nexenta®

Global Leader in Software-Defined Storage.

NexentaStor 5.1.1 HA CLI
QuickStart Guide

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 5.1.1 HA cluster using the NexentaStor 5.1.1 Command Line Interface (CLI). For additional user documentation, see https://nexenta.com/products/documentation.

Document History

Revision	Description
January, 2018	5.1.1 GA version

Table of Contents

About NexentaStor HA

An HA cluster 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 must be built using shared storage devices that are accessible from both the nodes. The high availability of the pools is maintained by the cluster software that manages the startup and failover of the HA services within the cluster.

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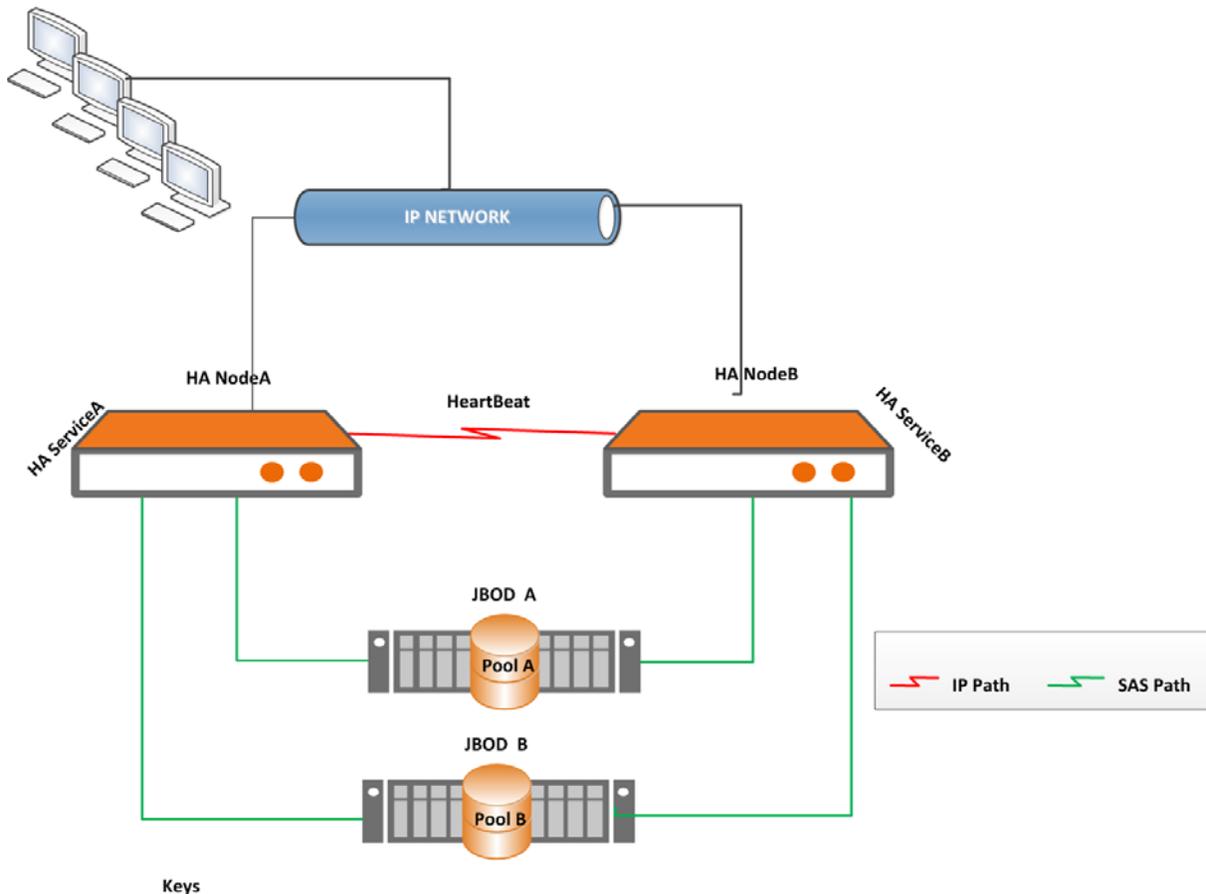
- Pools should be built using shared disks.
- Pools must be explicitly imported on the node that handles HA service creation request.

When the failed node is repaired and restarted, it rejoins the cluster and the administrator can control when the redistribution of the services.

Each node in the cluster communicates with the other node through different heartbeat mechanisms. The HA cluster detects system failure when no heartbeat updates from a node have been seen across all mechanisms for a specified time period.

Definition of Terms

The terms in Table 1 are used throughout this document.

Table 1: Definition of Terms

Term	Definition
HA Cluster	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. `hacluster` lists all the HA cluster CLI subcommands. `hacluster <subcommand name> -h` lists the available options for the subcommand.
HA Service	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. `haservice` lists all the HA service CLI subcommands. `haservice <subcommand name> -h` lists the available options for the subcommand.

An HA cluster:
- Is a pair of NexentaStor appliances with the HA feature licensed on both nodes.
- Runs a defined set of services and monitors each cluster node member for failures. These NexentaStor appliances are connected through various communication channels and exchange heartbeats that provide information about their states.
- Can have one or more multiple pools per HA service.
- Includes a built-in monitoring mechanism that detects a failure in the interface (for example, public network interface or Fibre Channel interface) between the clients and the datasets that can trigger an automatic failover to the other node in the cluster.

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

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

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

For a successful rollout of HA services, ensure that the following items are in place:
- License token for the HA feature is available and activated for each node in the cluster.
- Static IP address for each node in a cluster is known.
- Shared physical storage across 2 nodes in a cluster exists.
- Private network connection established via a network patch cable using two unused ethernet ports between the heads is recommended.

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
type            ENTERPRISE
product         NexentaStor
version         5.1.1
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
capacity        no limit
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:

```
ADDRESS         HOSTNAME     ALIAS     PROTOCOL
::1             localhost    -         ipv6
127.0.0.1       localhost    loghost   ipv4
10.3.53.109     nef06        -         ipv4
10.3.53.110     nef07        -         ipv4
```

Enable the HA Service

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

```
CLI@nef06> svc list ha

NAME   DESCRIPTION         STATE
ha     HA cluster service  online
```

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
NODE      ADDRESS          CRC        STARTUP TIME
nef07     10.3.53.110      ba52       2016-03-24T16:49:44
nef06     10.3.53.109      ba52       2016-03-24T16:49:43
```

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
NAME            FLAGS   VALUE
system.date     --      Fri Mar 31 08:58:11 2017
```

To set the system date and time:

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

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.

1. Verify if there is any cluster already defined.

```
CLI@nef06> hacluster status
No cluster defined
```

2. 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".

3. Check status 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   5.1.1
nef07  up       0/0      10.3.53.110   778f4c71   5.1.1

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

Note: An entry for the remote node will be automatically added to /etc/hosts if it did not exist before.

WITH ONLY PRIVATE NETWORK HEARTBEATS

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.

   ```
   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   5.1.1
nef07 up         0/0       10.3.53.110   778f4c71   5.1.1

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

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
    ```

    ```
    == 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   5.1.1
    nef07 up         0/0       10.3.53.110   778f4c71   5.1.1

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

Advanced Configuration

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

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

Example:

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.

```
CLI@nexenta> hacluster add-net-heartbeat <first-node> <first-ip> <second-
node> <second-ip>

Example:

    CLI@nef06> hacluster add-net-heartbeat nef06 nef06-priv2 nef07 nef07-
    priv2
    CLI@nef06> hacluster status -e
    == Heartbeats ==
    ID   TYPE   FROM    TO      PEER ADDRESS   STATUS   POOL
    0    net    nef06   nef07   nef07          up       -
    1    net    nef06   nef07   nef07-priv2    up       -
    2    net    nef06   nef07   nef07-priv     up       -
    3    net    nef07   nef06   nef06-priv     up       -
    4    net    nef07   nef06   nef06          up       -
    5    net    nef07   nef06   nef06-priv2    up       -
```

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>
```

Manage Cluster

You can manage the cluster using the `hacluster` subcommands in Table 4.

Table 4: Managing an HA Cluster

Cluster Tasks	Description	Command
Reset	Removes the cluster service without destroying it.	`CLI@nexenta> hacluster reset <cluster>`
Remove/destroy cluster	Once you remove a cluster, all clustered pools remain imported. If any cluster services are not in an Exported or Available state, they will need to be repaired before this command will succeed. The administrator must explicitly repair the broken service first, and then destroy the cluster.	To destroy a cluster: `CLI@nexenta> hacluster destroy <cluster>`
Remove a shared pool from the cluster	Before removing a shared pool from a cluster, make sure you remove the HA service associated with that pool using the `haservice delete-pool` command. Otherwise, the `poolremove` or `pool export` command will fail.	`CLI@nexenta> pool remove [-nv] <pool> <disk>` after removing the HA service which hosts the pool.
Import a shared pool	To put a shared pool under cluster control, you must import it.	`CLI@nexenta> pool import [-nvD] [-s paths] [-c <cache-file>] [-R altroot] [-o <properties>] <pool> [<new-name>]`
Find the list of exported pools	The `pool import` subcommand with no argument yields the list of exported pools.	`CLI@nexenta> pool import`

Configuring HA Services

Prerequisites

Before creating the HA service after the cluster is created, complete the following tasks:

- Verify that the cluster exists. See the previous chapter on how to create a cluster.
- Create a new pool or import an existing pool using devices that reside on shared storage.
- Validate that the shared pool is visible to both nodes and eligible to be placed under cluster control.
- Discover unclustered pools that are available to become members of a cluster.
- HA services can be created only from imported pools. Pools must be explicitly imported on the node that initiated the service creation request.
- The HA service name must be identical to the pool name and must contain only alpha-numeric characters.
       See the *NexentaStor 5.1.1 CLI Configuration QuickStart* for instructions on how to create and import a pool.
- After you have completed all these configuration steps, proceed with creating an HA service for the shared pool. You do not have to explicitly add a pool to the HA service. You can create an HA service with zero to any number of virtual IP addresses (VIPs).
- Optionally, add more disks or network interfaces to the heartbeat configuration after the service is configured.

       Note: You don't require VIP for FibreChannel (FC) failover.

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. 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.

You can create an HA service with or without virtual IP addresses (VIPs). Nexenta recommends to create VIP(s) for nfs, smb shares and iSCSI LUNs for successful failover of the shares in case of a disaster.

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:** The HA service name must match the pool name, a required argument to the command.

The example above configures the following:
- Using the −m flag, the node nef06 is designated as the main or local server from which the HA service hapool should run.
- 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    -
```

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 the 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>
```

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.

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
nef07       broken_unsafe    automatic   yes
==pools==
NAME     GUID                   PRIMARY
hapool   13660110209781822772   yes
==VIP==
NAME     ADDRESS                    IPv6   NODE     NIC
vip01    10.3.53.111/255.255.255.0  no     nef06    e1000g0
                                           nef07    e1000g0
init timeout:   20
run timeout:    8
disk heartbeats: not available
==SCSI reservations==
NODE        DISK      TYPE
universe    c2t2d0    SCSI2
```

Update Default Netmask and VIP

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`.

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

Example:

```
CLI@nexenta> haservice update-vip -a 1.2.3.4/255.0.0.0 hapool vip05
```

Preparing FC and iSCSI for Failover

Table 3 below provides a checklist that can be used to prepare for a successful iSCSI and FC volume failovers. 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 QuickStart* at https://nexenta.com/products/documentation.

Table 3: FC and iSCSI Failover Checklist

Checklist Items	Related FC CLI Commands	Related iSCSI CLI Commands
Configure targets	`fctarget`	`iscsitarget, iscsiauth`
Configure target groups	`targetgroup`	`targetgroup`
Map LUNs	`Lunmapping, logicalunit`	`Lunmapping, logicalunit`
Verify that you can mount volume shares for clients		
Check for network connectivity between primary and failover node		

Following are the additional steps needed for a successful FC failover:

3. Enable ALUA on any one of the HA cluster node.

4. Ensure that you have cluster setup before enabling ALUA.

```
CLI@nexenta> hactl alua-enable
```

5. 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
 SYSTEM ALUA enabled      : True
 SYSTEM Node ID           : 0
 STMFHA ALUA enabled      : True
 STMFHA Node ID           : 0

nodeB reports
 SYSTEM ALUA enabled      : True
 SYSTEM Node ID           : 1
 STMFHA ALUA enabled      : True
 STMFHA Node ID           : 1
```

Failover

INTERFACE MONITORING

Interface monitoring monitors the status of services interfaces (aka front-end interfaces). When enabled, if the system detects that all front-end interfaces supporting a given service are down, it will automatically failover that service, independent of the state of the head node.

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 service. 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 `fcmon` in the command:

```
CLI@nexenta> hactl monitor-enable fcmon
```

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

NODE        STATUS      MODE        UNBLOCKED
 nef06       stopped   automatic   yes
nef07       running   automatic   yes
```

6. Now check the status of the HA cluster from the surviving node.
```
CLI@nef07> hacluster status

== Nodes ==
NODE      STATUS   SERVICES   ADDRESS       HostId     Release
nef06     up       0/1        10.3.53.109   1fb4ba7c   3.12.0
nef07     up       1/1        10.3.53.110   808b556b   3.12.0
```

7. From nef07 verify that the pool(hapool) imported to it.
```
CLI@nef07> 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
```

To force a failover of all running services from one cluster node to another:
```
CLI@nexenta> haservice failover <from-node> <to-node>
```

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.

```
Action            Description
--halt            stop the appliance without powering it down/rebooting
--reboot          to reboot the appliance
--poweroff        to power down the appliance
```

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).

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.

Service Operation	Command
List all the created HA services	`CLI@nexenta> haservice list`

Stop service on a specific node. Start service on a specific node.	`CLI@nexenta> haservice stop <service> <node>`
	`CLI@nexenta> haservice stop hapool nef06`
	`CLI@nexenta> haservice start hapool nef06`

Remove a specific service. Once the service has been destroyed, the associated pools are exported.	`CLI@nexenta> haservice destroy hapool` Deletes the service `hapool`.
Force a failover of all running services from one cluster node to another.	`CLI@nexenta> haservice failover [-ynvs] <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	`CLI@nexenta> haservice mark hapool nef07` Marks the service `hapool` as recovered.

head or possible data corruption.	
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` and netmask `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`
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`

Troubleshooting

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

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

  ADDRESS        HOSTNAME    ALIAS     PROTOCOL
  ::1            localhost   -         ipv6
  127.0.0.1      localhost   loghost   ipv4
  10.3.53.109    nef06       -          ipv4
  10.3.53.110    nef07       -         ipv4
  ```

  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
install/nef         5.8K     Wed Dec 21 13:45:34 2016     0
messages            264.6K   Fri Feb  3 09:52:55 2017     4
nef                 1.1M     Fri Feb  3 09:53:06 2017     8
rsf/nef-api         12.1M    Fri Feb  3 09:52:31 2017     0
rsf/rsfmon          1.2M     Fri Feb  3 09:52:55 2017     2
rsf/stmfha          2.9K     Wed Feb  1 16:30:21 2017     2
```

Now print all the rsf/rsfmon journal.

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

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
  c0t5000C500563648C3d0s0 not reserved
  c0t5000C5000FA601B7d0s0 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:
 c0t5000C50056366407d0s0 not reserved
 c0t5000C5000FA601B7d0s0 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     151  -                  nef06,nef07 -         nef07     nef06
    ```

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
    ```