands. After running these commands, the wizard brings the ClusterService group online.

7. Verify that the gcoip resource that monitors the virtual IP address for inter-cluster communication is online.

   ```
   # hares -state gcoip
   ```

Installing and configuring VCS at the secondary site

Perform the following steps to set up a VCS cluster at the secondary site.

To install and configure VCS at the secondary site

1. At the secondary site, install and configure VCS cluster.

   Note the following points for this task:

   - During VCS configuration, answer the prompts to configure global cluster. This step configures the virtual IP for inter-cluster communication.

   See the *Cluster Server Configuration and Upgrade Guide* for instructions.

2. Verify that the gcoip resource that monitors the virtual IP address for inter-cluster communication is online.

   ```
   # hares -state gcoip
   ```

Securing communication between the wide-area connectors

Perform the following steps to configure secure communication between the wide-area connectors.

To secure communication between the wide-area connectors

1. Verify that security is configured in both the clusters. You can use the `installer -security` command to configure security.

   For more information, see the *Cluster Server Configuration and Upgrade Guide*.

2. Establish trust between the clusters.

   For example in a VCS global cluster environment with two clusters, perform the following steps to establish trust between the clusters:

   - On each node of the first cluster, enter the following command:

     ```
     # export EAT_DATA_DIR=/var/VRTSvcs/vcsauth/data/WAC;
     /opt/VRTSvcs/bin/vcsat setuptrust -b
     IP_address_of_any_node_from_the_second_cluster:14149 -s high
     ```

     The command obtains and displays the security certificate and other details of the root broker of the second cluster.
     If the details are correct, enter y at the command prompt to establish trust.
     For example:

```
The hash of above credential is
  b36a2607bf48296063068e3fc49188596aa079bb
  Do you want to trust the above?(y/n) y
```

- On each node of the second cluster, enter the following command:

  ```
  # export EAT_DATA_DIR=/var/VRTSvcs/vcsauth/data/WAC
  /opt/VRTSvcs/bin/vcsat setuptrust -b
  IP_address_of_any_node_from_the_first_cluster:14149 -s high
  ```

  The command obtains and displays the security certificate and other details of the root broker of the first cluster.
  If the details are correct, enter y at the command prompt to establish trust.
  Alternatively, if you have passwordless communication set up on the cluster, you can use the `installer -securitytrust` option to set up trust with a remote cluster.

3
- Skip the remaining steps in this procedure if you used the `installer -security` command after the global cluster was set up.
- Complete the remaining steps in this procedure if you had a secure cluster and then used the `gcoconfig` command.

On each cluster, take the `wac` resource offline on the node where the `wac` resource is online. For each cluster, run the following command:

```
# hares -offline wac -sys node_where_wac_is_online
```

4 Update the values of the StartProgram and MonitorProcesses attributes of the `wac` resource:

```
# haconf -makerw
hares -modify wac StartProgram \
"/opt/VRTSvcs/bin/wacstart -secure"
hares -modify wac MonitorProcesses \
"/opt/VRTSvcs/bin/wac -secure"
haconf -dump -makero
```

5 On each cluster, bring the `wac` resource online. For each cluster, run the following command on any node:

```
# hares -online wac -sys systemname
```

Configuring remote cluster objects

After you set up the VCS and replication infrastructure at both sites, you must link the two clusters. You must configure remote cluster objects at each site to link the two clusters. The Remote Cluster Configuration wizard provides an easy interface to link clusters.

To configure remote cluster objects

- Start the Remote Cluster Configuration wizard.

  From Cluster Explorer, click **Edit > Add/Delete Remote Cluster**.

  You must use the same IP address as the one assigned to the IP resource in the ClusterService group. Global clusters use this IP address to communicate and exchange ICMP heartbeats between the clusters.

Configuring additional heartbeat links (optional)

You can configure additional heartbeat links to exchange ICMP heartbeats between the clusters.

To configure an additional heartbeat between the clusters (optional)

1. On Cluster Explorer's **Edit** menu, click **Configure Heartbeats**.
2. In the Heartbeat configuration dialog box, enter the name of the heartbeat and select the check box next to the name of the cluster.
3. Click the icon in the **Configure** column to open the Heartbeat Settings dialog box.
4. Specify the value of the Arguments attribute and various timeout and interval fields. Click **+** to add an argument value; click **-** to delete it.

   If you specify IP addresses in the Arguments attribute, make sure the IP addresses have DNS entries.
5. Click **OK**.
6. Click **OK** in the Heartbeat configuration dialog box.

   Now, you can monitor the state of both clusters from the Java Console.

Configuring the Steward process (optional)

In case of a two-cluster global cluster setup, you can configure a Steward to prevent potential split-brain conditions, provided the proper network infrastructure exists.

See "About the Steward process: Split-brain in two-cluster global clusters" on page 496.

To configure the Steward process for clusters not running in secure mode

1. Identify a system that will host the Steward process.

   To configure Steward in a dual-stack configuration, ensure that you enable IPv4 and IPv6 on the system that will host the Steward process. You must also plumb both the IPv4 and IPv6 addresses on the host system.

2. Make sure that both clusters can connect to the system through a ping command.

3. Copy the file steward from a node in the cluster to the Steward system. The file resides at the following path:

   ```
   /opt/VRTSvcs/bin/
   ```

4. In both the clusters, set the Stewards attribute to the IP address of the system running the Steward process.

   The steward attribute must contain the IPv4 or IPv6 address of the steward server depending on whether the cluster node is configured with IPv4 or IPv6 respectively.

   When a cluster node is configured with IPv4, the steward attribute must be set to the IPv4 address of the steward server. When a cluster node is configured with IPv6, the Steward attribute must be set to the IPv6 address.

   For example:

   ```
   cluster cluster1938 (
   UserNames = { admin = gNOgNInKOjOOmWOiNL }
   ClusterAddress = "10.182.147.19"
   Administrators = { admin }
   CredRenewFrequency = 0
   CounterInterval = 5
   Stewards = {"10.212.100.165", "10.212.101.162"}
   }
   ```

5. On the system designated to host the Steward, start the Steward process:

   ```
   # steward -start
   ```

To configure the Steward process for clusters running in secure mode

1. Verify that the prerequisites for securing Steward communication are met.

   See "Prerequisites for clusters running in secure mode" on page 500.

   To verify that the wac process runs in secure mode, do the following:

   - Check the value of the wac resource attributes:

```
# hares -value wac StartProgram
```

The value must be “/opt/VRTSvcs/bin/wacstart –secure.”

```
# hares -value wac MonitorProcesses
```

The value must be “/opt/VRTSvcs/bin/wac –secure.”

- List the wac process:

```
# ps -ef | grep wac
```

The wac process must run as “/opt/VRTSvcs/bin/wac –secure.”

2. Identify a system that will host the Steward process.
3. Make sure that both clusters can connect to the system through a ping command.
4. Perform this step *only* if VCS is not already installed on the Steward system. If VCS is already installed, skip to step 5.
   - Install the VRTSvcs and VRTSperl RPMs.
   - If the cluster UUID is not configured, configure it by using `/opt/VRTSvcs/bin/uuidconfig.pl`.
   - On the system that is designated to run the Steward process, run the `installer -securityonenode` command.
     The installer prompts for a confirmation if VCS is not configured or if VCS is not running on all nodes of the cluster. Enter **y** when the installer prompts whether you want to continue configuring security.
     For more information about the `-securityonenode` option, see the *Cluster Server Configuration and Upgrade Guide*.

5 Generate credentials for the Steward using `/opt/VRTSvcs/bin/steward_secure.pl` or perform the following steps:

```
# unset EAT_DATA_DIR

# unset EAT_HOME_DIR

# /opt/VRTSvcs/bin/vcsauth/vcsauthserver/bin/vssat createpd -d
VCS_SERVICES -t ab

# /opt/VRTSvcs/bin/vcsauth/vcsauthserver/bin/vssat addprpl  -t ab
-d VCS_SERVICES -p  STEWARD -s password

# mkdir -p /var/VRTSvcs/vcsauth/data/STEWARD

# export EAT_DATA_DIR=/var/VRTSvcs/vcsauth/data/STEWARD

# /opt/VRTSvcs/bin/vcsat setuptrust  -s high -b localhost:14149
```

6 Set up trust on all nodes of the GCO clusters:

```
# export EAT_DATA_DIR=/var/VRTSvcs/vcsauth/data/WAC

# vcsat setuptrust -b <IP_of_Steward>:14149 -s high
```

7 Set up trust on the Steward:

```
# export EAT_DATA_DIR=/var/VRTSvcs/vcsauth/data/STEWARD

# vcsat setuptrust -b <VIP_of_remote_cluster1>:14149 -s high

# vcsat setuptrust -b <VIP_of_remote_cluster2>:14149 -s high
```

8 In both the clusters, set the Stewards attribute to the IP address of the system running the Steward process.

For example:

```
cluster cluster1938 (
UserNames = { admin = gNOgNInKOjOOmWOiNL }
ClusterAddress = "10.182.147.19"
Administrators = { admin }
CredRenewFrequency = 0
CounterInterval = 5
Stewards = {"10.212.100.165", "10.212.101.162}
}
```

9 On the system designated to run the Steward, start the Steward process:

```
# /opt/VRTSvcs/bin/steward -start -secure
```

To stop the Steward process

◆ To stop the Steward process that is not configured in secure mode, open a new command window and run the following command:

```
# steward -stop
```

To stop the Steward process running in secure mode, open a new command window and run the following command:

```
# steward -stop -secure
```

Configuring service groups for global cluster setup

Perform the following steps to configure the service groups for disaster recovery.

To configure service groups

1 At the secondary site, set up the application for high availability.

Configure VCS service groups for the application. Create a configuration that is similar to the one in the first cluster.

- You can do this by either using Cluster Manager (Java Console) to copy and paste resources from the primary cluster, or by copying the configuration from the main.cf file in the primary cluster to the secondary cluster.
- Make appropriate changes to the configuration. For example, you must modify the SystemList attribute to reflect the systems in the secondary cluster.

- Make sure that the name of the service group (appgroup) is identical in both clusters.

2 To assign remote administration privileges to users for remote cluster operations, configure users with the same name and privileges on both clusters.

   See "User privileges for cross-cluster operations" on page 89.

3 If your setup uses BIND DNS, add a resource of type DNS to the application service group at each site.

   Refer to the *Cluster Server Bundled Agent's Reference Guide* for more details.

4 At each site, perform the following depending on the replication technology you have set up:

   - Volume Replicator
     Configure VCS to manage and monitor Replicated Volume Groups (RVGs).
     See "Configuring VCS service group for VVR-based replication" on page 512.
   - A supported third-party replication technology
     Install and configure the corresponding VCS agent for replication.
     See the Installation and Configuration Guide for the corresponding VCS replication agent for instructions.

Configuring VCS service group for VVR-based replication

Perform the following steps to configure VCS to monitor Volume Replicator (VVR). Then, set an online local hard group dependency from application service group (appgroup) to replication service group (appgroup_rep) to ensure that the service groups fail over and switch together.

To create the RVG resources in VCS

1 Create a new service group, say appgroup_rep.

2 Copy the DiskGroup resource from the appgroup to the new group.

3 Configure new resources of type IP and NIC in the appgroup_rep service group. The IP resource monitors the virtual IP that VVR uses for replication.

4 Configure a new resource of type RVG in the new (appgroup_rep) service group.

5 Configure the RVG resource.

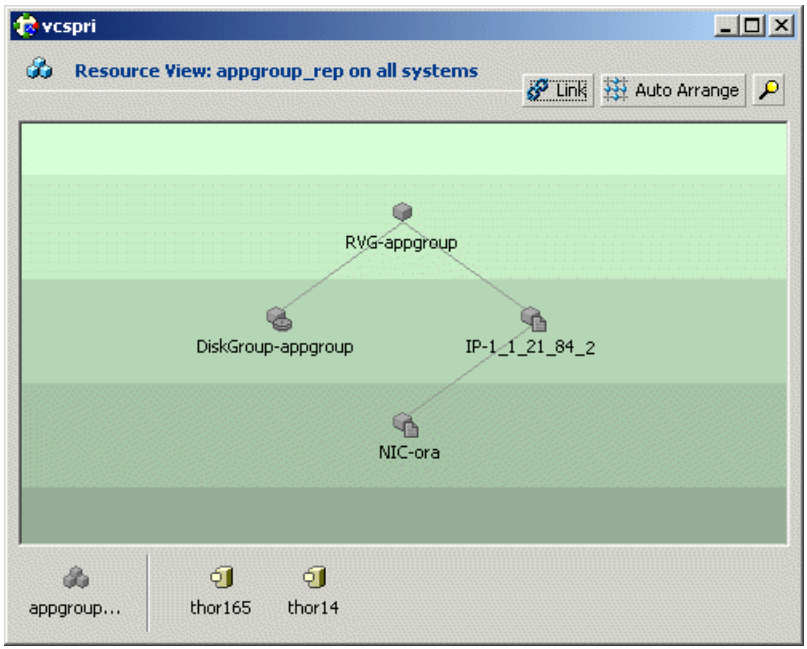
   See the *Veritas InfoScale™ Replication Administrator's Guide* for more information about the resource.

   Note that the RVG resource starts, stops, and monitors the RVG in its current state and does not promote or demote VVR when you want to change the direction of replication. That task is managed by the RVGPrimary agent.

6 Set dependencies as per the following information:

- RVG resource depends on the IP resource.
- RVG resource depends on the DiskGroup resource.
- IP resource depends on the NIC resource.

The service group now looks like:



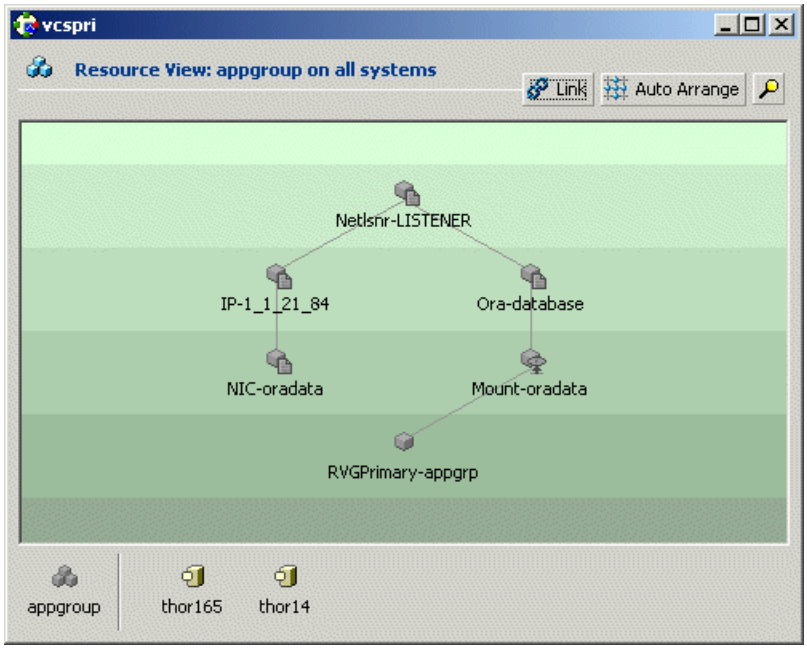
7 Delete the DiskGroup resource from the appgroup service group.

8 In the application service group (appgroup), add a resource of type RVGPrimary and configure its attributes.

See the *Veritas InfoScale™ Replication Administrator's Guide* for more information about the resource.

9 Set resource dependencies such that the Mount resource depends on the RVGPrimary resource.

The appgroup now looks like:



To link the application and replication service groups

1 In the Cluster Explorer configuration tree, click the cluster name.

2 In the view panel, click the **Service Groups** tab.

This opens the service group dependency graph.

3 Click **Link**.

4 Click the parent group, appgroup, and move the mouse toward the child group, appgroup_rep.

5 Click the child group appgroup_rep.

6 In the Link Service Groups dialog box, click the online local relationship and the hard dependency type and click **OK**.

Configuring a service group as a global service group

Run the Global Group Configuration wizard to configure the application service group (appgroup) as a global group.

To create the global service group

1 In the service group tree of Cluster Explorer, right-click the application service group (appgroup).

2 Select **Configure As Global** from the menu.

3 Enter the details of the service group to modify (appgroup).

4 From the **Available Clusters** box, click the clusters on which the group can come online. The local cluster is not listed as it is implicitly defined to be part of the ClusterList. Click the right arrow to move the cluster name to the **ClusterList** box.

5 Select the policy for cluster failover:

- **Manual** prevents a group from automatically failing over to another cluster.
- **Auto** enables a group to automatically fail over to another cluster if it is unable to fail over within the cluster, or if the entire cluster faults.
- **Connected** enables a group to automatically fail over to another cluster if it is unable to fail over within the cluster.

6 Click **Next**.

7 Enter or review the connection details for each cluster.

Click the **Configure** icon to review the remote cluster information for each cluster.

8 Enter the IP address of the remote cluster, the IP address of a cluster system, or the host name of a cluster system.

9 Enter the user name and the password for the remote cluster and click **OK**.

10 Click **Next**.

11 Click **Finish**.

12 Save the configuration.

The appgroup service group is now a global group and can be failed over between clusters.

About IPv6 support with global clusters

You can configuring global clusters to support IPv6 or migrate existing global clusters configured on IPv4 to IPv6.

For details, review the following topics:

- See "Prerequisites for configuring a global cluster to support IPv6" on page 516.
- See "Migrating an InfoScale Availability cluster from IPv4 to IPv6 when Virtual IP (ClusterAddress) is configured" on page 516.
- See "Migrating an InfoScale Availability cluster to IPv6 in a GCO deployment" on page 518.

Prerequisites for configuring a global cluster to support IPv6

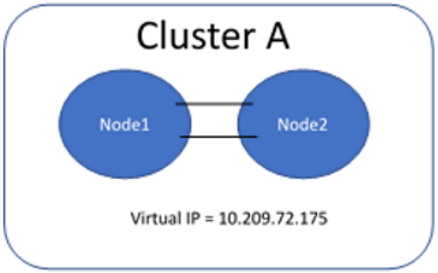
You can configure global clusters to link clusters at separate locations and enable wide-area failover and disaster recovery. The installer adds basic global cluster information to the VCS configuration file. You must perform additional configuration tasks to update a global cluster to support IPv6.

Consider the following prerequisites before updating a global cluster to support IPv6:

- You must have at least two clusters that are not a part of any other global cluster.
- Every cluster must have the required licenses.
- Clusters must run on the same platform and must use the same InfoScale version. The platform versions can be different.
- Global service groups must be configured with identical names.
- The port for the Wide Area Connector (WAC) process (default: 14155) is open across firewalls.

Migrating an InfoScale Availability cluster from IPv4 to IPv6 when Virtual IP (ClusterAddress) is configured

The following graphic shows a sample migration scenario for an InfoScale Availability cluster that is not a part of any global cluster. The cluster is configured using IPv4 and a virtual IP is configured. To migrate Cluster A to IPv6 configuration, you must update the InfoScale Availability configuration to update the IP addresses of the IPv4 IP resources to IPv6 within the cluster. Ensure that the IPv6 stack is enabled before you proceed with the migration.



To migrate a cluster where a virtual IP is configured to an IPv6 configuration

1. Set the cluster configuration to read-write mode.

   ```
   # haconf -makerw
   ```

2. Replace the virtual IP address of the IP resource in the cluster with IPv6 address.

   ```
   # hares -modify IPResourceName Address IPv6Address
   ```

   For example:

   ```
   # hares -modify vip Address 2620:128:f0a2:9005::117
   ```

3. Add the PrefixLen attribute to the IP resource and assign an appropriate value to it.

   ```
   # hares -modify IPResourceName PrefixLen prefixLength
   ```

   For example:

   ```
   # hares -modify vip PrefixLen 64
   ```

4. Remove the NetMask attribute of the IP resource from the configuration.

   ```
   # hares -modify IPResourceName NetMask ""
   ```

   For example:

   ```
   # hares -modify vip NetMask ""
   ```

5. Set the value of the ClusterAddress attribute to IPv6 address.

   ```
   # haclus -modify ClusterAddress IPv6Address
   ```

   For example:

   ```
   haclus -modify ClusterAddress 2620:128:f0a2:9005::117
   ```

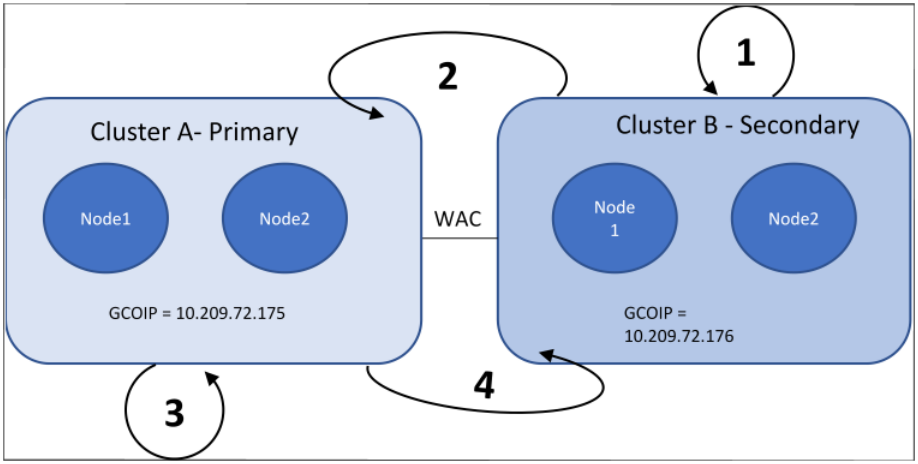
6. Ensure that all the cluster nodes are now configured using IPv6 addresses.

7. Save the modified cluster configuration and make it read only.

   ```
   # haconf -dump -makero
   ```

Migrating an InfoScale Availability cluster to IPv6 in a GCO deployment

The following graphic shows a sample migration scenario where two InfoScale Availability clusters are a part of an existing global cluster. The clusters are configured using IPv4, and the GCO IP is configured for each cluster.



Migrating clusters to IPv6 in a GCO deployment involves the following steps:

1. Updating the secondary cluster to IPv6.

   To migrate a local VCS cluster to IPv6

2. Updating the primary cluster with the details of the IP changes in the secondary cluster.

   To update the remote cluster with the details of the IP changes on the local cluster

3. Updating the primary cluster to IPv6.

   To migrate a local VCS cluster to IPv6

4. Updating the secondary cluster with the details of the IP changes in the primary cluster.

   To update the remote cluster with the details of the IP changes on the local cluster

---

Note: Veritas recommends that you update the secondary cluster to IPv6 first. Doing so ensures that the primary cluster is operational while the update of the secondary cluster is in progress. It also allows for an easy rollback in case of issues with the update.

To migrate a local VCS cluster to IPv6

1. Set the cluster to read-write mode.

   ```
   # haconf -makerw
   ```

2. Replace the virtual IP address of the IP resource in the cluster with an IPv6 address.

   ```
   # hares -modify IPResourceName Address IPv6Address
   ```

   For example:

   ```
   # hares -modify gcoip Address 2620:128:f0a2:9005::117
   ```

3. Add the PrefixLen attribute to the IP resource and assign appropriate value to it.

   ```
   # hares -modify IPResourceName PrefixLen prefixLength
   ```

   For example:

   ```
   # hares -modify gcoip PrefixLen 64
   ```

4. Remove the NetMask attribute of the IP resource from the configuration.

   ```
   # hares -modify IPResourceName NetMask ""
   ```

   For example:

   ```
   # hares -modify gcoip NetMask ""
   ```

5. Set the value of the ClusterAddress attribute to IPv6 address.

   ```
   # haclus -modify ClusterAddress IPv6Address
   ```

   For example:

   ```
   haclus -modify ClusterAddress 2620:128:f0a2:9005::117
   ```

6. Ensure that all cluster nodes are now configured using IPv6 addresses.

7. Save the modified cluster configuration and make it read only.

   ```
   # haconf -dump -makero
   ```

8. Restart the WAC resource.

9. Check WAC communication and ICMP status.

   ```
   # hastatus -sum
   ```

To update the remote cluster with the details of the IP changes on the local cluster

1. Make cluster configuration writable.

   ```
   # haconf -makerw
   ```

2. Modify the remote cluster address with an IPv6 address.

   ```
   # haclus -modify ClusterAddress IPv6Address -clus
   remoteClusterName
   ```

   For example:

   ```
   # haclus -modify ClusterAddress 2620:128:f0a2:9005::117 -clus
   gcocluster2
   ```

3. Modify the IP address of the ICMP agent with the IPv6 address of the remote cluster.

   ```
   # hahb -modify Icmp Arguments IPv6Address -clus remoteClusterName
   ```

   For example:

   ```
   # hahb -modify Icmp Arguments 2620:128:f0a2:9005::117 -clus
   gcocluster2
   ```

4. Update the cluster configuration with the modified data.

   ```
   # haconf -dump -makero
   ```

5. Restart the WAC resource on the secondary sites.

6. Check WAC communication and ICMP status.

   ```
   # hastatus -sum
   ```

About cluster faults

In the global cluster setup, consider a case where the primary cluster suffers a failure. The Oracle service group cannot fail over in the local cluster and must fail over globally, to a node in another cluster.

In this situation, VCS sends an alert indicating that the cluster is down.

An administrator can bring the group online in the remote cluster.

The RVGPrimary agent ensures that Volume Replicator volumes are made writable and the DNS agent ensures that name services are resolved to the remote site. The application can be started at the remote site.

About the type of failure

If a disaster disables all processing power in your primary data center, heartbeats from the failover site to the primary data center fail. VCS sends an alert signalling cluster failure. If you choose to take action on this failure, VCS prompts you to declare the type of failure.

You can choose one of the following options to declare the failure:

- Disaster, implying permanent loss of the primary data center
- Outage, implying the primary may return to its current form in some time
- Disconnect, implying a split-brain condition; both clusters are up, but the link between them is broken
- Replica, implying that data on the takeover target has been made consistent from a backup source and that the RVGPrimary can initiate a takeover when the service group is brought online. This option applies to Volume Replicator environments only.

You can select the groups to be failed over to the local cluster, in which case VCS brings the selected groups online on a node based on the group's FailOverPolicy attribute. It also marks the groups as being OFFLINE in the other cluster. If you do not select any service groups to fail over, VCS takes no action except implicitly marking the service groups as offline in the failed cluster.

Switching the service group back to the primary

You can switch the service group back to the primary after resolving the fault at the primary site. Before switching the application to the primary site, you must resynchronize any changed data from the active Secondary site since the failover. This can be done manually through Volume Replicator or by running a VCS action from the RVGPrimary resource.

To switch the service group when the primary site has failed and the secondary did a takeover

1. In the **Service Groups** tab of the configuration tree, right-click the resource.
2. Click **Actions**.
3. Specify the details of the action:
   - From the **Action** list, choose fbsync.
   - Click the system on which to execute the action.
   - Click **OK**.

This begins a fast-failback of the replicated data set. You can monitor the value of the ResourceInfo attribute for the RVG resource to determine when the resynchronization has completed.

4 Once the resynchronization completes, switch the service group to the primary cluster.

- In the **Service Groups** tab of the Cluster Explorer configuration tree, right-click the service group.
- Click **Switch To**, and click **Remote switch**.
- In the Switch global group dialog box, click the cluster to switch the group. Click the specific system, or click **Any System**, and click **OK**.

About setting up a disaster recovery fire drill

The disaster recovery fire drill procedure tests the fault-readiness of a configuration by mimicking a failover from the primary site to the secondary site. This procedure is done without stopping the application at the primary site and disrupting user access, interrupting the flow of replicated data, or causing the secondary site to need resynchronization.

The initial steps to create a fire drill service group on the secondary site that closely follows the configuration of the original application service group and contains a point-in-time copy of the production data in the Replicated Volume Group (RVG). Bringing the fire drill service group online on the secondary site demonstrates the ability of the application service group to fail over and come online at the secondary site, should the need arise. Fire drill service groups do not interact with outside clients or with other instances of resources, so they can safely come online even when the application service group is online.

You must conduct a fire drill only at the secondary site; do not bring the fire drill service group online on the node hosting the original application.

You can override the FireDrill attribute and make fire drill resource-specific.

Before you perform a fire drill in a disaster recovery setup that uses Volume Replicator, perform the following steps:

- Configure the fire drill service group.
  See "About creating and configuring the fire drill service group manually" on page 523.
  See "About configuring the fire drill service group using the Fire Drill Setup wizard" on page 526.
  If you configure the fire drill service group manually using the command line or the Cluster Manager (Java Console), set an offline local dependency between

the fire drill service group and the application service group to make sure a fire drill does not block an application failover in case a disaster strikes the primary site. If you use the Fire Drill Setup (fdsetup) wizard, the wizard creates this dependency.

- Set the value of the ReuseMntPt attribute to 1 for all Mount resources.
- After the fire drill service group is taken offline, reset the value of the ReuseMntPt attribute to 0 for all Mount resources.

VCS also supports HA fire drills to verify a resource can fail over to another node in the cluster.

See "About testing resource failover by using HA fire drills" on page 225.

Note: You can conduct fire drills only on regular VxVM volumes; volume sets (vset) are not supported.

VCS provides hardware replication agents for array-based solutions, such as Hitachi Truecopy, EMC SRDF, and so on . If you are using hardware replication agents to monitor the replicated data clusters, refer to the VCS replication agent documentation for details on setting up and configuring fire drill.

About creating and configuring the fire drill service group manually

You can create the fire drill service group using the command line or Cluster Manager (Java Console.) The fire drill service group uses the duplicated copy of the application data.

Creating and configuring the fire drill service group involves the following tasks:

- See "Creating the fire drill service group" on page 523.
- See "Linking the fire drill and replication service groups" on page 524.
- See "Adding resources to the fire drill service group" on page 525.
- See "Configuring the fire drill service group" on page 525.
- See "Enabling the FireDrill attribute" on page 526.

Creating the fire drill service group

This section describes how to use the Cluster Manager (Java Console) to create the fire drill service group and change the failover attribute to false so that the fire drill service group does not failover to another node during a test.

To create the fire drill service group

1. Open the **Veritas Cluster Manager - Java Console** from the **Apps** menu on the Start screen.
2. Log on to the cluster and click **OK**.
3. Click the **Service Group** tab in the left pane and click the **Resources** tab in the right pane.
4. Right-click the cluster in the left pane and click **Add Service Group**.
5. In the **Add Service Group** dialog box, provide information about the new service group.
   - In Service Group name, enter a name for the fire drill service group
   - Select systems from the Available Systems box and click the arrows to add them to the Systems for Service Group box.
   - Click **OK**.

To disable the AutoFailOver attribute

1. Click the **Service Group** tab in the left pane and select the fire drill service group.
2. Click the **Properties** tab in the right pane.
3. Click the **Show all attributes** button.
4. Double-click the **AutoFailOver** attribute.
5. In the **Edit Attribute** dialog box, clear the **AutoFailOver** check box.
6. Click **OK** to close the **Edit Attribute** dialog box.
7. Click the **Save and Close Configuration** icon in the tool bar.

Linking the fire drill and replication service groups

Create an online local firm dependency link between the fire drill service group and the replication service group.

To link the service groups

1. In Cluster Explorer, click the System tab in the left pane and click the**Service Groups** tab in the right pane.
2. Click **Link**.

3 Click the fire drill service group, drag the link and click the replication service group.

4 Define the dependency. Choose the **online local** and **firm** options and click OK.

Adding resources to the fire drill service group

Add resources to the new fire drill service group to recreate key aspects of the application service group.

To add resources to the service group

1 In Cluster Explorer, click the **Service Group** tab in the left pane, click the application service group and click the **Resources** tab in the right pane.

2 Right-click the resource at the top of the tree, select **Copy** and click **Self and Child Nodes**.

3 In the left pane, click the fire drill service group.

4 Right-click the right pane, and click **Paste**.

5 In the Name Clashes dialog box, specify a way for the resource names to be modified, for example, insert an FD_ prefix. Click **Apply**.

6 Click **OK**.

Configuring the fire drill service group

After copying resources to the fire drill service group, edit the resources so they will work properly with the duplicated data. The attributes must be modified to reflect the configuration at the remote site. Bringing the service group online without modifying resource attributes is likely to result in a cluster fault and interruption in service.

To configure the service group

1 In Cluster Explorer, click the **Service Group** tab in the left pane, click the fire drill service group in the left pane and click the **Resources** tab in the right pane.

2 Right-click the RVGPrimary resource and click **Delete**.

3 Right-click the resource to be edited and click **View > Properties View**. If a resource to be edited does not appear in the pane, click **Show All Attributes**.

4 Edit attributes to reflect the configuration at the remote site. For example, change the MountV resources so that they point to the volumes used in the fire drill service group. Similarly, reconfigure the DNS and IP resources.

Enabling the FireDrill attribute

You must edit certain resource types so they are FireDrill-enabled. Making a resource type FireDrill-enabled changes the way that VCS checks for concurrency violations. Typically, when FireDrill is not enabled, resources can not come online on more than one node in a cluster at a time. This behavior prevents multiple nodes from using a single resource or from answering client requests. Fire drill service groups do not interact with outside clients or with other instances of resources, so they can safely come online even when the application service group is online.

Typically, you would enable the FireDrill attribute for the resource type that are used to configure the agent.

To enable the FireDrill attribute

1. In Cluster Explorer, click the **Types** tab in the left pane, right-click the type to be edited, and click **View > Properties View.**
2. Click **Show All Attributes.**
3. Double click **FireDrill**.
4. In the **Edit Attribute** dialog box, enable **FireDrill** as required, and click **OK**.

   Repeat the process of enabling the FireDrill attribute for all required resource types.

About configuring the fire drill service group using the Fire Drill Setup wizard

Use the Fire Drill Setup Wizard to set up the fire drill configuration.

The wizard performs the following specific tasks:

- Creates a Cache object to store changed blocks during the fire drill, which minimizes disk space and disk spindles required to perform the fire drill.
- Configures a VCS service group that resembles the real application group.

The wizard works only with application groups that contain one disk group. The wizard sets up the first RVG in an application. If the application has more than one RVG, you must create space-optimized snapshots and configure VCS manually, using the first RVG as reference.

You can schedule the fire drill for the service group using the `fdsched` script.

See “Scheduling a fire drill” on page 528.

Running the fire drill setup wizard

To run the wizard

1. Start the RVG Secondary Fire Drill wizard on the Volume Replicator secondary site, where the application service group is offline and the replication group is online as a secondary:

   ```
   # /opt/VRTSvcs/bin/fdsetup
   ```

2. Read the information on the Welcome screen and press the **Enter** key.
3. The wizard identifies the global service groups. Enter the name of the service group for the fire drill.
4. Review the list of volumes in disk group that could be used for a space-optimized snapshot. Enter the volumes to be selected for the snapshot. Typically, all volumes used by the application, whether replicated or not, should be prepared, otherwise a snapshot might not succeed.

   Press the **Enter** key when prompted.
5. Enter the cache size to store writes when the snapshot exists. The size of the cache must be large enough to store the expected number of changed blocks during the fire drill. However, the cache is configured to grow automatically if it fills up. Enter disks on which to create the cache.

   Press the **Enter** key when prompted.
6. The wizard starts running commands to create the fire drill setup.

   Press the **Enter** key when prompted.

   The wizard creates the application group with its associated resources. It also creates a fire drill group with resources for the application (Oracle, for example), the Mount, and the RVGSnapshot types.

   The application resources in both service groups define the same application, the same database in this example. The wizard sets the FireDrill attribute for the application resource to 1 to prevent the agent from reporting a concurrency violation when the actual application instance and the fire drill service group are online at the same time.

About configuring local attributes in the fire drill service group

The fire drill setup wizard does not recognize localized attribute values for resources. If the application service group has resources with local (per-system) attribute values, you must manually set these attributes after running the wizard.

Verifying a successful fire drill

If you configured the fire drill service group manually using the command line or the Cluster Manager (Java Console), make sure that you set an offline local dependency between the fire drill service group and the application service group. This dependency ensures that a fire drill does not block an application failover in case a disaster strikes the primary site.

Bring the fire drill service group online on a node that does not have the application running. Verify that the fire drill service group comes online. This action validates that your disaster recovery solution is configured correctly and the production service group will fail over to the secondary site in the event of an actual failure (disaster) at the primary site.

If the fire drill service group does not come online, review the VCS engine log to troubleshoot the issues so that corrective action can be taken as necessary in the production service group.

You can also view the fire drill log, located at `/tmp/fd-servicegroup.pid`

Remember to take the fire drill offline once its functioning has been validated. Failing to take the fire drill offline could cause failures in your environment. For example, if the application service group were to fail over to the node hosting the fire drill service group, there would be resource conflicts, resulting in both service groups faulting.

Scheduling a fire drill

You can schedule the fire drill for the service group using the fdsched script. The fdsched script is designed to run only on the lowest numbered node that is currently running in the cluster. The scheduler runs the command `hagrp - online` *`firedrill_group`* `-any` at periodic intervals.

To schedule a fire drill

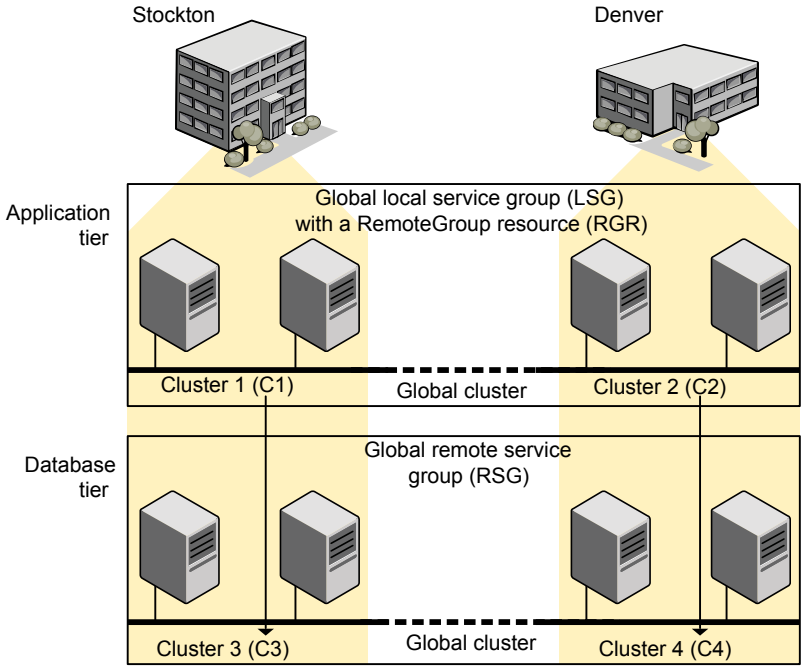
1. Add the file `/opt/VRTSvcs/bin/fdsched` to your crontab.
2. To make fire drills highly available, add the `fdsched` file to each node in the cluster.

Multi-tiered application support using the RemoteGroup agent in a global environment

Figure 17-5 represents a two-site, two-tier environment. The application cluster, which is globally clustered between L.A. and Denver, has cluster dependencies up and down the tiers. Cluster 1 (C1), depends on the remote service group for cluster

3 (C3). At the same time, cluster 2 (C2) also depends on the remote service group for cluster 4 (C4).

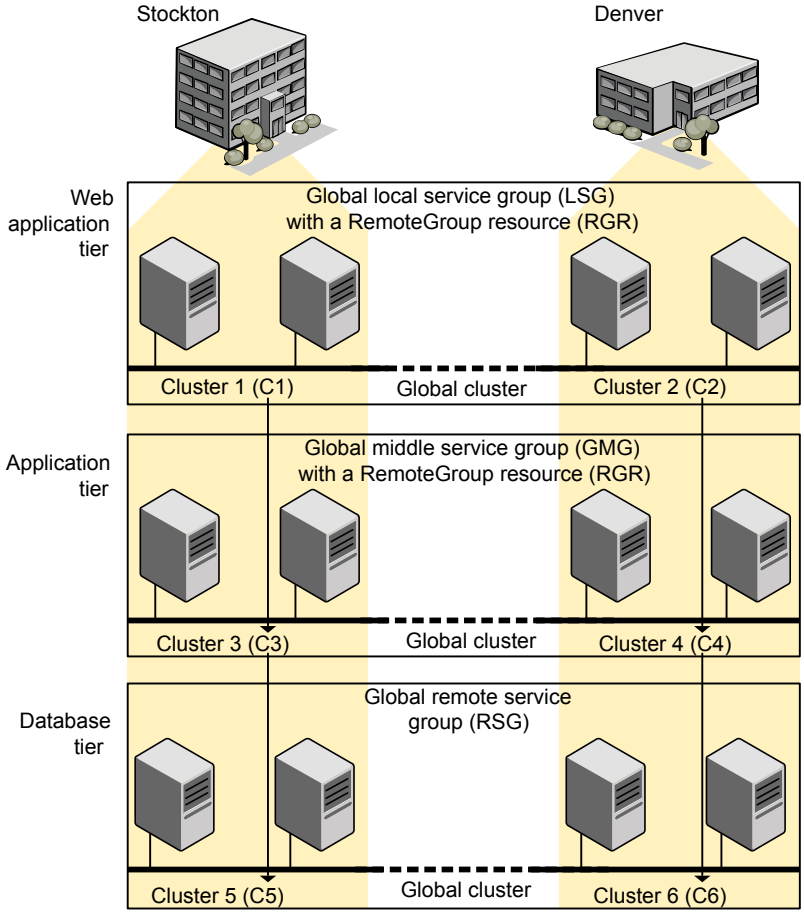
Figure 17-5 A VCS two-tiered globally clustered application and database



Just as a two-tier, two-site environment is possible, you can also tie a three-tier environment together.

Figure 17-6 represents a two-site, three-tier environment. The application cluster, which is globally clustered between L.A. and Denver, has cluster dependencies up and down the tiers. Cluster 1 (C1), depends on the RemoteGroup resource on the DB tier for cluster 3 (C3), and then on the remote service group for cluster 5 (C5). The stack for C2, C4, and C6 functions the same.

Figure 17-6 A three-tiered globally clustered application, database, and storage



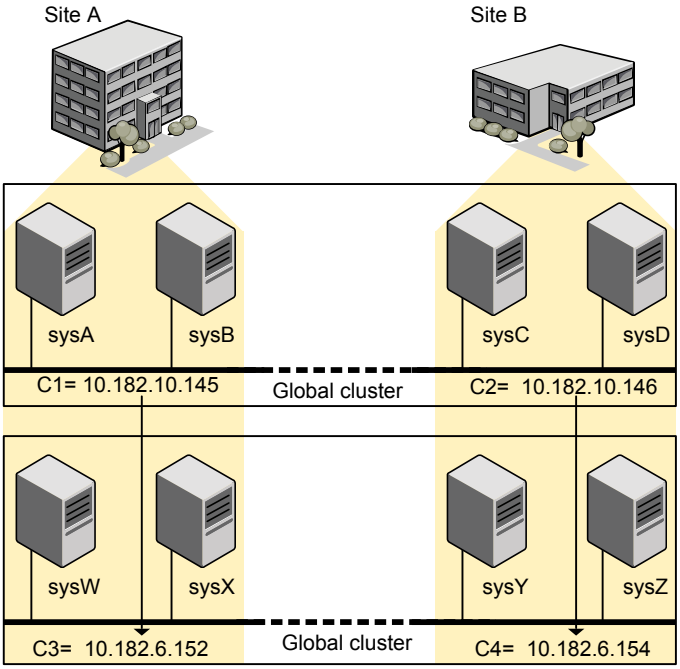
Test scenario for a multi-tiered environment

In the following scenario, eight systems reside in four clusters. Each tier contains a global cluster. The global local service group in the top tier depends on the global remote service group in the bottom tier.

The following main.cf files show this multi-tiered environment. The FileOnOff resource is used to test the dependencies between layers. Note that some attributes have been edited for clarity, and that these clusters are not running in secure mode.

Figure 17-7 shows the scenario for testing.

Figure 17-7 A VCS two-tiered globally clustered scenario



About the main.cf file for cluster 1

The contents of the main.cf file for cluster 1 (C1) in the top tier, containing the sysA and sysB nodes.

```
include "types.cf"

cluster C1 (
    ClusterAddress = "10.182.10.145"
    )

remotecluster C2 (
    ClusterAddress = "10.182.10.146"
    )

heartbeat Icmp (
    ClusterList = { C2 }
    AYATimeout = 30
    Arguments @C2 = { "10.182.10.146" }
```

```
    )

system sysA (
    )

system sysB (
    )

group LSG (
    SystemList = { sysA = 0, sysB = 1 }
    ClusterList = { C2 = 0, C1 = 1 }
    AutoStartList = { sysA, sysB }
    ClusterFailOverPolicy = Auto
    )

    FileOnOff filec1 (
        PathName = "/tmp/c1"
        )

    RemoteGroup RGR (
        IpAddress = "10.182.6.152"
        // The above IPAddress is the highly available address of C3—
        // the same address that the wac uses
        Username = root
        Password = xxxyyy
        GroupName = RSG
        VCSSysName = ANY
        ControlMode = OnOff
        )
```

About the main.cf file for cluster 2

The contents of the main.cf file for cluster 2 (C2) in the top tier, containing the sysC and sysD nodes.

```
include "types.cf"

cluster C2 (
    ClusterAddress = "10.182.10.146"
    )

remotecluster C1 (
    ClusterAddress = "10.182.10.145"
```

```
    )

heartbeat Icmp (
    ClusterList = { C1 }
    AYATimeout = 30
    Arguments @C1 = { "10.182.10.145" }
    )

system sysC (
    )

system sysD (
    )

group LSG (
    SystemList = { sysC = 0, sysD = 1 }
    ClusterList = { C2 = 0, C1 = 1 }
    Authority = 1
    AutoStartList = { sysC, sysD }
    ClusterFailOverPolicy = Auto
    )

    FileOnOff filec2 (
        PathName = filec2
        )

    RemoteGroup RGR (
        IpAddress = "10.182.6.154"
        // The above IPAddress is the highly available address of C4—
        // the same address that the wac uses
        Username = root
        Password = vvvyyy
        GroupName = RSG
        VCSSysName = ANY
        ControlMode = OnOff
        )
```

About the main.cf file for cluster 3

The contents of the main.cf file for cluster 3 (C3) in the bottom tier, containing the sysW and sysX nodes.

```
include "types.cf"

cluster C3 (
    ClusterAddress = "10.182.6.152"
    )

remotecluster C4 (
    ClusterAddress = "10.182.6.154"
    )

heartbeat Icmp (
    ClusterList = { C4 }
    AYATimeout = 30
    Arguments @C4 = { "10.182.6.154" }
    )

system sysW (
    )

system sysX (
    )

group RSG (
    SystemList = { sysW = 0, sysX = 1 }
    ClusterList = { C3 = 1, C4 = 0 }
    AutoStartList = { sysW, sysX }
    ClusterFailOverPolicy = Auto
    )

    FileOnOff filec3 (
        PathName = "/tmp/filec3"
        )
```

About the main.cf file for cluster 4

The contents of the main.cf file for cluster 4 (C4) in the bottom tier, containing the sysY and sysZ nodes.

```
include "types.cf"

cluster C4 (
    ClusterAddress = "10.182.6.154"
    )
```

```
remotecluster C3 (
    ClusterAddress = "10.182.6.152"
    )

heartbeat Icmp (
    ClusterList = { C3 }
    AYATimeout = 30
    Arguments @C3 = { "10.182.6.152" }
    )

system sysY (
    )

system sysZ (
    )

group RSG (
    SystemList = { sysY = 0, sysZ = 1 }
    ClusterList = { C3 = 1, C4 = 0 }
    Authority = 1
    AutoStartList = { sysY, sysZ }
    ClusterFailOverPolicy = Auto
    )

    FileOnOff filec4 (
        PathName = "/tmp/filec4"
        )
```

Chapter 18

Administering global clusters from the command line

This chapter includes the following topics:

- About administering global clusters from the command line
- About global querying in a global cluster setup
- Administering global service groups in a global cluster setup
- Administering resources in a global cluster setup
- Administering clusters in global cluster setup
- Administering heartbeats in a global cluster setup

About administering global clusters from the command line

For remote cluster operations, you must configure a VCS user with the same name and privileges in each cluster.

See "User privileges for cross-cluster operations" on page 89.

Review the following procedures to administer global clusters from the command-line.

See "About global querying in a global cluster setup" on page 537.

See "Administering global service groups in a global cluster setup" on page 544.

See "Administering resources in a global cluster setup" on page 546.

See "Administering clusters in global cluster setup" on page 546.

See "Administering heartbeats in a global cluster setup" on page 549.

About global querying in a global cluster setup

VCS enables you to query global cluster objects, including service groups, resources, systems, resource types, agents, and clusters. You may enter query commands from any system in the cluster. Commands to display information on the global cluster configuration or system states can be executed by all users; you do not need root privileges. Only global service groups may be queried.

See "Querying global cluster service groups" on page 537.

See "Querying resources across clusters" on page 538.

See "Querying systems" on page 540.

See "Querying clusters" on page 540.

See "Querying status" on page 542.

See "Querying heartbeats" on page 542.

Querying global cluster service groups

This topic describes how to perform a query on global cluster service groups:

To display service group attribute values across clusters

- Use the following command to display service group attribute values across clusters:

  ```
  hagrp -value service_group attribute [system]
  [-clus cluster | -localclus]
  ```

  The option `-clus` displays the attribute value on the cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

  If the attribute has local scope, you must specify the system name, except when querying the attribute on the system from which you run the command.

To display the state of a service group across clusters

- Use the following command to display the state of a service group across clusters:

```
hagrp -state [service_groups -sys systems]
[-clus cluster | -localclus]
```

The option `-clus` displays the state of all service groups on a cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

To display service group information across clusters

- Use the following command to display service group information across clusters:

```
hagrp -display [service_groups] [-attribute attributes]
[-sys systems] [-clus cluster | -localclus]
```

The option `-clus` applies to global groups only. If the group is local, the cluster name must be the local cluster name, otherwise no information is displayed.

To display service groups in a cluster

- Use the following command to display service groups in a cluster:

```
hagrp -list [conditionals] [-clus cluster | -localclus]
```

The option `-clus` lists all service groups on the cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

To display usage for the service group command

- Use the following command to display usage for the service group command:

```
hagrp [-help [-modify|-link|-list]]
```

Querying resources across clusters

This topic describes how to perform queries on resources:

To display resource attribute values across clusters

- Use the following command to display resource attribute values across clusters:

```
hares -value resource attribute [system]
[-clus cluster | -localclus]
```

The option `-clus` displays the attribute value on the cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

If the attribute has local scope, you must specify the system name, except when querying the attribute on the system from which you run the command.

To display the state of a resource across clusters

- Use the following command to display the state of a resource across clusters:

```
hares -state [resource -sys system]
[-clus cluster | -localclus]
```

The option `-clus` displays the state of all resources on the specified cluster; the option `-localclus` specifies the local cluster. Specifying a system displays resource state on a particular system.

To display resource information across clusters

- Use the following command to display resource information across clusters:

```
hares -display [resources] [-attribute attributes]
[-group service_groups] [-type types] [-sys systems]
[-clus cluster | -localclus]
```

The option `-clus` lists all service groups on the cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

To display a list of resources across clusters

- Use the following command to display a list of resources across clusters:

```
hares -list [conditionals] [-clus cluster | -localclus]
```

The option `-clus` lists all resources that meet the specified conditions in global service groups on a cluster as designated by the variable *cluster*.

To display usage for the resource command

- Use the following command to display usage for the resource command:

```
hares -help [-modify | -list]
```

Querying systems

This topic describes how to perform queries on systems:

To display system attribute values across clusters

- Use the following command to display system attribute values across clusters:

  hasys -value *system attribute* [-clus *cluster* | -localclus]

  The option `-clus` displays the values of a system attribute in the cluster as designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

To display the state of a system across clusters

- Use the following command to display the state of a system across clusters:

  hasys -state [*system*] [-clus *cluster* | -localclus]

  Displays the current state of the specified system. The option `-clus` displays the state in a cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster. If you do not specify a system, the command displays the states of all systems.

For information about each system across clusters

- Use the following command to display information about each system across clusters:

  hasys -display [*systems*] [-attribute *attributes*] [-clus *cluster* | -localclus]

  The option `-clus` displays the attribute values on systems (if specified) in a cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

For a list of systems across clusters

- Use the following command to display a list of systems across clusters:

  hasys -list [*conditionals*] [-clus *cluster* | -localclus]

  Displays a list of systems whose values match the given conditional statements. The option `-clus` displays the systems in a cluster designated by the variable *cluster*; the option `-localclus` specifies the local cluster.

Querying clusters

This topic describes how to perform queries on clusters:

For the value of a specific cluster attribute on a specific cluster

- Use the following command to obtain the value of a specific cluster attribute on a specific cluster:

  ```
  haclus -value attribute [cluster] [-localclus]
  ```

  The attribute must be specified in this command. If you do not specify the cluster name, the command displays the attribute value on the local cluster.

To display the state of a local or remote cluster

- Use the following command to display the state of a local or remote cluster:

  ```
  haclus -state [cluster] [-localclus]
  ```

  The variable *cluster* represents the cluster. If a cluster is not specified, the state of the local cluster and the state of all remote cluster objects as seen by the local cluster are displayed.

For information on the state of a local or remote cluster

- Use the following command for information on the state of a local or remote cluster:

  ```
  haclus -display [cluster] [-localclus]
  ```

  If a cluster is not specified, information on the local cluster is displayed.

For a list of local and remote clusters

- Use the following command for a list of local and remote clusters:

  ```
  haclus -list [conditionals]
  ```

  Lists the clusters that meet the specified conditions, beginning with the local cluster.

To display usage for the cluster command

- Use the following command to display usage for the cluster command:

  ```
  haclus [-help [-modify]]
  ```

To display the status of a faulted cluster

- Use the following command to display the status of a faulted cluster:

  haclus -status *cluster*

  Displays the status on the specified faulted cluster. If no cluster is specified, the command displays the status on all faulted clusters. It lists the service groups that were not in the OFFLINE or the FAULTED state before the fault occurred. It also suggests corrective action for the listed clusters and service groups.

Querying status

This topic describes how to perform queries on status of remote and local clusters:

For the status of local and remote clusters

- Use the following command to obtain the status of local and remote clusters:

  hastatus

Querying heartbeats

The hahb command is used to manage WAN heartbeats that emanate from the local cluster. Administrators can monitor the "health of the remote cluster via heartbeat commands and mechanisms such as Internet, satellites, or storage replication technologies. Heartbeat commands are applicable only on the cluster from which they are issued.

Note: You must have Cluster Administrator privileges to add, delete, and modify heartbeats.

The following commands are issued from the command line.

For a list of heartbeats configured on the local cluster

- Use the following command for a list of heartbeats configured on the local cluster:

  hahb -list [*conditionals*]

  The variable *conditionals* represents the conditions that must be met for the heartbeat to be listed.

To display information on heartbeats configured in the local cluster

- Use the following command to display information on heartbeats configured in the local cluster:

  ```
  hahb -display [heartbeat ...]
  ```

  If *heartbeat* is not specified, information regarding all heartbeats configured on the local cluster is displayed.

To display the state of the heartbeats in remote clusters

- Use the following command to display the state of heartbeats in remote clusters:

  ```
  hahb -state [heartbeat] [-clus cluster]
  ```

  For example, to get the state of heartbeat Icmp from the local cluster to the remote cluster phoenix:

  ```
  hahb -state Icmp -clus phoenix
  ```

To display an attribute value of a configured heartbeat

- Use the following command to display an attribute value of a configured heartbeat:

  ```
  hahb -value heartbeat attribute [-clus cluster]
  ```

  The `-value` option provides the value of a single attribute for a specific heartbeat. The cluster name must be specified for cluster-specific attribute values, but not for global.

  For example, to display the value of the ClusterList attribute for heartbeat Icmp:

  ```
  hahb -value Icmp ClusterList
  ```

  Note that ClusterList is a global attribute.

To display usage for the command `hahb`

- Use the following command to display usage for the command hahb:

  ```
  hahb [-help [-modify]]
  ```

  If the -modify option is specified, the usage for the `hahb -modify` option is displayed.

Administering global service groups in a global cluster setup

Operations for the VCS global clusters option are enabled or restricted depending on the permissions with which you log on. The privileges associated with each user role are enforced for cross-cluster, service group operations.

This topic includes commands to administer global service groups.

See the `hagrp` (1M) manual page for more information.

To administer global service groups in a global cluster setup

- Depending on the administrative task you want to perform on global service groups, run the `hagrp` command as follows:

To bring a service group online across clusters for the first time	**`hagrp -online -force`**
To bring a service group online across clusters	**`hagrp -online`** ***`service_group -sys system`*** **`[-clus`** ***`cluster`*** **`| -localclus]`** The option `-clus` brings the service group online on the system designated in the cluster. If a system is not specified, the service group is brought online on any node within the cluster. The option `-localclus` brings the service group online in the local cluster.
To bring a service group online on any node	**`hagrp -online [-force]`** ***`service_group`*** **`-any [-clus`** ***`cluster`*** **`| -localclus]`** The option `-any` specifies that HAD brings a failover group online on the optimal system, based on the requirements of service group workload management and existing group dependencies. If bringing a parallel group online, HAD brings the group online on each system designated in the SystemList attribute.
To display the resources for a service group	**`hagrp -resources`** ***`service_group`*** **`[-clus`** ***`cluster_name`*** **`| -localclus]`** The option `-clus` displays information for the cluster designated by the variable *cluster_name*; the option `-localclus` specifies the local cluster.

To take a service group offline across clusters	**hagrp -offline [-force] [-ifprobed] *service_group* -sys *system* [-clus *cluster* | -localclus]** The option `-clus` takes offline the service group on the system designated in the cluster.
To take a service group offline anywhere	**hagrp -offline [-ifprobed] *service_group* -any [-clus *cluster* | -localclus]** The option `-any` specifies that HAD takes a failover group offline on the system on which it is online. For a parallel group, HAD takes the group offline on each system on which the group is online. HAD adheres to the existing group dependencies when taking groups offline.
To switch a service group across clusters	**hagrp -switch *service_group* -to *system* [-clus *cluster* | -localclus [-nopre]]** The option `-clus` identifies the cluster to which the service group will be switched. The service group is brought online on the system specified by the -to system argument. If a system is not specified, the service group may be switched to any node within the specified cluster. The option `-nopre` indicates that the VCS engine must switch the service group regardless of the value of the PreSwitch service group attribute.
To switch a service group anywhere	**hagrp -switch *service_group* -any [-clus *cluster* | -localclus]** The `-any` option specifies that the VCS engine switches a service group to the best possible system on which it is currently not online, based on the value of the group's FailOverPolicy attribute. The VCS engine switches a global service group from a system to another system in the local cluster or a remote cluster. If you do not specify the `-clus` option, the VCS engine by default assumes `-localclus` option and selects an available system within the local cluster. The option `-clus` identifies the remote cluster to which the service group will be switched. The VCS engine then selects the target system on which to switch the service group.
To switch a parallel global service group across clusters	**hagrp -switch** VCS brings the parallel service group online on all possible nodes in the remote cluster.

Administering resources in a global cluster setup

This topic describes how to administer resources.

See the `hares` (1M) manual page for more information.

To administer resources in a global cluster setup

- Depending on the administrative task you want to perform for resources, run the `hares` command as follows:

To take action on a resource across clusters	**`hares -action`** ***`resource token`*** **`[-actionargs`** ***`arg1 ...`*** **`] [-`***`sys system`***`] [-clus`** ***`cluster`*** **`| -localclus]`** The option `-clus` implies resources on the cluster. If the designated system is not part of the local cluster, an error is displayed. If the `-sys` option is not used, it implies resources on the local node.
To invoke the Info function across clusters	**`hares -refreshinfo`** ***`resource`*** **`[-sys`** ***`system`***`] [-clus`** ***`cluster`*** **`| -localclus]`** Causes the Info function to update the value of the ResourceInfo resource level attribute for the specified resource if the resource is online. If no system or remote cluster is specified, the Info function runs on local system(s) where the resource is online.
To display usage for the resource command	**`hares [-help [-modify |-list]]`**

Administering clusters in global cluster setup

The topic includes commands that are used to administer clusters in a global cluster setup.

See the `haclus` (1M) manual page for more information.

To administer clusters in global cluster setup

- Depending on the administrative task you want to perform on the clusters, run the `haclus` command as follows:

  The variable *cluster* in the following commands represents the cluster.

To add a remote cluster object	**`haclus -add`** ***`cluster ip`*** This command does not apply to the local cluster.

To delete a remote cluster object	**haclus -delete *cluster***
To modify an attribute of a local or remote cluster object	**haclus -modify *attribute value* [-clus *cluster*]...**
To declare the state of a cluster after a disaster	**haclus -declare *disconnet/outage/disaster/replica* -clus *cluster* [-failover]**
To manage cluster alerts	See "Managing cluster alerts in a global cluster setup" on page 547.
To change the cluster name	See "Changing the cluster name in a global cluster setup" on page 548.

Managing cluster alerts in a global cluster setup

This topic includes commands to manage cluster alerts.

See the `haalert` (1M) manual page for more information.

To manage cluster alerts

◆ Run the `haalert` command to manage cluster alerts.

`haalert -testfd`	Generates a simulated "cluster fault" alert that is sent to the VCS engine and GUI.
`haalert -display`	For each alert, the command displays the following information: ■ alert ID ■ time when alert occurred ■ cluster on which alert occurred ■ object name for which alert occurred ■ (cluster name, group name, and so on). ■ informative message about alert
`haalert -list`	For each alert, the command displays the following information: ■ time when alert occurred ■ alert ID

`haalert -delete` *`alert_id`* `-notes` *`"description"`*	Deletes a specific alert. You must enter a text message within quotes describing the reason for deleting the alert. The comment is written to the engine log as well as sent to any connected GUI clients.
`haalert -help`	Displays the usage text

Changing the cluster name in a global cluster setup

This topic describes how to change the ClusterName attribute in a global cluster configuration. The instructions describe how to rename VCSPriCluster to VCSPriCluster2 in a two-cluster configuration, comprising clusters VCSPriCluster and VCSSecCluster configured with the global group AppGroup.

Before changing the cluster name, make sure the cluster is not part of any ClusterList, in the wide-area Heartbeat agent and in global service groups.

To change the name of a cluster

1. Run the following commands from cluster VCSPriCluster:

```
hagrp -offline ClusterService -any
hagrp -modify AppGroup ClusterList -delete VCSPriCluster
haclus -modify ClusterName VCSPriCluster2
hagrp -modify AppGroup ClusterList -add VCSPriCluster2 0
```

2. Run the following commands from cluster VCSSecCluster:

```
hagrp -offline ClusterService -any
hagrp -modify appgrp ClusterList -delete VCSPriCluster
hahb -modify Icmp ClusterList -delete VCSPriCluster
haclus -delete VCSPriCluster
haclus -add VCSPriCluster2 your_ip_address
hahb -modify Icmp ClusterList -add VCSPriCluster2
hahb -modify Icmp Arguments your_ip_address -clus VCSPriCluster2
hagrp -modify AppGroup ClusterList -add VCSPriCluster2 0
hagrp -online ClusterService -any
```

3. Run the following command from the cluster renamed to VCSPriCluster2:

```
hagrp -online ClusterService -any
```

Removing a remote cluster from a global cluster setup

This topic includes commands that are used to remove the remote cluster from the cluster list.

To remove a remote cluster in a global cluster setup

1. Run the following command:

   ```
   hagrp -offline ClusterService -any
   ```

2. Remove the remote cluster from the ClusterList of all the global groups using the following command:

   ```
   hagrp -modify <global_group> ClusterList -delete <remote_clus>
   ```

3. Remove the remote cluster from the ClusterList of all the heartbeats using the following command:

   ```
   hahb -modify Icmp ClusterList -delete <remote_clus>
   ```

4. Delete the remote cluster using the following command:

   ```
   haclus -delete <remote_clus>
   ```

Administering heartbeats in a global cluster setup

This topic includes commands that are used to administer heartbeats.

See the `hahb` (1M) manual page for more information.

To administer heartbeats in a global cluster setup

- Depending on the administrative task you want to perform for heartbeats, run the `hahb` command as follows:

To create a heartbeat	**`hahb -add heartbeat`** For example, type the following command to add a new IcmpS heartbeat. This represents a heartbeat sent from the local cluster and immediately forks off the specified agent process on the local cluster. **`hahb -add IcmpS`**

To modify a heartbeat	**hahb -modify *heartbeat attribute value* ... [-clus *cluster*]** If the attribute is local, that is, it has a separate value for each remote cluster in the ClusterList attribute, the option `-clus cluster` must be specified. Use `-delete -keys` to clear the value of any list attributes. For example, type the following command to modify the ClusterList attribute and specify targets "phoenix and "houston for the newly created heartbeat: **hahb -modify Icmp ClusterList phoenix houston** To modify the Arguments attribute for target phoenix: **hahb -modify Icmp Arguments phoenix.example.com -clus phoenix**
To delete a heartbeat	**hahb -delete *heartbeat***
To change the scope of an attribute to cluster-specific	**hahb -local *heartbeat attribute*** For example, type the following command to change the scope of the attribute AYAInterval from global to cluster-specific: **hahb -local Icmp AYAInterval**
To change the scope of an attribute to global	**hahb -global *heartbeat attribute value* ... | *key* ... | *key value* ...** For example, type the following command to change the scope of the attribute AYAInterval from cluster-specific to cluster-generic: **hahb -global Icmp AYAInterval 60**

Chapter 19

Setting up replicated data clusters

This chapter includes the following topics:

- About replicated data clusters
- How VCS replicated data clusters work
- About setting up a replicated data cluster configuration
- About migrating a service group
- About setting up a fire drill

About replicated data clusters

The Replicated Data Cluster (RDC) configuration provides both local high availability and disaster recovery functionality in a single VCS cluster.

You can set up RDC in a VCS environment using Volume Replicator (VVR).

A Replicated Data Cluster (RDC) uses data replication to assure data access to nodes. An RDC exists within a single VCS cluster. In an RDC configuration, if an application or a system fails, the application is failed over to another system within the current primary site. If the entire primary site fails, the application is migrated to a system in the remote secondary site (which then becomes the new primary).

For VVR replication to occur, the disk groups containing the Replicated Volume Group (RVG) must be imported at the primary and secondary sites. The replication service group must be online at both sites simultaneously, and must be configured as a hybrid VCS service group.

The application service group is configured as a failover service group. The application service group must be configured with an *online local hard* dependency on the replication service group.

Note: VVR supports multiple replication secondary targets for any given primary. However, RDC for VCS supports only one replication secondary for a primary.

An RDC configuration is appropriate in situations where dual dedicated LLT links are available between the primary site and the disaster recovery secondary site but lacks shared storage or SAN interconnect between the primary and secondary data centers. In an RDC, data replication technology is employed to provide node access to data in a remote site.

Note: You must use dual dedicated LLT links between the replicated nodes.

How VCS replicated data clusters work

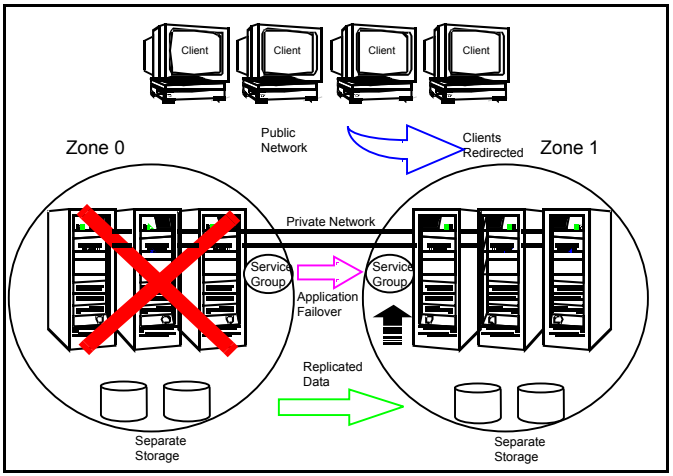
To understand how a replicated data cluster configuration works, let us take the example of an application configured in a VCS replicated data cluster. The configuration has two system zones:

- Primary zone (zone 0) comprising nodes located at the primary site and attached to the primary storage
- Secondary zone (zone 1) comprising nodes located at the secondary site and attached to the secondary storage

The application is installed and configured on all nodes in the cluster. Application data is located on shared disks within each RDC site and is replicated across RDC site to ensure data concurrency. The application service group is online on a system in the current primary zone and is configured to fail over in the cluster.

Figure 19-1 depicts an application configured on a VCS replicated data cluster.

Figure 19-1 A VCS replicated data cluster configuration



In the event of a system or application failure, VCS attempts to fail over the application service group to another system within the same RDC site. However, in the event that VCS fails to find a failover target node within the primary RDC site, VCS switches the service group to a node in the current secondary RDC site (zone 1).

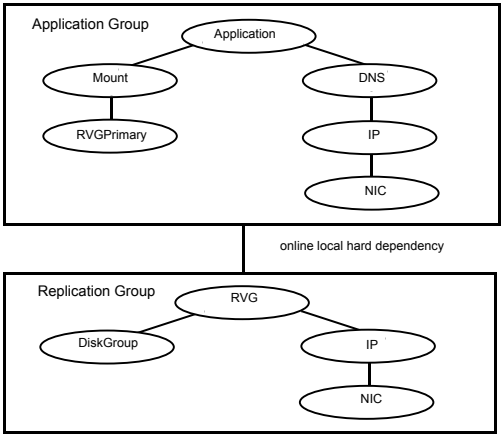
About setting up a replicated data cluster configuration

This topic describes the steps for planning, configuring, testing, and using the VCS RDC configuration to provide a robust and easy-to-manage disaster recovery protection for your applications. It describes an example of converting a single instance Oracle database configured for local high availability in a VCS cluster to a disaster-protected RDC infrastructure. The solution uses Volume Replicator to replicate changed data.

About typical replicated data cluster configuration

Figure 19-2 depicts a dependency chart of a typical RDC configuration.

Figure 19-2 Dependency chart of a typical RDC configuration



In this example, a single-instance application is configured as a VCS service group (DiskGroup) on a four-node cluster, with two nodes in the primary RDC system zone and two in the secondary RDC system zone. In the event of a failure on the primary node, VCS fails over the application to the second node in the primary zone.

The process involves the following steps:

- Setting Up Replication
- Configuring the Service Groups
- Configuring the Service Group Dependencies

About setting up replication

Veritas Volume Replicator (VVR) technology is a license-enabled feature of Veritas Volume Manager (VxVM), so you can convert VxVM-managed volumes into replicated volumes managed using VVR. In this example, the process involves grouping the Oracle data volumes into a Replicated Volume Group (RVG), and creating the VVR Secondary on hosts in another VCS cluster, located in your DR site.

When setting up VVR, it is a best practice to use the same DiskGroup and RVG name on both sites. If the volume names are the same on both zones, the Mount resources will mount the same block devices, and the same Oracle instance will start on the secondary in case of a failover.

Configuring the service groups

This topic describes how to configure service groups.

To configure the replication group

1. Create a hybrid service group (oragrp_rep) for replication.

   See "Types of service groups" on page 34.

2. Copy the DiskGroup resource from the application to the new group. Configure the resource to point to the disk group that contains the RVG.
3. Configure new resources of type IP and NIC.
4. Configure a new resource of type RVG in the service group.
5. Set resource dependencies as per the following information:
   - RVG resource depends on the IP resource
   - RVG resource depends on the DiskGroup resource
     IP resource depends on the NIC resource
6. Set the SystemZones attribute of the child group, oragrp_rep, such that all nodes in the primary RDC zone are in system zone 0 and all nodes in the secondary RDC zone are in system zone 1.

To configure the application service group

1. In the original Oracle service group (oragroup), delete the DiskGroup resource.
2. Add an RVGPrimary resource and configure its attributes.

   Set the value of the RvgResourceName attribute to the name of the RVG type resource that will be promoted and demoted by the RVGPrimary agent.

   Set the AutoTakeover and AutoResync attributes from their defaults as desired.

3. Set resource dependencies such that all Mount resources depend on the RVGPrimary resource. If there are a lot of Mount resources, you can set the TypeDependencies attribute for the group to denote that the Mount resource type depends on the RVGPRimary resource type.

4. Set the SystemZones attribute of the Oracle service group such that all nodes in the primary RDC zone are in system zone 0 and all nodes in the secondary RDC zone are in zone 1. The SystemZones attribute of both the parent and the child group must be identical.
5. If your setup uses BIND DNS, add a resource of type DNS to the oragroup service group. Set the Hostname attribute to the canonical name of the host or virtual IP address that the application uses on that cluster. This ensures DNS updates to the site when the group is brought online. A DNS resource would be necessary only if the nodes in the primary and the secondary RDC zones are in different IP subnets.

Configuring the service group dependencies

Set an online local hard group dependency from application service group to the replication service group to ensure that the service groups fail over and switch together.

1. In the Cluster Explorer configuration tree, select the cluster name.
2. In the view panel, click the **Service Groups** tab. This opens the service group dependency graph.
3. Click **Link**.
4. Click the parent group oragroup and move the mouse toward the child group, oragroup_rep.
5. Click the child group oragroup_rep.
6. On the Link Service Groups dialog box, click the online local relationship and the hard dependency type and click **OK**.

About migrating a service group

In the RDC set up for the Oracle database, consider a case where the primary RDC zone suffers a total failure of the shared storage. In this situation, none of the nodes in the primary zone see any device.

The Oracle service group cannot fail over locally within the primary RDC zone, because the shared volumes cannot be mounted on any node. So, the service group must fail over, to a node in the current secondary RDC zone.

The RVGPrimary agent ensures that VVR volumes are made writable and the DNS agent ensures that name services are resolved to the DR site. The application can be started at the DR site and run there until the problem with the local storage is corrected.

If the storage problem is corrected, you can switch the application to the primary site using VCS.

Switching the service group

Before switching the application back to the original primary RDC zone, you must resynchronize any changed data from the active DR site since the failover. This can be done manually through VVR or by running a VCS action from the RVGPrimary resource.

To switch the service group

1. In the **Service Groups** tab of the configuration tree, right-click the resource.
2. Click **Actions**.
3. Specify the details of the action as follows:
   - From the **Action** list, choose **fbsync**.
   - Click the system on which to execute the action.
   - Click **OK**.

   This begins a fast-failback of the replicated data set. You can monitor the value of the ResourceInfo attribute for the RVG resource to determine when the resynchronization has completed.
4. Once the resynchronization completes, switch the service group to the primary cluster. In the **Service Groups** tab of the of the Cluster Explorer configuration tree, right-click the service group.
5. Click **Switch To** and select the system in the primary RDC zone to switch to and click OK.

About setting up a fire drill

You can use fire drills to test the configuration's fault readiness by mimicking a failover without stopping the application in the primary data center.

See “About setting up a disaster recovery fire drill” on page 522.

Chapter 20

Setting up campus clusters

This chapter includes the following topics:

- About campus cluster configuration
- VCS campus cluster requirements
- Typical VCS campus cluster setup
- How VCS campus clusters work
- About setting up a campus cluster configuration
- Fire drill in campus clusters
- About the DiskGroupSnap agent
- About running a fire drill in a campus cluster

About campus cluster configuration

The campus cluster configuration provides local high availability and disaster recovery functionality in a single VCS cluster. This configuration uses data mirroring to duplicate data at different sites. There is no Host or Array base replication involved.

VCS supports campus clusters that employ disk groups mirrored with Veritas Volume Manager.

VCS campus cluster requirements

Review the following requirements for VCS campus clusters:

- You must install VCS.
- You must have a single VCS cluster with at least one node in each of the two sites, where the sites are separated by a physical distance of no more than 80 kilometers. When the sites are separated more than 80 kilometers, you can run Global Cluster Option (GCO) configuration.
- You must have redundant network connections between nodes. All paths to storage must also be redundant.

  Veritas recommends the following in a campus cluster setup:

  - A common cross-site physical infrastructure for storage and LLT private networks.

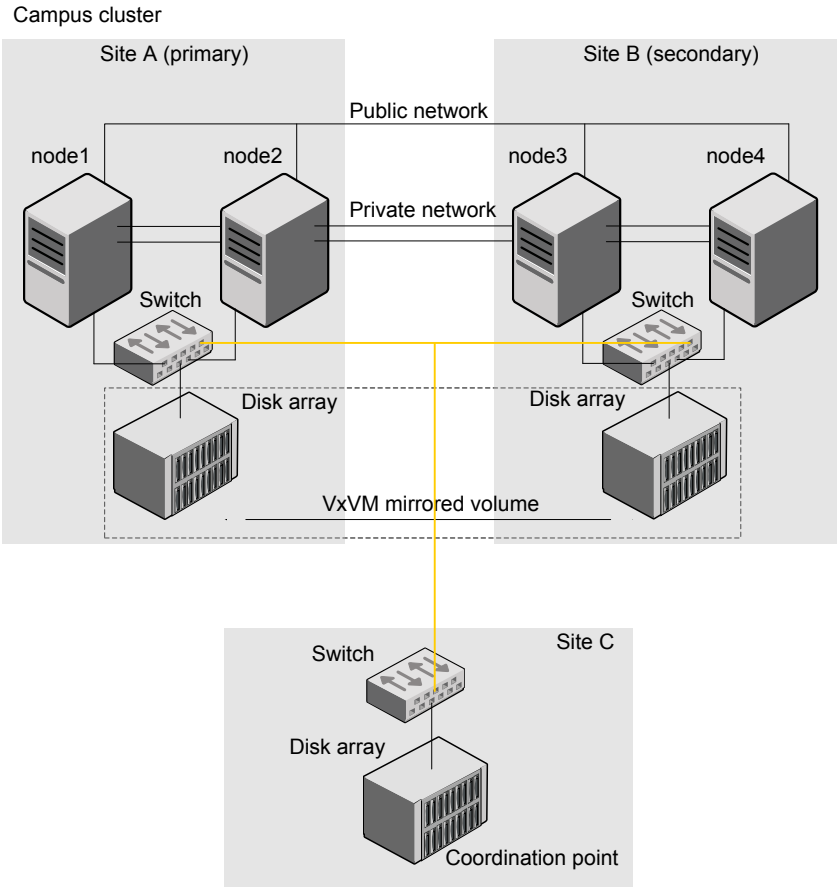
    Veritas recommends a common cross-site physical infrastructure for storage and LLT private networks
  - Technologies such as Dense Wavelength Division Multiplexing (DWDM) for network and I/O traffic across sites. Use redundant links to minimize the impact of network failure.
- You must install Veritas Volume Manager with the FMR license and the Site Awareness license.
- Veritas recommends that you configure I/O fencing to prevent data corruption in the event of link failures.

  See the Cluster Server Configuration and Upgrade Guide for more details.
- You must configure storage to meet site-based allocation and site-consistency requirements for VxVM.
  - All the nodes in the site must be tagged with the appropriate VxVM site names.
  - All the disks must be tagged with the appropriate VxVM site names.
  - The VxVM site names of both the sites in the campus cluster must be added to the disk groups.
  - The allsites attribute for each volume in the disk group must be set to on. (By default, the value is set to on.)
  - The siteconsistent attribute for the disk groups must be set to on.

Typical VCS campus cluster setup

Figure 20-1 depicts a typical VCS campus cluster setup.

Figure 20-1 Typical VCS campus cluster setup



VCS campus cluster typically has the following characteristics:

- Single VCS cluster spans multiple sites.
  In the sample figure, VCS is configured on four nodes: node 1 and node 2 are located at site A and node 3 and node 4 at site B.
- I/O fencing is configured with one coordinator disk from each site of the campus cluster and another coordinator disk from a third site.
  Figure 20-1 illustrates a typical setup with disk-based I/O fencing. You can also configure server-based I/O fencing.

See "About I/O fencing in campus clusters" on page 565.

- The shared data is located on mirrored volumes on a disk group configured using Veritas Volume Manager.
- The volumes that are required for the application have mirrors on both the sites.
- All nodes in the cluster are tagged with the VxVM site name. All disks that belong to a site are tagged with the corresponding VxVM site name.
- The disk group is configured in VCS as a resource of type DiskGroup and is mounted using the Mount resource type.

How VCS campus clusters work

This topic describes how VCS works with VxVM to provide high availability in a campus cluster environment.

In a campus cluster setup, VxVM automatically mirrors volumes across sites. To enhance read performance, VxVM reads from the plexes at the local site where the application is running. VxVM writes to plexes at both the sites.

In the event of a storage failure at a site, VxVM detaches all the disks at the failed site from the disk group to maintain data consistency. When the failed storage comes back online, VxVM automatically reattaches the site to the disk group and recovers the plexes.

See the *Storage Foundation Cluster File System High Availability Administrator's Guide* for more information.

When service group or system faults occur, VCS fails over service groups based on the values you set for the cluster attribute SiteAware and the service group attribute AutoFailOver.

See "Service group attributes" on page 719.

See "Cluster attributes" on page 760.

For campus cluster setup, you must define sites and add systems to the sites that you defined. A system can belong to only one site. Sit e definitions are uniform across VCS, You can define sites Veritas InfoScale Operations Manager, and VxVM. You can define site dependencies to restrict connected applications to fail over within the same site.

You can define sites by using:

- Veritas InfoScale Operations Manager
  For more information on configuring sites, see the latest version of the *Veritas InfoScale Operations Manager User guide*.

Depending on the value of the AutoFailOver attribute, VCS failover behavior is as follows:

0	VCS does not fail over the service group.
1	VCS fails over the service group to another suitable node. By default, the AutoFailOver attribute value is set to 1.
2	VCS fails over the service group if another suitable node exists in the same site. Otherwise, VCS waits for administrator intervention to initiate the service group failover to a suitable node in the other site. This configuration requires the HA/DR license enabled. Veritas recommends that you set the value of AutoFailOver attribute to 2.

Sample definition for these service group attributes in the VCS main.cf is as follows:

```
cluster VCS_CLUS (
        PreferredFencingPolicy = Site
        SiteAware = 1
        )
site MTV (
        SystemList = { sys1, sys2 }
        )
site SFO (
        Preference = 2
        SystemList = { sys3, sys4 }
        )
```

The sample configuration for hybrid_group with AutoFailover = 1 and failover_group with AutoFailover = 2 is as following:

```
hybrid_group (
    Parallel = 2
    SystemList = { sys1 = 0, sys2 = 1, sys3 = 2, sys4 = 3 }
)

failover_group (
    AutoFailover = 2
    SystemList = { sys1 = 0, sys2 = 1, sys3 = 2, sys4 = 3 }
)
```

Table 20-1 lists the possible failure scenarios and how VCS campus cluster recovers from these failures.

Table 20-1 Failure scenarios in campus cluster

Failure	Description and recovery
Node failure	■ A node in a site fails. If the value of the AutoFailOver attribute is set to 1, VCS fails over the service group to another system within the same site defined for cluster or SystemZone defined by SystemZones attribute for the service group or defined by Veritas InfoScale Operations Manager. ■ All nodes in a site fail. If the value of the AutoFailOver attribute is set to 0, VCS requires administrator intervention to initiate a fail over in both the cases of node failure.
Application failure	The behavior is similar to the node failure.
Storage failure - one or more disks at a site fails	VCS does not fail over the service group when such a storage failure occurs. VxVM detaches the site from the disk group if any volume in that disk group does not have at least one valid plex at the site where the disks failed. VxVM does not detach the site from the disk group in the following cases: ■ None of the plexes are configured on the failed disks. ■ Some of the plexes are configured on the failed disks, and at least one plex for a volume survives at each site. If only some of the disks that failed come online and if the vxrelocd daemon is running, VxVM relocates the remaining failed disks to any available disks. Then, VxVM automatically reattaches the site to the disk group and resynchronizes the plexes to recover the volumes. If all the disks that failed come online, VxVM automatically reattaches the site to the disk group and resynchronizes the plexes to recover the volumes.
Storage failure - all disks at both sites fail	VCS acts based on the DiskGroup agent's PanicSystemOnDGLoss attribute value. See the *Cluster Server Bundled Agents Reference Guide* for more information.

Table 20-1 Failure scenarios in campus cluster *(continued)*

Failure	Description and recovery
Site failure	All nodes and storage at a site fail. Depending on the value of the AutoFailOver attribute, VCS fails over the service group as follows: ■ If the value is set to 1, VCS fails over the service group to a system. ■ If the value is set to 2, VCS requires administrator intervention to initiate the service group failover to a system in the other site. Because the storage at the failed site is inaccessible, VCS imports the disk group in the application service group with all devices at the failed site marked as NODEVICE. When the storage at the failed site comes online, VxVM automatically reattaches the site to the disk group and resynchronizes the plexes to recover the volumes.
Network failure (LLT interconnect failure)	Nodes at each site lose connectivity to the nodes at the other site The failure of all private interconnects between the nodes can result in split brain scenario and cause data corruption. Review the details on other possible causes of split brain and how I/O fencing protects shared data from corruption. See "About data protection" on page 265. Veritas recommends that you configure I/O fencing to prevent data corruption in campus clusters. When the cluster attribute PreferredFencingPolicy is set as Site, the fencing driver gives preference to the node with higher site priority during the race for coordination points. VCS uses the site-level attribute Preference to determine the node weight. See "About I/O fencing in campus clusters" on page 565.

Table 20-1 Failure scenarios in campus cluster *(continued)*

Failure	Description and recovery
Network failure (LLT and storage interconnect failure)	Nodes at each site lose connectivity to the storage and the nodes at the other site Veritas recommends that you configure I/O fencing to prevent split brain and serial split brain conditions. ■ If I/O fencing is configured: The site that do not win the race triggers a system panic. See "About I/O fencing in campus clusters" on page 565. When you restore the network connectivity, VxVM detects the storage at the failed site, reattaches the site to the disk group, and resynchronizes the plexes to recover the volumes. ■ If I/O fencing is not configured: If the application service group was online at site A during such failure, the application service group remains online at the same site. Because the storage is inaccessible, VxVM detaches the disks at the failed site from the disk group. At site B where the application service group is offline, VCS brings the application service group online and imports the disk group with all devices at site A marked as NODEVICE. So, the application service group is online at both the sites and each site uses the local storage. This causes inconsistent data copies and leads to a site-wide split brain. When you restore the network connectivity between sites, a serial split brain may exist. See the *Storage Foundation Administrator's Guide* for details to recover from a serial split brain condition.

About I/O fencing in campus clusters

You must configure I/O fencing to prevent corruption of shared data in the event of a network partition.

See "About membership arbitration" on page 245.

See "About data protection" on page 265.

You can use coordinator disks, coordination point server (CP server), or a mix of both as coordination points to configure I/O fencing.

In a campus cluster setup, you can configure I/O fencing as follows:

- Two coordinator disks at one site and one coordinator disk at the other site
  In this case, the site that has two coordinator disks has a higher probability to win the race. The disadvantage with this configuration is that if the site that has two coordinator disks encounters a site failure, then the other site also commits

suicide. With this configuration, I/O fencing cannot distinguish between an inaccessible disk and a failed preempt operation.

- One coordinator disk in each of the two sites and a third coordinator disk at a third site
  This configuration ensures that fencing works even if one of the sites becomes unavailable. A coordinator disk in a third site allows at least a sub-cluster to continue operations in the event of a site failure in a campus cluster. The site that can access the coordinator disk in the third site in addition to its local coordinator disk wins the race. However, if both the sites of the campus cluster are unable to access the disk at the third site, each site gains one vote and the nodes at both the sites commit suicide.

You can also configure a coordination point server (CP server) at the third site instead of a third coordinator disk.

See the *Cluster Server Installation Guide* for more details to configure I/O fencing.

About setting up a campus cluster configuration

You must perform the following tasks to set up a campus cluster:

- Preparing to set up a campus cluster configuration
- Configuring I/O fencing to prevent data corruption
- Configuring VxVM disk groups for campus cluster configuration
- Configuring VCS service group for campus clusters

Preparing to set up a campus cluster configuration

Before you set up the configuration, review the VCS campus cluster requirements.

See " VCS campus cluster requirements" on page 559.

To prepare to set up a campus cluster configuration

1. Set up the physical infrastructure.
   - Set up access to the local storage arrays and to remote storage arrays on each node.
   - Set up private heartbeat network.

See " Typical VCS campus cluster setup" on page 560.

2 Install VCS on each node to form a cluster with at least one node in each of the two sites.

See the *Cluster Server Configuration and Upgrade Guide* for instructions.

3 Install VxVM on each node with the required licenses.

See the *Storage Foundation and High Availability Configuration and Upgrade Guide* for instructions.

Configuring I/O fencing to prevent data corruption

Perform the following tasks to configure I/O fencing to prevent data corruption in the event of a communication failure.

See "About I/O fencing in campus clusters" on page 565.

See the *Cluster Server Configuration and Upgrade Guide* for more details.

To configure I/O fencing to prevent data corruption

1 Set up the storage at a third site.

You can extend the DWDM to the third site to have FC SAN connectivity to the storage at the third site. You can also use iSCSI targets as the coordinator disks at the third site.

2 Set up I/O fencing.

Configuring VxVM disk groups for campus cluster configuration

Configure the campus cluster sites and configure VxVM disk groups for remote mirroring. You can also configure VxVM disk groups for remote mirroring using Veritas InfoScale Operations Manager.

Note: In cloud environments where one campus cluster is configured on one site and another on the second site, FSS volumes in campus cluster configurations can be used to replicate data for achieving high data availability across sites. In this FSS-campus cluster configuration, site tags can be added based on the name of the site to make data highly available during site failures.

See the *Storage Foundation Cluster File System High Availability Administrator's Guide* for more information on the VxVM commands.

To configure VxVM disk groups for campus cluster configuration

1. Set the site name for each host:

   ```
   # vxdctl set site=sitename
   ```

   The site name is stored in the /etc/vx/volboot file. Use the following command to display the site names:

   ```
   # vxdctl list | grep siteid
   ```

2. Set the site name for all the disks in an enclosure:

   ```
   # vxdisk settag site=sitename encl:enclosure
   ```

   To tag specific disks, use the following command:

   ```
   # vxdisk settag site=sitename disk
   ```

3. Verify that the disks are registered to a site.

   ```
   # vxdisk listtag
   ```

4. Create a disk group with disks from both the sites.

   ```
   # vxdg -s init diskgroup siteA_disk1 siteB_disk2
   ```

5. Configure site-based allocation on the disk group that you created for each site that is registered to the disk group.

   ```
   # vxdg -g diskgroup addsite sitename
   ```

6. Configure site consistency on the disk group.

   ```
   # vxdg -g diskgroup set siteconsistent=on
   ```

7. Create one or more mirrored volumes in the disk group.

   ```
   # vxassist -g diskgroup make volume size nmirror=1/2
   ```

   With the Site Awareness license installed on all hosts, the volume that you create has the following characteristics by default:

   - The `allsites` attribute is set to `on`; the volumes have at least one plex at each site.

- The volumes are automatically mirrored across sites.
- The read policy `rdpol` is set to `siteread`.
- The volumes inherit the site consistency value that is set on the disk group.

Configuring VCS service group for campus clusters

Follow the procedure to configure the disk groups under VCS control and set up the VCS attributes to define failover in campus clusters.

To configure VCS service groups for campus clusters

1. Create a VCS service group (app_sg) for the application that runs in the campus cluster.

```
hagrp -add app_sg
hagrp -modify app_sg SystemList node1 0 node2 1 node3 2 node4 3
```

2. Set up the system zones or sites. Configure the SystemZones attribute for the service group. Skip this step when sites are configured through Veritas InfoScale Operations Manager.

```
hagrp -modify app_sg SystemZones node1 0 node2 0 node3 1 node4 1
```

3. Set up the group fail over policy. Set the value of the AutoFailOver attribute for the service group.

```
hagrp -modify app_sg AutoFailOver 2
```

4. For the disk group you created for campus clusters, add a DiskGroup resource to the VCS service group app_sg.

```
hares -add dg_res1 DiskGroup app_sg
hares -modify dg_res1 DiskGroup diskgroup_name
hares -modify dg_res1 Enabled 1
```

5. Configure the application and other related resources to the app_sg service group.
6. Bring the service group online.

Fire drill in campus clusters

Fire drill tests the disaster-readiness of a configuration by mimicking a failover without stopping the application and disrupting user access.

The process involves creating a fire drill service group, which is similar to the original application service group. Bringing the fire drill service group online on the remote node demonstrates the ability of the application service group to fail over and come online at the site, should the need arise.

Fire drill service groups do not interact with outside clients or with other instances of resources, so they can safely come online even when the application service group is online. Conduct a fire drill only at the remote site; do not bring the fire drill service group online on the node hosting the original application.

About the DiskGroupSnap agent

The DiskGroupSnap agent verifies the VxVM disk groups and volumes for site awareness and disaster readiness in a campus cluster environment. To perform a fire drill in campus clusters, you must configure a resource of type DiskGroupSnap in the fire drill service group.

Note: To perform fire drill, the application service group must be online at the primary site.

During fire drill, the DiskGroupSnap agent does the following:

- For each node in a site, the agent correlates the value of the SystemZones attribute for the application service group to the VxVM site names for that node.
- For the disk group in the application service group, the agent verifies that the VxVM site tags are defined for the disk group.
- For the disk group in the application service group, the agent verifies that the disks at the secondary site are not tagged with the same VxVM site name as the disks at the primary site.
- The agent verifies that all volumes in the disk group have a plex at each site.

See the *Cluster Server Bundled Agents Reference Guide* for more information on the agent.

About running a fire drill in a campus cluster

This topic provides information on how to run a fire drill in campus clusters.

Do the following tasks to perform fire drill:

- Configuring the fire drill service group
- Running a successful fire drill in a campus cluster

Configuring the fire drill service group

This topic provides information on how to configure the fire drill service group.

To configure the fire drill service group

1. Configure a fire drill service group similar to the application service group with the following exceptions:
   - The AutoFailOver attribute must be set to 0.
   - Network-related resources must not be configured.
   - The disk group names for the DiskGroup and the Mount resources in the fire drill service group must be appended with "_fd".

     For example, if the value of the DiskGroup attribute in the application service group is ccdg, then the corresponding value in the fire drill service group must be ccdg_fd.

     If the value of the BlockDevice attribute for the Mount resource in the application service group is /dev/vx/dsk/ccdg/ccvol, then the corresponding value in the fire drill service group must be /dev/vx/dsk/ccdg_fd/ccvol.
2. Add a resource of type DiskGroupSnap. Define the TargetResName and the FDSiteName attributes for the DiskGroupSnap resource.

   See the *Cluster Server Bundled Agent Reference Guide* for attribute descriptions.
3. Create a dependency such that the DiskGroup resource depends on the DiskGroupSnap resource.
4. Create a group dependency such that the fire drill service group has an offline local dependency on the application service group.

Running a successful fire drill in a campus cluster

Bring the fire drill service group online on a node within the system zone that does not have the application running. Verify that the fire drill service group comes online. This action validates that your solution is configured correctly and the production service group will fail over to the remote site in the event of an actual failure (disaster) at the local site.

You must take the fire drill service group offline before you shut down the node or stop VCS locally on the node where the fire drill service group is online or where the disk group is online. Otherwise, after the node restarts you must manually reattach the fire drill site to the disk group that is imported at the primary site.

Note: For the applications for which you want to perform fire drill, you must set the value of the FireDrill attribute for those application resource types to 1. After you complete fire drill, reset the value to 0.

To run a successful fire drill

1. Override the FireDrill attribute at the resource level for resources of the fire drill service group.
2. Set the FireDrill attribute for the application resources to 1. This prevents the agent from reporting a concurrency violation, when the application service group and the fire drill service group are online at the same time.
3. Bring the fire drill service group online.

   If the fire drill service group does not come online, review the VCS engine log to troubleshoot the issues so that corrective action can be taken as necessary in the production service group.

   Warning: You must take the fire drill service group offline after you complete the fire drill so that the failover behavior of the application service group is not impacted. Otherwise, when a disaster strikes at the primary site, the application service group cannot fail over to the secondary site due to resource conflicts.

4. After you complete the fire drill, take the fire drill service group offline.
5. Reset the Firedrill attribute for the resource to 0.
6. Undo or override the Firedrill attribute from the resource level.

Section 7

Troubleshooting and performance

- Chapter 21. VCS performance considerations
- Chapter 22. Troubleshooting and recovery for VCS

Chapter 21

VCS performance considerations

This chapter includes the following topics:

- How cluster components affect performance
- How cluster operations affect performance
- About scheduling class and priority configuration
- VCS agent statistics
- About VCS tunable parameters

How cluster components affect performance

VCS and its agents run on the same systems as the applications. Therefore, VCS attempts to minimize its impact on overall system performance. The main components of clustering that have an impact on performance include the kernel; specifically, GAB and LLT, the VCS engine (HAD), and the VCS agents. For details on attributes or commands mentioned in the following sections, see the chapter on administering VCS from the command line and the appendix on VCS attributes.

See "How kernel components (GAB and LLT) affect performance" on page 575.

See " How the VCS engine (HAD) affects performance" on page 575.

See " How agents affect performance" on page 576.

See "How the VCS graphical user interfaces affect performance" on page 577.

How kernel components (GAB and LLT) affect performance

Typically, overhead of VCS kernel components is minimal. Kernel components provide heartbeat and atomic information exchange among cluster systems. By default, each system in the cluster sends two small heartbeat packets per second to other systems in the cluster.

Heartbeat packets are sent over all network links configured in the `/etc/llttab` configuration file.

System-to-system communication is load-balanced across all private network links. If a link fails, VCS continues to use all remaining links. Typically, network links are private and do not increase traffic on the public network or LAN. You can configure a public network (LAN) link as low-priority, which by default generates a small (approximately 64-byte) unicast packet per second from each system, and which will carry data only when all private network links have failed.

How the VCS engine (HAD) affects performance

The VCS engine, HAD, runs as a daemon process. By default it runs as a high-priority process, which ensures it sends heartbeats to kernel components and responds quickly to failures. HAD runs logging activities in a separate thread to reduce the performance impact on the engine due to logging.

VCS runs in a loop waiting for messages from agents, `ha` commands, the graphical user interfaces, and the other systems. Under normal conditions, the number of messages processed by HAD is few. They mainly include heartbeat messages from agents and update messages from the global counter. VCS may exchange additional messages when an event occurs, but typically overhead is nominal even during events. Note that this depends on the type of event; for example, a resource fault may involve taking the group offline on one system and bringing it online on another system. A system fault invokes failing over all online service groups on the faulted system.

To continuously monitor VCS status, use the VCS graphical user interfaces or the command `hastatus`. Both methods maintain connection to VCS and register for events, and are more efficient compared to running commands like `hastatus -summary` or `hasys` in a loop.

The number of clients connected to VCS can affect performance if several events occur simultaneously. For example, if five GUI processes are connected to VCS, VCS sends state updates to all five. Maintaining fewer client connections to VCS reduces this overhead.

How agents affect performance

The VCS agent processes have the most impact on system performance. Each agent process has two components: the agent framework and the agent functions. The agent framework provides common functionality, such as communication with the HAD, multithreading for multiple resources, scheduling threads, and invoking functions. Agent functions implement agent-specific functionality. Review the performance guidelines to follow when configuring agents.

See "Monitoring resource type and agent configuration" on page 576.

Monitoring resource type and agent configuration

By default, VCS monitors each resource every 60 seconds. You can change this by modifying the MonitorInterval attribute for the resource type. You may consider reducing monitor frequency for non-critical or resources with expensive monitor operations. Note that reducing monitor frequency also means that VCS may take longer to detect a resource fault.

By default, VCS also monitors offline resources. This ensures that if someone brings the resource online outside of VCS control, VCS detects it and flags a concurrency violation for failover groups. To reduce the monitoring frequency of offline resources, modify the OfflineMonitorInterval attribute for the resource type.

The VCS agent framework uses multithreading to allow multiple resource operations to run in parallel for the same type of resources. For example, a single Mount agent handles all mount resources. The number of agent threads for most resource types is 10 by default. To change the default, modify the NumThreads attribute for the resource type. The maximum value of the NumThreads attribute is 30.

Continuing with this example, the Mount agent schedules the `monitor` function for all mount resources, based on the MonitorInterval or OfflineMonitorInterval attributes. If the number of mount resources is more than NumThreads, the monitor operation for some mount resources may be required to wait to execute the `monitor` function until the thread becomes free.

Additional considerations for modifying the NumThreads attribute include:

- If you have only one or two resources of a given type, you can set NumThreads to a lower value.
- If you have many resources of a given type, evaluate the time it takes for the `monitor` function to execute and the available CPU power for monitoring. For example, if you have 50 mount points, you may want to increase NumThreads to get the ideal performance for the Mount agent without affecting overall system performance.

You can also adjust how often VCS monitors various functions by modifying their associated attributes. The attributes MonitorTimeout, OnlineTimeOut, and OfflineTimeout indicate the maximum time (in seconds) within which the monitor, online, and offline functions must complete or else be terminated. The default for the MonitorTimeout attribute is 60 seconds. The defaults for the OnlineTimeout and OfflineTimeout attributes is 300 seconds. For best results, Veritas recommends measuring the time it takes to bring a resource online, take it offline, and monitor before modifying the defaults. Issue an online or offline command to measure the time it takes for each action. To measure how long it takes to monitor a resource, fault the resource and issue a probe, or bring the resource online outside of VCS control and issue a probe.

Agents typically run with normal priority. When you develop agents, consider the following:

- If you write a custom agent, write the monitor function using C or C++. If you write a script-based monitor, VCS must invoke a new process each time with the monitor. This can be costly if you have many resources of that type.
- If monitoring the resources is proving costly, you can divide it into cursory, or shallow monitoring, and the more extensive deep (or in-depth) monitoring. Whether to use shallow or deep monitoring depends on your configuration requirements.

As an additional consideration for agents, properly configure the attribute SystemList for your service group. For example, if you know that a service group can go online on SystemA and SystemB only, do not include other systems in the SystemList. This saves additional agent processes and monitoring overhead.

How the VCS graphical user interfaces affect performance

The VCS graphical user interface, Cluster Manager (Java Console) maintains a persistent connection to HAD, from which it receives regular updates regarding cluster status. For best results, run the GUI on a system outside the cluster to avoid impact on node performance.

How cluster operations affect performance

Review the following topics that describe how the following operations on systems, resources, and service groups in the cluster affect performance:

A cluster system boots	See " VCS performance consideration when booting a cluster system" on page 578.

A resource comes online	See " VCS performance consideration when a resource comes online" on page 579.
A resource goes offline	See " VCS performance consideration when a resource goes offline" on page 579.
A service group comes online	See "VCS performance consideration when a service group comes online" on page 579.
A service group goes offline	See "VCS performance consideration when a service group goes offline" on page 580.
A resource fails	See " VCS performance consideration when a resource fails" on page 580.
A system fails	See " VCS performance consideration when a system fails" on page 581.
A network link fails	See " VCS performance consideration when a network link fails" on page 582.
A system panics	See " VCS performance consideration when a system panics" on page 582.
A service group switches over	See " VCS performance consideration when a service group switches over" on page 585.
A service group fails over	See " VCS performance consideration when a service group fails over" on page 585.

VCS performance consideration when booting a cluster system

When a cluster system boots, the kernel drivers and VCS process start in a particular order. If it is the first system in the cluster, VCS reads the cluster configuration file main.cf and builds an in-memory configuration database. This is the LOCAL_BUILD state. After building the configuration database, the system transitions into the RUNNING mode. If another system joins the cluster while the first system is in the LOCAL_BUILD state, it must wait until the first system transitions into RUNNING mode. The time it takes to build the configuration depends on the number of service groups in the configuration and their dependencies, and the number of resources per group and resource dependencies. VCS creates an object for each system, service group, type, and resource. Typically, the number of systems, service groups and types are few, so the number of resources and resource dependencies determine how long it takes to build the configuration database and get VCS into RUNNING mode. If a system joins a cluster in which at least one system is in RUNNING mode, it builds the configuration from the lowest-numbered system in that mode.

Note: Bringing service groups online as part of AutoStart occurs after VCS transitions to RUNNING mode.

VCS performance consideration when a resource comes online

The online function of an agent brings the resource online. This function may return before the resource is fully online. The subsequent monitor determines if the resource is online, then reports that information to VCS. The time it takes to bring a resource online equals the time for the resource to go online, plus the time for the subsequent monitor to execute and report to VCS.

Most resources are online when the online function finishes. The agent schedules the monitor immediately after the function finishes, so the first monitor detects the resource as online. However, for some resources, such as a database server, recovery can take longer. In this case, the time it takes to bring a resource online depends on the amount of data to recover. It may take multiple monitor intervals before a database server is reported online. When this occurs, it is important to have the correct values configured for the OnlineTimeout and OnlineWaitLimit attributes of the database server resource type.

VCS performance consideration when a resource goes offline

Similar to the online function, the offline function takes the resource offline and may return before the resource is actually offline. Subsequent monitoring confirms whether the resource is offline. The time it takes to offline a resource equals the time it takes for the resource to go offline, plus the duration of subsequent monitoring and reporting to VCS that the resource is offline. Most resources are typically offline when the offline function finishes. The agent schedules the monitor immediately after the offline function finishes, so the first monitor detects the resource as offline.

VCS performance consideration when a service group comes online

The time it takes to bring a service group online depends on the number of resources in the service group, the service group dependency structure, and the time to bring the group's resources online. For example, if service group G1 has three resources, R1, R2, and R3 (where R1 depends on R2 and R2 depends on R3), VCS first onlines R3. When R3 is online, VCS onlines R2. When R2 is online, VCS onlines R1. The time it takes to online G1 equals the time it takes to bring all resources online. However, if R1 depends on both R2 and R3, but there was no dependency between them, the online operation of R2 and R3 is started in parallel. When both are online, R1 is brought online. The time it takes to online the group is Max (the time to online R2 and R3), plus the time to online R1. Typically, broader service group trees allow more parallel operations and can be brought online faster. More

complex service group trees do not allow much parallelism and serializes the group online operation.

VCS performance consideration when a service group goes offline

Taking service groups offline works from the top down, as opposed to the online operation, which works from the bottom up. The time it takes to offline a service group depends on the number of resources in the service group and the time to offline the group's resources. For example, if service group G1 has three resources, R1, R2, and R3 where R1 depends on R2 and R2 depends on R3, VCS first offlines R1. When R1 is offline, VCS offlines R2. When R2 is offline, VCS offlines R3. The time it takes to offline G1 equals the time it takes for all resources to go offline.

VCS performance consideration when a resource fails

The time it takes to detect a resource fault or failure depends on the MonitorInterval attribute for the resource type. When a resource faults, the next monitor detects it. The agent may not declare the resource as faulted if the ToleranceLimit attribute is set to non-zero. If the `monitor` function reports offline more often than the number set in ToleranceLimit, the resource is declared faulted. However, if the resource remains online for the interval designated in the ConfInterval attribute, previous reports of offline are not counted against ToleranceLimit.

When the agent determines that the resource is faulted, it calls the clean function (if implemented) to verify that the resource is completely offline. The monitor following clean verifies the offline. The agent then tries to restart the resource according to the number set in the RestartLimit attribute (if the value of the attribute is non-zero) before it gives up and informs HAD that the resource is faulted. However, if the resource remains online for the interval designated in ConfInterval, earlier attempts to restart are not counted against RestartLimit.

In most cases, ToleranceLimit is 0. The time it takes to detect a resource failure is the time it takes the agent monitor to detect failure, plus the time to clean up the resource if the clean function is implemented. Therefore, the time it takes to detect failure depends on the MonitorInterval, the efficiency of the monitor and clean (if implemented) functions, and the ToleranceLimit (if set).

In some cases, the failed resource may hang and may also cause the monitor to hang. For example, if the database server is hung and the monitor tries to query, the monitor will also hang. If the `monitor` function is hung, the agent eventually kills the thread running the function. By default, the agent times out the `monitor` function after 60 seconds. This can be adjusted by changing the MonitorTimeout attribute. The agent retries monitor after the MonitorInterval. If the `monitor` function times out consecutively for the number of times designated in the attribute FaultOnMonitorTimeouts, the agent treats the resource as faulted. The agent calls

clean, if implemented. The default value of FaultOnMonitorTimeouts is 4, and can be changed according to the type. A high value of this parameter delays detection of a fault if the resource is hung. If the resource is hung and causes the `monitor` function to hang, the time to detect it depends on MonitorTimeout, FaultOnMonitorTimeouts, and the efficiency of `monitor` and clean (if implemented).

VCS performance consideration when a system fails

When a system crashes or is powered off, it stops sending heartbeats to other systems in the cluster. By default, other systems in the cluster wait 21 seconds before declaring it dead. The time of 21 seconds derives from 16 seconds default timeout value for LLT peer inactive timeout, plus 5 seconds default value for GAB stable timeout.

The default peer inactive timeout is 16 seconds, and can be modified in the `/etc/llttab` file.

For example, to specify 12 seconds:

```
set-timer peerinact:1200
```

Note: After modifying the peer inactive timeout, you must unconfigure, then restart LLT before the change is implemented. To unconfigure LLT, type `lltconfig -U`. To restart LLT, type `lltconfig -c`.

GAB stable timeout can be changed by specifying:

```
gabconfig -t timeout_value_milliseconds
```

Though this can be done, we do not recommend changing the values of the LLT peer inactive timeout and GAB stable timeout.

If a system boots, it becomes unavailable until the reboot is complete. The reboot process kills all processes, including HAD. When the VCS process is killed, other systems in the cluster mark all service groups that can go online on the rebooted system as autodisabled. The AutoDisabled flag is cleared when the system goes offline. As long as the system goes offline within the interval specified in the ShutdownTimeout value, VCS treats this as a system reboot. You can modify the default value of the ShutdownTimeout attribute.

See System attributes on page 746.

VCS performance consideration when a network link fails

If a system loses a network link to the cluster, other systems stop receiving heartbeats over the links from that system. LLT detects this and waits for 16 seconds before declaring the system lost a link.

See " VCS performance consideration when a system fails" on page 581.

You can modify the LLT peer inactive timeout value in the `/etc/llttab` file.

For example, to specify 12 seconds:

```
set-timer peerinact:1200
```

Note: After modifying the peer inactive timeout, you must unconfigure, then restart LLT before the change is implemented. To unconfigure LLT, type `lltconfig -U`. To restart LLT, type `lltconfig -c`.

VCS performance consideration when a system panics

There are several instances in which GAB will intentionally panic a system. For example, GAB panics a system if it detects an internal protocol error or discovers an LLT node-ID conflict. Other instances are as follows:

- Client process failure
  See "About GAB client process failure" on page 582.
- Registration monitoring
  See "About GAB client registration monitoring" on page 584.
- Network failure
  See "About network failure and GAB IOFENCE message" on page 585.
- Quick reopen
  See "About quick reopen" on page 585.

About GAB client process failure

If a GAB client process such as HAD fails to heartbeat to GAB, the process is killed. If the process hangs in the kernel and cannot be killed, GAB halts the system. If the `-k` option is used in the `gabconfig` command, GAB tries to kill the client process until successful, which may have an effect on the entire cluster. If the `-b` option is used in `gabconfig`, GAB does not try to kill the client process. Instead, it panics the system when the client process fails to heartbeat. This option cannot be turned off once set.

HAD heartbeats with GAB at regular intervals. It registers with GAB for a heartbeat timeout of 30 seconds (default value). You can configure the VCS environment variables, VCS_GAB_TIMEOUT_SECS andVCS_GAB_PEAKLOAD_TIMEOUT_SECS, to ensure that GAB exhibits a dynamic behavior to determine the load average of a node (per CPU load). Using the variable values and the average system load, GAB decides the grace period after which it kills HAD.

If the average load on the node is minimum and HAD hangs in the kernel such that it cannot heartbeat with GAB within the VCS_GAB_TIMEOUT_SECS timeout, GAB tries to kill HAD by sending a SIGABRT signal. Upon an unsuccessful attempt, GAB retries till the number of retries reaches the gab_kill_ntries-1 value. In case GAB cannot kill HAD with a SIGABRT signal, GAB sends a SIGKILL and closes the port. When the average load is minimum, GAB does not dynamically adapt to the load and hence does not consider the VCS_GAB_PEAKLOAD_TIMEOUT_SECS timeout value to determine the grace period to keep HAD alive.

If the average load on the node is high, HAD cannot communicate with GAB because of CPU load or delays in its I/O path with file systems. Depending on the average load, the operating system sends a load average number to GAB. The load average number ranges from 5 (minimum load) through 10 (maximum load). GAB uses the load average number to compute a grace period that adapts exponentially based on the load within the user specified bounds of the VCS_GAB_TIMEOUT_SECS and VCS_GAB_PEAKLOAD_TIMEOUT_SECS variables. GAB waits for HAD to send heartbeats during the grace period after which it kills HAD by sending a SIGABRT signal. Even after a SIGABRT signal, if GAB does not succeed, it sends a SIGKILL and closes the port.

Tunables considered by GAB to calculate the timeout period for HAD:

- GAB considers the VCS_GAB_TIMEOUT_SECS timeout to calculate the timeout period for HAD if both the VCS_GAB_TIMEOUT_SECS and VCS_GAB_TIMEOUT timeouts are set.
- GAB considers the VCS_GAB_TIMEOUT timeout if the VCS_GAB_TIMEOUT_SECS timeout is not set.
- GAB cannot exponentially adapt to determine the grace period for HAD if the VCS_GAB_PEAKLOAD_TIMEOUT_SECS timeout is not set or if its value is the same as the VCS_GAB_TIMEOUT_SECS timeout.

By default, GAB tries to kill HAD five times before closing the port. The number of times GAB tries to kill HAD is a kernel tunable parameter, gab_kill_ntries, and is configurable. The minimum value for this tunable is 3 and the maximum is 10.

Port closure is an indication to other nodes that HAD on this node has been killed. Should HAD recover from its stuck state, it first processes pending signals. Here it receive the SIGKILL first and get killed.

After GAB sends a SIGKILL signal, it waits for a specific amount of time for HAD to get killed. If HAD survives beyond this time limit, GAB panics the system. This time limit is a kernel tunable parameter, gab_isolate_time, and is configurable. The minimum value for this timer is 16 seconds and maximum is 4 minutes.

About GAB client registration monitoring

The registration monitoring feature lets you configure GAB behavior when HAD is killed and does not reconnect after a specified time interval.

This scenario may occur in the following situations:

- The system is very busy and the hashadow process cannot restart HAD.
- The HAD and hashadow processes were killed by user intervention.
- The hashadow process restarted HAD, but HAD could not register.
- A hardware failure causes termination of the HAD and hashadow processes.
- Any other situation where the HAD and hashadow processes are not run.

When this occurs, the registration monitoring timer starts. GAB takes action if HAD does not register within the time defined by the VCS_GAB_RMTIMEOUT parameter, which is defined in the `vcsenv` file. The default value for VCS_GAB_RMTIMEOUT is 200 seconds.

When HAD cannot register after the specified time period, GAB logs a message every 15 seconds saying it will panic the system.

You can control GAB behavior in this situation by setting the VCS_GAB_RMACTION parameter in the vcsenv file.

- To configure GAB to panic the system in this situation, set:

  ```
  VCS_GAB_RMACTION=panic
  ```

  In this configuration, killing the HAD and hashadow processes results in a panic unless you start HAD within the registration monitoring timeout interval.

- To configure GAB to log a message in this situation, set:

  ```
  VCS_GAB_RMACTION=SYSLOG
  ```

  The default value of this parameter is SYSLOG, which configures GAB to log a message when HAD does not reconnect after the specified time interval.
  In this scenario, you can choose to restart HAD (using hastart) or unconfigure GAB (using `gabconfig -U`).

When you enable registration monitoring, GAB takes no action if the HAD process unregisters with GAB normally, that is if you stop HAD using the hastop command.

About network failure and GAB IOFENCE message

If a network partition occurs, a cluster can split into two or more separate sub-clusters. When two clusters join as one, GAB ejects one sub-cluster. GAB prints diagnostic messages and sends iofence messages to the sub-cluster being ejected.

The systems in the sub-cluster process the iofence messages depending on the type of GAB port that a user client process or a kernel module uses:

- If the GAB client is a user process, then GAB tries to kill the client process.
- If the GAB client is a kernel module, then GAB panics the system.

The `gabconfig` command's `-k` and `-j` options apply to the user client processes. The `-k` option prevents GAB from panicking the system when it cannot kill the user processes. The `-j` option panics the system and does not kill the user process when GAB receives the iofence message.

About quick reopen

If a system leaves cluster and tries to join the cluster before the new cluster is configured (default is five seconds), the system is sent an iofence message with reason set to "quick reopen”. When the system receives the message, it tries to kill the client process.

VCS performance consideration when a service group switches over

The time it takes to switch a service group equals the time to offline a service group on the source system, plus the time to bring the service group online on the target system.

VCS performance consideration when a service group fails over

The time it takes to fail over a service group when a resource faults equals the following:

- The time it takes to detect the resource fault
- The time it takes to offline the service group on source system
- The time it takes for the VCS policy module to select target system
- The time it takes to bring the service group online on target system

The time it takes to fail over a service group when a system faults equals the following:

- The time it takes to detect system fault

- The time it takes to offline the dependent service groups on other running systems
- The time it takes for the VCS policy module to select target system
- The time it takes to bring the service group online on target system

The time it takes the VCS policy module to determine the target system is negligible in comparison to the other factors.

If you have a firm group dependency and the child group faults, VCS offlines all immediate and non-immediate parent groups before bringing the child group online on the target system. Therefore, the time it takes a parent group to be brought online also depends on the time it takes the child group to be brought online.

About scheduling class and priority configuration

VCS allows you to specify priorities and scheduling classes for VCS processes. VCS supports the following scheduling classes:

- RealTime (specified as "RT in the configuration file)
- TimeSharing (specified as "TS in the configuration file)

About priority ranges

Table 21-1 displays the platform-specific priority range for RealTime, TimeSharing, and SRM scheduling (SHR) processes.

Table 21-1 Priority ranges

Platform	Scheduling class	Default priority range wWeak / strong	Priority range using #ps commands
Linux	RT TS	1/ 99	L-high priority task N-high priority task **Note:** On Linux, use `#ps -ael`

Default scheduling classes and priorities

Table 21-2 lists the default class and priority values used by VCS. The class and priority of trigger processes are determined by the attributes ProcessClass (default = TS) and ProcessPriority (default = ""). Both attributes can be modified according to the class and priority at which the trigger processes run.

Table 21-2 Default scheduling classes and priorities

Process	Engine	Process created by engine	Agent	Script
Default scheduling class	RT	TS	TS	TS
Default priority (Linux)	Min: 0 Max: 99	0	0	0

When priority is set to "" (empty string), VCS converts the priority to a default value. For RT, the default priority equals two less than the strongest priority supported by the RealTime class. So, if the strongest priority supported by the RealTime class is 59, the default priority for the RT class is 57.

Note: For standard configurations, Veritas recommends using the default values for scheduling unless specific configuration requirements dictate otherwise.

Note that the default priority value is platform-specific. When priority is set to "" (empty string), VCS converts the priority to a value specific to the platform on which the system is running. For TS, the default priority equals the strongest priority supported by the TimeSharing class. For RT, the default priority equals two less than the strongest priority supported by the RealTime class. So, if the strongest priority supported by the RealTime class is 59, the default priority for the RT class is 57.

VCS agent statistics

You can configure VCS to track the time taken for monitoring resources.

You can use these statistics to configure the MonitorTimeout attribute.

You can also detect potential problems with resources and systems on which resources are online by analyzing the trends in the time taken by the resource's monitor cycle. Note that VCS keeps track of monitor cycle times for online resources only.

VCS calculates the time taken for a monitor cycle to complete and computes an average of monitor times after a specific number of monitor cycles and stores the average in a resource-level attribute.

VCS also tracks increasing trends in the monitor cycle times and sends notifications about sudden and gradual increases in monitor times.

VCS uses the following parameters to compute the average monitor time and to detect increasing trends in monitor cycle times:

- Frequency: The number of monitor cycles after which the monitor time average is computed and sent to the VCS engine.
  For example, if Frequency is set to 10, VCS computes the average monitor time after every 10 monitor cycles.
- ExpectedValue: The expected monitor time (in milliseconds) for a resource.
  VCS sends a notification if the actual monitor time exceeds the expected monitor time by the ValueThreshold. So, if you set this attribute to 5000 for a FileOnOff resource, and if ValueThreshold is set to 40%, VCS will send a notification only when the monitor cycle for the FileOnOff resource exceeds the expected time by over 40%, that is 7000 milliseconds.
- ValueThreshold: The maximum permissible deviation (in percent) from the expected monitor time. When the time for a monitor cycle exceeds this limit, VCS sends a notification about the sudden increase or decrease in monitor time.
  For example, a value of 100 means that VCS sends a notification if the actual monitor time deviates from the expected time by over 100%.
  VCS sends these notifications conservatively. If 12 consecutive monitor cycles exceed the threshold limit, VCS sends a notification for the first spike, and then a collective notification for the next 10 consecutive spikes.
- AvgThreshold: The threshold value (in percent) for increase in the average monitor cycle time for a resource.
  VCS maintains a running average of the time taken by the monitor cycles of a resource. The first such computed running average is used as a benchmark average. If the current running average for a resource differs from the benchmark average by more than this threshold value, VCS regards this as a sign of gradual increase or decrease in monitor cycle times and sends a notification about it for the resource. Whenever such an event occurs, VCS resets the internally maintained benchmark average to this new average. VCS sends notifications regardless of whether the deviation is an increase or decrease in the monitor cycle time.
  For example, a value of 25 means that if the actual average monitor time is 25% more than the benchmark monitor time average, VCS sends a notification.

Tracking monitor cycle times

VCS marks sudden changes in monitor times by comparing the time taken for each monitor cycle with the ExpectedValue. If this difference exceeds the ValueThreshold, VCS sends a notification about the sudden change in monitor time. Note that VCS sends this notification only if monitor time increases.

VCS marks gradual changes in monitor times by comparing the benchmark average and the moving average of monitor cycle times. VCS computes the benchmark average after a certain number of monitor cycles and computes the moving average after every monitor cycle. If the current moving average exceeds the benchmark average by more than the AvgThreshold, VCS sends a notification about this gradual change in the monitor cycle time.

VCS attributes enabling agent statistics

This topic describes the attributes that enable VCS agent statistics.

MonitorStatsParam A resource type-level attribute, which stores the required parameter values for calculating monitor time statistics.

```
static str MonitorStatsParam = { Frequency = 10,
ExpectedValue = 3000, ValueThreshold = 100,
AvgThreshold = 40 }
```

- Frequency: Defines the number of monitor cycles after which the average monitor cycle time should be computed and sent to the engine. If configured, the value for this attribute must be between 1 and 30. It is set to 0 by default.
- ExpectedValue: The expected monitor time in milliseconds for all resources of this type. Default=3000.
- ValueThreshold: The acceptable percentage difference between the expected monitor cycle time (ExpectedValue) and the actual monitor cycle time. Default=100.
- AvgThreshold: The acceptable percentage difference between the benchmark average and the moving average of monitor cycle times. Default=40

MonitorTimeStats Stores the average time taken by a number of monitor cycles specified by the Frequency attribute along with a timestamp value of when the average was computed.

```
str MonitorTimeStats{} = { Avg = "0", TS = "" }
```

This attribute is updated periodically after a number of monitor cycles specified by the Frequency attribute. If Frequency is set to 10, the attribute stores the average of 10 monitor cycle times and is updated after every 10 monitor cycles.

The default value for this attribute is 0.

ComputeStats

A flag that specifies whether VCS keeps track of the monitor times for the resource.

```
boolean ComputeStats = 0
```

The value 0 indicates that VCS will not keep track of the time taken by the monitor routine for the resource. The value 1 indicates that VCS keeps track of the monitor time for the resource.

The default value for this attribute is 0.

About VCS tunable parameters

VCS has some tunable parameters that you can configure to enhance the performance of specific features. However, Veritas Technologies recommends that the user not change the tunable kernel parameters without assistance from Veritas Technologies support personnel. Several of the tunable parameters preallocate memory for critical data structures, and a change in their values could increase memory use or degrade performance.

Warning: Do not adjust the VCS tunable parameters for kernel modules such as VXFEN without assistance from Veritas Technologies support personnel.

About LLT tunable parameters

LLT provides various configuration and tunable parameters to modify and control the behavior of the LLT module. This section describes some of the LLT tunable parameters that can be changed at run-time and at LLT start-time.

See "About Low Latency Transport (LLT)" on page 238.

The tunable parameters are classified into two categories:

- LLT timer tunable parameters
  See "About LLT timer tunable parameters" on page 590.
- LLT flow control tunable parameters
  See "About LLT flow control tunable parameters" on page 595.

See "Setting LLT tunable parameters" on page 599.

About LLT timer tunable parameters

Table 21-3 lists the LLT timer tunable parameters. The timer values are set in .01 sec units. The command `lltconfig -T query` can be used to display current timer values.

Table 21-3 LLT timer tunable parameters

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
peerinact	LLT marks a link of a peer node as “inactive," if it does not receive any packet on that link for this timer interval. Once a link is marked as "inactive," LLT will not send any data on that link.	1600	■ Change this value for delaying or speeding up node/link inactive notification mechanism as per client’s notification processing logic. ■ Increase the value for planned replacement of faulty network cable /switch. ■ In some circumstances, when the private networks links are very slow or the network traffic becomes very bursty, increase this value so as to avoid false notifications of peer death. Set the value to a high value for planned replacement of faulty network cable or faulty switch.	The timer value should always be higher than the peertrouble timer value.
rpeerinact	Mark RDMA channel of a RDMA link as "inactive", if the node does not receive any packet on that link for this timer interval. Once RDMA channel is marked as "inactive", LLT does not send any data on the RDMA channel of that link, however, it may continue to send data over non-RDMA channel of that link until peerinact expires. You can view the status of the RDMA channel of a RDMA link using lltstat -nvv -r command. This parameter is supported only on selected versions of Linux.	700	Decrease the value of this tunable for speeding up the RDMA link failure recovery. If the links are unstable, and they are going up and down frequently then do not decrease this value.	This timer value should always be greater than peertrouble timer value and less than peerinact value.

Table 21-3 LLT timer tunable parameters *(continued)*

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
peertrouble	LLT marks a high-pri link of a peer node as "troubled", if it does not receive any packet on that link for this timer interval. Once a link is marked as "troubled", LLT will not send any data on that link till the link is up.	200	■ In some circumstances, when the private networks links are very slow or nodes in the cluster are very busy, increase the value. ■ Increase the value for planned replacement of faulty network cable /faulty switch.	This timer value should always be lower than peerinact timer value. Also, It should be close to its default value.
peertroublelo	LLT marks a low-pri link of a peer node as "troubled", if it does not receive any packet on that link for this timer interval. Once a link is marked as "troubled", LLT will not send any data on that link till the link is available.	400	■ In some circumstances, when the private networks links are very slow or nodes in the cluster are very busy, increase the value. ■ Increase the value for planned replacement of faulty network cable /faulty switch.	This timer value should always be lower than peerinact timer value. Also, It should be close to its default value.
heartbeat	LLT sends heartbeat packets repeatedly to peer nodes after every heartbeat timer interval on each highpri link.	50	In some circumstances, when the private networks links are very slow (or congested) or nodes in the cluster are very busy, increase the value.	This timer value should be lower than peertrouble timer value. Also, it should not be close to peertrouble timer value.
heartbeatlo	LLT sends heartbeat packets repeatedly to peer nodes after every heartbeatlo timer interval on each low pri link.	100	In some circumstances, when the networks links are very slow or nodes in the cluster are very busy, increase the value.	This timer value should be lower than peertroublelo timer value. Also, it should not be close to peertroublelo timer value.

Table 21-3 LLT timer tunable parameters *(continued)*

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
timetoreqhb	If LLT does not receive any packet from the peer node on a particular link for "timetoreqhb" time period, it attempts to request heartbeats (sends 5 special heartbeat requests (hbreqs) to the peer node on the same link) from the peer node. If the peer node does not respond to the special heartbeat requests, LLT marks the link as “expired” for that peer node. The value can be set from the range of 0 to (peerinact -200). The value 0 disables the request heartbeat mechanism.	1400	Decrease the value of this tunable for speeding up node/link inactive notification mechanism as per client’s notification processing logic. Disable the request heartbeat mechanism by setting the value of this timer to 0 for planned replacement of faulty network cable /switch. In some circumstances, when the private networks links are very slow or the network traffic becomes very bursty, don’t change the value of this timer tunable.	This timer is set to ‘peerinact - 200’ automatically every time when the peerinact timer is changed.
reqhbtime	This value specifies the time interval between two successive special heartbeat requests. See the timetoreqhb parameter for more information on special heartbeat requests.	40	Veritas recommends that you do not change this value.	Not applicable
timetosendhb	LLT sends out of timer context heartbeats to keep the node alive when LLT timer does not run at regular interval. This option specifies the amount of time to wait before sending a heartbeat in case of timer not running. If this timer tunable is set to 0, the out of timer context heartbeating mechanism is disabled.	200	Disable the out of timer context heart-beating mechanism by setting the value of this timer to 0 for planned replacement of faulty network cable /switch. In some circumstances, when the private networks links are very slow or nodes in the cluster are very busy, increase the value	This timer value should not be more than peerinact timer value. Also, it should not be close to the peerinact timer value.

Table 21-3 LLT timer tunable parameters *(continued)*

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
sendhbcap	This value specifies the maximum time for which LLT will send contiguous out of timer context heartbeats.	18000	Veritas recommends that you do not change this value.	NA
oos	If the out-of-sequence timer has expired for a node, LLT sends an appropriate NAK to that node. LLT does not send a NAK as soon as it receives an oos packet. It waits for the oos timer value before sending the NAK.	10	Do not change this value for performance reasons. Lowering the value can result in unnecessary retransmissions/negative acknowledgement traffic. You can increase the value of oos if the round trip time is large in the cluster (for example, campus cluster).	Not applicable
retrans	LLT retransmits a packet if it does not receive its acknowledgement for this timer interval value.	10	Do not change this value. Lowering the value can result in unnecessary retransmissions. You can increase the value of retrans if the round trip time is large in the cluster (for example, campus cluster).	Not applicable
service	LLT calls its service routine (which delivers messages to LLT clients) after every service timer interval.	100	Do not change this value for performance reasons.	Not applicable
arp	LLT flushes stored address of peer nodes when this timer expires and relearns the addresses.	0	This feature is disabled by default.	Not applicable
arpreq	LLT sends an arp request when this timer expires to detect other peer nodes in the cluster.	3000	Do not change this value for performance reasons.	Not applicable

About LLT flow control tunable parameters

Table 21-4 lists the LLT flow control tunable parameters. The flow control values are set in number of packets. The command `lltconfig -F query` can be used to display current flow control settings.

Table 21-4 LLT flow control tunable parameters

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
highwater	When the number of packets in transmit queue for a node reaches highwater, LLT is flow controlled.	200	If a client generates data in bursty manner, increase this value to match the incoming data rate. Note that increasing the value means more memory consumption so set an appropriate value to avoid wasting memory unnecessarily. Lowering the value can result in unnecessary flow controlling the client.	This flow control value should always be higher than the lowwater flow control value.
lowwater	When LLT has flow controlled the client, it will not start accepting packets again till the number of packets in the port transmit queue for a node drops to lowwater.	100	Veritas does not recommend to change this tunable.	This flow control value should be lower than the highwater flow control value. The value should not be close the highwater flow control value.
rporthighwater	When the number of packets in the receive queue for a port reaches highwater, LLT is flow controlled.	200	If a client generates data in bursty manner, increase this value to match the incoming data rate. Note that increasing the value means more memory consumption so set an appropriate value to avoid wasting memory unnecessarily. Lowering the value can result in unnecessary flow controlling the client on peer node.	This flow control value should always be higher than the rportlowwater flow control value.

Table 21-4 LLT flow control tunable parameters *(continued)*

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
rportlowwater	When LLT has flow controlled the client on peer node, it will not start accepting packets for that client again till the number of packets in the port receive queue for the port drops to rportlowwater.	100	Veritas does not recommend to change this tunable.	This flow control value should be lower than the rpothighwater flow control value. The value should not be close the rporthighwater flow control value.

Table 21-4 LLT flow control tunable parameters *(continued)*

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
window	This is the maximum number of un-ACKed packets LLT will put in flight.	50	For performance reason, the adaptive window feature is enabled, by default for port 5 (cfs) and port 24(cvm). You can manually enable adaptive window for other ports by changing the value of the LLT_AW_PORT_LIST parameter in the /etc/sysconfig/llt file. ■ To disable the adaptive window, change the value of the LLT_ENABLE_AWINDOW parameter to zero. ■ To enable adaptive window for ports other than 5 and 24, add the port numbers in LLT_AW_PORT_LIST separated by comma. For example: LLT_AW_PORT_LIST=: '"5,24,0,1,5,14"'. ■ To set window size for ports other than those listed in the LLT_AW_PORT_LIST parameter, change the value of the window parameter as per the private network speed. Example: set-flow window: 2000 Change the value as per the private networks speed. Lowering the value irrespective of network speed may result in unnecessary retransmission of out of window sequence packets.	This flow control value should not be higher than the difference between the highwater flow control value and the lowwater flow control value. The value of this parameter (window) should be aligned with the value of the bandwidth delay product.

Table 21-4 LLT flow control tunable parameters *(continued)*

LLT parameter	Description	Default	When to change	Dependency with other LLT tunable parameters
linkburst	It represents the number of back-to-back packets that LLT sends on a link before the next link is chosen.	32	For performance reasons, its value should be either 0 or at least 32.	This flow control value should not be higher than the difference between the highwater flow control value and the lowwater flow control value.
ackval	LLT sends acknowledgement of a packet by piggybacking an ACK packet on the next outbound data packet to the sender node. If there are no data packets on which to piggyback the ACK packet, LLT waits for ackval number of packets before sending an explicit ACK to the sender.	10	Do not change this value for performance reasons. Increasing the value can result in unnecessary retransmissions.	Not applicable
sws	To avoid Silly Window Syndrome, LLT transmits more packets only when the count of un-acked packet goes to below of this tunable value.	40	For performance reason, its value should be changed whenever the value of the window tunable is changed as per the formula given below: sws = window *4/5.	Its value should be lower than that of window. Its value should be close to the value of window tunable.
largepktlen	When LLT has packets to delivers to multiple ports, LLT delivers one large packet or up to five small packets to a port at a time. This parameter specifies the size of the large packet.	1024	Veritas does not recommend to change this tunable.	Not applicable

About miscellaneous LLT tunable parameters

LLT parameter	Description	Default value	When to change	Dependency on other parameters
hbthread	Indicates whether heartbeating should be done only using the OS timer context or also using per-CPU threads. See "Configuring heartbeat threads to improve cluster resiliency" on page 107.	1	Set this value to 0 when heartbeating should be done only using the OS timer context.	Not applicable

Setting LLT tunable parameters

You can set the LLT tunable parameters either with the `lltconfig` command or in the `/etc/llttab` file. You can use the `lltconfig` command to change a parameter on the local node at run time. Veritas recommends you run the command on all the nodes in the cluster to change the values of the parameters. To set an LLT parameter across system reboots, you must include the parameter definition in the `/etc/llttab` file. Default values of the parameters are taken if nothing is specified in /etc/lltab. The parameters values specified in the /etc/lltab file come into effect at LLT start-time only. Veritas recommends that you specify the same definition of the tunable parameters in the `/etc/llttab` file of each node.

To get and set a timer tunable:

- To get the current list of timer tunable parameters using lltconfig command:

```
# lltconfig -T query
```

- To set a timer tunable parameter using the `lltconfig` command:

```
# lltconfig -T timer tunable:value
```

- To set a timer tunable parameter in the `/etc/llttab` file:

```
set-timer  timer tunable:value
```

To get and set a flow control tunable:

- To get the current list of flow control tunable parameters using lltconfig command:

```
# lltconfig -F query
```

- To set a flow control tunable parameter using the `lltconfig` command:

  ```
  # lltconfig -F flowcontrol tunable:value
  ```

- To set a flow control tunable parameter in the `/etc/llttab` file:

  ```
  set-flow  flowcontrol tunable:value
  ```

See the `lltconfig`(1M) and `llttab`(1M) manual pages.

About GAB tunable parameters

GAB provides various configuration and tunable parameters to modify and control the behavior of the GAB module.

See "About Group Membership Services/Atomic Broadcast (GAB)" on page 237.

These tunable parameters not only provide control of the configurations like maximum possible number of nodes in the cluster, but also provide control on how GAB behaves when it encounters a fault or a failure condition. Some of these tunable parameters are needed when the GAB module is loaded into the system. Any changes to these load-time tunable parameters require either unload followed by reload of GAB module or system reboot. Other tunable parameters (run-time) can be changed while GAB module is loaded, configured, and cluster is running. Any changes to such a tunable parameter will have immediate effect on the tunable parameter values and GAB behavior.

These tunable parameters are defined in the following file:

`/etc/sysconfig/gab`

See "About GAB load-time or static tunable parameters" on page 600.

See "About GAB run-time or dynamic tunable parameters" on page 602.

About GAB load-time or static tunable parameters

Table 21-5 lists the static tunable parameters in GAB that are used during module load time. Use the `gabconfig -e` command to list all such GAB tunable parameters.

You can modify these tunable parameters only by adding new values in the GAB configuration file. The changes take effect only on reboot or on reload of the GAB module.

Table 21-5 GAB static tunable parameters

GAB parameter	Description	Values (default and range)
numnids	Maximum number of nodes in the cluster	Default: 128 Range: 1-128
numports	Maximum number of ports in the cluster	Default: 32 Range: 1-32
flowctrl	Number of pending messages in GAB queues (send or receive) before GAB hits flow control. This can be overwritten while cluster is up and running with the `gabconfig -Q` option. Use the `gabconfig` command to control value of this tunable.	Default: 128 Range: 1-1024
logbufsize	GAB internal log buffer size in bytes	Default: 48100 Range: 8100-65400
msglogsize	Maximum messages in internal message log	Default: 256 Range: 128-4096
isolate_time	Maximum time to wait for isolated client Can be overridden at runtime See "About GAB run-time or dynamic tunable parameters" on page 602.	Default: 120000 msec (2 minutes) Range: 160000-240000 (in msec)
kill_ntries	Number of times to attempt to kill client Can be overridden at runtime See "About GAB run-time or dynamic tunable parameters" on page 602.	Default: 5 Range: 3-10
conn_wait	Maximum number of wait periods (as defined in the stable timeout parameter) before GAB disconnects the node from the cluster during cluster reconfiguration	Default: 12 Range: 1-256

Table 21-5 GAB static tunable parameters *(continued)*

GAB parameter	Description	Values (default and range)
ibuf_count	Determines whether the GAB logging daemon is enabled or disabled The GAB logging daemon is enabled by default. To disable, change the value of gab_ibuf_count to 0. The disable login to the gab daemon while cluster is up and running with the `gabconfig -K` option. Use the `gabconfig` command to control value of this tunable.	Default: 8 Range: 0-32
kstat_size	Number of system statistics to maintain in GAB	Default: 60 Range: 0 - 240

About GAB run-time or dynamic tunable parameters

You can change the GAB dynamic tunable parameters while GAB is configured and while the cluster is running. The changes take effect immediately on running the `gabconfig` command. Note that some of these parameters also control how GAB behaves when it encounters a fault or a failure condition. Some of these conditions can trigger a PANIC which is aimed at preventing data corruption.

You can display the default values using the `gabconfig -l` command. To make changes to these values persistent across reboots, you can append the appropriate command options to the `/etc/gabtab` file along with any existing options. For example, you can add the `-k` option to an existing `/etc/gabtab` file that might read as follows:

```
gabconfig -c -n4
```

After adding the option, the `/etc/gabtab` file looks similar to the following:

```
gabconfig -c -n4 -k
```

Table 21-6 describes the GAB dynamic tunable parameters as seen with the `gabconfig -l` command, and specifies the command to modify them.

Table 21-6 GAB dynamic tunable parameters

GAB parameter	Description and command

Control port seed	This option defines the minimum number of nodes that can form the cluster. This option controls the forming of the cluster. If the number of nodes in the cluster is less than the number specified in the `gabtab` file, then the cluster will not form. For example: if you type `gabconfig -c -n4`, then the cluster will not form until all four nodes join the cluster. If this option is enabled using the `gabconfig -x` command then the node will join the cluster even if the other nodes in the cluster are not yet part of the membership.
	
	Use the following command to set the number of nodes that can form the cluster:
	
	`gabconfig -n count`
	
	Use the following command to enable control port seed. Node can form the cluster without waiting for other nodes for membership:
	
	`gabconfig -x`

Halt on process death	Default: Disabled
	
	This option controls GAB's ability to halt (panic) the system on user process death. If _had and _hashadow are killed using `kill -9`, the system can potentially lose high availability. If you enable this option, then the GAB will PANIC the system on detecting the death of the client process. The default behavior is to disable this option.
	
	Use the following command to enable halt system on process death:
	
	`gabconfig -p`
	
	Use the following command to disable halt system on process death:
	
	`gabconfig -P`

Table 21-6 GAB dynamic tunable parameters *(continued)*

GAB parameter	Description and command

Missed heartbeat halt	Default: Disabled
	
	If this option is enabled then the system will panic on missing the first heartbeat from the VCS engine or the `vxconfigd` daemon in a CVM environment. The default option is to disable the immediate panic.
	
	This GAB option controls whether GAB can panic the node or not when the VCS engine or the vxconfigd daemon miss to heartbeat with GAB. If the VCS engine experiences a hang and is unable to heartbeat with GAB, then GAB will NOT PANIC the system immediately. GAB will first try to abort the process by sending SIGABRT (kill_ntries - default value 5 times) times after an interval of "iofence_timeout" (default value 15 seconds). If this fails, then GAB will wait for the "isolate timeout" period which is controlled by a global tunable called isolate_time (default value 2 minutes). If the process is still alive, then GAB will PANIC the system.
	
	If this option is enabled GAB will immediately HALT the system in case of missed heartbeat from client.
	
	Use the following command to enable system halt when process heartbeat fails:
	
	`gabconfig -b`
	
	Use the following command to disable system halt when process heartbeat fails:
	
	`gabconfig -B`

Table 21-6 GAB dynamic tunable parameters *(continued)*

GAB parameter	Description and command

Halt on rejoin	Default: Disabled
	
	This option allows the user to configure the behavior of the VCS engine or any other user process when one or more nodes rejoin a cluster after a network partition. By default GAB will not PANIC the node running the VCS engine. GAB kills the userland process (the VCS engine or the vxconfigd process). This recycles the user port (port h in case of the VCS engine) and clears up messages with the old generation number programmatically. Restart of the process, if required, must be handled outside of GAB control, e.g., for hashadow process restarts _had.
	
	When GAB has kernel clients (such as fencing, VxVM, or VxFS), then the node will always PANIC when it rejoins the cluster after a network partition. The PANIC is mandatory since this is the only way GAB can clear ports and remove old messages.
	
	Use the following command to enable system halt on rejoin:
	
	`gabconfig -j`
	
	Use the following command to disable system halt on rejoin:
	
	`gabconfig -J`

Keep on killing	Default: Disabled If this option is enabled, then GAB prevents the system from PANICKING when the VCS engine or the vxconfigd process fail to heartbeat with GAB and GAB fails to kill the VCS engine or the vxconfigd process. GAB will try to continuously kill the VCS engine and will not panic if the kill fails. Repeat attempts to kill process if it does not die `gabconfig -k`

Table 21-6 GAB dynamic tunable parameters *(continued)*

GAB parameter	Description and command

Quorum flag	Default: Disabled
	This is an option in GAB which allows a node to IOFENCE (resulting in a PANIC) if the new membership set is < 50% of the old membership set. This option is typically disabled and is used when integrating with other products
	Enable iofence quorum
	`gabconfig -q`
	Disable iofence quorum
	`gabconfig -d`

GAB queue limit	Default: Send queue limit: 128
	Default: Recv queue limit: 128
	GAB queue limit option controls the number of pending message before which GAB sets flow. Send queue limit controls the number of pending message in GAB send queue. Once GAB reaches this limit it will set flow control for the sender process of the GAB client. GAB receive queue limit controls the number of pending message in GAB receive queue before GAB send flow control for the receive side.
	Set the send queue limit to specified value
	`gabconfig -Q sendq:value`
	Set the receive queue limit to specified value
	`gabconfig -Q recvq:value`

IOFENCE timeout	Default: 15000(ms) This parameter specifies the timeout (in milliseconds) for which GAB will wait for the clients to respond to an IOFENCE message before taking next action. Based on the value of kill_ntries , GAB will attempt to kill client process by sending SIGABRT signal. If the client process is still registered after GAB attempted to kill client process for the value of kill_ntries times, GAB will halt the system after waiting for additional isolate_timeout value. Set the iofence timeout value to specified value in milliseconds. `gabconfig -f value`

Table 21-6 GAB dynamic tunable parameters *(continued)*

GAB parameter	Description and command
Stable timeout	Default: 5000(ms) Specifies the time GAB waits to reconfigure membership after the last report from LLT of a change in the state of local node connections for a given port. Any change in the state of connections will restart GAB waiting period. Set the stable timeout to specified value `gabconfig -t stable`
Isolate timeout	Default: 120000(ms) This tunable specifies the timeout value for which GAB will wait for client process to unregister in response to GAB sending SIGKILL signal. If the process still exists after isolate timeout GAB will halt the system `gabconfig -S isolate_time:value`
Kill_ntries	Default: 5 This tunable specifies the number of attempts GAB will make to kill the process by sending SIGABRT signal. `gabconfig -S kill_ntries:value`
Driver state	This parameter shows whether GAB is configured. GAB may not have seeded and formed any membership yet.
Partition arbitration	This parameter shows whether GAB is asked to specifically ignore jeopardy. See the `gabconfig` (1M) manual page for details on the `-s` flag.

About VXFEN tunable parameters

The section describes the VXFEN tunable parameters and how to reconfigure the VXFEN module.

Table 21-7 describes the tunable parameters for the VXFEN driver.

Table 21-7 VXFEN tunable parameters

vxfen Parameter	Description and Values: Default, Minimum, and Maximum
dbg_log_size	Size of debug log in bytes ■ Values Default: 524288(512KB) Minimum: 65536(64KB) Maximum: 1048576(1MB)
vxfen_max_delay	Specifies the maximum number of seconds that the smaller sub-cluster waits before racing with larger sub-clusters for control of the coordinator disks when a network partition occurs. This value must be greater than the vxfen_min_delay value. ■ Values Default: 60 Minimum: 1 Maximum: 600
vxfen_min_delay	Specifies the minimum number of seconds that the smaller sub-cluster waits before racing with larger sub-clusters for control of the coordinator disks when a network partition occurs. This value must be smaller than or equal to the vxfen_max_delay value. ■ Values Default: 1 Minimum: 1 Maximum: 600
vxfen_vxfnd_tmt	Specifies the time in seconds that the I/O fencing driver VxFEN waits for the I/O fencing daemon VXFEND to return after completing a given task. ■ Values Default: 60 Minimum: 10 Maximum: 600

Table 21-7 VXFEN tunable parameters *(continued)*

vxfen Parameter	Description and Values: Default, Minimum, and Maximum
panic_timeout_offst	Specifies the time in seconds based on which the I/O fencing driver VxFEN computes the delay to pass to the GAB module to wait until fencing completes its arbitration before GAB implements its decision in the event of a split-brain. You can set this parameter in the `vxfenmode` file and use the `vxfenadm` command to check the value. Depending on the vxfen_mode, the GAB delay is calculated as follows: ■ For scsi3 mode: 1000 * (panic_timeout_offst + vxfen_max_delay) ■ For customized mode: 1000 * (panic_timeout_offst + max (vxfen_vxfnd_tmt, vxfen_loser_exit_delay)) ■ Default: 10

In the event of a network partition, the smaller sub-cluster delays before racing for the coordinator disks. The time delay allows a larger sub-cluster to win the race for the coordinator disks. The vxfen_max_delay and vxfen_min_delay parameters define the delay in seconds.

Configuring the VXFEN module parameters

After adjusting the tunable kernel driver parameters, you must reconfigure the VXFEN module for the parameter changes to take effect.

The following example procedure changes the value of the vxfen_min_delay parameter.

On each Linux node, edit the file /etc/sysconfig/vxfen to change the value of the vxfen driver tunable global parameters, vxfen_max_delay and vxfen_min_delay.

Note: You must restart the VXFEN module to put any parameter change into effect.

To configure the VxFEN parameters and reconfigure the VxFEN module

1. Stop all the applications that are not configured under VCS. Use native application commands to stop the application.
2. Stop VCS on all the nodes. Run the following command on each node:

```
# hastop -local
```

3 Stop the VxFEN driver.

```
# systemctl stop vxfen
```

4 Edit the /etc/sysconfig/vxfen file.

For example, change the entry from:

```
vxfen_min_delay=1
```

to:

```
vxfen_min_delay=30
```

5 Start the VXFEN module.

```
# systemctl start vxfen
```

6 Start all the applications that are not configured under VCS. Use native application commands to start the applications.

7 Start VCS.

```
#  hastart
```

About AMF tunable parameters

You can set the Asynchronous Monitoring Framework (AMF) kernel module tunable using the following command:

```
# amfconfig -T tunable_name=tunable_value,
tunable_name=tunable_value...
```

Table 21-8 lists the possible tunable parameters for the AMF kernel:

Table 21-8 AMF tunable parameters

AMF parameter	Description	Value
dbglogsz	AMF maintains an in-memory debug log. This parameter (specified in units of KBs) controls the amount of kernel memory allocated for this log.	Min - 4 Max - 512 Default - 256
processhashsz	AMF stores registered events in an event type specific hash table. This parameter controls the number of buckets allocated for the hash table used to store process-related events.	Min - 64 Max - 8192 Default - 2048

Table 21-8 AMF tunable parameters *(continued)*

AMF parameter	Description	Value
mnthashsz	AMF stores registered events in an event type specific hash table. This parameter controls the number of buckets allocated for the hash table used to store mount-related events.	Min - 64 Max - 8192 Default - 512
conthashsz	AMF stores registered events in an event type specific hash table. This parameter controls the number of buckets allocated for the hash table used to store container-related events.	Min - 1 Max - 64 Default - 32
filehashsz	AMF stores registered events in an event type specific hash table. This parameter controls the number of buckets allocated for the hash table used to store file-related events.	Min - 1 Max - 64 Default - 32
dirhashsz	AMF stores registered events in an event type specific hash table. This parameter controls the number of buckets allocated for the hash table used to store directory-related events.	Min - 1 Max - 64 Default - 32

The parameter values that you update are reflected after you reconfigure AMF driver. Note that if you unload the module, the updated values are lost. You must unconfigure the module using the `amfconfig -U` or equivalent command and then reconfigure using the `amfconfig -c` command for the updated tunables to be effective. If you want to set the tunables at module load time, you can write these `amfconfig` commands in the `amftab` file.

See the `amftab`(4) manual page for details.

Chapter 22

Troubleshooting and recovery for VCS

This chapter includes the following topics:

- VCS message logging
- Troubleshooting the VCS engine
- Troubleshooting Low Latency Transport (LLT)
- Troubleshooting Group Membership Services/Atomic Broadcast (GAB)
- Troubleshooting VCS startup
- Troubleshooting issues with systemd unit service files
- Troubleshooting Intelligent Monitoring Framework (IMF)
- Troubleshooting service groups
- Troubleshooting resources
- Troubleshooting sites
- Troubleshooting I/O fencing
- Troubleshooting notification
- Troubleshooting and recovery for global clusters
- Troubleshooting the steward process
- Troubleshooting licensing
- Troubleshooting secure configurations

- Troubleshooting wizard-based configuration issues
- Troubleshooting issues with the Veritas High Availability view

VCS message logging

VCS generates two types of logs: the engine log and the agent log. Log file names are appended by letters; letter A indicates the first log file, B the second, and C the third. After three files, the least recent file is removed and another file is created. For example, if engine_A.log is filled to its capacity, it is renamed as engine_B.log and the engine_B.log is renamed as engine_C.log. In this example, the least recent file is engine_C.log and therefore it is removed. You can update the size of the log file using the LogSize cluster level attribute.

See "Cluster attributes" on page 760.

The engine log is located at /var/VRTSvcs/log/engine_A.log. The format of engine log messages is:

Timestamp (Year/MM/DD) | Mnemonic | Severity | UMI | Message Text

- *Timestamp*: the date and time the message was generated.
- *Mnemonic*: the string ID that represents the product (for example, VCS).
- *Severity*: levels include CRITICAL, ERROR, WARNING, NOTICE, and INFO (most to least severe, respectively).
- *UMI*: a unique message ID.
- *Message Text*: the actual message generated by VCS.

A typical engine log resembles:

```
2011/07/10 16:08:09 VCS INFO V-16-1-10077 Received new
cluster membership
```

The agent log is located at `/var/VRTSvcs/log/<agent>.log`. The format of agent log messages resembles:

Timestamp (Year/MM/DD) | Mnemonic | Severity | UMI | Agent Type | Resource Name | Entry Point | Message Text

A typical agent log resembles:

```
2011/07/10 10:38:23 VCS WARNING V-16-2-23331
Oracle:VRT:monitor:Open for ora_lgwr failed, setting
cookie to null.
```

Note that the logs on all nodes may not be identical because

- VCS logs local events on the local nodes.
- All nodes may not be running when an event occurs.

VCS prints the warning and error messages to STDERR.

If the VCS engine, Command Server, or any of the VCS agents encounter some problem, then First Failure Data Capture (FFDC) logs are generated and dumped along with other core dumps and stack traces to the following location:

- For VCS engine: `$VCS_DIAG/diag/had`
- For Command Server: `$VCS_DIAG/diag/CmdServer`
- For VCS agents: `$VCS_DIAG/diag/agents/`*`type`*, where *type* represents the specific agent type.

The default value for variable $VCS_DIAG is `/var/VRTSvcs/`.

If the debug logging is not turned on, these FFDC logs are useful to analyze the issues that require professional support.

Log unification of VCS agent's entry points

Earlier, the logs of VCS agents implemented using script and C/C++ language were scattered between the engine log file and agent log file respectively.

The logging is improved so that logs of all the entry points will be logged in respective agent log file. For example, the logs of Mount agent can be found in the Mount agent log file located under /var/VRTSvcs/log directory.

Moreover, using LogViaHalog attribute, user can switch back to the older logging behavior. This attribute supports two values 0 and 1. By default the value is 0, which means the agent's log will go into their respective agent log file. If value is set to 1, then the C/C++ entry point's logs will go into the agent log file and the script entry point's logs will go into the engine log file using halog command.

Note: Irrespective of the value of LogViaHalog, the script entry point's logs that are executed in the container will go into the engine log file.

Enhancing First Failure Data Capture (FFDC) to troubleshoot VCS resource's unexpected behavior

FFDC is the process of generating and dumping information on unexpected events.

Earlier, FFDC information is generated on following unexpected events:

- Segmentation fault

- When agent fails to heartbeat with engine

From VCS 6.2 version, the capturing of the FFDC information on unexpected events has been extended to resource level and to cover VCS events. This means, if a resource faces an unexpected behavior then FFDC information will be generated.

The current version enables the agent to log detailed debug logging functions during unexpected events with respect to resource, such as,

- Monitor entry point of a resource reported OFFLINE/INTENTIONAL OFFLINE when it was in ONLINE state.
- Monitor entry point of a resource reported UNKNOWN.
- If any entry point times-out.
- If any entry point reports failure.
- When a resource is detected as ONLINE or OFFLINE for the first time.

Now whenever an unexpected event occurs FFDC information will be automatically generated. And this information will be logged in their respective agent log file.

GAB message logging

If GAB encounters some problem, then First Failure Data Capture (FFDC) logs are also generated and dumped.

When you have configured GAB, GAB also starts a GAB logging daemon (`/opt/VRTSgab/gablogd`). GAB logging daemon is enabled by default. You can change the value of the GAB tunable parameter `gab_ibuf_count` to disable the GAB logging daemon.

See "About GAB load-time or static tunable parameters" on page 600.

This GAB logging daemon collects the GAB related logs when a critical events such as an iofence or failure of the master of any GAB port occur, and stores the data in a compact binary form. You can use the `gabread_ffdc` utility as follows to read the GAB binary log files:

```
/opt/VRTSgab/gabread_ffdc-kernel_version  binary_logs_files_location
```

You can change the values of the following environment variables that control the GAB binary log files:

- GAB_FFDC_MAX_INDX: Defines the maximum number of GAB binary log files The GAB logging daemon collects the defined number of log files each of eight MB size. The default value is 20, and the files are named `gablog.1` through `gablog.20`. At any point in time, the most recent file is the `gablog.1` file.
- GAB_FFDC_LOGDIR: Defines the log directory location for GAB binary log files

The default location is:

`/var/log/gab_ffdc`

Note that the gablog daemon writes its log to the `glgd_A.log` and `glgd_B.log` files in the same directory.

You can either define these variables in the following GAB startup file or use the `export` command. You must restart GAB for the changes to take effect.

`/etc/sysconfig/gab`

Enabling debug logs for agents

This section describes how to enable debug logs for VCS agents.

To enable debug logs for agents

1. Set the configuration to read-write:

```
# haconf -makerw
```

2. Enable logging and set the desired log levels. Use the following command syntax:

```
# hatype -modify $typename LogDbg DBG_1 DBG_2 DBG_4 DBG_21
```

The following example shows the command line for the IPMultiNIC resource type.

```
# hatype -modify IPMultiNIC LogDbg DBG_1 DBG_2 DBG_4 DBG_21
```

See the description of the `LogDbg` attribute for more information.

See “Resource type attributes” on page 703.

3. For script-based agents, run the `halog` command to add the messages to the engine log:

```
# halog -addtags DBG_1 DBG_2 DBG_4 DBG_21
```

4. Save the configuration.

```
# haconf -dump -makero
```

If DBG_AGDEBUG is set, the agent framework logs for an instance of the agent appear in the agent log on the node on which the agent is running.

Enabling debug logs for IMF

Run the following commands to enable additional debug logs for Intelligent Monitoring Framework (IMF). The messages get logged in the agent-specific log file `/var/VRTSvcs/log/<agentname>_A.log`.

See "Troubleshooting Intelligent Monitoring Framework (IMF)" on page 639.

To enable additional debug logs

1. For Process, Mount, and Application agents:

```
# hatype -modify <agentname> LogDbg
DBG_AGDEBUG DBG_AGTRACE DBG_AGINFO DBG_1 DBG_2
DBG_3 DBG_4 DBG_5 DBG_6 DBG_7
```

2. For Oracle and Netlsnr agents:

```
# hatype -modify <agentname> LogDbg
DBG_AGDEBUG DBG_AGTRACE DBG_AGINFO DBG_1 DBG_2
DBG_3 DBG_4 DBG_5 DBG_6 DBG_7
DBG_8 DBG_9 DBG_10
```

3. For CFSMount agent:

```
# hatype -modify <agentname> LogDbg
DBG_AGDEBUG DBG_AGTRACE DBG_AGINFO DBG_1 DBG_2
DBG_3 DBG_4 DBG_5 DBG_6 DBG_7
DBG_8 DBG_9 DBG_10 DBG_11 DBG_12
DBG_13 DBG_14 DBG_15 DBG_16
DBG_17 DBG_18 DBG_19 DBG_20 DBG_21
```

4. For CVMvxconfigd agent, you do not have to enable any additional debug logs.
5. For AMF driver in-memory trace buffer:

```
# amfconfig -S errlevel all all
```

If you had enabled AMF driver in-memory trace buffer, you can view the additional logs using the `amfconfig -p dbglog` command.

Enabling debug logs for the VCS engine

You can enable debug logs for the VCS engine, VCS agents, and HA commands in two ways:

- To enable debug logs at run-time, use the `halog -addtags` command.

- To enable debug logs at startup, use the VCS_DEBUG_LOG_TAGS environment variable. You must set the VCS_DEBUG_LOG_TAGS before you start HAD or before you run HA commands.
  See " VCS environment variables" on page 76.
  Examples:

```
# export VCS_DEBUG_LOG_TAGS="DBG_TRACE DBG_POLICY"
# hastart

# export VCS_DEBUG_LOG_TAGS="DBG_AGINFO DBG_AGDEBUG DBG_AGTRACE"
# hastart

# export VCS_DEBUG_LOG_TAGS="DBG_IPM"
# hagrp -list
```

Note: Debug log messages are verbose. If you enable debug logs, log files might fill up quickly.

Enabling debug logs for VxAT

VxAT writes messages to the client log and the server log.

Client log

You can review the messages in this log to debug SSL communication issues and local host authentication issues.

To enable client logging

◆ Run the following commands sequentially:

```
# export AtClientDebugLog=4:/file/name

# export VRTSat_API_DEBUG_LEVEL=10

# export VRTSat_API_DEBUG_FILE=/tmp/apilog.txt
```

The `apilog.txt` file indicates the API calling sequence and the AT library load path.

To reset this variable, run:

```
# export AtClientDebugLog=
```

Server Log

You can review the messages in this log to debug authentication issues on the server side.

To enable server logging

- Set the DebugLevel attribute value in the AT configuration file at:

  `/var/VRTSvcs/vcsauth/data/VCSAUTHSERVER/root/.VRTSat/profile/VRTSatlocal.conf`

  This attribute takes values from 1 (lowest) through 4 (highest).

To view the server log messages, see the broker log at:

`/var/VRTSvcs/log/vcsauthserver.log`

By default, the debug logs are written to the /var/VRTSvcs/log/vcsauthserver.log file. You may want to change the default location for various reasons, like audits and security.

To change debug log location

1. Stop `vcsauthserver`.

   ```
   # /opt/VRTSvcs/bin/vcsauth/vcsauthserver/bin/vcsauthserver.sh
   stop
   ```

2. In the broker configuration file, `VRTSatlocal.conf`, modify the `UpdatedDebugLogFileName` attribute to include the desired file name and path.

3. Start `vcsauthserver`.

   ```
   # /opt/VRTSvcs/bin/vcsauth/vcsauthserver/bin/vcsauthserver.sh
   ```

From this point onwards, the debug logs are written to the custom file location.

About debug log tags usage

The following table illustrates the use of debug tags:

Entity	Debug logs used
Agent functions	DBG_1 to DBG_21
Agent framework	DBG_AGTRACE DBG_AGDEBUG DBG_AGINFO
Icmp agent	DBG_HBFW_TRACE DBG_HBFW_DEBUG DBG_HBFW_INFO

Entity	Debug logs used
HAD	DBG_AGENT (for agent-related debug logs)
	DBG_ALERTS (for alert debug logs)
	DBG_CTEAM (for GCO debug logs)
	DBG_GAB, DBG_GABIO (for GAB debug messages)
	DBG_GC (for displaying global counter with each log message)
	DBG_INTERNAL (for internal messages)
	DBG_IPM (for Inter Process Messaging)
	DBG_JOIN (for Join logic)
	DBG_LIC (for licensing-related messages)
	DBG_NTEVENT (for NT Event logs)
	DBG_POLICY (for engine policy)
	DBG_RSM (for RSM debug messages)
	DBG_TRACE (for trace messages)
	DBG_SECURITY (for security-related messages)
	DBG_LOCK (for debugging lock primitives)
	DBG_THREAD (for debugging thread primitives)
	DBG_HOSTMON (for HostMonitor debug logs)

Gathering VCS information for support analysis

You must run the `hagetcf` command to gather information when you encounter issues with VCS. Veritas Technical Support uses the output of these scripts to assist with analyzing and solving any VCS problems. The `hagetcf` command gathers information about the installed software, cluster configuration, systems, logs, and related information and creates a gzip file.

See the `hagetcf`(1M) manual page for more information.

To gather VCS information for support analysis

◆ Run the following command on each node:

```
# /opt/VRTSvcs/bin/hagetcf
```

The command prompts you to specify an output directory for the gzip file. You may save the gzip file to either the default `/tmp` directory or a different directory.

Troubleshoot and fix the issue.

See "Troubleshooting the VCS engine" on page 625.

See "Troubleshooting VCS startup" on page 632.

See "Troubleshooting service groups" on page 642.

See "Troubleshooting resources" on page 648.

See "Troubleshooting notification" on page 664.

See "Troubleshooting and recovery for global clusters" on page 664.

See "Troubleshooting the steward process" on page 668.

If the issue cannot be fixed, then contact Veritas technical support with the file that the `hagetcf` command generates.

Verifying the metered or forecasted values for CPU, Mem, and Swap

This section lists commands for verifying or troubleshooting the metered and forecast data for the parameters metered, such as CPU, Mem, and Swap.

The VCS HostMonitor agent stores metered available capacity values for CPU, Mem, and Swap in absolute values in MHz, MB, and MB units respectively in a respective statlog database. The database files are present in /opt/VRTSvcs/stats. The primary database files are *.vcs_host_stats.data* and *.vcs_host_stats.index*. The other files present are used for archival purposes.

The HostMonitor agent uses statlog to get the forecast of available capacity and updates the system attribute HostAvailableForecast in the local system.

To gather the data when VCS is running, perform the following steps:

1. Stop the HostMonitor agent and restart it after you complete troubleshooting, thus letting you verify the auto-populated value for the system attribute HostAvailableForecast.
2. Copy the following statlog database files to a different location.
   - /var/VRTSvcs/stats/.vcs_host_stats.data
   - /var/VRTSvcs/stats/.vcs_host_stats.index

3. Save the HostAvailableForecast values for comparison.
4. Now you can restart the HostMonitor agent.
5. Gather the data as follows:
   - To view the metered data, run the following command

     ```
     # /opt/VRTSvcs/bin/vcsstatlog
      --dump /var/VRTSvcs/stats/copied_vcs_host_stats
     ```

   - To get the forecasted available capacity for CPU, Mem, and Swap for a system in cluster, run the following command on the system on which you copied the statlog database:

     ```
     # /opt/VRTSvcs/bin/vcsstatlog --shell --load \
     /var/VRTSvcs/stats/copied_vcs_host_stats --names CPU --holts \
     --npred 1  --csv
     # /opt/VRTSvcs/bin/vcsstatlog --shell --load \
     /var/VRTSvcs/stats/copied_vcs_host_stats --names Mem --holts \
     --npred 1  --csv
     # /opt/VRTSvcs/bin/vcsstatlog --shell --load \
     /var/VRTSvcs/stats/copied_vcs_host_stats --names Swap --holts \
     --npred 1  --csv
     ```

Gathering LLT and GAB information for support analysis

You must run the `getcomms` script to gather LLT and GAB information when you encounter issues with LLT and GAB. The `getcomms` script also collects core dump and stack traces along with the LLT and GAB information.

To gather LLT and GAB information for support analysis

1. If you had changed the default value of the GAB_FFDC_LOGDIR parameter, you must again export the same variable before you run the `getcomms` script.

   See "GAB message logging" on page 615.

2. Run the following command to gather information:

   ```
   # /opt/VRTSgab/getcomms
   ```

   The script uses rsh by default. Make sure that you have configured passwordless rsh. If you have passwordless ssh between the cluster nodes, you can use the `-ssh` option. To gather information on the node that you run the command, use the `-local` option.

   Troubleshoot and fix the issue.

   See "Troubleshooting Low Latency Transport (LLT)" on page 627.

   See "Troubleshooting Group Membership Services/Atomic Broadcast (GAB)" on page 631.

   If the issue cannot be fixed, then contact Veritas technical support with the file `/tmp/commslog.<time_stamp>.tar` that the `getcomms` script generates.

Gathering IMF information for support analysis

You must run the `getimf` script to gather information when you encounter issues with IMF (Intelligent Monitoring Framework).

To gather IMF information for support analysis

- Run the following command on each node:

  ```
  # /opt/VRTSamf/bin/getimf
  ```

  Troubleshoot and fix the issue.

  See "Troubleshooting Intelligent Monitoring Framework (IMF)" on page 639.

  If the issue cannot be fixed, then contact Veritas technical support with the file that the `getimf` script generates.

Message catalogs

VCS includes multilingual support for message catalogs. These binary message catalogs (BMCs), are stored in the following default locations. The variable *language* represents a two-letter abbreviation.

`/opt/VRTS/messages/language/module_name`

The VCS command-line interface displays error and success messages in VCS-supported languages. The `hamsg` command displays the VCS engine logs in VCS-supported languages.

The BMCs are:

amf.bmc	Asynchronous Monitoring Framework (AMF) messages
fdsetup.bmc	fdsetup messages
gab.bmc	GAB command-line interface messages
gcoconfig.bmc	gcoconfig messages
hagetcf.bmc	hagetcf messages
hagui.bmc	hagui messages
haimfconfig.bmc	haimfconfig messages
hazonesetup.bmc	hazonesetup messages
hazoneverify.bmc	hazoneverify messages
llt.bmc	LLT command-line interface messages
uuidconfig.bmc	uuidconfig messages
VRTSvcsAgfw.bmc	Agent framework messages
VRTSvcsAlerts.bmc	VCS alert messages
VRTSvcsApi.bmc	VCS API messages
VRTSvcsce.bmc	VCS cluster extender messages
VRTSvcsCommon.bmc	Common module messages
VRTSvcs<*platform*>Agent.bmc	VCS bundled agent messages
VRTSvcsHad.bmc	VCS engine (HAD) messages
VRTSvcsHbfw.bmc	Heartbeat framework messages
VRTSvcsNetUtils.bmc	VCS network utilities messages
VRTSvcs<*platformagent_name*>.bmc	VCS enterprise agent messages
VRTSvcsTriggers.bmc	VCS trigger messages
VRTSvcsVirtUtils.bmc	VCS virtualization utilities messages
VRTSvcsWac.bmc	Wide-area connector process messages

vxfen*.bmc	Fencing messages

Troubleshooting the VCS engine

This topic includes information on troubleshooting the VCS engine.

See "Preonline IP check" on page 626.

HAD diagnostics

When the VCS engine HAD dumps core, the core is written to the directory `$VCS_DIAG/diag/had`. The default value for variable *$VCS_DIAG* is `/var/VRTSvcs/`.

When HAD core dumps, review the contents of the `$VCS_DIAG/diag/had` directory. See the following logs for more information:

- Operating system console log
- Engine log
- hashadow log

VCS runs the script `/opt/VRTSvcs/bin/vcs_diag` to collect diagnostic information when HAD and GAB encounter heartbeat problems. The diagnostic information is stored in the `$VCS_DIAG/diag/had` directory.

When HAD starts, it renames the directory to `had.timestamp`, where *timestamp* represents the time at which the directory was renamed.

HAD restarts continuously

When you start HAD by using the `hastart` command, HAD might restart continuously. The system is unable to come to a RUNNING state and loses its port h membership.

Recommended action: Check the `engine_A.log` file for a message with the identifier V-16-1-10125. The following message is an example:

VCS INFO V-16-1-10125 GAB timeout set to 30000 ms

The value indicates the timeout that is set for HAD to heartbeat with GAB. If system is heavily loaded, a timeout of 30 seconds might be insufficient for HAD to heartbeat with GAB. If required, set the timeout to an appropriate higher value.

DNS configuration issues cause GAB to kill HAD

If HAD is periodically killed by GAB for no apparent reason, review the HAD core files for any DNS resolver functions (res_send(), res_query(), res_search() etc) in the stack trace. The presence of DNS resolver functions may indicate DNS configuration issues.

The VCS High Availability Daemon (HAD) uses the gethostbyname() function. On UNIX platforms, if the file `/etc/nsswitch.conf` has DNS in the hosts entry, a call to the gethostbyname() function may lead to calls to DNS resolver methods.

If the name servers specified in the `/etc/resolve.conf` are not reachable or if there are any DNS configuration issues, the DNS resolver methods called may block HAD, leading to HAD not sending heartbeats to GAB in a timely manner.

Seeding and I/O fencing

When I/O fencing starts up, a check is done to make sure the systems that have keys on the coordinator points are also in the GAB membership. If the `gabconfig` command in `/etc/gabtab` allows the cluster to seed with less than the full number of systems in the cluster, or the cluster is forced to seed with the `gabconfig -x` command, it is likely that this check will not match. In this case, the fencing module will detect a possible split-brain condition, print an error, and HAD will not start.

It is recommended to let the cluster automatically seed when all members of the cluster can exchange heartbeat signals to each other. In this case, all systems perform the I/O fencing key placement after they are already in the GAB membership.

Preonline IP check

You can enable a preonline check of a failover IP address to protect against network partitioning. The check pings a service group's configured IP address to verify that it is not already in use. If it is, the service group is not brought online.

A second check verifies that the system is connected to its public network and private network. If the system receives no response from a broadcast ping to the public network and a check of the private networks, it determines the system is isolated and does not bring the service group online.

To enable the preonline IP check, do one of the following:

- If preonline trigger script is not already present, copy the preonline trigger script from the sample triggers directory into the triggers directory:

  ```
  # cp /opt/VRTSvcs/bin/sample_triggers/VRTSvcs/preonline_ipc
  /opt/VRTSvcs/bin/triggers/preonline
  ```

Change the file permissions to make it executable.

- If preonline trigger script is already present, create a directory such as `/preonline` and move the existing preonline trigger as T0preonline to that directory. Copy the preonline_ipc trigger as T1preonline to the same directory.
- If you already use multiple triggers, copy the preonline_ipc trigger as TNpreonline, where TN is the next higher TNumber.

Troubleshooting Low Latency Transport (LLT)

This section includes error messages associated with Low Latency Transport (LLT) and provides descriptions and the recommended action.

LLT startup script displays errors

If more than one system on the network has the same clusterid-nodeid pair and the same Ethernet sap/UDP port, then the LLT startup script displays error messages similar to the following:

```
LLT lltconfig ERROR V-14-2-15238 node 1 already exists
in cluster 8383 and has the address - 00:18:8B:E4:DE:27
LLT lltconfig ERROR V-14-2-15241 LLT not configured,
use -o to override this warning
LLT lltconfig ERROR V-14-2-15664 LLT could not
configure any link
LLT lltconfig ERROR V-14-2-15245 cluster id 1 is
already being used by nid 0 and has the
address - 00:04:23:AC:24:2D
LLT lltconfig ERROR V-14-2-15664 LLT could not
configure any link
```

Recommended action: Ensure that all systems on the network have unique clusterid-nodeid pair. You can use the `lltdump -f device -D` command to get the list of unique clusterid-nodeid pairs connected to the network. This utility is available only for LLT-over-ethernet.

LLT detects cross links usage

If LLT detects more than one link of a system that is connected to the same network, then LLT logs a warning message similar to the following in the `/var/adm/streams/error.date` log:

```
LLT WARNING V-14-1-10498 recvarpack cross links? links 0 and 2 saw
the same peer link number 1 for node 1
```

Recommended Action: This is an informational message. LLT supports cross links. However, if this cross links is not an intentional network setup, then make sure that no two links from the same system go to the same network. That is, the LLT links need to be on separate networks.

LLT link status messages

Table 22-1 describes the LLT logs messages such as trouble, active, inactive, or expired in the syslog for the links.

Table 22-1 LLT link status messages

Message	Description and Recommended action

`LLT INFO V-14-1-10205 link 1 (link_name) node 1 in trouble`	This message implies that LLT did not receive any heartbeats on the indicated link from the indicated peer node for LLT peertrouble time. The default LLT peertrouble time is 2s for hipri links and 4s for lo-pri links.
	Recommended action: If these messages sporadically appear in the syslog, you can ignore them. If these messages flood the syslog, then perform one of the following:
	■ Increase the peertrouble time to a higher value (but significantly lower than the peerinact value). Run the following command:
	`lltconfig -T peertrouble:<value> for hipri link`
	`lltconfig -T peertroublelo:<value> for lopri links.`
	See the `lltconfig`(1m) manual page for details.
	■ Replace the LLT link.
	See "Adding and removing LLT links" on page 103.

`LLT INFO V-14-1-10024 link 0 (link_name) node 1 active`	This message implies that LLT started seeing heartbeats on this link from that node. Recommended action: No action is required. This message is informational.

Table 22-1 LLT link status messages *(continued)*

Message	Description and Recommended action
`LLT INFO V-14-1-10032 link 1 (link_name) node 1 inactive 5 sec (510)`	This message implies that LLT did not receive any heartbeats on the indicated link from the indicated peer node for the indicated amount of time. If the peer node has not actually gone down, check for the following: ■ Check if the link has got physically disconnected from the system or switch. ■ Check for the link health and replace the link if necessary. See "Adding and removing LLT links" on page 103.

`LLT INFO V-14-1-10510 sent hbreq (NULL) on link 1 (link_name) node 1. 4 more to go.`
`LLT INFO V-14-1-10510 sent hbreq (NULL) on link 1 (link_name) node 1. 3 more to go.`
`LLT INFO V-14-1-10510 sent hbreq (NULL) on link 1 (link_name) node 1. 2 more to go.`
`LLT INFO V-14-1-10032 link 1 (link_name) node 1 inactive 6 sec (510)`
`LLT INFO V-14-1-10510 sent hbreq (NULL) on link 1 (link_name) node 1. 1 more to go.`
`LLT INFO V-14-1-10510 sent hbreq (NULL) on link 1 (link_name) node 1. 0 more to go.`
`LLT INFO V-14-1-10032 link 1 (link_name) node 1 inactive 7 sec (510)`
`LLT INFO V-14-1-10509 link 1 (link_name) node 1 expired`
	This message implies that LLT did not receive any heartbeats on the indicated link from the indicated peer node for more than LLT peerinact time. LLT attempts to request heartbeats (sends 5 hbreqs to the peer node) and if the peer node does not respond, LLT marks this link as "expired" for that peer node.
	Recommended action: If the peer node has not actually gone down, check for the following:
	■ Check if the link has got physically disconnected from the system or switch.
	■ Check for the link health and replace the link if necessary.
	See "Adding and removing LLT links" on page 103.

`LLT INFO V-14-1-10499 recvarpreq link 0 for node 1 addr change from 00:00:00:00:00:00 to 00:18:8B:E4:DE:27`	This message is logged when LLT learns the peer node's address. Recommended action: No action is required. This message is informational.

Table 22-1 LLT link status messages *(continued)*

Message	Description and Recommended action

On local node that detects the link failure:
`LLT INFO V-14-1-10519 link 0 down`
`LLT INFO V-14-1-10585 local link 0 down for 1 sec`
 	
`LLT INFO V-14-1-10586 send linkdown_ntf on link 1 for local link 0`
`LLT INFO V-14-1-10590 recv linkdown_ack from node 1 on link 1 for local link 0`
`LLT INFO V-14-1-10592 received ack from all the connected nodes`
 	
On peer nodes:
`LLT INFO V-14-1-10589 recv linkdown_ntf from node 0 on link 1 for peer link 0`
`LLT INFO V-14-1-10587 send linkdown_ack to node 0 on link 1 for peer link 0`
	These messages are printed when you have enabled LLT to detect faster failure of links. When a link fails or is disconnected from a node (cable pull, switch failure, and so on), LLT on the local node detects this event and propagates this information to all the peer nodes over the LLT hidden link. LLT marks this link as disconnected when LLT on the local node receives the acknowledgment from all the nodes.

Cluster does not form after installing and starting VCS or SFHA

When LLT and GAB services are started cluster wide, LLT drivers on all the nodes form connections with one another on GAB ports. In some setup configurations, operating system's firewall blocks LLT state-machine related packets which are required for forming connections. As a result, the cluster does not form and the gabconfig –a command does not show the membership.

Recommended action:

- RHEL 7 , RHEL 8 and supported RHEL-compatible distributions:
  # `systemctl stop firewalld`
  To make the changes persistent across restart:
  # `chkconfig --level 235 firewalld off`
- SLES 12 and SLEs 15: # `systemctl stop SuSEfirewall2`
  To make the changes persistent across restar, use one of the following command:
  # `chkconfig --level 235 SuSEfirewall2 off`
  or
  # `systemctl disable SuSEfirewall2`

Troubleshooting Group Membership Services/Atomic Broadcast (GAB)

This section includes error messages associated with Group Membership Services/Atomic Broadcast (GAB) and provides descriptions and the recommended action.

Delay in port reopen

If a GAB port was closed and reopened before LLT state cleanup action is completed, then GAB logs a message similar to the following:

```
GAB INFO V-15-1-20102 Port v: delayed reopen
```

Recommended Action: If this issue occurs during a GAB reconfiguration, and does not recur, the issue is benign. If the issue persists, collect commslog from each node, and contact Veritas support.

Node panics due to client process failure

If VCS daemon does not heartbeat with GAB within the configured timeout specified in VCS_GAB_TIMEOUT (default 30sec) environment variable, the node panics with a message similar to the following:

```
GAB Port h halting node due to client process failure at 3:109
```

GABs attempt (five retries) to kill the VCS daemon fails if VCS daemon is stuck in the kernel in an uninterruptible state or the system is heavily loaded that the VCS daemon cannot die with a SIGKILL.

Recommended Action:

- In case of performance issues, increase the value of the VCS_GAB_TIMEOUT environment variable to allow VCS more time to heartbeat.
  See " VCS environment variables" on page 76.
- In case of a kernel problem, configure GAB to not panic but continue to attempt killing the VCS daemon.

  Do the following:

  - Run the following command on each node:

    ```
    gabconfig -k
    ```

  - Add the "-k" option to the gabconfig command in the `/etc/gabtab` file:

```
gabconfig -c -k -n 6
```

- In case the problem persists, collect sar or similar output, collect crash dumps, run the Veritas Operations and Readiness Tools (SORT) data collector on all nodes, and contact Veritas Technical Support.

Troubleshooting VCS startup

This topic includes error messages associated with starting VCS (shown in bold text), and provides descriptions of each error and the recommended action.

"VCS:10622 local configuration missing"

The local configuration is missing.

Recommended Action: Start the VCS engine, HAD, on another system that has a valid configuration file. The system with the configuration error pulls the valid configuration from the other system.

Another method is to install the configuration file on the local system and force VCS to reread the configuration file. If the file appears valid, verify that is not an earlier version.

Type the following commands to verify the configuration:

```
# cd /etc/VRTSvcs/conf/config
# hacf -verify .
```

"VCS:10623 local configuration invalid"

The local configuration is invalid.

Recommended Action: Start the VCS engine, HAD, on another system that has a valid configuration file. The system with the configuration error "pulls" the valid configuration from the other system.

Another method is to correct the configuration file on the local system and force VCS to re-read the configuration file. If the file appears valid, verify that is not an earlier version.

Type the following commands to verify the configuration:

```
# cd /etc/VRTSvcs/conf/config
# hacf -verify .
```

"VCS:11032 registration failed. Exiting"

GAB was not registered or has become unregistered.

Recommended Action: GAB is registered by the `gabconfig` command in the file /etc/gabtab. Verify that the file exists and that it contains the command `gabconfig -c`.

GAB can become unregistered if LLT is set up incorrectly. Verify that the configuration is correct in /etc/lltab. If the LLT configuration is incorrect, make the appropriate changes and reboot.

"Waiting for cluster membership."

This indicates that GAB may not be seeded. If this is the case, the command `gabconfig -a` does not show any members, and the following messages may appear on the console or in the event log.

```
GAB: Port a registration waiting for seed port membership
 GAB: Port h registration waiting for seed port membership
```

Troubleshooting issues with systemd unit service files

This section lists the scenarios in which you may encounter issues with systemd unit service files and describes how to resolve those issues.

If a unit service has failed and the corresponding module is still loaded, systemd cannot unload it and so its package cannot be removed

A unit service may go into the Failed state during startup while the corresponding module is in the Loaded state due to some configuration issue. If this situation occurs, the module cannot be cleanly unloaded using `systemd` commands.

For example, you may encounter this message when you start the `vxfen` service:

```
localhost]:~ # systemctl start vxfen
Job for vxfen.service failed because the control process exited
 with error code. See "systemctl status vxfen.service" and
 "journalctl -xe" for details.
```

If you encounter this message, view the status of service:

```
localhost]:~ # systemctl status -l vxfen
? vxfen.service - VERITAS I/O Fencing (VxFEN)
Loaded: loaded (/opt/VRTSvcs/vxfen/bin/vxfen; enabled;
 vendor preset: disabled)
Active: failed (Result: exit-code) since Wed 2017-04-19 17:40:13 IST;
 14s ago
Process: 14721 ExecStart=/opt/VRTSvcs/vxfen/bin/vxfen start 2>&1
 (code=exited, status=1/FAILURE)
Apr 19 17:40:12 localhost] systemd[1]: Starting VERITAS I/O Fencing
 (VxFEN)...
Apr 19 17:40:12 localhost] vxfen[14721]: Starting vxfen..
Apr 19 17:40:12 localhost] vxfen[14721]: Loaded 4.4.21-69-default on
 kernel 4.4.21-69-default
Apr 19 17:40:13 localhost] vxfen[14721]: vxfen cannot generate vxfentab
 due to missing file /etc/vxfendg.
Apr 19 17:40:13 localhost] systemd[1]: vxfen.service: Control process
 exited, code=exited status=1
Apr 19 17:40:13 localhost] systemd[1]: Failed to start VERITAS I/O
 Fencing (VxFEN).
Apr 19 17:40:13 localhost] systemd[1]: vxfen.service: Unit entered
 failed state.
Apr 19 17:40:13 localhost] systemd[1]: vxfen.service: Failed with result
 'exit-code'.
```

Then, view the status of the module:

```
localhost]:~ # lsmod | grep vxfen
vxfen 372992 0
veki 19214 4 gab,llt,vxfen

localhost]:~ # /opt/VRTSvcs/vxfen/bin/vxfen status
loaded
```

The `systemd` daemon does not accept a `systemctl stop` *`unitServiceFile`* command when the service is in the Failed state or the Inactive state.

For example, if the `vxfen` service is in the Failed or the Inactive state, the following command fails to unload the module successfully:

```
systemctl stop vxfen
```

Additionally, if a module is in the Loaded state, the corresponding rpm package cannot be removed during uninstallation.

Recommended action

To cleanly stop and unload the module, run the `stop` command from the source script file as follows:

```
serviceSourceScript stop
```

For example:

```
localhost]:~ # /opt/VRTSvcs/vxfen/bin/vxfen stop
VxFEN: Module already unconfigured. Only unloading

localhost]:~ #/opt/VRTSvcs/vxfen/bin/vxfen status
not loaded
```

To view the source path of any service, you can use the `systemctl` command as follows:

```
# systemctl show unitServiceFile -p SourcePath
```

For example:

```
# systemctl show vxfen -p SourcePath
SourcePath=/opt/VRTSvcs/vxfen/bin/vxfen
```

After the module is cleanly unloaded, you can perform the next startup action or proceed to uninstall the associated package.

If a unit service is active and the corresponding process is stopped outside of systemd, the service cannot be started again using ‘systemctl start’

While a unit service is in the ‘active (running)’ or ‘active (exited)’ state, the corresponding process may be stopped outside of the `systemd` context. In that case, you cannot start the service again using the `systemctl start` command.

In the following example, the HA process (had) is not running, but the unit service appears to be in the active (running) state:

```
( root@localhost-vm1 ):[ ~ ]#systemctl status vcs
vcs.service - VERITAS Cluster Server (VCS)
Loaded: loaded (/opt/VRTSvcs/bin/vcs; enabled)
Active: active (running) since Sat 2017-04-22 00:10:24 IST;
 2 days ago
Process: 6671 ExecStart=/opt/VRTSvcs/bin/vcs start 2>&1
 (code=exited, status=0/SUCCESS)
CGroup: /system.slice/vcs.service
|-7724 /usr/lib/fs/vxfs/vxfsckd -p /var/adm/cfs/vxfsckd-pid
```

```
`-8046 /usr/sbin/vxnotify -g crsdg2324 -icfspdvACLMSk
Apr 22 00:16:30 localhost-vm1 Had[6753]: VCS ERROR V-16-2-13066
(localhost-vm1) Agent is calling clean for resou...leted.
Apr 22 00:16:34 localhost-vm1 Had[6753]: VCS ERROR V-16-2-13066
(localhost-vm0) Agent is calling clean for resou...leted.
Apr 22 00:17:31 localhost-vm1 AgentFramework[6835]: VCS ERROR
 V-16-2-13006 Thread(140152660154112) Resource(cssd): ...time.
Apr 22 00:17:31 localhost-vm1 Had[6753]: VCS ERROR
 V-16-2-13006 (localhost-vm1) Resource(cssd): clean procedure
 ... time.
Apr 22 00:17:35 localhost-vm1 Had[6753]: VCS ERROR
 V-16-2-13006 (localhost-vm0) Resource(cssd): clean procedure
 ... time.
Apr 22 00:22:37 localhost-vm1 AgentFramework[6835]: VCS ERROR
 V-16-2-13078 Thread(140152660154112) Resource(cssd) -...mpts.
Apr 22 00:22:37 localhost-vm1 AgentFramework[6835]: VCS ERROR
 V-16-2-13071 Thread(140152660154112) Resource(cssd): ...t(0).
Apr 22 00:22:37 localhost-vm1 Had[6753]: VCS ERROR V-16-1-54031
 Resource cssd (Owner: Unspecified, Group: cvm) is ...t-vm24
Apr 22 00:22:44 localhost-vm1 Had[6753]: VCS ERROR V-16-1-54031
 Resource cssd (Owner: Unspecified, Group: cvm) is ...t-vm23
Apr 24 21:51:11 localhost-vm1 systemd[1]: Started VERITAS
 Cluster Server (VCS).
Hint: Some lines were ellipsized, use -l to show in full.

( root@localhost-vm1 ):[ ~ ]# ps -ef | grep had
root 13735 10930 0 21:52 pts/2 00:00:00 grep --color=auto had

( root@localhost-vm1 ):[ ~ ]#/opt/VRTSvcs/bin/vcs status
had is stopped
```

In this case, `systemd` cannot process the `systemctl start vcs` command.

Recommended action

To work around this issue, run the following commands sequentially:

```
( root@localhost-vm1 )[ ~ ] # systemctl stop vcs
...

( root@localhost-vm1 )[ ~ ] # systemctl start vcs
...
```

Or, run the restart command as follows:

```
( root@localhost-vm1 )[ ~ ] #systemctl restart vcs
```

Verify that unit service is running:

```
( root@localhost-vm1 ):[ ~ ]# systemctl status vcs
vcs.service - VERITAS Cluster Server (VCS)
Loaded: loaded (/opt/VRTSvcs/bin/vcs; enabled)
Active: active (running) since Mon 2017-04-24 21:57:49 IST;
 3s ago
Process: 15339 ExecStop=/opt/VRTSvcs/bin/vcs stop 2>&1
 (code=exited, status=0/SUCCESS)
Process: 15356 ExecStart=/opt/VRTSvcs/bin/vcs start 2>&1
 (code=exited, status=0/SUCCESS)
CGroup: /system.slice/vcs.service
|- 7724 /usr/lib/fs/vxfs/vxfsckd -p /var/adm/cfs/vxfsckd-pid
|- 8046 /usr/sbin/vxnotify -g crsdg2324 -icfspdvACLMSk
|-15394 /opt/VRTSvcs/bin/had
`-15400 /opt/VRTSvcs/bin/hashadow
Apr 24 21:57:49 localhost-vm1 systemd[1]: Starting VERITAS
 Cluster Server (VCS)...
Apr 24 21:57:49 localhost-vm1 vcs[15356]: Starting VCS: [ OK ]
Apr 24 21:57:49 localhost-vm1 systemd[1]: Started VERITAS
 Cluster Server (VCS).
Apr 24 21:57:49 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-10619 'HAD' starting on: localhost-vm1
Apr 24 21:57:49 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-10620 Waiting for local cluster configuration status
Apr 24 21:57:49 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-10625 Local cluster configuration valid
Apr 24 21:57:49 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-11034 Registering for cluster membership
Apr 24 21:57:49 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-11035 Waiting for cluster membership
Apr 24 21:57:54 localhost-vm1 Had[15394]: VCS INFO
 V-16-1-10077 Received new cluster membership
Apr 24 21:57:54 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-10086 System localhost-vm1 (Node '1') is in Regular
 Membership - Membership: 0x2
Apr 24 21:57:54 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-10073 Building from local configuration
Apr 24 21:57:55 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-10066 Entering RUNNING state
Apr 24 21:57:55 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-50311 VCS Engine: running with security OFF
```

```
Apr 24 22:01:22 localhost-vm1 Had[15394]: VCS INFO
 V-16-1-10077 Received new cluster membership
Apr 24 22:01:22 localhost-vm1 Had[15394]: VCS NOTICE
 V-16-1-10086 System localhost-vm0 (Node '0') is in Regular
 Membership - Membership: 0x3
```

Verify that the corresponding services are started:

```
( root@localhost-vm1 ):[ ~ ]# /opt/VRTSvcs/bin
/vcs status
had (pid 15394) is running...

( root@localhost-vm1 ):[ ~ ]# ps -ef | grep had
root 15394 1 0 21:57 ? 00:00:00 /opt/VRTSvcs/bin/had
root 15400 1 0 21:57 ? 00:00:00 /opt/VRTSvcs/bin/hashadow
```

If a unit service takes longer than the default timeout to stop or start the corresponding service, it goes into the Failed state

Sometimes, the unit service file may take more time than the default timeout value to stop or to start the service. In this scenario, the service goes into the ‘failed’ state.

For example:

```
( root@localhost-vm1 )[ ~ ] # systemctl status vcs
vcs.service - VERITAS Cluster Server (VCS)
Loaded: loaded (/opt/VRTSvcs/bin/vcs; enabled)
Active: failed (Result: timeout) since Tue 2017-04-25 21:01:39 IST;
 6s ago
Process: 26546 ExecStart=/opt/VRTSvcs/bin/vcs start 2>&1
 (code=exited, status=0/SUCCESS)
Apr 25 21:00:09 localhost systemd[1]: Stopping VERITAS Cluster
 Server (VCS)...
Apr 25 21:01:07 localhost AgentFramework[26625]: VCS ERROR
 V-16-20006-1005 CVMCluster:cvm_clus:monitor:node - st...ster
 reason: user initiated stop
Apr 25 21:01:07 localhost AgentFramework[26625]: VCS ERROR
 V-16-2-13066 Thread(140447847638784) Agent is calling...ted.
Apr 25 21:01:07 localhost Had[26588]: VCS ERROR V-16-2-13066
 (localhost) Agent is calling clean for reso...leted.
Apr 25 21:01:39 localhost systemd[1]: vcs.service stopping timed
 out. Terminating.
Apr 25 21:01:39 localhost systemd[1]: Stopped VERITAS Cluster
 Server (VCS).
```

```
Apr 25 21:01:39 localhost systemd[1]: Unit vcs.service entered
 failed state.
Hint: Some lines were ellipsized, use -l to show in full.
```

Recommended action

To work around this issue, add a custom timeout value (in seconds) to the unit service file. The `TimeoutSec` parameter lets you configure the amount of time that the system must wait before it reports that the start or stop operation of a service is has failed.

The following example displays the parameter that is used to set a custom timeout in the unit service file:

```
( root@localhost-vm1 )[ ~ ] # vim /usr/lib/systemd
/system/vcs.service
[Unit]
Description=VERITAS Cluster Server (VCS)
SourcePath=/opt/VRTSvcs/bin/vcs
...
...
[Service]
...
...
...
TimeoutSec=300
[Install]
...
```

After you specify the custom timeout value in a unit service file, you must reload the `systemd` daemon so that the configuration is updated:

```
( root@localhost-vm1 )[ ~ ] # systemctl --system
 daemon-reload
```

Then, start or stop the unit service file:

```
( root@localhost-vm1 )[ ~ ] # systemctl start vcs
```

Troubleshooting Intelligent Monitoring Framework (IMF)

Review the following logs to isolate and troubleshoot Intelligent Monitoring Framework (IMF) related issues:

- System console log for the given operating system
- VCS engine Log : /var/VRTSvcs/log/engine_A.log
- Agent specific log : /var/VRTSvcs/log/*<agentname>*_A.log
- AMF in-memory trace buffer : View the contents using the `amfconfig -p dbglog` command

See "Enabling debug logs for IMF" on page 617.

See "Gathering IMF information for support analysis" on page 623.

Table 22-2 lists the most common issues for intelligent resource monitoring and provides instructions to troubleshoot and fix the issues.

Table 22-2 IMF-related issues and recommended actions

Issue	Description and recommended action
Intelligent resource monitoring has not reduced system utilization	If the system is busy even after intelligent resource monitoring is enabled, troubleshoot as follows: ■ Check the agent log file to see whether the `imf_init` agent function has failed. If the `imf_init` agent function has failed, then do the following: ■ Make sure that the AMF_START environment variable value is set to 1. See "Environment variables to start and stop VCS modules" on page 81. ■ Make sure that the AMF module is loaded. See "Administering the AMF kernel driver" on page 109. ■ Make sure that the IMF attribute values are set correctly for the following attribute keys: ■ The value of the Mode key of the IMF attribute must be set to 1, 2, or 3. ■ The value of the MonitorFreq key of the IMF attribute must be be set to either 0 or a value greater than 0. For example, the value of the MonitorFreq key can be set to 0 for the Process agent. Refer to the appropriate agent documentation for configuration recommendations corresponding to the IMF-aware agent. Note that the IMF attribute can be overridden. So, if the attribute is set for individual resource, then check the value for individual resource. See "Resource type attributes" on page 703. ■ Verify that the resources are registered with the AMF driver. Check the `amfstat` command output. ■ Check the LevelTwoMonitorFreq attribute settings for the agent. For example, Process agent must have this attribute value set to 0. Refer to the appropriate agent documentation for configuration recommendations corresponding to the IMF-aware agent.

Table 22-2 IMF-related issues and recommended actions *(continued)*

Issue	Description and recommended action
Enabling the agent's intelligent monitoring does not provide immediate performance results	The actual intelligent monitoring for a resource starts only after a steady state is achieved. So, it takes some time before you can see positive performance effect after you enable IMF. This behavior is expected. For more information on when a steady state is reached, see the following topic: See "How intelligent resource monitoring works" on page 40.
Agent does not perform intelligent monitoring despite setting the IMF mode to 3	For the agents that use AMF driver for IMF notification, if intelligent resource monitoring has not taken effect, do the following: ■ Make sure that IMF attribute's Mode key value is set to three (3). See "Resource type attributes" on page 703. ■ Review the the agent log to confirm that `imf_init()` agent registration with AMF has succeeded. AMF driver must be loaded before the agent starts because the agent registers with AMF at the agent startup time. If this was not the case, start the AMF module and restart the agent. See "Administering the AMF kernel driver" on page 109.
AMF module fails to unload despite changing the IMF mode to 0	Even after you change the value of the Mode key to zero, the agent still continues to have a hold on the AMF driver until you kill the agent. To unload the AMF module, all holds on it must get released. If the AMF module fails to unload after changing the IMF mode value to zero, do the following: ■ Run the `amfconfig -Uof` command. This command forcefully removes all holds on the module and unconfigures it. ■ Then, unload AMF. See "Administering the AMF kernel driver" on page 109.
When you try to enable IMF for an agent, the `haimfconfig -enable -agent <agent_name>` command returns a message that IMF is enabled for the agent. However, when VCS and the respective agent is running, the `haimfconfig -display` command shows the status for agent_name as DISABLED.	A few possible reasons for this behavior are as follows: ■ The agent might require some manual steps to make it IMF-aware. Refer to the agent documentation for these manual steps. ■ The agent is a custom agent and is not IMF-aware. For information on how to make a custom agent IMF-aware, see the *Cluster Server Agent Developer's Guide*. ■ If the preceding steps do not resolve the issue, contact Veritas technical support.

Troubleshooting service groups

This topic cites the most common problems associated with bringing service groups online and taking them offline. Bold text provides a description of the problem. Recommended action is also included, where applicable.

VCS does not automatically start service group

VCS does not automatically start a failover service group if the VCS engine (HAD) in the cluster was restarted by the hashadow process.

This behavior prevents service groups from coming online automatically due to events such as GAB killing HAD because to high load, or HAD committing suicide to rectify unexpected error conditions.

System is not in RUNNING state

Recommended Action: Type `hasys -display system` to check the system status. When the system is not in running state, we may start VCS if there are no issues found.

Service group not configured to run on the system

The SystemList attribute of the group may not contain the name of the system.

Recommended Action: Use the output of the command `hagrp -display service_group` to verify the system name.

Service group not configured to autostart

If the service group is not starting automatically on the system, the group may not be configured to AutoStart, or may not be configured to AutoStart on that particular system.

Recommended Action: Use the output of the command `hagrp -display service_group AutoStartList node_list` to verify the values of the AutoStart and AutoStartList attributes.

Service group is frozen

Recommended Action: Use the output of the command `hagrp -display service_group` to verify the value of the Frozen and TFrozen attributes. Use the command `hagrp -unfreeze` to unfreeze the group. Note that VCS will not take a frozen service group offline.

Failover service group is online on another system

The group is a failover group and is online or partially online on another system.

Recommended Action: Use the output of the command `hagrp -display` *`service_group`* to verify the value of the State attribute. Use the command `hagrp -offline` to offline the group on another system.

A critical resource faulted

Output of the command `hagrp -display` *`service_group`* indicates that the service group has faulted.

Recommended Action: Use the command `hares -clear` to clear the fault.

Service group autodisabled

When VCS does not know the status of a service group on a particular system, it autodisables the service group on that system. Autodisabling occurs under the following conditions:

- When the VCS engine, HAD, is not running on the system.
  Under these conditions, all service groups that include the system in their SystemList attribute are autodisabled. This does not apply to systems that are powered off.
- When all resources within the service group are not probed on the system.

Recommended Action: Use the output of the command `hagrp -display` *`service_group`* to verify the value of the AutoDisabled attribute.

Warning: To bring a group online manually after VCS has autodisabled the group, make sure that the group is not fully or partially active on any system that has the AutoDisabled attribute set to 1 by VCS. Specifically, verify that all resources that may be corrupted by being active on multiple systems are brought down on the designated systems. Then, clear the AutoDisabled attribute for each system: `# hagrp -autoenable` *`service_group`* `-sys` *`system`*

Service group is waiting for the resource to be brought online/taken offline

Recommended Action: Review the IState attribute of all resources in the service group to locate which resource is waiting to go online (or which is waiting to be taken offline). Use the `hastatus` command to help identify the resource. See the

engine and agent logs in /var/VRTSvcs/log for information on why the resource is unable to be brought online or be taken offline.

To clear this state, make sure all resources waiting to go online/offline do not bring themselves online/offline. Use the `hagrp -flush` command or the `hagrp -flush -force` command to clear the internal state of VCS. You can then bring the service group online or take it offline on another system.

For more information on the `hagrp -flush` and `hagrp -flush -force` commands.

Warning: Exercise caution when you use the -force option. It can lead to situations where a resource status is unintentionally returned as FAULTED. In the time interval that a resource transitions from 'waiting to go offline' to 'not waiting', if the agent has not completed the offline agent function, the agent may return the state of the resource as OFFLINE. VCS considers such unexpected offline of the resource as FAULT and starts recovery action that was not intended.

Service group is waiting for a dependency to be met.

Recommended Action: To see which dependencies have not been met, type `hagrp -dep` *`service_group`* to view service group dependencies, or `hares -dep` *`resource`* to view resource dependencies.

Service group not fully probed.

This occurs if the agent processes have not monitored each resource in the service group. When the VCS engine, HAD, starts, it immediately "probes" to find the initial state of all of resources. (It cannot probe if the agent is not returning a value.) A service group must be probed on all systems included in the SystemList attribute before VCS attempts to bring the group online as part of AutoStart. This ensures that even if the service group was online prior to VCS being brought up, VCS will not inadvertently bring the service group online on another system.

Recommended Action: Use the output of `hagrp -display` *`service_group`* to see the value of the ProbesPending attribute for the system's service group. The value of the ProbesPending attribute corresponds to the number of resources which are not probed, and it should be zero when all resources are probed. To determine which resources are not probed, verify the local "Probed" attribute for each resource on the specified system. Zero means waiting for probe result, 1 means probed, and 2 means VCS not booted. See the engine and agent logs for information.

Service group does not fail over to the forecasted system

This behaviour is expected when other events occur between the time the `hagrp -forecast` command was executed and the time when the service group or system actually fail over or switch over. VCS might select a system which is different than one mentioned in the `hagrp -forecast` command.

VCS logs detailed messages in the engine log, including the reason for selecting a specific node as a target node over other possible target nodes.

Service group does not fail over to the BiggestAvailable system even if FailOverPolicy is set to BiggestAvailable

Sometimes, a service group might not fail over to the biggest available system even when FailOverPolicy is set to BiggestAvailable.

To troubleshoot this issue, check the engine log located in */var/VRTSvcs/log/engine_A.log* to find out the reasons for not failing over to the biggest available system. This may be due to the following reasons:

- If , one or more of the systems in the service group's SystemList did not have forecasted available capacity, you see the following message in the engine log:
  One of the systems in SystemList of group group_name, system system_name does not have forecasted available capacity updated
- If the `hautil -sys` command does not list forecasted available capacity for the systems, you see the following message in the engine log:
  Failed to forecast due to insufficient data
  This message is displayed due to insufficient recent data to be used for forecasting the available capacity.
  The default value for the MeterInterval key of the cluster attribute MeterControl is 120 seconds. There will be enough recent data available for forecasting after 3 metering intervals (6 minutes) from time the VCS engine was started on the system. After this, the forecasted values are updated every ForecastCycle * MeterInterval seconds. The ForecastCycle and MeterInterval values are specified in the cluster attribute MeterControl.
- If one or more of the systems in the service group's SystemList have stale forecasted available capacity, you can see the following message in the engine log:
  System system_name has not updated forecasted available capacity since last 2 forecast cycles
  This issue is caused when the HostMonitor agent stops functioning. Check if HostMonitor agent process is running by issuing one of the following commands on the system which has stale forecasted values:

- `# ps -aef|grep HostMonitor`
- `# haagent -display HostMonitor`

  If HostMonitor agent is not running, you can start it by issuing the following command:

  ```
  # hagent -start HostMonitor -sys system_name
  ```

  Even if HostMonitor agent is running and you see the above message in the engine log, it means that the HostMonitor agent is not able to forecast, and it will log error messages in the HostMonitor_A.log file in the /var/VRTSvcs/log/ directory.

Restoring metering database from backup taken by VCS

When VCS detects that the metering database has been corrupted or has been accidently deleted outside VCS, it creates a backup of the database. The data available in memory is backed up and a message is logged, and administrative intervention is required to restore the database and reinstate metering and forecast.

The following log message will appear in the engine log:

```
The backup of metering database from in-memory data is created and
saved in <path for metering database backup>. Administrative
intervention required to use the backup database.
```

To restore the database for metering and forecast functionality

1. Stop the HostMonitor agent using the following command:

   ```
   # /opt/VRTSvcs/bin/haagent -stop HostMonitor -force -sys system name
   ```

2. Delete the database if it exists.

   ```
   # rm /var/VRTSvcs/stats/.vcs_host_stats.data \
   /var/VRTSvcs/stats/.vcs_host_stats.index
   ```

3. Restore metering database from the backup.

   ```
   # cp /var/VRTSvcs/stats/.vcs_host_stats_bkup.data \
   /var/VRTSvcs/stats/.vcs_host_stats.data
   ```

   ```
   # cp /var/VRTSvcs/stats/.vcs_host_stats_bkup.index \
   /var/VRTSvcs/stats/.vcs_host_stats.index
   ```

4 Setup the database.

```
# /opt/VRTSvcs/bin/vcsstatlog --setprop \
/var/VRTSvcs/stats/.vcs_host_stats rate 120
```

```
# /opt/VRTSvcs/bin/vcsstatlog --setprop \
/var/VRTSvcs/stats/.vcs_host_stats compressto \
/var/VRTSvcs/stats/.vcs_host_stats_daily
```

```
# /opt/VRTSvcs/bin/vcsstatlog --setprop \
/var/VRTSvcs/stats/.vcs_host_stats compressmode avg
```

```
# /opt/VRTSvcs/bin/vcsstatlog --setprop \
/var/VRTSvcs/stats/.vcs_host_stats compressfreq 24h
```

5 Start the HostMonitor agent.

```
# /opt/VRTSvcs/bin/haagent -start HostMonitor -sys system name
```

Initialization of metering database fails

The metering database can fail to initialize if there is an existing corrupt database. In this case, VCS cannot open and initialize the database.

The following log message will appear in the engine log:

```
Initialization of database to save metered data failed and will not
be able to get forecast. Error corrupt statlog database, error
code=19, errno=0
```

To restore the metering and forecast functionality,

1 Stop the HostMonitor agent using the following command

```
# /opt/VRTSvcs/bin/haagent -stop HostMonitor -force -sys system name
```

2 Delete the metering database.

```
# rm /var/VRTSvcs/stats/.vcs_host_stats.data \
/var/VRTSvcs/stats/.vcs_host_stats.index
```

3 Start the HostMonitor agent.

```
# /opt/VRTSvcs/bin/haagent -start HostMonitor -sys system name
```

Error message appears during service group failover or switch

During a service group failover or switch, if the system tries to unreserve more than the reserved capacity, the following error message appears:

```
VCS ERROR V-16-1-56034 ReservedCapacity for attribute dropped below zero for
system sys1 when unreserving for group service_group; setting to zero.
```

This error message appears only if the service group's load requirements change during the transition period.

Recommended Action: No action is required. You can ignore this message.

Troubleshooting resources

This topic cites the most common problems associated with bringing resources online and taking them offline. Bold text provides a description of the problem. Recommended action is also included, where applicable.

Service group brought online due to failover

VCS attempts to bring resources online that were already online on the failed system, or were in the process of going online. Each parent resource must wait for its child resources to be brought online before starting.

Recommended Action: Verify that the child resources are online.

Waiting for service group states

The state of the service group prevents VCS from bringing the resource online.

Recommended Action: Review the state of the service group.

Waiting for child resources

One or more child resources of parent resource are offline.

Recommended Action: Bring the child resources online first.

Waiting for parent resources

One or more parent resources are online.

Recommended Action: Take the parent resources offline first.

See “Troubleshooting resources” on page 648.

Waiting for resource to respond

The resource is waiting to come online or go offline, as indicated. VCS directed the agent to run an online entry point for the resource.

Recommended Action: Verify the resource's IState attribute. See the engine and agent logs in /var/VRTSvcs/engine_A.log and /var/VRTSvcs/*agent*_A.log for information on why the resource cannot be brought online.

Agent not running

The resource's agent process is not running.

Recommended Action: Use `hastatus -summary` to see if the agent is listed as faulted. Restart the agent:

```
# haagent -start resource_type -sys system
```

Invalid agent argument list.

The scripts are receiving incorrect arguments.

Recommended Action: Verify that the arguments to the scripts are correct. Use the output of `hares -display resource` to see the value of the ArgListValues attribute. If the ArgList attribute was dynamically changed, stop the agent and restart it.

To stop the agent:

◆ `# haagent -stop resource_type -sys system`

To restart the agent

◆ `# haagent -start resource_type -sys system`

The Monitor entry point of the disk group agent returns ONLINE even if the disk group is disabled

This is expected agent behavior. VCS assumes that data is being read from or written to the volumes and does not declare the resource as offline. This prevents potential data corruption that could be caused by the disk group being imported on two hosts.

You can deport a disabled disk group when all I/O operations are completed or when all volumes are closed. You can then reimport the disk group to the same system. Reimporting a disabled disk group may require a system reboot.

Note: A disk group is disabled if data including the kernel log, configuration copies, or headers in the private region of a significant number of disks is invalid or inaccessible. Volumes can perform read-write operations if no changes are required to the private regions of the disks.

Troubleshooting sites

The following sections discuss troubleshooting the sites. Bold text provides a description of the problem. Recommended action is also included, where applicable.

Online propagate operation was initiated but service group failed to be online

Recommended Action: Check the State and PolicyIntention attributes for the parent and child service groups using the command:

```
<hagrp -display -attribute State PolicyIntention -all>
```

And check for errors in the engine log.

VCS panics nodes in the preferred site during a network-split

Recommended Action:

- Check the node weights using the command `<vxfenconfig -a>`.
  Nodes in the high preference site should have higher weight than nodes in the medium or low preference sites.
- If the node weights are not set, then check the cluster attributes UseFence and PreferredFencingPolicy.
- Check the engine log for errors.
- If the node weights are correct, then check the fencing logs for more details.

Configuring of stretch site fails

Recommended Action:

- For Campus Cluster, in a non-CVM configuration, check if DiskGroup is imported at least once on all nodes.
  Disk group should be listed in *server-> <hostname> -> Disk Groups*, and imported on one node and also should be in the deported state on other nodes of cluster.

- If Disk Group is not appearing in the *server-> <Hostname> -> Disk Groups,* then right click on *<Hostname>* and run **Rescan Disks**.
- For Campus cluster, tag all enclosures whose disks are part of Disk Groups used for campus cluster setup

Renaming a Site

Recommended Action: For renaming the site, re-run the Stretch Cluster Configuration flow by changing the Site names

Troubleshooting I/O fencing

The following sections discuss troubleshooting the I/O fencing problems. Review the symptoms and recommended solutions.

Node is unable to join cluster while another node is being ejected

A cluster that is currently fencing out (ejecting) a node from the cluster prevents a new node from joining the cluster until the fencing operation is completed. The following are example messages that appear on the console for the new node:

```
...VxFEN ERROR V-11-1-25 ... Unable to join running cluster
since cluster is currently fencing
a node out of the cluster.
```

If you see these messages when the new node is booting, the vxfen startup script on the node makes up to five attempts to join the cluster.

To manually join the node to the cluster when I/O fencing attempts fail

◆ If the vxfen script fails in the attempts to allow the node to join the cluster, restart vxfen driver with the command:

```
# systemctl stop vxfen
# systemctl start vxfen
```

If the command fails, restart the new node.

The vxfentsthdw utility fails when SCSI TEST UNIT READY command fails

While running the vxfentsthdw utility, you may see a message that resembles as follows:

```
Issuing SCSI TEST UNIT READY to disk reserved by other node
FAILED.
Contact the storage provider to have the hardware configuration
fixed.
```

The disk array does not support returning success for a `SCSI TEST UNIT READY` command when another host has the disk reserved using SCSI-3 persistent reservations. This happens with the Hitachi Data Systems 99XX arrays if bit 186 of the system mode option is not enabled.

Manually removing existing keys from SCSI-3 disks

Review the following procedure to remove specific registration and reservation keys created by another node from a disk.

See "About the vxfenadm utility" on page 295.

Note: If you want to clear all the pre-existing keys, use the vxfenclearpre utility.

See "About the vxfenclearpre utility" on page 300.

To remove the registration and reservation keys from disk

1. Create a file to contain the access names of the disks:

   ```
   # vi /tmp/disklist
   ```

   For example:

   ```
   /dev/sdu
   ```

2. Read the existing keys:

   ```
   # vxfenadm -s all -f /tmp/disklist
   ```

   The output from this command displays the key:

   ```
   Device Name: /dev/sdu

   Total Number Of Keys: 1
   key[0]:
   [Numeric Format]: 86,70,66,69,65,68,48,50
   [Character Format]: VFBEAD02
   [Node Format]: Cluster ID: 48813 Node ID: 2
     Node Name: unknown
   ```

3 If you know on which node the key (say A1) was created, log in to that node and enter the following command:

```
# vxfenadm -x -kA1 -f /tmp/disklist
```

The key A1 is removed.

4 If you do not know on which node the key was created, follow 5 through 7 to remove the key.

5 Register a second key "A2" temporarily with the disk:

```
# vxfenadm -m -k A2 -f /tmp/disklist
Registration completed for disk path /dev/sdu
```

6 Remove the first key from the disk by preempting it with the second key:

```
# vxfenadm -p -kA2 -f /tmp/disklist -vA1
key: A2------ prempted the key: A1------ on disk
/dev/sdu
```

7 Remove the temporary key registered in 5.

```
# vxfenadm -x -kA2 -f /tmp/disklist
Deleted the key : [A2------] from device /dev/sdu
```

No registration keys exist for the disk.

System panics to prevent potential data corruption

When a node experiences a split-brain condition and is ejected from the cluster, it panics and displays the following console message:

```
VXFEN:vxfen_plat_panic: Local cluster node ejected from cluster to
prevent potential data corruption.
```

A node experiences the split-brain condition when it loses the heartbeat with its peer nodes due to failure of all private interconnects or node hang. Review the behavior of I/O fencing under different scenarios and the corrective measures to be taken.

See "How I/O fencing works in different event scenarios" on page 272.

Cluster ID on the I/O fencing key of coordinator disk does not match the local cluster's ID

If you accidentally assign coordinator disks of a cluster to another cluster, then the fencing driver displays an error message similar to the following when you start I/O fencing:

```
000068 06:37:33 2bdd5845  0 ... 3066 0 VXFEN WARNING V-11-1-56
 Coordinator disk has key with cluster id 48813
 which does not match local cluster id 57069
```

The warning implies that the local cluster with the cluster ID 57069 has keys. However, the disk also has keys for cluster with ID 48813 which indicates that nodes from the cluster with cluster id 48813 potentially use the same coordinator disk.

You can run the following commands to verify whether these disks are used by another cluster. Run the following commands on one of the nodes in the local cluster. For example, on sys1:

```
sys1> # lltstat -C
57069

sys1> # cat /etc/vxfentab
/dev/vx/rdmp/disk_7
/dev/vx/rdmp/disk_8
/dev/vx/rdmp/disk_9

sys1> # vxfenadm -s /dev/vx/rdmp/disk_7
Reading SCSI Registration Keys...
Device Name: /dev/vx/rdmp/disk_7
Total Number Of Keys: 1
key[0]:
 [Numeric Format]:  86,70,48,49,52,66,48,48
 [Character Format]: VFBEAD00
 [Node Format]: Cluster ID: 48813 Node ID: 0 Node Name: unknown
```

Where *disk_7*, *disk_8*, and *disk_9* represent the disk names in your setup.

Recommended action: You must use a unique set of coordinator disks for each cluster. If the other cluster does not use these coordinator disks, then clear the keys using the vxfenclearpre command before you use them as coordinator disks in the local cluster.

See "About the vxfenclearpre utility" on page 300.

Fencing startup reports preexisting split-brain

The vxfen driver functions to prevent an ejected node from rejoining the cluster after the failure of the private network links and before the private network links are repaired.

For example, suppose the cluster of system 1 and system 2 is functioning normally when the private network links are broken. Also suppose system 1 is the ejected system. When system 1 restarts before the private network links are restored, its membership configuration does not show system 2; however, when it attempts to register with the coordinator disks, it discovers system 2 is registered with them. Given this conflicting information about system 2, system 1 does not join the cluster and returns an error from vxfenconfig that resembles:

```
vxfenconfig: ERROR: There exists the potential for a preexisting
split-brain. The coordinator disks list no nodes which are in
the current membership. However, they also list nodes which are
not in the current membership.

I/O Fencing Disabled!
```

Also, the following information is displayed on the console:

```
<date> <system name> vxfen: WARNING: Potentially a preexisting
<date> <system name> split-brain.
<date> <system name> Dropping out of cluster.
<date> <system name> Refer to user documentation for steps
<date> <system name> required to clear preexisting split-brain.
<date> <system name>
<date> <system name> I/O Fencing DISABLED!
<date> <system name>
<date> <system name> gab: GAB:20032: Port b closed
```

However, the same error can occur when the private network links are working and both systems go down, system 1 restarts, and system 2 fails to come back up. From the view of the cluster from system 1, system 2 may still have the registrations on the coordination points.

Assume the following situations to understand preexisting split-brain in server-based fencing:

- There are three CP servers acting as coordination points. One of the three CP servers then becomes inaccessible. While in this state, one client node leaves the cluster, whose registration cannot be removed from the inaccessible CP server. When the inaccessible CP server restarts, it has a stale registration from the node which left the VCS cluster. In this case, no new nodes can join the cluster. Each node that attempts to join the cluster gets a list of registrations

from the CP server. One CP server includes an extra registration (of the node which left earlier). This makes the joiner node conclude that there exists a preexisting split-brain between the joiner node and the node which is represented by the stale registration.

- All the client nodes have crashed simultaneously, due to which fencing keys are not cleared from the CP servers. Consequently, when the nodes restart, the vxfen configuration fails reporting preexisting split brain.

These situations are similar to that of preexisting split-brain with coordinator disks, where you can solve the problem running the `vxfenclearpre` command. A similar solution is required in server-based fencing using the `cpsadm` command.

See "Clearing preexisting split-brain condition" on page 656.

Clearing preexisting split-brain condition

Review the information on how the VxFEN driver checks for preexisting split-brain condition.

See "Fencing startup reports preexisting split-brain" on page 655.

See "About the vxfenclearpre utility" on page 300.

Table 22-3 describes how to resolve a preexisting split-brain condition depending on the scenario you have encountered:

Table 22-3 Recommened solution to clear pre-existing split-brain condition

Scenario	Solution
Actual potential split-brain condition—system 2 is up and system 1 is ejected	**1** Determine if system1 is up or not. **2** If system 1 is up and running, shut it down and repair the private network links to remove the split-brain condition. **3** Restart system 1.

Table 22-3 Recommened solution to clear pre-existing split-brain condition *(continued)*

Scenario	Solution
Apparent potential split-brain condition—system 2 is down and system 1 is ejected (Disk-based fencing is configured)	**1** Physically verify that system 2 is down. Verify the systems currently registered with the coordination points. Use the following command for coordinator disks: `# vxfenadm -s all -f /etc/vxfentab` The output of this command identifies the keys registered with the coordinator disks. **2** Clear the keys on the coordinator disks as well as the data disks in all shared disk groups using the `vxfenclearpre` command. The command removes SCSI-3 registrations and reservations. See "About the vxfenclearpre utility" on page 300. **3** Make any necessary repairs to system 2. **4** Restart system 2.
Apparent potential split-brain condition—system 2 is down and system 1 is ejected (Server-based fencing is configured)	**1** Physically verify that system 2 is down. Verify the systems currently registered with the coordination points. Use the following command for CP servers: `# cpsadm -s cp_server -a list_membership -c cluster_name` where *cp_server* is the virtual IP address or virtual hostname on which CP server is configured, and *cluster_name* is the VCS name for the VCS cluster (application cluster). The command lists the systems registered with the CP server. **2** Clear the keys on the coordinator disks as well as the data disks in all shared disk groups using the `vxfenclearpre` command. The command removes SCSI-3 registrations and reservations. After removing all stale registrations, the joiner node will be able to join the cluster. **3** Make any necessary repairs to system 2. **4** Restart system 2.

Registered keys are lost on the coordinator disks

If the coordinator disks lose the keys that are registered, the cluster might panic when a cluster reconfiguration occurs.

To refresh the missing keys

- Use the vxfenswap utility to replace the coordinator disks with the same disks. The vxfenswap utility registers the missing keys during the disk replacement.

  See "Refreshing lost keys on coordinator disks" on page 314.

Replacing defective disks when the cluster is offline

If the disk in the coordinator disk group becomes defective or inoperable and you want to switch to a new diskgroup in a cluster that is offline, then perform the following procedure.

In a cluster that is online, you can replace the disks using the vxfenswap utility.

See "About the vxfenswap utility" on page 303.

Review the following information to replace coordinator disk in the coordinator disk group, or to destroy a coordinator disk group.

Note the following about the procedure:

- When you add a disk, add the disk to the disk group vxfencoorddg and retest the group for support of SCSI-3 persistent reservations.
- You can destroy the coordinator disk group such that no registration keys remain on the disks. The disks can then be used elsewhere.

To replace a disk in the coordinator disk group when the cluster is offline

1. Log in as superuser on one of the cluster nodes.
2. If VCS is running, shut it down:

   ```
   # hastop -all
   ```

   Make sure that the port h is closed on all the nodes. Run the following command to verify that the port h is closed:

   ```
   # gabconfig -a
   ```

3. Stop I/O fencing on each node:

   ```
   # systemctl stop vxfen
   ```

   This removes any registration keys on the disks.

4 Import the coordinator disk group. The file /etc/vxfendg includes the name of the disk group (typically, vxfencoorddg) that contains the coordinator disks, so use the command:

```
# vxdg -tfC import `cat /etc/vxfendg`
```

where:

-t specifies that the disk group is imported only until the node restarts.

-f specifies that the import is to be done forcibly, which is necessary if one or more disks is not accessible.

-C specifies that any import locks are removed.

5 To remove disks from the disk group, use the VxVM disk administrator utility, vxdiskadm.

You may also destroy the existing coordinator disk group. For example:

- Verify whether the coordinator attribute is set to on.

```
# vxdg list vxfencoorddg |  grep flags: | grep coordinator
```

- Destroy the coordinator disk group.

```
# vxdg -o coordinator destroy vxfencoorddg
```

6 Add the new disk to the node and initialize it as a VxVM disk.

Then, add the new disk to the vxfencoorddg disk group:

- If you destroyed the disk group in step 5, then create the disk group again and add the new disk to it.
  See the *Cluster Server Installation Guide* for detailed instructions.
- If the diskgroup already exists, then add the new disk to it.

```
# vxdg -g vxfencoorddg -o coordinator adddisk disk_name
```

7 Test the recreated disk group for SCSI-3 persistent reservations compliance.

See "Testing the coordinator disk group using the -c option of vxfentsthdw" on page 289.

8 After replacing disks in a coordinator disk group, deport the disk group:

```
# vxdg deport `cat /etc/vxfendg`
```

9 On each node, start the I/O fencing driver:

```
# systemctl start vxfen
```

10 Verify that the I/O fencing module has started and is enabled.

```
# gabconfig -a
```

Make sure that port b membership exists in the output for all nodes in the cluster.

```
# vxfenadm -d
```

Make sure that I/O fencing mode is not disabled in the output.

11 If necessary, restart VCS on each node:

```
# hastart
```

The vxfenswap utility exits if rcp or scp commands are not functional

The vxfenswap utility displays an error message if rcp or scp commands are not functional.

To recover the vxfenswap utility fault

◆ Verify whether the rcp or scp functions properly.

Make sure that you do not use echo or cat to print messages in the .bashrc file for the nodes.

If the vxfenswap operation is unsuccessful, use the `vxfenswap -a cancel` command if required to roll back any changes that the utility made.

See "About the vxfenswap utility" on page 303.

Troubleshooting CP server

All CP server operations and messages are logged in the /var/VRTScps/log directory in a detailed and easy to read format. The entries are sorted by date and time. The logs can be used for troubleshooting purposes or to review for any possible security issue on the system that hosts the CP server.

The following files contain logs and text files that may be useful in understanding and troubleshooting a CP server:

- /var/VRTScps/log/cpserver_[ABC].log
- /var/VRTSvcs/log/vcsauthserver.log (Security related)

- If the vxcpserv process fails on the CP server, then review the following diagnostic files:
  - /var/VRTScps/diag/FFDC_CPS_*pid*_vxcpserv.log
  - /var/VRTScps/diag/stack_*pid*_vxcpserv.txt

Note: If the vxcpserv process fails on the CP server, these files are present in addition to a core file. VCS restarts vxcpserv process automatically in such situations.

The file /var/VRTSvcs/log/vxfen/vxfend_[ABC].log contains logs that may be useful in understanding and troubleshooting fencing-related issues on a VCS cluster (client cluster) node.

See "Troubleshooting issues related to the CP server service group" on page 661.

See "Checking the connectivity of CP server" on page 661.

See "Issues during fencing startup on VCS cluster nodes set up for server-based fencing" on page 662.

See "Issues during online migration of coordination points" on page 663.

Troubleshooting issues related to the CP server service group

If you cannot bring up the CPSSG service group after the CP server configuration, perform the following steps:

- Verify that the CPSSG service group and its resources are valid and properly configured in the VCS configuration.
- Check the VCS engine log (`/var/VRTSvcs/log/engine_[ABC].log`) to see if any of the CPSSG service group resources are FAULTED.
- Review the sample dependency graphs to make sure the required resources are configured correctly.
  See "About the CP server service group" on page 261.

Checking the connectivity of CP server

You can test the connectivity of CP server using the `cpsadm` command.

You must have set the environment variables CPS_USERNAME and CPS_DOMAINTYPE to run the `cpsadm` command on the VCS cluster (client cluster) nodes.

To check the connectivity of CP server

- Run the following command to check whether a CP server is up and running at a process level:

  ```
  # cpsadm -s cp_server -a ping_cps
  ```

  where *cp_server* is the virtual IP address or virtual hostname on which the CP server is listening.

Troubleshooting server-based fencing on the VCS cluster nodes

The file /var/VRTSvcs/log/vxfen/vxfend_[ABC].log contains logs files that may be useful in understanding and troubleshooting fencing-related issues on a VCS cluster (application cluster) node.

Issues during fencing startup on VCS cluster nodes set up for server-based fencing

Table 22-4 Fencing startup issues on VCS cluster (client cluster) nodes

Issue	Description and resolution
`cpsadm` command on the VCS cluster gives connection error	If you receive a connection error message after issuing the `cpsadm` command on the VCS cluster, perform the following actions: ■ Ensure that the CP server is reachable from all the VCS cluster nodes. ■ Check the /etc/vxfenmode file and ensure that the VCS cluster nodes use the correct CP server virtual IP or virtual hostname and the correct port number. ■ For HTTPS communication, ensure that the virtual IP and ports listed for the server can listen to HTTPS requests.
Authorization failure	Authorization failure occurs when the nodes on the client clusters and or users are not added in the CP server configuration. Therefore, fencing on the VCS cluster (client cluster) node is not allowed to access the CP server and register itself on the CP server. Fencing fails to come up if it fails to register with a majority of the coordination points. To resolve this issue, add the client cluster node and user in the CP server configuration and restart fencing.

Table 22-4 Fencing startup issues on VCS cluster (client cluster) nodes *(continued)*

Issue	Description and resolution
Authentication failure	If you had configured secure communication between the CP server and the VCS cluster (client cluster) nodes, authentication failure can occur due to the following causes: ■ The client cluster requires its own private key, a signed certificate, and a Certification Authority's (CA) certificate to establish secure communication with the CP server. If any of the files are missing or corrupt, communication fails. ■ If the client cluster certificate does not correspond to the client's private key, communication fails. ■ If the CP server and client cluster do not have a common CA in their certificate chain of trust, then communication fails. See "About secure communication between the VCS cluster and CP server" on page 263.

Issues during online migration of coordination points

During online migration of coordination points using the vxfenswap utility, the operation is automatically rolled back if a failure is encountered during validation of coordination points from any of the cluster nodes.

Validation failure of the new set of coordination points can occur in the following circumstances:

- The /etc/vxfenmode.test file is not updated on all the VCS cluster nodes, because new coordination points on the node were being picked up from an old /etc/vxfenmode.test file. The /etc/vxfenmode.test file must be updated with the current details. If the /etc/vxfenmode.test file is not present, vxfenswap copies configuration for new coordination points from the /etc/vxfenmode file.
- The coordination points listed in the /etc/vxfenmode file on the different VCS cluster nodes are not the same. If different coordination points are listed in the /etc/vxfenmode file on the cluster nodes, then the operation fails due to failure during the coordination point snapshot check.
- There is no network connectivity from one or more VCS cluster nodes to the CP server(s).
- Cluster, nodes, or users for the VCS cluster nodes have not been added on the new CP servers, thereby causing authorization failure.

Vxfen service group activity after issuing the vxfenswap command

The Coordination Point agent reads the details of coordination points from the `vxfenconfig -l` output and starts monitoring the registrations on them.

Thus, during vxfenswap, when the vxfenmode file is being changed by the user, the Coordination Point agent does not move to FAULTED state but continues monitoring the old set of coordination points.

As long as the changes to vxfenmode file are not committed or the new set of coordination points are not reflected in `vxfenconfig -l` output, the Coordination Point agent continues monitoring the old set of coordination points it read from `vxfenconfig -l` output in every monitor cycle.

The status of the Coordination Point agent (either ONLINE or FAULTED) depends upon the accessibility of the coordination points, the registrations on these coordination points, and the fault tolerance value.

When the changes to vxfenmode file are committed and reflected in the `vxfenconfig -l` output, then the Coordination Point agent reads the new set of coordination points and proceeds to monitor them in its new monitor cycle.

Troubleshooting notification

Occasionally you may encounter problems when using VCS notification. This section cites the most common problems and the recommended actions. Bold text provides a description of the problem.

Notifier is configured but traps are not seen on SNMP console.

Recommended Action: Verify the version of SNMP traps supported by the console: VCS notifier sends SNMP v2.0 traps. If you are using HP OpenView Network Node Manager as the SNMP, verify events for VCS are configured using xnmevents. You may also try restarting the OpenView daemon (ovw) if, after merging VCS events in vcs_trapd, the events are not listed in the OpenView Network Node Manager Event configuration.

By default, notifier assumes the community string is public. If your SNMP console was configured with a different community, reconfigure it according to the notifier configuration. See the *Cluster Server Bundled Agents Reference Guide* for more information on NotifierMngr.

Troubleshooting and recovery for global clusters

This topic describes the concept of disaster declaration and provides troubleshooting tips for configurations using global clusters.

Disaster declaration

When a cluster in a global cluster transitions to the FAULTED state because it can no longer be contacted, failover executions depend on whether the cause was due to a split-brain, temporary outage, or a permanent disaster at the remote cluster.

If you choose to take action on the failure of a cluster in a global cluster, VCS prompts you to declare the type of failure.

- *Disaster*, implying permanent loss of the primary data center
- *Outage*, implying the primary may return to its current form in some time
- *Disconnect*, implying a split-brain condition; both clusters are up, but the link between them is broken
- *Replica*, implying that data on the takeover target has been made consistent from a backup source and that the RVGPrimary can initiate a takeover when the service group is brought online. This option applies to VVR environments only.

You can select the groups to be failed over to the local cluster, in which case VCS brings the selected groups online on a node based on the group's FailOverPolicy attribute. It also marks the groups as being offline in the other cluster. If you do not select any service groups to fail over, VCS takes no action except implicitly marking the service groups as offline on the downed cluster.

Lost heartbeats and the inquiry mechanism

The loss of internal and all external heartbeats between any two clusters indicates that the remote cluster is faulted, or that all communication links between the two clusters are broken (a wide-area split-brain).

VCS queries clusters to confirm the remote cluster to which heartbeats have been lost is truly down. This mechanism is referred to as inquiry. If in a two-cluster configuration a connector loses all heartbeats to the other connector, it must consider the remote cluster faulted. If there are more than two clusters and a connector loses all heartbeats to a second cluster, it queries the remaining connectors before declaring the cluster faulted. If the other connectors view the cluster as running, the querying connector transitions the cluster to the UNKNOWN state, a process that minimizes false cluster faults. If all connectors report that the cluster is faulted, the querying connector also considers it faulted and transitions the remote cluster state to FAULTED.

VCS alerts

VCS alerts are identified by the alert ID, which is comprised of the following elements:

- `alert_type`—The type of the alert
- `cluster`—The cluster on which the alert was generated
- `system`—The system on which this alert was generated
- `object`—The name of the VCS object for which this alert was generated. This could be a cluster or a service group.

Alerts are generated in the following format:

```
alert_type-cluster-system-object
```

For example:

```
GNOFAILA-Cluster1-oracle_grp
```

This is an alert of type GNOFAILA generated on cluster Cluster1 for the service group oracle_grp.

Types of alerts

VCS generates the following types of alerts.

- CFAULT—Indicates that a cluster has faulted
- GNOFAILA—Indicates that a global group is unable to fail over within the cluster where it was online. This alert is displayed if the ClusterFailOverPolicy attribute is set to Manual and the wide-area connector (wac) is properly configured and running at the time of the fault.
- GNOFAIL—Indicates that a global group is unable to fail over to any system within the cluster or in a remote cluster.

  Some reasons why a global group may not be able to fail over to a remote cluster:

  - The ClusterFailOverPolicy is set to either Auto or Connected and VCS is unable to determine a valid remote cluster to which to automatically fail the group over.
  - The ClusterFailOverPolicy attribute is set to Connected and the cluster in which the group has faulted cannot communicate with one ore more remote clusters in the group's ClusterList.
  - The wide-area connector (wac) is not online or is incorrectly configured in the cluster in which the group has faulted

Managing alerts

Alerts require user intervention. You can respond to an alert in the following ways:

- If the reason for the alert can be ignored, use the Alerts dialog box in the Java console or the `haalert` command to delete the alert. You must provide a comment as to why you are deleting the alert; VCS logs the comment to engine log.
- Take an action on administrative alerts that have actions associated with them.
- VCS deletes or negates some alerts when a negating event for the alert occurs.

An administrative alert will continue to live if none of the above actions are performed and the VCS engine (HAD) is running on at least one node in the cluster. If HAD is not running on any node in the cluster, the administrative alert is lost.

Actions associated with alerts

This section describes the actions you can perform on the following types of alerts:

- CFAULT—When the alert is presented, clicking **Take Action** guides you through the process of failing over the global groups that were online in the cluster before the cluster faulted.
- GNOFAILA—When the alert is presented, clicking **Take Action** guides you through the process of failing over the global group to a remote cluster on which the group is configured to run.
- GNOFAIL—There are no associated actions provided by the consoles for this alert

Negating events

VCS deletes a CFAULT alert when the faulted cluster goes back to the running state

VCS deletes the GNOFAILA and GNOFAIL alerts in response to the following events:

- The faulted group's state changes from FAULTED to ONLINE.
- The group's fault is cleared.
- The group is deleted from the cluster where alert was generated.

Concurrency violation at startup

VCS may report a concurrency violation when you add a cluster to the ClusterList of the service group. A concurrency violation means that the service group is online on two nodes simultaneously.

Recommended Action: Verify the state of the service group in each cluster before making the service group global.

Troubleshooting the steward process

When you start the steward, it blocks the command prompt and prints messages to the standard output. To stop the steward, run the following command from a different command prompt of the same system:

If the steward is running in secure mode: `steward -stop - secure`

If the steward is not running in secure mode: `steward -stop`

In addition to the standard output, the steward can log to its own log files:

- steward_A.log
- steward-err_A.log

Use the tststew utility to verify that:

- The steward process is running
- The steward process is sending the right response

Use the following command to verify the state of the remote cluster:

```
# tststew -server <steward addr> [-secure] [-rclus <rclus name>]
-ping <rclus addr>
```

For example:

- Example 1 : Client IPv4 and Steward server IPv4 and secure cluster is configured

  ```
  # tststew -server 10.209.72.146 -secure -rclus testgcoclus2 -ping
  10.209.72.177

  Steward (10.209.72.146): testgcoclus2 is up
  ```

- Example 2 : Client IPv6 and Steward server IPv6 and secure cluster is configured

  ```
  # tststew -server 2620:128:f0a2:9005::107 -secure -rclus
  testgcoclus2 -ping 2620:128:f0a2:9005::120

  Steward (2620:128:f0a2:9005::107): testgcoclus2 is up
  ```

Troubleshooting licensing

This section cites problems you may encounter with VCS licensing. It provides instructions on how to validate license keys and lists the error messages associated with licensing.

Validating license keys

The installer script handles most license key validations. However, if you install a VCS key outside of the installer (using `vxlicinst`, for example), you can validate the key using the procedure described below.

1. The `vxlicinst` command handles some of the basic validations:

   node lock: Ensures that you are installing a node-locked key on the correct system

   demo hard end date: Ensures that you are not installing an expired demo key

2. Run the `vxlicrep` command to make sure a VCS key is installed on the system. The output of the command resembles:

```
License Key      = XXXX-XXXX-XXXX-XXXX-XXXX-XXXX
Product Name     = VERITAS Cluster Server
Serial number    = XXXX
License Type     = PERMANENT
OEM ID           = XXXX
Site License     = YES
Editions Product = YES
Features :
Platform:        = Unused
Version                   =  8.0
Tier                      = Tiern3
Reserved                  = 0
Mode                      = VCS
Global Cluster Option     = Enabled
CPU_TIER                  = 0
```

3. Look for the following in the command output:

   Make sure the Product Name lists the name of your purchased component, for example, Cluster Server. If the command output does not return the product name, you do not have a VCS key installed.

   If the output shows the License Type for a VCS key as DEMO, ensure that the Demo End Date does not display a past date.

   Make sure the *Mode* attribute displays the correct value.

   If you have purchased a license key for the Global Cluster Option, make sure its status is Enabled.

4. Start VCS. If HAD rejects a license key, see the licensing error message at the end of the engine_A log file.

Licensing error messages

This section lists the error messages associated with licensing. These messages are logged to the file /var/VRTSvcs/log/engine_A.log.

[Licensing] Insufficient memory to perform operation

The system does not have adequate resources to perform licensing operations.

[Licensing] No valid VCS license keys were found

No valid VCS keys were found on the system.

[Licensing] Unable to find a valid base VCS license key

No valid base VCS key was found on the system.

[Licensing] License key cannot be used on this OS platform

This message indicates that the license key was meant for a different platform. For example, a license key meant for Windows is used on Linux platform.

[Licensing] VCS evaluation period has expired

The VCS base demo key has expired

[Licensing] License key can not be used on this system

Indicates that you have installed a key that was meant for a different system (i.e. node-locked keys)

[Licensing] Unable to initialize the licensing framework

This is a VCS internal message. Call Veritas Technical Support.

[Licensing] QuickStart is not supported in this release

VCS QuickStart is not supported in this version of VCS.

[Licensing] Your evaluation period for the feature has expired. This feature will not be enabled the next time VCS starts

The evaluation period for the specified VCS feature has expired.

Troubleshooting secure configurations

This section includes error messages associated with configuring security and configuring FIPS mode. It also provides descriptions about the messages and the recommended action.

FIPS mode cannot be set

When you use the `vcsencrypt` command or `vcsdecrypt` command or use utilities that use these commands, the system sometimes displays the following error message:

```
VCS ERROR V-16-1-10351 Could not set FIPS mode
```

Recommended Action: Ensure that the random number generator is defined on your system for encryption to work correctly. Typically, the files required for random number generation are `/dev/random` and `/dev/urandom`.

Broker does not start

The AB configuration failure is undetected. If the AB setup fails, `vxatd` (default broker name) and the CPI scripts do not report the failure and its cause.

When the `vxatd -o -a` command is used to configure the authentication broker in AB-only mode, the return value is always 0 (success). As a result, CPI installers fail to detect any broker configuration failures.

Workaround:

- To find out if the command truly succeeded, run the 'vssat showcred' command, and manually verify that a credential is received from the root broker.
- Check for clock skew.

AT initialization fails

Check the installation for any missing or corrupted files. For embedded clients, check whether the EAT_HOME_DIR value is set.

Troubleshooting wizard-based configuration issues

This section lists common troubleshooting issues that you may encounter during or after the steps related to the VCS Cluster Configuration Wizard and the Veritas High Availability Configuration Wizard.

Running the 'hastop -all' command detaches virtual disks

The `hastop -all` command takes offline all the components and components groups of a configured application, and then stops the VCS cluster. In the process, the command detaches the virtual disks from the VCS cluster nodes. (2920101)

Workaround: If you want to stop the VCS cluster (and not the applications running on cluster nodes), instead of the “hastop –all”, use the following command:

```
hastop -all -force
```

This command stops the cluster without affecting the virtual disks attached to the VCS cluster nodes.

Troubleshooting issues with the Veritas High Availability view

This section lists common troubleshooting scenarios that you may encounter when using the Veritas High Availability view.

Veritas High Availability view does not display the application monitoring status

The Veritas High Availability view may either display a HTTP 404 Not Found error or may not show the application health status at all.

Verify the following conditions and then refresh the Veritas High Availability view:

- Verify that the Veritas High Availability Console host is running and is accessible over the network.
- Verify that the VMware Web Service is running on the vCenter Server.
- Verify that the VMware Tools Service is running on the guest virtual machine.
- Verify that the Veritas Storage Foundation Messaging Service (xprtld process) is running on the Veritas High Availability Console and the virtual machine. If it is stopped, type the following on the command prompt:

  ```
  net start xprtld
  ```

- Verify that ports 14152, 14153, and 5634 are not blocked by a firewall.
- Log out of the vSphere Client and then login again. Then, verify that the Veritas High Availability plugin is installed and enabled.

Veritas High Availability view may freeze due to special characters in application display name

For a monitored application, if you specify a display name that contains special characters, one or both of the following symptoms may occur:

- The Veritas high availability view may freeze
- The Veritas high availability view may display an Adobe exception error message

  Based on your browser settings, the Adobe exception message may or may not appear. However, in both cases the tab may freeze. (2923079)

  Workaround:

  Reset the display name using only those characters that belong to the following list:

  - any alphanumeric character
  - space
  - underscore

  Use the following command to reset the display name:

  hagrp -modify *sg name* UserAssoc -update Name *"modified display name without special characters"*

In the Veritas High Availability tab, the Add Failover System link is dimmed

If the system that you clicked in the inventory view of the vSphere Client GUI to launch the Veritas High Availability tab is not part of the list of failover target systems for that application, the Add Failover System link is dimmed. (2932281)

Workaround: In the vSphere Client GUI inventory view, to launch the Veritas High Availability tab, click a system from the existing list of failover target systems for the application. The Add Failover System link that appears in the drop-down list if you click **More**, is no longer dimmed.

Section 8

Appendixes

- Appendix A. VCS user privileges—administration matrices
- Appendix B. VCS commands: Quick reference
- Appendix C. Cluster and system states
- Appendix D. VCS attributes
- Appendix E. Accessibility and VCS

Appendix A

VCS user privileges—administration matrices

This appendix includes the following topics:

- About administration matrices
- Administration matrices

About administration matrices

In general, users with Guest privileges can run the following command options: -display, -state, and -value.

Users with privileges for Group Operator and Cluster Operator can execute the following options: -online, -offline, and -switch.

Users with Group Administrator and Cluster Administrator privileges can run the following options -add, -delete, and -modify.

See “About VCS user privileges and roles” on page 86.

Administration matrices

Review the matrices in the following topics to determine which command options can be executed within a specific user role. Checkmarks denote the command and option can be executed. A dash indicates they cannot.

Agent Operations (haagent)

Table A-1 lists agent operations and required privileges.

Table A-1 User privileges for agent operations

Agent Operation	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Start agent	–	–	–	✓	✓
Stop agent	–	–	–	✓	✓
Display info	✓	✓	✓	✓	✓
List agents	✓	✓	✓	✓	✓

Attribute Operations (haattr)

Table A-2 lists attribute operations and required privileges.

Table A-2 User privileges for attribute operations

Attribute Operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Add	–	–	–	–	✓
Change default value	–	–	–	–	✓
Delete	–	–	–	–	✓
Display	✓	✓	✓	✓	✓

Cluster Operations (haclus, haconf)

Table A-3 lists cluster operations and required privileges.

Table A-3 User privileges for cluster operations

Cluster Operations	Cluster Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Display	✓	✓	✓	✓	✓
Modify	–	–	–	–	✓
Add	–	–	–	–	✓

Table A-3 User privileges for cluster operations *(continued)*

Cluster Operations	Cluster Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Delete	–	–	–	–	✓
Declare	–	–	–	✓	✓
View state or status	✓	✓	✓	✓	✓
Update license	–	–	–	–	✓
Make configuration read-write	–	–	✓	–	✓
Save configuration	–	–	✓	–	✓
Make configuration read-only	–	–	✓	–	✓

Service group operations (hagrp)

Table A-4 lists service group operations and required privileges.

Table A-4 User privileges for service group operations

Service Group Operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Add and delete	–	–	–	–	✓
Link and unlink	–	–	–	–	✓
Clear	–	✓	✓	✓	✓
Bring online	–	✓	✓	✓	✓
Take offline	–	✓	✓	✓	✓
View state	✓	✓	✓	✓	✓
Switch	–	✓	✓	✓	✓
Freeze/unfreeze	–	✓	✓	✓	✓
Freeze/unfreeze persistent	–	–	✓	–	✓
Enable	–	–	✓	–	✓

Table A-4 User privileges for service group operations *(continued)*

Service Group Operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Disable	–	–	✓	–	✓
Modify	–	–	✓	–	✓
Display	✓	✓	✓	✓	✓
View dependencies	✓	✓	✓	✓	✓
View resources	✓	✓	✓	✓	✓
List	✓	✓	✓	✓	✓
Enable resources	–	–	✓	–	✓
Disable resources	–	–	✓	–	✓
Flush	–	✓	✓	✓	✓
Autoenable	–	✓	✓	✓	✓
Ignore	–	✓	✓	✓	✓

Heartbeat operations (hahb)

Table A-5 lists heartbeat operations and required privileges.

Table A-5 User privileges for heartbeat operations

Heartbeat Operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Add	–	–	–	–	✓
Delete	–	–	–	–	✓
Make local	–	–	–	–	✓
Make global	–	–	–	–	✓
Display	✓	✓	✓	✓	✓
View state	✓	✓	✓	✓	✓
List	✓	✓	✓	✓	✓

Log operations (halog)

Table A-6 lists log operations and required privileges.

Table A-6 User privileges for log operations

Log operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Enable debug tags	–	–	–	–	✓
Delete debug tags	–	–	–	–	✓
Add messages to log file	–	–	✓	–	✓
Display	✓	✓	✓	✓	✓
Display log file info	✓	✓	✓	✓	✓

Resource operations (hares)

Table A-7 lists resource operations and required privileges.

Table A-7 User privileges for resource operations

Resource operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Add	–	–	✓	–	✓
Delete	–	–	✓	–	✓
Make attribute local	–	–	✓	–	✓
Make attribute global	–	–	✓	–	✓
Link and unlink	–	–	✓	–	✓
Clear	–	✓	✓	✓	✓
Bring online	–	✓	✓	✓	✓
Take offline	–	✓	✓	✓	✓
Modify	–	–	✓	–	✓
View state	✓	✓	✓	✓	✓

Table A-7 User privileges for resource operations *(continued)*

Resource operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Display	✓	✓	✓	✓	✓
View dependencies	✓	✓	✓	✓	✓
List, Value	✓	✓	✓	✓	✓
Probe	–	✓	✓	✓	✓
Override attribute	–	–	✓	–	✓
Remove overrides	–	–	✓	–	✓
Run an action	–	✓	✓	✓	✓
Refresh info	–	✓	✓	✓	✓
Flush info	–	✓	✓	✓	✓

System operations (hasys)

Table A-8 lists system operations and required privileges.

Table A-8 User privileges for system operations

System operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Add	–	–	–	–	✓
Delete	–	–	–	–	✓
Freeze and unfreeze	–	–	–	✓	✓
Freeze and unfreeze persistent	–	–	–	–	✓
Freeze and evacuate	–	–	–	–	✓
Display	✓	✓	✓	✓	✓
Start forcibly	–	–	–	–	✓
Modify	–	–	–	–	✓

Table A-8 User privileges for system operations *(continued)*

System operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
View state	✓	✓	✓	✓	✓
List	✓	✓	✓	✓	✓
Update license	–	–	–	–	✓

Resource type operations (hatype)

Table A-9 lists resource type operations and required privileges.

Table A-9 User privileges for resource type operations

Resource type operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Add	–	–	–	–	✓
Delete	–	–	–	–	✓
Display	✓	✓	✓	✓	✓
View resources	✓	✓	✓	✓	✓
Modify	–	–	–	–	✓
List	✓	✓	✓	✓	✓

User operations (hauser)

Table A-10 lists user operations and required privileges.

Table A-10 User privileges for user operations

User operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Add	–	–	–	–	✓
Delete	–	–	–	–	✓

Table A-10 User privileges for user operations *(continued)*

User operations	Guest	Group Operator	Group Admin.	Cluster Operator	Cluster Admin.
Update	✓ **Note:** If configuration is read/write	✓ **Note:** If configuration is read/write	✓	✓ **Note:** If configuration is read/write	✓
Display	✓	✓	✓	✓	✓
List	✓	✓	✓	✓	✓
Modify privileges	–	–	✓	–	✓

Appendix B

VCS commands: Quick reference

This appendix includes the following topics:

- About this quick reference for VCS commands
- VCS command line reference

About this quick reference for VCS commands

This section lists commonly-used VCS commands and options. For more information about VCS commands and available options, see the man pages associated with VCS commands.

VCS command line reference

Table B-1 lists the VCS commands for cluster operations.

Table B-1 VCS commands for cluster operations

Cluster operations	Command line
Start VCS	`hastart [-onenode]` `hastart -force system_name`

Table B-1 VCS commands for cluster operations *(continued)*

Cluster operations	Command line

Stop VCS	`hastop -local [-force|-evacuate|-noautodisable]`
	`hastop -local [-force|-evacuate -noautodisable]`
	`hastop -sys system_name [-force|-evacuate|-noautodisable]`
	`hastop -sys system_name [-force|-evacuate -noautodisable]`
	`hastop -all [-force]`

Enable/disable read-write access to VCS configuration	`haconf -makerw` `haconf -dump -makero`
Add user	`hauser -add user_name`

Table B-2 lists the VCS commands for service group, resource, and site operations.

Table B-2 VCS commands for service group, resource, and site operations

Service group, resource, or site operations	Command line

Configure service groups	`hagrp -add |-delete group_name`
	`hagrp -modify group attribute value`
	`hagrp -link parent_group child_group dependency`

Configure resources	`hares -add resource type group_name`
	`hares -delete resource_name`
	`hares -modify resource attribute value`
	`hares -link parent_resource child_resource`

Configure agents or resource types	`hatype -modify type attribute value`

Table B-2 VCS commands for service group, resource, and site operations *(continued)*

Service group, resource, or site operations	Command line

Bring service groups online and take them offline	`hagrp -online service_group -sys system_name`
	`hagrp -online service_group -site site_name`
	`hagrp -offline service_group -sys system_name`
	`hagrp -offline service_group -site site_name`

Bring resources online and take them offline	`hares -online resource_name -sys system_name` `hares -offline resource_name -sys system_name`
Freezing/unfreezing service groups	`hagrp -freeze group_name [-persistent]` `hagrp -unfreeze group_name [-persistent]`

Table B-3 lists the VCS commands for status and verification.

Table B-3 VCS commands for status and verification

Status and verification	Command line
Cluster status	hastatus -summary
LLT status/verification	lltconfig lltconfig -a list lltstat lltstat -nvv
GAB status/verification	gabconfig -a gabconfig -av
AMF kernel driver status/verification	amfconfig amfstat

Table B-4 lists the VCS commands for cluster communication.

Table B-4 VCS commands for cluster communication

Communication	Command line
Starting and stopping LLT	`lltconfig -c` `lltconfig -U`
Starting and stopping GAB	`gabconfig -c -n seed_number` `gabconfig -U`

Table B-5 lists the VCS commands for system operations.

Table B-5 VCS commands for system operations

System operations	Command line
List systems in a cluster	hasys -list
Retrieve detailed information about each system	hasys -display system_name
Freezing/unfreezing systems	`hasys -freeze [-persistent][-evacuate] system_name` `hasys -unfreeze [-persistent] system_name`

Appendix C

Cluster and system states

This appendix includes the following topics:

- Remote cluster states
- System states

Remote cluster states

In global clusters, the "health" of the remote clusters is monitored and maintained by the wide-area connector process. The connector process uses heartbeats, such as Icmp, to monitor the state of remote clusters. The state is then communicated to HAD, which then uses the information to take appropriate action when required. For example, when a cluster is shut down gracefully, the connector transitions its local cluster state to EXITING and notifies the remote clusters of the new state. When the cluster exits and the remote connectors lose their TCP/IP connection to it, each remote connector transitions their view of the cluster to EXITED.

To enable wide-area network heartbeats, the wide-area connector process must be up and running. For wide-area connectors to connect to remote clusters, at least one heartbeat to the specified cluster must report the state as ALIVE.

There are three hearbeat states for remote clusters: HBUNKNOWN, HBALIVE, and HBDEAD.

See "Examples of system state transitions" on page 692.

Table C-1 provides a list of VCS remote cluster states and their descriptions.

Table C-1 VCS state definitions

State	Definition
INIT	The initial state of the cluster. This is the default state.

Table C-1 VCS state definitions *(continued)*

State	Definition
BUILD	The local cluster is receiving the initial snapshot from the remote cluster.
RUNNING	Indicates the remote cluster is running and connected to the local cluster.
LOST_HB	The connector process on the local cluster is not receiving heartbeats from the remote cluster
LOST_CONN	The connector process on the local cluster has lost the TCP/IP connection to the remote cluster.
UNKNOWN	The connector process on the local cluster determines the remote cluster is down, but another remote cluster sends a response indicating otherwise.
FAULTED	The remote cluster is down.
EXITING	The remote cluster is exiting gracefully.
EXITED	The remote cluster exited gracefully.
INQUIRY	The connector process on the local cluster is querying other clusters on which heartbeats were lost.
TRANSITIONING	The connector process on the remote cluster is failing over to another node in the cluster.

Examples of cluster state transitions

Following are examples of cluster state transitions:

- If a remote cluster joins the global cluster configuration, the other clusters in the configuration transition their "view" of the remote cluster to the RUNNING state:
  INIT -> BUILD -> RUNNING
- If a cluster loses all heartbeats to a remote cluster in the RUNNING state, inquiries are sent. If all inquiry responses indicate the remote cluster is actually down, the cluster transitions the remote cluster state to FAULTED:
  RUNNING -> LOST_HB -> INQUIRY -> FAULTED
- If at least one response does not indicate the cluster is down, the cluster transitions the remote cluster state to UNKNOWN:
  RUNNING -> LOST_HB -> INQUIRY -> UNKNOWN
- When the ClusterService service group, which maintains the connector process as highly available, fails over to another system in the cluster, the remote clusters

transition their view of that cluster to TRANSITIONING, then back to RUNNING after the failover is successful:
RUNNING -> TRANSITIONING -> BUILD -> RUNNING

- When a remote cluster in a RUNNING state is stopped (by taking the ClusterService service group offline), the remote cluster transitions to EXITED:
RUNNING -> EXITING -> EXITED

System states

Whenever the VCS engine is running on a system, it is in one of the states described in the table below. States indicate a system's current mode of operation. When the engine is started on a new system, it identifies the other systems available in the cluster and their states of operation. If a cluster system is in the state of RUNNING, the new system retrieves the configuration information from that system. Changes made to the configuration while it is being retrieved are applied to the new system before it enters the RUNNING state.

If no other systems are up and in the state of RUNNING or ADMIN_WAIT, and the new system has a configuration that is not invalid, the engine transitions to the state LOCAL_BUILD, and builds the configuration from disk. If the configuration is invalid, the system transitions to the state of STALE_ADMIN_WAIT.

See "Examples of system state transitions" on page 692.

Table C-2 provides a list of VCS system states and their descriptions.

Table C-2 VCS system states

State	Definition
ADMIN_WAIT	The running configuration was lost. A system transitions into this state for the following reasons: ■ The last system in the RUNNING configuration leaves the cluster before another system takes a snapshot of its configuration and transitions to the RUNNING state. ■ A system in LOCAL_BUILD state tries to build the configuration from disk and receives an unexpected error from hacf indicating the configuration is invalid.
CURRENT_ DISCOVER_WAIT	The system has joined the cluster and its configuration file is valid. The system is waiting for information from other systems before it determines how to transition to another state.

Table C-2 VCS system states *(continued)*

State	Definition
CURRENT_PEER_WAIT	The system has a valid configuration file and another system is doing a build from disk (LOCAL_BUILD). When its peer finishes the build, this system transitions to the state REMOTE_BUILD.
EXITING	The system is leaving the cluster.
EXITED	The system has left the cluster.
EXITING_FORCIBLY	An `hastop -force` command has forced the system to leave the cluster.
FAULTED	The system has left the cluster unexpectedly.
INITING	The system has joined the cluster. This is the initial state for all systems.
LEAVING	The system is leaving the cluster gracefully. When the agents have been stopped, and when the current configuration is written to disk, the system transitions to EXITING.
LOCAL_BUILD	The system is building the running configuration from the disk configuration.
REMOTE_BUILD	The system is building a running configuration that it obtained from a peer in a RUNNING state.
RUNNING	The system is an active member of the cluster.
STALE_ADMIN_WAIT	The system has an invalid configuration and there is no other system in the state of RUNNING from which to retrieve a configuration. If a system with a valid configuration is started, that system enters the LOCAL_BUILD state. Systems in STALE_ADMIN_WAIT transition to STALE_PEER_WAIT.
STALE_DISCOVER_WAIT	The system has joined the cluster with an invalid configuration file. It is waiting for information from any of its peers before determining how to transition to another state.
STALE_PEER_WAIT	The system has an invalid configuration file and another system is doing a build from disk (LOCAL_BUILD). When its peer finishes the build, this system transitions to the state REMOTE_BUILD.
UNKNOWN	The system has not joined the cluster because it does not have a system entry in the configuration.

Examples of system state transitions

Following are examples of system state transitions:

- If VCS is started on a system, and if that system is the only one in the cluster with a valid configuration, the system transitions to the RUNNING state:
  INITING -> CURRENT_DISCOVER_WAIT -> LOCAL_BUILD -> RUNNING
- If VCS is started on a system with a valid configuration file, and if at least one other system is already in the RUNNING state, the new system transitions to the RUNNING state:
  INITING -> CURRENT_DISCOVER_WAIT -> REMOTE_BUILD -> RUNNING
- If VCS is started on a system with an invalid configuration file, and if at least one other system is already in the RUNNING state, the new system transitions to the RUNNING state:
  INITING -> STALE_DISCOVER_WAIT -> REMOTE_BUILD -> RUNNING
- If VCS is started on a system with an invalid configuration file, and if all other systems are in STALE_ADMIN_WAIT state, the system transitions to the STALE_ADMIN_WAIT state as shown below. A system stays in this state until another system with a valid configuration file is started.
  INITING -> STALE_DISCOVER_WAIT -> STALE_ADMIN_WAIT
- If VCS is started on a system with a valid configuration file, and if other systems are in the ADMIN_WAIT state, the new system transitions to the ADMIN_WAIT state.
  INITING -> CURRENT_DISCOVER_WAIT -> ADMIN_WAIT
- If VCS is started on a system with an invalid configuration file, and if other systems are in the ADMIN_WAIT state, the new system transitions to the ADMIN_WAIT state.
  INITING -> STALE_DISCOVER_WAIT -> ADMIN_WAIT
- When a system in RUNNING state is stopped with the `hastop` command, it transitions to the EXITED state as shown below. During the LEAVING state, any online system resources are taken offline. When all of the system's resources are taken offline and the agents are stopped, the system transitions to the EXITING state, then EXITED.
  RUNNING -> LEAVING -> EXITING -> EXITED

Appendix D

VCS attributes

This appendix includes the following topics:

- About attributes and their definitions
- Resource attributes
- Resource type attributes
- Service group attributes
- System attributes
- Cluster attributes
- Heartbeat attributes (for global clusters)
- Remote cluster attributes
- Site attributes

About attributes and their definitions

In addition to the attributes listed in this appendix, see the *Cluster Server Agent Developer's Guide*.

You can modify the values of attributes labelled user-defined from the command line or graphical user interface, or by manually modifying the main.cf configuration file. You can change the default values to better suit your environment and enhance performance.

When changing the values of attributes, be aware that VCS attributes interact with each other. After changing the value of an attribute, observe the cluster systems to confirm that unexpected behavior does not impair performance.

The values of attributes labelled system use only are set by VCS and are read-only. They contain important information about the state of the cluster.

The values labeled agent-defined are set by the corresponding agent and are also read-only.

Attribute values are case-sensitive.

See "About VCS attributes" on page 70.

Resource attributes

Table D-1 lists resource attributes.

Table D-1 Resource attributes

Resource attributes	Description
ArgListValues (agent-defined)	List of arguments passed to the resource's agent on each system. This attribute is resource-specific and system-specific, meaning that the list of values passed to the agent depend on which system and resource they are intended. The number of values in the ArgListValues should not exceed 425. This requirement becomes a consideration if an attribute in the ArgList is a keylist, a vector, or an association. Such type of non-scalar attributes can typically take any number of values, and when they appear in the ArgList, the agent has to compute ArgListValues from the value of such attributes. If the non-scalar attribute contains many values, it will increase the size of ArgListValues. Hence when developing an agent, this consideration should be kept in mind when adding a non-scalar attribute in the ArgList. Users of the agent need to be notified that the attribute should not be configured to be so large that it pushes that number of values in the ArgListValues attribute to be more than 425. ■ Type and dimension: string-vector ■ Default: non-applicable.

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
AutoStart (user-defined)	Indicates if a resource should be brought online as part of a service group online, or if it needs the `hares -online` command. For example, you have two resources, R1 and R2. R1 and R2 are in group G1. R1 has an AutoStart value of 0, R2 has an AutoStart value of 1. In this case, you see the following effects: # hagrp -online G1 -sys sys1 Brings only R2 to an ONLINE state. The group state is ONLINE and not a PARTIAL state. R1 remains OFFLINE. # hares -online R1 -sys sys1 Brings R1 online, the group state is ONLINE. # hares -offline R2 -sys sys1 Brings R2 offline, the group state is PARTIAL. Resources with a value of zero for AutoStart, contribute to the group's state only in their ONLINE state and not for their OFFLINE state. ■ Type and dimension: boolean-scalar ■ Default: 1
ComputeStats (user-defined)	Indicates to agent framework whether or not to calculate the resource’s monitor statistics. ■ Type and dimension: boolean-scalar ■ Default: 0
ConfidenceLevel (agent-defined)	Indicates the level of confidence in an online resource. Values range from 0–100. Note that some VCS agents may not take advantage of this attribute and may always set it to 0. Set the level to 100 if the attribute is not used. ■ Type and dimension: integer-scalar ■ Default: 0
Critical (user-defined)	Indicates whether a fault of this resource should trigger a failover of the entire group or not. If Critical is 0 and no parent above has Critical = 1, then the resource fault will not cause group failover. ■ Type and dimension: boolean-scalar ■ Default: 1

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
Enabled (user-defined)	Indicates agents monitor the resource. If a resource is created dynamically while VCS is running, you must enable the resource before VCS monitors it. For more information on how to add or enable resources, see the chapters on administering VCS from the command line and graphical user interfaces. When Enabled is set to 0, it implies a disabled resource. See "Troubleshooting VCS startup" on page 632. ■ Type and dimension: boolean-scalar ■ Default: If you specify the resource in main.cf prior to starting VCS, the default value for this attribute is 1, otherwise it is 0.
Flags (system use only)	Provides additional information for the state of a resource. Primarily this attribute raises flags pertaining to the resource. Values: ADMIN WAIT—The running configuration of a system is lost. RESTARTING —The agent is attempting to restart the resource because the resource was detected as offline in latest monitor cycle unexpectedly. See RestartLimit attribute for more information. STATE UNKNOWN—The latest monitor call by the agent could not determine if the resource was online or offline. MONITOR TIMEDOUT —The latest monitor call by the agent was terminated because it exceeded the maximum time specified by the static attribute MonitorTimeout. UNABLE TO OFFLINE—The agent attempted to offline the resource but the resource did not go offline. This flag is also set when a resource faults and the clean function completes successfully, but the subsequent monitor hangs or is unable to determine resource status. ■ Type and dimension: integer-scalar ■ Default: Not applicable.
Group (system use only)	String name of the service group to which the resource belongs. ■ Type and dimension: string-scalar ■ Default: Not applicable.

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
IState (system use only)	The internal state of a resource. In addition to the State attribute, this attribute shows to which state the resource is transitioning. Values: NOT WAITING—Resource is not in transition. WAITING TO GO ONLINE—Agent notified to bring the resource online but procedure not yet complete. WAITING FOR CHILDREN ONLINE—Resource to be brought online, but resource depends on at least one offline resource. Resource transitions to waiting to go online when all children are online. WAITING TO GO OFFLINE—Agent notified to take the resource offline but procedure not yet complete. WAITING TO GO OFFLINE (propagate)—Same as above, but when completed the resource's children will also be offline. WAITING TO GO ONLINE (reverse)—Resource waiting to be brought online, but when it is online it attempts to go offline. Typically this is the result of issuing an offline command while resource was waiting to go online. WAITING TO GO OFFLINE (path) - Agent notified to take the resource offline but procedure not yet complete. When the procedure completes, the resource's children which are a member of the path in the dependency tree will also be offline. WAITING TO GO OFFLINE (reverse) - Resource waiting to be brought offline, but when it is offline it attempts to go online. Typically this is the result of issuing an online command while resource was waiting to go offline. WAITING TO GO ONLINE (reverse/path) - Resource waiting to be brought online, but when online it is brought offline. Resource transitions to WAITING TO GO OFFLINE (path). Typically this is the result of fault of a child resource while resource was waiting to go online. WAITING FOR PARENT OFFLINE – Resource waiting for parent resource to go offline. When parent is offline the resource is brought offline. **Note:** Although this attribute accepts integer types, the command line indicates the text representations.

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
IState (system use only)	WAITING TO GO ONLINE (reverse/propagate)—Same as above, but resource propagates the offline operation. IStates on the source system for migration operations: ■ WAITING FOR OFFLINE VALIDATION (migrate) – This state is applicable for resource on source system and indicates that migration operation has been accepted and VCS is validating whether migration is possible. ■ WAITING FOR MIGRATION OFFLINE – This state is applicable for resource on source system and indicates that migration operation has passed the prerequisite checks and validations on the source system. ■ WAITING TO COMPLETE MIGRATION – This state is applicable for resource on source system and indicates that migration process is complete on the source system and the VCS engine is waiting for the resource to come online on target system. IStates on the target system for migration operations: ■ WAITING FOR ONLINE VALIDATION (migrate) – This state is applicable for resource on target system and indicates that migration operations are accepted and VCS is validating whether migration is possible. ■ WAITING FOR MIGRATION ONLINE – This state is applicable for resource on target system and indicates that migration operation has passed the prerequisite checks and validations on the source system. ■ WAITING TO COMPLETE MIGRATION (online) – This state is applicable for resource on target system and indicates that migration process is complete on the source system and the VCS engine is waiting for the resource to come online on target system. ■ Type and dimension: integer-scalar ■ Default: 1 NOT WAITING
LastOnline (system use only)	Indicates the system name on which the resource was last online. This attribute is set by VCS. ■ Type and dimension: string-scalar ■ Default: Not applicable

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
ManageFaults (user-defined)	Specifies whether VCS responds to a resource fault by calling the Clean entry point. Its value supersedes all the values assigned to the attribute at service group level. This attribute can take the following values: ■ ACT: VCS invokes the Clean function with `CleanReason` set to Online Hung. ■ IGNORE: VCS changes the resource state to ONLINE|ADMIN_WAIT. ■ NULL (Blank): VCS takes action based on the values set for the attribute at the service group level. Default value: “”
MonitorMethod (system use only)	Specifies the monitoring method that the agent uses to monitor the resource: ■ Traditional—Poll-based resource monitoring ■ IMF—Intelligent resource monitoring See “About resource monitoring” on page 38. Type and dimension: string-scalar Default: Traditional
MonitorOnly (system use only)	Indicates if the resource can be brought online or taken offline. If set to 0, resource can be brought online or taken offline. If set to 1, resource can only be monitored. **Note:** This attribute can only be affected by the command `hagrp -freeze`. ■ Type and dimension: boolean-scalar ■ Default: 0
MonitorTimeStats (system use only)	Valid keys are Average and TS. Average is the average time taken by the monitor function over the last Frequency number of monitor cycles. TS is the timestamp indicating when the engine updated the resource’s Average value. ■ Type and dimension: string-association ■ Default: Average = 0 TS = ""
Name (system use only)	Contains the actual name of the resource. ■ Type and dimension: string-scalar ■ Default: Not applicable.

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
Path (system use only)	Set to 1 to identify a resource as a member of a path in the dependency tree to be taken offline on a specific system after a resource faults. ■ Type and dimension: boolean-scalar ■ Default: 0
Probed (system use only)	Indicates whether the state of the resource has been determined by the agent by running the monitor function. ■ Type and dimension: boolean-scalar ■ Default: 0
ResourceInfo (system use only)	This attribute has three predefined keys: State: values are Valid, Invalid, or Stale. Msg: output of the info agent function of the resource on stdout by the agent framework. TS: timestamp indicating when the ResourceInfo attribute was updated by the agent framework ■ Type and dimension: string-association ■ Default: State = Valid Msg = "" TS = ""
ResourceOwner (user-defined)	This attribute is used for VCS email notification and logging. VCS sends email notification to the person that is designated in this attribute when events occur that are related to the resource. Note that while VCS logs most events, not all events trigger notifications. VCS also logs the owner name when certain events occur. Make sure to set the severity level at which you want notifications to be sent to ResourceOwner or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-scalar ■ Default: "" ■ Example: "jdoe@example.com"
ResourceRecipients (user-defined)	This attribute is used for VCS email notification. VCS sends email notification to persons designated in this attribute when events related to the resource occur and when the event's severity level is equal to or greater than the level specified in the attribute. Make sure to set the severity level at which you want notifications to be sent to ResourceRecipients or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-association ■ email id: The e-mail address of the person registered as a recipient for notification. severity: The minimum level of severity at which notifications must be sent.

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
Signaled (system use only)	Indicates whether a resource has been traversed. Used when bringing a service group online or taking it offline. ■ Type and dimension: integer-association ■ Default: Not applicable.
Start (system use only)	Indicates whether a resource was started (the process of bringing it online was initiated) on a system. ■ Type and dimension: integer -scalar ■ Default: 0
State (system use only)	Resource state displays the state of the resource and the flags associated with the resource. (Flags are also captured by the Flags attribute.) This attribute and Flags present a comprehensive view of the resource's current state. Values: ONLINE OFFLINE FAULTED OFFLINE|MONITOR TIMEDOUT OFFLINE|STATE UNKNOWN OFFLINE|ADMIN WAIT ONLINE|RESTARTING ONLINE|MONITOR TIMEDOUT ONLINE|STATE UNKNOWN ONLINE|UNABLE TO OFFLINE ONLINE|ADMIN WAIT FAULTED|MONITOR TIMEDOUT FAULTED|STATE UNKNOWN A FAULTED resource is physically offline, though unintentionally. **Note:** Although this attribute accepts integer types, the command line indicates the text representations. Type and dimension: integer -scalar Default: 0

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
TriggerEvent (user-defined)	A flag that turns Events on or off. ■ Type and dimension: boolean-scalar ■ Default: 0
TriggerPath (user-defined)	Enables you to customize the trigger path. ■ Type and dimension: string-scalar ■ Default: "" If a trigger is enabled but the trigger path at the service group level and at the resource level is "" (default), VCS invokes the trigger from the $VCS_HOME/bin/*triggers* directory. The TriggerPath value is case-sensitive. VCS does not trim the leading spaces or trailing spaces in the Trigger Path value. If the path contains leading spaces or trailing spaces, the trigger might fail to get executed. The path that you specify is relative to $VCS_HOME and the trigger path defined for the service group. Specify the path in the following format: *ServiceGroupTriggerPath*/*Resource*/*Trigger* If TriggerPath for service group sg1 is mytriggers/sg1 and TriggerPath for resource res1 is "", you must store the trigger script in the $VCS_HOME/mytriggers/sg1/res1 directory. For example, store the resstatechange trigger script in the $VCS_HOME/mytriggers/sg1/res1 directory. Yon can manage triggers for all resources for a service group more easily. If TriggerPath for resource res1 is mytriggers/sg1/vip1 in the preceding example, you must store the trigger script in the $VCS_HOME/mytriggers/sg1/vip1 directory. For example, store the resstatechange trigger script in the $VCS_HOME/mytriggers/sg1/vip1 directory. Modification of TriggerPath value at the resource level does not change the TriggerPath value at the service group level. Likewise, modification of TriggerPath value at the service group level does not change the TriggerPath value at the resource level.
TriggerResRestart (user-defined)	Determines whether or not to invoke the resrestart trigger if resource restarts. See "About the resrestart event trigger" on page 455. If this attribute is enabled at the group level, the resrestart trigger is invoked irrespective of the value of this attribute at the resource level. See "Service group attributes" on page 719. ■ Type and dimension: boolean-scalar ■ Default: 0 (disabled)

Table D-1 Resource attributes *(continued)*

Resource attributes	Description
TriggerResState Change (user-defined)	Determines whether or not to invoke the resstatechange trigger if the resource changes state. See "About the resstatechange event trigger" on page 456. If this attribute is enabled at the group level, then the resstatechange trigger is invoked irrespective of the value of this attribute at the resource level. See "Service group attributes" on page 719. ■ Type and dimension: boolean-scalar ■ Default: 0 (disabled)
TriggersEnabled (user-defined)	Determines if a specific trigger is enabled or not. Triggers are disabled by default. You can enable specific triggers on all nodes or only on selected nodes. Valid values are RESFAULT, RESNOTOFF, RESSTATECHANGE, RESRESTART, and RESADMINWAIT. To enable triggers on a specific node, add trigger keys in the following format: TriggersEnabled@node1 = {RESADMINWAIT, RESNOTOFF} The resadminwait trigger and resnotoff trigger are enabled on node1. To enable triggers on all nodes in the cluster, add trigger keys in the following format: TriggersEnabled = {RESADMINWAIT, RESNOTOFF} The resadminwait trigger and resnotoff trigger are enabled on all nodes. ■ Type and dimension: string-keylist ■ Default: {}

Resource type attributes

You can override some static attributes for resource types.

For more information on any attribute listed below, see the chapter on setting agent parameters in the *Cluster Server Agent Developer's Guide*.

Table D-2 lists the resource type attributes.

Table D-2 Resource type attributes

Resource type attributes	Description
ActionTimeout (user-defined)	Timeout value for the Action function. ■ Type and dimension: integer-scalar ■ Default: 30 seconds
AdvDbg (user-defined)	Enables activation of advanced debugging: ■ Type and dimension: string-keylist ■ Default: Not applicable For information about the AdvDbg attribute, see the *Cluster Server Agent Developer's Guide*.
AgentClass (user-defined)	Indicates the scheduling class for the VCS agent process. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority ■ Type and dimension: string-scalar ■ Default: TS
AgentDirectory (user-defined)	Complete path of the directory in which the agent binary and scripts are located. Agents look for binaries and scripts in the following directories: ■ Directory specified by the AgentDirectory attribute ■ /opt/VRTSvcs/bin/type/ ■ /opt/VRTSagents/ha/bin/type/ If none of the above directories exist, the agent does not start. Use this attribute in conjunction with the AgentFile attribute to specify a different location or different binary for the agent. ■ Type and dimension: string-scalar ■ Default = ""
AgentFailedOn (system use only)	A list of systems on which the agent for the resource type has failed. ■ Type and dimension: string-keylist ■ Default: Not applicable.

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
AgentFile (user-defined)	Complete name and path of the binary for an agent. If you do not specify a value for this attribute, VCS uses the agent binary at the path defined by the AgentDirectory attribute. ■ Type and dimension: string-scalar ■ Default = ""
AgentPriority (user-defined)	Indicates the priority in which the agent process runs. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority Type and dimension: string-scalar Default: 0
AgentReplyTimeout (user-defined)	The number of seconds the engine waits to receive a heartbeat from the agent before restarting the agent. ■ Type and dimension: integer-scalar ■ Default: 130 seconds
AgentStartTimeout (user-defined)	The number of seconds after starting the agent that the engine waits for the initial agent "handshake" before restarting the agent. ■ Type and dimension: integer-scalar ■ Default: 60 seconds

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
AlertOnMonitorTimeouts (user-defined) **Note:** This attribute can be overridden.	When a monitor times out as many times as the value or a multiple of the value specified by this attribute, then VCS sends an SNMP notification to the user. If this attribute is set to a value, say N, then after sending the notification at the first monitor timeout, VCS also sends an SNMP notification at each N-consecutive monitor timeout including the first monitor timeout for the second-time notification. When AlertOnMonitorTimeouts is set to 0, VCS will send an SNMP notification to the user only for the first monitor timeout; VCS will not send further notifications to the user for subsequent monitor timeouts until the monitor returns a success. The AlertOnMonitorTimeouts attribute can be used in conjunction with the FaultOnMonitorTimeouts attribute to control the behavior of resources of a group configured under VCS in case of monitor timeouts. When FaultOnMonitorTimeouts is set to 0 and AlertOnMonitorTimeouts is set to some value for all resources of a service group, then VCS will not perform any action on monitor timeouts for resources configured under that service group, but will only send notifications at the frequency set in the AlertOnMonitorTimeouts attribute. ■ Type and dimension: integer-scalar ■ Default: 0
ArgList (user-defined)	An ordered list of attributes whose values are passed to the open, close, online, offline, monitor, clean, info, and action functions. ■ Type and dimension: string-vector ■ Default: Not applicable.
AttrChangedTimeout (user-defined) **Note:** This attribute can be overridden.	Maximum time (in seconds) within which the attr_changed function must complete or be terminated. ■ Type and dimension: integer-scalar ■ Default: 60 seconds
CleanRetryLimit (user-defined)	Number of times to retry the clean function before moving a resource to ADMIN_WAIT state. If set to 0, clean is re-tried indefinitely. The valid values of this attribute are in the range of 0-1024. ■ Type and dimension: integer-scalar ■ Default: 0
CleanTimeout (user-defined) **Note:** This attribute can be overridden.	Maximum time (in seconds) within which the clean function must complete or else be terminated. ■ Type and dimension: integer-scalar ■ Default: 60 seconds

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
CloseTimeout (user-defined) **Note:** This attribute can be overridden.	Maximum time (in seconds) within which the close function must complete or else be terminated. ■ Type and dimension: integer-scalar ■ Default: 60 seconds
ConfInterval (user-defined) **Note:** This attribute can be overridden.	When a resource has remained online for the specified time (in seconds), previous faults and restart attempts are ignored by the agent. (See ToleranceLimit and RestartLimit attributes for details.) ■ Type and dimension: integer-scalar ■ Default: 600 seconds
ContainerOpts (system use only)	Specifies information that passes to the agent that controls the resources. These values are only effective when you set the ContainerInfo service group attribute. ■ RunInContainer When the value of the RunInContainer key is 1, the agent function (entry point) for that resource runs inside of the local container. When the value of the RunInContainer key is 0, the agent function (entry point) for that resource runs outside the local container (in the global environment). ■ PassCInfo When the value of the PassCInfo key is 1, the agent function receives the container information that is defined in the service group's ContainerInfo attribute. An example use of this value is to pass the name of the container to the agent. When the value of the PassCInfo key is 0, the agent function does not receive the container information that is defined in the service group's ContainerInfo attribute.
EPClass (user-defined)	Enables you to control the scheduling class for the agent functions (entry points) other than the online entry point whether the entry point is in C or scripts. The following values are valid for this attribute: ■ RT (Real Time) ■ TS (Time Sharing) ■ -1—indicates that VCS does not use this attribute to control the scheduling class of entry points. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority ■ Type and dimension: string-scalar ■ Default: -1

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
EPPriority (user-defined)	Enables you to control the scheduling priority for the agent functions (entry points) other than the online entry point. The attribute controls the agent function priority whether the entry point is in C or scripts. The following values are valid for this attribute: ■ 0—indicates the default priority value for the configured scheduling class as given by the EPClass attribute for the operating system. ■ Greater than 0—indicates a value greater than the default priority for the operating system. Veritas recommends a value of greater than 0 for this attribute. A system that has a higher load requires a greater value. ■ -1—indicates that VCS does not use this attribute to control the scheduling priority of entry points. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority ■ Type and dimension: string-scalar ■ Default: -1
ExternalStateChange (user-defined) **Note:** This attribute can be overridden.	Defines how VCS handles service group state when resources are intentionally brought online or taken offline outside of VCS control. The attribute can take the following values: **OnlineGroup:** If the configured application is started outside of VCS control, VCS brings the corresponding service group online. **OfflineGroup:** If the configured application is stopped outside of VCS control, VCS takes the corresponding service group offline. **OfflineHold:** If a configured application is stopped outside of VCS control, VCS sets the state of the corresponding VCS resource as offline. VCS does not take any parent resources or the service group offline. OfflineHold and OfflineGroup are mutually exclusive.

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
FaultOnMonitorTimeouts (user-defined) **Note:** This attribute can be overridden.	When a monitor times out as many times as the value specified, the corresponding resource is brought down by calling the clean function. The resource is then marked FAULTED, or it is restarted, depending on the value set in the RestartLimit attribute. When FaultOnMonitorTimeouts is set to 0, monitor failures are not considered indicative of a resource fault. A low value may lead to spurious resource faults, especially on heavily loaded systems. ■ Type and dimension: integer-scalar ■ Default: 4
FaultPropagation (user-defined) **Note:** This attribute can be overridden.	Specifies if VCS should propagate the fault up to parent resources and take the entire service group offline when a resource faults. The value 1 indicates that when a resource faults, VCS fails over the service group, if the group's AutoFailOver attribute is set to 1. The value 0 indicates that when a resource faults, VCS does not take other resources offline, regardless of the value of the Critical attribute. The service group does not fail over on resource fault. ■ Type and dimension: boolean-scalar ■ Default: 1
FireDrill (user-defined)	Specifies whether or not fire drill is enabled for the resource type. If the value is: ■ 0: Fire drill is disabled. ■ 1: Fire drill is enabled. You can override this attribute. ■ Type and dimension: boolean-scalar ■ Default: 0

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
IMF (user-defined) **Note:** This attribute can be overridden.	Determines whether the IMF-aware agent must perform intelligent resource monitoring. You can also override the value of this attribute at resource-level. Type and dimension: integer-association This attribute includes the following keys: ■ Mode Define this attribute to enable or disable intelligent resource monitoring. Valid values are as follows: ■ 0—Does not perform intelligent resource monitoring ■ 1—Performs intelligent resource monitoring for offline resources and performs poll-based monitoring for online resources ■ 2—Performs intelligent resource monitoring for online resources and performs poll-based monitoring for offline resources ■ 3—Performs intelligent resource monitoring for both online and for offline resources ■ MonitorFreq This key value specifies the frequency at which the agent invokes the monitor agent function. The value of this key is an integer. You can set this attribute to a non-zero value in some cases where the agent requires to perform poll-based resource monitoring in addition to the intelligent resource monitoring. See the *Cluster Server Bundled Agents Reference Guide* for agent-specific recommendations. After the resource registers with the IMF notification module, the agent calls the monitor agent function as follows: ■ After every (MonitorFreq x MonitorInterval) number of seconds for online resources ■ After every (MonitorFreq x OfflineMonitorInterval) number of seconds for offline resources ■ RegisterRetryLimit If you enable IMF, the agent invokes the imf_register agent function to register the resource with the IMF notification module. The value of the RegisterRetyLimit key determines the number of times the agent must retry registration for a resource. If the agent cannot register the resource within the limit that is specified, then intelligent monitoring is disabled until the resource state changes or the value of the Mode key changes.
IMFRegList (user-defined)	An ordered list of attributes whose values are registered with the IMF notification module. ■ Type and dimension: string-vector ■ Default: Not applicable.

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
InfoInterval (user-defined)	Duration (in seconds) after which the info function is invoked by the agent framework for ONLINE resources of the particular resource type. If set to 0, the agent framework does not periodically invoke the info function. To manually invoke the info function, use the command `hares -refreshinfo`. If the value you designate is 30, for example, the function is invoked every 30 seconds for all ONLINE resources of the particular resource type. ■ Type and dimension: integer-scalar ■ Default: 0
IntentionalOffline (user-defined)	Defines how VCS reacts when a configured application is intentionally stopped outside of VCS control. Add this attribute for agents that support detection of an intentional offline outside of VCS control. Note that the intentional offline feature is available for agents registered as V51 or later. The value 0 instructs the agent to register a fault and initiate the failover of a service group when the supported resource is taken offline outside of VCS control. The value 1 instructs VCS to take the resource offline when the corresponding application is stopped outside of VCS control. ■ Type and dimension: boolean-scalar ■ Default: 0
InfoTimeout (user-defined)	Timeout value for info function. If function does not complete by the designated time, the agent framework cancels the function's thread. ■ Type and dimension: integer-scalar ■ Default: 30 seconds
LevelTwoMonitorFreq (user-defined)	Specifies the frequency at which the agent for this resource type must perform second-level or detailed monitoring. Type and dimension: integer-scalar Default: 0

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
LogDbg (user-defined)	Indicates the debug severities enabled for the resource type or agent framework. Debug severities used by the agent functions are in the range of DBG_1–DBG_21. The debug messages from the agent framework are logged with the severities DBG_AGINFO, DBG_AGDEBUG and DBG_AGTRACE, representing the least to most verbose. ■ Type and dimension: string-keylist ■ Default: {} (none) The LogDbg attribute can be overridden. Using the LogDbg attribute, you can set DBG_AGINFO, DBG_AGTRACE, and DBG_AGDEBUG severities at the resource level, but it does not have an impact as these levels are agent-type specific. Veritas recommends to set values between DBG_1 to DBG_21 at resource level using the LogDbg attribute.
LogFileSize (user-defined)	Specifies the size (in bytes) of the agent log file. Minimum value is 64 KB. Maximum value is 134217728 bytes (128MB). ■ Type and dimension: integer-scalar ■ Default: 33554432 (32 MB)
LogViaHalog (user-defined)	Enables the log of all the entry points to be logged either in the respective agent log file or the engine log file based on the values configured. ■ 0: The agent's log goes into the respective agent log file. ■ 1: The C/C++ entry point's logs goes into the agent log file and the script entry point's logs goes into the engine log file using the `halog` command. Type: boolean-scalar Default: 0

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
MigrateWaitLimit (user-defined)	Number of monitor intervals to wait for a resource to migrate after the migrating procedure is complete. MigrateWaitLimit is applicable for the source and target node because the migrate operation takes the resource offline on the source node and brings the resource online on the target node. You can also define MigrateWaitLimit as the number of monitor intervals to wait for the resource to go offline on the source node after completing the migrate procedure and the number of monitor intervals to wait for the resource to come online on the target node after resource is offline on the source node. ■ Type and dimension: integer-scalar ■ Default: 2 **Note:** This attribute can be overridden. Probes fired manually are counted when MigrateWaitLimit is set and the resource is waiting to migrate. For example, if the MigrateWaitLimit of a resource is set to 5 and the MonitorInterval is set to 60 (seconds), the resource waits for a maximum of five monitor intervals (that is, 5 x 60), and if all five monitors within MigrateWaitLimit report the resource as online on source node, it sets the ADMIN_WAIT flag. If you run another probe, the resource waits for four monitor intervals (that is, 4 x 60), and if the fourth monitor does not report the state as offline on source, it sets the ADMIN_WAIT flag. This procedure is repeated for 5 complete cycles. Similarly, if resource not moved to online state within the MigrateWaitLimit then it sets the ADMIN_WAIT flag.
MigrateTimeout (user-defined)	Maximum time (in seconds) within which the migrate procedure must complete or else be terminated. ■ Type and dimension: integer-scalar ■ Default: 600 seconds **Note:** This attribute can be overridden.
MonitorInterval (user-defined) **Note:** This attribute can be overridden.	Duration (in seconds) between two consecutive monitor calls for an ONLINE or transitioning resource. **Note:** Note: The value of this attribute for the MultiNICB type must be less than its value for the IPMultiNICB type. See the *Cluster Server Bundled Agents Reference Guide* for more information. A low value may impact performance if many resources of the same type exist. A high value may delay detection of a faulted resource. ■ Type and dimension: integer-scalar ■ Default: 60 seconds

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
MonitorStatsParam (user-defined)	Stores the required parameter values for calculating monitor time statistics. `static str MonitorStatsParam = {Frequency = 10, ExpectedValue = 3000, ValueThreshold = 100, AvgThreshold = 40}` Frequency: The number of monitor cycles after which the average monitor cycle time should be computed and sent to the engine. If configured, the value for this attribute must be between 1 and 30. The value 0 indicates that the monitor cycle ti me should not be computed. Default=0. ExpectedValue: The expected monitor time in milliseconds for all resources of this type. Default=100. ValueThreshold: The acceptable percentage difference between the expected monitor cycle time (ExpectedValue) and the actual monitor cycle time. Default=100. AvgThreshold: The acceptable percentage difference between the benchmark average and the moving average of monitor cycle times. Default=40. ■ Type and dimension: integer-association ■ Default: Different value for each parameter.
MonitorTimeout (user-defined) **Note:** This attribute can be overridden.	Maximum time (in seconds) within which the monitor function must complete or else be terminated. ■ Type and dimension: integer-scalar ■ Default: 60 seconds
NumThreads (user-defined)	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. If the number of resources being managed by the agent is less than or equal to the NumThreads value, only that many number of threads are created in the agent. Addition of more resources does not create more service threads. Similarly deletion of resources causes service threads to exit. Thus, setting NumThreads to 1 forces the agent to just use 1 service thread no matter what the resource count is. The agent framework limits the value of this attribute to 30. ■ Type and dimension: integer-scalar ■ Default: 10
OfflineMonitorInterval (user-defined) **Note:** This attribute can be overridden.	Duration (in seconds) between two consecutive monitor calls for an OFFLINE resource. If set to 0, OFFLINE resources are not monitored. ■ Type and dimension: integer-scalar ■ Default: 300 seconds

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
OfflineTimeout (user-defined) **Note:** This attribute can be overridden.	Maximum time (in seconds) within which the offline function must complete or else be terminated. ■ Type and dimension: integer-scalar ■ Default: 300 seconds
OfflineWaitLimit (user-defined) **Note:** This attribute can be overridden.	Number of monitor intervals to wait for the resource to go offline after completing the offline procedure. Increase the value of this attribute if the resource is likely to take a longer time to go offline. Probes fired manually are counted when OfflineWaitLimit is set and the resource is waiting to go offline. For example, say the OfflineWaitLimit of a resource is set to 5 and the MonitorInterval is set to 60. The resource waits for a maximum of five monitor intervals (five times 60), and if all five monitors within OfflineWaitLimit report the resource as online, it calls the clean agent function. If the user fires a probe, the resource waits for four monitor intervals (four times 60), and if the fourth monitor does not report the state as offline, it calls the clean agent function. If the user fires another probe, one more monitor cycle is consumed and the resource waits for three monitor intervals (three times 60), and if the third monitor does not report the state as offline, it calls the clean agent function. ■ Type and dimension: integer-scalar ■ Default: 0
OnlineClass (user-defined)	Enables you to control the scheduling class for the online agent function (entry point). This attribute controls the class whether the entry point is in C or scripts. The following values are valid for this attribute: ■ RT (Real Time) ■ TS (Time Sharing) ■ -1—indicates that VCS does not use this attribute to control the scheduling class of entry points. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority ■ Type and dimension: string-scalar ■ Default: -1

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
OnlinePriority (user-defined)	Enables you to control the scheduling priority for the online agent function (entry point). This attribute controls the priority whether the entry point is in C or scripts. The following values are valid for this attribute: ■ 0—indicates the default priority value for the configured scheduling class as given by the OnlineClass for the operating system. Veritas recommends that you set the value of the OnlinePriority attribute to 0. ■ Greater than 0—indicates a value greater than the default priority for the operating system. ■ -1—indicates that VCS does not use this attribute to control the scheduling priority of entry points. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority ■ Type and dimension: string-scalar ■ Default: -1
OnlineRetryLimit (user-defined) **Note:** This attribute can be overridden.	Number of times to retry the `online` operation if the attempt to online a resource is unsuccessful. This parameter is meaningful only if the clean operation is implemented. ■ Type and dimension: integer-scalar ■ Default: 0
OnlineTimeout (user-defined) **Note:** This attribute can be overridden.	Maximum time (in seconds) within which the online function must complete or else be terminated. ■ Type and dimension: integer-scalar ■ Default: 300 seconds

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
OnlineWaitLimit (user-defined) **Note:** This attribute can be overridden.	Number of monitor intervals to wait for the resource to come online after completing the online procedure. Increase the value of this attribute if the resource is likely to take a longer time to come online. Each probe command fired from the user is considered as one monitor interval. For example, say the OnlineWaitLimit of a resource is set to 5. This means that the resource will be moved to a faulted state after five monitor intervals. If the user fires a probe, then the resource will be faulted after four monitor cycles, if the fourth monitor does not report the state as ONLINE. If the user again fires a probe, then one more monitor cycle is consumed and the resource will be faulted if the third monitor does not report the state as ONLINE. ■ Type and dimension: integer-scalar ■ Default: 2
OpenTimeout (user-defined) **Note:** This attribute can be overridden.	Maximum time (in seconds) within which the open function must complete or else be terminated. ■ Type and dimension: integer-scalar ■ Default: 60 seconds
Operations (user-defined)	Indicates valid operations for resources of the resource type. Values are OnOnly (can online only), OnOff (can online and offline), None (cannot online or offline). ■ Type and dimension: string-scalar ■ Default: OnOff
RestartLimit (user-defined) **Note:** This attribute can be overridden.	Number of times to retry bringing a resource online when it is taken offline unexpectedly and before VCS declares it FAULTED. ■ Type and dimension: integer-scalar ■ Default: 0

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
ScriptClass (user-defined)	Indicates the scheduling class of the script processes (for example, online) created by the agent. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority ■ Type and dimension: string-scalar ■ Default: -1 ■ Type and dimension: string-scalar ■ Default: TS
ScriptPriority (user-defined)	Indicates the priority of the script processes created by the agent. Use only one of the following sets of attributes to configure scheduling class and priority for VCS: ■ AgentClass, AgentPriority, ScriptClass, and ScriptPriority Or ■ OnlineClass, OnlinePriority, EPClass, and EPPriority ■ Type and dimension: string-scalar ■ Default: 0
SourceFile (user-defined)	File from which the configuration is read. Do not configure this attribute in main.cf. Make sure the path exists on all nodes before running a command that configures this attribute. ■ Type and dimension: string-scalar ■ Default: .\types.cf
SupportedActions (user-defined)	Valid action tokens for the resource type. ■ Type and dimension: string-vector ■ Default: {}
SupportedOperations (user-defined)	Indicates the additional operations for a resource type or an agent. Only `migrate` keyword is supported. ■ Type and dimension: string-keylist ■ Default: {} An example of a resource type that supports migration is a Kernel-based Virtual Machine (KVM).

Table D-2 Resource type attributes *(continued)*

Resource type attributes	Description
ToleranceLimit (user-defined) **Note:** This attribute can be overridden.	After a resource goes online, the number of times the monitor function should return OFFLINE before declaring the resource FAULTED. A large value could delay detection of a genuinely faulted resource. ■ Type and dimension: integer-scalar ■ Default: 0
TypeOwner (user-defined)	This attribute is used for VCS notification. VCS sends notifications to persons designated in this attribute when an event occurs related to the agent's resource type. If the agent of that type faults or restarts, VCS send notification to the TypeOwner. Note that while VCS logs most events, not all events trigger notifications. Make sure to set the severity level at which you want notifications to be sent to TypeOwner or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-scalar ■ Default: "" ■ Example: "jdoe@example.com"
TypeRecipients (user-defined)	The email-ids set in the TypeRecipients attribute receive email notification for events related to a specific agent. There are only two types of events related to an agent for which notifications are sent: ■ Agent fault - Warning ■ Agent restart - Information

Service group attributes

Table D-3 lists the service group attributes.

Table D-3 Service group attributes

Service Group Attributes	Definition
ActiveCount (system use only)	Number of resources in a service group that are active (online or waiting to go online). When the number drops to zero, the service group is considered offline. ■ Type and dimension: integer-scalar ■ Default: Not applicable.

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
AdministratorGroups (user-defined)	List of operating system user account groups that have administrative privileges on the service group. This attribute applies to clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: {} (none)
Administrators (user-defined)	List of VCS users with privileges to administer the group. **Note:** A Group Administrator can perform all operations related to a specific service group, but cannot perform generic cluster operations. See "About VCS user privileges and roles" on page 86. ■ Type and dimension: string-keylist ■ Default: {} (none)
Authority (user-defined)	Indicates whether or not the local cluster is allowed to bring the service group online. If set to 0, it is not, if set to 1, it is. Only one cluster can have this attribute set to 1 for a specific global group. See "About serialization–The Authority attribute" on page 492. ■ Type and dimension: integer-scalar ■ Default: 0
AutoClearCount (System use only)	Indicates the number of attempts that the VCS engine made to clear the state of the service group that has faulted and does not have a failover target. This attribute is used only if the AutoClearLimit attribute is set for the service group.
AutoClearInterval (user-defined)	Indicates the interval in seconds after which a service group that has faulted and has no failover target is cleared automatically. The state of the service group is cleared only if AutoClearLimit is set to a non-zero value. Default: 0
AutoclearLimit (user-defined)	Defines the number of attempts to be made to clear the Faulted state of a service group. Disables the auto-clear feature when set to zero.

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
AutoDisabled (system use only)	Indicates that VCS does not know the status of a service group (or specified system for parallel service groups). This could occur because the group is not probed (on specified system for parallel groups) in the SystemList attribute. Or the VCS engine is not running on a node designated in the SystemList attribute, but the node is visible. When VCS does not know the status of a service group on a node but you want VCS to consider the service group enabled, perform this command to change the AutoDisabled value to 0. `hagrp -autoenable grp -sys sys1` This command instructs VCS that even though VCS has marked the service group auto-disabled, you are sure that the service group is not online on sys1. For failover service groups, this is important because the service groups now can be brought online on remaining nodes. ■ Type and dimension: boolean-scalar ■ Default: 0
AutoFailOver (user-defined)	Indicates whether VCS initiates an automatic failover if the service group faults. The attribute can take the following values: ■ 0—VCS does not fail over the service group. ■ 1—VCS automatically fails over the service group if a suitable node exists for failover. ■ 2—VCS automatically fails over the service group only if a suitable node exists in the same system zone or sites where the service group was online. To set the value as 2, you must have enabled HA/DR license and the service group must not be hybrid. If you have not defined system zones or sites, the failover behavior is similar to 1. See " About controlling failover on service group or system faults" on page 354. ■ Type and dimension: integer-scalar ■ Default: 1 (enabled)

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
AutoRestart (user-defined)	Restarts a service group after a faulted persistent resource becomes online. The attribute can take the following values: ■ 0—Autorestart is disabled. ■ 1—Autorestart is enabled. ■ 2—When a faulted persistent resource recovers from a fault, the VCS engine clears the faults on all non-persistent faulted resources on the system. It then restarts the service group. See "About service group dependencies" on page 407. **Note:** This attribute applies only to service groups containing persistent resources. ■ Type and dimension: integer-scalar ■ Default: 1 (enabled)
AutoStart (user-defined)	Designates whether a service group is automatically started when VCS is started. ■ Type and dimension: boolean-scalar ■ Default: 1 (enabled)
AutoStartIfPartial (user-defined)	Indicates whether to initiate bringing a service group online if the group is probed and discovered to be in a PARTIAL state when VCS is started. ■ Type and dimension: boolean-scalar ■ Default: 1 (enabled)

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
AutoStartList (user-defined)	List of systems on which, under specific conditions, the service group will be started with VCS (usually at system boot). For example, if a system is a member of a failover service group's AutoStartList attribute, and if the service group is not already running on another system in the cluster, the group is brought online when the system is started. VCS uses the AutoStartPolicy attribute to determine the system on which to bring the service group online. **Note:** For the service group to start, AutoStart must be enabled and Frozen must be 0. Also, beginning with 1.3.0, you must define the SystemList attribute prior to setting this attribute. ■ Type and dimension: string-keylist ■ Default: {} (none)
AutoStartPolicy (user-defined)	Sets the policy VCS uses to determine the system on which a service group is brought online during an autostart operation if multiple systems exist. This attribute has three options: Order (default)—Systems are chosen in the order in which they are defined in the AutoStartList attribute. Load—Systems are chosen in the order of their capacity, as designated in the AvailableCapacity system attribute. System with the highest capacity is chosen first. **Note:** You cannot set the value Load when the cluster attribute Statistics is set to Enabled. Priority—Systems are chosen in the order of their priority in the SystemList attribute. Systems with the lowest priority is chosen first. ■ Type and dimension: string-scalar ■ Default: Order

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
CapacityReserved (system use only)	Indicates whether capacity is reserved to bring service groups online or to fail them over. Capacity is reserved only when the service group attribute FailOverPolicy is set to BiggestAvailable. This attribute is localized. ■ Type and dimension: Boolean ■ Default: "" Possible values: 1: Capacity is reserved. 0: Capacity is not reserved. The value can be reset using the `hagrp -flush` command. To list this attribute, use the -all option with the `hagrp -display` command.
ClusterFailOverPolicy (user-defined)	Determines how a global service group behaves when a cluster faults or when a global group faults. The attribute can take the following values: Manual—The group does not fail over to another cluster automatically. Auto—The group fails over to another cluster automatically if it is unable to fail over within the local cluster, or if the entire cluster faults. Connected—The group fails over automatically to another cluster only if it is unable to fail over within the local cluster. ■ Type and dimension: string-scalar ■ Default: Manual
ClusterList (user-defined)	Specifies the list of clusters on which the service group is configured to run. ■ Type and dimension: integer-association ■ Default: {} (none)

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition

ContainerInfo
(user-defined)
	Specifies if you can use the service group with the container. Assign the following values to the ContainerInfo attribute:
	■ Name: The name of the container.
	■ Type: The type of container. You can set this to Zone.
	■ Enabled: Specify the value as 1 to enable the container. Specify the value as 0 to disable the container. Specify the value as 2 to enable failovers from physical computers to virtual machines and from virtual machines to physical computers. Refer to the *Storage Foundation and High Availability Solutions Virtualization Guide* for more information on use of value 2 for the Enabled key.
	You can set a per-system value or a global value for this attribute.
	■ Type and dimension: string-scalar
	■ Default: {}
	For a given service group, you can either set the ContainerInfo attribute for the group or the ResContainerInfo attribute for the resources in the group. You can modify ContainerInfo only if the following conditions are met:
	■ ResContainerInfo is not localized for any resource in the group.
	■ ResContainerInfo is not set or its Name key is set to an empty string for all resources in the group.
	See "Resource attributes" on page 694.
	You can change the attribute scope from local to global as follows:
	`# hagrp -local <service_group_name> <attribute_name>`
	You can change the attribute scope from global to local as follows:
	`# hagrp -global <service_group_name> <attribute_name> <value> ... | <key> ... | {<key> <value>} ...`
	For more information about the `-local` option and the `-global` option, see the man pages associated with the `hagrp` command.

CurrentCount (system use only)	Number of systems on which the service group is active. ■ Type and dimension: integer-scalar ■ Default: Not applicable.

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
DisableFaultMessages (user-defined)	Suppresses fault and failover messages, for a group and its resources, from getting logged in the VCS engine log file. This attribute does not suppress the information messages getting logged in the log file. The attribute can take the following values: ■ 0 - Logs all the fault and failover messages for the service group and its resources. ■ 1 - Disables the fault and failover messages of the service groups, but continues to log resource messages. ■ 2 - Disables the fault and failover messages of the service group resources, but continues to log service group messages. ■ 3 - Disables the fault and failover messages of both service groups and its resources.
DeferAutoStart (system use only)	Indicates whether HAD defers the auto-start of a global group in the local cluster in case the global cluster is not fully connected. ■ Type and dimension: boolean-scalar ■ Default: Not applicable
Enabled (user-defined)	Indicates if a service group can be failed over or brought online. The attribute can have global or local scope. If you define local (system-specific) scope for this attribute, VCS prevents the service group from coming online on specified systems that have a value of 0 for the attribute. You can use this attribute to prevent failovers on a system when performing maintenance on the system. ■ Type and dimension: boolean-scalar ■ Default: 1 (enabled)
Evacuate (user-defined)	Indicates if VCS initiates an automatic failover when user issues `hastop -local -evacuate`. ■ Type and dimension: boolean-scalar ■ Default: 1
Evacuating (system use only)	Indicates the node ID from which the service group is being evacuated. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
EvacList (system use only)	Contains list of pairs of low priority service groups and the systems on which they will be evacuated. For example: `Grp1 EvacList  grp2 Sys0  grp3 Sys0  grp4 Sys4` Type and dimension: string-association Default: Not applicable.
EvacuatingForGroup (system use only)	Displays the name of the high priority service group for which evacuation is in progress. The service group name is visible only as long as the evacuation is in progress. Type and dimension: string-scalar Default: Not applicable.
Failover (system use only)	Indicates service group is in the process of failing over. ■ Type and dimension: boolean-scalar ■ Default: Not applicable

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
FailOverPolicy (user-defined)	Defines the failover policy used by VCS to determine the system to which a group fails over. It is also used to determine the system on which a service group has been brought online through manual operation. The policy is defined only for clusters that contain multiple systems: Priority—The system defined as the lowest priority in the SystemList attribute is chosen. Load—The system with the highest value of AvailableCapacity is chosen. RoundRobin—Systems are chosen based on the active service groups they host. The system with the least number of active service groups is chosen first. BiggestAvailable—Systems are chosen based on the forecasted available capacity for all systems in the SystemList. The system with the highest available capacity forecasted is selected. **Note:** VCS selects the node in an alphabetical order when VCS detects two systems with same values set for the policy Priority, Load, RoundRobin, or BiggestAvailable. Prerequisites for setting FailOverPolicy to BiggestAvailable: ■ The cluster attribute Statistics must be set to Enabled. ■ Veritas recommends that the cluster attribute HostMeters should contain at least one key. ■ The service group attribute Load must contain at least one key. ■ You cannot change the attribute from BiggestAvailable to some other value, when the service group attribute CapacityReserved is set to 1 because the VCS engine reserves system capacity when it determines BiggestAvailable for the service group. When the service group online transition completes and after the next forecast cycle, CapacityReserved is reset. ■ Type and dimension: string-scalar ■ Default: Priority

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
FaultPropagation (user-defined)	Specifies if VCS should propagate the fault up to parent resources and take the entire service group offline when a resource faults. The value 1 indicates that when a resource faults, VCS fails over the service group, if the group's AutoFailOver attribute is set to 1. If The value 0 indicates that when a resource faults, VCS does not take other resources offline, regardless of the value of the Critical attribute. The service group does not fail over on resource fault. ■ Type and dimension: boolean-scalar ■ Default: 1
FromQ (system use only)	Indicates the system name from which the service group is failing over. This attribute is specified when service group failover is a direct consequence of the group event, such as a resource fault within the group or a group switch. ■ Type and dimension: string-association ■ Default: Not applicable
Frozen (system use only)	Disables all actions, including autostart, online and offline, and failover, except for monitor actions performed by agents. (This convention is observed by all agents supplied with VCS.) ■ Type and dimension: boolean-scalar ■ Default: 0 (not frozen)
GroupOwner (user-defined)	This attribute is used for VCS email notification and logging. VCS sends email notification to the person designated in this attribute when events occur that are related to the service group. Note that while VCS logs most events, not all events trigger notifications. Make sure to set the severity level at which you want notifications to be sent to GroupOwner or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-scalar ■ Default: ""

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
GroupRecipients (user-defined)	This attribute is used for VCS email notification. VCS sends email notification to persons designated in this attribute when events related to the service group occur and when the event's severity level is equal to or greater than the level specified in the attribute. Make sure to set the severity level at which you want notifications to be sent to GroupRecipients or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-association ■ email id: The email address of the person registered as a recipient for notification. severity: The minimum level of severity at which notifications must be sent.
Guests (user-defined)	List of operating system user accounts that have Guest privileges on the service group. This attribute applies to clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: ""
IntentOnline (system use only)	Indicates whether to keep service groups online or offline. VCS sets this attribute to 1 if an attempt has been made to bring the service group online. For failover groups, VCS sets this attribute to 0 when the group is taken offline. For parallel groups, it is set to 0 for the system when the group is taken offline or when the group faults and can fail over to another system. VCS sets this attribute to 2 for service groups if VCS attempts to autostart a service group; for example, attempting to bring a service group online on a system from AutoStartList. ■ Type and dimension: integer-scalar ■ Default: Not applicable.

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
IntentionalOnlineList (system use only)	Lists the nodes where a resource that can be intentionally brought online is found ONLINE at first probe. IntentionalOnlineList is used along with AutoStartList to determine the node on which the service group should go online when a cluster starts. ■ Type and dimension: string-keylist ■ Default: Not applicable
LastSuccess (system use only)	Indicates the time when service group was last brought online. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Load
(user-defined)
	Indicates the multidimensional value expressing load exerted by a service group on the system.
	When the cluster attribute Statistics is enabled, the allowed key values are CPU, Mem, and Swap. The value for these keys in corresponding units as specified in MeterUnit (cluster attribute).
	When the cluster attribute Statistics is not enabled, the allowed key value is Units.
	■ Type and dimension: float-association
	■ Default: Not applicable
	The following additional considerations apply:
	■ You cannot change this attribute when the service group attribute CapacityReserved is set to 1 in the cluster and when the FailOverPolicy is set to BiggestAvailable. This is because the VCS engine reserves system capacity based on the service group attribute Load.
	When the service group's online transition completes and after the next forecast cycle, CapacityReserved is reset.
	■ If the FailOverPolicy is set to BiggestAvailable for a service group, the attribute Load must be specified with at least one of the following keys:
	  ■ CPU
	  ■ Mem
	  ■ Swap

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
ManageFaults (user-defined)	Specifies if VCS manages resource failures within the service group by calling the Clean function for the resources. This attribute can take the following values. NONE—VCS does not call the Clean function for any resource in the group. You must manually handle resource faults. See " About controlling Clean behavior on resource faults" on page 361. ■ Type and dimension: string-scalar ■ Default: ALL
ManualOps (user-defined)	Indicates if manual operations are allowed on the service group. ■ Type and dimension: boolean-scalar ■ Default = 1 (enabled)
MeterWeight (user-defined)	Represents the weight given for the cluster attribute's HostMeters key to determine a target system for a service group when more than one system meets the group attribute's Load requirements. ■ Type and dimension: integer-association ■ Default: Not applicable Additional considerations for configuring this attribute in main.cf or changing it at run time: ■ This is an optional service group attribute. If it is not defined for a group, the VCS considers the cluster attribute MeterWeight. ■ To override this attribute at an individual group level, define it at run time or in the main.cf file. Ensure that keys are subsets of the cluster attribute HostAvailableMeters. ■ You cannot change this attribute when the service group attribute CapacityReserved is set to 1 in the cluster. ■ The values for the keys represent weights of the corresponding parameters. It should be in range of 0 to 10.
MigrateQ (system use only)	Indicates the system from which the service group is migrating. This attribute is specified when group failover is an indirect consequence (in situations such as a system shutdown or another group faults and is linked to this group). ■ Type and dimension: string-association ■ Default: Not applicable

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
NumRetries (system use only)	Indicates the number of attempts made to bring a service group online. This attribute is used only if the attribute OnlineRetryLimit is set for the service group. ■ Type and dimension: integer-scalar ■ Default: Not applicable
OnlineAtUnfreeze (system use only)	When a node or a service group is frozen, the OnlineAtUnfreeze attribute specifies how an offline service group reacts after it or a node is unfrozen. ■ Type and dimension: integer-scalar ■ Default: Not applicable
OnlineClearParent	When this attribute is enabled for a service group and the service group comes online or is detected online, VCS clears the faults on all online type parent groups, such as online local, online global, and online remote. ■ Type and dimension: boolean-scalar ■ Default: 0 For example, assume that both the parent group and the child group faulted and both cannot failover. Later, when VCS tries again to bring the child group online and the group is brought online or detected online, the VCS engine clears the faults on the parent group, allowing VCS to restart the parent group too.
OnlineRetryInterval (user-defined)	Indicates the interval, in seconds, during which a service group that has successfully restarted on the same system and faults again should be failed over, even if the attribute OnlineRetryLimit is non-zero. This prevents a group from continuously faulting and restarting on the same system. ■ Type and dimension: integer-scalar ■ Default: 0
OnlineRetryLimit (user-defined)	If non-zero, specifies the number of times the VCS engine tries to restart a faulted service group on the same system on which the group faulted, before it gives up and tries to fail over the group to another system. ■ Type and dimension: integer-scalar ■ Default: 0

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
OperatorGroups (user-defined)	List of operating system user groups that have Operator privileges on the service group. This attribute applies to clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: ""
Operators (user-defined)	List of VCS users with privileges to operate the group. A Group Operator can only perform online/offline, and temporary freeze/unfreeze operations pertaining to a specific group. See "About VCS user privileges and roles" on page 86. ■ Type and dimension: string-keylist ■ Default: ""
Parallel (user-defined)	Indicates if service group is failover (0), parallel (1), or hybrid(2). ■ Type and dimension: integer-scalar ■ Default: 0
PathCount (system use only)	Number of resources in path not yet taken offline. When this number drops to zero, the engine may take the entire service group offline if critical fault has occurred. ■ Type and dimension: integer-scalar ■ Default: Not applicable
PCVAllowOnline (system use only)	Indicates whether ProPCV-enabled resources in a service group can be brought online on a node outside VCS control. ■ Type and dimension: boolean-scalar ■ Default: Not applicable See "About preventing concurrency violation" on page 365.

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
PolicyIntention (system use only)	Functions as a lock on service groups listed in the `hagrp -online -propagate` command and `hagrp -offline -propagate` command: ■ Type and dimension: integer-scalar When PolicyIntention is set to a non-zero value for the service groups in dependency tree, this attribute protects the service groups from any other operation. PolicyIntention can take three values. ■ The value 0 indicates that the service group is not part of the `hagrp -online -propagate` operation or the `hagrp -offline -propagate` operation. ■ The value 1 indicates that the service group is part of the `hagrp -online -propagate` operation. ■ The value 2 indicates that the service group is part of the `hagrp -offline -propagate` operation.

PreOnline
(user-defined)
	Indicates that the VCS engine should not bring online a service group in response to a manual group online, group autostart, or group failover. The engine should instead run the PreOnline trigger.
	You can set a local (per-system) value or a global value for this attribute. A per-system value enables you to control the firing of PreOnline triggers on specific nodes in the cluster.
	■ Type and dimension: boolean-scalar
	■ Default: 0
	You can change the attribute scope from local to global as follows:
	`# hagrp -local <service_group_name> <attribute_name>`
	You can change the attribute scope from global to local as follows:
	`# hagrp -global <service_group_name> <attribute_name> <value> ... | <key> ... | {<key> <value>} ...`
	For more information about the `-local` option and the `-global` option, see the man pages associated with the `hagrp` command.

PreOnlining (system use only)	Indicates that VCS engine invoked the preonline script; however, the script has not yet returned with group online. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
PreonlineTimeout (user-defined)	Defines the maximum amount of time in seconds the preonline script takes to run the command `hagrp -online -nopre` for the group. Note that HAD uses this timeout during evacuation only. For example, when a user runs the command `hastop -local -evacuate` and the Preonline trigger is invoked on the system on which the service groups are being evacuated. ■ Type and dimension: integer-scalar ■ Default: 300
Prerequisites (user-defined)	An unordered set of name=value pairs denoting specific resources required by a service group. If prerequisites are not met, the group cannot go online. The format for Prerequisites is: Prerequisites() = {Name=Value, name2=value2}. Names used in setting Prerequisites are arbitrary and not obtained from the system. Coordinate name=value pairs listed in Prerequisites with the same name=value pairs in Limits(). See System limits and service group prerequisites on page 389. ■ Type and dimension: integer-association

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
PreSwitch (user-defined)	Indicates whether VCS engine should invoke PreSwitch actions in response to a manual service group switch operation. **Note:** The engine does not invoke the PreSwitch action during a group fault or when you use `-any` option to switch a group. This attribute must be defined in the global group definition on the remote cluster. This attribute takes the following values: 0—VCS engine switches the service group normally. 1—VCS engine switches the service group based on the output of PreSwitch action of the resources. If you set the value as 1, the VCS engine looks for any resource in the service group that supports PreSwitch action. If the action is not defined for any resource, the VCS engine switches a service group normally. If the action is defined for one or more resources, then the VCS engine invokes PreSwitch action for those resources. If all the actions succeed, the engine switches the service group. If any of the actions fail, the engine aborts the switch operation. The engine invokes the PreSwitch action in parallel and waits for all the actions to complete to decide whether to perform a switch operation. The VCS engine reports the action's output to the engine log. The PreSwitch action does not change the configuration or the cluster state. See "Administering global service groups in a global cluster setup" on page 544. ■ Type and dimension: boolean-scalar ■ Default: 0
PreSwitching (system use only)	Indicates that the VCS engine invoked the agent's PreSwitch action; however, the action is not yet complete. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
PrintTree (user-defined)	Indicates whether or not the resource dependency tree is written to the configuration file. The value 1 indicates the tree is written. **Note:** For very large configurations, the time taken to print the tree and to dump the configuration is high. ■ Type and dimension: boolean-scalar ■ Default: 1

Priority
(user-defined)
	Enables users to designate and prioritize the service group. VCS does not interpret the value; rather, this attribute enables the user to configure the priority of a service group and the sequence of actions required in response to a particular event.
	If the cluster-level attribute value for PreferredFencingPolicy is set to Group, VCS uses this Priority attribute value to calculate the node weight to determine the surviving subcluster during I/O fencing race.
	VCS assigns the following node weight based on the priority of the service group:
	`Priority  Node weight`
	`1  625`
	`2  125`
	`3  25`
	`4  5`
	`0 or >=5  1`
	A higher node weight is associated with higher values of Priority. The node weight is the sum of node weight values for service groups which are ONLINE/PARTIAL.
	See "About preferred fencing" on page 248.
	■ Type and dimension: integer-scalar
	■ Default: 0

Probed (system use only)	Indicates whether all enabled resources in the group have been detected by their respective agents. ■ Type and dimension: boolean-scalar ■ Default: Not applicable

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
ProbesPending (system use only)	The number of resources that remain to be detected by the agent on each system. ■ Type and dimension: integer-scalar ■ Default: Not applicable
ProPCV (user-defined)	Indicates whether the service group is proactively prevented from concurrency violation for ProPCV-enabled resources. ■ Type and dimension: boolean-scalar ■ Default: 0 See "About preventing concurrency violation" on page 365.
Responding (system use only)	Indicates VCS engine is responding to a failover event and is in the process of bringing the service group online or failing over the node. ■ Type and dimension: integer-scalar ■ Default: Not applicable
Restart (system use only)	For internal use only. ■ Type and dimension: integer-scalar ■ Default: Not applicable
SourceFile (user-defined)	File from which the configuration is read. Do not configure this attribute in main.cf. Make sure the path exists on all nodes before running a command that configures this attribute. Make sure the path exists on all nodes before configuring this attribute. ■ Type and dimension: string-scalar ■ Default: ./main.cf

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
State (system use only)	Group state on each system: OFFLINE— All non-persistent resources are offline. ONLINE —All resources whose AutoStart attribute is equal to 1 are online. FAULTED—At least one critical resource in the group is faulted or is affected by a fault. PARTIAL—At least one, but not all, resources with Operations=OnOff is online, and not all AutoStart resources are online. STARTING—Group is attempting to go online. STOPPING— Group is attempting to go offline. MIGRATING— Group is attempting to migrate a resource from the source system to the target system. This state should be seen only as a combination of multiple states such as, ONLINE|STOPPING|MIGRATING, OFFLINE|STARTING|MIGRATING, and OFFLINE|MIGRATING. A group state may be a combination of multiple states described above. For example, OFFLINE | FAULTED, OFFLINE | STARTING, PARTIAL | FAULTED, PARTIAL | STARTING, PARTIAL | STOPPING, ONLINE | STOPPING ■ Type and dimension: integer-scalar ■ Default: Not applicable.

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
SysDownPolicy (user-defined)	Determines whether a service group is autodisabled when the system is down and if the service group is taken offline when the system is rebooted or is shut down gracefully. If SysDownPolicy contains the key AutoDisableNoOffline, the following conditions apply: ■ The service group is autodisabled when system is down, gracefully shut down, or is detected as down. ■ The service group is not taken offline when the system reboots or shuts down gracefully. Valid values: Empty keylist or the key AutoDisableNoOffline Default: Empty keylist For example, if a service group with SysDownPolicy = AutoDisableNoOffline is online on system *sys1*, it has the following effect for various commands: ■ The `hastop -local -evacuate` command for *sys1* is rejected ■ The `hastop -sysoffline` command is accepted but the service group with SysDownPolicy = AutoDisableNoOffline is not taken offline. ■ The `hastop -all` command is rejected.
SystemList (user-defined)	List of systems on which the service group is configured to run and their priorities. Lower numbers indicate a preference for the system as a failover target. **Note:** You must define this attribute prior to setting the AutoStartList attribute. ■ Type and dimension: integer-association ■ Default: "" (none)

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
SystemZones (user-defined)	Indicates the virtual sublists within the SystemList attribute that grant priority in failing over. Values are string/integer pairs. The string key is the name of a system in the SystemList attribute, and the integer is the number of the zone. Systems with the same zone number are members of the same zone. If a service group faults on one system in a zone, it is granted priority to fail over to another system within the same zone, despite the policy granted by the FailOverPolicy attribute. ■ Type and dimension: integer-association ■ Default: "" (none) **Note:** You cannot modify this attribute when SiteAware is set as 1 and Sites are defined.
Tag (user-defined)	Identifies special-purpose service groups created for specific VCS products. ■ Type and dimension: string-scalar ■ Default: Not applicable.
TargetCount (system use only)	Indicates the number of target systems on which the service group should be brought online. ■ Type and dimension: integer-scalar ■ Default: Not applicable.
TFrozen (user-defined)	Indicates if service groups can be brought online or taken offline on nodes in the cluster. Service groups cannot be brought online or taken offline if the value of the attribute is 1. ■ Type and dimension: boolean-scalar ■ Default: 0 (not frozen)
ToQ (system use only)	Indicates the node name to which the service is failing over. This attribute is specified when service group failover is a direct consequence of the group event, such as a resource fault within the group or a group switch. ■ Type and dimension: string-association ■ Default: Not applicable

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
TriggerEvent (user-defined)	For internal use only. ■ Type and dimension: boolean-scalar ■ Default: Not applicable
TriggerPath (user-defined)	Enables you to customize the trigger path. If a trigger is enabled but the trigger path is "" (default), VCS invokes the trigger from the $VCS_HOME/bin/*triggers* directory. If you specify an alternate directory, VCS invokes the trigger from that path. The value is case-sensitive. VCS does not trim the leading spaces or trailing spaces in the Trigger Path value. If the path contains leading spaces or trailing spaces, the trigger might fail to get executed. The path that you specify must be in the following format: $VCS_HOME/*TriggerPath*/*Trigger* For example, if TriggerPath is set to mytriggers/sg1, VCS looks for the preonline trigger scripts in the $VCS_HOME/mytriggers/sg1/preonline/ directory. ■ Type and dimension: string-scalar ■ Default: ""
TriggerResFault (user-defined)	Defines whether VCS invokes the resfault trigger when a resource faults. The value 0 indicates that VCS does not invoke the trigger. ■ Type and dimension: boolean-scalar ■ Default: 1
TriggerResRestart (user-defined)	Determines whether or not to invoke the resrestart trigger if resource restarts. See "About the resrestart event trigger" on page 455. To invoke the resrestart trigger for a specific resource, enable this attribute at the resource level. See "Resource attributes" on page 694. ■ Type and dimension: boolean-scalar ■ Default: 0 (disabled)

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
TriggerResStateChange (user-defined)	Determines whether or not to invoke the resstatechange trigger if resource state changes. See "About the resstatechange event trigger" on page 456. To invoke the resstatechange trigger for a specific resource, enable this attribute at the resource level. See "Resource attributes" on page 694. ■ Type and dimension: boolean-scalar ■ Default: 0 (disabled)

TriggersEnabled
(user-defined)
	Determines if a specific trigger is enabled or not.
	Triggers are disabled by default. You can enable specific triggers on all nodes or on selected nodes. Valid values are VIOLATION, NOFAILOVER, PREONLINE, POSTONLINE, POSTOFFLINE, RESFAULT, RESSTATECHANGE, and RESRESTART.
	To enable triggers on a node, add trigger keys in the following format:
	TriggersEnabled@node1 = {POSTOFFLINE, POSTONLINE}
	The postoffline trigger and postonline trigger are enabled on node1.
	To enable triggers on all nodes in the cluster, add trigger keys in the following format:
	TriggersEnabled = {POSTOFFLINE, POSTONLINE}
	The postoffline trigger and postonline trigger are enabled on all nodes.
	■ Type and dimension: string-keylist
	■ Default: {}
	You can change the attribute scope from local to global as follows:
	`# hagrp -local <service_group_name> <attribute_name>`
	You can change the attribute scope from global to local as follows:
	`# hagrp -global <service_group_name> <attribute_name> <value> ... | <key> ... | {<key> <value>} ...`
	For more information about the `-local` option and the `-global` option, see the man pages associated with the `hagrp` command.

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
TypeDependencies (user-defined)	Creates a dependency (via an ordered list) between resource types specified in the service group list, and all instances of the respective resource type. ■ Type and dimension: string-keylist ■ Default: ""
UnSteadyCount (system use only)	Represents the total number of resources with pending online or offline operations. This is a localized attribute. ■ Type and dimension: integer ■ Default: 0 To list this attribute, use the -all option with the `hagrp -display` command. The `hagrp -flush` command resets this attribute.
UserAssoc (user-defined)	This is a free form string-association attribute to hold any key-value pair. "Name" and "UITimeout" keys are reserved by VCS health view. You must not delete these keys or update the values corresponding to these keys, but you can add other keys and use it for any other purpose. ■ Type and dimension: string-association ■ Default: {} You can change the attribute scope from local to global as follows: **# hagrp -local <service_group_name> <attribute_name>** You can change the attribute scope from global to local as follows: **# hagrp -global <service_group_name> <attribute_name> <value> ... | <key> ... | {<key> <value>} ...** For more information about the `-local` option and `-global` option, see the man pages associated with the `hagrp` command.
UserIntGlobal (user-defined)	Use this attribute for any purpose. It is not used by VCS. ■ Type and dimension: integer-scalar ■ Default: 0

Table D-3 Service group attributes *(continued)*

Service Group Attributes	Definition
UserStrGlobal (user-defined)	VCS uses this attribute in the ClusterService group. Do not modify this attribute in the ClusterService group. Use the attribute for any purpose in other service groups. ■ Type and dimension: string-scalar ■ Default: 0
UserIntLocal (user-defined)	Use this attribute for any purpose. It is not used by VCS. ■ Type and dimension: integer-scalar ■ Default: 0
UserStrLocal (user-defined)	Use this attribute for any purpose. It is not used by VCS. ■ Type and dimension: string-scalar ■ Default: ""

System attributes

Table D-4 lists the system attributes.

Table D-4 System attributes

System Attributes	Definition
AgentsStopped (system use only)	This attribute is set to 1 on a system when all agents running on the system are stopped. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Table D-4 System attributes *(continued)*

System Attributes	Definition
AvailableCapacity (system use only)	Indicates the available capacity of the system. The function of this attribute depends on the value of the cluster-level attribute Statistics. If the value of the Statistics is: ■ Enabled—Any change made to AvailableCapacity does not affect VCS behavior. ■ MeteringOnly or Disabled—This attribute is populated based on Capacity and DynamicLoad (system attributes) and Load (service group attribute) specified using the `hasys -load` command. The key for this value is Units. You cannot configure this attribute in the main.cf file. ■ Type and dimension: float-association ■ Default: Not applicable

Table D-4 System attributes *(continued)*

System Attributes	Definition
Capacity (user-defined)	Represents total capacity of a system. The possible values are: ■ Enabled—The attribute Capacity is auto-populated by meters representing system capacity in absolute units. It has all the keys specified in HostMeters, such as CPU, Mem, and Swap. The values for keys are set in corresponding units as specified in the Cluster attribute MeterUnit. You cannot configure this attribute in the main.cf file if the cluster-level attribute Statistics is set to Enabled. If the cluster-level attribute Statistics is enabled, any change made to Capacity does not affect VCS behavior. ■ MeteringOnly or Disabled—Allows you to define a value for Capacity if the value of the cluster-level attribute Statistics is set to MeteringOnly or Disabled. This value is relative to other systems in the cluster and does not reflect any real value associated with a particular system. The key for this value is Units. The default value for this attribute is 100 Units. For example, the administrator may assign a value of 200 to a 16-processor system and 100 to an 8-processor system. You can configure this attribute in the main.cf file and also modify the value at run time if Statistics is set to MeteringOnly or Disabled. ■ Type and dimension: float-association ■ Default: Depends on the value set for Statistics.
ConfigBlockCount (system use only)	Number of 512-byte blocks in configuration when the system joined the cluster. ■ Type and dimension: integer-scalar ■ Default: Not applicable
ConfigCheckSum (system use only)	Sixteen-bit checksum of configuration identifying when the system joined the cluster. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Table D-4 System attributes *(continued)*

System Attributes	Definition
ConfigDiskState (system use only)	State of configuration on the disk when the system joined the cluster. ■ Type and dimension: integer-scalar ■ Default: Not applicable
ConfigFile (system use only)	Directory containing the configuration files. ■ Type and dimension: string-scalar ■ Default: "`/etc/VRTSvcs/conf/config`"
ConfigInfoCnt (system use only)	The count of outstanding CONFIG_INFO messages the local node expects from a new membership message. This attribute is non-zero for the brief period during which new membership is processed. When the value returns to 0, the state of all nodes in the cluster is determined. ■ Type and dimension: integer-scalar ■ Default: Not applicable
ConfigModDate (system use only)	Last modification date of configuration when the system joined the cluster. ■ Type and dimension: integer-scalar ■ Default: Not applicable
CPUThresholdLevel (user-defined)	Determines the threshold values for CPU utilization based on which various levels of logs are generated. The notification levels are Critical, Warning, Note, and Info, and the logs are stored in the file engine_A.log. If the Warning level is crossed, a notification is generated. The values are configurable at a system level in the cluster. ■ For example, the administrator may set the value of CPUThresholdLevel as follows: ■ CPUThresholdLevel={Critical=95, Warning=80, Note=75, Info=60} ■ Type and dimension: integer-association ■ Default: Critical=90, Warning=80, Note=70, Info=60
CPUUsage (system use only)	This attribute is deprecated. VCS monitors system resources on startup.
CPUUsageMonitoring	This attribute is deprecated. VCS monitors system resources on startup.

Table D-4 System attributes *(continued)*

System Attributes	Definition
CurrentLimits (system use only)	System-maintained calculation of current value of Limits. CurrentLimits = Limits - (additive value of all service group Prerequisites). ■ Type and dimension: integer-association ■ Default: Not applicable
DiskHbStatus (system use only)	Deprecated attribute. Indicates status of communication disks on any system. ■ Type and dimension: string-association ■ Default: Not applicable
DynamicLoad (user-defined)	System-maintained value of current dynamic load. The value is set external to VCS with the `hasys -load` command. When you specify the dynamic system load, VCS does not use the static group load. ■ Type and dimension: float-association ■ Default: "" (none)
EngineRestarted (system use only)	Indicates whether the VCS engine (HAD) was restarted by the hashadow process on a node in the cluster. The value 1 indicates that the engine was restarted; 0 indicates it was not restarted. ■ Type and dimension: boolean-scalar ■ Default: 0

EngineVersion
(system use only)
	Specifies the major, minor, maintenance-patch, and point-patch version of VCS.
	The value of EngineVersion attribute is in hexa-decimal format. To retrieve version information:
	`Major Version: EngineVersion >> 24 & 0xff`
	`Minor Version: EngineVersion >> 16 & 0xff`
	`Maint Patch: EngineVersion >>  8 & 0xff`
	`Point Patch: EngineVersion & 0xff`
	■ Type and dimension: integer-scalar
	■ Default: Not applicable

Table D-4 System attributes *(continued)*

System Attributes	Definition
FencingWeight (user-defined)	Indicates the system priority for preferred fencing. This value is relative to other systems in the cluster and does not reflect any real value associated with a particular system. If the cluster-level attribute value for PreferredFencingPolicy is set to System, VCS uses this FencingWeight attribute to determine the node weight to ascertain the surviving subcluster during I/O fencing race. See "About preferred fencing" on page 248. ■ Type and dimension: integer-scalar ■ Default: 0
Frozen (system use only)	Indicates if service groups can be brought online on the system. Groups cannot be brought online if the attribute value is 1. ■ Type and dimension: boolean-scalar ■ Default: 0
GUIIPAddr (user-defined)	Determines the local IP address that VCS uses to accept connections. Incoming connections over other IP addresses are dropped. If GUIIPAddr is not set, the default behavior is to accept external connections over all configured local IP addresses. See "User privileges for CLI commands" on page 88. ■ Type and dimension: string-scalar ■ Default: ""
HostAvailableForecast (system use only)	Indicates the forecasted available capacities of the systems in a cluster based on the past metered AvailableCapacity. The HostMonitor agent auto-populates values for this attribute, if the cluster attribute Statistics is set to Enabled. It has all the keys specified in HostMeters, such as CPU, Mem, and Swap. The values for keys are set in corresponding units as specified in the Cluster attribute MeterUnit. You cannot configure this attribute in main.cf. ■ Type and dimension: float-association ■ Default: Not applicable

Table D-4 System attributes *(continued)*

System Attributes	Definition
HostMonitor (system use only)	List of host resources that the HostMonitor agent monitors. The values of keys such as Mem and Swap are measured in MB or GB, and CPU is measured in MHz or GHz. ■ Type and dimension: string-keylist ■ Default: { CPU, Mem, Swap }
HostUtilization (system use only)	Indicates the percentage usage of the resources on the host as computed by the HostMonitor agent. This attribute populates all parameters specified in the cluster attribute HostMeters if Statistics is set to MeterHostOnly or Enabled. ■ Type and dimension: integer-association ■ Default: Not applicable
LicenseType (system use only)	Indicates the license type of the base VCS key used by the system. Possible values are: 0—DEMO 1—PERMANENT 2—PERMANENT_NODE_LOCK 3—DEMO_NODE_LOCK 4—NFR 5—DEMO_EXTENSION 6—NFR_NODE_LOCK 7—DEMO_EXTENSION_NODE_LOCK ■ Type and dimension: integer-scalar ■ Default: Not applicable
Limits (user-defined)	An unordered set of name=value pairs denoting specific resources available on a system. Names are arbitrary and are set by the administrator for any value. Names are not obtained from the system. The format for Limits is: Limits = { Name=Value, Name2=Value2}. ■ Type and dimension: integer-association ■ Default: ""

Table D-4 System attributes *(continued)*

System Attributes	Definition
LinkHbStatus (system use only)	Indicates status of private network links on any system. Possible values include the following: `LinkHbStatus = { nic1 = UP, nic2 = DOWN }` Where the value UP for *nic1* means there is at least one peer in the cluster that is visible on *nic1*. Where the value DOWN for *nic2* means no peer in the cluster is visible on *nic2*. ■ Type and dimension: string-association ■ Default: Not applicable
LLTNodeId (system use only)	Displays the node ID defined in the file. /etc/llttab. ■ Type and dimension: integer-scalar ■ Default: Not applicable
LoadTimeCounter (system use only)	System-maintained internal counter of how many seconds the system load has been above LoadWarningLevel. This value resets to zero anytime system load drops below the value in LoadWarningLevel. If the cluster-level attribute Statistics is enabled, any change made to LoadTimeCounter does not affect VCS behavior. ■ Type and dimension: integer-scalar ■ Default: Not applicable
LoadTimeThreshold (user-defined)	How long the system load must remain at or above LoadWarningLevel before the LoadWarning trigger is fired. If set to 0 overload calculations are disabled. If the cluster-level attribute Statistics is enabled, any change made to LoadTimeThreshold does not affect VCS behavior. ■ Type and dimension: integer-scalar ■ Default: 600

Table D-4 System attributes *(continued)*

System Attributes	Definition
LoadWarningLevel (user-defined)	A percentage of total capacity where load has reached a critical limit. If set to 0 overload calculations are disabled. For example, setting LoadWarningLevel = 80 sets the warning level to 80 percent. The value of this attribute can be set from 1 to 100. If set to 1, system load must equal 1 percent of system capacity to begin incrementing the LoadTimeCounter. If set to 100, system load must equal system capacity to increment the LoadTimeCounter. If the cluster-level attribute Statistics is enabled, any change made to LoadWarningLevel does not affect VCS behavior. ■ Type and dimension: integer-scalar ■ Default: 80
MemThresholdLevel (user-defined)	Determines the threshold values for memory utilization based on which various levels of logs are generated. The notification levels are Critical, Warning, Note, and Info, and the logs are stored in the file engine_A.log. If the Warning level is crossed, a notification is generated. The values are configurable at a system level in the cluster. For example, the administrator may set the value of MemThresholdLevel as follows: ■ MemThresholdLevel={Critical=95, Warning=80, Note=75, Info=60} ■ Type and dimension: integer-association ■ Default: Critical=90, Warning=80, Note=70, Info=60

Table D-4 System attributes *(continued)*

System Attributes	Definition
MeterRecord (system use only)	Acts as an internal system attribute with predefined keys. This attribute is updated only when the Cluster attribute AdpativePolicy is set to Enabled. ■ Type and dimension: integer-association ■ Default: Not applicable Possible keys are: ■ AvailableGC: Stores the value of the cluster attribute GlobalCounter when the HostMonitor agent updates the system attribute AvailableCapacity. If the value of AvailableGC for a system in running state is older than the last 24 values of the cluster attribute GlobalCounter it indicates: ■ Values are not updated at the required frequency by the HostMonitor agent. ■ Values of system attributes AvailableCapacity and HostUtilization are stale. ■ ForecastGC: Stores cluster attribute GlobalCounter value when system HostAvailableForecast attribute is updated. If the value of ForecastGC for a system in running state is older than the last 72 values of the cluster attribute GlobalCounter it indicates : ■ HostMonitor agent is not forecasting the available capacity at the required frequency. ■ The values of the system attribute HostAvailableForecast are stale. ■ If any of the running systems in SystemList have stale value in HostAvailableForecast when FailOverPolicy is set to BiggestAvailable, then VCS does not apply BiggestAvailable policy. Instead, it considers Priority as the FailOverPolicy.
NoAutoDisable (system use only)	When set to 0, this attribute autodisables service groups when the VCS engine is taken down. Groups remain autodisabled until the engine is brought up (regular membership). This attribute's value is updated whenever a node joins (gets into RUNNING state) or leaves the cluster. This attribute cannot be set manually. ■ Type and dimension: boolean-scalar ■ Default: 0

Table D-4 System attributes *(continued)*

System Attributes	Definition
NodeId (system use only)	System (node) identification specified in: /etc/llttab. ■ Type and dimension: integer-scalar ■ Default: Not applicable
OnGrpCnt (system use only)	Number of groups that are online, or about to go online, on a system. ■ Type and dimension: integer-scalar ■ Default: Not applicable
ReservedCapacity (system use only)	Indicates the reserved capacity on the systems for service groups which are coming online and with FailOverPolicy is set to BiggestAvailable. It has all of the keys specified in HostMeters, such as CPU, Mem, and Swap. The values for keys are set in corresponding units as specified in the Cluster attribute MeterUnit. ■ Type and dimension: float-association ■ Default: Not applicable When the service group completes online transition and after the next forecast cycle, ReservedCapacity is updated. You cannot configure this attribute in main.cf.
ShutdownTimeout (user-defined)	Determines whether to treat system reboot as a fault for service groups running on the system. On many systems, when a reboot occurs the processes are stopped first, then the system goes down. When the VCS engine is stopped, service groups that include the failed system in their SystemList attributes are autodisabled. However, if the system goes down within the number of seconds designated in ShutdownTimeout, service groups previously online on the failed system are treated as faulted and failed over. Veritas recommends that you set this attribute depending on the average time it takes to shut down the system. If you do not want to treat the system reboot as a fault, set the value for this attribute to 0. ■ Type and dimension: integer-scalar ■ Default: 600 seconds

Table D-4 System attributes *(continued)*

System Attributes	Definition
SourceFile (user-defined)	File from which the configuration is read. Do not configure this attribute in main.cf. Make sure the path exists on all nodes before running a command that configures this attribute. ■ Type and dimension: string-scalar ■ Default: ./main.cf
SupportedProtocol (System use only)	A system-level attribute that displays the protocol numbers supported by the running system node. ■ Type and dimension: integer-association ■ Default: 8.0.0.0000 =11000
SwapThresholdLevel (user-defined)	Determines the threshold values for swap space utilization based on which various levels of logs are generated. The notification levels are Critical, Warning, Note, and Info, and the logs are stored in the file engine_A.log. If the Warning level is crossed, a notification is generated. The values are configurable at a system level in the cluster. ■ For example, the administrator may set the value of SwapThresholdLevel as follows: ■ SwapThresholdLevel={Critical=95, Warning=80, Note=75, Info=60} ■ Type and dimension: integer-association ■ Default: Critical=90, Warning=80, Note=70, Info=60
SysInfo (system use only)	Provides platform-specific information, including the name, version, and release of the operating system, the name of the system on which it is running, and the hardware type. ■ Type and dimension: string-scalar ■ Default: Not applicable
SysName (system use only)	Indicates the system name. ■ Type and dimension: string-scalar ■ Default: Not applicable
SysState (system use only)	Indicates system states, such as RUNNING, FAULTED, EXITED, etc. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Table D-4 System attributes *(continued)*

System Attributes	Definition
SystemLocation (user-defined)	Indicates the location of the system. ■ Type and dimension: string-scalar ■ Default: ""
SystemOwner (user-defined)	Use this attribute for VCS email notification and logging. VCS sends email notification to the person designated in this attribute when an event occurs related to the system. Note that while VCS logs most events, not all events trigger notifications. Make sure to set the severity level at which you want notifications to SystemOwner or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-scalar ■ Default: "" ■ Example: "unknown"
SystemRecipients (user-defined)	This attribute is used for VCS email notification. VCS sends email notification to persons designated in this attribute when events related to the system occur and when the event's severity level is equal to or greater than the level specified in the attribute. Make sure to set the severity level at which you want notifications to be sent to SystemRecipients or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-association ■ email id: The e-mail address of the person registered as a recipient for notification. severity: The minimum level of severity at which notifications must be sent.
TFrozen (user-defined)	Indicates whether a service group can be brought online on a node. Service group cannot be brought online if the value of this attribute is 1. ■ Type and dimension: boolean-scalar ■ Default: 0

Table D-4 System attributes *(continued)*

System Attributes	Definition
TRSE (system use only)	Indicates in seconds the time to Regular State Exit. Time is calculated as the duration between the events of VCS losing port h membership and of VCS losing port a membership of GAB. ■ Type and dimension: integer-scalar ■ Default: Not applicable
UpDownState (system use only)	This attribute has four values: Down (0): System is powered off, or GAB and LLT are not running on the system. Up but not in cluster membership (1): GAB and LLT are running but the VCS engine is not. Up and in jeopardy (2): The system is up and part of cluster membership, but only one network link (LLT) remains. Up (3): The system is up and part of cluster membership, and has at least two links to the cluster. ■ Type and dimension: integer-scalar ■ Default: Not applicable
UserInt (user-defined)	Stores integer values you want to use. VCS does not interpret the value of this attribute. ■ Type and dimension: integer-scalar ■ Default: 0
VCSFeatures (system use only)	Indicates which VCS features are enabled. Possible values are: 0—No features enabled (VCS Simulator) 1—L3+ is enabled 2—Global Cluster Option is enabled Even though VCSFeatures attribute is an integer attribute, when you query the value with the `hasys -value` command or the `hasys -display` command, it displays as the string L10N for value 1 and DR for value 2. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Cluster attributes

Table D-5 lists the cluster attributes.

Table D-5 Cluster attributes

Cluster Attributes	Definition
AdministratorGroups (user-defined)	List of operating system user account groups that have administrative privileges on the cluster. This attribute applies to clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: ""
Administrators (user-defined)	Contains list of users with Administrator privileges. ■ Type and dimension: string-keylist ■ Default: ""
AutoClearQ (System use only)	Lists the service groups scheduled to be auto-cleared. It also indicates the time at which the auto-clear for the group will be performed.
AutoStartTimeout (user-defined)	If the local cluster cannot communicate with one or more remote clusters, this attribute specifies the number of seconds the VCS engine waits before initiating the AutoStart process for an AutoStart global service group. ■ Type and dimension: integer-scalar ■ Default: 150 seconds
AutoAddSystemtoCSG (user-defined)	Indicates whether the newly joined or added systems in cluster become part of the SystemList of the ClusterService service group if the service group is configured. The value 1 (default) indicates that the new systems are added to SystemList of ClusterService. The value 0 indicates that the new systems are not added to SystemList of ClusterService. ■ Type and dimension: integer-scalar ■ Default: 1
BackupInterval (user-defined)	Time period in minutes after which VCS backs up the configuration files if the configuration is in read-write mode. The value 0 indicates VCS does not back up configuration files. Set this attribute to at least 3. See "Scheduling automatic backups for VCS configuration files" on page 120. ■ Type and dimension: integer-scalar ■ Default: 0

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
CID (system defined)	The CID provides universally unique identification for a cluster. VCS populates this attribute once the engine passes an hacf-generated snapshot to it. This happens when VCS is about to go to a RUNNING state from the LOCAL_BUILD state. Once VCS receives the snapshot from the engine, it reads the file /etc/vx/.uuids/clusuuid file. VCS uses the file's contents as the value for the CID attribute. The clusuuid file's first line must not be empty. If the file does not exists or is empty VCS then exits gracefully and throws an error. A node that joins a cluster in the RUNNING state receives the CID attribute as part of the REMOTE_BUILD snapshot. Once the node has joined completely, it receives the snapshot. The node reads the file /etc/vx/.uuids/clusuuid to compare the value that it received from the snapshot with value that is present in the file. If the value does not match or if the file does not exist, the joining node exits gracefully and does not join the cluster. To populate the /etc/vx/.uuids/clusuuid file, run the /opt/VRTSvcs/bin/uuidconfig.pl utility. See "Configuring and unconfiguring the cluster UUID value" on page 167. You cannot change the value of this attribute with the haclus –modify command. ■ Type and dimension: string-scalar ■ Default: ""
ClusState (system use only)	Indicates the current state of the cluster. ■ Type and dimension: integer-scalar ■ Default: Not applicable.
ClusterAddress (user-defined)	Specifies the cluster's virtual IP address (used by a remote cluster when connecting to the local cluster). ■ Type and dimension: string-scalar ■ Default: ""
ClusterLocation (user-defined)	Specifies the location of the cluster. ■ Type and dimension: string-scalar ■ Default: ""
ClusterName (user-defined)	The name of cluster. ■ Type and dimension: string-scalar ■ Default: ""

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
ClusterOwner (user-defined)	This attribute used for VCS notification. VCS sends notifications to persons designated in this attribute when an event occurs related to the cluster. Note that while VCS logs most events, not all events trigger notifications. Make sure to set the severity level at which you want notifications to be sent to ClusterOwner or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. See "About VCS event notification" on page 431. ■ Type and dimension: string-scalar ■ Default: "" ■ Example: "jdoe@example.com"
ClusterRecipients (user-defined)	This attribute is used for VCS email notification. VCS sends email notification to persons designated in this attribute when events related to the cluster occur and when the event's severity level is equal to or greater than the level specified in the attribute. Make sure to set the severity level at which you want notifications to be sent to ClusterRecipients or to at least one recipient defined in the SmtpRecipients attribute of the NotifierMngr agent. ■ Type and dimension: string-association ■ email id: The e-mail address of the person registered as a recipient for notification. severity: The minimum level of severity at which notifications must be sent.
ClusterTime (system use only)	The number of seconds since January 1, 1970. This is defined by the lowest node in running state. ■ Type and dimension: string-scalar ■ Default: Not applicable
ClusterUUID (system use only)	Unique ID assigned to the cluster by Availability Manager. ■ Type and dimension: string-scalar ■ Default: Not applicable
CompareRSM (system use only)	Indicates if VCS engine is to verify that replicated state machine is consistent. This can be set by running the hadebug command. ■ Type and dimension: integer-scalar ■ Default: 0
ConnectorState (system use only)	Indicates the state of the wide-area connector (wac). If 0, wac is not running. If 1, wac is running and communicating with the VCS engine. ■ Type and dimension: integer-scalar ■ Default: Not applicable.

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
CounterInterval (user-defined)	Intervals counted by the attribute GlobalCounter indicating approximately how often a broadcast occurs that will cause the GlobalCounter attribute to increase. The default value of the GlobalCounter increment can be modified by changing CounterInterval. If you increase this attribute to exceed five seconds, consider increasing the default value of the ShutdownTimeout attribute. ■ Type and dimension: integer-scalar ■ Default: 5
CounterMissAction (user-defined)	Specifies the action that must be performed when the GlobalCounter is not updated for CounterMissTolerance times the CounterInterval. Possible values are LogOnly and Trigger. If you set CounterMissAction to LogOnly, the system logs the message in Engine Log and Syslog. If you set CounterMissAction to Trigger, the system invokes a trigger which has default action of collecting the comms tar file. ■ Type and dimension: string-scalar ■ Default: LogOnly
CounterMissTolerance (user-defined)	Specifies the time interval that can lapse since the last update of GlobalCounter before VCS reports an issue. If the GlobalCounter does not update within CounterMissTolerance times CounterInterval, VCS reports the issue. Depending on the CounterMissAction.value, appropriate action is performed. ■ Type and dimension: integer-scalar ■ Default: 20
CredRenewFrequency (user-defined)	The number of days after which the VCS engine renews its credentials with the authentication broker. For example, the value 5 indicates that credentials are renewed every 5 days; the value 0 indicates that credentials are not renewed. ■ Type and dimension: integer-scalar ■ Default = 0
DeleteOnlineResource (user-defined)	Defines whether you can delete online resources. Set this value to 1 to enable deletion of online resources. Set this value to 0 to disable deletion of online resources. You can override this behavior by using the -force option with the hares -delete command. ■ Type and dimension: integer-scalar ■ Default = 1
DumpingMembership (system use only)	Indicates that the engine is writing or dumping the configuration to disk. ■ Type and dimension: vector ■ Default: Not applicable.

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
EngineClass (user-defined)	The scheduling class for the VCS engine (HAD). The attribute can take the following values: RT, TS where RT = realtime and TS = timeshare. For information on the significance of these values, see the operating system documentation. ■ Type and dimension: string-scalar ■ Default: RT
EnableFFDC (user-defined)	Enables or disables FFDC logging. By default, FFDC logging is enabled. ■ Type and dimension: boolean-scalar ■ Default: 1
EnableVMAutoDiscovery (user-defined)	Enables or disables auto discovery of virtual machines. By default, auto discovery of virtual machines is disabled. ■ Type and dimension: integer-scalar ■ Default: 0
EnablePBF (user-defined)	Enables or disables priority based failover. When set to 1 (one), VCS gives priority to the online of high priority service group, by ensuring that its Load requirement is met on the system. ■ Type and dimension: boolean-scalar ■ Default: 0 (zero)
EnginePriority (user-defined)	The priority in which HAD runs. Generally, a greater priority value indicates higher scheduling priority. A range of priority values is assigned to each scheduling class. For more information on the range of priority values, see the operating system documentation. ■ Type and dimension: string-scalar ■ Default: ""

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
EngineShutdown (user-defined)	Defines the options for the hastop command. The attribute can assume the following values: Enable—Process all hastop commands. This is the default behavior. Disable—Reject all hastop commands. DisableClusStop—Do not process the hastop -all command; process all other hastop commands. PromptClusStop—Prompt for user confirmation before running the hastop -all command; process all other hastop commands. PromptLocal—Prompt for user confirmation before running the hastop -local command; reject all other hastop commands. PromptAlways—Prompt for user confirmation before running any hastop command. ■ Type and dimension: string-scalar ■ Default: Enable
FipsMode (system use only)	Indicates whether FIPS mode is enabled for the cluster. The value depends on the mode of the broker on the system. If FipsMode is set to 1, FIPS mode is enabled. If FipsMode is set to 0, FIPS mode is disabled. ■ Type and dimension: integer -scalar ■ Default: Not applicable You can verify the value of FipsMode as follows: `# haclus -value FipsMode`
GlobalCounter (system use only)	This counter increases incrementally by one for each counter interval. It increases when the broadcast is received. VCS uses the GlobalCounter attribute to measure the time it takes to shut down a system. By default, the GlobalCounter attribute is updated every five seconds. This default value, combined with the 600-second default value of the ShutdownTimeout attribute, means if system goes down within 120 increments of GlobalCounter, it is treated as a fault. Change the value of the CounterInterval attribute to modify the default value of GlobalCounter increment. ■ Type and dimension: integer-scalar ■ Default: Not applicable

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
Guests (user-defined)	List of operating system user accounts that have Guest privileges on the cluster. This attribute is valid clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: ""
GuestGroups (user-defined)	List of operating system user groups that have Guest privilege on the cluster. ■ Type and dimension: string-keylist ■ Default: Not applicable
DefaultGuestAccess (user-defined)	Indicates whether any authenticated user should have guest access to the cluster by default. The default guest access can be: ■ 0: Guest access for privileged users only. ■ 1: Guest access for everyone. ■ Type and dimension: boolean-scalar ■ Default: 0
GroupLimit (user-defined)	Maximum number of service groups. ■ Type and dimension: integer-scalar ■ Default: 200
HacliUserLevel (user-defined)	This attribute has two, case-sensitive values: NONE–hacli is disabled for all users regardless of role. COMMANDROOT–hacli is enabled for root only. **Note:** The command `haclus -modify HacliUserLevel` can be executed by root only. ■ Type and dimension: string-scalar ■ Default: NONE
HostAvailableMeters (System use only)	Lists the meters that are available for measuring system resources. You cannot configure this attribute in the main.cf file. ■ Type and dimension: string-association Keys are the names of parameters and values are the names of meter libraries. ■ Default: HostAvailableMeters = { CPU = “libmeterhost_cpu.so”, Mem = “libmeterhost_mem.so”, Swap = “libmeterhost_swap.so”}

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
HostMeters (user-defined)	Indicates the parameters (CPU, Mem, or Swap) that are currently metered in the cluster. ■ Type and dimension: string-keylist ■ Default: HostMeters = {“CPU”, “Mem”, “Swap”} You can configure this attribute in the main.cf file. You cannot modify the value at run time. The keys must be one or more from CPU, Mem, or Swap.
LockMemory (user-defined)	Controls the locking of VCS engine pages in memory. This attribute has the following values. Values are case-sensitive: ALL: Locks all current and future pages. CURRENT: Locks current pages. NONE: Does not lock any pages. ■ Type and dimension: string-scalar ■ Default: ALL
LogClusterUUID (user-defined)	Enables or disables logging of the cluster UUID in each log message. By default, cluster UUID is not logged. ■ Type and dimension: boolean-scalar ■ Default: 0
LogSize (user-defined)	Indicates the size of engine log files in bytes. Minimum value is = 65536 (equal to 64KB) Maximum value = 134217728 (equal to 128MB) ■ Type and dimension: integer-scalar ■ Default: 134217728 (128 MB)

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
MeterControl (user-defined)	Indicates the intervals at which metering and forecasting for the system attribute AvailableCapacity are done for the keys specified in HostMeters. ■ Type and dimension: integer-association This attribute includes the following keys: ■ MeterInterval Frequency in seconds at which metering is done by the HostMonitor agent. The value for this key can equal or exceed 30. The default value is 120 indicating that the HostMonitor agent meters available capacity and updates the System attribute AvailableCapacity every 120 seconds. The HostMonitor agent checks for changes in the available capacity for every monitoring cycle and when there is a change, the HostMonitor agent updates the values in the same monitoring cycle . The MeterInterval value applies only if Statistics is set to Enabled or MeterHostOnly. ■ ForecastCycle The number of metering cycles after which forecasting of available capacity is done. The value for this key can equal or exceed 1. The default value is 3 indicating that forecasting of available capacity is done after every 3 metering cycles. Assuming the default MeterInterval value of 120 seconds, forecasting is done after 360 seconds or 6 minutes. The ForecastCycle value applies only if Statistics is set to Enabled. You can configure this attribute in main.cf. You cannot modify the value at run time. The values of MeterInterval and ForecastCycle apply to all keys of HostMeters.
MeterUnit	Represents units for parameters that are metered. ■ Type and dimension: string-association ■ Default: CPU=CPU, Mem = MB, Swap = MB} You can configure this attribute in main.cf; if configured in main.cf, then it must contain units for all the keys as specified in HostMeters. You cannot modify the value at run time. When Statistics is set to Enabled then service group attribute Load, and the following system attributes are represented in corresponding units for parameters such as CPU, Mem, or Swap: ■ AvailableCapacity ■ HostAvailableForecast ■ Capacity ■ ReservedCapacity The values of keys such as Mem and Swap can be represented in MB or GB, and CPU can be represented in CPU, MHz or GHz.

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
MeterWeight (user-defined)	Indicates the default meter weight for the service groups in the cluster. You can configure this attribute in the main.cf file, but you cannot modify the value at run time. If the attribute is defined in the main.cf file, it must have at least one key defined. The weight for the key must be in the range of 0 to 10. Only keys from HostAvailableMeters are allowed in this attribute. ■ Type and dimension: integer-association ■ Default: {CPU = 10, Mem = 5, Swp = 1}
Notifier (system use only)	Indicates the status of the notifier in the cluster; specifically: State—Current state of notifier, such as whether or not it is connected to VCS. Host—The host on which notifier is currently running or was last running. Default = None Severity—The severity level of messages queued by VCS for notifier. Values include Information, Warning, Error, and SevereError. Default = Warning Queue—The size of queue for messages queued by VCS for notifier. ■ Type and dimension: string-association ■ Default: Different values for each parameter.
OpenExternalCommunicationPort (user-defined)	Indicates whether communication over the external communication port for VCS is allowed or not. By default, the external communication port for VCS is 14141. ■ Type and dimension: string-scalar ■ Valid values: YES, NO ■ Default: YES ■ YES: The external communication port for VCS is open. ■ NO: The external communication port for VCS is not open. **Note:** When the external communication port for VCS is not open, RemoteGroup resources created by the RemoteGroup agent cannot access VCS.
OperatorGroups (user-defined)	List of operating system user groups that have Operator privileges on the cluster. This attribute is valid clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: ""
Operators (user-defined)	List of users with Cluster Operator privileges. ■ Type and dimension: string-keylist ■ Default: ""

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
PanicOnNoMem (user-defined)	Indicate the action that you want VCS engine (HAD) to take if it cannot receive messages from GAB due to low-memory. ■ If the value is 0, VCS exits with warnings. ■ If the value is 1, VCS calls the GAB library routine to panic the system. ■ Default: 0
PreferredFencingPolicy	The I/O fencing race policy to determine the surviving subcluster in the event of a network partition. Valid values are Disabled, System, Group, or Site. Disabled: Preferred fencing is disabled. The fencing driver favors the subcluster with maximum number of nodes during the race for coordination points. System: The fencing driver gives preference to the system that is more powerful than others in terms of architecture, number of CPUs, or memory during the race for coordination points. VCS uses the system-level attribute FencingWeight to calculate the node weight. Group: The fencing driver gives preference to the node with higher priority service groups during the race for coordination points. VCS uses the group-level attribute Priority to determine the node weight. Site: The fencing driver gives preference to the node with higher site priority during the race for coordination points. VCS uses the site-level attribute Preference to determine the node weight. See "About preferred fencing" on page 248. ■ Type and dimension: string-scalar ■ Default: "Disabled"
PrintMsg (user-defined)	Enables logging TagM messages in engine log if set to 1. ■ Type and dimension: boolean-scalar ■ Default: 0
ProcessClass (user-defined)	Indicates the scheduling class processes created by the VCS engine. For example, triggers. ■ Type and dimension: string-scalar ■ Default = TS
ProcessPriority (user-defined)	The priority of processes created by the VCS engine. For example triggers. ■ Type and dimension: string-scalar ■ Default: ""

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
ProtocolNumber (System use only)	A cluster-level attribute that displays the cluster protocol number for the cluster. ■ Type and dimension: Integer-scalar ■ Default: Not applicable
ReadOnly (user-defined)	Indicates that cluster is in read-only mode. ■ Type and dimension: integer-scalar ■ Default: 1
ResourceLimit (user-defined)	Maximum number of resources. ■ Type and dimension: integer-scalar ■ Default: 5000
SecInfo256 (user-defined)	Enables the creation of secure passwords when this attribute is added to the `main.cf` file with the security key as the value of the attribute. ■ Type and dimension: string-scalar ■ Default: "" See "Encrypting agent passwords" on page 97.
IV256 (user-defined)	Enables the creation of secure passwords when this attribute is added to the `main.cf` file. This initialization vector (IV) is a fixed-size input to a cryptographic primitive that is typically required to be random. It adds randomness to the beginning of the encryption process. ■ Type and dimension: string-scalar ■ Default: "" **Note:** This attribute is used to encrypt or decrypt passwords when the cluster protocol version is 10000 or earlier. See "Encrypting agent passwords" on page 97.

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
IV256List (user-defined)	Enables the creation of secure passwords by maintaining a list of key-value pairs. Each encrypted password is saved as a key along with the IV that was used to encrypt the password. ■ Type and dimension: string-association ■ Default: "" This attribute gets added to or update in the `main.cf` file at the following events: ■ When a user gets added to a cluster using the `hauser -add` command in case of a non-secure cluster. ■ When you use the `vcsencrypt -vcs` or the `vcsencrypt -agent` command to encrypt the VCS password or an agent password respectively. If you use any of these commands, you do not need to run `haconf -dump` to make the configuration read-only again; this task is handled in the background. You cannot modify IV256List by using the `haconf - modify` command. However, you can use the `vcsencrypt -delkey` *`encryptedPasswordValue`* command to delete stale entries from this attribute. **Note:** This attribute and the `-delkey` option of the `vcsencrypt` command are applicable only with cluster protocol version 11000 or later. See "Encrypting agent passwords" on page 97.
SecInfoLevel (user-defined)	Denotes the password encryption privilege level. ■ Type and dimension: string-scalar ■ Default: R See "Encrypting agent passwords" on page 97.
SecureClus (user-defined)	Indicates whether the cluster runs in secure mode. The value 1 indicates the cluster runs in secure mode. This attribute cannot be modified when VCS is running. ■ Type and dimension: boolean-scalar ■ Default: 0

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
SiteAware (user-defined)	Indicates whether sites are configured for a cluster or not. ■ Type and dimension: boolean-scalar ■ Default: 0 Possible values are: ■ 1: Sites are configured. ■ 0: Sites are not configured. You can configure a site from Veritas InfoScale Operations Manager . This attribute will be automatically set to 1 when configured using Veritas InfoScale Operations Manager. If site information is not configured for some nodes in the cluster, those nodes are placed under a default site that has the lowest preference. See "Site attributes" on page 780.
SourceFile (user-defined)	File from which the configuration is read. Do not configure this attribute in main.cf. Make sure the path exists on all nodes before running a command that configures this attribute. ■ Type and dimension: string-scalar ■ Default: Not applicable.
Statistics (user-defined)	Indicates if statistics gathering is enabled and whether the FailOverPolicy can be set to BiggestAvailable. ■ Type and dimension: string-scalar ■ Default: Enabled You cannot modify the value at run time. Possible values are: ■ Enabled: The HostMonitor agent meters host utilization and forecasts the available capacity for the systems in the cluster. With this value set, FailOverPolicy for any service group cannot be set to Load. ■ MeterHostOnly: The HostMonitor agent meters host utilization but it does not forecast the available capacity for the systems in the cluster. The service group attribute FailOverPolicy cannot be set to BiggestAvailable. ■ Disabled: The HostMonitor agent is not started. Both metering of host utilization and forecasting of available capacity are disabled. The service group attribute FailOverPolicy cannot be set to BiggestAvailable. See "Service group attributes" on page 719.
Stewards (user-defined)	The IP address and hostname of systems running the steward process. ■ Type and dimension: string-keylist ■ {}

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
SystemRebootAction (user-defined)	Determines whether frozen service groups are ignored on system reboot. ■ Type and dimension: string-keylist ■ Default: "" If the SystemRebootAction value is IgnoreFrozenGroup, VCS ignores service groups that are frozen (TFrozen and Frozen) and takes the remaining service groups offline. If the frozen service groups have firm dependencies or hard dependencies on any other service groups which are not frozen, VCS gives an error. If the SystemRebootAction value is "", VCS tries to take all service groups offline. Because VCS cannot be gracefully stopped on a node where a frozen service group is online, applications on the node might get killed. **Note:** The SystemRebootAction attribute applies only on system reboot and system shutdown.
TypeLimit (user-defined)	Maximum number of resource types. ■ Type and dimension: integer-scalar ■ Default: 100
UseFence (user-defined)	Indicates whether the cluster uses SCSI-3 I/O fencing. The value SCSI3 indicates that the cluster uses either disk-based or server-based I/O fencing. The value NONE indicates it does not use either. ■ Type and dimension: string-scalar ■ Default: NONE
UserNames (user-defined)	List of VCS users. The installer uses `admin` as the default user name. ■ Type and dimension: string-association ■ Default: ""
VCSFeatures (system use only)	Indicates which VCS features are enabled. Possible values are: 0—No features are enabled (VCS Simulator) 1—L3+ is enabled 2—Global Cluster Option is enabled Even though the VCSFeatures is an integer attribute, when you query the value with the haclus -value command or the haclus -display command, it displays as the string L10N for value 1 and DR for value 2. ■ Type and dimension: integer-scalar ■ Default: Not applicable.

Table D-5 Cluster attributes *(continued)*

Cluster Attributes	Definition
VCSMode (system use only)	Denotes the mode for which VCS is licensed. Even though the VCSMode is an integer attribute, when you query the value with the haclus -value command or the haclus -display command, it displays as the string UNKNOWN_MODE for value 0 and VCS for value 7. ■ Type and dimension: integer-scalar ■ Default: Not applicable
WACPort (user-defined)	The TCP port on which the wac (Wide-Area Connector) process on the local cluster listens for connection from remote clusters. Type and dimension: integer-scalar ■ Default: 14155

Heartbeat attributes (for global clusters)

Table D-6 lists the heartbeat attributes. These attributes apply to global clusters.

Table D-6 Heartbeat attributes

Heartbeat Attributes	Definition
AgentState (system use only)	The state of the heartbeat agent. ■ Type and dimension: integer-scalar ■ Default: INIT
Arguments (user-defined)	List of arguments to be passed to the agent functions. For the Icmp agent, this attribute can be the IP address of the remote cluster. ■ Type and dimension: string-vector ■ Default: ""
AYAInterval (user-defined)	The interval in seconds between two heartbeats. ■ Type and dimension: integer-scalar ■ Default: 60 seconds
AYARetryLimit (user-defined)	The maximum number of lost heartbeats before the agent reports that heartbeat to the cluster is down. ■ Type and dimension: integer-scalar ■ Default: 3

Table D-6 Heartbeat attributes *(continued)*

Heartbeat Attributes	Definition
AYATimeout (user-defined)	The maximum time (in seconds) that the agent will wait for a heartbeat AYA function to return ALIVE or DOWN before being canceled. ■ Type and dimension: integer-scalar ■ Default: 30
CleanTimeOut (user-defined)	Number of seconds within which the Clean function must complete or be canceled. ■ Type and dimension: integer-scalar ■ Default: 300 seconds
ClusterList (user-defined)	List of remote clusters. ■ Type and dimension: string-keylist ■ Default: ""
InitTimeout (user-defined)	Number of seconds within which the Initialize function must complete or be canceled. ■ Type and dimension: integer-scalar ■ Default: 300 seconds
LogDbg (user-defined)	The log level for the heartbeat. ■ Type and dimension: string-keylist ■ Default: ""
State	The state of the heartbeat. ■ Type and dimension: integer-scalar ■ Default: Not applicable
StartTimeout (user-defined)	Number of seconds within which the Start function must complete or be canceled. ■ Type and dimension: integer-scalar ■ Default: 300 seconds
StopTimeout (user-defined)	Number of seconds within which the Stop function must complete or be canceled without stopping the heartbeat. ■ Type and dimension: integer-scalar ■ Default: 300 seconds

Remote cluster attributes

Table D-7 lists the RemoteCluster attributes. These attributes apply to remote clusters.

Table D-7 Remote cluster attributes

Remote cluster Attributes	Definition
AdministratorGroups (system use only)	List of operating system user account groups that have administrative privileges on the cluster. This attribute applies to clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: " "
Administrators (system use only)	Contains list of users with Administrator privileges. ■ Type and dimension: string-keylist ■ Default: ""
CID (system use only)	The CID of the remote cluster. See "Cluster attributes" on page 760.
ClusState (system use only)	Indicates the current state of the remote cluster as perceived by the local cluster. ■ Type and dimension: integer-scalar ■ Default: Not applicable
ClusterAddress (user-defined)	Specifies the remote cluster's virtual IP address, which is used to connect to the remote cluster by the local cluster. ■ Type and dimension: string-scalar ■ Default: ""
ClusterName (system use only)	The name of cluster. ■ Type and dimension: string-scalar ■ Default: ""
ClusterUUID (system use only)	Unique ID assigned to the cluster by Availability Manager. ■ Type and dimension: string-scalar ■ Default: Not applicable

Table D-7 Remote cluster attributes *(continued)*

Remote cluster Attributes	Definition
ConnectTimeout (user-defined)	Specifies the time in milliseconds for establishing the WAC to WAC connection. ■ Type and dimension: integer-scalar ■ Default: 300
DeclaredState (user-defined)	Specifies the declared state of the remote cluster after its cluster state is transitioned to FAULTED. See "Disaster declaration" on page 665. ■ Type and dimension: string-scalar ■ Default: "" The value can be set to one of the following values: ■ Disaster ■ Outage ■ Disconnect ■ Replica

EngineVersion
(system use only)
	Specifies the major, minor, maintenance-patch, and point-patch version of VCS.
	The value of EngineVersion attribute is in hexa-decimal format. To retrieve version information:
	`Major Version: EngineVersion >> 24 & 0xff`
	`Minor Version: EngineVersion >> 16 & 0xff`
	`Maint Patch: EngineVersion >>  8 & 0xff`
	`Point Patch: EngineVersion & 0xff`
	■ Type and dimension: integer-scalar
	■ Default: Not applicable

Guests (system use only)	List of operating system user accounts that have Guest privileges on the cluster. This attribute is valid for clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: ""

Table D-7 Remote cluster attributes *(continued)*

Remote cluster Attributes	Definition
OperatorGroups (system use only)	List of operating system user groups that have Operator privileges on the cluster. This attribute is valid for clusters running in secure mode. ■ Type and dimension: string-keylist ■ Default: 300 seconds
Operators (system use only)	List of users with Cluster Operator privileges. ■ Type and dimension: string-keylist ■ Default: ""
RemoteConnectInterval (user-defined)	Specifies the time in seconds between two successive attempts to connect to the remote cluster. ■ Type and dimension: integer-scalar ■ Default: 5
SocketTimeout (user-defined)	Specifies the time in seconds for WAC to WAC heartbeat. If no IAA is received in the specified time, connection with the remote WAC is assumed to be broken. ■ Type and dimension: integer-scalar ■ Default: 180
SourceFile (system use only)	File from which the configuration is read. Do not configure this attribute in main.cf. Make sure the path exists on all nodes before running a command that configures this attribute. ■ Type and dimension: string-scalar ■ Default: Not applicable.

Table D-7 Remote cluster attributes *(continued)*

Remote cluster Attributes	Definition
VCSFeatures (system use only)	Indicates which VCS features are enabled. Possible values are: 0—No features are enabled (VCS Simulator) 1—L3+ is enabled 2—Global Cluster Option is enabled Even though the VCSFeatures is an integer attribute, when you query the value with the haclus -value command or the haclus -display command, it displays as the string L10N for value 1 and DR for value 2. ■ Type and dimension: integer-scalar ■ Default: Not applicable.
VCSMode (system use only)	Denotes the mode for which VCS is licensed. Even though the VCSMode is an integer attribute, when you query the value with the haclus -value command or the haclus -display command, it displays as the string UNKNOWN_MODE for value 0 and VCS for value 7. ■ Type and dimension: integer-scalar ■ Default:Not applicable
WACPort (system use only)	The TCP port on which the wac (Wide-Area Connector) process on the remote cluster listens for connection from other clusters. ■ Type and dimension: integer-scalar ■ Default: 14155

Site attributes

Table D-8 lists the site attributes.

Table D-8 Site attributes

Site Attributes	Definition

Preference
(user-defined)
	Indicates the preference for a site.
	■ Type and dimension : integer-scalar
	■ Default : 3
	If preferred fencing is enabled, I/O fencing uses the Preference value to compute node weight as follows:
	`Preference  Node weight`
	`1  10000`
	`2  1000`
	`3  100`
	`4  10`

SystemList (user-defined)	Indicates the list of systems configured under a site. ■ Type and dimension : string-keylist ■ Default : ""

Appendix E

Accessibility and VCS

This appendix includes the following topics:

- About accessibility in VCS
- Navigation and keyboard shortcuts
- Support for accessibility settings
- Support for assistive technologies

About accessibility in VCS

Veritas InfoScale products meet federal accessibility requirements for software as defined in Section 508 of the Rehabilitation Act:

http://www.access-board.gov/508.htm

Cluster Server provides shortcuts for major graphical user interface (GUI) operations and menu items. Cluster Server is compatible with operating system accessibility settings as well as a variety of assistive technologies. All manuals also are provided as accessible PDF files, and the online help is provided as HTML, which appears in a compliant viewer.

Navigation and keyboard shortcuts

VCS uses standard operating system navigation keys and keyboard shortcuts. For its unique functions, VCS uses its own navigation keys and keyboard shortcuts which are documented below.

Navigation in the Java Console

Table E-1 lists keyboard navigation rules and shortcuts used in Cluster Manager (Java Console), in addition to those provided by the operating system.

Table E-1 Keyboard inputs and shortcuts

VCS keyboard input	Result
[Shift F10]	Opens a context-sensitive pop-up menu
[Spacebar]	Selects an item
[Ctrl Tab]	Navigates outside a table
[F2]	Enables editing a cell

Navigation in the Web console

The Web console supports standard browser-based navigation and shortcut keys for supported browsers.

All Veritas GUIs use the following keyboard navigation standards:

- Tab moves the cursor to the next active area, field, or control, following a preset sequence.
- Shift+Tab moves the cursor in the reverse direction through the sequence.
- Up-arrow and Down-arrow keys move the cursor up and down the items of a list.
- Either Enter or the Spacebar activates your selection. For example, after pressing Tab to select Next in a wizard panel, press the Spacebar to display the next screen.

Support for accessibility settings

Veritas software responds to operating system accessibility settings.

On UNIX systems, you can change the accessibility settings by using desktop preferences or desktop controls.

Support for assistive technologies

Veritas provides support for assistive technologies as follows:

- Cluster Manager (Java Console) is compatible with JAWS 4.5.
- Though graphics in the documentation can be read by screen readers, setting your screen reader to ignore graphics may improve performance.
- Veritas has not tested screen readers for languages other than English.