

Only you can see this message



This story's distribution setting is on. [Learn more](#)

How to Backup Gsuite to Backblaze B2 (Using rclone + EC2)



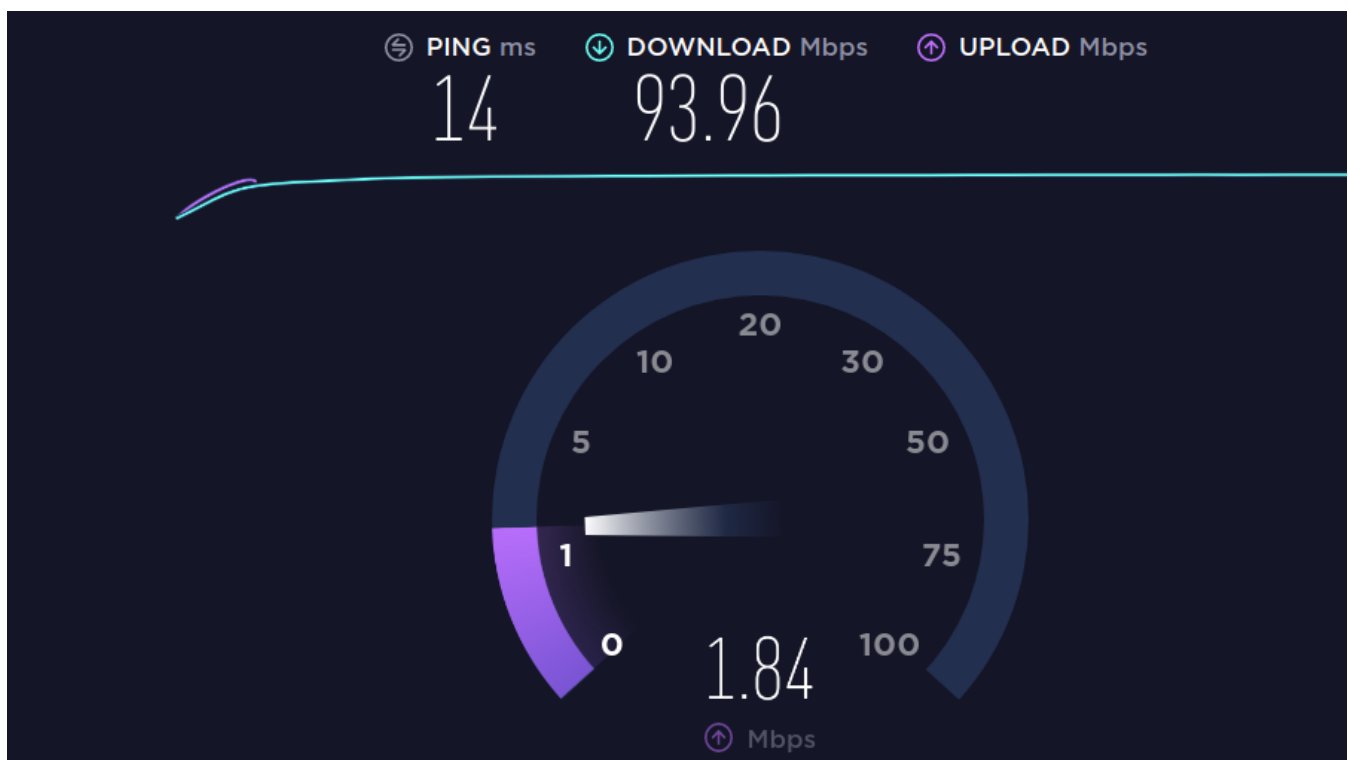
Daniel Rosehill

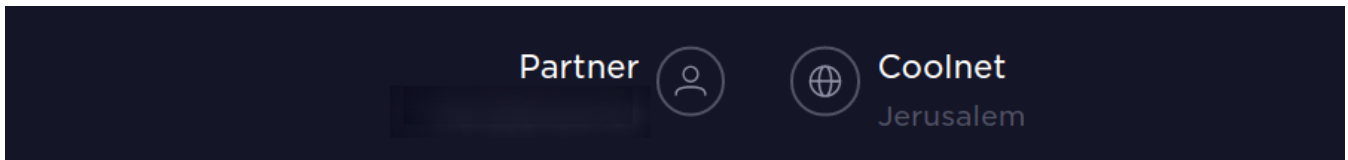
May 7 · 5 min read

In order to finish up my cloud backups last week, I wanted to move a copy of my **G Suite** storage to another cloud location.

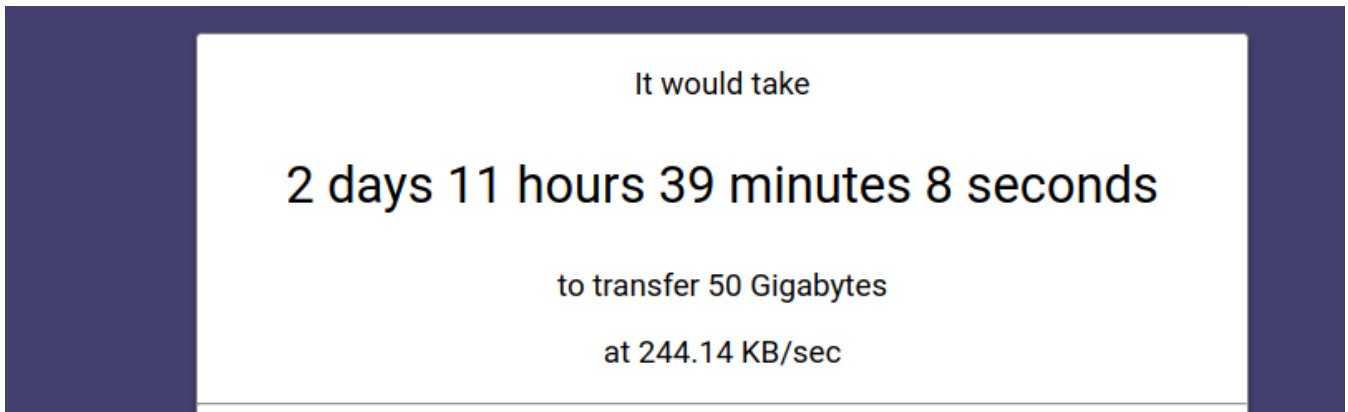
There was only one problem:

- My full Google Takeout was about 50 GB in size.
- My residential internet connection has an average upload speed





Putting those two numbers together things were not looking good. Putting those two numbers together yielded this result:



Therefore, the only viable solution was to go cloud-to-cloud.

Looking at the C2C backup space there are many interesting providers on the market like Skyvia, Datto, Multcloud, Barracuda and more — but finding a solution that would capture an entire Google Takeout (versus a [G Suite](#) export that focused only on key data like email) and then move it up to Backblaze B2 was proving an uphill struggle.

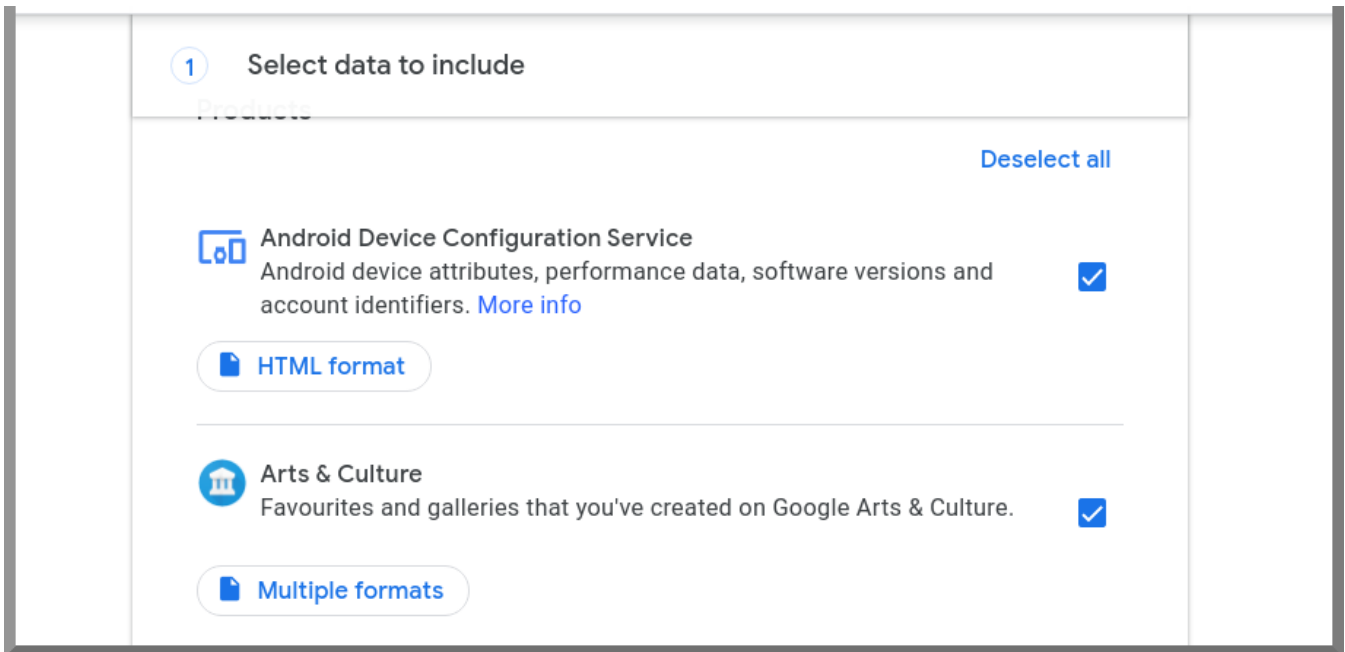
So — putting together a few rudimentary tools — I decided to do things the DIY way. Here's how.

. . .

1. Run a Google Takeout

Firstly, run a Google Takeout to capture all your data from Google.

My whole idea here was to have my whole [G Suite](#) user data in another cloud repository. So I chose to take out every service that I could. That added about 20 GB worth of YouTube videos. If you want to pull out a lighter archive, you can exclude services — but then, of course, it won't be a complete backup.

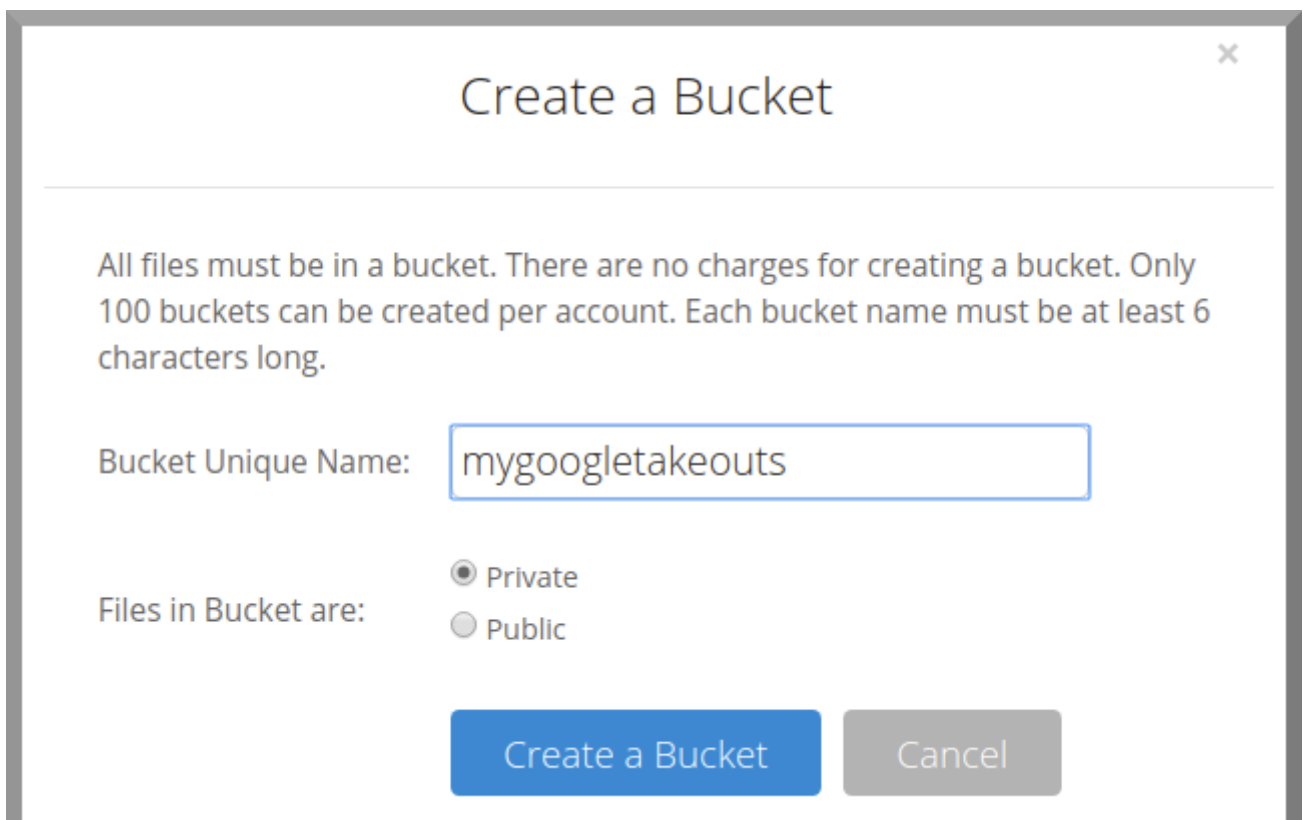


. . .

2. Get Everything Ready on the B2 Side

Of course, you should also have everything ready to go on your destination — which in this case is Backblaze B2.

As I was backing up to Backblaze B2 I configured an access key for rclone on EC2 and a bucket.



And creating the key:

×

Add Application Key

Name of Key:
(keyName)

rcloneonec2

Allow access to Bucket(s):
(optional)
(bucketName)

All ▾

Type of Access:
(optional)
(capabilities)

☒ Read and Write

☐ Read Only

☐ Write Only

File name prefix:
(optional)
(namePrefix)

Allow access to file names that start with this.

Duration (seconds):
(optional)
(validDurationSeconds)

Positive integer less than 1000 days (in seconds).

Create New Key

Cancel

. . .

3. Fire up an EC2 Instance With a Desktop Environment

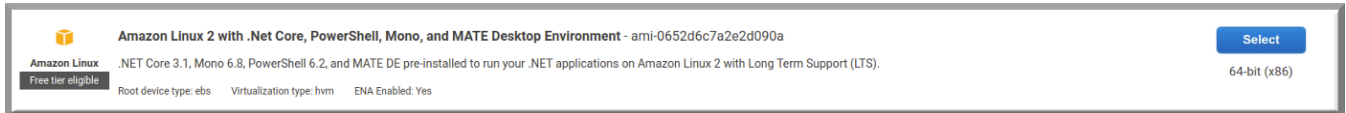
In AWS, you'll want to create an instance that you can get a GUI / desktop environment with easily.

I simply searched the Amazon Machine Image (AMI) library for 'desktop'

Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch an instance.

This template running Amazon Linux is fine:



Make sure you select an instance type that's going to have enough storage to hold your Takeouts after you download them and before you upload.

My Takeout came to about 50 GB so I went for an instance with 75 GB to leave a little bit of room for the operating system:

<input type="checkbox"/>	General purpose	t3.2xlarge	8	32	EBS only	Yes	Up to 5 Gigabit	Yes
<input checked="" type="checkbox"/>	General purpose	m5ad.large	2	8	1 x 75 (SSD)	Yes	Up to 10 Gigabit	Yes
<input type="checkbox"/>	General purpose	m5ad.xlarge			1 x 150 (SSD)	Yes	Up to 10 Gigabit	Yes

Launch your instance.

...

4. Start the Server, Set up VNC Server and Rclone

Go get a graphical user interface (GUI) so that we can just download our Takeouts from the Gsuite interface we need to install a GUI.

Firstly, download the .pem key file and then SSH into the EC2 instance.

As this machine already came with one installed, we simply need to run:

```
vncserver :1
```

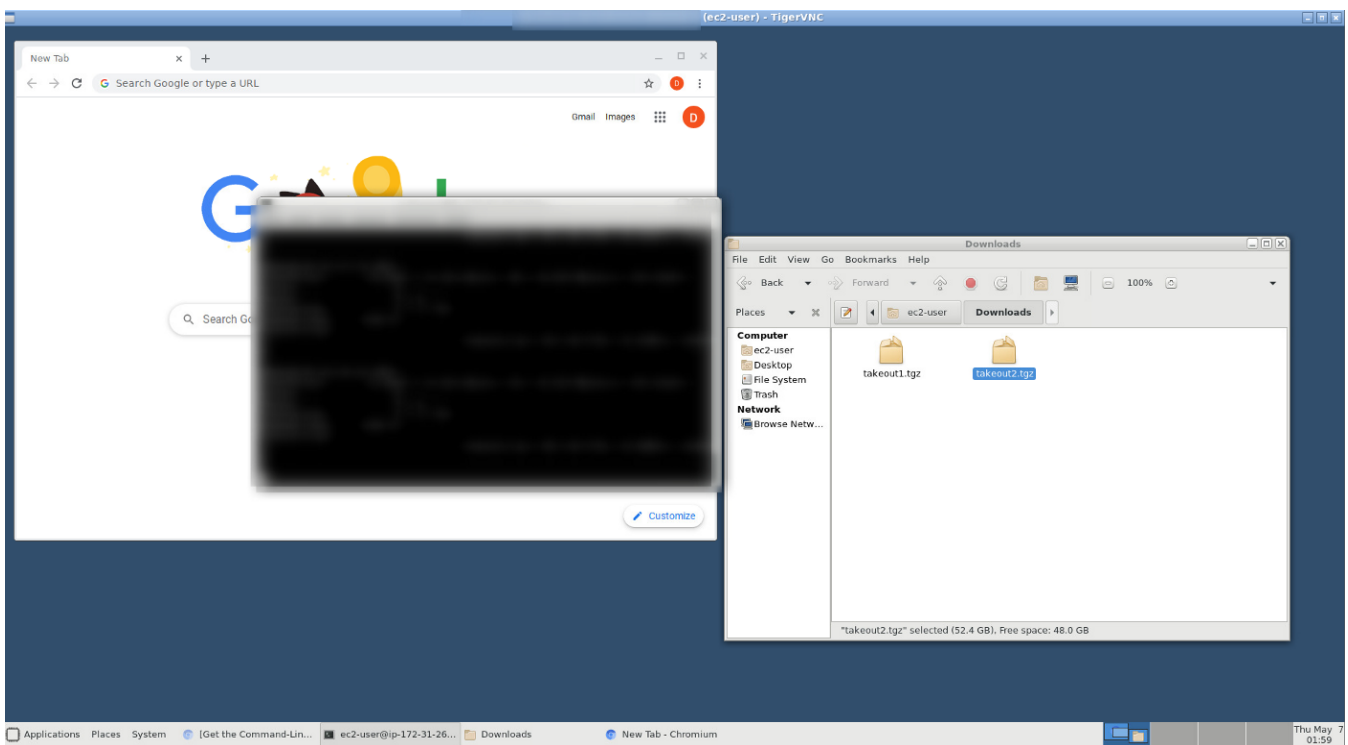
in a terminal

In order to start running the VNC server.

After installing tinyvnc on the computer we are connecting from, and after copying the .pem file into a local directory we then need to run this command — replacing PEM_FILE with the location of our file of course we need to create an SSH connection to establish the VNC tunnel:

```
ssh -L 5901:localhost:5901 -i PEM_FILE ec2-user@INSTANCE_IP
```

You should now be able to log in to a VNC view of your EC2 desktop:



• • •

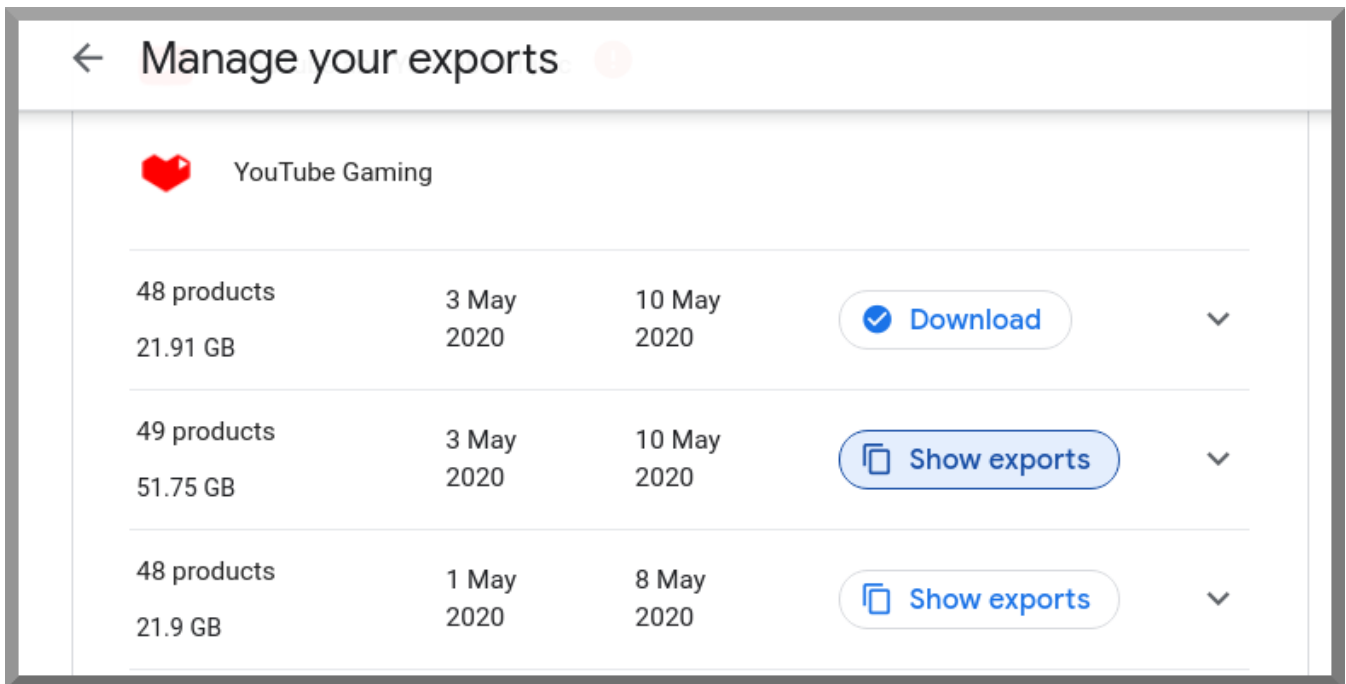
5. Download Your Google Takeout

Next, in your EC2 machine, download your Google Takeout.

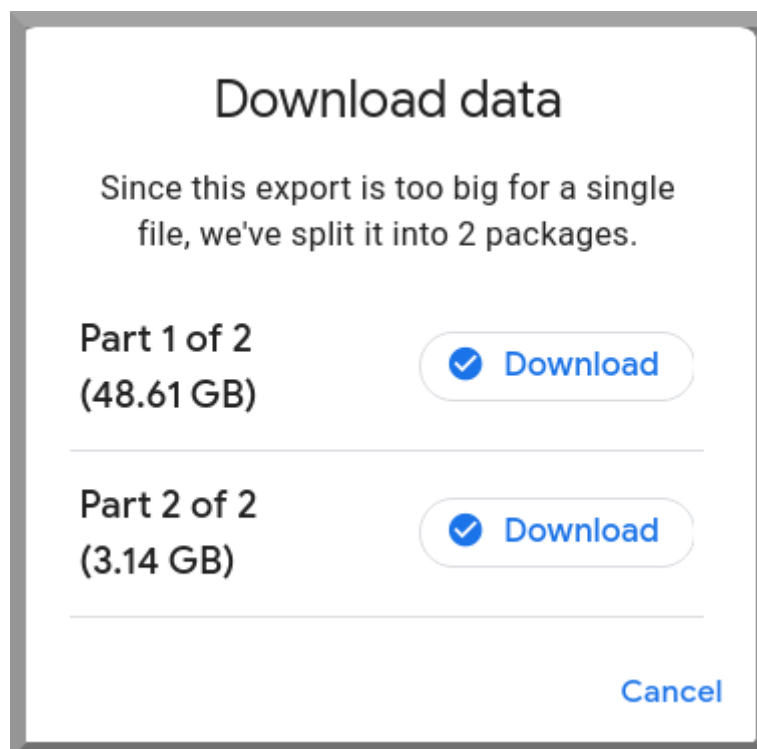
If we were simply moving over a Gdrive, we could do this totally cloud to cloud through rclone.

But because we're moving a Takeout up to B2 we need an actual web browser — hence the GUI and our choice of instance type.

To do this navigate to the Google Takeout and select the export that you need:

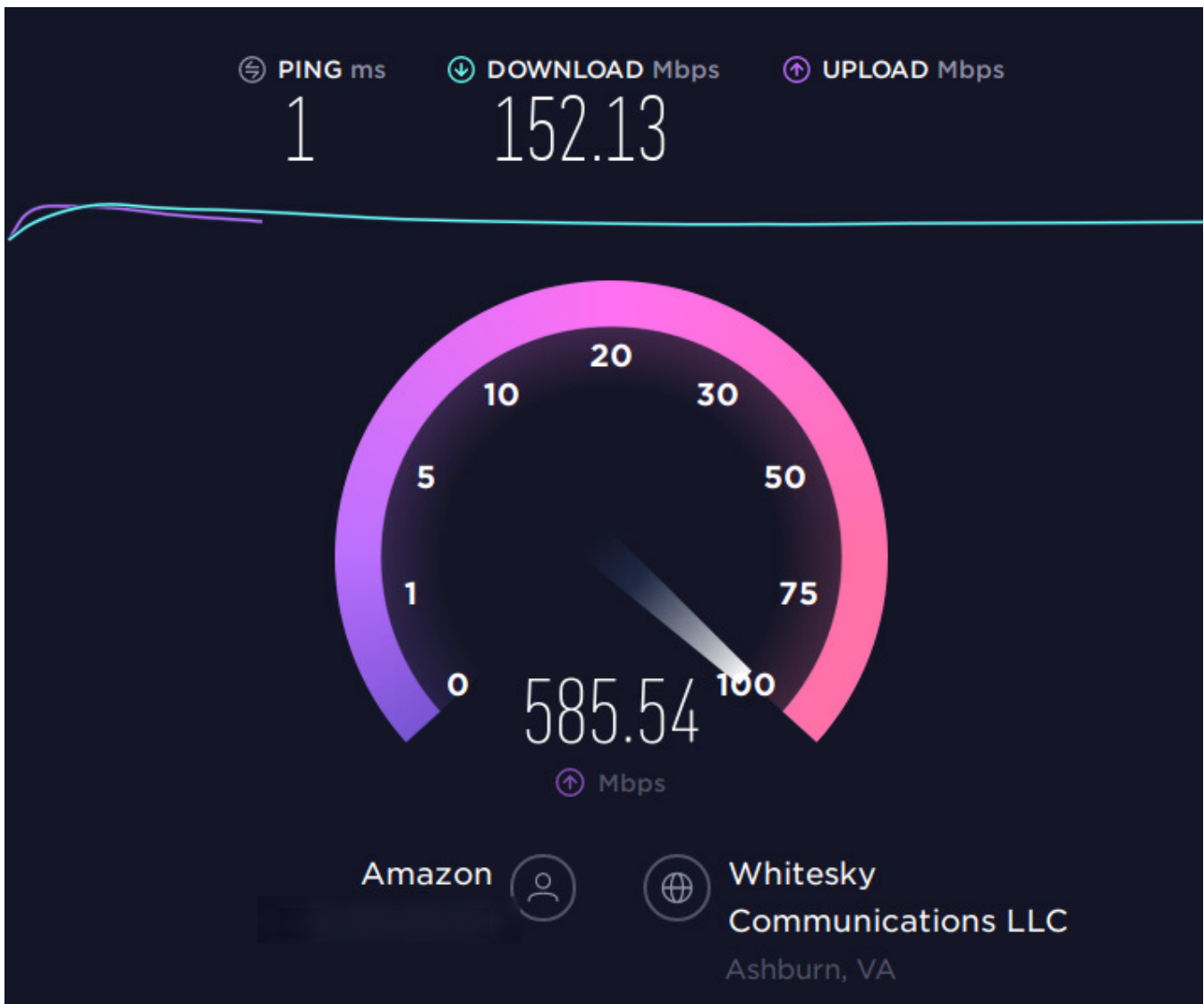


Download your files onto the EC2 instance:



The instance should have a fast connection — almost certainly faster than what you have at home — so this shouldn't take too long.

While you're doing that you might want to see how fast the line on your instance is. Here's what I got:



That's almost 300 times the upload speed of my home connection!

. . .

6. Configure Rclone

The rest is pretty straightforward.

Install the rclone command line interface (CLI):

```
https://aws.amazon.com/amazon-linux-2/  
[ec2-user@ ~]$ sudo yum install rclone
```

That's:


```
sudo yum install rclone
```

Then start the configuration process by entering:

```
rclone --config
```

The CLI will guide through the installation steps. You simply need to select the storage type and input the access key you created in B2:

```
Choose a number from below, or type in your own value
1 / A stackable unification remote, which can appear to merge the contents of several remotes
  \ "union"
2 / Alias for a 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"
```

Finally, it's time to run the big command!

With your Takeout archives in the folder you want them to be in you can simply run a sync to push them up to B2:





```
rclone -v sync takeouts/ B2:/yourbucket
```

And sit back and observe:

```
2020/05/07 02:07:35 INFO :
Transferred:      41.581G / 51.924 GBytes, 80%, 21.500 MBytes/s, ETA 8m12s
Errors:           0
Checks:           0 / 0, -
Transferred:      1 / 2, 50%
Elapsed time:     33m0.3s
Transferring:
```

I moved about 50 GB of data up to the cloud in roughly 30 minutes — as opposed to the time estimate of almost 3 days (running 24/7) had I chosen to run it from my local area network. So cloud backups can definitely save time and frustration.

Use the web UI to double check that the files have uploaded as expected:

<input type="checkbox"/>	 takeout1.tgz	3.4 GB 05/07/2020 04:35	
<input type="checkbox"/>	 takeout2.tgz	52.4 GB 05/07/2020 04:45	

And that’s it!

Follow these steps to back your **G Suite** Takeouts up to B2!

[Backup](#) [Rclone](#) [AWS](#) [Ec2](#) [Technology](#)

[About](#) [Help](#) [Legal](#)

Get the Medium app

