BeeGFS

High-Availability

BeeGFS.io

2022

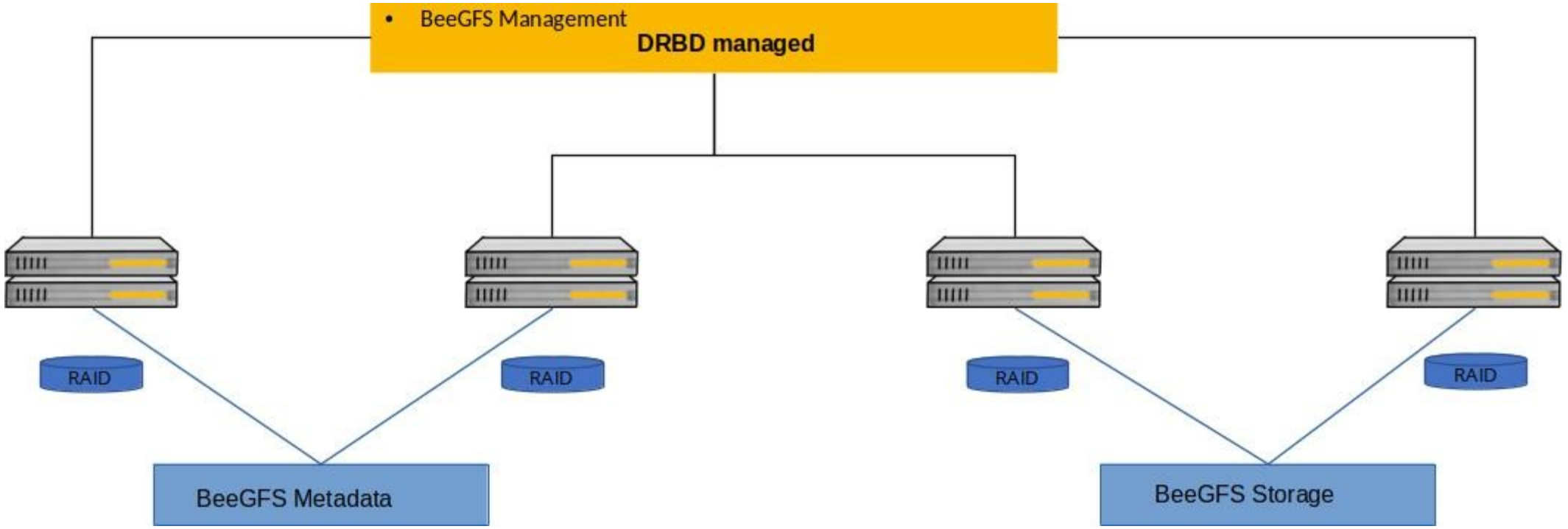
Agenda

- Basic Concepts
  - Multipath
  - DRBD
  - Pacemaker / Corosync: Resourcen
  - Pacemaker / Corosync: Resource Groups
  - Pacemaker / Corosync: Constraints

- High Availability
  - Hybrid Solutions

High Availability – The Training Cluster Design

- **4 Node Pacemaker Cluster with shared and non-shared Storage**
  - BeeGFS mgmtd server non-shared
  - BeeGFS metadata and storage shared



High Availability – 3rd Party Software Setup (RHEL)

- Multipath (for shared disk setup)
- DRBD (for shared nothing setup)
- Pacemaker

  Additional needed Repositories:
  - Elrepo Repository (yum install elrepo-release)
  - High Availability Repository
- Install Software:

```
server1:#> yum install pacemaker pcs drbd90-utils kmod-drbd90 fence-agents-all
```

PART 1: Setup Multipath

- Multipath is installed per default, but /etc/multipath.conf must created separately
- Copy example multipath.conf from /usr/share/doc/device-mapper-multipath to /etc

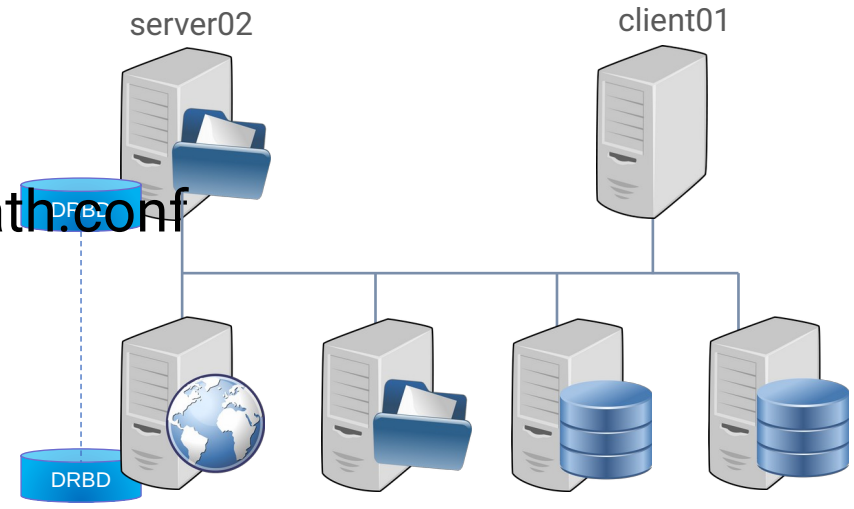
```
server1:#> cp /usr/share/doc/device-mapper-multipath/multipath.conf   /etc/
```

- Edit /etc/multipath.conf
  - Blacklist local disks
  - Find the wwid of the shared disks

```
server1:#> cd /dev/disk/by-id; ls -l;
```

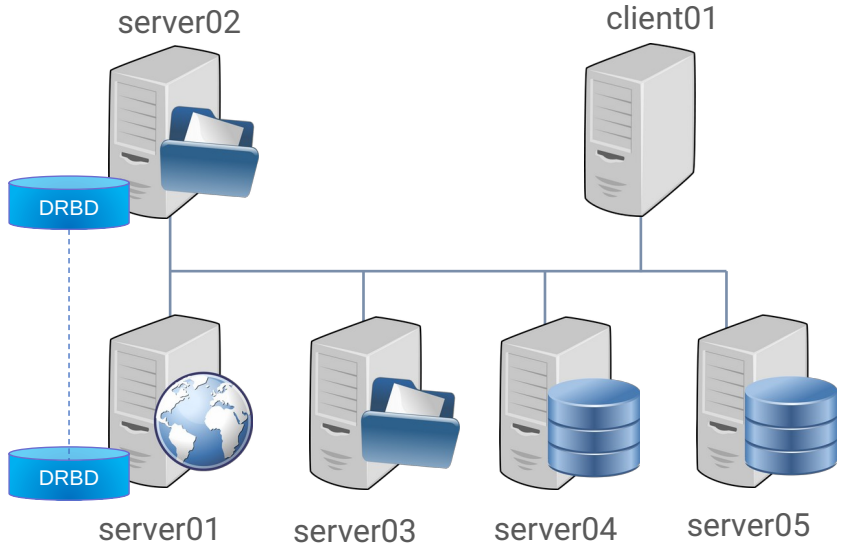
  - Add the wwid to the multipath section in /etc/multipath.conf

```
Multipaths {
        multipath {
                wwid            36001405497b6913ae614b1f84bab3823
                alias             meta1

        }

}
```



PART 1: DIY

- Configure multipath in your cluster
  - Copy the example configuration
  - Find the wwid of your disks
  - Edit /etc/multipath.conf
  - Changes come into effect after restart of multipathd
  - Once you have it running on one node, you can copy it to the other nodes



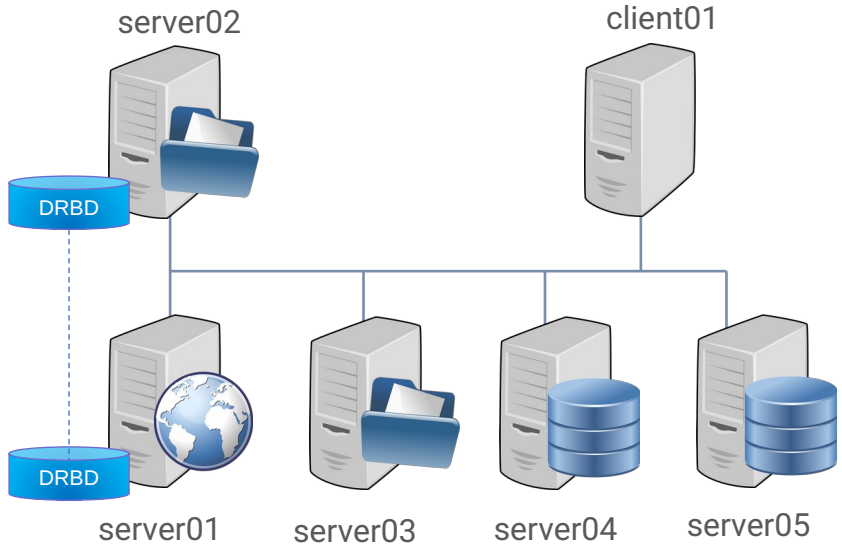
PART 1: Setup DRBD 1

- Needed Environment
  - One disk volume per server. Preferably of equal size
    - For Test Environment: create a logical Volume of 100MB size

```
server1:#> lvcreate -L 100M -n mgmtlv rootvg
```

    - Do this on all four cluster nodes!!

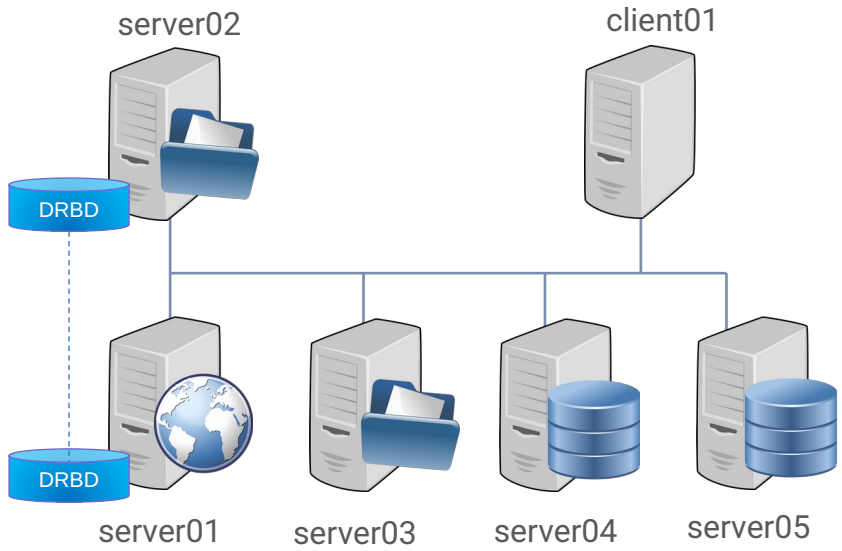
  - A private Network connection does improve performance
  - For RHEL/CentOS/Rocky/Alma: You need elrepo for DRBD



PART 1: Setup DRBD 2

- Configuration files in /etc/drbd.d
  - Create a file beegfs.res
  - If firewall is running open the ports for DRBD

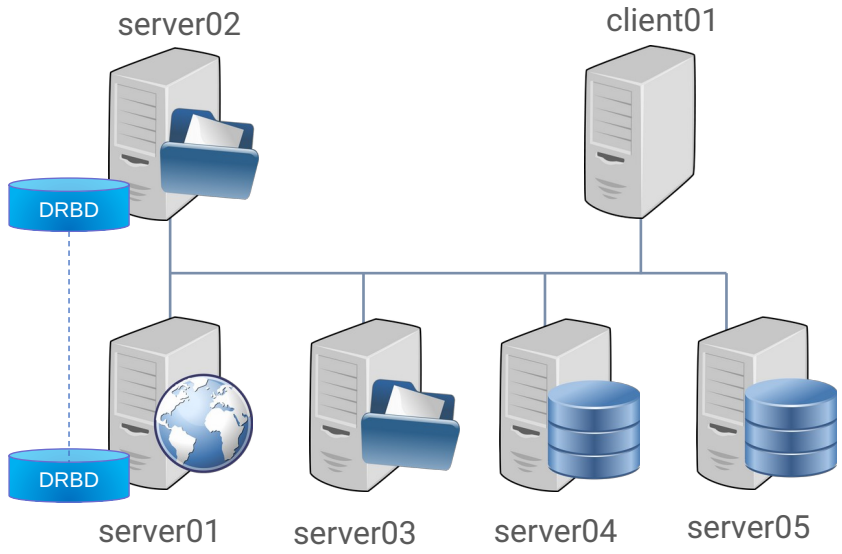
    server1:#> firewall-cmd –zone=public –add-port=7790/tcp –permanent; firewall-cmd --reload
  - A private Network connection does improve performance



PART 1: Setup DRBD 3

- Example of a configuration for a resource file in /etc/drbd.d

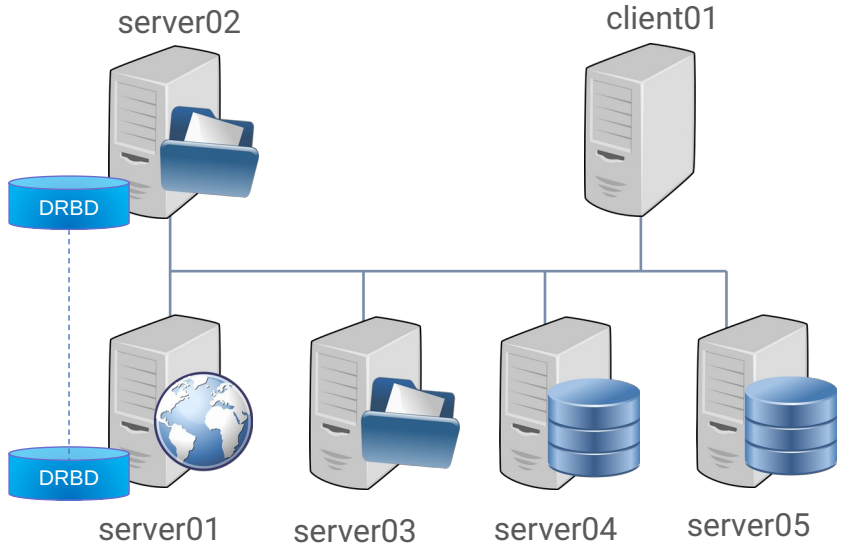
```
resource beegfs {
  device /dev/drbd0;
  disk /dev/rootvg/mgmtlv;
  meta-disk internal;
  on bee-2-1 {
       address 192.168.10.20:7790;
       node-id 0;
  }
  on bee-2-2 {
       address 192.168.10.21:7790;
       node-id 1;
  }
  on bee-2-3 {
       address 192.168.10.22:7790;
       node-id 2;
  }
  on bee-2-4 {
       address 192.168.10.23:7790;
       node-id 3;
  }
  connection-mesh {
      hosts bee-2-1 bee-2-2 bee-2-3 bee-2-4;
  }
}
```



PART 1: Setup DRBD 4

- Necessary changes in /etc/drbd.d/global_common.conf

```
global {
   usage-count no;
   udev-always-use-vnr;
 }
startup  {
        wfc-timeout 100;
        degr-wfc-timeout 120;
   }
options {
        quorum majority;
        on-no-quorum suspend-io;
   }
 net {
        protocol C;
 }
```



PART 1: Setup DRBD 5

- Load DRBD kernel module

  Server1 #> modprobe drbd

- Automatically load Kernel module on reboot

  Server1 #> echo drbd > /etc/modules-load.d/drbd.conf

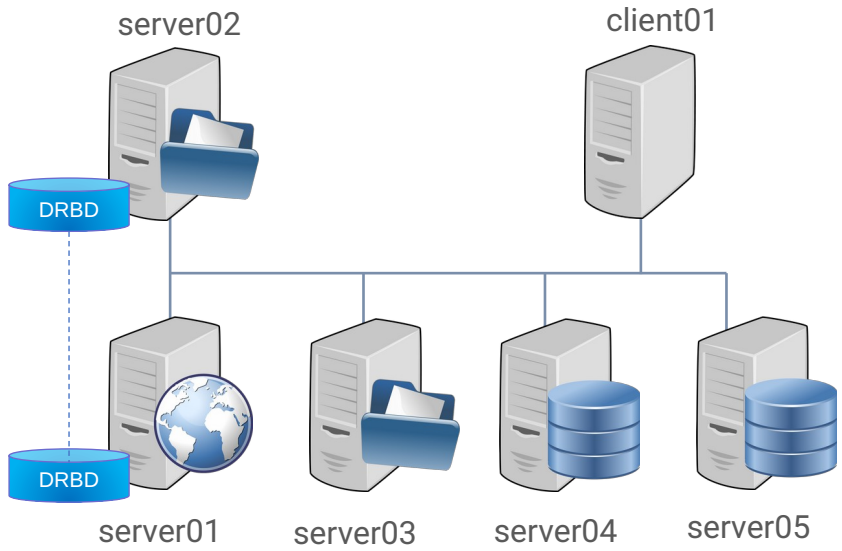
- Startup DRBD
  - Initialize DRBD Volume (on all nodes)

    server1:#> drdbdadm create-md beegfs
  - Start DRBD (on all nodes)

    server1:#> drdbdadm up beegfs
  - Check Status (one Node)

    server1:#> drdbdadm status beegfs

    server1 # drbdadm status
    beegfs role:Secondary
      disk:Inconsistent
      server2 role:Secondary
        peer-disk:Inconsistent



PART 1: Setup DRBD 6

- Startup DRBD (continue)
  - Force primary (only on first startup, only on one node)

```
server1:#> drdbdadm  primary --force beegfs
```

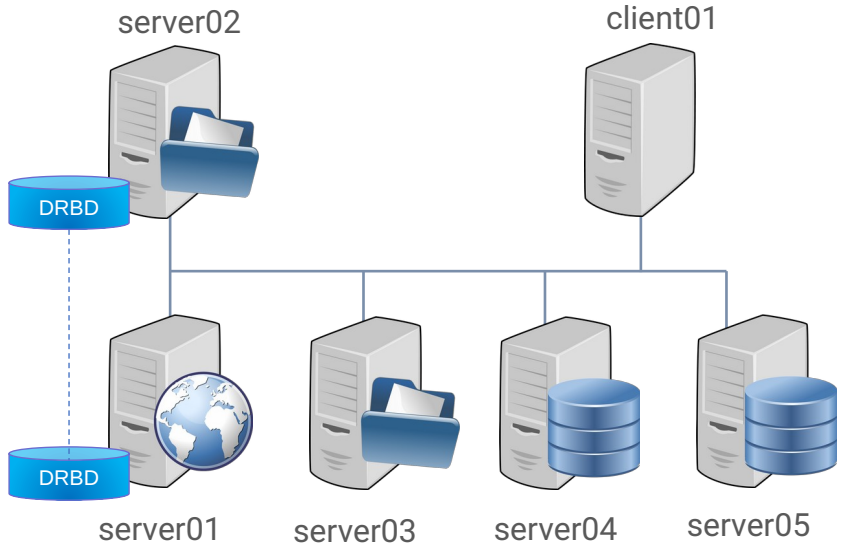
```
Server1 #> drbdadm status
res0 role:Primary
  disk:UpToDate
  p7-uyuni role:Secondary
    replication:SyncSource peer-disk:Inconsistent done:5.32
```

  - After initial sync

```
res0 role:Primary
  disk:UpToDate
  p7-uyuni role:Secondary
    peer-disk:UpToDate
```

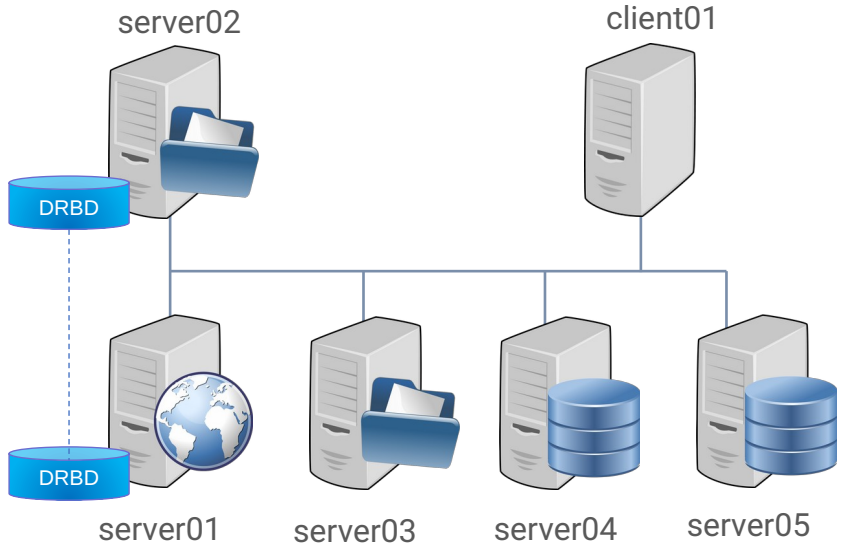
  - You can create a file system on /dev/drbd1

```
Server1 #> mkfs.xfs /dev/drbd1
```



PART 1: DIY

- Setup DRBD in your cluster
  - Install DRBD packages
  - Check Firewall
  - Decide which volumes to use
  - Create config file
  - Perform initial setup of DRBD

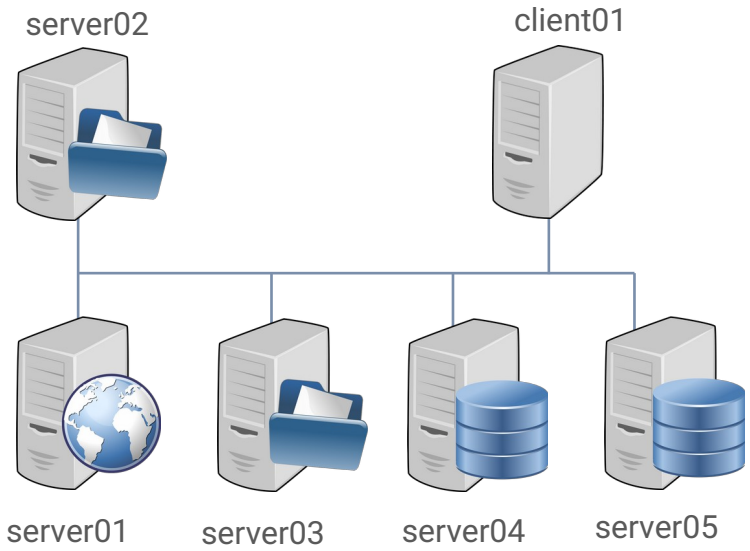


PART 2: Setup Pacemaker 1

- Naming conventions:
  - **Resource**: Every application, disk, file system, IP-Address that is controlled by pacemaker is a resource.
  - **Resource Group**: A collection of resources, that are dependent on each other
  - **Constraint**: A restriction or a set of restrictions for resources or resource group
  - **STONITH**: “Shoot The Other Node In The Head”. A security mechanism to avoid race conditions.

PART 2: Setup Pacemaker 2; plan your cluster

- Before setting up Pacemaker get following information
  - Network (IP-Addresses including virtual IP's)
  - Disk (Storage) Setup
  - STONITH Device
  - Cluster resources
  - Cluster resource groups
  - Dependencies of cluster resources
  - /etc/hosts identical on all cluster nodes or DNS



Before you start

- Collect all Information
  - Multipath configuration
  - Network configuration
  - Disk setup
- IP-Address
  - Collect all virt. IP-Addresses for Managenent, Metadata and Storage
  - Pacemaker configuration: Define IP-Addresses with a label
- ConnInterface file
  - Bind BeeGFS Services to the interface label

PART 2: Setup Pacemaker 3; basic setup

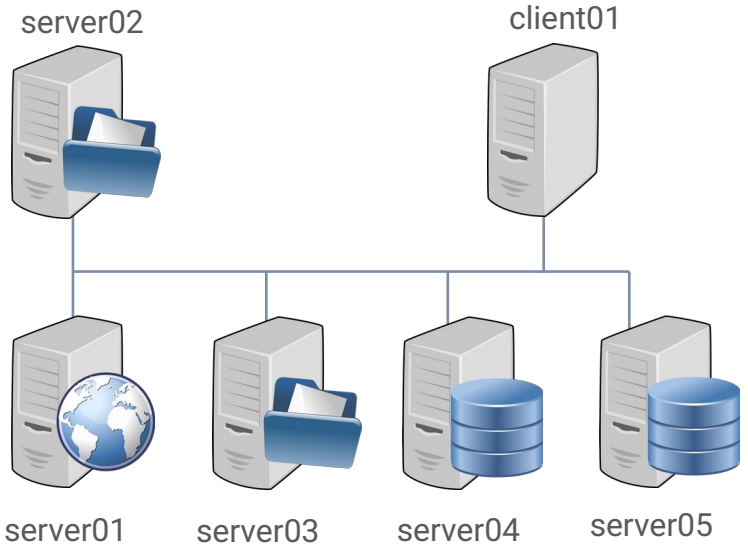
- Pacemaker setup includes
  - Set password for user hacluster (on all nodes in the cluster)

    Server1 #> passwd hacluster
    Server1 #> systemctl start pcsd
  - If a firewall is running, open ports

    Server1 #> firewall-cmd –zone=public –add-service=high-availability \
    Server1 #> --permanent; firewall-cmd --reload
  - Authenticate cluster nodes (one node)

    Server1 #> pcs host auth <node1> <node2>…
  - Create pacemaker cluster (one node)

    Server1 #> pcs cluster setup beegfs_cluster –start <node1> <node2> ..<--force>



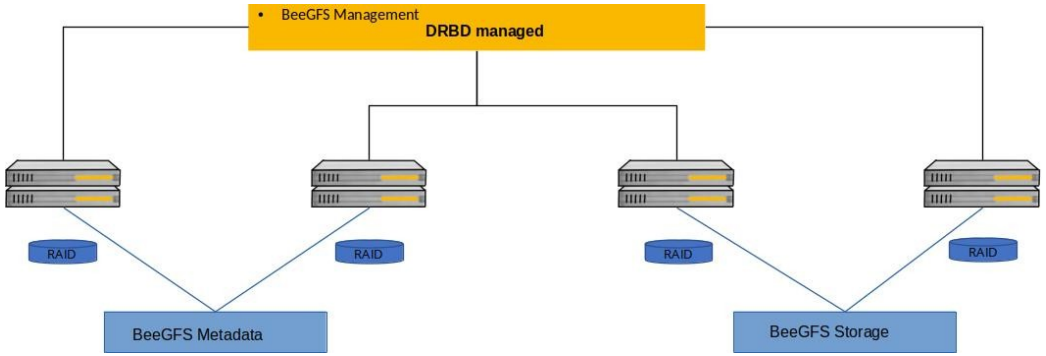
PART 3: Add DRBD as a Pacemaker resource

- DRBD is a clone resource
  - What does this mean?
  - A clone resource cannot be member of a resource group
  - DRBD has a Master/Slave relation (constraint)
  - Create resource DRBD

```
Server1 #> pcs resource create res_drbd ocf:linbit:drbd drbd_resource=beegfs \
Server1 #> drbdconf=/etc/drbd.conf op monitor interval=60s
```

  - Create clone resource with master/slave relation

```
Server1 #>pcs resource promotable res_drbd promoted_max=1 \
Server1 #>clone-max=4 clone-node-max=1 notify=true
```

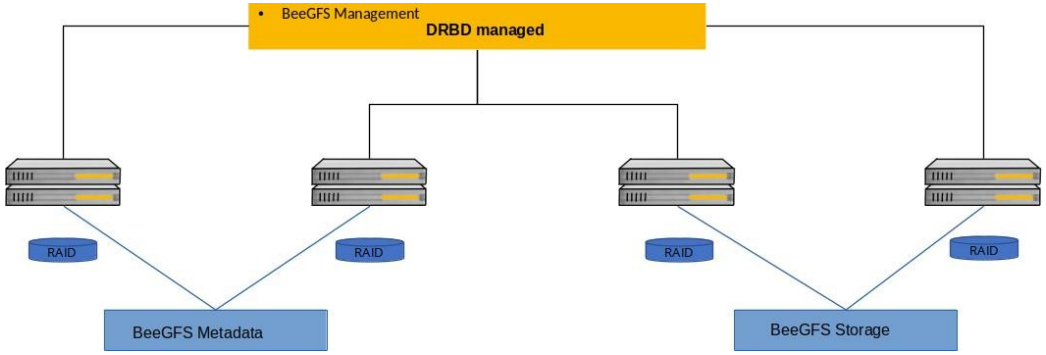


PART 3: DRBD: What if?

- DRBD clone resource does not start?
  - Check Logs!
  - Check if STONITH enabled
    - If you don't have a STONITH device, disable STONITH in the global options

```
Server1 #> pcs property set stonith-enabled=false
```

  - Check Firewall
    - Are the Ports open
  - Check SELinux
    - Disable SELinux or at least set it to permissive

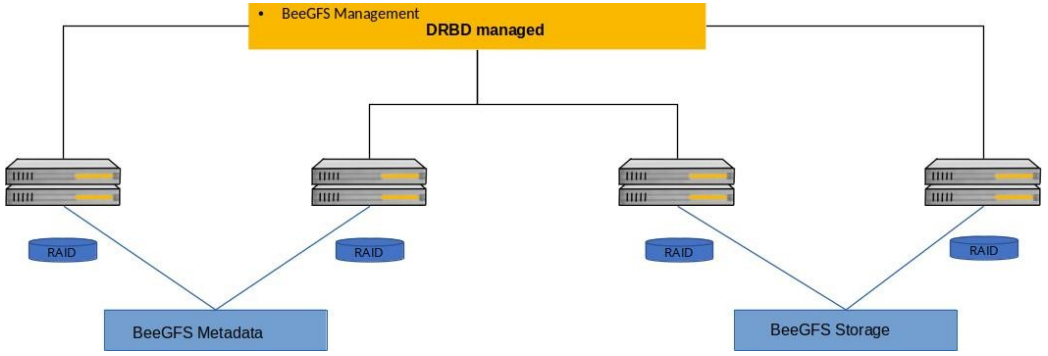



```
Server1 #> setsebool -P daemons_enable_cluster_mode 1
Server1 #> ausearch -c 'drbdsetup' --raw | audit2allow -M my-drbdsetup; \
Server1 #> semodule -X 300 -i my-drbdsetup.pp
Server1 #> ausearch -c 'crm_resource' –raw | audit2allow -M my-crmresource; \
Server1 #> semodule -i my-crmresource.pp
```

PART 3: DRBD: If everything goes as planned

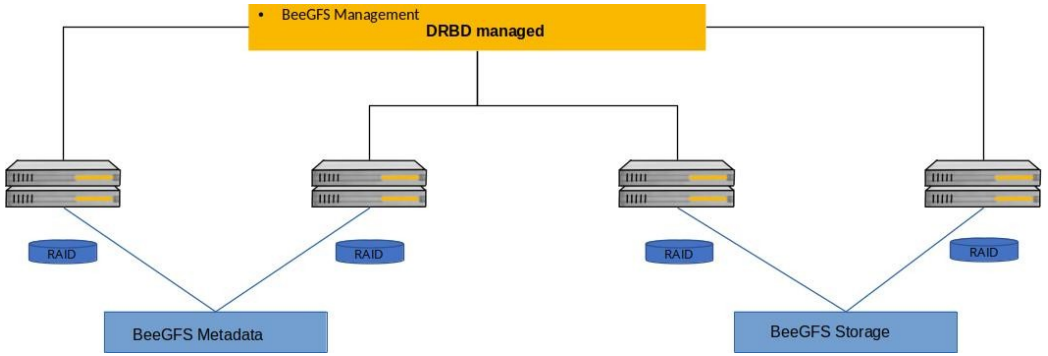
- If everything goes as planned, you will see something like this:

```
[root@bee-2-1 ~]# pcs status
Cluster name: beegfs_cluster
Cluster Summary:
  * Stack: corosync
  * Current DC: bee-2-2 (version 2.1.2-4.el8_6.3-ada5c3b36e2) - partition with quorum
  * Last updated: Fri Nov 11 15:12:57 2022
  * Last change:  Fri Nov 11 14:39:29 2022 by hacluster via crmd on bee-2-1
  * 4 nodes configured
  * 4 resource instances configured

Node List:
  * Online: [ bee-2-1 bee-2-2 bee-2-3 bee-2-4 ]

Full List of Resources:
  * Clone Set: res_drbd-clone [res_drbd] (promotable):
    * Masters: [ bee-2-4 ]
    * Slaves: [ bee-2-1 bee-2-2 bee-2-3 ]

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: inactive/disabled
[root@bee-2-1 ~]#
```



PART 3: DIY DRBD as a pacemaker resource

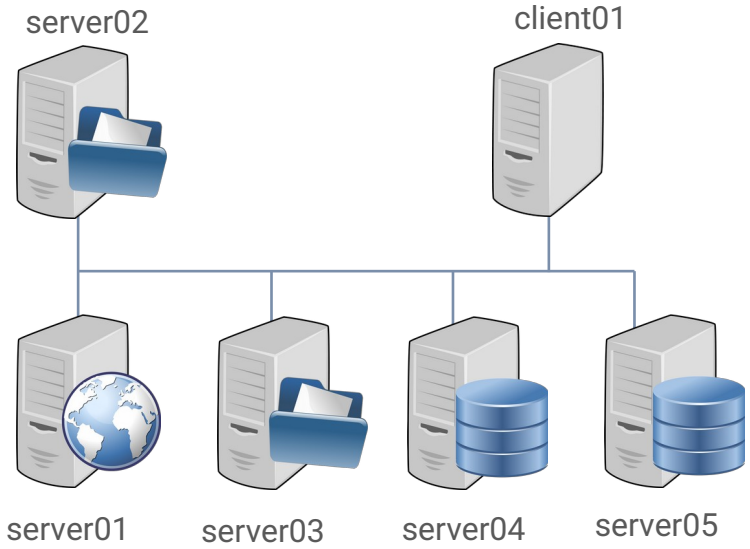
- Setup DRBD as pacemaker resource
  - Define the DRBD resource
  - Make sure it is started on all cluster nodes
  - Create Master/slave dependencies



File Systems

The underlying file systems for BeeGFS have to be cluster aware as well. This means, there should be a resource agent for the file systems available.

- XFS
  - XFS is integrated in the file system resource agent
- Ext4
  - EXT4 is integrated in the file system resource agent
- ZFS
  - There is no default resource agent for ZFS. BeeGFS can provide a resource agent



Before you continue: IP-Addresses and names for BeeGFS Services

- IP-Adress
- Each BeeGFS Service (Management, Metadata, Storage) needs its own Failover IP-Address
- Name

  Each BeeGFS Service (Management, Metadata, Storage) needs a unique name

	Management	Metadata	Storage
IP-Address			
Name			

Part 4: Configuring Resources and Groups: Management

- Create IP resource and add it to a resource group “gr_mgmtd”

```
Server1 #>pcs resource create res_mgmtd_IP ocf:heartbeat:IPaddr2 ip=192.168.10.24 \
Server1 #>cidr_netmask=24 nic=enp2s0 iflabel=mgmtd --group=gr_mgmtd
```

- Bind resource group “gr_mgmtd“ to DRBD primary.
  - This is a colocation constraint

```
Server1 #>pcs constraint colocation add gr_mgmtd with res_drbd-clone INFINITY with-rsc-role=Master
```

- Add a file system resource to the resource group “gr_mgmtd“

```
Server1 #> pcs resource create res_fs_mgmtd ocf:heartbeat:Filesystem device=/dev/drbd0 \
Server1 #> directory=/BeeGFS/mgmtd fstype=ext4 options=noatime,nodiratime,nobarrier --group=gr_mgmtd
```

Part 4: Configuring Resources and Groups: Management

- Configure BeeGFS Management Server
  - Create a connAuthFile
  - Create a connInterfaceFile
  - Configure BeeGFS Management Server
  - Check, that automatic start for BeeGFS Management Server is disabled in systemd

```
Server1 #> beegfs-setup-mgmtd -p /BeeGFS/mgmtd -S bgfs-mgmtd
```

- Distribute connAuthFile, connInterfaceFile and beegfs-mgmtd.conf to all members of the cluster
- Check configuration with a manual start of BeeGFS Management Server

Part 4: Configuring Resources and Groups: Management

- If BeeGFS Management Server started successfully stop it
- Check that the beegfs-pacemaker rpm is installed on all cluster nodes
- Add BeeGFS Management Server to the “gr_mgmtd” resource group

```
Server1 #> pcs resource create res_bgfs_mgmtd ocf:beegfs:beegfs-mgmt-daemon \
Server1 #> cfgFile=/etc/beegfs/beegfs-mgmtd.conf pidfile=/var/run/beegfs-ha/beegfs-mgmtd.pid \
Server1#>  --group=gr_mgmtd
```

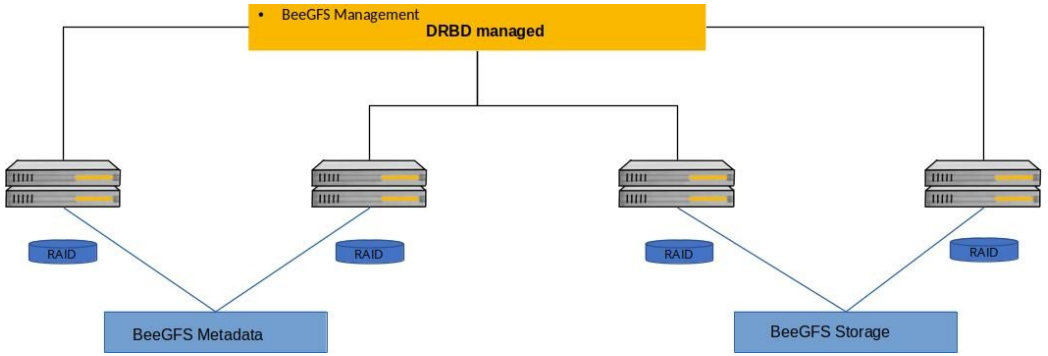
Part 4: Configuring Resources and Groups: Management

- If everything works as designed:

```
[root@bee-2-2 log]# pcs status
Cluster name: beegfs_cluster
Cluster Summary:
  * Stack: corosync
  * Current DC: bee-2-4 (version 2.1.2-4.el8_6.3-ada5c3b36e2) - partition with quorum
  * Last updated: Mon Nov 14 14:49:28 2022
  * Last change:  Mon Nov 14 14:38:25 2022 by root via cibadmin on bee-2-1
  * 4 nodes configured
  * 7 resource instances configured

Node List:
  * Online: [ bee-2-1 bee-2-2 bee-2-3 bee-2-4 ]

Full List of Resources:
  * Clone Set: res_drbd-clone [res_drbd] (promotable):
    * Masters: [ bee-2-1 ]
    * Slaves: [ bee-2-2 bee-2-3 bee-2-4 ]
  * Resource Group: gr_mgmtd:
    * res_mgmtd_IP      (ocf::heartbeat:IPaddr2):      Started bee-2-1
    * fs_mgmtd    (ocf::heartbeat:Filesystem):   Started bee-2-1
    * res_bgfs_mgmtd    (ocf::beegfs:beegfs-mgmt-daemon):    Started bee-2-1

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: inactive/disabled
```

PART 3: DIY Setup BeeGFS Management Server

- Setup BeeGFS Management Server
  - Configure Failover IP in pacemaker and create a resource group
  - Create a location constraint of the resource group to DRBD
  - Configure file system in pacemaker and add to the created resource group
  - Configure BeeGFS Management Server
  - Configure BeeGFSF Management Server in pacemaker and add to the created resource group



Part 4: Configuring Resources and Groups: Metadata

- BeeGFS Metadata is on shared storage!
- There are two metadata servers in the cluster
  - Add connAuthFile to beegfs-meta.conf
  - Create a connInterfaceFile
  - Configure BeeGFS Metadata Server in multimode
  - Check, that automatic start for BeeGFS Metadata Server is disabled in systemd

- Distribute connAuthFile, connInterfaceFile and beegfs-meta.conf to all members of the cluster where you want to run BeeGFS Metadata
- Check configuration with a manual start of BeeGFS Metadata Server

Part 4: Configuring Resources and Groups: Metadata 1

- Create IP resource and add it to a resource group “gr_meta1“

```
Server1 #>pcs resource create res_meta1_IP ocf:heartbeat:IPaddr2 ip=192.168.10.25 \
Server1 #>cidr_netmask=24 nic=enp2s0 iflabel=meta1 --group=gr_meta1
```

- Create a constraint, that “gr_meta1“ can only run on two nodes in the cluster

```
Server1 #>pcs constraint location gr_meta1 avoids bee-2-3=INFINITY bee-2-4=INFINITY
Server1 #>pcs constraint location gr_meta1 prefers bee-2-1=100 bee-2-2=50
```

- Create file system resource and add it to “gr_meta1“

```
Server1 #> pcs resource create res_fs_meta1 ocf:heartbeat:Filesystem device=/dev/mapper/meta1 \
Server1 #> fstype=xfs directory=/BeeGFS/meta1 options=noatime,nodiratime --group=gr_meta1
```

- Configure BeeGFS Metadata Server
  - Check next slide

Part 4: Configuring Resources and Groups: Metadata 1

- Configure BeeGFS Metadata Server
  - Add connAuthFile to beegfs-meta.conf
  - Create a connInterfaceFile
  - Configure BeeGFS Metadata Server in multimode
  - Check TCP and UDP ports for BeeGFS Metadata
  - Check, that automatic start for BeeGFS Metadata Server is disabled in systemd
  - Test BeeGFS Metadata Server configuration by starting manually

```
Server1 #> beegfs-setup-meta -p /BeeGFS/meta -s 1 -m 192.168.10.24 -S bgfs-meta1 \
Server1 #> -c /etc/beegfs/meta1.d/beegfs-meta.conf
```

Part 4: Configuring Resources and Groups: Metadata 1

- Create Metadata resource and add it to a resource group “gr_meta1“

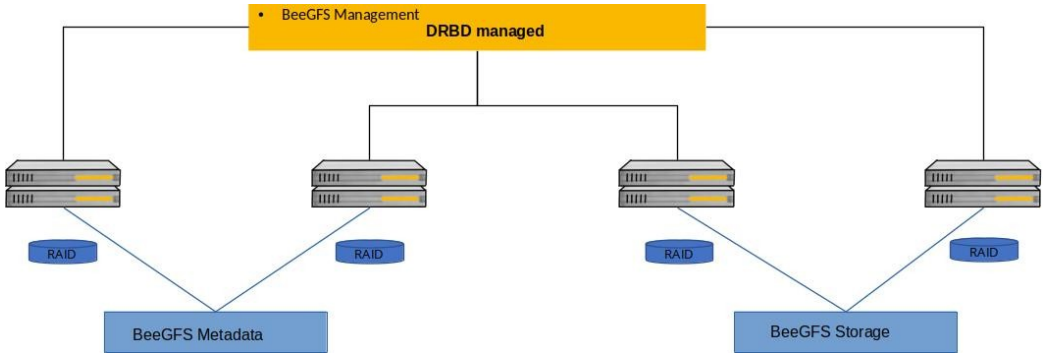
```
Server1 #>pcs resource create res_bgfs_meta1 ocf:beegfs:beegfs-meta-daemon \
Server1 #>cfgFile=/etc/beegfs/meta1.d/beegfs-meta.conf pidfile=/var/run/beegfs-ha/beegfs-meta1.pid \
Server1 #> --group=gr_meta1
```

- After that a „pcs status“ should deliver following result:

```
Server1 #> Node List:
  * Online: [ bee-2-1 bee-2-2 bee-2-3 bee-2-4 ]

Full List of Resources:
  * Clone Set: res_drbd-clone [res_drbd] (promotable):
    * Masters: [ bee-2-3 ]
    * Slaves: [ bee-2-1 bee-2-2 bee-2-4 ]
  * Resource Group: gr_mgmtd:
    * res_mgmtd_IP  (ocf::heartbeat:IPaddr2):      Started bee-2-3
    * fs_mgmtd (ocf::heartbeat:Filesystem):   Started bee-2-3
    * res_bgfs_mgmtd     (ocf::beegfs:beegfs-mgmt-daemon):   Started bee-2-3
  * Resource Group: gr_meta1:
    * res_meta1_IP  (ocf::heartbeat:IPaddr2):      Started bee-2-1
    * res_fs_meta1  (ocf::heartbeat:Filesystem):   Started bee-2-1
    * res_bgfs_meta1     (ocf::beegfs:beegfs-meta-daemon):   Started bee-2-1
```

PART 4: DIY Setup BeeGFS Metadata Server and add it to the cluster

- Setup BeeGFS Metadata Server
  - Configure Failover IP in pacemaker and create a resource group
  - Create a location constraint of the resource group
  - Configure file system in pacemaker and add to the created resource group
  - Configure BeeGFS Metadata Server
  - Configure BeeGFSF Metada Server in pacemaker and add to the created resource group
- Do the same for the second Metadata Server



Part 5: Configuring Resources and Groups: Storage

- BeeGFS Storage is on shared storage!
- BeeGFS Storage must be configured in multimode!
- There are two storage servers in the cluster
  - Add connAuthFile to beegfs-storage.conf
  - Create a connInterfaceFile
  - Check TCP and UDP ports for BeeGFS Storage
  - Check, that automatic start for BeeGFS Storage Server is disabled in systemd

- Distribute connAuthFile, connInterfaceFile and beegfs-storage.conf to all members of the cluster where you want to run BeeGFS Storage
- Check configuration with a manual start of BeeGFS Storage Server

Part 5: Configuring Resources and Groups: Storage 1

- Create IP resource and add it to a resource group “gr_stor1“

```
Server1 #>pcs resource create res_stor1_IP ocf:heartbeat:IPaddr2 ip=192.168.10.27 \
Server1 #>cidr_netmask=24 nic=enp2s0 iflabel=stor1 --group=gr_stor1
```

- Create a constraint, that “gr_stor1“ can only run on two nodes in the cluster

```
Server1 #>pcs constraint location gr_stor1 avoids bee-2-1=INFINITY bee-2-2=INFINITY
Server1 #>pcs constraint location gr_stor1 prefers bee-2-3=100 bee-2-4=50
```

- Create file system resource and add it to “gr_stor1“

```
Server1 #> pcs resource create res_fs_stor1 ocf:heartbeat:Filesystem \
Server1 #> device=/dev/disk/by-id/wwn-0x600140516f9122f6be54979bba37385d  \
Server1 #> fstype=xfs directory=/BeeGFS/stor1 options=noatime,nodiratime --group=gr_stor1
```

Part 5: Configuring Resources and Groups: Storage 1

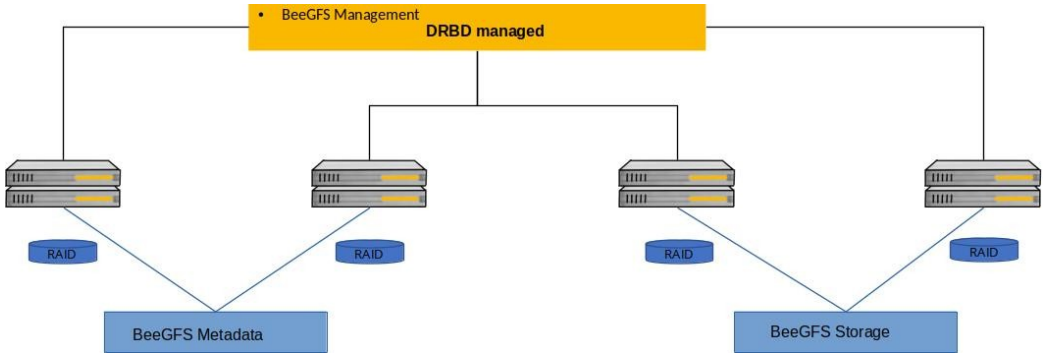
- Configure BeeGFS Storage Server
  - Add connAuthFile to beegfs-storage.conf
  - Create a connInterfaceFile
  - Configure BeeGFS Storage Server in multimode
  - Check, that automatic start for BeeGFS Storage Server is disabled in systemd
  - Test BeeGFS Storage Server configuration by starting manually

```
Server1 #> beegfs-setup-storage -p /BeeGFS/stor1 -s 1 -i 101 -m 192.168.10.24 -S bgfs-stor1 \
Server1 #> -c /etc/beegfs/stor1.d/beegfs-storage.conf
```

```
Server1 #>pcs resource create res_bgfs_stor1 ocf:beegfs:beegfs-storage-daemon \
Server1 #>cfgFile=/etc/beegfs/stor1.d/beegfs-storage.conf pidfile=/var/run/beegfs-ha/beegfs-stor1.pid \
Server1 #> --group=gr_stor1
```

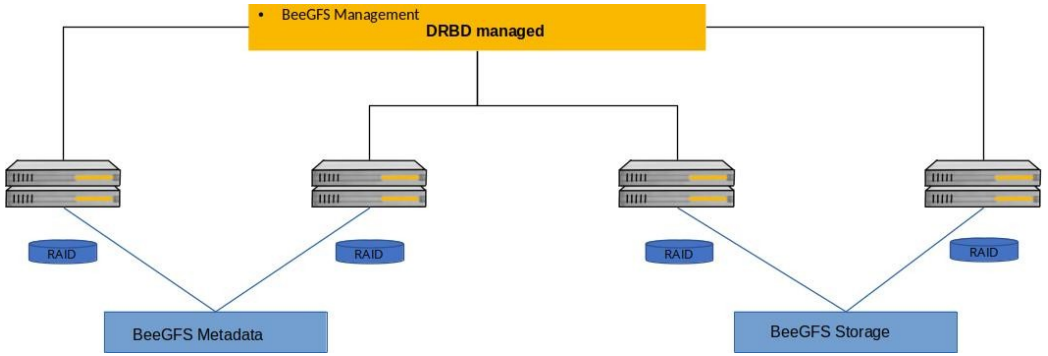
PART 5: DIY Setup BeeGFS Storage Server and add it to the cluster

- Setup BeeGFS Storage Server
  - Configure Failover IP in pacemaker and create a resource group
  - Create a location constraint of the resource group
  - Configure file system in pacemaker and add to the created resource group
  - Configure BeeGFS Storage Server
  - Configure BeeGFSF Storage Server in pacemaker and add to the created resource group
- Do the same for the second Storage Server



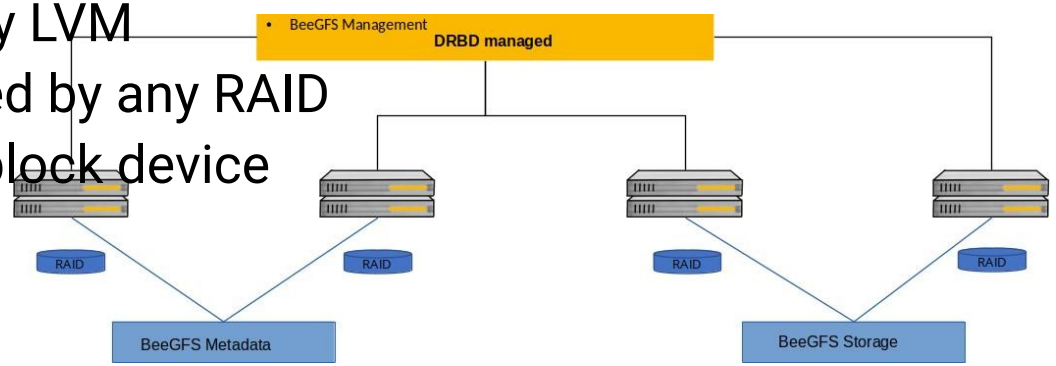
PART 6: Fencing aka STONITH

- There are many fencing options available
- Most used is via the IPMI interface
- In this environment we use SBD (Storage Based Death)



PART 6: STONITH SBD configuration 1

- There are many fencing options available
- Most used is via the IPMI interface
- In this environment we use SBD (Storage Based Death)
- Some dependencies for SBD
  - A block device must be shared by all nodes of the cluster
  - All nodes must have write access to the shared device
  - The shared block device cannot host a filesystem
  - The shared block device cannot be managed by LVM
  - The shared block device should not be managed by any RAID
  - 4 Megabyte in size is sufficient for the shared block device



PART 6: STONITH SBD configuration 2

- Initialize SBD device

```
Server1 #> sbd -d /dev/mapper/sbd create
```

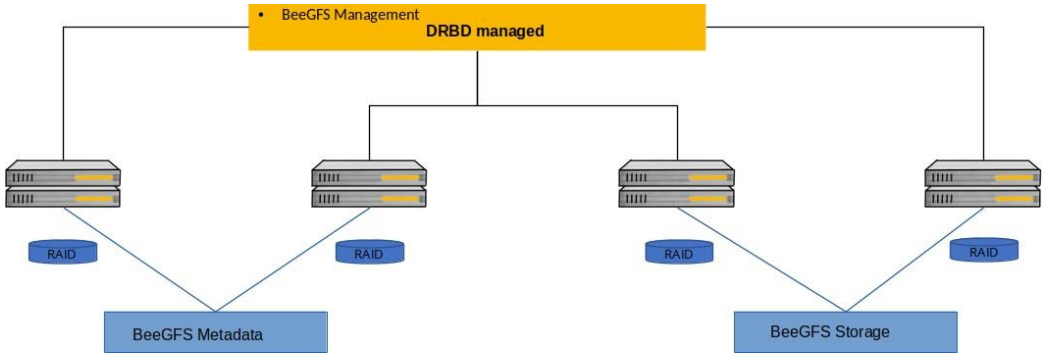
- If you are running SBD in a multipath group you might use this setup

```
Server1 #> sbd -d /dev/mapper/sbd -4 180 -1 90 create
```

  - The -4 option is used to specify msgwait timeout
  - The -1 options is used to specify the watchdog timeout

- Check what has been written to the device

```
[root@bee-1-1 ~]# sbd -d /dev/mapper/sbd dump
==Dumping header on disk /dev/mapper/sbd
Header version     : 2.1
UUID               : e3eea116-e1de-4192-8ae5-8c3fe42b93f2
Number of slots    : 255
Sector size        : 512
Timeout (watchdog) : 5
Timeout (allocate) : 2
Timeout (loop)     : 1
Timeout (msgwait)  : 10
==Header on disk /dev/mapper/sbd is dumped
```



PART 6: STONITH SBD configuration 3

- Edit the SBD configuration file /etc/sysconfig/sbd

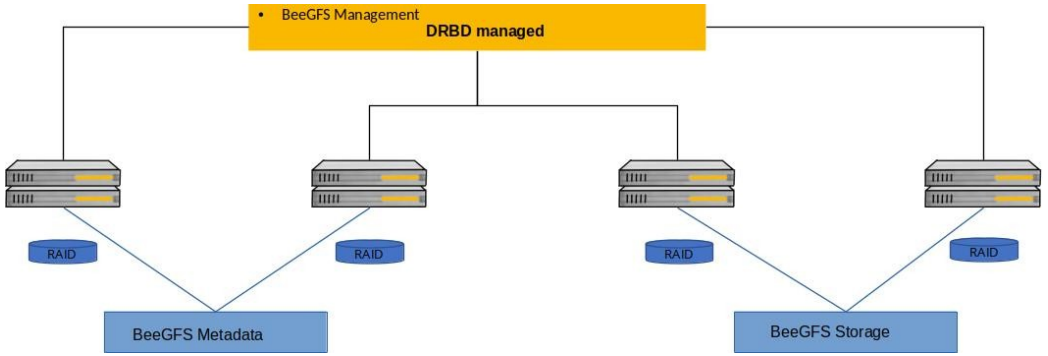
```
[root@bee-1-1 ~]# vi /etc/sysconfig/sbd
SBD_DEVICE=”/dev/mapper/sbd”
SBD_PACEMAKER=yes
SBD_STARTMODE=always
SBD_DELAY_START=no
SBD_WATCHDOG_DEV=/dev/watchdog
SBD_WATCHDOG_TIMEOUT=5
Optional: SBD_OPTS=”-W -P”
```

- Distribute /etc/sysconfig/sbd file to all nodes in the cluster
- Enable SBD in systemd (on all nodes)

```
Server1 #> systemctl enable sbd
```

- Restart pacemaker (on all nodes)

```
Server1 #> systemctl restart pacemaker
```



PART 6: STONITH SBD configuration 4

- After restart you should see the cluster nodes

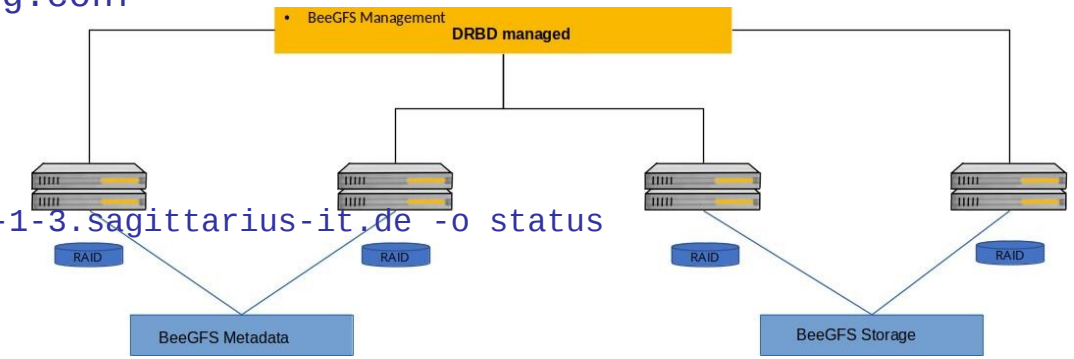
```
[root@bee-1-2 ~]# sbd -d /dev/mapper/sbd list
0       bee-1-4.sagittarius-it.de       clear
1       bee-1-3.sagittarius-it.de       clear
2       bee-1-1.sagittarius-it.de       clear
3       bee-1-2.sagittarius-it.de       clear
```

- Load softdog module in kernel an enable module at reboot (on all nodes)

```
Server1 #> modprobe softdog
Server1 #> echo softdog > /etc/modules-load.d/softdog.conf
```

- Test SBD

```
[root@bee-1-1 ~]#  fence_sbd --devices=/dev/mapper/sbd -n bee-1-3.sagittarius-it.de -o status
Status: ON
```



PART 6: STONITH SBD configuration 5

- Create STONITH resource in pacemaker

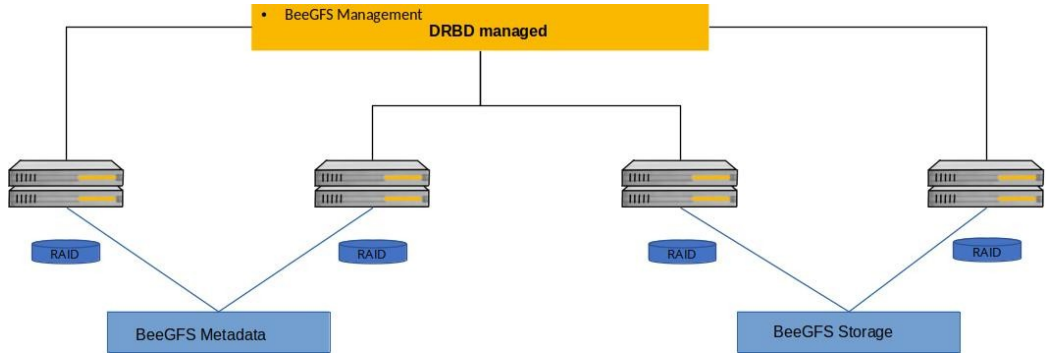
```
Server1 #> pcs stonith create fence-sbd fence_sbd devices=”/dev/mapper/sbd” power_timeout=20
```

- Set STONITH parameters

```
[root@bee-1-1 ~]# pcs property set stonith-watchdog-timeout=0
[root@bee-1-1 ~]# pcs property set stonith-timeout=40s
```

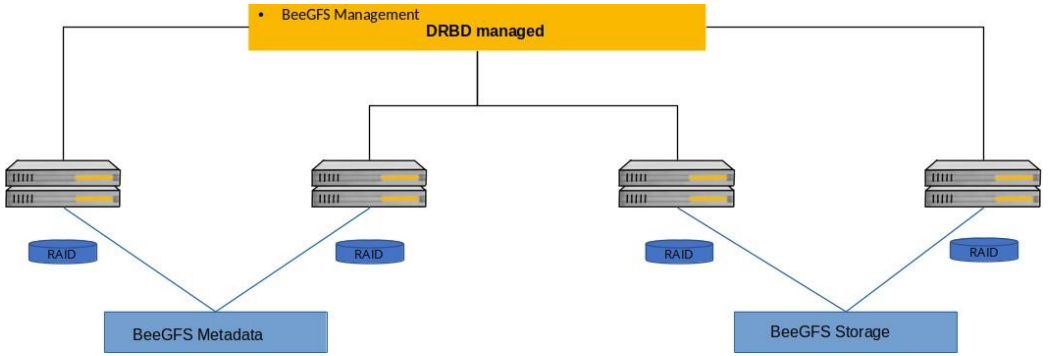
- Enable STONITH again

```
[root@bee-1-1 ~]# pcs property set stonith-enabled=true
```



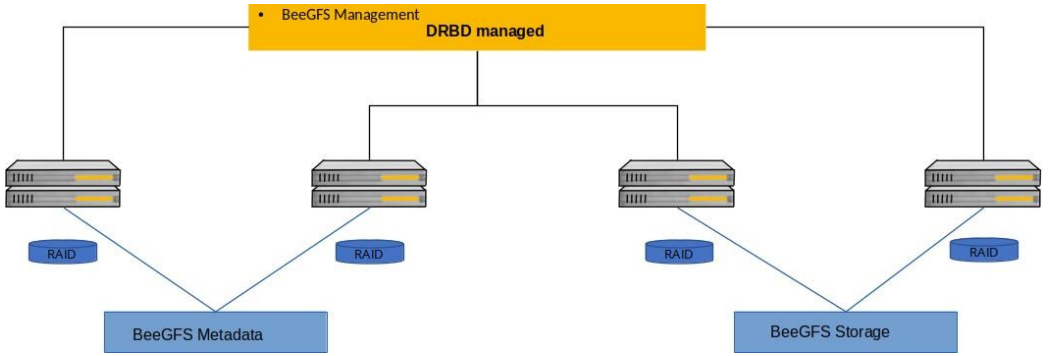
PART 7: Setting Alerts

- Pacemaker can be configured to send alerts
- Available alerts
  - Log
  - SMTP
  - SNMP
- Predefined scripts are available



PART 7: Setting Alerts 2

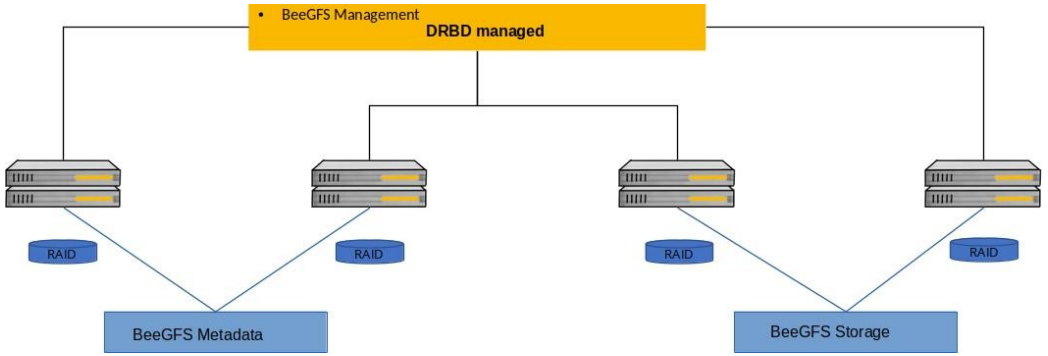
- Alert Scripts are in /usr/share/pacemaker/alerts
- Available alerts
  - alert_file.sh
  - alert_smtp.sh
  - alert_snmp.sh
- Copy the script to directory of your choice. For example: /var/lib/pacemaker



PART 7: Setting Alerts 3

- Create alert resource in pacemaker

```
Server1 #> pcs alert create id=smtp_alert path=/var/lib/pacemaker/alert_smtp.sh options email_sender=pacemaker@example.com

Server1 #> pcs alert recipient add smtp_alert value=admin@example.com
```



Starting and Stopping Services and Resources

- Start cluster HA environment services

```
server1:~ # systemctl start pacemaker
server1:~ # systemctl start pcsd
```

- Stop cluster HA environment services

```
server1:~ # systemctl stop pacemaker
server1:~ # systemctl stop pcsd
```

- Start cluster resources

```
Server1 #> pcs resource enable <resource name>
```

- Stop cluster resources

```
Server1 #> pcs resource disable <resource name>
```

Checking Resources and Cluster status

Check status of cluster resources

```
[root@server1]# pcs status
Cluster name: beegfs-cluster
Stack: corosync
Current DC: server1 (version 1.1.21-4.el7-f14e36fd43) - partition with quorum
Last updated: Fri Oct 30 14:07:02 2020
Last change: Fri Oct 30 13:56:12 2020 by root via cibadmin on server1

2 nodes configured
5 resources configured

Online: [ server1 server2 ]

Full list of resources:

 Master/Slave Set: drbd_res-clone [drbd_res]
     Masters: [ server1 ]
     Slaves: [ server2 ]
 Resource Group: gr_beegfs_mgmtd
     b_mgmtd_IP      (ocf::heartbeat:IPaddr2):    Started server1
     fs_res1     (ocf::heartbeat:Filesystem):         Started server1
     beegfs_mgmtd    (systemd:beegfs-mgmtd):  Started server1

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/disabled
```

High Availability – Hybrid Solution

- More affordable
- Additional data safety
- Reduced failover risk
- Reduced overhead in storage capacity
- All BeeGFS services supported

- Complexity
- Write penalty for replicated data
- Additional Shared Storage

Thank You

Follow BeeGFS: