Duply

Installation

```
sudo apt-get install duply python-paramiko trickle
```

Configuration

Create backup profile:

```
duply gitlab create
```

Profile file `$HOME/.duply/gitlab/conf` was created.

```
IMPORTANT
     Copy the whole profile folder after the first backup to a safe place.
It contains everything needed to restore your backups. You will need it if
you have to restore the backup from another system
     (e.g. after a system crash). Keep access to these files restricted as
they contain all information (gpg data, ftp data) to access and modify your
backups.
```

```
     Repeat this step after all configuration changes. Some configuration
options are crucial for restoration.
```

Generate random password:

```
openssl rand -base64 20
```

| ~/.duply/gitlab/conf

```
#GPG_KEY=
GPG_PW='<generated passsword>'
```

Configure backup section:

| ~/.duply/gitlab/conf

```
# Paramiko SSH is very CPU consuming
#TARGET='scp://gitlabbackup@192.168.0.230//mnt/backup/gitlabbackup'

TARGET='sftp://gitlabbackup@192.168.0.230//mnt/backup/gitlabbackup'
```

```
# Limit network speed
DUPL_PRECMD="trickle -s -u 1500 -d 256"

SOURCE='/'
MAX_AGE=6M
MAX_FULL_BACKUPS=2
MAX_FULLS_WITH_INCRS=2

MAX_FULLBKP_AGE=3M
DUPL_PARAMS="$DUPL_PARAMS --full-if-older-than $MAX_FULLBKP_AGE "


VOLSIZE=256
DUPL_PARAMS="$DUPL_PARAMS --volsize $VOLSIZE "

VERBOSITY=4
TEMP_DIR=/tmp

# Specify different id_rsa file:
DUPL_PARAMS="$DUPL_PARAMS --ssh-options=-
oIdentityFile='/root/.duply/gitlab/id_rsa' "
DUPL_PARAMS="$DUPL_PARAMS --no-compression " # Do not use GZip to
compress files on remote system.
DUPL_PARAMS="$DUPL_PARAMS --ssh-options=-carcfour128 "  # light cipher
for NAS
DUPL_PARAMS="$DUPL_PARAMS --ssh-options=-oCompression=no "  # disable
ssh compression
DUPL_PARAMS="$DUPL_PARAMS --asynchronous-upload " # uploads in
background
```

~/.duply/gitlab/exclude

```
+ /etc/gitlab
+ /opt
+ /home
+ /root
- **
```

Note: duply dosn't make any cleanup or deletions during `backup` action. So destination storage can be full very quickly. To perform maintenance of old backup accordign to `MAX_AGE`, `MAX_FULL_BACKUPS`, `MAX_FULLS_WITH_INCRS` and `MAX_FULLBKP_AGE` parameters it is need to call duply with `purge` and `cleanup` commands. See cron script example below.

Example options:

- MAX_FULL_BACKUPS=2
- MAX_FULLS_WITH_INCRS=1

Will keep 2 full backup sets, but only one with increments (last one).

Sometimes it is good to check whether incremental backups are meaningful (it depends on type of data stored). If command

```
duply gitlab status
```

shows that number of volume with each increment is similar to full backup, then making increments has no sense, ane time to keep increments can be short e.g. MAX_FULLBKP_AGE=7D

Usage

Start the backup

```
sudo duply gitlab backup --progress
```

```
duply gitlab status
duply gitlab list

duply gitlab verify  # long operation
```

cron script

```
#!/bin/bash -ue
set -o pipefail
trap "banner error; echo LINE: $LINENO" ERR

duply gitlab backup
duply gitlab purge --force # list outdated backup archives and delete them
duply gitlab-to-grinnux purgeIncr --force
duply gitlab-to-grinnux purgeFull --force
duply gitlab cleanup --extra-clean --force > /dev/null # list broken backup
files and delete them
banner ALL OK
```

shell function

```
#!/bin/bash -ueE
set -o pipefail
trap "banner error; echo LINE: $LINENO" ERR

run_duply() {
    echo "====================================================="
    duply ${1} backup
```

```
    echo "======================================================"
    duply ${1} cleanup --extra-clean --force
    duply ${1} purge --force
    duply ${1} purgeIncr --force
    duply ${1} purgeFull --force
    echo "======================================================"
    duply ${1} cleanup --extra-clean --force > /dev/null
    echo "======================================================"
    banner ${1} OK
}
```

SFTP and rbash

/etc/passwd

```
mybackup:x:1002:1002:Backup,,,:/home/mybackup:/bin/rbash
```

/etc/ssh/sshd_config

```
Subsystem sftp internal-sftp
```

Verbosity

VERBOSITY=5

Lots of info

```
Ignoring incremental Backupset (start_time: Fri Aug  5 13:49:11 2016;
needed: Fri Jun 17 14:20:07 2016)
Ignoring incremental Backupset (start_time: Fri Aug  5 13:49:11 2016;
needed: Thu Jul 21 14:33:52 2016)
Added incremental Backupset (start_time: Fri Aug  5 13:49:11 2016 /
end_time: Wed Aug 10 14:54:49 2016)
Ignoring incremental Backupset (start_time: Wed Aug 10 14:54:49 2016;
needed: Fri Jun 17 14:20:07 2016)
```

VERBOSITY=4

Usefull report:

```
-------------[ Backup Statistics ]--------------
```

```
StartTime 1479517583.39 (Sat Nov 19 01:06:23 2016)
EndTime 1479517751.96 (Sat Nov 19 01:09:11 2016)
ElapsedTime 168.57 (2 minutes 48.57 seconds)
SourceFiles 41
SourceFileSize 13621422991 (12.7 GB)
NewFiles 1
NewFileSize 4096 (4.00 KB)
DeletedFiles 0
ChangedFiles 1
ChangedFileSize 13621360640 (12.7 GB)
ChangedDeltaSize 0 (0 bytes)
DeltaEntries 2
RawDeltaSize 6101758 (5.82 MB)
TotalDestinationSizeChange 5214178 (4.97 MB)
Errors 0
-------------------------------------------------
```

Issues

no acceptable kex algorithm

ssh: Exception: Incompatible ssh peer (no acceptable kex algorithm)

Python paramiko module needs upgrade

```
apt-get install python-pip python-dev python-cffi libffi-dev build-essential
pip install --ugprade cffi
pip install pycparser==2.13
pip install --ugprade cryptography
```

To solve error “AssertionError: sorry, but this version only supports 100 named groups” please install

```
pip install pycparser==2.13
```

```
pip install --ugprade paramiko
```

can't be deleted

```
 duply mybackup purge --force
```

```
Last full backup date: Wed May 24 01:11:54 2017
There are backup set(s) at time(s):
Thu Nov 24 01:05:26 2016
Fri Nov 25 01:09:43 2016
Sat Nov 26 01:10:50 2016
Which can't be deleted because newer sets depend on them.
```

```
No old backup sets found, nothing deleted.
```

Solution is to run:

```
duply mybackup purgeIncr --force
```

```
duply mybackup purgeFull --force
```