How-To Geek

Google › Google Drive

How to Use rclone to Back Up to Google Drive on Linux

DAVE MCKAY @thegurkha
UPDATED APR 29, 2020, 2:03 PM EST | 6 MIN READ

Google

There's still no official Linux client for Google Drive, but you can back up to your Google Drive using the `rclone` utility right from the command line. We show you how.

Where's Google Drive on Linux?

Despite promising Linux support "coming soon" back in 2012, there's no indication that Google will ever produce a native Linux client for Google Drive. There are several unofficial third-party solutions, such as InSync, overGrive and ODrive, and some file browsers allow integration with your Google Drive, such as Files in GNOME.

The third-party applications are commercial products, requiring either an outright purchase or a subscription. They work well they don't cost much, and in fact, overGrive does have a free version, offering limited functionality for no cost.

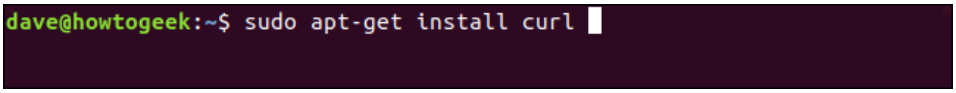
But what if you want to create and run backups from the command line? Or to incorporate that functionality into scripts? That's all possible thanks to an amazing application called `rclone`. In fact, with `rclone` you can back up, download, and synchronize files to over forty different cloud solutions. It's like rsync for clouds.

Installing rclone

`rclone` almost certainly won't be installed on your Linux computer by default. Happily, there's an installation script that should work on all distributions. The installation process uses curl. On the computers used to research this article, Fedora 31 and Manjaro 18.1.0 already had `curl` installed but `curl` had to be installed on Ubuntu 18.04 LTS.

On Ubuntu, run this command to install it:

```
sudo apt-get install curl
```



Once `curl` has been installed, install `rclone` with this command:

```
curl https://rclone.org/install.sh | sudo bash
```

```
dave@howtogeek:~$ curl https://rclone.org/install.sh | sudo bash
```

When the `rclone` installation has finished, you'll see a success message.

```
Processing manual pages under /usr/share/man/zh_TW...
Purging old database entries in /usr/share/man/sl...
Processing manual pages under /usr/share/man/sl...
Purging old database entries in /usr/share/man/tr...
Processing manual pages under /usr/share/man/tr...
Purging old database entries in /usr/share/man/ja...
Processing manual pages under /usr/share/man/ja...
Processing manual pages under /usr/local/man...
Updating index cache for path `/usr/local/man/man1'. Wait...done.
Checking for stray cats under /usr/local/man...
Checking for stray cats under /var/cache/man/oldlocal...
1 man subdirectory contained newer manual pages.
1 manual page was added.
0 stray cats were added.
18 old database entries were purged.

rclone v1.50.2 has successfully installed.
Now run "rclone config" for setup. Check https://rclone.org/docs/ for
more details.

dave@howtogeek:~$
```

This has installed the `rclone` program on your Linux computer. The next step is to run through the setup process and authenticate `rclone` to access your Google Drive.

Creating an rclone Remote Connection

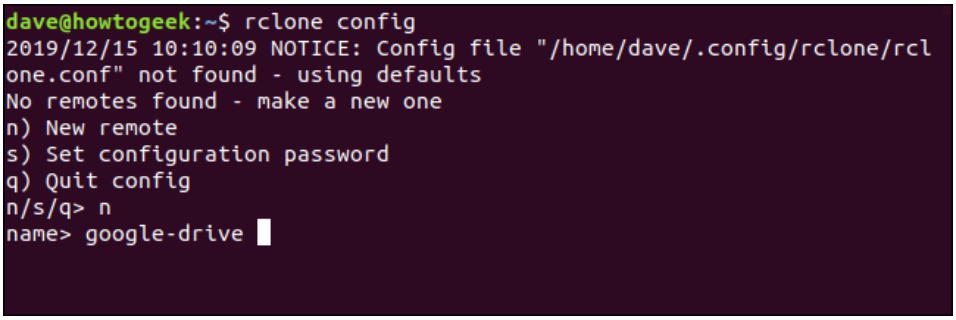
Connections to remote cloud services are called "remotes" in the `rclone` world. We need to create one for Google Drive. Start the `rclone` configuration process with this command:

```
rclone config
```

```
dave@howtogeek:~$ rclone config
```

There are a lot of questions in the configuration process. But don't be disheartened, many of them can be left at their default values and simply accepted by pressing "Enter."

`rclone` tells us there are no remotes configured. Press “n” and press “Enter” to create a new remote. It will prompt you for a name. We’re going to call it “google-drive.” Use whatever name you like.



A long menu allows you to choose the type of storage you’re creating a remote connection to.

```
n/s/q> n
name> google-drive
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
 1 / 1Fichier
   \ "fichier"
 2 / Alias for an existing remote
   \ "alias"
 3 / Amazon Drive
   \ "amazon cloud drive"
 4 / Amazon S3 Compliant Storage Provider (AWS, Alibaba, Ceph, Digital
Ocean, Dreamhost, IBM COS, Minio, etc)
   \ "s3"
 5 / Backblaze B2
   \ "b2"
 6 / Box
   \ "box"
 7 / Cache a remote
   \ "cache"
 8 / Citrix Sharefile
```

Scroll through the list until you see the entry for Google Drive, and make a note of its number.

```
   \ "sharefile"
 9 / Dropbox
   \ "dropbox"
10 / Encrypt/Decrypt a remote
   \ "crypt"
11 / FTP Connection
   \ "ftp"
12 / Google Cloud Storage (this is not Google Drive)
   \ "google cloud storage"
13 / Google Drive
   \ "drive"
14 / Google Photos
   \ "google photos"
15 / Hubic
   \ "hubic"
16 / JottaCloud
   \ "jottacloud"
17 / Koofr
   \ "koofr"
18 / Local Disk
   \ "local"
```

We can see that in this instance, it is number 13. Enter this as the storage type and press “Enter.”

```
25 / Pcloud
   \ "pcloud"
26 / Put.io
   \ "putio"
27 / QingCloud Object Storage
   \ "qingstor"
28 / SSH/SFTP Connection
   \ "sftp"
29 / Transparently chunk/split large files
   \ "chunker"
30 / Union merges the contents of several remotes
   \ "union"
31 / Webdav
   \ "webdav"
32 / Yandex Disk
   \ "yandex"
33 / http Connection
   \ "http"
34 / premiumize.me
   \ "premiumizeme"
Storage> 13
```

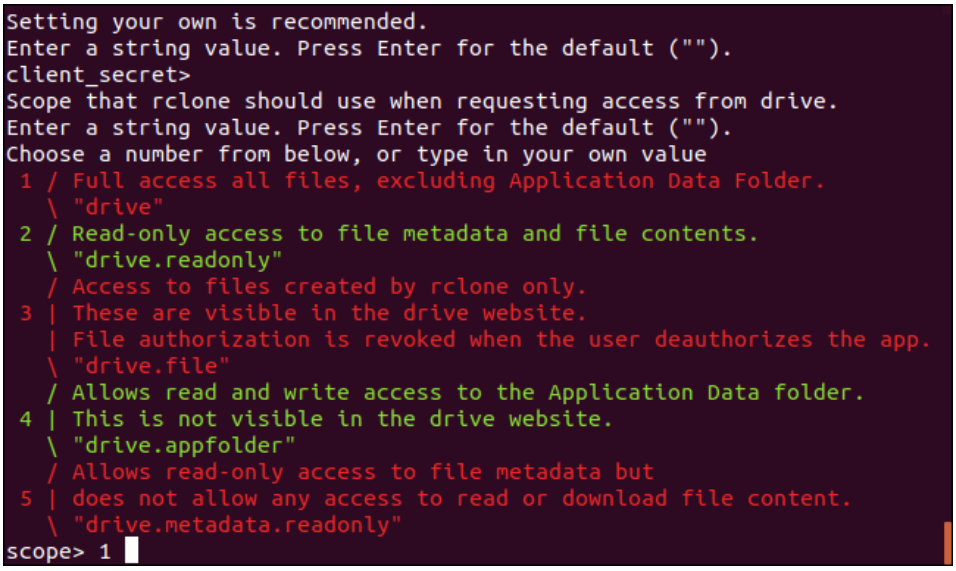
You're prompted for a Google Application Client ID. Press "Enter" to accept the default.

```
30 / Union merges the contents of several remotes
   \ "union"
31 / Webdav
   \ "webdav"
32 / Yandex Disk
   \ "yandex"
33 / http Connection
   \ "http"
34 / premiumize.me
   \ "premiumizeme"
Storage> 13
** See help for drive backend at: https://rclone.org/drive/ **

Google Application Client Id
Setting your own is recommended.
See https://rclone.org/drive/#making-your-own-client-id for how to cre
ate your own.
If you leave this blank, it will use an internal key which is low perf
ormance.
Enter a string value. Press Enter for the default ("").
client_id>
```

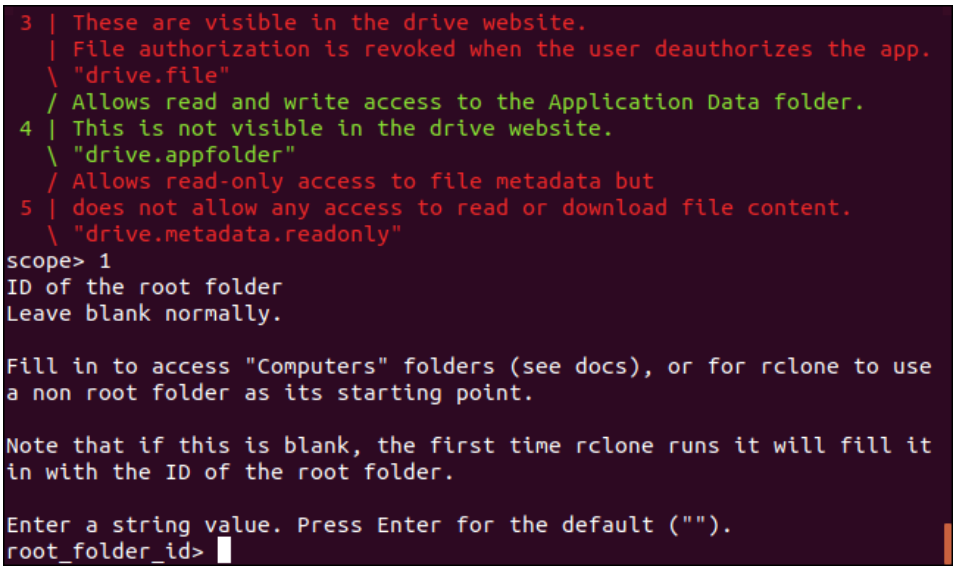
You're then prompted for a Google Application Client Secret.

```
32 / Yandex Disk
   \ "yandex"
33 / http Connection
   \ "http"
34 / premiumize.me
   \ "premiumizeme"
Storage> 13
** See help for drive backend at: https://rclone.org/drive/ **

Google Application Client Id
Setting your own is recommended.
See https://rclone.org/drive/#making-your-own-client-id for how to cre
ate your own.
If you leave this blank, it will use an internal key which is low perf
ormance.
Enter a string value. Press Enter for the default ("").
client_id>
Google Application Client Secret
Setting your own is recommended.
Enter a string value. Press Enter for the default ("").
client_secret>
```

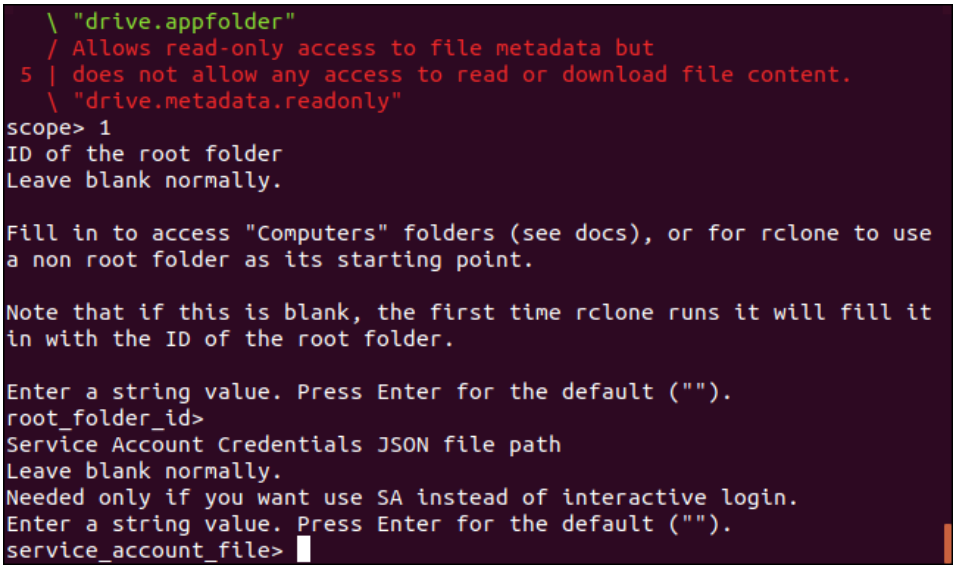
Again, just press “Enter.” You’re asked to provide the scope that `rclone` will have when it is operating on your Google Drive. Press “1” and then press “Enter.”



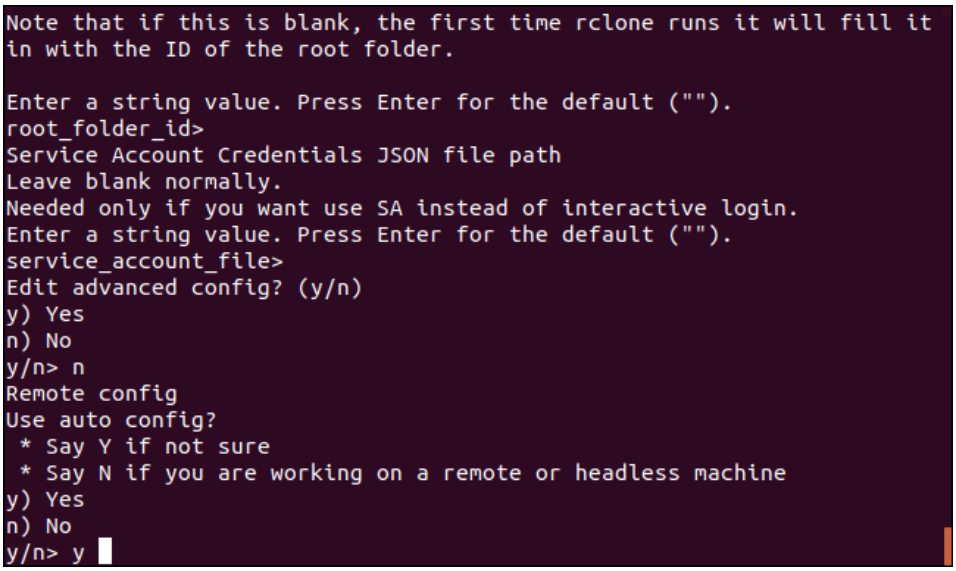
For the “ID of the root folder”, just press “Enter.”



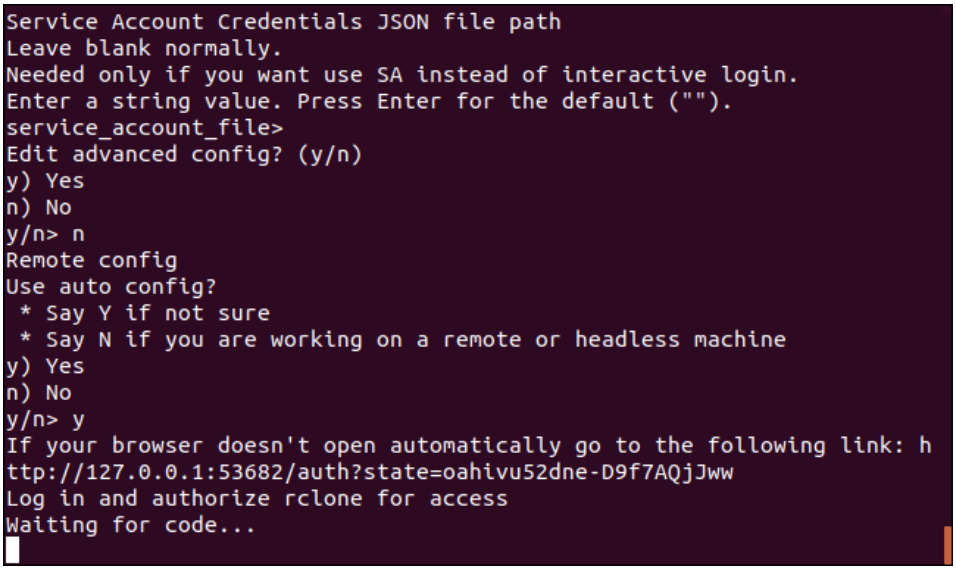
At the “Service Account Credentials” prompt, press “Enter.”



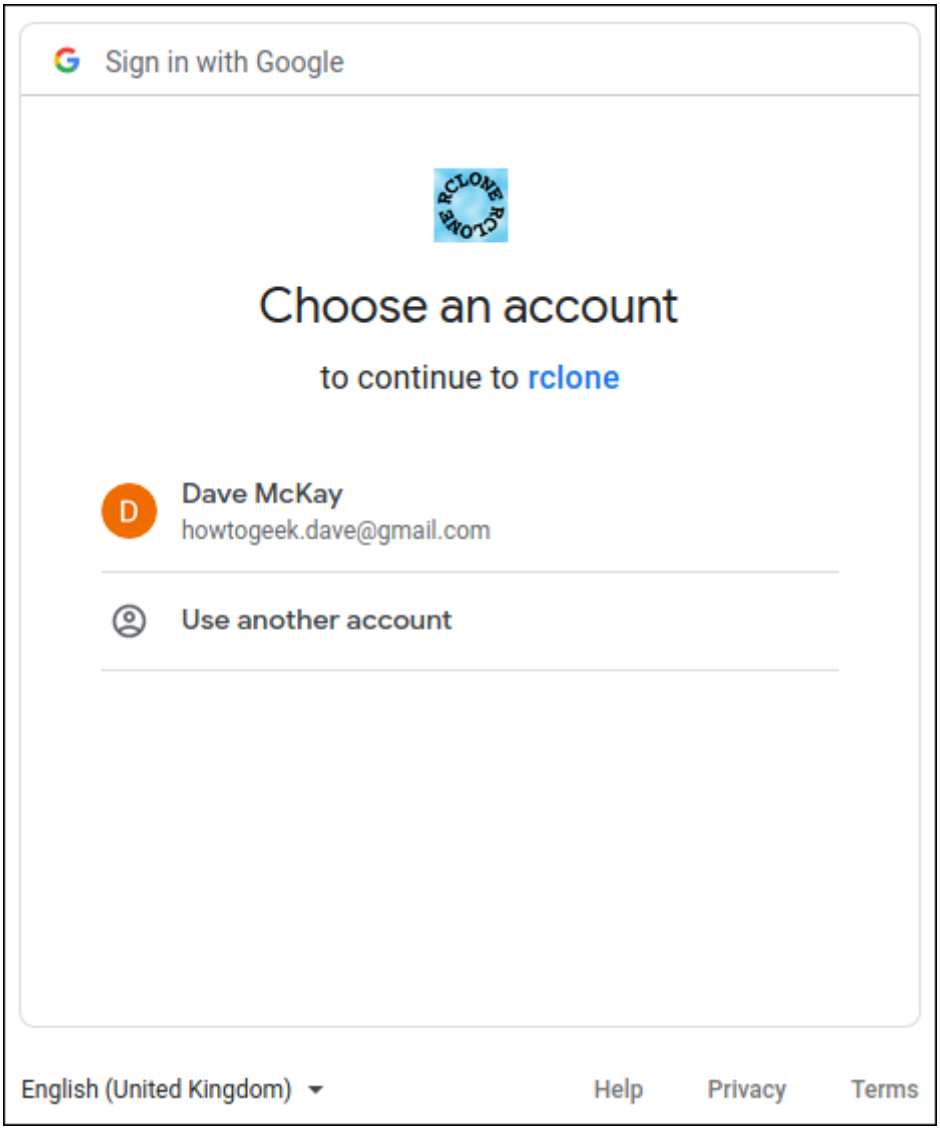
At the “Edit advanced config” prompt, just press “Enter.” At the “Use auto config” menu, press “y” and then press “Enter.”



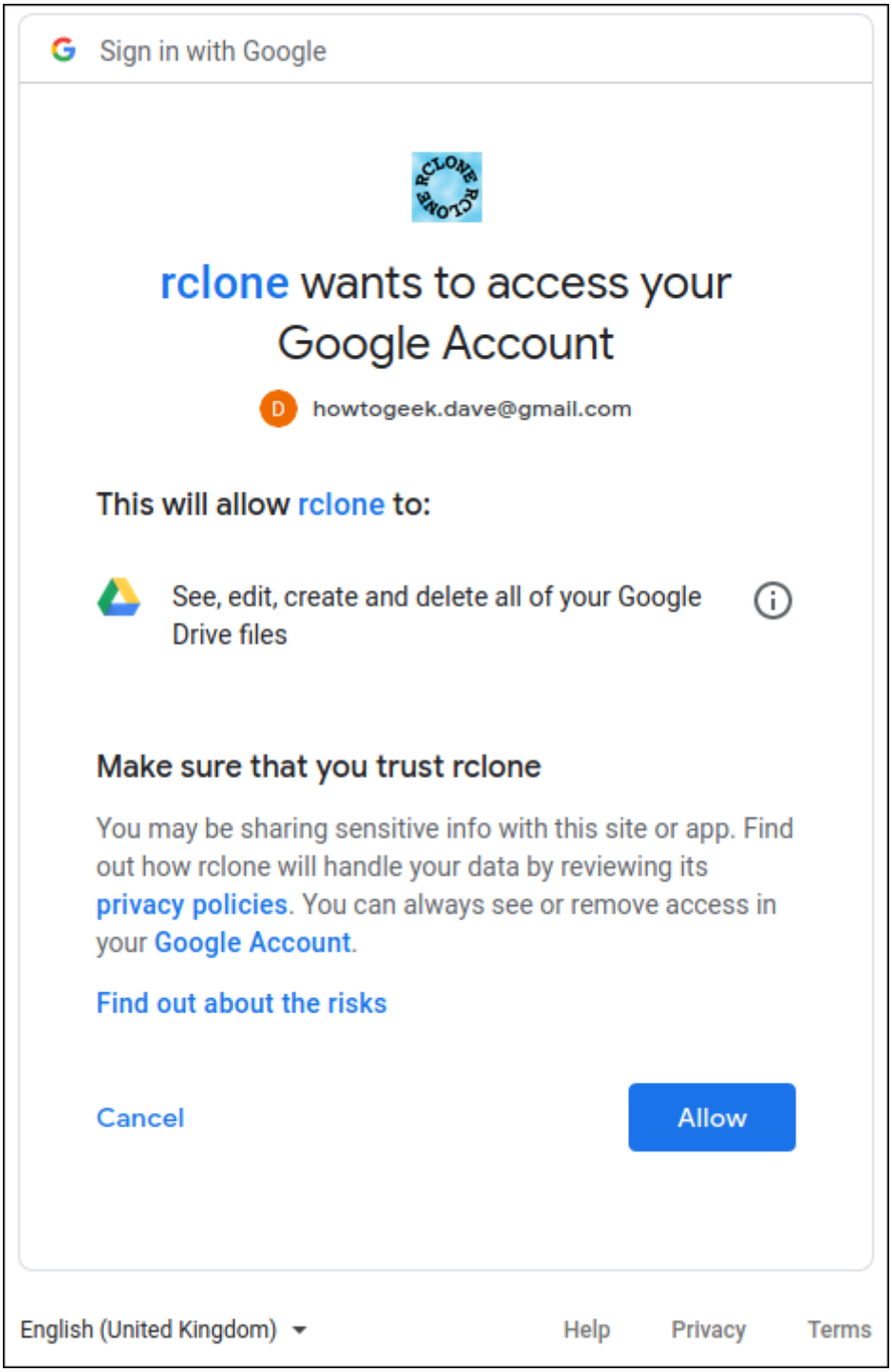
This causes `rclone` to communicate to your Google Drive, and to launch your browser to allow you to give permission for `rclone` to interact with your Google Drive.



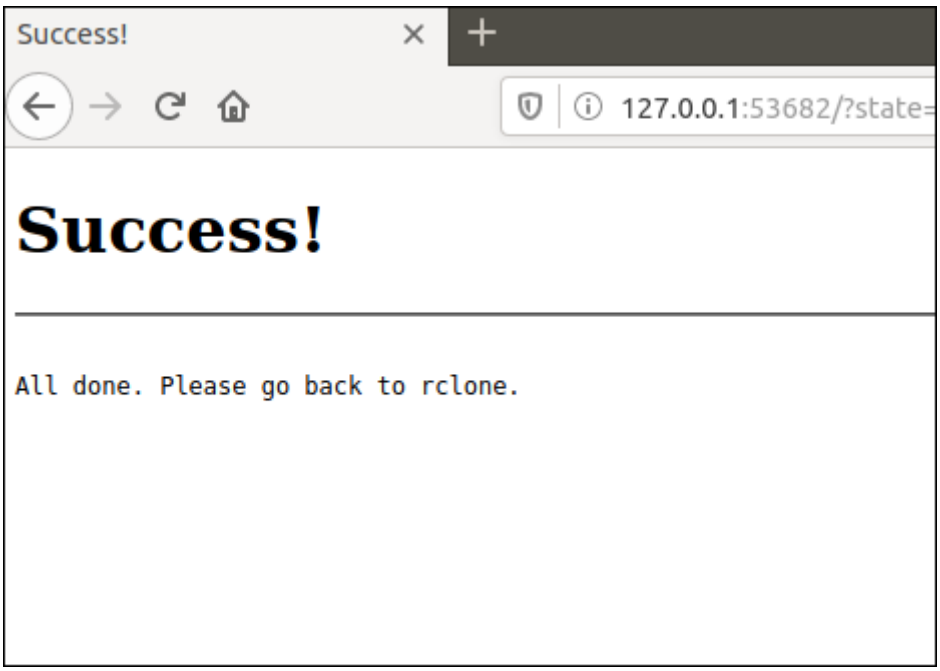
In your browser window, click on the Google account you wish to use.



Click the “Allow” button to allow `rclone` to have access to your Google Drive.



When authenticate has completed, you'll see a "Success!" message in the browser window. You can close the browser and return to the terminal window.



At the “Configure this as a team drive” prompt, type “n” and then press “Enter.”

```
service_account_file>
Edit advanced config? (y/n)
y) Yes
n) No
y/n> n
Remote config
Use auto config?
 * Say Y if not sure
 * Say N if you are working on a remote or headless machine
y) Yes
n) No
y/n> y
If your browser doesn't open automatically go to the following link: h
ttp://127.0.0.1:53682/auth?state=oahivu52dne-D9f7AQjJww
Log in and authorize rclone for access
Waiting for code...
Got code
Configure this as a team drive?
y) Yes
n) No
y/n> n
```

At the “Yes, Edit, Delete” menu type “y” and then press “Enter.”

```
Waiting for code...
Got code
Configure this as a team drive?
y) Yes
n) No
y/n> n
--------------------
[google-drive]
type = drive
scope = drive
token = {"access_token":"ya29.Il-1ByqSBaaxrs-PmHeFL8QdUgnPN5AuxOHLxUN8
rAZIzWnYVUvupY_vVaWAISMCy8tuW_g8Fovi2rN2mWYP_eM676Jpqog-VU2yzcnKx_Gtlt
9Igkp1bIPA14YBFCVGYQ","token_type":"Bearer","refresh_token":"1//036by-
fvB-tz3CgYIARAAGAMSNgF-L9Ir_dvp21IfOmnCD7vMPwiWtccLn75MRnZ7LErWaIKYWGv
flw_vG9TbebhI0A1wSXv_1w","expiry":"2019-12-15T11:15:43.78853476-05:00"
}
--------------------
y) Yes this is OK
e) Edit this remote
d) Delete this remote
y/e/d> y
```

At the final menu, type “q” and press “Enter.”

```
flw_vG9TbebhI0A1wSXv_1w","expiry":"2019-12-15T11:15:43.78853476-05:00"
}
--------------------
y) Yes this is OK
e) Edit this remote
d) Delete this remote
y/e/d> y
Current remotes:

Name                 Type
====                 ====
google-drive         drive

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
```

The rclone Back Up Script

The `rclone` application is very feature-rich. That’s great, but it does mean there are a lot of options. The command we’re going to look at below copies files from your local computer to your Google Drive. This is a one-way copy to the cloud; it isn’t a two-way synchronization between your Google Drive and your local computer—although `rclone` can do that. We’re using this as a basic form of off-site backup.

Type (or copy and paste) this into a text editor and save it to your computer. We called it `gbk.sh`. You can call it whatever makes sense to you.

```
#!/bin/bash

/usr/bin/rclone copy --update --verbose --transfers 30
```

Here's what the parameters mean:

- **copy**: Copy the files from the local computer to the remote storage, skipping over files that are already present on the remote storage.
- **–update**: Skip any files that are on the remote storage that have a modified time that is newer than the file on the local computer.
- **–verbose**: Gives information about every file that is transferred.
- **–transfers 30**: This sets the number of files to copy in parallel.
- **–checkers 8**: How many "checkers" to run in parallel. Checkers monitor the transfers that are in progress.,
- **–contimeout 60s**: The connection timeout. It sets the time that `rclone` will try to make a connection to the remote storage.
- **–timeout 300s**: If a transfer becomes idle for this amount of time, it is considered broken and is disconnected.
- **–retries 3**: If there are this many errors, the entire copy action will be restarted.
- **–low-level-retries 10**: A low-level retry tries to repeat one failing operation, such as a single HTTP request. This value sets the limit for the number of retries.
- **–stats 1s**: `rclone` can provide statistics on the transferred files. This sets the frequency of update of the statistics to one second.
- **"/home/dave/Documents"**: The local directory to we're going to copy to the remote storage.
- **"google-drive:LinuxDocs"**: The destination directory in the remote storage. Note the use of "google-drive", which is the name we gave to this remote connection during the the `rclone config` sequence. Also note the colon ":" that is used as a separator between the remote storage name and the directory name. Subdirectories are separated by the usual "/"

forward slash. If the destination directory does not exist, it will be created.

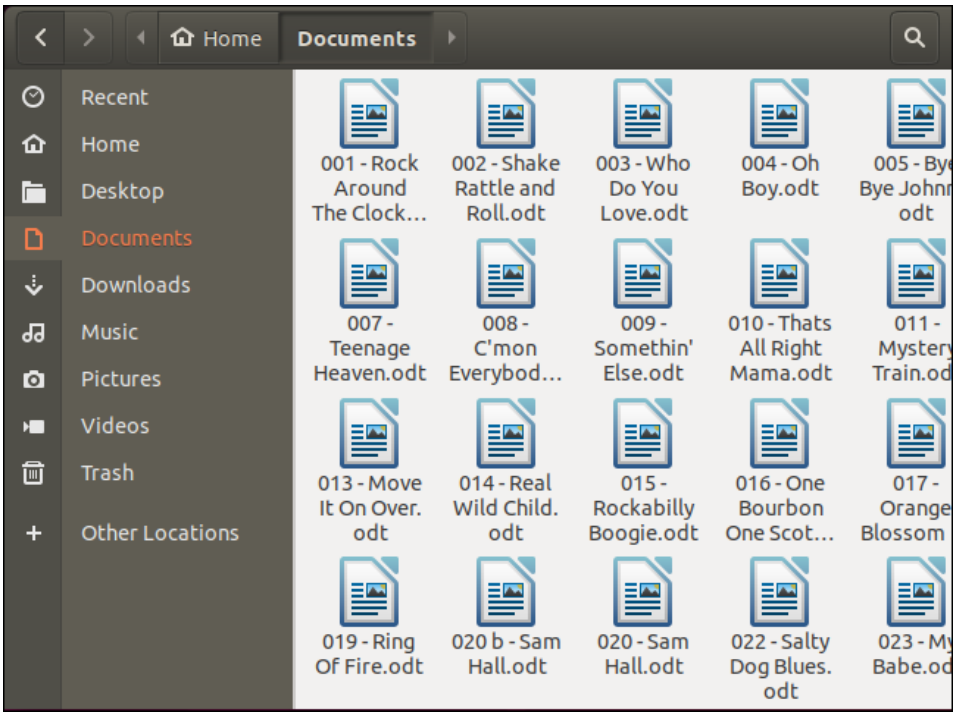
Some of these values are the defaults, but we've included them here so that we can discuss them. That way, if you need to change a value, you know which parameter to adjust.

Make the script executable with this command:

```
chmod +x gbk.sh
```

Running the Back Up Script

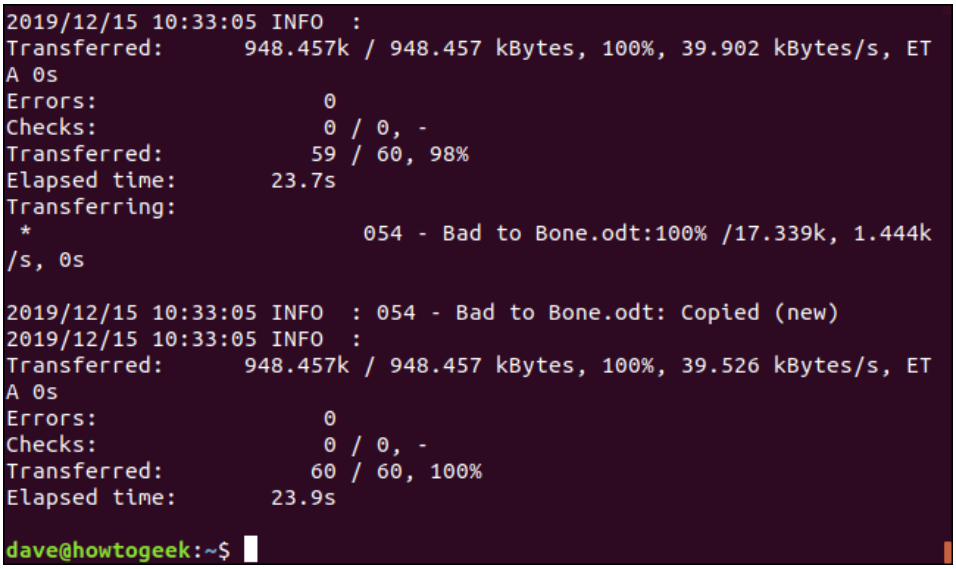
Our back up script is going to copy our Documents folder to our Google Drive. In our Documents folder, we've got a collection of sheet music.



We can launch the back up script with this command:

```
./gbk.sh
```

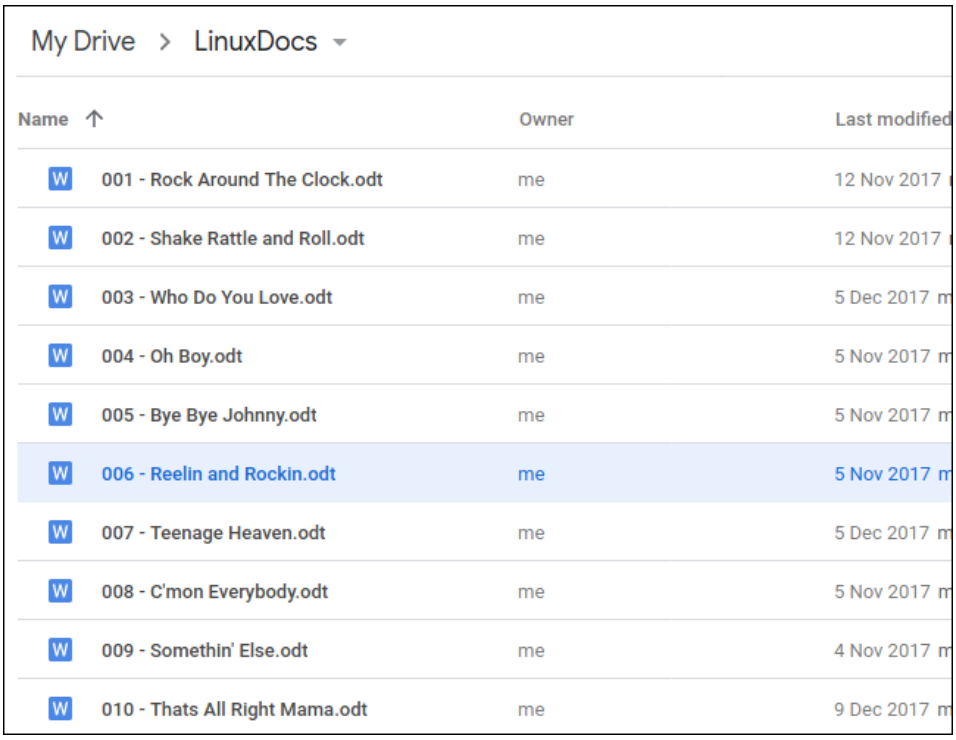
We asked for statistics updates every one second (`--stats 1s`), and we also asked for verbose output (`--verbose`). It'll come as no surprise then that we get a lot of screen output. It's usually a good option to turn on verbose output for new functionality so that you can spot problems. You can turn down the amount of output once you're happy things are running smoothly.



We get a final summary telling us 60 files were transferred with no errors. The transfer took roughly 24 seconds.

Let's check on our Google Drive and see what happened in our cloud storage.

A “LinuxDocs” directory has been created, so that looks promising. If we double-click it to take a look inside, we can see that the files have all been transferred to our Google Drive.



Using rclone to View Files On Google Drive

We can use `rclone` to peek into the folder on Google Drive, right from the terminal window:

```
rclone ls google-drive:/LinuxDocs
```

```
dave@howtogeek:~$ rclone ls google-drive:/LinuxDocs
    17748 058 - Ghost Riders.odt
    17215 056 - Going Back Home.odt
    15977 057 - Mercedes Benz.odt
    17623 060 - Ubangi Stomp.odt
    15653 059 - Psycho Killer.odt
    14973 055 - All Right Now.odt
    17755 054 - Bad to Bone.odt
    17969 052 - This Train.odt
    15292 053 - Highway to Hell.odt
    17856 043 - I Knew The Bride When She Used To Rock and Roll.odt
    15307 051 - All Over Now.odt
    15546 035 - Fight For Your Right.odt
    16572 050 - Jungle Rock.odt
    19979 020 b - Sam Hall.odt
    21042 020 - Sam Hall.odt
    16671 019 - Ring Of Fire.odt
    16942 023 - My Babe.odt
    16363 048 - Matchbox.odt
    14931 014 - Real Wild Child.odt
    13690 010 - Thats All Right Mama.odt
```

The Tip of the Iceberg

That's great that we can perform this type of copy straight from the command line. We can incorporate the use of our cloud storage into scripts, and we could schedule the execution of back up scripts using `cron`.

`rclone` has an absolute wealth of commands, you're encouraged to check out their documentation and excellent website. We've barely scratched the surface here, and a bit of reading and playing with `rclone` will pay back the effort many times over.

Strictly speaking, this isn't a true backup. It is an off-site, remote copy of your files and data, which is definitely a good thing to have, but it is just a copy of files. It doesn't offer versioning or other features that true backup solutions would offer.

So use `rclone` in conjunction with other backup techniques. As another layer to an existing backup regime `rclone` is an easy way to get your data stored in a location that is geographically removed from your home or office. And that's got to be a good thing.

DAVE MCKAY

Dave McKay first used computers when punched paper tape was in vogue, and he has been programming ever since. After over 30 years in the IT industry, he is now a full-time technology journalist. During his career, he has worked as a freelance programmer, manager of an international software development team, an IT services project manager, and, most recently, as a Data Protection Officer. His writing has been published by howtogeek.com, cloudsavvyit.com, itenterpriser.com, and opensource.com. Dave is a Linux evangelist and open source advocate. **READ FULL BIO »**

The above article may contain affiliate links, which help support How-To Geek.