HDFS

HDFS is a distributed file-system, part of the Apache Hadoop framework.

Paths are specified as `remote:` or `remote:path/to/dir`.

Here is an example of how to make a remote called `remote`. First run:

```
rclone config
```

This will guide you through an interactive setup process:

```
No remotes found - make a new one
n) New remote
s) Set configuration password
q) Quit config
n/s/q> n
name> remote
Type of storage to configure.
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value
[skip]
XX / Hadoop distributed file system
   \ "hdfs"
[skip]
Storage> hdfs
** See help for hdfs backend at: https://rclone.org/hdfs/ **

hadoop name node and port
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value
 1 / Connect to host namenode at port 8020
   \ "namenode:8020"
namenode> namenode.hadoop:8020
hadoop user name
Enter a string value. Press Enter for the default ("").
Choose a number from below, or type in your own value
 1 / Connect to hdfs as root
   \ "root"
username> root
Edit advanced config? (y/n)
y) Yes
n) No (default)
y/n> n
Remote config
--------------------
[remote]
type = hdfs
namenode = namenode.hadoop:8020
username = root
--------------------
y) Yes this is OK (default)
e) Edit this remote
d) Delete this remote
y/e/d> y
Current remotes:

Name                 Type
====                 ====
hadoop               hdfs

e) Edit existing remote
n) New remote
```

```
d) Delete remote
r) Rename remote
c) Copy remote
s) Set configuration password
q) Quit config
e/n/d/r/c/s/q> q
```

This remote is called `remote` and can now be used like this

See all the top level directories

```
rclone lsd remote:
```

List the contents of a directory

```
rclone ls remote:directory
```

Sync the remote `directory` to `/home/local/directory`, deleting any excess files.

```
rclone sync -i remote:directory /home/local/directory
```

Setting up your own HDFS instance for testing

You may start with a manual setup or use the docker image from the tests:

If you want to build the docker image

```
git clone https://github.com/rclone/rclone.git
cd rclone/fstest/testserver/images/test-hdfs
docker build --rm -t rclone/test-hdfs .
```

Or you can just use the latest one pushed

```
docker run --rm --name "rclone-hdfs" -p 127.0.0.1:9866:9866 -p 127.0.0.1:8020:8020 --hostname
```

NB it need few seconds to startup.

For this docker image the remote needs to be configured like this:

```
[remote]
type = hdfs
namenode = 127.0.0.1:8020
username = root
```

You can stop this image with `docker kill rclone-hdfs` (**NB** it does not use volumes, so all data uploaded will be lost.)

Modified time

Time accurate to 1 second is stored.

Checksum

No checksums are implemented.

Usage information

You can use the `rclone about remote:` command which will display filesystem size and current usage.

Restricted filename characters

In addition to the default restricted characters set the following characters are also replaced:

Character	Value	Replacement
:	0x3A	:

Invalid UTF-8 bytes will also be replaced.

Limitations

- No server-side `Move` or `DirMove`.
- Checksums not implemented.

Standard Options

Here are the standard options specific to hdfs (Hadoop distributed file system).

--hdfs-namenode

hadoop name node and port

- Config: namenode
- Env Var: RCLONE_HDFS_NAMENODE
- Type: string
- Default: ""
- Examples:
  - "namenode:8020"
    - Connect to host namenode at port 8020

--hdfs-username

hadoop user name

- Config: username
- Env Var: RCLONE_HDFS_USERNAME
- Type: string
- Default: ""
- Examples:
    - "root"
        - Connect to hdfs as root

Advanced Options

Here are the advanced options specific to hdfs (Hadoop distributed file system).

--hdfs-service-principal-name

Kerberos service principal name for the namenode

Enables KERBEROS authentication. Specifies the Service Principal Name (/) for the namenode.

- Config: service_principal_name
- Env Var: RCLONE_HDFS_SERVICE_PRINCIPAL_NAME
- Type: string
- Default: ""
- Examples:
    - "hdfs/namenode.hadoop.docker"
        - Namenode running as service 'hdfs' with FQDN 'namenode.hadoop.docker'.

--hdfs-data-transfer-protection

Kerberos data transfer protection: authentication|integrity|privacy

Specifies whether or not authentication, data signature integrity checks, and wire encryption is required when communicating the the datanodes. Possible values are 'authentication', 'integrity' and 'privacy'. Used only with KERBEROS enabled.

- Config: data_transfer_protection
- Env Var: RCLONE_HDFS_DATA_TRANSFER_PROTECTION
- Type: string
- Default: ""
- Examples:
    - "privacy"
        - Ensure authentication, integrity and encryption enabled.

--hdfs-encoding

This sets the encoding for the backend.

See: the encoding section in the overview for more info.

- Config: encoding
- Env Var: RCLONE_HDFS_ENCODING

- Type: MultiEncoder
- Default: Slash,Colon,Del,Ctl,InvalidUtf8,Dot