



Reconfigure and update

BeeGFS on NetApp with E-Series Storage

NetApp

March 21, 2024

This PDF was generated from <https://docs.netapp.com/us-en/beegfs/administer-clusters-reconfigure.html> on March 21, 2024. Always check docs.netapp.com for the latest.

Table of Contents

- Reconfigure and update 1
 - Reconfigure the HA cluster and BeeGFS 1
 - Update the HA cluster and BeeGFS 2

Reconfigure and update

Reconfigure the HA cluster and BeeGFS

Use Ansible to reconfigure the cluster.

Overview

Generally reconfiguring any aspect of the BeeGFS HA cluster should be done by updating your Ansible inventory and re-running the `ansible-playbook` command. This includes updating alerts, changing the permanent fencing configuration, or adjusting BeeGFS service configuration. These are adjusted using the `group_vars/ha_cluster.yml` file and a full list of options can be found in the [Specify Common File Node Configuration](#) section.

See below for additional details on select configuration options that administrators should be aware of when performing maintenance or servicing the cluster.

How to Disable and Enable Fencing

Fencing is enabled/required by default when setting up the cluster. In some instances it may be desirable to temporarily disable fencing to ensure nodes aren't accidentally shutdown when performing certain maintenance operations (such as upgrading the operating system). While this can be disabled manually, there are tradeoffs administrators should be aware of.

OPTION 1: Disable fencing using Ansible (recommended).

When fencing is disabled using Ansible, the on-fail action of the BeeGFS monitor is changed from "fence" to "standby". This means if the BeeGFS monitor detects a failure it will attempt to place the node in standby and failover all BeeGFS services. Outside active troubleshooting/testing this is typically more desirable than option 2. The disadvantage is if a resource fails to stop on the original node it will be blocked from starting elsewhere (which is why fencing is typically required for production clusters).

1. In your Ansible inventory at `groups_vars/ha_cluster.yml` add the following configuration:

```
beegfs_ha_cluster_crm_config_options:  
  stonith-enabled: False
```

2. Rerun the Ansible playbook to apply the changes to the cluster.

OPTION 2: Disable fencing manually.

In some instances you may want to temporarily disable fencing without rerunning Ansible, perhaps to facilitate troubleshooting or testing of the cluster.



In this configuration if the BeeGFS monitor detects a failure, the cluster will attempt to stop the corresponding resource group. It will NOT trigger a full failover or attempt to restart or move the impacted resource group to another host. To recover, address any issues then run `pcs resource cleanup` or manually place the node in standby.

Steps:

1. To determine if fencing (stonith) is globally enabled or disabled run: `pcs property show stonith-enabled`
2. To disable fencing run: `pcs property set stonith-enabled=false`
3. To enable fencing run: `pcs property set stonith-enabled=true`

Note: This setting will be overridden the next time you run the Ansible playbook.

Update the HA cluster and BeeGFS

Use Ansible to update BeeGFS and the HA cluster.

Overview

BeeGFS is versioned following a `major.minor.patch` versioning scheme and BeeGFS HA Ansible roles are provided for each supported BeeGFS `major.minor` version (for example `beegfs_ha_7_2` and `beegfs_ha_7_3`). Each of the HA roles is pinned to the latest BeeGFS patch version at the time the Ansible collection was released.

Ansible should be used for all BeeGFS upgrades, including moving between major, minor, and patch versions of BeeGFS. To update BeeGFS you will first need to update the BeeGFS Ansible collection, which will also pull in the latest fixes and enhancements to the deployment/management automation and underlying HA cluster. Even after updating to the latest version of the collection, BeeGFS will not be upgraded until `ansible-playbook` is ran with the `-e "beegfs_ha_force_upgrade=true"` set.



For more information on BeeGFS versions see the [BeeGFS Upgrade documentation](#).

Tested Upgrade Paths

Each version of the BeeGFS collection is tested with specific versions of BeeGFS to ensure interoperability between all components. Testing is also performed to ensure upgrades can be performed from the BeeGFS version(s) supported by the last version of the collection, to those supported in the latest release.

Original Version	Upgrade Version	Multirail	Details
7.2.6	7.3.2	Yes	Upgrading beegfs collection from v3.0.1 to v3.1.0, multirail added
7.2.6	7.2.8	No	Upgrading beegfs collection from v3.0.1 to v3.1.0
7.2.8	7.3.1	Yes	Upgrade using beegfs collection v3.1.0, multirail added
7.3.1	7.3.2	Yes	Upgrade using beegfs collection v3.1.0

BeeGFS Upgrade Steps

The follow sections provide the steps to update the BeeGFS Ansible collection and BeeGFS itself. Pay special attention to any extra step(s) for updating BeeGFS major or minor versions.

Step 1: Upgrade BeeGFS Collection

For collection upgrades with access to [Ansible Galaxy](#), run the following command:

```
ansible-galaxy collection install netapp_eseries.beegfs --upgrade
```

For offline collection upgrades, download the collection from [Ansible Galaxy](#) by clicking on the desired Install Version` and then Download tarball. Transfer the tarball to your Ansible control node and run the following command.

```
ansible-galaxy collection install netapp_eseries-beegfs-<VERSION>.tar.gz  
--upgrade
```

See [Installing Collections](#) for more information.

Step 2: Update Ansible Inventory

Make any required or desired updates to your cluster's Ansible inventory files. See the [Version Upgrade Notes](#) section below for details about your specific upgrade requirements. See the [Use Custom Architectures](#) section for general information on configuring your BeeGFS HA inventory.

Step 3: Update Ansible Playbook (when updating major or minor versions only)

If you are moving between major or minor versions, in the `playbook.yml` file used to deploy and maintain the cluster, update the name of the `beegfs_ha_<VERSION>` role to reflect the desired version. For example, if you wanted to deploy BeeGFS 7.3 this would be `beegfs_ha_7_3`:

```
- hosts: all  
  gather_facts: false  
  any_errors_fatal: true  
  collections:  
    - netapp_eseries.beegfs  
  tasks:  
    - name: Ensure BeeGFS HA cluster is setup.  
      ansible.builtin.import_role: # import_role is required for tag  
        availability.  
        name: beegfs_ha_7_3
```

For more details on the contents of this playbook file see the [Deploy the BeeGFS HA cluster](#) section.

Step 4: Run the BeeGFS Upgrade

To apply the BeeGFS update:

```
ansible-playbook -i inventory.yml beegfs_ha_playbook.yml -e  
"beegfs_ha_force_upgrade=true" --tags beegfs_ha
```

Behind the scenes the BeeGFS HA role will handle:

- Ensure the cluster is in an optimal state with each BeeGFS service located on its preferred node.
- Put the cluster in maintenance mode.
- Update the HA cluster components (if needed).
- Upgrade each file node one at a time as follows:
 - Place it into standby and failover its services to the secondary node.
 - Upgrade BeeGFS packages.
 - Fallback back services.
- Move the cluster out of maintenance mode.

Version Upgrade Notes

Upgrading from BeeGFS version 7.2.6 or 7.3.0

Changes to Connection Based Authentication

BeeGFS versions released after 7.3.1 will no longer allow services to start without either specifying a `connAuthFile` or setting `connDisableAuthentication=true` in the service's configuration file. It is highly recommended to enable connection based authentication security. See [BeeGFS Connection Based Authentication](#) for more information.

By default the `beegfs_ha*` roles will generate and distribute this file, also adding it to the Ansible control node at `<playbook_directory>/files/beegfs/<beegfs_mgmt_ip_address>_connAuthFile`. The `beegfs_client` role will also check for the presence of this file and supply it to the clients if available.



If the `beegfs_client` role was not used to configure clients, this file will need to be manually distributed to each client and the `connAuthFile` configuration in the `beegfs-client.conf` file set to use it. When upgrading from a previous version of BeeGFS where connection based authentication was not enabled, clients will loose access unless connection based authentication is disabled as part of the upgrade by setting `beegfs_ha_conn_auth_enabled: false` in `group_vars/ha_cluster.yml` (not recommended).

For additional details and alternate configuration options see the step to configure connection authentication in the [Specify Common File Node Configuration](#) section.

Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.