Backing up Nextcloud

Consistent Backup and Restore of Complex Application Data

Sebastian Lederer, Heike Jurzik

Backing up Nextcloud

Consistent Backup and Restore of Complex Application Data

Sebastian Lederer, Heike Jurzik
dassIT GmbH, Bareos GmbH & Co. KG

January 21, 2019

This paper shows how to back up and restore complex application data. It uses Nextcloud as an example – a typical web application with a PHP web server and a database, so a setup that can serve as an example for many other applications.

Contents

1 Introduction 2

2 Backing up Applications 2

3 Planning Nextcloud Backups 3
- 3.1 Backup of Folders 3
- 3.2 Single Files 3
- 3.3 The Database 4

4 Bareos & Nextcloud 4
- 4.1 Shutting down the Application . . 4
- 4.2 Exporting the Application Data . . 4
  - 4.2.1 Using Bareos Directives . . 4
- 4.3 Streaming Data (`bpipe`) 5
- 4.4 Application Plugins 6
  - 4.4.1 `xtrabackup`, `bareos_percona` 6
  - 4.4.2 Bareos FD 6

5 Conclusion 7

6 About Bareos 7

7 About Nextcloud 7

1 Introduction

Administrators who want to use Bareos to back up and restore applications need to find a way to handle the data as well as the configuration. In case of more complex (server) applications there might even be a database that needs to be included in the backup/restore plan. It gets really challenging if the application offers 24/7 services to customers and a downtime results in losing money and/or clients.

This paper shows how to back up and restore Nextcloud, a client/server program suite that offers file hosting, calendar services, media streaming, document viewers, audio/video conferencing and more. All methods presented in this whitepaper can be used to back up other (complex) applications, generally without any downtime.

2 Backing up Applications

When planning application backups, it's important to keep in mind that they differ from file and image backups. Simple file backups can save documents, photos, and videos, but not the program that created them. An image backup produces an image of the entire OS instead, including files and applications. If a file backup is not enough and an image backup is too much, then a backup of the application might be the preferred approach.

Applications have an internal state that is stored in different places (see Figure 1). For example, parts of the data may be cached in the RAM, and the parts of the data will be written to

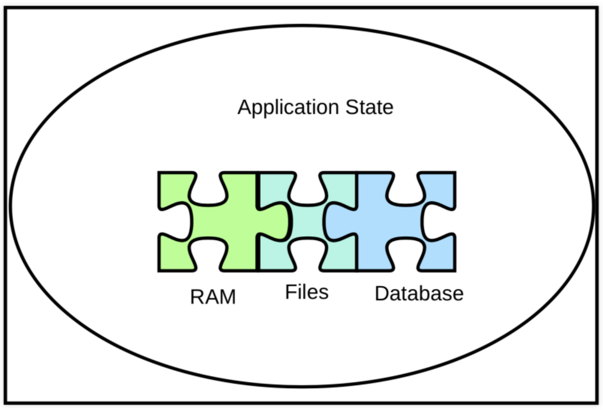


Figure 1: *Application State Puzzle*

a file later. Until that happens, the data in the file is incorrect. Sometimes an application stores information about files in a database table for faster access. Of course, the application might write data to a file at the same this file is being backed up. A backup plan needs to consider all puzzle pieces – after the recovery process they need to fit together again.

3 Planning Nextcloud Backups

To backup a Nextcloud installation, there are four major components to pay attention to:

- The folder `nextcloud/config`
- The folder `nextcloud/data`
- The folder `nextcloud/theme`
- The database (MySQL/MariaDB, SQLite, PostgreSQL)

Nextcloud files and folders are located in the filesystem itself, using conventional directory structures. The database stores additional information about the files, i.e. the number of files, permissions, timestamps, etc. After a manual solution (backing up folders, files, and the database) we'll look at more automated solutions (see section 4).

3.1 Backup of Folders

Before creating a copy of the `nextcloud` folder and its content, make sure to activate the Nextcloud maintenance mode (see section 4.2.1) to lock the sessions of logged-in users and prevent new logins to avoid inconsistencies in your data.

After that, use your preferred backup program (or simply copy the folders with `rsync`) to a place outside the Nextcloud environment, i.e. to an external harddrive or a NAS device.

3.2 Single Files

Files uploaded by Nextcloud users end up in (`nextcloud/data/`*`USER`*`/files`). If a user deletes a file via the Nextcloud web interface or desktop/mobile client, it is not deleted permanently. Instead, it is moved to the trash can (`Deleted files`, see Figure 2) until the user manually deletes it or the app `Deleted Files` removes it to make room for new files.

If a file has been permanently deleted, it is not enough to simply put it back in the folder `nextcloud/data/`*`USER`*`/files`. The Nextcloud web interface and clients have no knowledge about that file. As a workaround Nextcloud's `occ` command can scan the filesystem for new files and add them to the database:

```
occ files:scan <username>
```

The `occ` tool is located in the `nextcloud` directory, for example in `/var/www/nextcloud` (on a Linux server). The PHP script must be run as HTTP user to make sure the correct permissions for the Nextcloud files and directories are maintained.[1]

[1] https://docs.nextcloud.com/server/stable/admin_manual/configuration_server/occ_command.html

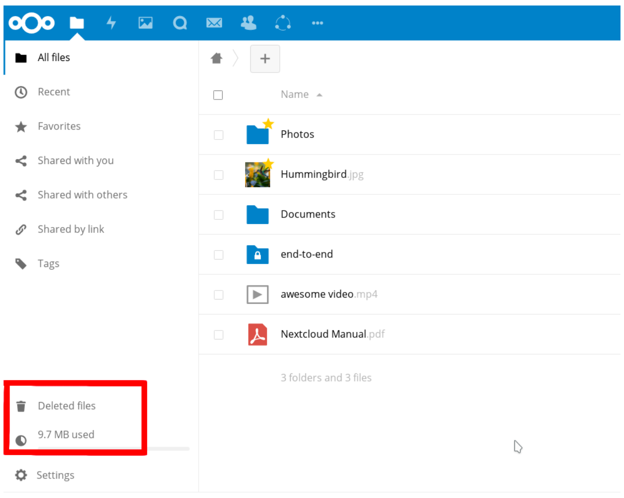


Figure 2: *The Nextcloud Trash Can*

3.3 The Database

In order to create a backup of the Nextcloud database, you can use the related DB tools, for example `mysqldump` (MySQL/MariaDB), `sqlite3` (SQLite), or `pg_dump` (PostgreSQL). Make sure to delete all existing database tables before you restore such a backup. For more information, please have a look at the Nextcloud manual.[2]

4 Bareos & Nextcloud

The manual way (shown in the last section) works well enough for small environments and a limited numbers of files. It takes too long, though, if there are lots of files and/or users. The next four sections present more automated methods and show how Bareos can handle the backup and restore process for Nextcloud servers, including the database.

In our examples we use a MySQL database. If your setup uses PostgreSQL or SQLite, please adjust the `mysqldump` command in the listings accordingly.

4.1 Shutting down the Application

The simplest solution is to quit the application before the backup. That way its internal state can be stored on disk. After the backup is finished, the application can be restarted. Administrators can write their own shell scripts and combine them with cron jobs or let Bareos handle this.

The problem with this quick and dirty approach: The service is being interrupted while the application shuts down and the backup job is running. It's definitely faster than some other methods shown below, but in some situations even this amount of downtime is not acceptable.

4.2 Exporting the Application Data

Before the actual backup runs the application data needs to be exported. For example, if the database is a MySQL database, then the `mysqldump` client performs a logical backup with a set of SQL statements to reproduce the original database object definitions and table data.

[2] https://docs.nextcloud.com/server/stable/admin_manual/maintenance/index.html

The result is a large file that can be backed up and restored later.

In this scenario the Nextcloud service is not interrupted and there is no downtime for the users. The disadvantage is that it may slow down the MySQL daemon and the backup job takes longer, since the export has to be done first. Also, if it's a large database, then it might take a long time (depending on the hardware) and the exported data takes up extra space.

4.2.1 Using Bareos Directives

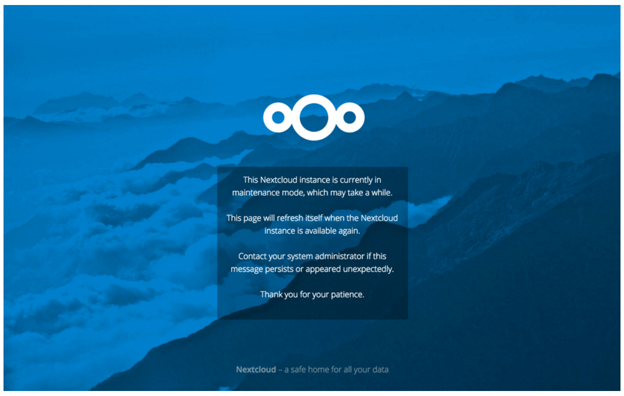
Bareos' `RunScript` directive allows the definition of external commands or scripts to run before or after a backup/restore job. Commands can be executed on the client side (`ClientRunBeforeJob` and `ClientRunAfterJob`) and on the Bareos director (`RunBeforeJob` and `RunAfterJob`).

This is a configuration example for a job called `backup-nextcloud` on a client machine called `ubuntu-fd`. It defines two external shell scripts that run before the backup job and after the backup job:

```
 Job {
   Name = "backup-nextcloud"
   JobDefs = "DefaultJob"
   Client = "ubuntu-fd"
   FileSet = "nextcloud"
   ClientRunBeforeJob = "/usr/local/\
sbin/nextcloud_before.sh"
   ClientRunAfterJob = "/usr/local/\
sbin/nextcloud_after.sh"
}
```

The restore job (`restore-nextcloud`) also has a directive that calls a script after the job is finished:

```
 Job {
   Name = "restore-nextcloud"
   Description = "Restore Nextcloud \
files and database"
   Type = Restore
   Client = ubuntu-fd
   FileSet = "nextcloud"
   Storage = File
   Pool = Incremental
```



This Nextcloud instance is currently in maintenance mode, which may take a while.

This page will refresh itself when the Nextcloud instance is available again.

Contact your system administrator if this message persists or appeared unexpectedly.

Thank you for your patience.

Nextcloud – a safe home for all your data

Figure 3: *The maintenance mode locks the session of logged-in users and prevents new logins.*

```
   Messages = Standard
   Where = /
   ClientRunAfterJob = "/usr/local/\
sbin/nextcloud_after_restore.sh
}
```

The first script that runs before the backup job (`nextcloud_before.sh`) looks like this:

```
 #!/bin/sh
dumpdir=/var/tmp/nextcloud-backup
dumpfile=$dumpdir/mysqldump.sql
if [ ! -d  "$dumpdir" ]
then
    mkdir -p "$dumpdir"
fi
/usr/local/sbin/occ maintenance:\
mode --on
mysqldump --all-databases \
--single-transaction >$dumpfile
```

First it creates the directory `/var/tmp/nextcloud-backup` if it does not already exist. If not, it creates the folder. The script calls the `occ` command to put Nextcloud in maintenance mode (see Figure 3) which means that users can't change anything while the backup is running. Then it runs `mysqldump` for all databases as a single transaction, so it doesn't lock all tables.

The `nextcloud_after_restore.sh` script dumps the database back and turns off the Nextcloud maintenance mode:

```
 #!/bin/sh
mysql </var/tmp/nextcloud-backup/\
mysqldump.sql
/usr/local/sbin/occ maintenance:\
mode --off
```

At the moment there is no error checking in any of the scripts. Keep in mind that Bareos ignores the return code of `RunAfterJob` scripts. While a `RunBeforeJob` script that exits with an error results in a failed Bareos job, a failed `RunAfterJob` will still show `job status ok` in Bareos, because the script runs after the job is finished.

4.3 Streaming Data (`bpipe`)

The next approach still uses the `mysqldump` command, but this time the exported data doesn't get stored in a temporary file. Instead the Bareos `bpipe` plugin is used to stream the database dump to Bareos for backup. The `bpipe` plugin also transmits the data from Bareos to another specified program for restore.

This method no longer requires the `RunBeforeJob` and `RunAfterJob` directives. Instead, the plugin is configured in the `Include` section of the job's `FileSet` resource in the file `bareos-dir.conf`:

```
 FileSet {
   Name = "nextcloud-bpipe"
   Description = "Back up Nextcloud \
dirs and dump MySQL DB"
   Include {
     Options {
       Signature = MD5 #calculate md5 \
checksum per file
       Compression = GZIP
       }
     File = /var/www/nextcloud
     Plugin = "bpipe:file=/MYSQL/dump.\
sql:reader=/usr/local/sbin/nextcloud_\
bpipe.sh:writer=/usr/local/sbin/\
nextcloud_bpipe_restore.sh"
   }
}
```

The bpipe plugin calls the two external scripts nextcloud_bpipe.sh and nextcloud_bpipe_restore.sh. They use mysqldump to read from stdin and write to stdout.

This approach can save a lot of time since the export and the backup can run at the same time. It also saves disk space because a temporary file for the database dump is no longer necessary.

Administrators also need to tell the file daemon to load the plugin, because the plugin runs on the Bareos client. Here is an example configuration for the file daemon (/etc/bareos/bareos-fd.d/client/myself.conf):

```
Client {
  Name = ubuntu-fd
  Plugin Directory = "/usr/lib/\
bareos/plugins"
  Plugin Names = "bpipe"
}
```

4.4 Application Plugins

Some applications have their own Bareos plugin. The bareos_percona plugin for example works with MySQL and MariaDB databases. It uses the xtrabackup tool from Percona to perform full and incremental backups of the database – the incremental dumps are a big advantage in this scenario, especially for larger ones.

4.4.1 xtrabackup and bareos_percona

First you need to install xtrabackup from the Percona repository.[3] Percona provides RPM packages (for Red Hat, CentOS and Amazon Linux AMI) and .deb packages for Ubuntu and Debian GNU/Linux.

The bareos_percona plugin is also required. It is not part of the main Bareos packages, instead it is available via the bareos-contrib repository:

```
git clone https://github.com/bareos/\
bareos-contrib.git
cp bareos-contrib/fd-plugins/bareos_\
percona/*.py /usr/lib/bareos/plugins
```

[3] https://www.percona.com/doc/percona-xtrabackup/LATEST/installation.html#installing-percona-xtrabackup-from-repositories

4.4.2 Bareos FD Configuration

This is an example for the file daemon configuration (/etc/bareos/bareos-fd.d/client/myself.conf) that loads the Python plugin:

```
Client {
  Name = ubuntu-fd
  Plugin Directory = "/usr/lib/\
bareos/plugins"
  Plugin Names = "python"
}
```

The definition of the FileSet resource looks like this:

```
FileSet {
  Name = "nextcloud-xtra"
  Description = "Backup nextcloud \
directories & DB with xtrabackup"
  Include {
    Options {
      Signature = MD5
      Compression = GZIP
    }
    File = /var/www/nextcloud
    Plugin = "python:module_path=\
/usr/lib/bareos/plugins:module_\
name=bareos-fd-percona"
  }
}
```

The restore process is a bit more complicated. The plugin does not do the restore for you; instead you get a temporary directory with all the files and you can use the percona commands to do a MySQL restore – of course, an automated setup with some extra shell scripts is possible.

The restore does not work when there are other files in the FileSet resource. So it's necessary to create two restore jobs: one to restore all files and another one to deal with the database.

This is a sample script to be called after the restore job:

```
#!/bin/sh
perconadir="/tmp/xtra-restore/\
_percona"
mysqldatadir="/var/lib/mysql"
```

```
mysqlservice="mysql"
mysqluser="mysql"
cd $perconadir || exit 1
backupdir=$(ls -t | head -1)
test -n "$backupdir" || exit 1
cd $backupdir || exit 1
basedir=$(ls | head -1)
inc_count=$(ls | wc -l)
inc_count=$(expr $inc_count - 1)
inc_dirs=$(ls | tail -$inc_count)

if [ $inc_count = 0 ]
then
  xtrabackup --prepare --target-dir=\
$basedir
else
  xtrabackup --prepare --apply-log-\
only --target-dir=$basedir
  count=1
  for i in $inc_dirs
  do

# last one needs different parameters
    if [ $count = $inc_count ]
    then
      xtrabackup --prepare --target-\
dir=$basedir --incremental-dir=$i
      else
      xtrabackup --prepare --apply-\
log-only --target-dir=$basedir \
--incremental-dir=$i
    fi
    count=$(expr $count + 1)
  done
fi

/usr/local/sbin/occ maintenance:mode \
--on
service $mysqlservice stop || exit 2
rm -rf $mysqldatadir/*
xtrabackup --copy-back --target-\
dir=$basedir
chown $mysqluser: -R $mysqldatadir/*
service $mysqlservice start
rm -rf $basedir
/usr/local/sbin/occ maintenance:mode \
--off
```

The plugin restores the last full backup and all necessary incremental backups to a temporary directory. The script collects and prepares these backup parts to recreate the MySQL data files with the `xtrabackup` tool that essentially does a transaction replay from the incremental backup files. It creates a complete new MySQL data directory that replaces the original one.

5 Conclusion

Of course, there is still room for improvement. It would be better to have a single restore job. Also, for applications like calendars or mailservers, it would be nice to be able to restore a single calendar entry or message. All of these scenarios would require a complex plugin.

The shown methods are not strictly limited to Nextcloud – it might give you some ideas on how to plan a backup/restore strategy for other applications. One advice: Test everything, especially the restore process.

6 About Bareos

Bareos[4] (**B**ackup **A**rchiving **Re**covery **O**pen **S**ourced) is a cross-network open source backup solution that preserves, archives and recovers data from all major operating systems. The Bareos project started 2010 as a Bacula fork and is now being developed under the AGPLv3 license. The company Bareos GmbH & Co. KG and their partners offer professional subscription and support services.

7 About Nextcloud

Nextcloud[5] offers the industry-leading, fully open source, self-hosted Content Collaboration Platform, combining the easy user interface of consumer-grade cloud solutions with the security and compliance measures enterprises need. Nextcloud brings together universal access to data through mobile, desktop and web interfaces with next-generation, on-premise secure communication and collaboration features like real-time document editing, chat and video calls, putting them under direct control of IT and integrated with existing infrastructure.

[4] https://www.bareos.com/en
[5] https://nextcloud.com