Technical Report

BeeGFS with NetApp E-Series

Solution Deployment

Abdel Sadek, NetApp
March 2019 | TR-4755-DEPLOY

Abstract

This document provides an overview of the building blocks for BeeGFS using NetApp® E-Series storage systems. These range from a basic solution with a hybrid E-Series in the back-end for both metadata target (MDT) and object storage target (OST), to a more advanced solution where the MDT is a separate all flash EF-Series array.

NetApp®

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Solution Overview

With a competitive price-to-performance ratio, a NetApp E-Series storage systems makes an excellent choice for BeeGFS back-end storage. E-Series also supports a wide range of high performing storage area network (SAN) protocols, making it a flexible choice for most use cases. E-Series supports the following protocols:

- InfiniBand (IB) running iSCSI Extensions for RDMA (iSER)
- IB running SCSI RDMA Protocol (SRP)
- NVMe over IB (NVMe/IB)
- NVMe over RDMA over Converged Ethernet (NVMe/RoCE)
- NVMe over Fibre Channel (NVMe/FC)
- Serial attached SCSI (SAS)
- Fibre Channel (FC)
- Internet Small Computer System Interface (iSCSI)

1.1 Solution Technology

BeeGFS is a parallel file system with an architecture that is based on four main services:

- **Management service.** Registers and monitors all other services.
- **Storage service.** Stores the distributed user file contents.
- **Metadata service.** Stores access permissions and striping information.
- **Client service.** Mounts the file system to access the stored data.

Note: For more information about BeeGFS, see the links provided in the “Where to Find Additional Information” section of this document.

NetApp E-Series storage systems provide a solid back-end target for both the storage service and the metadata service. This document provides the steps to set up and scale out BeeGFS configurations based on E-Series building blocks.

You must verify the compatibility of all components (operating systems, multipathing, host bus adapters (HBAs), switches and so on). To verify the components, see the NetApp Interoperability Matrix Tool (IMT).

For information about host setup, see SANtricity Software Express Configuration for Linux in the E-Series and SANtricity 11 Documentation Center.

1.2 Use Case Summary

This solution applies to the following use cases:

- **Basic building block.** For this use case, both metadata and data reside on the same E-Series hybrid target.
- **Advanced building block.** For this use case, metadata resides on all flash EF-Series system while data resides on an E-Series system with HDDs.

2 Basic Building Block

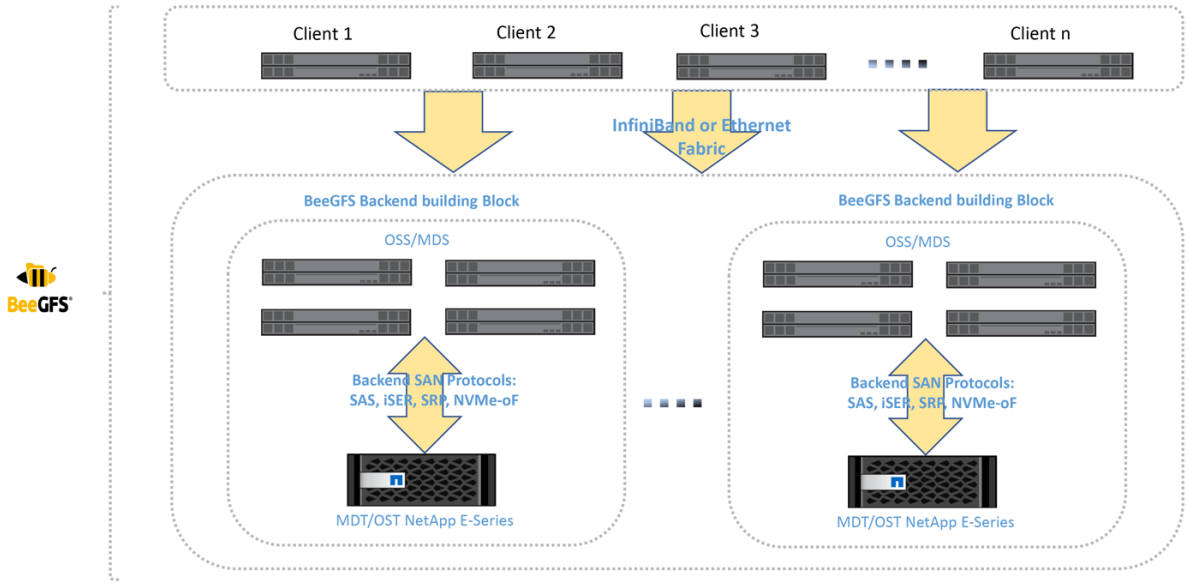
2.1 Technology Requirements

As shown in Figure 1, the basic back-end building block is comprised of an E-Series hybrid array with both SSDs and HDDs for data. Volumes created from the SSDs serve as MDTs, and volumes created from the

HDDs serve as OSTs. You can connect an E-Series array directly to up to four object storage servers (OSS), with at least one of them serving as a metadata server (MDS) as well.

Adding a switch to the back-end increases the potential number of OSSs/MDSs for each target array. Keeping the back-end directly connected without a switch helps lower the price for the user.

Figure 1) BeeGFS Basic building block on E-Series.



Hardware Requirements

In the following deployment procedures, 12GB SAS HBAs connect each MDS or OSS to the E-Series array. Any other block protocol supported by E-Series can be used. For simplicity, this use case uses one MDS, two OSSs, and two clients. Table 1 lists the hardware components that are required to implement the use case.

Table 1) Hardware requirements.

Host ID	Name	BeeGFS Role	Hardware Specifications
–	ictm0802c02	• MDT • OST	• Model: NetApp E5760 • Drives (60 total): SSDs 12; HDDs 48. • Controller firmware version: 08.50.00.03 • Controller NVSRAM version: N5700-850834-D02 • SANtricity System Manager version: 11.50.0G00.0007
1	ictm0803h11-mds	• MDS • Management server	• Memory: 64GB • CPU: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz • HBAs: IB: Mellanox MCX556A-EDAT 100Gbps.

Host ID	Name	BeeGFS Role	Hardware Specifications
			SAS: Broadcom SAS9300-8e 12 Gbps
2	ictm0803h12-oss	OSS	• Memory: 64GB • CPU: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz • HBAs: IB: Mellanox MCX556A-EDAT 100 Gbps • SAS: Broadcom SAS9300-8e 12Gbps
3	ictm0803h13-oss	OSS	• Memory: 64GB • CPU: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz • HBAs: IB: Mellanox MCX556A-EDAT 100Gbps • SAS: Broadcom SAS9300-8e 12Gbps
–	ictm0803sw01	IB switch	• Make/model: Mellanox MSB 7800 100Gbps • Software version: 3.6.8008
4	ictm0803h10-client	Client (compute node)	• Memory: 64GB • CPU: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz • HBAs: IB: Mellanox MCX556A-EDAT 100Gbps
5	ictm0803h09-client	Client (compute node)	• Memory: 64GB • CPU: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz • HBAs: IB: Mellanox MCX556A-EDAT 100Gbps

Software Requirements

Table 2 lists the software components that are required to implement the basic building block use case.

Table 2) Software requirements.

Software	Version
Operating system	SUSE Linux Enterprise Server (SLES) 12 SP3
NetApp SANtricity software release	11.50

2.2 Deployment Procedures

BeeGFS deployment with E-Series follows four main steps:

1. Set up E-Series storage array.
2. Set up server's back-end.

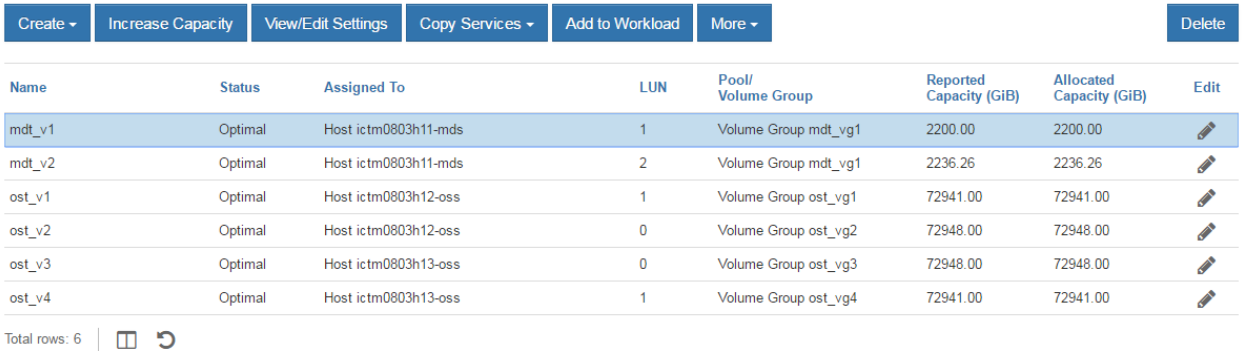
3. Set up front-end and client servers.
4. Set up BeeGFS.

Set Up E-Series Storage Array

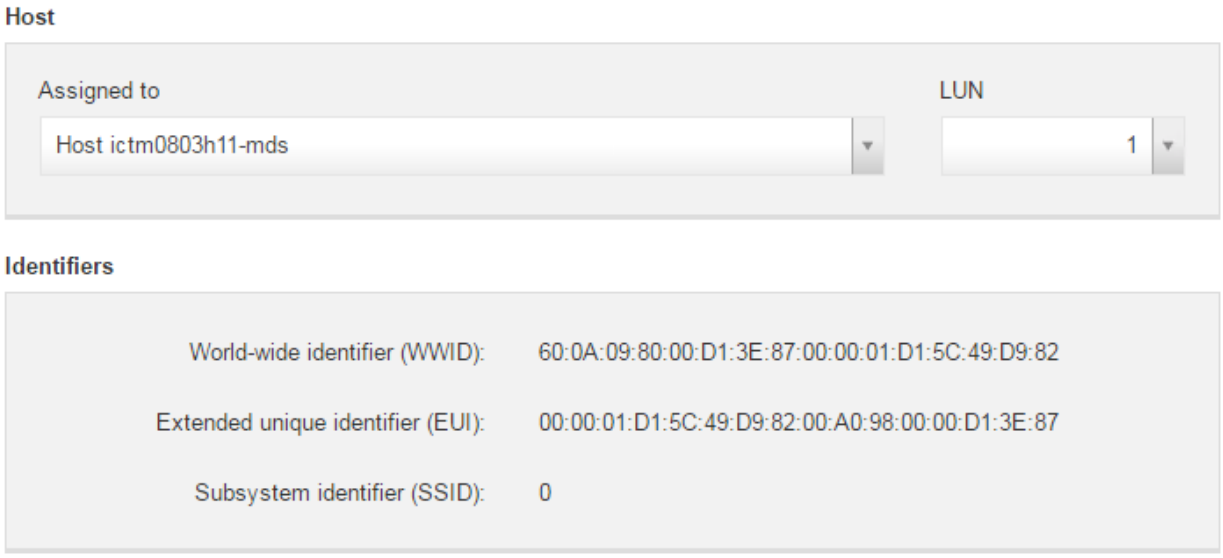
Follow the SANtricity 11.50 online help to create MDT volumes and OST volumes.

1. Using SANtricity System Manager, assign the volumes to the appropriate servers, such as:
   a. Assign volumes for storage service to storage servers.
   b. Assign volumes for metadata service to MDS.

   In this example, a metadata volume `mdt_v1` is mapped to `host ictm0803h11-mds`.



2. To find the worldwide identifier (WWID) of `mdt_v1`, select the volume and click View/Edit Settings. The WWID appears at the bottom of the dialog box.



Set Up Server's Back-End

For detailed instructions for setting up the host, see the Linux express guide.

If you want to easily map the device mapper names on the host to the corresponding volumes on the E-Series storage array, use the following steps to configure user friendly names.

In the following example, volume `mdt_v1` has WWID `3600a098000d13e87000001d15c49d982`, as shown in the previous section.

1. Edit the file `/etc/multipath.conf` as follows:

```
defaults {
            user_friendly_names yes
}
multipaths {
                multipath {
                            wwid 3600a098000d13e87000001d15c49d982
                            alias mdt_v1
                }
}
```

2. Restart multipath service.

```
#  systemctl restart multipathd
```

3. After you have followed the procedure on all OSSs and MDSs, device mapper multipathing shows the friendly names of the devices.

    a. On the MDS:

```
ictm0803h11-mds:~ # multipath -ll
mdt_v1 (3600a098000d13e87000001d15c49d982) dm-2 NETAPP,INF-01-00
size=2.1T features='4 queue_if_no_path pg_init_retries 50 retain_attached_hw_handler' hwhandler='1
alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 0:0:0:1 sdc 8:32 active ready running
`-+- policy='service-time 0' prio=10 status=enabled
`- 0:0:1:1 sdf 8:80 active ready running
mdt_v2 (3600a098000e3c4d0000001f35c49d97b) dm-3 NETAPP,INF-01-00
size=2.2T features='4 queue_if_no_path pg_init_retries 50 retain_attached_hw_handler' hwhandler='1
alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 0:0:1:2 sdg 8:96 active ready running
`-+- policy='service-time 0' prio=10 status=enabled
`- 0:0:0:2 sdd 8:48 active ready running
```

    b. On the first OSS:

```
ictm0803h12-oss:~ # multipath -ll
ost_v2 (3600a098000e3c4d0000001f55c49d980) dm-2 NETAPP,INF-01-00
size=71T features='4 queue_if_no_path pg_init_retries 50 retain_attached_hw_handler' hwhandler='1
alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 0:0:3:0 sdg 8:96 active ready running
`-+- policy='service-time 0' prio=10 status=enabled
`- 0:0:2:0 sde 8:64 active ready running
ost_v1 (3600a098000d13e87000001d45c49d990) dm-3 NETAPP,INF-01-00
size=71T features='4 queue_if_no_path pg_init_retries 50 retain_attached_hw_handler' hwhandler='1
alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 0:0:2:1 sdf 8:80 active ready running
`-+- policy='service-time 0' prio=10 status=enabled
`- 0:0:3:1 sdh 8:112 active ready running
```

    c. On the second OSS:

```
ictm0803h13-oss:~ # multipath -ll
ost_v4 (3600a098000e3c4d0000001f85c49d98b) dm-3 NETAPP,INF-01-00
size=71T features='4 queue_if_no_path pg_init_retries 50 retain_attached_hw_handler' hwhandler='1
alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 0:0:2:1 sdf 8:80 active ready running
`-+- policy='service-time 0' prio=10 status=enabled
`- 0:0:3:1 sdh 8:112 active ready running
ost_v3 (3600a098000d13e87000001d75c49d998) dm-2 NETAPP,INF-01-00
size=71T features='4 queue_if_no_path pg_init_retries 50 retain_attached_hw_handler' hwhandler='1
alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 0:0:3:0 sdg 8:96 active ready running
`-+- policy='service-time 0' prio=10 status=enabled
```

```
`- 0:0:2:0 sde 8:64 active ready running
```

Set Up Front-End and Client Servers

1. On the Mellanox IB switch, enable subnet manager.

   Through the CLI, run the following commands:

```
ictm0803sw01 [standalone: master] # configure terminal
ictm0803sw01 [standalone: master] (config) # ib smnode ictm0803sw01 enable
ictm0803sw01 [standalone: master] (config) # write terminal
```

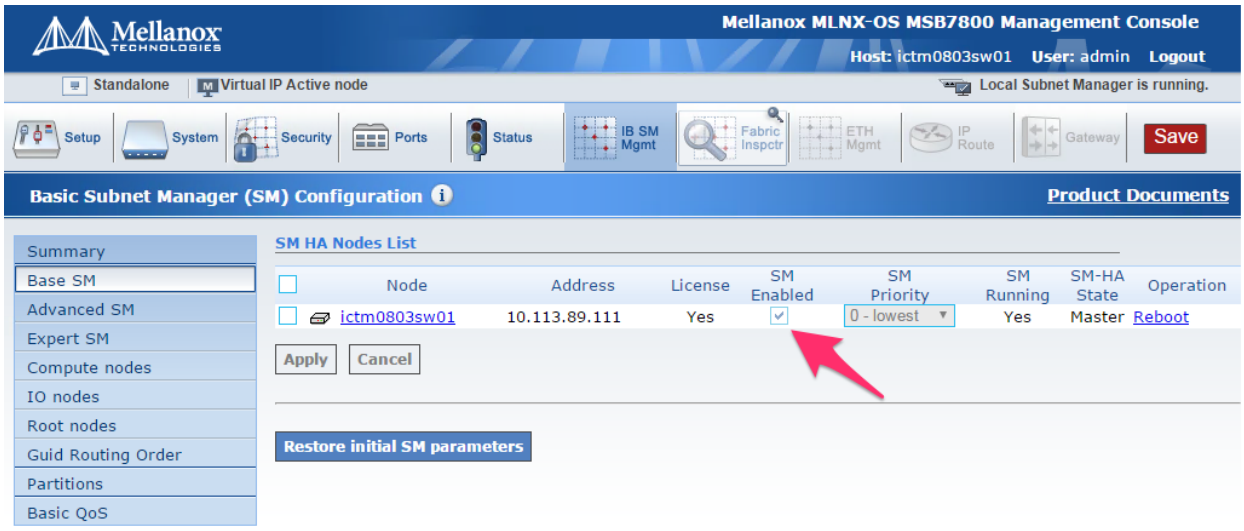
   Through the GUI, complete the following steps:

   a. Log into the web browser using the switch IP.

   b. Go to IB SM Mgmt, and select Base SM.

   c. Select the SM Enabled option.

   d. Click Save.



2. On all nodes (MDS, OSSs, and clients), set up the IP addresses on the IB ports. The IB drivers are part of the inbox RDMA-core driver that comes with the OS. The following example shows the procedure on the MDS:

   a. Start and enable RDMA service with IPoIB enabled.

```
# cat /etc/rdma/rdma.conf | grep -i IPoIB
# Load IPoIB
IPOIB_LOAD=yes
# systemctl enable rdma.service
# systemctl start rdma.service
# systemctl status rdma.service
```

   b. Verify that the IB port is up.

```
ictm0803h11-mds:~ # ibstatus mlx5_1
Infiniband device 'mlx5_1' port 1 status:
default gid: fe80:0000:0000:0000:9803:9b03:0071:f769
base lid: 0x3
sm lid: 0x1
state: 4: ACTIVE
phys state: 5: LinkUp
rate: 100 Gb/sec (4X EDR)
link_layer: InfiniBand
```

   c. Assign the IP address.

Note: The IP address is based on SLES 12 SP3. Other Linux versions, such as RHEL or CentOS, might be different:

```
ictm0803h11-mds:~ # cat /etc/sysconfig/network/ifcfg-ib1
BOOTPROTO='static'
BROADCAST=''
ETHTOOL_OPTIONS=''
IPADDR='192.168.1.11/24'
IPOIB_MODE='connected'
MTU=''
NAME='MT28800 Family [ConnectX-5 Ex]'
NETWORK=''
REMOTE_IPADDR=''
STARTMODE='auto'

ictm0803h11-mds:~ # ip addr show ib1
7: ib1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 65520 qdisc pfifo_fast state UP group default qlen
256
link/infiniband 80:00:08:86:fe:80:00:00:00:00:00:00:98:03:9b:03:00:71:f7:69 brd
00:ff:ff:ff:ff:12:40:1b:ff:ff:00:00:00:00:00:00:ff:ff:ff:ff
inet 192.168.1.11/24 brd 192.168.1.255 scope global ib1
valid_lft forever preferred_lft forever
inet6 fe80::9a03:9b03:71:f769/64 scope link
valid_lft forever preferred_lft forever
```

3. Verify that the following conditions are true:
   - The hosts can ping each other's IPs and host name.
   - Volumes are assigned to the appropriate hosts.
   - IB IPs can ping each other.

The configuration meets all requirements. You can set up your BeeGFS.

Set Up BeeGFS

Different services are required to set up BeeGFS, namely management, metadata, storage, and client services. You must install the corresponding packages listed in Table 3.

Table 3) Required packages for BeeGFS.

Node	Packages
Management server	beegfs-mgmtd
MDS	beegfs-meta libbeegfs-ib
Storage server	beegfs-storage libbeegfs-ib
Client	beegfs-client beegfs-helperd beegfs-utils libbeegfs-ib

Note: libbeegfs-ib only required if you are connecting the clients through IB.

After the packages are installed, you must start the corresponding services.

Set Up Management Service

To set up the management service, complete the following steps:

1. Set up the management location: `# /opt/beegfs/sbin/beegfs-setup-mgmtd -p <management_data_location>`.

```
ictm0803h11-mds:~# /opt/beegfs/sbin/beegfs-setup-mgmtd -p /data/beegfs/mgmt/
```

2. Start the management service and enable it at boot.

```
ictm0803h11-mds:~# systemctl start beegfs-mgmt.service
```

```
ictm0803h11-mds:~# systemctl enable beegfs-mgmt.service
ictm0803h11-mds:~# systemctl status beegfs-mgmt.service
```

Set Up Metadata Service

To set up the metadata service, complete the following steps:

1. Format each mapped volumes: # mkfs.ext4 -i 2048 -I 512 -J size=400 -Odir_index,filetype <mapped volume>.

```
ictm0803h11-mds:~ # mkfs.ext4 -i 2048 -I 512 -J size=400 -Odir_index,filetype /dev/mapper/mdt_v1
```

2. Mount each mapped volumes: mount -onoatime,nodiratime,nobarrier <mapped volume> <mount point>.

```
ictm0803h11-mds:~ # mount -onoatime,nodiratime,nobarrier /dev/mapper/mdt_v1 /mnt/mdt_v1
```

3. Add the following line to `/etc/fstab` if you want to auto mount at boot:

```
/dev/mapper/mdt_v1 /mnt/mdt_v1 ext4 rw,noatime,nodiratime,nobarrier,nofail 0 0
```

4. Configure the metadata service:

   # /opt/beegfs/sbin/beegfs-setup-meta -p <mount point> -s <host id> -m <management node>.

   In this example, metadata service is configured on host ID 1 (as shown in Table 1):

```
ictm0803h11-mds:~# /opt/beegfs/sbin/beegfs-setup-meta -p /mnt/mdt_v1 -s 1  -m ictm0803h11-mds
```

5. To specify which network interface that the MDSs should use for communicating with other MDSs, OSSs, and clients, create a text file under `/etc/beegfs`, with the interface names to be used listed one per line. In this example, `ib1` is used. The absolute path of the file must be specified in the configuration file: `/etc/beegfs-meta.conf`.

   Note: For simplicity, the name of the file is `connInterfacesFile.conf`, which is also the name of the variable in the configuration file.

```
ictm0803h11-mds:~ # cat /etc/beegfs/connInterfacesFile.conf
ib1
ictm0803h11-mds:~ # cat /etc/beegfs/beegfs-meta.conf | grep connInterfacesFile
connInterfacesFile            = /etc/beegfs/connInterfacesFile
```

6. At boot, restart and enable the metadata service.

```
ictm0803h11-mds:~# systemctl start beegfs-meta.service
ictm0803h11-mds:~# systemctl enable beegfs-meta.service
ictm0803h11-mds:~# systemctl status beegfs-meta.service
```

Configure Storage Service

To configure the storage service, complete the following steps:

1. Format each mapped volume: # mkfs.xfs -d su=128k,sw=8 -l version=2,su=128k <mapped volume>.

   Where:

   - su is the segment size of the volume.
   - sw is the number of disks in the RAID group not including the parity disks.

```
ictm0803h12-oss:~# mkfs.xfs -d su=128k,sw=8 -l version=2,su=128k /dev/mapper/ost_v1
```

2. Mount each mapped volumes: # mount -onoatime,nodiratime,logbufs=8,logbsize=256k,largeio,inode64,swalloc,allocsize=131072k,nobarrier <mapped volume> <mount point>.

```
ictm0803h12-oss:~ # mount -
onoatime,nodiratime,logbufs=8,logbsize=256k,largeio,inode64,swalloc,allocsize=131072k,nobarrier
/dev/mapper/ost_v1 /mnt/ost_v1
```

3. Add the following line to `/etc/fstab` if you want to auto mount at boot:

```
/dev/mapper/ost_v1 /mnt/ost_v1 xfs
rw,noatime,nodiratime,logbufs=8,logbsize=256k,largeio,inode64,swalloc,allocsize=131072k,nobarrier,n
ofail 0 0
```

4. Configure the storage service: # /opt/beegfs/sbin/beegfs-setup-storage -p <mount point> -s <host id> -i <target id > -m <management node>.

   In this example, the system has two storage hosts as indicated in previous host table. Each host has two volumes:

   - Storage01 with host ID 2.

```
ictm0803h12-oss:~ # /opt/beegfs/sbin/beegfs-setup-storage -p /mnt/ost_v1 -s 2 -i 201 -m
ictm0803h11-mds
ictm0803h12-oss:~ # /opt/beegfs/sbin/beegfs-setup-storage -p /mnt/ost_v2 -s 2 -i 202 -m
ictm0803h11-mds
```

   - Storage02 with host ID 3.

```
ictm0803h13-oss:~ # /opt/beegfs/sbin/beegfs-setup-storage -p /mnt/ost_v3 -s 3 -i 301 -m
ictm0803h11-mds
ictm0803h13-oss:~ # /opt/beegfs/sbin/beegfs-setup-storage -p /mnt/ost_v4 -s 3 -i 302 -m
ictm0803h11-mds
```

5. To specify which network interface that the OSSs should use to communicate with other OSSs, the MDSs, and clients, create a text file under `/etc/beegfs`, with the interface names to be used listed one per line. In this example, `ib1` is used. The absolute path of the file must be specified in the configuration file: `/etc/beegfs-storage.conf`.

   Note: For simplicity, the name of the file is `connInterfacesFile.conf`, which is also the name of the variable in the configuration file.

```
ictm0803h12-oss:~ # cat /etc/beegfs/connInterfacesFile.conf
ib1
ictm0803h12-oss:~ # cat /etc/beegfs/beegfs-storage.conf | grep connInterfacesFile
connInterfacesFile            = /etc/beegfs/connInterfacesFile
```

6. At boot, restart and enable the storage service.

```
ictm0803h12-oss:~ # systemctl start beegfs-storage.service
ictm0803h12-oss:~ # systemctl enable beegfs-storage.service
ictm0803h12-oss:~ # systemctl status beegfs-storage.service
```

Modify the Client Service

To modify the client service, complete the following steps:

1. If you are using IB, as shown in the following example, modify `/etc/beegfs/beegfs-client-autobuild.conf`.

```
buildArgs=-j8 BEEGFS_OPENTK_IBVERBS=1 OFED_INCLUDE_PATH=/usr/src/openib/include
```

   Note: Use the `OFED_INCLUDE_PATH` parameter only if you are using the out-of-box OpenFabrics Enterprise Distribution (OFED) driver. This example uses the in-the-box RDMA core package (which comes with the operating system), therefore, this parameter is not required.

2. To specify which network interface that the client should use to communicate with the OSSs and the MDSs, create a text file under `/etc/beegfs`, with the interface names to be used listed one per line. In this example, `ib1` is used. The absolute path of the file must be specified in the configuration file: `/etc/beegfs-client.conf`. For simplicity, the name of the file is `connInterfacesFile.conf`, which is also the name of the variable in the configuration file.

```
ictm0803h10-client:~ # cat /etc/beegfs/connInterfacesFile.conf
ib1
ictm0803h10-client:~ # cat /etc/beegfs/beegfs-client.conf | grep connInterfacesFile
```

```
connInterfacesFile            = /etc/beegfs/connInterfacesFile.conf
```

3. Start the client service: # /opt/beegfs/sbin/beegfs-setup-client -m <managment node>.

```
ictm0803h10-client:~ # /opt/beegfs/sbin/beegfs-setup-client -m ictm0803h11-mds
```

This mounts the BeeGFS file system by default on `/mnt/beegfs`.

The mount point can be changed in `/etc/beegfs/beegfs-mounts.conf`.

Verify Setup

To verify that the setup is complete, run the following commands from each client:

1. List the metadata nodes.

```
ictm0803h10-client:~ # beegfs-ctl --listnodes --nodetype=meta --nicdetails
ictm0803h11-mds [ID: 1]
Ports: UDP: 8005; TCP: 8005
Interfaces:
+ ib1[ip addr: 192.168.1.11; type: RDMA]
+ ib1[ip addr: 192.168.1.11; type: TCP]
Number of nodes: 1
Root: 1
```

2. List the storage nodes.

```
ictm0803h10-client:~ # beegfs-ctl --listnodes --nodetype=storage --nicdetails
ictm0803h12-oss [ID: 2]
Ports: UDP: 8003; TCP: 8003
Interfaces:
+ ib1[ip addr: 192.168.1.12; type: RDMA]
+ ib1[ip addr: 192.168.1.12; type: TCP]
ictm0803h13-oss [ID: 3]
Ports: UDP: 8003; TCP: 8003
Interfaces:
+ ib1[ip addr: 192.168.1.13; type: RDMA]
+ ib1[ip addr: 192.168.1.13; type: TCP]
Number of nodes: 2
```

3. List the client nodes.

```
ictm0803h10-client:~ # beegfs-ctl --listnodes --nodetype=client --nicdetails
312C-5C531A93-ictm0803h10-client [ID: 1]
Ports: UDP: 8004; TCP: 0
Interfaces:
+ ib1[ip addr: 192.168.1.10; type: RDMA]
+ ib1[ip addr: 192.168.1.10; type: TCP]
72F1-5C532A97-ictm0803h09 [ID: 2]
Ports: UDP: 8004; TCP: 0
Interfaces:
+ ib1[ip addr: 192.168.1.9; type: RDMA]
+ ib1[ip addr: 192.168.1.9; type: TCP]
Number of nodes: 2
```

4. Display the connections that the client is actually using.

```
ictm0803h10-client:~ # beegfs-net
mgmt_nodes
=============
ictm0803h11-mds [ID: 1]
Connections: TCP: 1 (192.168.1.11:8008);

meta_nodes
=============
ictm0803h11-mds [ID: 1]
Connections: RDMA: 1 (192.168.1.11:8005);

storage_nodes
=============
ictm0803h12-oss [ID: 2]
```

```
Connections: RDMA: 2 (192.168.1.12:8003);
ictm0803h13-oss [ID: 3]
Connections: RDMA: 2 (192.168.1.13:8003);
```

5. Display connectivity to the different services.

```
 ictm0803h10-client:~ # beegfs-check-servers
Management
==========
ictm0803h11-mds [ID: 1]: reachable at 192.168.1.11:8008 (protocol: TCP)

Metadata
==========
ictm0803h11-mds [ID: 1]: reachable at 192.168.1.11:8005 (protocol: RDMA)

Storage
==========
ictm0803h12-oss [ID: 2]: reachable at 192.168.1.12:8003 (protocol: RDMA)
ictm0803h13-oss [ID: 3]: reachable at 192.168.1.13:8003 (protocol: RDMA)
```

6. Display free space and inodes of the storage target and the MDTs.

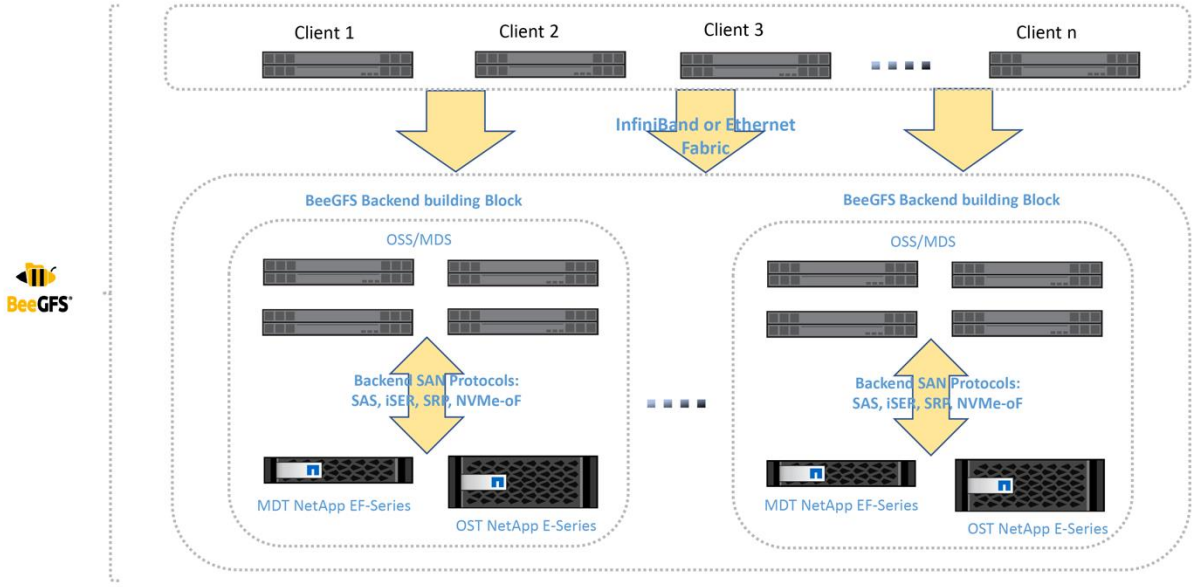
```
 ictm0803h09:~ # beegfs-df
METADATA SERVERS:
TargetID Cap. Pool Total Free % ITotal IFree %
======== ========= ===== ==== = ====== ===== =
1 normal 1649.3GiB 1649.0GiB 100% 1100.0M 1099.9M 100%

STORAGE TARGETS:
TargetID Cap. Pool Total Free % ITotal IFree %
======== ========= ===== ==== = ====== ===== =
201 normal 72939.0GiB 72939.0GiB 100% 2917.6M 2917.6M 100%
202 normal 72946.0GiB 72946.0GiB 100% 2917.9M 2917.9M 100%
301 normal 72946.0GiB 72946.0GiB 100% 2917.9M 2917.9M 100%
302 normal 72939.0GiB 72939.0GiB 100% 2917.6M 2917.6M 100%
```

3 Advanced Building Block

In the advanced building block use case, the MDT resides on a separate all-flash EF-Series array, in this case an EF570 system. The OST remains on an E-Series HDD-based target, in this case an E5700 system, as shown in Figure 2. The models can vary depending on the performance and capacity requirements.

Figure 2) BeeGFS advanced building block on E-Series systems.



To set up the advanced building block use, you can use the same deployment procedures for the basic building block use case.

4 Conclusion

This document provides instructions to set up a BeeGFS configuration with E-Series as a back-end storage in a short amount of time. The instructions focus mainly on functionality and interoperability.

This document can be referenced in any other document that addresses detailed subjects such as tuning parameters for performance or any special use cases of BeeGFS with E-Series.

Where to Find Additional Information

To learn more about the information that is described in this document, see the following documents and/or websites:

- E-Series and SANtricity 11 Documentation Center
  https://docs.netapp.com/ess-11/index.jsp
- SANtricity Software Express Configuration for Linux
  http://docs.netapp.com/ess-11/index.jsp?topic=%2Fcom.netapp.doc.ssm-exp-ic-lin%2Fhome.html
- Official BeeGFS Documentation
  https://www.beegfs.io/wiki/TableOfContents

Version History

Version	Date	Document Version History
Version 1.0	March 2019	Initial release.

Refer to the Interoperability Matrix Tool (IMT) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

TR-4755-DEPLOY-0319