install and configure borgbackup

Storage Box · https://www.hyyat.com/en/knowledgebase/install-and-configure-borgbackup/

Introduction

BorgBackup (short: Borg) is a deduplicating backup program. Compression and authenticated encryption are also supported as options.

Borg's main goal is to provide an efficient and secure backup solution. Thanks to deduplication, the backup process with Borg is very fast and makes Borg very interesting for daily backups. You may notice that Borg is significantly quicker than some other methods, depending on the amount of data and the number of changes you need to back up. With Borg, all data is already encrypted on the client side, which makes Borg a good choice for hosted systems.

More information about BorgBackup can be found on the official website.

Step 1 - Installation

There are three ways to install Borg.

- Distribution package
- Standalone binary
- From source

In the Borg Documentation you will find very detailed descriptions of the different methods. That's why we do not go into detail here.

For compatibility reasons, please use a current version of Borg! (> = 1.0.9)

Step 2 - Workflow with Borg

Step 2.1 - Activate Borg and configure your storage box

For Borg to be enabled on your storage box, you must first enable the service on the Robot webinterface. To do this, go to the settings page of your storage box in Robot and click on activate in "SSH support".

Since SSH is not available on the storage box, you must use SFTP or something similar for this step.

For Borg, you can use password authentication, but authentication via the public key is recommended. This is especially recommended if you want to automate the backups with cronjobs.

To use Borg your SSH key is not (!) required in the RFC4716 format, like with SFTP/SCP. You need to store your normal public key. If you use both Borg and SFTP/SCP, then both keys (RFC4716 format and normal) need to be stored.

Create the folder '.ssh' in your storage box and store the file 'authorized_keys' in it. This must contain your public key:

```
ssh-rsa AAAAB3NzaC1yc2EAAAA.......rdj7eitNUjlIV8ovvAH/6SAsKD6
```

Set the permissions for the '.ssh' folder to '0700' and for the 'authorized_keys' to '0600'.

Your home directory on your storage box / backup space is not allowed to have write permissions for Group and Others, otherwise authenticating via keyfile is not possible. By default this is set, but it can be changed.

Now you have to create the directory for the backup repository in the storage box. For example, create a folder 'backups', and below that, a folder 'server1'. The folder 'server1' will then be initialized as a Borg repository in the next step. Under 'backups' you could then create further directories for other servers you want to back up.

```
/backups/server1
```

Step 2.2 - Initialize Borg repository

If you are using an SSH key, and this is not the default key, you have the option to specify the desired key using the environment variable BORG_RSH. You can specify the SSH command that Borg should use. The standard would be just 'ssh'.

```
$ export BORG_RSH='ssh -i /home/userXY/.ssh/id_ed25519'
```

When initializing Borg, you will be prompted for a password for your repository. Only with this password can you access the repository in the future. It is therefore required for every read or write operation on the repository. You must be able to remember the password well because it cannot be restored! To avoid having to enter the password every time Borg calls, you can optionally set the environment variable BORG_PASSPHRASE.

```
$ export BORG_PASSPHRASE="top_secret_passphrase"
```

First, you need to initialize the Borg repository. The repository is nothing more than a folder on your storage box that Borg provides with some basic structures. All backups are stored in this folder.

The following command initializes the '/backups/server1' folder on your storage box.

```
$ borg init --encryption=repokey ssh://u123456@u123456.your-storagebox.de:23/./backups/server1
```

Step 2.3 - Create first backup

For example, use the following command to back up the 'src' and 'build' folders from your home directory to the repository on your storage box. You must give each backup a unique name. A timestamp is useful for creating unique names.

```
$ borg create ssh://u123456@u123456.your-storagebox.de:23/./backups/server1::2017_11_11_initial
~/src ~/built
```

You can call Borg create using many other options. You can do this, for example, to view the progress of a backup while it is processing or to see statistics about the backup once it is finished. In addition, you can specify exclude patterns and other things.

For more information, please visit the Borg create documentation.

Step 2.4 - Following (incremental) backups

The following backups are identical to the first one. Thanks to deduplication, however, they are much faster and extremely memory-efficient, since they are only incremental.

You only need to adjust the name of the backup during the follow-up backup. Remember, you must use unique names as mentioned above.

Just use the '--stats' option on the next backup to see how efficient it is.

```
$ borg create --stats ssh://u123456@u123456.your-storagebox.de:23/./backups/server1::2017_11_12 ~/src ~/built
```

Step 2.5 - More Borg commands including List archives, restore backups

The Borg documentation provides a very detailed description of all Borg commands.

It is best to start with a look at the quickstart section and then dive into the usage section to get into the details.

The documentation provides many examples of listing archives or restoring backups. It is also possible, for example, to display diffs between backups or to delete old backups to recover storage space.

Step 2.6 - Automate backups with Cron

Create a directory for the log file.

```
$ mkdir -p /var/log/borg
```

First, create a script which will execute the backups. This could look like the following script and be under `/usr/local/bin/backup.sh`.

```
#!/usr/bin/env bash

##
## Set environment variables
##

## if you don't use the standard SSH key,
## you have to specify the path to the key like this
# export BORG_RSH="ssh -i /home/userXY/.ssh/id_ed25519"

## You can save your borg passphrase in an environment
## variable, so you don't need to type it in when using borg
# export BORG_PASSPHRASE="top_secret_passphrase"

##
## Set some variables
##

LOG="/var/log/borg/backup.log"
BACKUP_USER="u602"
REPOSITORY_DIR="server1"

## Tip: If using with a Backup Space you have to use
## 'your-storagebox.de' instead of 'your-backup.de'

REPOSITORY="ssh://${BACKUP_USER}@${BACKUP_USER}.your-storagebox.de:23/./backups/${REPOSITORY_DIR}"

##
## Output to a logfile
##

exec > >(tee -i ${LOG})
exec 2>&1

echo "###### Backup started: $(date) ######"

##
## At this place you could perform different tasks
## that will take place before the backup, e.g.
##
## - Create a list of installed software
```

```
## - Create a database dump
##

##
## Transfer the files into the repository.
## In this example the folders root, etc,
## var/www and home will be saved.
## In addition you find a list of excludes that should not
## be in a backup and are excluded by default.
##

echo "Transfer files ..."
borg create -v --stats                   \
    $REPOSITORY::'{now:%Y-%m-%d_%H:%M}'  \
    /root                                \
    /etc                                 \
    /var/www                             \
    /home                                \
    --exclude /dev                       \
    --exclude /proc                      \
    --exclude /sys                       \
    --exclude /var/run                   \
    --exclude /run                       \
    --exclude /lost+found                \
    --exclude /mnt                       \
    --exclude /var/lib/lxcfs

echo "###### Backup ended: $(date) ######"
```

Now test the script before you create the cronjob.

```
$ chmod u+x /usr/local/bin/backup.sh
$ /usr/local/bin/backup.sh
```

If everything works fine, you can now run the script as a cronjob. Open crontab as root:

`crontab -e`

And add the following line to run a daily backup at 00:00.

`0 0 * * * /usr/local/bin/backup.sh > /dev/null 2>&1`

Step 3 - Hints

Step 3.1 - Full system backup

If you want to backup the entire system on your Linux server, you should remember that not all files and folders belong in a backup. Some should be excluded by default.

For this, the create command has an `--exclude` option or you can specify an exclude file. The usage is described in detail in the Borg create documentation.

Here is an example call to `borg create` for a backup of the complete system:

```
borg create -v --stats                   \
    $REPOSITORY::'{now:%Y-%m-%d_%H:%M}'  \
    /                                    \
    --exclude /dev                       \
    --exclude /proc                      \
    --exclude /sys                       \
    --exclude /var/run                   \
    --exclude /run                       \
    --exclude /lost+found                \
    --exclude /mnt                       \
    --exclude /var/lib/lxcfs
```

Step 3.2 - Deduplication and reliability

Since BorgBackup uses deduplication, you can make backups very quickly and without using much storage.

But you also have to be aware that each file is saved exactly once. Should a file be damaged by a disk failure, for example, this file will be corrupted in all following backups.

Therefore, it is best practice to store very important data in more than one repository!

Step 3.3 - Borg version on the server

To avoid compatibility issues, it is recommended that you use the same version of Borg Backup on your server and on the Storage Box / Backup Space.

For each major update there is a version available, which is regularly and promptly updated by us. You can specify the version that you want to use on your Storage Box / Backup Space with the Borg --remote-path parameter. If the parameter is not specified, the latest version is used, which is available on the Storage Box / Backup Space.

Currently versions 1.0, 1.1 and 1.2 are installed. The latest version, so 1.2. is the default version. If you still want to use version 1.1, use:

```
$ borg init --encryption = repokey --remote-path=borg-1.1 ssh://u123456@u123456.your-storagebox.de:23/./backups/server1
```

borg-1.1 stands for version 1.1.x.

The changelog of the BorgBackup documentation provides information on the changes between versions and possible compatibility issues, if any.

Step 3.4 - Borg and SSH

BorgBackup uses SSH over port 23. SSH access is limited to Borg and login is not possible!

Step 3.5 - Use Borg and SFTP / SCP in parallel with keyfile

As described above, Borg requires the normal public key, while SFTP/SCP requires the SSH key in RFC4716 format. If you use both Borg and SFTP/SCP, both keys (RFC4716 format and normal) must be stored in the authorized_keys file.

Step 3.6 - Borg keyfile and password

The password you choose for your Borg repository will not be saved with us and can not be recovered by us! Keep it safe.

In repokey mode (default), the repo key is located in the repo config, i.e. on the storage box. It is recommended that you save a backup of the key. More information can be found in the Borg Documentation.

Conclusion

With Borgbackup you have installed and configured a space-saving and automated software for your backups.

Related service: Storage Box from Hyyat Host, with technical support that replies within 24 hours.

Related guides

- Access with WebDAV
- Access with SSH/rsync/BorgBackup
- Add additional users and sub-accounts
- Access with SFTP/SCP

Need this service from Hyyat Host?Our team handles it for you, with support replying to tickets within 24 hours at most.
Web HostingSaudi Hosting