Hortonworks Data Platform

HDFS NFS Gateway User Guide

(June 1, 2017)

docs.cloudera.com

Hortonworks Data Platform: HDFS NFS Gateway User Guide

The Hortonworks Data Platform, powered by Apache Hadoop, is a massively scalable and 100% open source platform for storing, processing and analyzing large volumes of data. It is designed to deal with data from many sources and formats in a very quick, easy and cost-effective manner. The Hortonworks Data Platform consists of the essential set of Apache Hadoop projects including MapReduce, Hadoop Distributed File System (HDFS), HCatalog, Pig, Hive, HBase, ZooKeeper and Ambari. Hortonworks is the major contributor of code and patches to many of these projects. These projects have been integrated and tested as part of the Hortonworks Data Platform release process and installation and configuration tools have also been included.

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. The Hortonworks Data Platform is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. All of our technology is, and will remain free and open source.

Please visit the Hortonworks Data Platform page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the Support or Training page. Feel free to Contact Us directly to discuss your specific needs.

Table of Contents

1. HDFS NFS Gateway User Guide

The NFS Gateway for HDFS allows clients to mount HDFS and interact with it through NFS, as if it were part of their local file system. The gateway supports NFSv3.

After mounting HDFS, a user can:

- Browse the HDFS file system through their local file system on NFSv3 client-compatible operating systems.
- Upload and download files between the HDFS file system and their local file system.
- Stream data directly to HDFS through the mount point. (File append is supported, but random write is not supported.)

Prerequisites

The NFS Gateway machine must be running all components that are necessary for running an HDFS client, such as a Hadoop core JAR file and a HADOOP_CONF directory.

The NFS Gateway can be installed on any DataNode, NameNode, or HDP client machine. Start the NFS server on that machine.

1.1. Configure the HDFS NFS Gateway

To configure the HDFS NFS gateway, complete the following steps.

1. The user running the NFS gateway must be able to proxy all users that are using NFS mounts.

   For example, if user "nfsserver" is running the gateway and users belong to groups "nfs-users1" and "nfs-users2", then set the following values in the `core-site.xml` file on the NameNode.

   

   Note

   Replace "nfsserver" with the user account that will start the gateway in your cluster.

```
<property>
  <name>hadoop.proxyuser.nfsserver.groups</name>
  <value>nfs-users1,nfs-users2</value>
  <description>
    The 'nfsserver' user is allowed to proxy all members of the
    'nfs-users1' and 'nfs-users2' groups. Set this to '*' to allow
    nfsserver user to proxy any group.
  </description>
</property>

<property>
  <name>hadoop.proxyuser.nfsserver.hosts</name>
  <value>nfs-client-host1.com</value>
```

```
  <description>
    This is the host where the nfs gateway is running. Set this to
    '*' to allow requests from any hosts to be proxied.
  </description>
</property>
```

The preceding properties are the only required configuration settings for the NFS gateway in non-secure mode.

Configuring the HDFS NFS Gateway on a Kerberized Cluster

For a Kerberized cluster, set the following properties in the `hdfs-site.xml` file:

```
  <property>
    <name>dfs.nfsgateway.keytab.file</name>
    <value>/etc/hadoop/conf/nfsserver.keytab</value> <!-- path to the
      nfs gateway keytab -->
  </property>

  <property>
    <name>dfs.nfsgateway.kerberos.principal</name>
    <value>nfsserver/_HOST@YOUR-REALM.COM</value>
  </property>
```

2. Configure settings for the HDFS NFS gateway:

   The NFS gateway uses the same settings that are used by the NameNode and DataNode. Configure the following properties based on your application's requirements:

   a. Edit the `hdfs-site.xml` file on your NFS gateway machine. Modify the following property:

   ```
   <property>
     <name>dfs.namenode.accesstime.precision</name>
     <value>3600000</value>
     <description>
       The access time for HDFS file is precise up to this value.
       The default value is 1 hour. Setting a value of 0 disables
       access times for HDFS.
     </description>
   </property>
   ```

   

   Note

   If the export is mounted with access time update allowed, make sure this property is not disabled in the configuration file. Only NameNode needs to restart after this property is changed. If you have disabled access time update by mounting with "noatime" you do NOT have to change this property nor restart your NameNode.

   b. Add the following property to the `hdfs-site.xml` file:

   ```
   <property>
       <name>dfs.nfs3.dump.dir</name>
       <value>/tmp/.hdfs-nfs</value>
   </property>
   ```

Note

The NFS client often reorders writes. Sequential writes can arrive at the NFS gateway at random order. This directory is used to temporarily save out-of-order writes before writing to HDFS. One needs to make sure the directory has enough space. For example, if the application uploads 10 files with each having 100MB, it is recommended for this directory to have 1GB space in case if a worst-case write reorder happens to every file.

c. Update the following property in the `hdfs-site.xml` file:

```
<property>
    <name>dfs.nfs.exports.allowed.hosts</name>
    <value>* rw</value>
</property>
```

Note

By default, the export can be mounted by any client. You must update this property to control access. The value string contains the machine name and access privilege, separated by whitespace characters. The machine name can be in single host, wildcard, or IPv4 network format. The access privilege uses `rw` or `ro` to specify `readwrite` or `readonly` access to exports. If you do not specifiy an access privilege, the default machine access to exports is `readonly`. Separate machine dentries by `;`. For example, `192.168.0.0/22 rw ; host*.example.com ; host1.test.org ro;`.

Restart the NFS gateway after this property is updated.

d. (Optional) Customize log settings by modifying the `log4j.property` file.

To change trace level, add the following:

```
log4j.logger.org.apache.hadoop.hdfs.nfs=DEBUG
```

To view more information about ONCRPC requests, add the following:

```
log4j.logger.org.apache.hadoop.oncrpc=DEBUG
```

3. Specify JVM heap space (HADOOP_NFS3_OPTS) for the NFS Gateway. You can increase the JVM heap allocation for the NFS gateway using this option.

To set this option, specify the following in the `hadoop-env.sh` file:

```
export HADOOP_NFS3_OPTS=<memory-setting(s)>
```

The following example specifies a 2 GB process heap (2GB starting size and 2GB maximum):

```
export HADOOP_NFS3_OPTS="-Xms2048m -Xmx2048m"
```

4. The dfs.nfs.rtmax and dfs.nfs.wtmax properties are HDFS configuration settings on the HDFS NFS gateway server. These options change the maximum read and write request size supported by the gateway. The default value for both settings is 1 MB. Increasing these values might improve the performance of large file transfers. The defaults are expected to work well for most deployments.

1.2. Start and Verify the NFS Gateway Service

Three daemons are required to provide NFS service: `rpcbind` (or `portmap`), `mountd` and `nfsd`. The NFS gateway process has both `nfsd` and `mountd`. It shares the HDFS root "/" as the only export.

We recommend using the `portmap` included in NFS gateway package, as shown below. The included `portmap` must be used on some Linux systems, for example, SLES 11 and RHEL 6.2.

1. Stop `nfs/rpcbind/portmap` services provided by the platform:

   ```
   service nfs stop

   service rpcbind stop
   ```

2. Start the included `portmap` package (needs root privileges), using one of the following two commands:

   ```
   hadoop portmap
   ```

   OR

   ```
   hadoop-daemon.sh start portmap
   ```

3. Start `mountd` and `nfsd`.

   No root privileges are required for this command. However, verify that the user starting the Hadoop cluster and the user starting the NFS gateway are the same.

   ```
   hadoop nfs3
   ```

   OR

   ```
   hadoop-daemon.sh start nfs3
   ```

   

   Note

   If the `hadoop-daemon.sh` script starts the NFS gateway, its log file can be found in the hadoop log folder (`/var/log/hadoop`).

   For example, if you launched the NFS gateway services as the root user, the log file would be found in a path similar to the following:

   ```
   /var/log/hadoop/root/hadoop-root-nfs3-63ambarihdp21.log
   ```

4. Stop the NFS gateway services.

```
hadoop-daemon.sh stop nfs3

hadoop-daemon.sh stop portmap
```

Next, verify the validity of NFS-related services.

1. Execute the following command to verify that all the services are up and running:

   ```
   rpcinfo -p $nfs_server_ip
   ```

   You should see output similar to the following:

   ```
     program vers proto   port
      100005    1   tcp   4242  mountd
      100005    2   udp   4242  mountd
      100005    2   tcp   4242  mountd
      100000    2   tcp    111  portmapper
      100000    2   udp    111  portmapper
      100005    3   udp   4242  mountd
      100005    1   udp   4242  mountd
      100003    3   tcp   2049  nfs
      100005    3   tcp   4242  mountd
   ```

2. Verify that the HDFS namespace is exported and can be mounted:

   ```
   showmount -e $nfs_server_ip
   ```

   You should see output similar to the following:

   ```
   Exports list on $nfs_server_ip :
   / (everyone)
   ```

1.3. Access HDFS

To access HDFS, first mount the export "/". Currently NFS v3 is supported. It uses TCP, as the transportation protocol is TCP.

1. Mount the HDFS namespace as follows:

   ```
   mount -t nfs -o
   vers=3,proto=tcp,nolock,sync,rsize=1048576,wsize=1048576
   $server:/ $mount_point
   ```

   Access HDFS as part of the local file system, except that hard/symbolic link and random write are not supported in this release.

Note

Because NLM is not supported, the mount option `nolock` is needed.

Use the `sync` option for performance when writing large files. The `sync` mount option to the NFS client improves the performance and reliability of writing large files to HDFS using the NFS gateway. If the sync option is specified, the NFS client machine flush writes operations to the NFS gateway

before returning control to the client application. A useful side effect of sync is that the client does not issue reordered writes. This reduces buffering requirements on the NFS gateway.

`sync` is specified on the client machine when mounting the NFS share.

User authentication and mapping:

NFS gateway uses `AUTH_UNIX`-style authentication, which means that the the login user on the client is the same user that NFS passes to the HDFS. For example, if the NFS client has current user as `admin`, when the user accesses the mounted directory, NFS gateway will access HDFS as user `admin`. To access HDFS as `hdfs` user, you must first switch the current user to `hdfs` on the client system before accessing the mounted directory.

2. Set up client machine users to interact with HDFS through NFS.

   The NFS gateway converts the User Identifier (UID) to username, and HDFS uses username to check permissions.

   The system administrator must ensure that the user on NFS client machine has the same name and UID as that on the NFS gateway machine. This is usually not a problem if you use the same user management system such as LDAP/NIS to create and deploy users to HDP nodes and to client node.

   If the user is created manually, you might need to modify the UID on either the client or NFS gateway host in order to make them the same:

   ```
   usermod -u 123 $myusername
   ```

   The following diagram illustrates how the UID and name are communicated between the NFS client, NFS gateway, and NameNode.