The GMU nextCloud Tools Documentation

Release 0.1.0

Mybrid Wonderful (The GMU), Gregg Yearwood (The GMU)

Nov 05, 2020

CONTENTS:

`thegmu-nextcloud-tools` is a collection of Python tools for providing additional features to nextCloud (https://nextcloud.com). The nextCloud software is a _Dropbox_ like file sharing service that you host yourself.

- Documentation: http://the-gmu-nextcloud-tools.readthedocs.org/
- Source Code: https://bitbucket.org/thegmu/thegmu-nextcloud-tools
- Download: http://pypi.python.org/pypi/thegmu-nextcloud-tools
- Mail: mybrid@thegmu.com

Please feel free to ask questions, report installation problems, bugs or any other issues to the email address provided above.

Note: Version 0.2.x is an alpha release. Overall requirements:

1. Linux, nextCloud supported
2. Python 3.6+
3. nextCloud 10.1.0
4. GIT repos mirrored on the nextCloud host.

CHAPTER

ONE

TOOLS

1.1 GIT Sync - mirror a GIT repo as an nextCloud folder

Here at The GMU, https://www.thegmu.com, we want to be able to share our GIT document folder with our Dropbox-like file sharing software, nextCloud, https://nextcloud.org. Our company is just two people, the two owners. We are both software developers and are comfortable using GIT for document control. Our intitial requirement is to be able to share our documents under GIT version control using the same features provided by nextCloud. For example:

1. We want to share files by email address with our accountant and other users who are not registered with our nextCloud server.
2. We also want to be able to have user created groups for sharing with groups.
3. We also want to have the ability to create temporary URL links for one time use.

All the features listed are available with nextCloud but missing from Github or Bitbucket where repositories are either public or private and no finer grain sharing control is available.

The design is simple enough in concept where the git repos are mirrored as read-only on the nextCloud host. The GIT sync tool updates nextCloud every minute with any new files or changes to the GIT repository.

The implementation was much more involved. For example one cannot just copy files from a GIT folder to the corresponding nextCloud folder because the file needs to be registered with the nextCloud database. This registration requires uploading the file from the server to the nextCloud software in the exact same way any client does.

GIT Sync Requirements:

1. Mirror a GIT repo as read-only to a configured nextCloud user account.
2. All accidental uploads to the GIT folder are to be reverted to the GIT version of the file.
3. A special transfer directory, 'Transfer', will allow uploads.
4. Files transfered to the transfer directory from say a phone are to be moved to an appropropriate GIT folder in a timely fashion.
5. Files transfered are purged weekly.
6. Daily email report of all files processed for the day.
7. Daily email of all errors processing files included in the daily email report.
8. Configuration file management. Adding a repo per user can be accomplished by creating a new configuration file.
9. Include and exclude directory options. The GIT repo directories can be filtered by using either an include or exclude list.

Note: Linux shell commands are frequently executed instead of using built-in Python compatible functions for the following reasons.

1. **nextCloud files**: sudo is required for nextCloud files that are owned by one account. Since we are using the Apache web server that account is 'www-data'.
2. **GIT files**: sudo is required for GIT files that are owned by one account on our server, 'root'.
3. **davfs**: sudo is required for the account that mounts the davfs folder used to upload to nextCloud.

CHAPTER

TWO

INTRODUCTION

2.1 System requirements and dependencies

This is a small scripting application with only one Python library used: PyYAML. However, Linux shell commands are used because different accounts owning different directories requires the use of 'sudo'.

1. nextCloud 10.1.0 system requirements and dependencies: https://doc.nextcloud.org/server/10.1/admin_manual/installation/system_requirements.html
2. Python 3.6/7 - We strongly recommend a Python virtualenv is used.
3. PyYAML - Configuration files are in yaml.
4. cron - cron is used to invoke the mirror script once-per-minute.
5. davfs - The nextCloud, DavFS URL for the user account is mounted in some */media* directory as a DavFS mount point for accessing the GIT directory.

2.2 Package Installation

This package is available from the Python Package Index. If you have pip you should be able to do:

```
$ pip install thegmu-nextcloud-tools
```

You can also download manually, extract and run `python setup.py install`.

Package *`thegmu-nextcloud-tools`* has been deployed using nextCloud 10.1.0 and Python 3.6.

2.3 System Administration Planning

Configuration beyond package installatoin will require setting system administration policy and planning thereof.

After installing the Python package what remains is configuration of various server files and services. The package currently requires manual configuration because various policies such as the number of git repos, how frequent to synchronize, and email policy all require planning of wide variability. The Python package represents the part of the application that requires no planning. After installing the package the policy and planning concerns are with the configuration file. The configuration file mostly addresses the GIT repo directory, the nextCloud directory and then their respective accounts. The final stage of policy and planning has to do with the cron jobs. How often should the synchronization happen? The script supports as synchronization as frequently as every minute. How often should email be sent and then again who is on the mailing list?

Note: The server installation requires running the script with an option to create at least one configuration file. The configuration files are passed to the synchronization script as a command line option within a cron job. The cron job utilizes environment variables intended to be set in the contab file that contain the list configuration file names.

Cron job script:

```
#!/bin/bash

#Ensure thegmu_nextcloud_git_sync.py is in PATH.

#PATH=$PATH:/usr/local/bin

#If running from a virtualenv then use activate.
# . /path/to/your/venv/bin/activate

# crontab file:
# THEGMU_CONFIG_DIR="/etc/thegmu"
# THEGMU_OCGS_CONFIG_FILES="gitcloudsync.host.yaml gitcloudsync.youraccount.yaml"

if [ "$THEGMU_CONFIG_DIR" == "" ]; then
    echo "$0: Missing environment variable THEGMU_CONFIG_DIR."
    exit -1
fi

if [ "$THEGMU_OCGS_CONFIG_FILES" == "" ]; then
    echo "$0: Missing environment variable THEGMU_OCGS_CONFIG_FILES."
    exit -1
fi

for configfile in $CONFIGFILES; do
    sudo thegmu_nextcloud_git_sync.py $CONFIGDIR/$configfile
done
```

CHAPTER

THREE

POLICY

3.1 Policy Questions

All the following questions need to be answered for fully configuring the application. System administrators are expected to understand how to create cron scripts from the sample cron scripts provided for the policies.

3.2 GIT Policy

- Q: How many GIT repo directories are to be mirrored?
- Q: Which Linux account will host the GIT repo?
- Q: Which GIT directories for a repo are to be included if any?
- Q: Which GIT directories for a repo are to be excluded if any?
- Q: Which GIT repo, branch, or tag version for a repo is to be mirrored?

Note: The configuration file includes a simple "git pull" command. This command can be modified with version configuration if needed. The GIT repo directory has files that are checked out. The branch that is currently checked out can be sync'ed without setting the branch in the configuration file.

- Q: Will there be large files copied to the GIT directory for mirroring?

Note: The mirror of the git directoy will mirror all files and thus files not under GIT control can be copied to the git directory. For example, you may have large video and audio files to mirrored in a read-only fashion identical to the GIT controled files.

- Q: Who can push to the GIT repo?

3.3 nextCloud Policy

- Q: Which nextCloud account will be mirrored too?
- Q: Will the nextCloud account share to a group?

3.4 Configuration File Policy

- Q: Who can edit the configuration files? If users are adding or editing configuration files then we do not recommend using */etc/thegmu* directory for configuration files.
- Q: Where should the configuration files reside? We recommend */etc/thegmu* if only the system administrator is editing the files.
- Q: Where should the application log files go? Each configuration file needs a distinct set of log files. Log files can go in the same directory configured with different file names or log files can go in separate directories with the same name.

3.5 GIT Sychronization Policy

- Q: How often should the GIT files be synced with nextCloud in cron?

Note: We recommend every minute. The application supports syncing every minute. Longer times like very five mintues may experience only some files copied per sync. If a file is copied the entire file is always copied, but on all files may be copied. This is due to a time-out of thirty seconds for files copied. This means only some files copies an happen every minute until all files of some new commit are synced. The DavFS performance is highly variable depending on the load of the nextCloud server. The DavFS cache needs to be flushed and this can take longer than the reported copy time by the Linux copy command. We found that limiting the DavFS copy load to at most thirty seconds creates manageable DavFS flush performance and a low impact on the nextCloud server.

3.6 Report Email Policy

You can safely ignore this section if you do not have a policy to email daily reports.

- Q: What GIT repos need reporting?
- Q: What email accounts need the report email?
- Q: How often should the report be run?

Note: The report is always daily. However, the changes throughout the day can be picked up by running the report hourly where the report will always contain all the files processed for the day.

3.7 On To Configuration

Once you have your policy plan in place then the next step is to create configuration file.

CHAPTER

FOUR

CONFIGURATION FILE

4.1 Create

To create a new configuration file then run the script with a create configuration file option, '-c':

```
thegmu_nextcloud_git_sync.py -c

# On host named 'fuzzyfinehat'.
% python3 thegmu_nextcloud_tools/thegmu_nextcloud_git_sync.py -c
% thegmu_nextcloud_git_sync.fuzzyfinehat.yaml file created.
```

4.2 Sample

The created configuration file name includes the host name. If you have more than one configuration file you'll to change the file name. Sample:

```
% python3 thegmu_nextcloud_tools/thegmu_nextcloud_git_sync.py -c
% thegmu_nextcloud_git_sync.fuzzyfinehat.yaml file created.
% cat thegmu_nextcloud_git_sync.fuzzyfinehat.yaml
name: 'thegmu-nextcloud-git-sync'
davfs:
  mount:
    source_url: 'https://www.yourdomain.com/oc/remote.php/dav/files/youraccount/'
    target_dir: '/media/davfs/yournextcloudaccount'
    max_active_seconds: 30
nextcloud:
  account:
    linux: 'www-data'
    nextcloud: 'yournextcloudaccount'
  directory:
    exclude:
      - 'Transfer'
    include: null
    root: '/path/to/nextcloud/data/yournextcloudaccount/files'
git:
  account:
    linux: 'yourlinuxaccount'
  directory:
    exclude:
      - '.git'
      - '__pycache__'
```

(continues on next page)

(continued from previous page)

```
      - 'Transfer'
    include: None
      root: '/home/yourlinuxaccount/path/to/git/repo'
  pull_cmd: 'sudo su - %s -c "cd %s && git pull"'
report:
  file:
    process: '/path/to/your/log/dir/thegmu_nextcloud_git_sync.processed.csv'
    error: '/path/to/your/log/dir/thegmu_nextcloud_git_sync.errors.csv'
  error:
    columns:
    - 'File'
    - 'Time'
    - 'Cause'
  column:
    timestamp: 'Time'
run:
  production_host: 'nextcloudhost'
  lockpath: '/tmp/thegmu_nextcloud_git_sync.py.lock'
test:
  account:
    linux: 'yourlinuxtestaccount'
  directory:
    run: '/home/yourlinuxtestaccount/path/to/test/git/repo'
    temp_test: "/home/yourlinuxtestaccount/path/to/mock/nextcloud/data"
  file:
    delete: '/home/yourlinuxtestaccount/path/to/mock/nextcloud/data/youraccount/files/
↪deleteme.txt'
    temp_test_nextcloud_data: "/home/yourlinuxtestaccount/path/to/test/git/repo/
↪archive/repo.tgz"
    zero_file: '/home/yourlinuxtestaccount/path/to/mock/nextcloud/data/youraccount/
↪files/zero_file.txt'
  mount:
    target_dir: '/media/davfs/youratestccount'
```

4.3 DavFS

- Install DavFS if not already installed. On Ubuntu:

```
sudo apt install davfs2
```

- Create the mount point. On Ubuntu create a `davfs` directory under `/media` and then the nextCloud account:

```
sudo mkdir -p /media/davfs/yournextcloudaccount
```

- nextCloud account DavFS URL and password. The credentials for the nextCloud account are kept in the DavFS secrets file, `/etc/davfs/secrets`. To mount the DavFS then three pieces of information are required in the secrets file: account name, account password and URL. #. Log into the nextCloud account and in the lower righthand corner of the main click on the 'Settings' wheel icon. A dialog will open that includes the DavFS URL:

```
https://www.thegmu.com/oc/remote.php/dav/files/gmucorp/
```

- Update DavFS credentials in the secrets file, `/etc/davfs/secrets`. Follow the instructions provided at the top of the file for adding the mount point entry.

- Test DavFS mount:

```
sudo mount /media/davfs/yournextcloudaccount
```

- Update DavFS entries in the configuration file. Edit the configuration files and update the `davfs` section with `source_url`, and `target_dir`. Also add the nextCloud account to the `nextcloud` section.

Note: The `davfs` section, entry of `max_active_seconds` of 30 seconds is recommended when running synchronization every minute. You may increase the `max_active_seconds` to reflect longer synchronization times. You should test your DavFS implementation using a copy set of files that takes longer than the `max_active_seconds` and monitor the nextCloud performance. Our tests using `rsync` with gigabytes of files would cause the DavFS mount to hang.

4.4 nextCloud

- Accounts: There are two account entries for the `nextcloud` section, the nextCloud user account and the Linux account. The Linux account represents the owner of the disks files. On Ubuntu/Apache the user is `www-data` with a default nextCloud installation.
- Directories: Update the root directory and the include, exclude directories given the answers to your policy questions and your nextCloud installation. Just realize that the root path is to the nextCloud specific account files and not the nextCloud server data root.

4.5 GIT

- Accounts: There is one account entry for the `git` section, Linux account that owns the git repo mirror. The Linux account represents the owner of the disks files.
- Directories: Update the root directory and the include, exclude directories given the answers to your policy questions and your GIT repo requirements.
- `pull_cmd`: Updates to GIT changes are done using GIT pull. The template configured command should work unless the version such as branch need to be added.

Note: The use of `sudo` to specify the Linux account ensures the root user account will work as well.

4.6 Report

- Files: If you only have one configuration file configured then the default file names will suffice, just update the path to reflect your policy. If multiple configuration files are used then each configuration file needs a distinct set of files by either specifying distinct directories or distinct file names per your policy.
- Error: no configuration required.
- Column: no configuratoin required.

4.7 Run

- `production_host`: This must be the current host of the nextCloud host. If the `production_host` does not match the current host then the test configuration is used.

> **Warning:** This must be set to the nextCloud host or a test run is executed using the values in the `test` section of the configuration file.

- `lockpath`: No change is required.

4.8 Test

No configuration required.

4.9 Test Configuration

Test your configuration by runing the script passing the configuration file:

```
sudo thegmu_nextcloud_git_sync.py thegmu_nextcloud_git_sync.yourhost.yaml
```

Next up is creating the `cron` scripts for synchronization every minute and optionally to send a daily email report of files processed for a day.

CHAPTER

FIVE

CRON

5.1 CRON Synchronization

To `root` or not to `root`. The synchronization uses sudo to mount DavFS as well as run the `git pull` command. A properly configured user account can be used as long as these commands execute.

- Backup: If using the `root` account then the crontab file `/etc/crontab` can be updated. Always make a backup, i.e.:

```
cp /etc/crontab /etc/crontab.bck
```

If using the account crontab then backup as:

```
crontab -l > crontab.txt.bck
crontab -l > crontab.txtk
```

- Environment: There are two environment variables to be set in the crontab configuration, one specifying the configuration file directory and the second specifying the files:

```
# /etc/crontab, ~/crontab.txt
# export THEGMU_CONFIG_DIR="/etc/thegmu"
# export THEGMU_OCGS_CONFIG_FILES="thegmu_nextcloud_git_sync.nextcloudhost.yaml␣
↪thegmu_nextcloud_git_sync.youraccount.yaml"
```

- Script: A `bash` script was installed to synchronize the list of configuration files:

```
# /etc/crontab
# m h dom mon dow user      command
*  *    * * *   root    test -x /usr/local/bin/thegmu_nextcloud_git_sync.sh && /
↪usr/local/bin/thegmu_nextcloud_git_sync.sh

# ~/crontab.txt
# m h dom mon dow   command
*  *    * * *   /usr/local/bin/thegmu_nextcloud_git_sync.sh
```

5.2 CRON Report

The application script provides a report of all files processed for the current day or previous day. The report is printed to console:

```
thegmu-nextcloud-git-sync.py -y thegmu_nextcloud_git_sync.yourhost.yaml
Processed:
Documents/foo.txt,20190331:055015.52,COPY
Documents/bar.txt,20190331:055015.52,COPY
```

The reported file listing is relative to the nextCloud directories as specified in the nextCloud interface. They are reported relative to nextCloud because of deletions. If accidental upload of files to the nextCloud/GIT directory occurs then the files will be deleted and reported as such.

The report `cron` job then just needs to redirect the output to a file and mail that file to a set of recipients.

Note: The package does not ship with any example report mail cron script because the variation is more than what is constant.

Below is an example we use currently. Feel free to copy this and update. We run this script daily as the `root` user by copying th
/etc/cron.daily/

Example report script:

```
2019-04-02 07:35:46 % cat /etc/cron.daily/gitcloudsync_gmucorp_daily_report
#!/bin/bash

SHELL=/bin/bash
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

MAILTO="greggy@thegmu.com mybrid@thegmu.com"
#MAILTO="mybrid@thegmu.com"
MAILFROM="greggy@thegmu.com"
MAILDAY=`date +"%m/%d/%Y"`
MAILSUBJECT="[GMU] $MAILDAY NextCloud 'gmucorp' files processed yesterday."

GITCLOUDSYNC_SCRIPT="/usr/local/bin/thegmu_nextcloud_git_sync.py"
GITCLOUDSYNC_YAML="/etc/thgmu/thegmu_nextcloud_git_sync.gmucorp.yaml"
GITCLOUDSYNC_MAIL_FILE="/tmp/gitcloudsync_mail.txt"

GITCLOUDSYNC_REPORT="$GITCLOUDSYNC_SCRIPT file not found."

if [ -f $GITCLOUDSYNC_SCRIPT ]; then
    echo "$GITCLOUDSYNC_SCRIPT running..."
    python3 $GITCLOUDSYNC_SCRIPT --yesterday_report $GITCLOUDSYNC_YAML >
↪$GITCLOUDSYNC_MAIL_FILE 2>&1
else
    echo $GITCLOUDSYNC_REPORT > $GITCLOUDSYNC_MAIL_FILE
fi
```

5.3 Configuration Complete

Configuration is now complete.

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

t

INDEX