



# Highly Available NFS Storage with DRBD and Pacemaker

---

## SUSE Linux Enterprise High Availability Extension 12

Florian Haas, Tanja Roth, and Thomas Schraitle

This document describes how to set up highly available NFS storage in a 2-node cluster, using the following components that are shipped with SUSE Linux Enterprise High Availability Extension 12: DRBD®, LVM2 (Logical Volume Manager version 2), and Pacemaker, the cluster resource management framework.

Publication date: Sep 25 2014

## Contents

- 1 Introduction 2
- 2 Prerequisites and Installation 2
- 3 Initial Configuration 3
- 4 Initial Configuration with Hawk 7
- 5 Cluster Resources for an HA NFS Server 12
- 6 Using the NFS Service 19
- 7 Legal Notice 19

# 1 Introduction

NFS (the Network File System) is one of the most long-lived and ubiquitous networked storage solutions on Linux. NFS is widely deployed and, at the time of writing, two NFS versions are of practical relevance to the enterprise: NFSv3 and NFSv4. The solution described in this document is applicable to NFS clients using either version.

## 2 Prerequisites and Installation

Before you proceed with *Section 3, “Initial Configuration”*, make sure that the following prerequisites are fulfilled.

### 2.1 Software Requirements

- SUSE® Linux Enterprise Server 12 is installed on all nodes that will be part of the cluster. All available online updates for the product are installed.
- SUSE Linux Enterprise High Availability Extension 12 is installed on all nodes that will be part of the cluster. All available online updates for the product are installed.

For instructions on how to install the High Availability Extension as add-on on top of SUSE Linux Enterprise Server, refer to *Book “Administration Guide” 3 “Installation and Basic Setup” 3.3 “Installation as Add-on”*.

To create a highly available NFS service, you need to have the following software packages installed:

- pacemaker: A cluster resource management framework which you will use to automatically start, stop, monitor, and migrate resources.
- corosync: A cluster messaging layer. Pacemaker uses Corosync for messaging and node membership.
- drbd and drbd-kmp- *your\_kernel*: Both belong to DRBD, the Kernel block-level synchronous replication facility which serves as an imported shared-nothing cluster building block.

- lvm2: Linux Logical Volume Manager, version 2, which you may use for easy and flexible data management including online volume expansion and point-in-time snapshots.
- nfs-kernel-server: The in-kernel Linux NFS daemon. It serves locally mounted file systems to clients via the NFS network protocol.

## Important: Product Registration And Updates

Proper registration at SUSE Customer Center is mandatory for the system to receive updates. During the registration process, the respective online update repositories are automatically configured.

The repositories for installation and update automatically provide the package versions needed for the setup of highly available NFS storage as described in this document.

## Note: Package Dependencies

You may be required to install packages other than the above-mentioned ones because of package dependencies. However, when using a package management utility such as zypper or YaST, these dependencies are automatically taken care of.

## 2.2 Services at Boot Time

After you have installed the required packages, take care of a few settings applying to your boot process.

Use systemctl or *YaST System Services (Runlevel)* to make sure that:

- Pacemaker *does* start automatically on system boot. This will also start Corosync.
- The drbd service *does not* start automatically on system boot. Pacemaker takes care of all DRBD-related functionality.

## 3 Initial Configuration

This section describes the initial configuration of a highly available NFS export in the context of the Pacemaker cluster manager. Note that the configuration described here will work for NFS clients using NFS versions 3 or 4.

## 3.1 Configuring a DRBD Resource

First, it is necessary to configure a DRBD resource to hold your data. This resource will act as the Physical Volume of an LVM Volume Group to be created later. This example assumes that the LVM Volume Group is to be called nfs. Hence, the DRBD resource uses that same name.

It is highly recommended that you put your resource configuration in a file whose name is identical to that of the resource. As the file must reside in the /etc/drbd.d/ directory, this example uses the /etc/drbd.d/nfs.res file. Its contents should look similar to this:

```
resource nfs {
    device /dev/drbd0;
    disk /dev/sda1;
    meta-disk internal;
    on alice {
        address 10.0.42.1:7790;
    }
    on bob {
        address 10.0.42.2:7790;
    }
}
```

After you have created this resource, copy the DRBD configuration files to the other DRBD node, using either scp or csync2. Proceed with initializing and synchronizing the resource, as specified in . For information about Csync2, refer to *Book “Administration Guide” 3 “Installation and Basic Setup”*3.5.4 “Transferring the Configuration to All Nodes”.

## 3.2 Configuring LVM

To use LVM with DRBD, it is necessary to change some options in the LVM configuration file (/etc/lvm/lvm.conf) and to remove stale cache entries on the nodes:

1. Open /etc/lvm/lvm.conf in a text editor.
2. Search for the line starting with filter and edit it as follows:

```
filter = [ "r|/dev/sda.*|" ]
```

This masks the underlying block device from the list of devices LVM scans for Physical Volume signatures. This way, LVM is instructed to read Physical Volume signatures from DRBD devices, rather than from the underlying backing block devices.

However, if you are using LVM *exclusively* on your DRBD devices, then you may also specify the LVM filter as such:

```
filter = [ "a|/dev/drbd.*|", "r|.*)" ]
```

3. In addition, disable the LVM cache by setting:

```
write_cache_state = 0
```

4. Save your changes to the file.
5. Delete `/etc/lvm/cache/.cache` to remove any stale cache entries.
6. Use Csync2 to replicate these changes to peer node.
7. Enable your DRBD volume and start the initial synchronization:

```
root # drbdadm primary --force RESOURCE
root # drbdadm up
```

Find more information about DRBD in *Book “Administration Guide” 15 “DRBD”*.

## **Important: Automatic Synchronization**

Execute all of the following steps only on the node where your resource is currently in the primary role. It is *not* necessary to repeat the commands on the DRBD peer node as the changes are automatically synchronized.

Now you can prepare the Physical Volume, create an LVM Volume Group with Logical Volumes and create file systems on the Logical Volumes.

1. To be able to create an LVM Volume Group, first initialize the DRBD resource as an LVM Physical Volume. To do so, issue the following command:

```
root # pvcreate /dev/drbd/by-res/nfs
```

2. Create an LVM Volume Group that includes this Physical Volume:

```
root # vgcreate nfs /dev/drbd/by-res/nfs
```

3. Create Logical Volumes in the Volume Group. This example assumes two Logical Volumes of 20 GB each, named sales and engineering:

```
root # lvcreate -n sales -L 20G nfs  
root # lvcreate -n engineering -L 20G nfs
```

4. Activate the Volume Group and create file systems on the new Logical Volumes. This example assumes ext3 as the file system type:

```
root # vgchange -ay nfs  
root # mkfs -t ext3 /dev/nfs/sales  
root # mkfs -t ext3 /dev/nfs/engineering
```

### 3.3 Initial Cluster Setup

Use the YaST cluster module for the initial cluster setup. The process is documented in *Book “Administration Guide” 3 “Installation and Basic Setup”3.5 “Manual Cluster Setup (YaST)”* and includes the following basic steps:

1. *Book “Administration Guide” 3 “Installation and Basic Setup”3.5.2 “Defining the Communication Channels”*
2. *Book “Administration Guide” 3 “Installation and Basic Setup”3.5.3 “Defining Authentication Settings”*
3. *Book “Administration Guide” 3 “Installation and Basic Setup”3.5.4 “Transferring the Configuration to All Nodes”*

Then start the OpenAIS/Corosync service as described in *Book “Administration Guide” 3 “Installation and Basic Setup”3.5.7 “Bringing the Cluster Online”*.

### 3.4 Creating a Basic Pacemaker Configuration

Configure a STONITH device as described in *Book “Administration Guide” 8 “Fencing and STONITH”*. Then use the following basic configuration.

For a highly available NFS server configuration that involves a 2-node cluster, you need to adjust the global cluster options `no-quorum-policy`: Must be set to `ignore`.

For more information about global cluster options, refer to Book “Administration Guide” 4 “Configuration and Administration Basics” 4.1 “Global Cluster Options”.

To adjust the options, open the CRM shell as `root` (or any non-`root` user that is part of the `haclient` group) and issue the following commands:

```
crm(live) configure
crm(live)configure# property no-quorum-policy="ignore"
crm(live)configure# rsc_defaults resource-stickiness="200"
crm(live)configure# commit
```

## 4 Initial Configuration with Hawk

This section describes the initial configuration of a highly available NFS export with Hawk.

### PROCEDURE 1: USING THE SETUP WIZARD

Hawk provides a wizard that guides you through all configuration steps of a selected template. Follow the instructions on the screen. If you need information about an option, click it to display a short help text in Hawk.



#### Note: Availability of Templates

Which templates are available to individual Hawk users may differ. The user's permissions to access templates can be regulated via ACL. See Book “Administration Guide” 9 “Access Control Lists” for details.

In the following procedure, we will use the wizard to configure an NFS server as example, which can be used as an NFS(v4/v3) fail-over server. The wizard relies on the cluster having been set up using the bootstrap scripts, so that key-based SSH access between nodes has been configured. You will be prompted for the following information:

- The `root` password for the machine that you are logged in to via Hawk. It needs to be the same as on all cluster nodes that Hawk needs to touch in order to modify their file systems.
- The ID of the base file system resource.

- Details for the NFSv4 file system root.
- Details for an NFSv3 export. A directory exported by an NFS server, which clients can integrate it into their system.
- A floating IP address.

The resulting Pacemaker configuration will contain the following resources:

#### **NFS Kernel Server**

Manages the in-kernel Linux NFS daemon that serves locally mounted file systems to clients via the NFS network protocol.

#### **NFSv4 Virtual File System Root**

Manages the virtual NFS root export, needed for NFSv4 clients. This resource does not hold any actual NFS-exported data, merely the empty directory (`/srv/nfs`) that the other NFS export is mounted into.

#### **Non-root NFS Export**

Manages the NFSv3 export .

#### **Floating IP Address**

A virtual, floating cluster IP address, allowing NFS clients to connect to the service no matter which physical node it is running on.

1. Start a Web browser and log in to the cluster as described in Book “*Administration Guide*” 5 “*Configuring and Managing Cluster Resources (Web Interface)*” 5.1.1 “*Starting Hawk and Logging In*”.
2. In the left navigation bar, select *Setup Wizard*. The *Cluster Setup Wizard* lists available resource templates. If you click an entry, Hawk displays a short help text about the template.



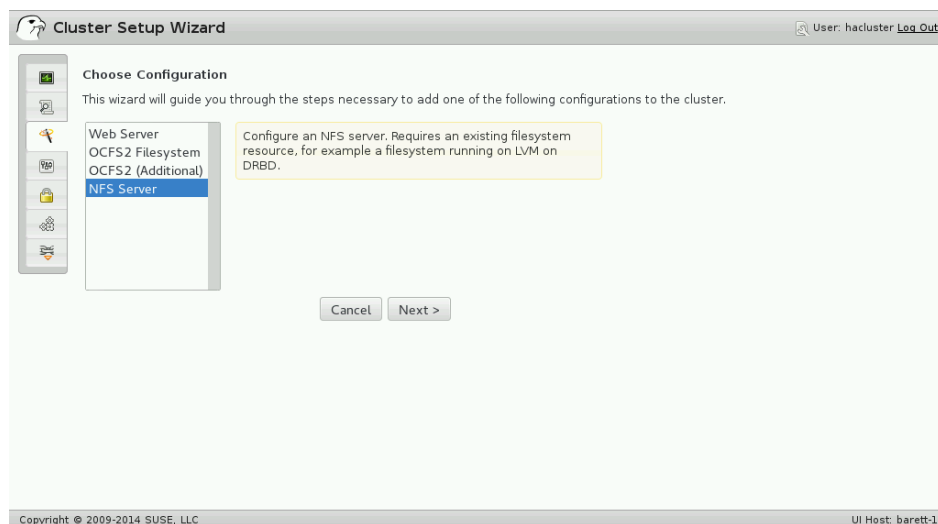


FIGURE 1: HAWK—SETUP WIZARD

3. Select the template for the resource you want to configure (in our case: NFS Server) and click *Next*.
4. To configure a highly available NFS server, proceed as follows:
  - a. Enter the root password for the current machine and click *Next*. Without the root password, the configuration wizard is not able to do the necessary configuration changes.
  - b. In the next screen, specify the *Base Filesystem* that is to be exported via NFS by entering its root ID. Click *Next*.
  - c. In the next screen, enter the details for the virtual NFSv4 file system root (needed for NFSv4 clients). Specify the following parameters and click *Next*.
    - Define a *Resource ID* to be used for this cluster resource.
    - Enter a *File System ID*. Hawk proposes 0 by default. The ID for the root file system must either be 0 or the string root.
    - Specify a *Mount Point*, for example: /srv/nfs.

- Enter a *Client Spec* for client access. For example 10.9.9.0/255.255.255.0. If you keep the value \*, which is proposed by Hawk, this would mean to allow all clients from everywhere.
  - Specify the *Mount Options*. For the NFSv4 file system root, Hawk proposes: rw,crossmnt.
- d. In the next screen, enter the details for the exported NFS mount point. Specify the following parameters and click *Next*.
- Define a *Resource ID* to be used for this cluster resource.
  - Enter a *File System ID*. Hawk proposes 1 by default, as the ID for NFS exports that do *not* represent an NFSv4 virtual file system root must be set to a unique positive integer, or a UUID string (32 hexadecimal digits with arbitrary punctuation).
  - Specify a *Mount Point*, for example: /srv/nfs/example.
  - Enter a *Client Spec* for client access. For example 10.9.9.0/255.255.255.0. If you keep the value \*, which is proposed by Hawk, this would mean to allow all clients from everywhere.
  - Specify the *Mount Options*. For the NFSv3 export, Hawk proposes: rw,mountpoint.
- e. In the next screen, configure a virtual IP added used to access the NFS mounts. Specify the following parameters.
- Define a *Resource ID* to be used for this cluster resource.
  - Enter an *IP Address* in dotted quad notation.
  - Optionally, enter a *Netmask*. If not specified, it will be determined automatically.
  - For LVS Direct Routing configuration, enable *LVS Support*. Otherwise leave it disabled.
- f. Click *Next*.
- The wizard displays the configuration snippet that will be applied to the CIB.



g. To apply it, click **Next**.

You have successfully configured an NFS(v4/v3) fail-over server.

1. Make sure you have executed all the following steps:
  - a. *Section 3.1, “Configuring a DRBD Resource”*
  - b. *Section 3.2, “Configuring LVM”*
  - c. *Section 3.3, “Initial Cluster Setup”*
2. Start a Web browser and log in to the cluster as described in Book “Administration Guide” 5 “Configuring and Managing Cluster Resources (Web Interface)” 5.1.1 “Starting Hawk and Logging In”.
3. In the left navigation bar, select **Setup Wizard**.
4. Select **NFS Server** template and click **Next**.
5. Enter your root passphrase.
6. Specify ID of your base file system.
7. Enter NFS export options for the virtual file system root. You can leave most values as they are, except for the following important text box:

#### **Mount Point**

Enter the mount point for your virtual file system root.

### *Mount options*

Add any additional mount options here.

8. Enter NFS export options for your mount point. Make sure the value in the text box *File system ID* is different than the root file system from the previous step.
9. Configure Virtual IP address and enter your virtual IP address.
10. Confirm the configuration.

## 5 Cluster Resources for an HA NFS Server

A highly available NFS service consists of the following cluster resources:

### *DRBD Primitive, and Master/Slave Resources*

These resources are used to replicate data. The master/slave resource is switched from and to the Primary and Secondary roles as deemed necessary by the cluster resource manager.

### *NFS Kernel Server Resource*

With this resource, Pacemaker ensures that the NFS server daemons are always available.

### *LVM and File System Resources*

The LVM Volume Group is made available on whichever node currently holds the DRBD resource in the Primary role. Apart from that, you need resources for one or more file systems residing on any of the Logical Volumes in the Volume Group. They are mounted by the cluster manager wherever the Volume Group is active.

### *NFSv4 Virtual File System Root*

A virtual NFS root export. (Only needed for NFSv4 clients).

### *Non-root NFS Exports*

One or more NFS exports, typically corresponding to the file system mounted from LVM Logical Volumes.

### *Resource for Floating IP Address*

A virtual, floating cluster IP address, allowing NFS clients to connect to the service no matter which physical node it is running on.

How to configure these resources (using the `crm` shell) is covered in detail in the following sections.

## EXAMPLE 1: NFS SCENARIO

The following configuration examples assume that 10.9.9.180 is the virtual IP address to use for an NFS server which serves clients in the 10.9.9.0/24 subnet.

The service is to host an NFSv4 virtual file system root hosted from /srv/nfs, with exports data served from /srv/nfs/sales and /srv/nfs/engineering.

Into these export directories, the cluster will mount ext3 file systems from Logical Volumes named sales and engineering, respectively. Both of these Logical Volumes will be part of a highly available Volume Group, named nfs, which is hosted on a DRBD device.

## 5.1 DRBD Primitive, and Master/Slave Resources

To configure these resources, issue the following commands from the crm shell:

```
crm(live)# configure
crm(live)configure# primitive p_drbd_nfs \
    ocf:linbit:drbd \
    params drbd_resource="nfs" \
    op monitor interval="15" role="Master" \
    op monitor interval="30" role="Slave"
crm(live)configure# ms ms_drbd_nfs p_drbd_nfs \
    meta master-max="1" master-node-max="1" clone-max="2" \
    clone-node-max="1" notify="true"
crm(live)configure# commit
```

This will create a Pacemaker Master/Slave resource corresponding to the DRBD resource nfs. Pacemaker should now activate your DRBD resource on both nodes, and promote it to the Master role on one of them.

Check this with the crm\_mon command, or by looking at the contents of /proc/drbd.

## 5.2 NFS Kernel Server Resource

In the crm shell, the resource for the NFS server daemons must be configured as a *clone* of an lsb resource type, as follows:

```
crm(live)configure# primitive p_nfsserver \
```

```
ocf:heartbeat:nfsserver \  
op monitor interval="30s"  
crm(live)configure# clone cl_ocf_heartbeat_nfsserver p_nfsserver  
crm(live)configure# commit
```



### Note: Resource Type Name and NFS Server init Script

The name of the `ocf` resource type must be identical to the file name of the NFS server init script, installed under `/etc/init.d`. SUSE Linux Enterprise Server ships the init script as `/etc/init.d/nfsserver` (package `nfs-kernel-server`). Hence the resource must be of type `ocf:heartbeat:nfsserver`.

After you have committed this configuration, Pacemaker should start the NFS Kernel server processes on both nodes.

## 5.3 LVM and File System Resources

1. Configure LVM and file system type resources as follows (but do *not commit* this configuration yet):

```
crm(live)configure# primitive p_lvm_nfs \  
ocf:heartbeat:LVM \  
params volgrpname="nfs" \  
op monitor interval="30s"  
crm(live)configure# primitive p_fs_engineering \  
ocf:heartbeat:Filesystem \  
params device=/dev/nfs/engineering \  
directory=/srv/nfs/engineering \  
fstype=ext3 \  
op monitor interval="10s"  
crm(live)configure# primitive p_fs_sales \  
ocf:heartbeat:Filesystem \  
params device=/dev/nfs/sales \  
directory=/srv/nfs/sales \  
fstype=ext3 \  

```

```
op monitor interval="10s"
```

2. Combine these resources into a Pacemaker resource *group*:

```
crm(live)configure# group g_nfs \  
p_lvm_nfs p_fs_engineering p_fs_sales
```

3. Add the following constraints to make sure that the group is started on the same node where the DRBD Master/Slave resource is in the Master role:

```
crm(live)configure# order o_drbd_before_nfs inf: \  
ms_drbd_nfs:promote g_nfs:start  
crm(live)configure# colocation c_nfs_on_drbd inf: \  
g_nfs ms_drbd_nfs:Master
```

4. Commit this configuration:

```
crm(live)configure# commit
```

After these changes have been committed, Pacemaker does the following:

- It activates all Logical Volumes of the nfs LVM Volume Group on the same node where DRBD is in the Primary role. Confirm this with vgdisplay or lvs.
- It mounts the two Logical Volumes to /srv/nfs/sales and /srv/nfs/engineering on the same node. Confirm this with mount (or by looking at /proc/mounts).

## 5.4 NFS Export Resources

Once your DRBD, LVM, and file system resources are working properly, continue with the resources managing your NFS exports. To create highly available NFS export resources, use the exportfs resource type.

### 5.4.1 NFSv4 Virtual File System Root

If clients exclusively use NFSv3 to connect to the server, you do not need this resource. In this case, continue with [Section 5.4.2, “Non-root NFS Exports”](#).

1. This is the root of the virtual NFSv4 file system.

```
crm(live)configure# primitive p_exportfs_root \
  ocf:heartbeat:exportfs \
  params fsid=0 \
    directory="/srv/nfs" \
    options="rw,crossmnt" \
    clientspec="10.9.9.0/255.255.255.0" \
  op monitor interval="30s"
crm(live)configure# clone cl_exportfs_root p_exportfs_root
```

This resource does not hold any actual NFS-exported data, merely the empty directory (/srv/nfs) that the other NFS exports are mounted into. Since there is no shared data involved here, we can safely *clone* this resource.

2. Since any data should be exported only on nodes where this clone has been properly started, add the following constraints to the configuration:

```
crm(live)configure# order o_root_before_nfs inf: \
  cl_exportfs_root g_nfs:start
crm(live)configure# colocation c_nfs_on_root inf: \
  g_nfs cl_exportfs_root
crm(live)configure# commit
```

After this, Pacemaker should start the NFSv4 virtual file system root on both nodes.

3. Check the output of the **exportfs -v** command to verify this.

## 5.4.2 Non-root NFS Exports

All NFS exports that do *not* represent an NFSv4 virtual file system root must set the fsid option. The value is set to either a unique positive integer (as used in the example), or a UUID string (32 hex digits with arbitrary punctuation).

1. Create NFS exports with the following commands:

```
crm(live)configure# primitive p_exportfs_sales \
  ocf:heartbeat:exportfs \
  params fsid=1 \
```



```

    directory="/srv/nfs/sales" \
    options="rw,mountpoint" \
    clientspec="10.9.9.0/255.255.255.0" \
    wait_for_lease_time_on_stop=true \
    op monitor interval="30s"
crm(live)configure# primitive p_exportfs_engineering \
    ocf:heartbeat:exportfs \
    params fsid=2 \
    directory="/srv/nfs/engineering" \
    options="rw,mountpoint" \
    clientspec="10.9.9.0/255.255.255.0" \
    wait_for_lease_time_on_stop=true \
    op monitor interval="30s"

```

2. After you have created these resources, add them to the existing g\_nfs resource group:

```
crm(live)configure# edit g_nfs
```

3. Edit the group configuration so it looks like this:

```

group g_nfs \
    p_lvm_nfs p_fs_engineering p_fs_sales \
    p_exportfs_engineering p_exportfs_sales

```

4. Commit this configuration:

```
crm(live)configure# commit
```

Pacemaker will export the NFS virtual file system root and the two other exports.

5. Confirm that the NFS exports are set up properly:

```
root # exportfs -v
```

## 5.5 Resource for Floating IP Address

To enable smooth and seamless failover, your NFS clients will be connecting to the NFS service via a floating cluster IP address, rather than via any of the hosts' physical IP addresses.

1. Add the following resource to the cluster configuration:

```
crm(live)configure# primitive p_ip_nfs \  
    ocf:heartbeat:IPaddr2 \  
    params ip=10.9.9.180 \  
    cidr_netmask=24 \  
    op monitor interval="30s"
```

2. Add the IP address to the resource group (like you did with the exportfs resources):

```
crm(live)configure# edit g_nfs
```

This is the final setup of the resource group:

```
group g_nfs \  
    p_lvm_nfs p_fs_engineering p_fs_sales \  
    p_exportfs_engineering p_exportfs_sales \  
    p_ip_nfs
```

3. Complete the cluster configuration:

```
crm(live)configure# commit
```

At this point Pacemaker will set up the floating cluster IP address.

4. Confirm that the cluster IP is running correctly:

```
root # ip address show
```

The cluster IP should be added as a secondary address to whatever interface is connected to the 10.9.9.0/24 subnet.



## Note: Connection of Clients

SUSE Linux Enterprise does not support to make your NFS exports bind to *only* this cluster IP address. The Kernel NFS server always binds to the wild card address (0.0.0.0 for IPv4). However, your clients must connect to the NFS exports through the floating IP address *only*, otherwise the clients will suffer service interruptions on cluster failover.

## 6 Using the NFS Service

This section outlines how to use the highly available NFS service from an NFS client. It covers NFS clients using NFS versions 3 and 4.

To connect to the NFS service, make sure to use the *virtual IP address* to connect to the cluster, rather than a physical IP configured on one of the cluster nodes' network interfaces. NFS version 3 requires that you specify the *full* path of the NFS export on the server.

### Important: Configure Virtual File System for NFSv4

Connecting to the NFS server with NFSv4 *will not work* unless you have configured an NFSv4 virtual file system root. For details on how to set this up, see [Section 5.4.1, “NFSv4 Virtual File System Root”](#).

In its simplest form, the command to mount the NFS export with NFSv3 looks like this:

```
root # mount -t nfs 10.9.9.180:/srv/nfs/engineering /home/engineering
```

For selecting a specific transport protocol (proto) and maximum read and write request sizes (rsize and wsizes):

```
root # mount -t nfs -o proto=udp,rsize=32768,wsizes=32768 \
10.9.9.180:/srv/nfs/engineering /home/engineering
```

To connect to a highly available NFS service, NFSv4 clients must use the floating cluster IP address (as with NFSv3), rather than any of the physical cluster nodes' addresses. NFS version 4 requires that you specify the NFS export path *relative* to the root of the virtual file system. Thus, to connect to the engineering export, you would use the following mount command (note the nfs4 file system type):

```
root # mount -t nfs4 10.9.9.180:/engineering /home/engineering
```

For further NFSv3 and NFSv4 mount options, consult the nfs man page.

## 7 Legal Notice

Copyright © 2011–2017 SUSE LLC and contributors; doc-team@suse.de. Used with permission.

Copyright © 2010–2017 LINBIT HA-Solutions GmbH.

**Trademark notice.** DRBD® and LINBIT® are trademarks or registered trademarks of LINBIT in Austria, the United States, and other countries. Other names mentioned in this document may be trademarks or registered trademarks of their respective owners.

**License information.** The text and illustrations in this document are licensed under a Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported license ("CC BY-NC-ND-3.0").

A summary of CC BY-NC-ND-3.0 is available at <http://creativecommons.org/licenses/by-nc-nd/3.0/>.

The full license text is available at <http://creativecommons.org/licenses/by-nc-nd/3.0/legalcode>.

In accordance with CC BY-NC-ND-3.0, if you distribute this document, you must provide the URL for the original version: [http://links.linbit.com/tech\\_guides](http://links.linbit.com/tech_guides).