NexentaStor 5.3 HA CLI
Configuration Guide

Date: March, 2020
Part Number: ns-5.3-HAcliconfigguide-RevA
# Table of Contents

About this Document ................................................................................................................. 4

## 1  High-Availability for NexentaStor on Bare-Metal................................................................. 5

### Overview..................................................................................................................................... 5
Definition of Terms 6
Preparing for HA 7
Outline of HA Service Configuration 7

### Configuring Cluster..................................................................................................................... 8
Verify HA Cluster Feature 8
Update the Host Table 8
Enable the HA Service 9
Discover Unclustered Nodes 9
Create a New Cluster 9
With Only Private Network Heartbeats (Recommended) 9
With Only Public Network Heartbeats 10
With Both Public and Private Network Heartbeats 11
Manage Cluster 12

### Configuring HA Services ............................................................................................................ 13
Prerequisites 13
Discover Unclustered Pools 13
Create an HA Service 14
View HA Services 14
Initiate HA Service 15
Verify the Service 15
Preparing FC and iSCSI for HA Services 16
Failover 17
Interface Monitoring 17
Node Failing Scenario 18
Failback 21
Manage HA Services 21

### Repairing HA Service .............................................................................................................. 23
Diagnose HA Service Status 23
View Suggested Repair Action(s) 23
Repair Broken Service 24
Using “haservice recover” 24

### Advanced Configurations and Operations ............................................................................... 25
Add Disks or Network Interfaces to the Heartbeat Configuration 25
Adding Disk Heartbeat 25
Adding Private Network Heartbeats to an Existing Cluster 25
Removing Heartbeat 26
Add Pools to an HA service 26
Update a HA Service 26
Adding a VIP to an Existing Service 26
Removing VIP from an Existing Service ................................................. 26
Updating Default Netmask, VIP, NICs from an Existing Service ........... 27
Fast-Failover ......................................................................................... 27
Replace a Clustered Node ..................................................................... 28
Troubleshooting .................................................................................... 29
An Expired HA License ......................................................................... 29
Diagnose Unsuccessful HA Cluster Creation or Cluster Start ................ 29
Verify Virtual IP Address ....................................................................... 30
View HA Service Status ......................................................................... 30
View RSF Logs ..................................................................................... 30
Recovering Lost Cluster Configuration ................................................. 31

2 High-Availability for NexentaStor VSA on VMware ................................. 33
Overview ................................................................................................ 33
NexentaStor VSA HA Cluster Setup ......................................................... 34
Configurations to be done on VMware Side ......................................... 34
Configurations to be done on NexentaStor VMs ..................................... 35
Failover Operations .............................................................................. 36
Additional Considerations for HA Operations in VMware .................. 36
Example of an Automatic Failover Scenario ........................................ 37
Cluster Recovery after a Hard Failover .................................................. 39
Troubleshooting .................................................................................... 39
Scenario 1: Unavailability of vCenter Server ....................................... 39
Scenario 2: Update vCenter Configuration ............................................ 39
About this Document

This documentation presents information specific to Nexenta products. The information is for reference purposes and is subject to change.

The NexentaStor High Availability (HA) cluster is an enterprise-proven add-on feature to NexentaStor that manages the availability of critical storage pools. This Admin Guide demonstrates the basic steps and commands for configuring and managing the NexentaStor 5.x HA cluster on bare-metal or on VMware using the NexentaStor 5.x Command Line Interface (CLI). For additional user documentation, see https://nexenta.com/products/documentation.

Document History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ns-5.3-HAcliQuickstart-RevA</td>
<td>March, 2020</td>
<td>5.3 GA version</td>
</tr>
</tbody>
</table>
1 High-Availability for NexentaStor on Bare-Metal

Overview

NexentaStor HA cluster on bare metal consists of two servers with shared storage that can have any number of configured HA services. Each service in the cluster contains one or more storage pools with associated VIPs. Pools are built using shared storage devices that are accessible from both nodes. In the event of a node failure, the high availability of the pools on bare-metal is maintained by the cluster software that uses two functions to provide reliable automated failover services:

1. Heartbeats: Each node in the cluster communicates with the other node through different heartbeat mechanisms such as Network and disk heartbeats. These heartbeats are used to monitor the status of the remote node. As long as a remote node responds to network or disk heartbeats, it is deemed operational. If a remote node stops responding to heartbeats for a specific time period, the HA cluster detects system failure and an automatic failover operation is initiated.

2. SCSI based disk reservations: SCSI persistent reservations on devices in the pool are used to protect the data in the pool in case of split brain scenarios where two nodes may concurrently attempt to take ownership of a pool, or a node that was temporarily hung attempts to write to a pool it no longer owns. From this release NexentaStor uses PGR3 as the default reservation to lock volume access to the other node.

When the failed node is repaired and restarted, it rejoins the cluster and the administrator can control the redistribution of the services. The following diagram shows a typical Active / Active 2-node (NodeA, NodeB) HA cluster configuration consisting of two HA services (ServiceA, ServiceB), each of which has one pool (Pool A, Pool B). Under normal operation, each node is responsible for providing services to one pool. In the event of a node failure, the surviving node takes over the HA services for both the pools.

- Pools should be built using shared disks.
- Pools must be explicitly imported on the node that handles HA service creation request.
Definition of Terms

The terms in Table 1 are used throughout this document.

Table 1: Definition of Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HA Cluster</td>
<td>The term HA Cluster refers to the set of hardware that encompasses a pair of NexentaStor appliances with the shared storage. These clustered nodes are configured to exchange heartbeats through specified communication channels that provide information about their states.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>HA Service</td>
<td>The term HA service refers to the service that is run on the clustered nodes to provide high availability access to the user’s data. HA service consists of a pool residing on shared storage and optionally one or more VIP addresses for data access. When the HA service detects a node failure, it transfers ownership of the shared storage to the other node in the cluster pair.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An HA cluster:
- Is a pair of NexentaStor nodes with shared SAS storage enclosures providing highly available block and file services.
- requires that both NexentaStor nodes be licensed with the highAvailability feature.
- provides a set of highly available services, and continuously monitors cluster nodes for component failures.
- can have multiple services, each service comprising one or more storage pools.
- automatically monitors the state of service network interfaces (either IP or FC based) and can trigger automatic service failover in the event of a failure.

An HA service:
- Has any number of VIPs that fail over with the pool.
- Has a sequence of startup and shutdown scripts that describe how to secure and release physical storage, import and export the pools, start file services, and VIPs. These scripts also contain configurable timeout parameters to set how long the nodes should wait before failing over the services.

HA services are managed independently from one another and can be in several possible states.

Table 2: HA Service States

<table>
<thead>
<tr>
<th>Service State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starting</td>
<td>The startup scripts associated with the service are being launched.</td>
</tr>
<tr>
<td>Running</td>
<td>The service is operating and the associated pool(s) is now accessible. For this state, VIPs are also configured.</td>
</tr>
<tr>
<td>Stopping</td>
<td>The shutdown scripts associated with the service are being launched.</td>
</tr>
<tr>
<td>Stopped</td>
<td>The shutdown scripts successfully completed, the service has halted and access to the associated pool is now denied.</td>
</tr>
<tr>
<td>Broken_safe</td>
<td>A problem occurred while starting the service on the server, but it has been stopped safely and can be started again elsewhere.</td>
</tr>
<tr>
<td>Broken_unsafe</td>
<td>A fatal problem occurred while starting or stopping the service on the server. The service cannot be safely run on any other server in the cluster until it has been repaired by the operator.</td>
</tr>
<tr>
<td>Bouncing</td>
<td>Upon user request, the stop scripts for a service are launched. If they succeed, the service is immediately restarted on this server.</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Panicking</td>
<td>A problem has been detected, and an attempt to take some corrective action is being made.</td>
</tr>
<tr>
<td>Panicked</td>
<td>A fatal error occurred whilst running a service, and it cannot run on any server until it is repaired.</td>
</tr>
<tr>
<td>Aborting</td>
<td>The startup scripts for a service have failed, and the stop scripts are being run in an attempt to make the service safe to run elsewhere.</td>
</tr>
</tbody>
</table>

**Note:** The failover and recovery time is largely dependent on the amount of time it takes to fail over the pool components (such as disks, filesystems, and volumes) to the alternate appliance. Failover time is reduced when using fewer volumes or file systems in each pool.

### Preparing for HA

To enable HA services on NexentaStor nodes, ensure that the following items are in place:

- both NexentaStor nodes have valid licenses with the highAvailability feature.
- both NexentaStor nodes are configured with static management IP addresses.
- both NexentaStor nodes are configured with shared access (typically over SAS) to backend storage devices.
- both NexentaStor nodes are interconnected, preferably with a patch cable directly attached on ethernet ports on each node.

### Outline of HA Service Configuration

- Create a cluster named `democluster` on cluster nodes `nef06` and `nef07`.
- Discover unclustered pools that are available to become members of the cluster.
- Create an HA service named `hapool` for a shared pool with one or more VIP services.
- Optionally, add more pools to the HA service `hapool`.
- Now initiate the service, start `hapool` service on `nef06`.
- Set the service `hapool` to failover manually or automatically.
Configuring Cluster

The cluster creation process involves the following tasks:

- Verify that the HA feature is licensed and enabled on both NexentaStor appliances in the cluster.
- Update the host table on both the nodes so that the cluster nodes can resolve the IP address(es) of it's partner without relying on DNS services.
- Verify that the system day and time settings on each of the cluster member nodes are in sync.
- Create a new cluster.
- Verify the cluster status.

Verify HA Cluster Feature

The HA feature is available after activating the product license with the HighAvailability option has been applied on each of the cluster member nodes. Using the command below, verify that the HA feature is indeed running on each node in the cluster.

```
CLI@nef06> license show
```

Command output:

```
PROPERTY      VALUE
guid          44454c4c-5200-1051-8058-b3c04f563532
valid         yes
status        ok
product       NexentaStor
version       5.3
licensee     h.s@acme.com
serial        SR-DEV-NS-201614507
features     highAvailability
issued        Mon Sep 19 17:00:00 2016
expires       Sun Oct 30 17:00:00 2016
subscription  Mon Sep 19 17:00:00 2016 - Sun Oct 30 17:00:00 2016 (valid)
```

If the HA feature is not listed under the features, contact sales@nexenta.com to obtain your HA license token then run:

```
CLI@nef06> license activate <Activation Token>
```

Update the Host Table

On the first node in a cluster environment, update the hosts to ensure that they resolve to each other.

```
CLI@nef06> net create host <IP address of the second node> <host name of the second node>
```

Example:

```
CLI@nef06> net create host 10.3.53.110 nef07
```

Do the equivalent on the second node.

```
CLI@nef07> net create host 10.3.53.109 nef06
```

Verify if they resolve to each other by running the following command on both the nodes.

```
CLI@nef06> net list host
```
Example:

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>HOSTNAME</th>
<th>ALIAS</th>
<th>PROTOCOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>::1</td>
<td>localhost</td>
<td>-</td>
<td>ipv6</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>localhost</td>
<td>loghost</td>
<td>ipv4</td>
</tr>
<tr>
<td>10.3.53.109</td>
<td>nef06</td>
<td>-</td>
<td>ipv4</td>
</tr>
<tr>
<td>10.3.53.110</td>
<td>nef07</td>
<td>-</td>
<td>ipv4</td>
</tr>
</tbody>
</table>

Enable the HA Service

To find out if the HA service is running on each cluster node:

CLI@nef06> svc list ha

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESCRIPTION</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ha</td>
<td>HA cluster service</td>
<td>online</td>
</tr>
</tbody>
</table>

If HA service is not online, enable it:

CLI@nef06> svc enable ha

Discover Unclustered Nodes

Run the following command from both the nodes.

CLI@nef06> hacluster find-nodes

<table>
<thead>
<tr>
<th>NODE</th>
<th>ADDRESS</th>
<th>CRC</th>
<th>STARTUP TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>nef07</td>
<td>10.3.53.110</td>
<td>ba52</td>
<td>2016-03-24T16:49:44</td>
</tr>
<tr>
<td>nef06</td>
<td>10.3.53.109</td>
<td>ba52</td>
<td>2016-03-24T16:49:43</td>
</tr>
</tbody>
</table>

Create a New Cluster

Before creating a cluster, verify that the system day and time settings on each of the cluster member nodes are in sync. Explicit time synchronization is required only if no NTP client is running on the nodes.

CLI@nef06> config list system.date

<table>
<thead>
<tr>
<th>NAME</th>
<th>FLAGS</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>system.date</td>
<td>--</td>
<td>Fri Mar 31 08:58:11 2017</td>
</tr>
</tbody>
</table>

To set the system date and time:

CLI@nef06> config set system.date=3/31/2017 9:20:00

With Only Private Network Heartbeats (Recommended)

The following example lets you create a cluster on private interfaces (nef06-priv, nef07-priv) WITHOUT having a heartbeat automatically assigned for you on the public interface.

This example uses nef06-priv (10.10.10.10) and nef07-priv (10.10.10.20) as endpoints of a back-to-back network cable (i.e. not connected to the LAN)

1. Verify if there is any cluster already defined.

CLI@nef06> hacluster status

No cluster defined
2. On each node add the private end point addresses (as they are not discoverable on the LAN) - assuming here the local end points have already been created (i.e. on nef06 nef06-priv has been created and assigned to an interface, and for nef07-priv on nef07):

   CLI@nef06> net create host 10.10.10.20 nef07-priv
   and on nef07:
   CLI@nef07> net create host 10.10.10.10 nef06-priv

3. Now create cluster using nef06-priv and nef07-priv. The example shown here creates a cluster named democluster on the two nodes nef06 and nef07 with the description –d “Nef Cluster”.

   CLI@nef06> hacluster create -d "Nef-cluster" nef06-priv,nef07-priv
democluster

4. Verify if the cluster is created. Use “–e” option along with the “hacluster status” command whenever you need to get a detailed output of the cluster.

   CLI@nef06> hacluster status –e

== Cluster status ==
NAME    STATUS  NODES  SERVICES  DESCRIPTION
democluster  ok      2/2    0/0       Nef cluster

== Cluster configuration ==
FC MONITORING SERIAL HEARTBEATS ENABLED NETWORK MONITORING
no             no                         yes

== Nodes ==
NODE  STATUS  SERVICES  ADDRESS      HostId   Release
nef06 up      0/0      10.3.53.109  688b5ce3  5.3
nef07 up      0/0      10.3.53.110  778f4c71  5.3

== Heartbeats ==
ID  TYPE  FROM  TO     PEER ADDRESS   STATUS  POOL
0   net   nef06 nef07  nef07-priv     up      -
1   net   nef07 nef06  nef06-priv     up      -

WITH ONLY PUBLIC NETWORK HEARTBEATS
The following example creates a cluster with two nodes, nef06 and nef07 that are not resolved to each other yet.

5. Verify if there is any cluster already defined.

   CLI@nef06> hacluster status
   No cluster defined

6. Create cluster using nef06 and nef07 as hosts.

   CLI@nef06> hacluster create -d "Nef cluster" nef06,nef07 democluster

   The example shown here creates a cluster named democluster using the two nodes nef06 and nef07 as hosts with the description –d “Nef Cluster”.

7. Check status of the cluster.

   CLI@nef06> hacluster status –e
WITH BOTH PUBLIC AND PRIVATE NETWORK HEARTBEATS
This section covers the steps to create a cluster using public (nef06, nef07) and private network heartbeats (nef06-priv, nef07-priv) using the –H option.

1. Create cluster using nef06, nef07 and nef06-priv, nef07-priv.

   CLI@nef06> hacluster create -d "Nef cluster" -H nef06-priv:nef07-priv, nef07-priv:nef06-priv nef06, nef07 democluster

2. Verify the status of the heartbeats.

   CLI@nef06> hacluster status -e
Manage Cluster

You can manage the cluster using the `hacluster` subcommands in Table 4. For more details on all the commands listed see the relevant sections that is discussed within the doc.

### Table 4: Managing an HA Cluster

<table>
<thead>
<tr>
<th>Cluster Tasks</th>
<th>Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>Removes the cluster service without destroying it. Use this command to destroy all cluster-resources.</td>
<td>CLI@nexenta&gt; hacluster reset &lt;cluster&gt;</td>
</tr>
<tr>
<td>Remove/destroy cluster</td>
<td>Destroys a cluster. Once you remove a cluster, all clustered pools remain imported. Any cluster service in a broken state will need to be repaired before <code>hacluster destroy</code> can be issued. You can destroy a cluster only when both nodes are alive.</td>
<td>CLI@nexenta&gt; hacluster destroy &lt;cluster&gt;</td>
</tr>
<tr>
<td>Check-VIP</td>
<td>Verifies the VIP is consistent, or checks if the VIP is resolved across all the nodes in the cluster.</td>
<td>CLI@nexenta&gt; hacluster check-vip &lt;vipname&gt;</td>
</tr>
<tr>
<td>Fast-failover</td>
<td>Performs fast failover from the node that is currently running the service. When you run this command as part of the fast-failover process, the pool does not get exported. This is achieved by halting the node that is currently running the service to avoid the pool export time.</td>
<td>CLI@nexenta&gt; hacluster fast-failover (--halt</td>
</tr>
<tr>
<td>Reinitialize</td>
<td>Resets cluster configuration files. This is only recommended to be run when <code>hacluster destroy</code> fails and after analyzing and fixing the issues that were blocking the cluster destroy action.</td>
<td>CLI@nexenta&gt; hacluster reinitialize See the Note in Replace a Clustered Node section to learn about the scenario where this reinitialization feature can be applied.</td>
</tr>
<tr>
<td>Find-nodes</td>
<td>Finds all nodes available for clustering.</td>
<td>CLI@nexenta&gt; hacluster find-nodes</td>
</tr>
<tr>
<td>Find-pools</td>
<td>Finds imported pools that can be used later when you create an HA service. On both nodes, ensure that the pool is available for clustering.</td>
<td>CLI@nexenta&gt; hacluster find-pools</td>
</tr>
<tr>
<td>Recover</td>
<td>Recovers the cluster settings if the configuration is lost.</td>
<td>CLI@nexenta&gt; hacluster recover</td>
</tr>
<tr>
<td>Replace</td>
<td>Rejoins the newly installed appliance to rejoin the cluster in a node-failing scenario. For example if you have a cluster consisting of nodes nodea and nodeb and you replaced nodeb with a new node nodenew and you want the new node to join the cluster.</td>
<td>CLI@nexenta&gt; hacluster replace [-nyv] &lt;node&gt; &lt;new&gt;</td>
</tr>
</tbody>
</table>
Configuring HA Services

Prerequisites

A HA service provides orchestrated failover and failback of all file and block services supported by one or more NexentaStor storage pool. A HA service is typically configured on top of one NexentaStor storage pool.

Before creating an HA service, you should have:
- completed the creation of an HA cluster as described in the previous section
- completed creation of a storage pool, visible from both NexentaStor cluster nodes
- the storage pool that will support the HA service must be imported
- the name of the HA service must match the name of the storage pool supporting the service. It must contain only alpha-numeric characters.
- one or more Virtual IP addresses available when creating an HA service for IP based storage services. On the other hand, no VIP is needed for an HA service protecting Fibre Channel services.

Discover Unclustered Pools

Using the command below, you can find imported pools that can be used later when you create an HA service. On both the nodes, ensure that the pool is available for clustering.

```
CLI@nexenta> hacluster find-pools
```

Command output:

```
NAME   GUID
---    ---
  test  1797983880284287424
 hapool 5795983885284640389
```

Note: this command will only show pools that have all of their components online and _visible_ to both nodes. If one node of the cluster does not detect all of the components of a pool, it will not be eligible to be added in as a service. Nexenta recommends that you always verify that the shared pool is in a healthy state before creating an HA service. The HA cluster command does not detect disk and pool failures.

```
CLI@nexenta> pool status <name of the pool>
```
Create an HA Service

We can now add a service named hapool to control the currently imported pool (also named hapool) on node nef06 with the VIP address vip01 (10.3.53.111/255.255.255.0 in the example below). When creating a service, you can specify the node where you intend the service to start using the –m flag. You can also specify the list of nodes on which the HA service is allowed to run. If you do not provide the node name, the service is initiated from a random node among available cluster nodes.

Create the HA service using the following command:

```
CLI@nexenta> haservice create [ -evnf] [ -d <description>] [ -m <node>] [ -r <timeout>] [ -i <delay>] [ -g <guid>] [ -N <nodes>] [ -V <vips>] <pool>
```

Example:
```
CLI@nexenta> haservice create -V vip01@10.3.53.111/255.255.255.0=nef06:e1000g0,nef07:e1000g0 -m nef06 hapool
```

Note:
1. The HA service name must match the pool name, a required argument to the command.
2. HA service VIPs are created, managed and maintained exclusively via the haservice command. Do not use the generic ip command to modify or destroy HA service VIPs.

The example above configures the following:
- Using the –m flag is recommended to ensure that the service is started on the node where the pool is currently imported. In the example shown here, the node nef06 is designated as the main or local server from which the HA service hapool should run.
- VIP associated with the HA service. VIP has the following structure:
  `vipname@ipaddress[/mask]=node1:linkname,node2:linkname`
- The VIP is created as part of the HA service creation so no additional steps are required to add a VIP to the HA service. These VIPs are resolvable for both nodes and assigns the address to the given interfaces.
- The HA service immediately starts on one of the cluster nodes right after service creation, either on the local node (if the –m flag is used) or on a different node as specified in the haservice create command.

View HA Services

You can view the list of HA services you created using the command below. If you do not specify the service name, all the services for the cluster are listed. The following command below displays the nodes, VIPs, HA service state, and cluster nodes associated with the hapool HA service.

```
CLI@nexenta> haservice list <service>
```

Command output:
```
NAME      DESCRIPTION   VIPs    NODES       RUNNING  STOPPED  BROKEN
hapool    zup service   vip01   nef06,nef07 nef06    nef07   -
```

Verifying the VIP Name Interface

The VIP name interface will be listed on the node that runs the HA service, nef06 in the example here.
```
CLI@nef06> ip list
```

NAME TYPE STATE ADDRESS
When you move the HA Service from nef06 to the other node nef07, the IP interface would automatically move to the other node, nef07.

```
CLI@nef07> ip list
```

```
NAME            TYPE      STATE  ADDRESS
lo0/v4          static    ok     127.0.0.1/8
lo0/v6          static    ok     ::1/128
vmxnet3s0/vip01 static    ok     10.3.53.111/24
vmxnet3s0/v4    dhcp      ok     10.3.53.110/24
vmxnet3s0/v6    addrconf  ok     fe80::250:56ff:feb8:f063/10
```

### Initiate HA Service

After you created an HA service, you can set the specific service to start either in automatic (-a) or manual (-m) mode. The command below specifies that the `hapool` service will be started automatically.

```
CLI@nexenta> haservice set-mode [-amnv] <service> <node>
```

Example:

```
CLI@nexenta> haservice set-mode -a hapool nef06
```

If you selected the manual mode, restart the service manually with the following CLI command.

```
CLI@nexenta> haservice start <service> <node>
```

Example:

```
CLI@nexenta> haservice start hapool nef06
```

See the **Troubleshooting** chapter at the end of this document if the HA service fails to start.

In order to start a cluster service you must have a local static IP on the interface first. The interface will be marked as blocked by RSF cluster if there is no pre-existing static IP address on it. And also to assign a VIP onto an interface you must already have an IP.

### Verify the Service

Verify the service you created.

```
CLI@nexenta> haservice status <service>
```

Example:

```
CLI@nexenta> haservice status hapool
==service==
  hapool
==status==
  NODE STATUS      MODE      UNBLOCKED
  nef06 broken_safe automatic yes
```
Preparing FC and iSCSI for HA Services

Table 3 below provides a checklist that can be used to prepare for a successful iSCSI and FC HA Services. When the following configurations are completed on one of the cluster nodes, they are shared across the nodes in the cluster.

For more information on creating and sharing FC and iSCSI CLI volumes, configuring targets, target groups, and mapping LUNS see the NexentaStor CLI Configuration Guide at https://nexenta.com/products/documentation.

Table 3: FC and iSCSI HA Service Checklist

<table>
<thead>
<tr>
<th>Checklist Items</th>
<th>Related FC CLI Commands</th>
<th>Related iSCSI CLI Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure targets</td>
<td>fctarget</td>
<td>iscsitarget, iscsiauth</td>
</tr>
<tr>
<td>Configure target groups</td>
<td>targetgroup</td>
<td>targetgroup</td>
</tr>
<tr>
<td>Map LUNs</td>
<td>Lunmapping, logicalunit</td>
<td>Lunmapping, logicalunit</td>
</tr>
</tbody>
</table>

Following are the additional steps needed for a successful FC HA Service:

1. ALUA is enabled by default, if it is OFF enable ALUA on any one of the HA cluster node.
2. Ensure that you have cluster setup before enabling ALUA.
   
   CLI@nexenta> hactl alua-enable

3. Verify if ALUA is enabled.
   
   CLI@nexenta> hactl alua-status
   ALUA configuration detected
   Configuration file : /opt/HAC/RSF-1/etc/ALUA.cfg
   Master host : nodeA
   Remote host : nodeB
   Local host node ID : 0
   Remote host node ID : 1
   Remote host for sync : nodeB

   nodeA reports
Failover

**INTERFACE MONITORING GLOBALLY ON ALL HA SERVICES**

Interface monitoring monitors the status of services interfaces (aka front-end interfaces). When enabled, if a network interface hosting a VIP goes down, then that HA service will become inaccessible to clients (even though the hosting server itself is up and running). In this situation the cluster will respond to the failure by moving the service/VIP to the remaining node in the cluster as long as the corresponding network interface for the VIP is available.

This default behavior where a service is moved when any monitored network interface goes down can be adjusted on a per-service basis. To fine tune the failover and fine tune the monitoring for each HA service separately see the next section Fine Tune Monitoring. You must use the REST API to customize the failover and to set different monitoring for each HA service.

To view the status of net or fc monitoring:

```
CLI@nexenta> hacluster status
== Cluster configuration ==
FC MONITORING  SERIAL HEARTBEATS ENABLED  NETWORK MONITORING
yes            no                         yes
```

Net monitoring is enabled by default whereas you can enable FC monitoring when you create the cluster. Also both FC and Net monitors can be enabled using the following commands.

To monitor the links that a HA service uses:

```
CLI@nexenta> hactl monitor-enable <monitor>
Example:
To enable network monitoring, use netmon in the command:
CLI@nexenta> hactl monitor-enable netmon
To enable fc monitoring, use fcmmon in the command:
CLI@nexenta> hactl monitor-enable fcmmon
```

**FINE TUNE INTERFACE MONITORING FOR EACH HA SERVICE**

When a cluster is created the default network interface monitoring behavior is to fail over a service when any interface hosting that service VIP(s) becomes unavailable. This action can be fine tuned on a per-service basis by setting a service specific monitor expression.
To fine tune the failover and to set the monitoring for each HA service separately use the following API call “Set monitor expressions for this service” under “rsf”.

```plaintext
PUT  rsf/clusters/(clusterName)/services/(serviceName)/monitorExpression
```

**Implementation Notes**
Set monitor expressions for this service

**Response Class**
Model | Model Schema
object

**Parameters**
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
<th>Parameter Type</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>clusterName</td>
<td>(required)</td>
<td>RSF-1 cluster name</td>
<td>path</td>
<td>string</td>
</tr>
<tr>
<td>serviceName</td>
<td>(required)</td>
<td>Service name</td>
<td>path</td>
<td>string</td>
</tr>
<tr>
<td>caller</td>
<td></td>
<td>API caller (for debug)</td>
<td>query</td>
<td>string</td>
</tr>
</tbody>
</table>

**Payload**

```json
[ "expressions": [
  "expression": {
    "elements": [
      null
    ],
    "operator": "\n"
  },
  "nodes": [ ]
]
```

**Response Messages**

<table>
<thead>
<tr>
<th>HTTP Status Code</th>
<th>Reason</th>
<th>Response Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td></td>
</tr>
<tr>
<td>202</td>
<td>In progress</td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>Invalid parameters</td>
<td></td>
</tr>
<tr>
<td>404</td>
<td>Not found</td>
<td></td>
</tr>
</tbody>
</table>

Where each element in "expressions" is either:

- The name of a network interface
- Special value FC means "Fibre Channel target mode port monitoring"
- Another nested expression object

The "operator" can be either:

- AND - all elements must be "up" for the overall expression to be "up"
- OR - all elements must be "down" for the overall expression to be "down"
- EOR, XOR and NOT are also supported.
The following example shows a HA service with 3 VIPs configured to run on interfaces bnx0, bnx1 and bnx2. And the expression below provides the functionality to trigger a failover if either of the first 2 VIPs become inaccessible. Since the third VIP is a management interface, we do not want the cluster to react to failure there.

```
{
  "expressions": [
    {
      "expression": {
        "elements": [
          "bnx0",
          "bnx1"
        ],
        "operator": "AND"
      },
      "nodes": [
        "node1",
        "node2"
      ]
    }
  ]
}
```

**NODE FAILING SCENARIO**

In the event of either node failing (nef06 failing in this example) the surviving node, nef07, takes over the HA service for the pool if it is built using the shared storage devices that are accessible from both the nodes.

The following example shows how the HA service (hapool) failed over to the node (nef07) when the node (nef06) was powered off.

1. List the HA services running on the clustered nodes.
   ```
   CLI@nef06> haservice list
   NAME    DESCRIPTION       NODES        RUNNING  STOPPED  BROKEN
   hapool  myclusterservice  nef06,nef07  nef06    nef07    -
   This example shows a single HA service (hapool) running on node nef06.
   ```

2. Verify the existing pools on both the nodes.
   ```
   CLI@nef06> pool list
   NAME   SIZE    ALLOC   FREE   AVAIL  DEDUP  EXPANDSZ  FRAG  HEALTH
   hapool 9.63G   100.2M  9.53G  99%    1.00x  -          0%    ONLINE
   rpool 12.47G  7.41G   5.06G  41%    1.00x  -          26%   ONLINE
   ```

3. Verify the status of the HA service.
   ```
   CLI@nef06> haservice status
   Service: hapool
   NODE      STATUS   MODE       UNBLOCKED
   nef06     running  automatic  yes
   nef07     stopped  automatic  yes
   From the above example you notice that the service is running on the node nef06.
   ```

4. Now power off the node that has the HA (nef06) service running.
   ```
   CLI@nef06> poweroff
   ```
5. Now check the status of the HA service from the surviving node. The example here shows only one cluster service running on nef06. If you have more than one service on the node, when failing over, all the services will fail over to the other node in the cluster.

CLI@nef07> haservice status
Service: hapool

<table>
<thead>
<tr>
<th>NODE</th>
<th>STATUS</th>
<th>MODE</th>
<th>UNBLOCKED</th>
</tr>
</thead>
<tbody>
<tr>
<td>nef06</td>
<td>unknown</td>
<td>automatic</td>
<td>yes</td>
</tr>
<tr>
<td>nef07</td>
<td>running</td>
<td>automatic</td>
<td>yes</td>
</tr>
</tbody>
</table>

6. Now check the status of the HA cluster from the surviving node.

CLI@nef07> hacluster status

== Nodes ==

<table>
<thead>
<tr>
<th>NODE</th>
<th>STATUS</th>
<th>SERVICES</th>
<th>ADDRESS</th>
<th>HostId</th>
<th>Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>nef06</td>
<td>down</td>
<td>0/1</td>
<td>10.3.53.109</td>
<td>1fb4ba7c</td>
<td>3.12.0</td>
</tr>
<tr>
<td>nef07</td>
<td>up</td>
<td>1/1</td>
<td>10.3.53.110</td>
<td>808b556b</td>
<td>3.12.0</td>
</tr>
</tbody>
</table>

7. From nef07 verify that the pool (hapool) imported to it.

CLI@nef07> pool list

<table>
<thead>
<tr>
<th>NAME</th>
<th>SIZE</th>
<th>ALLOC</th>
<th>FREE</th>
<th>AVAIL</th>
<th>DEDUP</th>
<th>EXPANDSZ</th>
<th>FRAG</th>
<th>HEALTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>hapool</td>
<td>9.63G</td>
<td>100.2M</td>
<td>9.53G</td>
<td>99%</td>
<td>1.00x</td>
<td>-</td>
<td>0%</td>
<td>ONLINE</td>
</tr>
<tr>
<td>rpool</td>
<td>12.47G</td>
<td>7.41G</td>
<td>5.06G</td>
<td>41%</td>
<td>1.00x</td>
<td>-</td>
<td>26%</td>
<td>ONLINE</td>
</tr>
</tbody>
</table>

To force a failover of all running services from one cluster node to another:

CLI@nexenta> haservice failover <from-node> <to-node>
Failback

When the failed node is repaired and restarted, it rejoins the cluster and you as an administrator can control where the pool is redistributed. Now to failback the pool (hapool) to the repaired node nef06 run the following command on the nef07 node.

```
CLI@nexenta> haservice move <service> <node>
```

Example:

```
CLI@nef07> haservice move hapool nef06
```

This sets the cluster back in its original configuration with the pool (hapool) on node (nef06).

View and Modify the Failover Reaction Settings

By default the failover settings for a cluster is set to “conservative”. Using the following `hactl` subcommands you can update the failover settings for a cluster. If you do not specify the “timeout” setting, it will implicitly apply the settings from the “reaction”.

Here the `<failover reaction>` controls the amount of time a heartbeat is missed before a node is considered down. And `<runtimeout>` dictates the amount of time a user can wait after a remote node down event is detected, before starting that service.

To view the default failover reaction settings:

```
CLI@nexenta> hactl failover-status
```

*Failover reaction: conservative*  
*Run timeout failover reaction: conservative*

To update the settings:

```
CLI@nexenta> hactl failover-set -f <reaction> [-r <timeout>]
```

Example:

```
CLI@nexenta> hactl failover-set -f aggressive -r normal
```

Manage HA Services

Using the `haservice` subcommands in Table 5, you can perform the following operations on an HA service:

Table 5: Managing an HA Service.

<table>
<thead>
<tr>
<th>Service Operation</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all the created HA services</td>
<td>CLI@nexenta&gt; haservice list</td>
</tr>
<tr>
<td>Stop service on a specific node.</td>
<td>CLI@nexenta&gt; haservice stop &lt;service&gt;</td>
</tr>
<tr>
<td>Start service on a specific node.</td>
<td>CLI@nexenta&gt; haservice stop hapool</td>
</tr>
<tr>
<td>Remove a specific service. Once the service has been</td>
<td>CLI@nexenta&gt; haservice destroy hapool</td>
</tr>
<tr>
<td>destroyed, the associated pools are exported.</td>
<td>Deletes the service hapool.</td>
</tr>
<tr>
<td>Trigger a failover of all running services</td>
<td>CLI@nexenta&gt; haservice failover [-ynvs]</td>
</tr>
</tbody>
</table>
from one cluster node to another. | `<from-node> <to-node>`
---|---
Fail over an HA service to a different node. | `CLI@nexenta> haservice move hapool nef07
Moves the service `hapool` to node `nef07`.
Mark the broken _unsafe_ service as recovered. Note: The failure that caused the broken service has to be resolved before running this or you will risk crashing the head or possible data corruption. | `CLI@nexenta> haservice mark hapool nef07
Marks the service `hapool` as recovered.`
Set the service to start on auto (\(-a\)) or manual (\(-m\)) mode. | `CLI@nexenta> haservice set-mode -m hapool nef06
Sets the HA service `hapool` to manual mode in node `nef06`.

Display the HA service status and the configured VIPs for a specified service. | `CLI@nexenta> haservice status <service name>`
Delete a pool from an HA service | `CLI@nexenta> haservice delete-pool hapool <pool name>`
Adding VIP to an existing HA service: The appliances in the HA Cluster group must be resolvable to each other. This means they must be able to detect each other on the network and communicate. When you configure or create the cluster, NexentaStor automatically adds the VIPs to the `/etc/hosts` file on both the appliances. In the following example, the VIP `vip05` with IP address `9.8.7.6/255.0.0.0` is added to the HA service named `hapool`. | `CLI@nexenta> haservice add-vip hapool vip05
9.8.7.6/255.0.0.0
nef06:e1000g0,nef07:e1000g0
For more details, see Update a HA Service section under Advanced Configurations chapter.
Delete, or update a VIP address associated with an HA service. | `CLI@nexenta> haservice delete-vip hapool vip05
CLI@nexenta> haservice update-vip hapool vip05

For more details, see Update a HA Service section under Advanced Configurations chapter.`
Repairing HA Service

When the service goes to broken_safe, broken_unsafe, use the following sections to repair the service.

Diagnose HA Service Status

You can view the status information on all HA services as shown by the example below. For a complete list of HA service status values, refer to Table 2.

```
CLI@nef06> haservice status

Command output:
status:
NODE    STATUS       MODE       UNBLOCKED
nef06   broken_safe   automatic  yes
nef07   stopped      manual     yes
```

View Suggested Repair Action(s)

If the service is in broken_safe, broken_unsafe state use the following command to view suggested actions to repair the HA service. Use this command to know if a specific HA service can or cannot be repaired at this time. In the list of suggested actions, you will be prompted to use either 'haservice recover' command or other manual repair actions to fix the service.

Following is an example that shows “haservice recover” command as the suggested repair action to fix the broken service.

```
CLI@nef06> haservice show-repairs -v HA
Available per- node repair actions for service 'HA':

Node: nef06
Resolution: All resources are offline - suggest transitioning service “HA” to stopped state
Broken: yes
Fixed: no
Split brain: no
State: broken_safe
Suggested repair action: haservice recover --offline HA nef06

Errors: no errors available
Online resources: no online resources available
Offline resources:
  c0t5000C5000FA53FFBd0s0 not reserved
  c0t5000C50056363648C3d0s0 not reserved
  c0t5000C50000F4601B7d0s0 not reserved
  c0t5000C50056366407d0s0 not reserved
  ZFS pool tank
  VIP vip01 (10.3.53.111)

Node: nef07
Resolution: Service “HA” is not in a broken state (currently stopped)
Broken: no
Fixed: no
Split brain: no
State: stopped
Suggested action: none
Errors: no errors available
Online resources: no online resources available
```
Offline resources:
c0t5000C500056366407d0s0 not reserved
c0t5000C50000FA601B7d0s0 not reserved
c0t5000C5000FA53FFBd0s0 not reserved
c0t5000C500563648C3d0s0 not reserved
ZFS pool tank
VIP vip01 (10.3.53.111)

**Repair Broken Service**

**Using “haservice recover”**

Use this section if the proposed suggested repair action was to use “haservice recover” command as in the above example and for all manual offline / manual online operations.

Note: haservice recover runs a sanity check before actually transitioning a service to the stopped (--offline), or running (--online) state. In other words, it does not trigger a broken service to the stopped state by DEFAULT. For e.g., it runs a sanity check to verify if the node still has the service’s pool still imported before transitioning the service.

This section covers the steps to repair a broken service that is in broken_safe, broken_unsafe on a particular node. Using this command you have an option to transition the service to either stopped (--offline), or running (--online) state as follows:

1. **List the HA service running on the nodes (nef06 and nef07).**
   
   CLI@nef06> haservice list
   
   NAME   GUID   DESCRIPTION VIPs      NODES            RUNNING  STOPPED   BROKEN
   HA     1516   -                 nef06,nef07         nef06,nef07

2. **You can trigger the service to go online/offline on a specific node. The following example sets the HA service to offline on node ‘nef06’ as proposed in the “Suggested repair action”**.

   CLI@nef06> haservice recover -yv --offline HA nef06
   
   Setting service 'HA' offline on node 'nef06'

3. **Verify if the service is stopped on nef06.**

   CLI@nef06> haservice list
   
   NAME   GUID   DESCRIPTION VIPs      NODES            RUNNING  STOPPED   BROKEN
   HA     1516   -                 nef06,nef07         nef06,nef07
Advanced Configurations and Operations

Add Disks or Network Interfaces to the Heartbeat Configuration

Once the cluster is configured, the nodes in the HA cluster constantly monitor the state and status of the other appliance in the cluster using a heartbeat mechanism. Heartbeats are used to ascertain that a cluster node has failed before the fallback node takes over its services.

NexentaStor provides several heartbeat channels to communicate between the nodes in the cluster. Each node in the Cluster communicates with the other node using a shared disk or a network interface. HA cluster detects a system failure when no updates are received from all heartbeat mechanisms for a specified time interval. To improve reliability, you can add more disks or network interface to the heartbeat configuration using the following commands.

**Adding Disk Heartbeat**

```bash
CLI@nexenta> hacluster add-disk-heartbeat [-nyv] <first-node> <second-node> <service> <disk>
```

Example:

```bash
CLI@nexenta> hacluster add-disk-heartbeat nef06 nef07 democluster <disk>
```

where democluster is the HA service running on the node where the disk resides.

**Adding Private Network Heartbeats to an Existing Cluster**

If a network interface fails on any one of the nodes, you can add a second interface (in the example shown below, nef06-priv2, nef07-priv2) to the heartbeat configuration to maintain the heartbeat communication. To add a network heartbeat after the cluster is created, Nexenta recommends using a private dedicated network connection available between the nodes.

1. On each node add the private end point addresses (as they are not discoverable on the LAN) - assuming here the local end points have already been created (i.e. on nef06 nef06-priv2 has been created and assigned to an interface, and for nef07-priv2 on nef07):

   On nef06:
   ```bash
   CLI@nef06> net create host 10.10.10.20 nef07-priv2
   ```
   and on nef07:
   ```bash
   CLI@nef07> net create host 10.10.10.10 nef06-priv2
   ```

2. Add the network heartbeat to the existing cluster.

   ```bash
   CLI@nexenta> hacluster add-net-heartbeat <first-node> <first-ip> <second-node> <second-ip>
   ```
   Example:
   ```bash
   CLI@nef06> hacluster add-net-heartbeat nef06 nef06-priv2 nef07 nef07-priv2
   ```
3. Verify the status of the heartbeats. Use “–e” option along with the “hacluster status” command to get a detailed output of the heartbeats.

```
CLI@nef06> hacluster status –e
== Heartbeats ==
<table>
<thead>
<tr>
<th>ID</th>
<th>TYPE</th>
<th>FROM</th>
<th>TO</th>
<th>PEER ADDRESS</th>
<th>STATUS</th>
<th>POOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>net</td>
<td>nef06</td>
<td>nef07</td>
<td>nef07</td>
<td>up</td>
<td>–</td>
</tr>
<tr>
<td>1</td>
<td>net</td>
<td>nef06</td>
<td>nef07</td>
<td>nef07-priv2</td>
<td>up</td>
<td>–</td>
</tr>
<tr>
<td>2</td>
<td>net</td>
<td>nef06</td>
<td>nef07</td>
<td>nef07-priv</td>
<td>up</td>
<td>–</td>
</tr>
<tr>
<td>3</td>
<td>net</td>
<td>nef07</td>
<td>nef06</td>
<td>nef06-priv</td>
<td>up</td>
<td>–</td>
</tr>
<tr>
<td>4</td>
<td>net</td>
<td>nef07</td>
<td>nef06</td>
<td>nef06</td>
<td>up</td>
<td>–</td>
</tr>
<tr>
<td>5</td>
<td>net</td>
<td>nef07</td>
<td>nef06</td>
<td>nef06-priv2</td>
<td>up</td>
<td>–</td>
</tr>
</tbody>
</table>
```

**REMOVING HEARTBEAT**

To remove a node from the cluster or to change the heartbeat disk or the NIC, you must delete the heartbeat.

```
CLI@nexenta> hacluster delete-heartbeat <id>
```

**Add Pools to an HA service**

You can also add a second pool to the HA service you created. When you add a second pool to the HA service, both pools are now associated with the same `hapool` service and will failover together as members of the cluster.

```
CLI@nexenta> haservice add-pool <service> <pool> <guid>
```

**Example:**

```
CLI@nexenta> haservice add-pool hapool smallHa <guid>
```

**Update a HA Service**

Use this section as guidelines to add/update/remove a VIP(s) from HA service without affecting the HA service.

**ADDING A VIP TO AN EXISTING SERVICE**

To add a VIP to an existing service, run the following command from the node that runs the HA service.

```
CLI@nexenta> haservice add-vip hapool vip05 9.8.7.6/255.0.0.0 nef06:e1000g0,nef07:e1000g0
```

VIP has the following structure:

```
Vipname ipaddress[/mask]node1:linkname,node2:linkname
```

In the above example, the VIP `vip05` with IP address `9.8.7.6` and netmask `255.0.0.0` are added to the HA service named `hapool`.

**REMOVING VIP FROM AN EXISTING SERVICE**

```
CLI@nexenta> haservice delete-vip hapool vip05
```
**UPDATING DEFAULT NETMASK, VIP, NICs FROM AN EXISTING SERVICE**

You can reconfigure the VIP address, netmask, NICs using the following command:

```
CLI@nexenta> haservice update-vip [-nv] [-a <address>] [-N <nics>] <service> <vip>
```

**Example:**

You provided a network mask as part of the VIP information when you created the HA service. You can modify this default netmask value according to your configuration needs at any time. The command below reconfigures the VIP address in a clustered service hapool.

The following command changes VIP address for 'vip05' from 9.8.7.6 to 10.3.120.170 in the HA service 'hapool'.

```
CLI@nexenta> haservice update-vip -a 10.3.120.170/255.255.255.0 -N node1:linkname,node2:linkname hapool vip05
```

**Fast-Failover**

A normal, manual failover operation takes care to effectively fail over a single service (or set of services) from one Cluster Node to the other. This will export pools from source node and import on destination node. In the case where you have a single pool, or you are failing over all pools (services) and wish to do so quickly, you can use the “Fast Failover” mechanism. When a fast fail over is invoked, what ultimately happens is the machine currently running the service you want to “fast fail over” is halted immediately (thereby forcing a fail over - and it's "fast" because 1) there is no waiting for service shutdown/pool export on the node currently running that service and 2) because the node taking over the service knows the remote node is about to halt it does not need to invoke the wait time to decide the other node has failed - it will start the service immediately).

Invoke “fast-failover” from the node that is currently running the service.

```
CLI@node> hacluster fast-failover (--halt | --reboot | --poweroff) <name of the target node where the service must failover>
```

When you run this command as part of the fast-failover process, the pool does not get exported. This is achieved by halting the node that is currently running the service to avoid the pool export time.

When you execute this command, you can choose to invoke certain actions on the halted-node after it will be halted for a certain period.

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--halt</td>
<td>stop the appliance without powering it down/rebooting</td>
</tr>
<tr>
<td>--reboot</td>
<td>to reboot the appliance</td>
</tr>
<tr>
<td>--poweroff</td>
<td>to power down the appliance</td>
</tr>
</tbody>
</table>
Replace a Clustered Node

The hacluster replace feature is useful when you need to physically replace a node that is part of an HA cluster with a new node without going through a full cluster rebuild. If the node being replaced will be reused later on the same network, you should make sure that this node is up and running when you run the hacluster replace command. This will ensure that the replaced node is properly decommissioned from the cluster. The hacluster replace command can also be run to replace a node that is down or has already been physically removed from the NexentaStor cluster configuration:

- If the node to be replaced is up and still part of the cluster then the replace operation will effectively deconfigure HA on that node. This operation resets clustering back to a bootstrap mode and removes all state information. However, if the node being replaced is down then the deconfigure step is skipped but completes the replace node operation.

- If a node was down when it was replaced and is then subsequently powered back on (with the implication that it will start sending heartbeats to its original peer node) then the cluster will detect this and lock down all services. In other words, they will continue running but failovers will be disabled (the original node from the cluster will see two sets of heartbeats, one from the new replaced node and one set from the old node - it will detect this conflicting set of heartbeats and as a result block all service fail over as a safety measure).

Once you physically replace a clustered node with a new server, use this hacluster replace command for the new server to join the existing cluster. For example, if you have a cluster consisting of nodes nodea and nodeb, and you replaced nodeb with nodenew, use this replace feature so that the new node nodenew joins the existing cluster.

For the new node nodenew to join the existing cluster, do the following:

1. Update the hosts on both nodes to ensure that they resolve to each other. On the first node nodea, run the following command.

   CLI@nodea> net create host [ -nv] <IP address of the new node> nodenew

2. Run the above command on nodenew to add nodea.

3. Now run the replace command on the other existing cluster node, nodea in this example.

   CLI@nodea> hacluster replace [ -nyv] nodeb nodenew

Arguments:
nodeb      Name of the existing node to be replaced.
nodenew    New node to replace the existing one.

Fail Back

Once the replacement is complete and the cluster is fully operational, you as an administrator can control where the pools get redistributed. For example, if the old-original node had pools/haservices on it and has now failed over to the other node, you may move them to the new node to restore the original configuration.

   CLI@nodea> haservice move <service> nodenew
   This sets the cluster back in its original configuration.

Note: When you physically replaced a node with a new server and you want to destroy the existing cluster on the surviving node and configure a new cluster between the surviving node and the new server, use hacluster reinitialize with caution. This command should be run only with the assistance of Nexenta Support.
Troubleshooting

This chapter includes tips and tools to help in troubleshooting common problems.

An Expired HA License

When the temporary HA license expires, the “system summary status” will return the node name as “Unlicensed_Cluster_Node”.

In order to re-license and to recover an HA cluster, you must do the following:

- Contact Nexenta sales and request an extension to the trial license.
- Apply the new license on the node where the license expired.
- Restart the HA service so that the cluster reloads the config file and re-joins the cluster. Run the “restart” on the node where the license expired.

```
CLI@nexenta> svc restart ha
```

- Verify if the cluster is back on by running hacluster status.

```
CLI@nexenta> hacluster status
```

Diagnose Unsuccessful HA Cluster Creation or Cluster Start

- Check if the HA feature is enabled on the clustered nodes using the `license show` command. If not enabled, obtain an HA license token from sales@nexenta.com then run the following command:

```
CLI@nexenta> license activate <Activation Token>
```

- Verify if both the nodes resolve to each other by running the following command on both the nodes.

```
CLI@nexenta> net list host
```

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>HOSTNAME</th>
<th>ALIAS</th>
<th>PROTOCOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>::1</td>
<td>localhost</td>
<td>-</td>
<td>ipv6</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>localhost</td>
<td>loghost</td>
<td>ipv4</td>
</tr>
<tr>
<td>10.3.53.109</td>
<td>nef06</td>
<td>-</td>
<td>ipv4</td>
</tr>
<tr>
<td>10.3.53.110</td>
<td>nef07</td>
<td>-</td>
<td>ipv4</td>
</tr>
</tbody>
</table>

If it is not mapped correctly, update the hosts on both the nodes to ensure that they resolve to each other. On the first node, run the following command.

```
CLI@nexenta> net create host [-nv] <IP address of the second node> <hostname of the second node>
```

Run the above command on the second node to update the hosts. Verify that the new entry was added correctly using the following command:

```
CLI@nexenta> net list host
```

You should be able to see the IP address of each other.

- Check the state of the HA services. See Table 2 for the list of possible service states.

```
CLI@nexenta> haservice status – to see status of all HA services
CLI@nexenta> haservice status <HA service name> – to see status of a specific HA service
```
Verify Virtual IP Address

In the example below, the `hacluster check-vip` command verifies the VIP `vip01` is consistent, or resolves across all the nodes in the cluster.

```
CLI@nexenta> hacluster check-vip vip01
```

Command output:
```
VIP Name  CONSISTENT
vip01     yes
```

View HA Service Status

You can view the status information on all HA services as shown by the example below. You can also query the status of a specific HA service.

For the complete list of HA service status values, refer to Table 2.

```
CLI@nexenta> haservice status
```

Command output:
```
status:
NODE    STATUS   MODE       UNBLOCKED
nef06   running  automatic  yes
nef07   stopped  manual     yes

pools:
NAME    GUID                PRIMARY
hapool  575415499326140623  yes

vips:
NAME     ADDRESS            IPv6  NODE    NIC
zvip     10.3.53.110/24     no    nef06   e1000g0
nef07    e1000g0
TESTVIP  10.10.10.10/16     no    nef06   e1000g0
Nef07    e1000g0

  init timeout:  20
  run timeout:   8
```

View RSF Logs

To view the information on why an HA service goes into a faulted state, consult the RSF logs using the `journal show` command.

First list all the available journals.

```
CLI@nexenta> journal list
```

```
NAME                 SIZE    UPDATED                   ARCHIVES
install/caiman       11.7K   Wed Dec 21 13:45:34 2016  0
install/messages     79.9K   Wed Dec 21 13:42:09 2016  0
```
Now print all the rsf/rsfmon journal.

```
CLI@nexenta> journal show rsf/rsfmon
```

### Recovering Lost Cluster Configuration

When you upgrade, if the cluster configuration is lost as shown in the examples below, run the “hacluster recover” command to recover the cluster settings.

```
CLI@nexenta> haservice list
No cluster defined
CLI@nexenta> hacluster status
No cluster defined

To recover the cluster configuration, run the following command.

```
CLI@nexenta> hacluster recover
Recover cluster database? [y/N] y
```

### Limitations for NFS v4 Clients

NFS Version 4 has removed support for UDP as an underlying transport protocol (as opposed to V3 which supported both UDP and TCP), therefore all NFSV4 connections are TCP based. This exclusive use of TCP in NFSv4 has implications for failover recovery time in certain scenarios due to TCP's TIME_WAIT (sometimes referred to as 2MSL) state that can be entered into during multiple failover operations.

The reason for a TCP socket to enter a TIME_WAIT state is to prevent delayed data packets from one connection being misinterpreted as part of a subsequent newly established connection on the same machine (applying stale data from a previous connection to the current one could have potentially disastrous effects on data integrity).

The implication of TIME_WAIT for failover is observed when HA services are moved from one node to another and then back again in a short period of time. Once the initial move is complete the originating server enters the TIME_WAIT state as part of the normal TCP protocol. If during this 'wait period' services are moved back to the originating server clients will be unable to re-connect until the TIME_WAIT period has expired (and in some cases the client connections will timeout themselves).

### Recommendations to Avoid Outage

Manual moves back and forth in quick succession (circa 2-4 minutes) between machines that provide data over NFSv4 should be avoided.
For machines where the failover was instigated as a result of a systems crash, TIME_WAIT is irrelevant as the TCP connections will have no knowledge of the previous connection.
2 High-Availability for NexentaStor VSA on VMware

Overview

NexentaStor High Availability is supported between NexentaStor Virtual Machines on VMware vSphere. This configuration is popular for customers looking to add full featured NAS file services to VMware vSphere environments on all-block SANs or Hyper-Converged Infrastructure such as VMware Virtual SAN and Nutanix. A NexentaStor HA cluster on VMware vSphere relies on two functions to provide reliable automated failover services:

1. **Heartbeats**: Each node in the cluster communicates with the other node through different heartbeat mechanisms such as Network and disk heartbeats. These heartbeats are used to monitor the status of the remote node. As long as a remote node responds to network or disk heartbeats, it is deemed operational. If a remote node stops responding to heartbeats for a specific time period, the HA cluster detects system failure and an automatic failover operation is initiated.

2. **vCenter control point**: vCenter is used to control the power state of NexentaStor virtual machines to protect against split brain scenarios and ensure that a storage pool is only imported on a single node at any time. This is used as an alternative to SCSI persistent reservation used on bare metal clusters.

When NexentaStor is deployed on VMware vSphere, storage pools are built on vmdks from the underlying VMware datastores. While these vmdks support the standard disk heartbeat mechanism used by NexentaStor HA, they generally do not support the SCSI disk reservation normally required to effectively protect data in the event of split brain scenarios. As an alternative, NexentaStor HA on VMware vSphere integrates with vCenter to control the state of NexentaStor virtual machines: when a NexentaStor virtual machine detects a fault on the
remote NexentaStor instance and decides to take over the pool and its services, it connects to vCenter and has it power off the remote virtual machine. Then it proceeds to import the pool and failover services.

In this configuration, NexentaStor virtual machines are configured with a vCenter account that has permissions to power off the two NexentaStor virtual machines. NexentaStor continuously monitors the ability to connect and login to vCenter. If a NexentaStor virtual machine stops getting heartbeats from the remote node and finds it cannot connect to vCenter, it goes to a broken-site-safe state and does nothing, deferring to the administrator to come in and figure out actual status of services and virtual machines.

Following a failover event, the non-responding NexentaStor virtual machine is powered off and all services should be running on the surviving NexentaStor node. It is up to the administrator to manual restart the failed NexentaStor virtual machine and restore cluster services.

The following sections list the configurations that should be executed on the VMware side and on the NexentaStor side to set up NexentaStor HA on VMware. However, these sections only provide examples of the standard NexentaStor CLI commands that should be executed on the NexentaStor Virtual Machines. For details on the configurations on the VMware side, please refer to the corresponding VMware documentation.

**NexentaStor VSA HA Cluster Setup**

Set up the configurations in the order they are listed here.

1. Deploy NexentaStor VSAs on VMware as described in NexentaStor Installation Guide.
2. Configure VM/Host affinity rules on VMware to make sure the NexentaStor VMs are kept on different physical ESXi hosts for better redundancy.
   As best practice, to prevent the VM from being suspended, disable the suspend feature on the advanced options in the VM by setting “suspend.disabled=TRUE”.
3. Create NexentaHA User in vCenter that will be used to force power OFF Nexenta VMs during failover operations.
4. Setup permission for the NexentaHA User.
5. Configure NexentaStor HA cluster.
   Setup the HA Cluster before you configure the vCenter access. This is required so the vCenter server info is provided only in one of the clustered nodes and HA cluster will replicate the info to both nodes.
6. Create vCenter host on the NexentaStor VSAs.
   Add the IP address of the vCenter host to the Host table on both nodes to be used for the HA cluster before you setup the vCenter access so that you can use the host information when you configure the vCenter access. Use host name when you configure the vCenter access so if you change the IP address of the vCenter server later, you will have to only update that vCenter host on NS nodes, instead of having to update the HA configuration.
7. Configure vCenter access to establish the connection between the NexentaStor VSAs and the vCenter host.

For details on the above listed configurations, refer to the sections drafted below.

**Configurations to be done on VMware Side**

The following sections use the listed name conventions as examples:

- ha cluster name: democluster
- 1st node name in a cluster: NS1
- 2nd node name in a cluster: NS2
Configure VM/Host Rules
It is generally recommended to keep the two NexentaStor VMs of an HA cluster running on different ESXi hosts. When vSphere DRS is enabled, you should configure specific host rules to make sure the NexentaStor VMs are kept on different hosts.

As best practice, to prevent the VM from being suspended, disable the suspend feature on the advanced options in the VM by setting "suspend.disabled=TRUE".

See VMware documentation for setting up the rules for NS1 and NS2 to run on different ESX hosts.

Create NexentaHA User in vCenter
Since NexentaStor HA on VMware does not support SCSI reservations, you must create a NexentaHA user to power OFF the remote failed node before the other node takes over a service, in the event of a NexentaStor VM failure.

Set Permissions for the NexentaHA User
The high-availability clustering logic uses a special "NexentaHA" user in vCenter to control the state of the NexentaStor Virtual Machines. You will need to create this "NexentaHA" user and give it permissions to "power OFF" the NexentaStor Virtual Machines in your NexentaStor cluster.

See VMware Documentation for creating the vCenter User and for assigning permission to the User.

Configurations to be done on NexentaStor VMs

Configure NexentaStor HA Cluster
The cluster creation process involves the following tasks:

1. Verify that the HA feature is licensed and enabled on both NexentaStor appliances in the cluster.
2. Update the host table on both the nodes so that the cluster nodes can resolve the IP address(es) of it's partner without relying on DNS services.
3. Verify that the system day and time settings on each of the cluster member nodes are in sync.
4. Create a new cluster.
   CLI@NS1> hacluster create NS1, NS2 democluster

   The example shown here creates a cluster named democluster using the two nodes NS1 and NS2 as hosts.
5. Verify the cluster status.
   CLI@NS1> hacluster status

For details on the above steps, see Configuring Cluster from the previous chapter.

Update the Host Table on Both Nodes to Add the IP Address of the vCenter Server

1. On the first node in a cluster environment, update the host to add the IP Address of the vCenter Server.
   CLI@NS1> net create host <IP address of the vCenter server> <host name of the vCenter server>
Example:

```
CLI@NS1> net create host x.x.x.x vcenter
```

2. Do the equivalent on the second node.

```
CLI@NS2> net create host x.x.x.x vcenter
```

3. Verify if they resolve to each other by running the following command on both the nodes.

```
CLI@NS1> net list host
```

**Example:**

```
ADDRESS      HOSTNAME    ALIAS    PROTOCOL
::1          localhost   -        ipv6
127.0.0.1    localhost   loghost  ipv4
x.x.x.x      vcenter     -        ipv4
```

Configure vCenter Access from the Clustered Nodes

This step can be run on any node of the NexentaStor HA cluster. The clustering service will take care of replicating the configuration info between nodes.

You will need to use the vCenter hostname (vCenter in this example) and the vCenter API port (443 by default), along with the vCenter user name (NexentaHA in this example) that you created in the previous steps.

```
CLI@NS1> hactl vcenter-configure vcenter:443 -u rsf@vsphere.hac -p 'rsfPa$$w0rd'
```

Configure vCenter? [y/N] y

**Verify the Connections to the vCenter Server**

You can then use the following command to test the connection between the vCenter server and the NexentaStor clustered nodes (NS1 and NS2).

```
CLI@NS1> hactl vcenter-show
```

**Example:**

```
SERVER    PORT  USER             STATUS
vcenter   443   rsf@vsphere.hac  online
```

Create HA Service

A HA service provides orchestrated failover and failback of all file and block services supported by one or more NexentaStor storage pool. A HA service is typically configured on top of one NexentaStor storage pool.

For information on creating a HA Service, see ConfiguringHAService.

### Failover Operations

### Additional Considerations for HA Operations in VMware

- In the case of a manual/soft failover the HA cluster does not invoke the vCenter server to Power OFF the remote NexentaStor VM.

  **Example of a Manual failover:**

  CLI@nexenta> haservice move <service> <node>

  **Example:**
CLI@NS1> haservice move tank1 NS2

- Whereas in the case of automatic/hard failover, as a result of losing contact with the other node, the HA cluster invokes the vCenter server to Power OFF the non-responding remote node.
- If the connection to the vCenter server is not available during an automatic/hard failover the RSF will mark the service as broken-site-safe because the cluster cannot invoke the vCenter to Power off the failed node. In this case you must override the broken-site-safe state once you manually power OFF the remote failed node.
  
  **See “Manually Override Broken-Site-Safe State” section for steps to override the broken-site-safe after the remote node is powered OFF.**
- If vCenter is unable to contact either the VM or the ESX node hosting the VM during an automatic/hard failover, then the VM Power OFF procedure will be invoked.

**Example of an Automatic Failover Scenario**

The following examples show one HA service tank1 on NS1 and two HA services tank2, tank3 added to the clustered node NS2. For more details on the prerequisites for configuring a HA service and the steps to add a HA service, see Configuring HA Services.

1. **Create HA Service**
   CLI@NS1> haservice create tank1
   CLI@NS2> haservice create tank2
   CLI@NS2> haservice create tank3

2. **List the HA Services you created.**
   CLI@NS1> haservice list
   
<table>
<thead>
<tr>
<th>NAME</th>
<th>GUID</th>
<th>DESCRIPTION</th>
<th>VIPs</th>
<th>NODES</th>
<th>RUNNING</th>
<th>STOPPED</th>
<th>BROKEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank1</td>
<td>53858558</td>
<td>No description</td>
<td>NS1,NS2</td>
<td>NS1</td>
<td>NS2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tank2</td>
<td>14705185</td>
<td>No description</td>
<td>NS1,NS2</td>
<td>NS2</td>
<td>NS1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tank3</td>
<td>86228049</td>
<td>No description</td>
<td>NS1,NS2</td>
<td>NS2</td>
<td>NS1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Now the cluster behaves like any other cluster on a bare-metal, except that SCSI reservations are disabled.

**Node (NS2) Failing Scenario**

Now that you have setup the HA service to the cluster, in the event of either node failing the surviving node will take over the HA service for the pool once the failed node is powered OFF automatically by the vCenter server. The following sections show how the HA service (tank2 and tank3) failed over to the node (NS1) when the node (NS2) failed.

**Note: As services fail over to the other node, the HA cluster will invoke the vCenter server to power off the failed node.**

1. **Status of the HA service before the Node NS2 fails.**
   CLI@NS1> haservice status
   
   **Service: tank1**
   
<table>
<thead>
<tr>
<th>NODE</th>
<th>STATUS</th>
<th>MODE</th>
<th>UNBLOCKED</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS1</td>
<td>running</td>
<td>automatic</td>
<td>yes</td>
</tr>
<tr>
<td>NS2</td>
<td>stopped</td>
<td>automatic</td>
<td>yes</td>
</tr>
</tbody>
</table>

   **Service: tank2**
   
<table>
<thead>
<tr>
<th>NODE</th>
<th>STATUS</th>
<th>MODE</th>
<th>UNBLOCKED</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS1</td>
<td>stopped</td>
<td>automatic</td>
<td>yes</td>
</tr>
<tr>
<td>NS2</td>
<td>running</td>
<td>automatic</td>
<td>yes</td>
</tr>
</tbody>
</table>

   **Service: tank3**
   
<table>
<thead>
<tr>
<th>NODE</th>
<th>STATUS</th>
<th>MODE</th>
<th>UNBLOCKED</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS1</td>
<td>stopped</td>
<td>automatic</td>
<td>yes</td>
</tr>
</tbody>
</table>
NS2      running  automatic  yes

From the above example you notice that the service tank2 and tank3 are running on node NS2.

2. To simulate a failed node, the example here shows the NexentaStor VM (NS2) that has the HA service (tank2 and tank3) is powered OFF manually.
   CLI@NS2> poweroff

View the Status of the HA Service

3. Check the status of the HA service from the surviving node NS1.
   CLI@NS1> haservice status

   Service: tank1
   NODE  STATUS     MODE  UNBLOCKED
   NS1   running  automatic  yes

   Service: tank2
   NODE  STATUS     MODE  UNBLOCKED
   NS1   starting automatic  yes

   Service: tank3
   NODE  STATUS     MODE  UNBLOCKED
   NS1   starting automatic  yes

   Note: Notice that there is no output service instances on NS2 since it is now offline.

4. Now check the status of the HA cluster from the surviving node.

5. From NS2 verify that the pool imported to it.
   CLI@NS2> pool list

   NAME   SIZE   ALLOC   FREE   AVAIL   DEDUP  EXPANDSZ  FRAG  HEALTH

Verify the Remote Failed Node is Powered OFF

   CLI@NS1> hacluster status

   == Nodes ==
   NODE  STATUS   SERVICES  ADDRESS      HostId    Release
   NS1   up       3/3       10.3.53.109  1fb4ba7c  3.12.0
   NS2   down     0/3       10.3.53.110  808b556b  3.12.0

   Note: In this case, NS2 was powered off when the HA service tank2 failed over to the surviving node. So for tank3, the cluster could not power OFF the remote node, because it was already powered off. So it continues with the service startup.
Cluster Recovery after a Hard Failover

A failover triggered by an actual failure will result in one NexentaStor VM being powered off and all services running on the surviving NexentaStor VM. To recover the cluster you will need to manually power on the failed NexentaStor VM using the vSphere Client.

Troubleshooting

Scenario 1: Unavailability of vCenter Server

If vCenter is unavailable when the cluster attempts to power off the remote node, then the services go into the broken_site_safe state.

```
CLI@NS1> haservice status
tank1    running            auto   unblocked    NONE    NONE    20  8
tank3    broken_site_safe   auto   unblocked    NONE    NONE    20  8
tank2    broken_site_safe   auto   unblocked    NONE    NONE    20  8
```

Manually Override Broken-Site-Safe State

In this case, you need to make sure the other node (NS2) is powered off before setting the broken_site_safe override for each service:

```
CLI@NS1> haservice override --set tank2
set broken-site-safe override for service 'tank2' ? [y/N] y
root@guest-ns1:~# haservice override --set tank3
set broken-site-safe override for service 'tank3' ? [y/N] y
```

```
CLI@NS1> haservice status
tank1    running   auto   unblocked    NONE    NONE    20  8
tank3    running   auto   unblocked    NONE    NONE    20  8
tank2    running   auto   unblocked    NONE    NONE    20  8
```

Scenario 2: Update vCenter Configuration

To update the vCenter configuration both nodes need to be UP in order to change the settings.

1. Updating the vCenter IP address requires simply updating the vCenter host on both NexentaStor VSAs as shown in the section Update the Host Table.

2. However, updating the Nexenta HA User or changing the password requires you to update the vCenter configuration from the following command on one of the nodes. When the change is made on one of the nodes, the HA cluster will propagate the changes to the other node in the cluster.

   ```
   CLI@NS1> hactl vcenter-update [-vyn] <server> -u <username> -p
   ```
<password>

Update vCenter configuration? [y/N] y