Enumeration and Restoration of Cloud Backups Created by rclone

This document assumes you have access to a *nix machine with rclone installed (it's on most repos) as well as optionally FUSE, and that you have access to the needed authentication and decryption credentials as well as the bucket ID the needed data is stored behind.

Demonstration Environment

For the purpose of this document a simplified bucket was created containing the following files demonstrating various states one might find a file in:

bar.txt	Unencrypted plain text file
fd8ufhgtl3g6rtpj0oqlc1jqlg	Plain text file "foo.txt" encrypted with filename encryption also enabled
ibnf43gvj2rvtucd3sounjgbeg	ubuntu_mini.iso encrypted with filename encryption enabled
ubuntu_mini.iso.bin	encrypted without filename encryption - note the ".bin" extension

Configuring rclone

You'll need to do this at least once on any host you hope to access files from.

1. # rclone config
2. Hit "n" for "New remote"
3. Enter a descriptive name such as "b2" - you'll want to keep this short.
4. Select a storage type. At the time of this writing "5" is the correct storage type for B2 but read the options list!
5. For "account" enter the application ID for the bucket in question.
6. For "key" enter the application key corresponding to the application ID above.
7. Accept default for hard_delete and hit "n" to skip advanced config, then hit "y" to confirm.
8. You will now see a list of current remotes including the one you just set up. You now need to create the crypted connection to this remote:
9. Enter "n" for "new" at the e/n/d/r/c/s/q> prompt.
10. Enter a descriptive name such as "b2-crypt" - again keep it short and relate it to the name of the remote it's connected to.
11. Select the "crypt" storage type, currently "9" but again this may change so read the options list.
12. For the "remote" field, type in the name of the remote you created in Step 3 above, followed by the bucket ID, separated by a colon, for example: b2:2f5b0f12-e299-43ab-9fa1-4c5b3b2fbf08 (it's a good practice to use uuidgen to create Bucket IDs because while B2 supports arbitrary bucket names, they must be globally unique).
13. For the next two steps select whether or not to encrypt the filenames and directory names. This must match how the files were uploaded.
14. Select "y" to type in your own password, then paste in the encryption password twice when prompted. The password will not echo.
15. Select "y" to type in your own salt passphrase, following the same steps as before for the regular password.
16. (Optional) Select "y" to enter advanced config, and type "true" for show_mapping. This is handy if filenames are encrypted and you want to see the decrypted filenames next to the encrypted filenames.
17. If everything looks good, enter "y" that it looks OK.
18. (Optional/Recommended) Enter "s" at the e/n/d/r/c/s/q prompt to protect your configuration then "a" to add a password.
19. Hit "q" to exit configuration.

Your configuration will be stored in ~/.config/rclone/ which *should* be portable.

Enumerating and Retrieving Files (using rclone commands)

Look here for everything: https://rclone.org/commands/

In terms of what's useful, the commands "rclone lsl" and "rclone ls" map roughly to "ls -lR" and "ls -R" and "rclone lsd" enumerates directory entries. The other useful command is "rclone copy" for obvious reasons. If you set a password to your rclone configuration you will be prompted to enter that each time you issue a command.

The command syntax is "rclone command path {otherpath}", so :

Examples:

```
rclone lsl b2-crypt: #Show all the contents
 56623104 2019-04-08 16:05:47.868000000 ubuntu_mini.iso

rclone copy b2-crypt:2f5b0f12-e299-43ab-9fa1-4c5b3b2fbf08/ubuntu_mini.iso
/local/destination/path # Copy one file

rclone copy b2-crypt:2f5b0f12-e299-43ab-9fa1-4c5b3b2fbf08/ /local/path/ #
Copy all the things from the bucket.

rclone move /local/path/ubuntu_mini.iso b2-crypt:2f5b0f12-e299-43ab-9fa1-
4c5b3b2fbf08/ # Move the file from local path to the cloud. But why?

rclone copy /local/path/ubuntu_mini.iso b2:2f5b0f12-e299-43ab-9fa1-
4c5b3b2fbf08/ # Copy it unencrypted.
```

Enumerating and Retrieving Files (using FUSE)

This works like any other FUSE mount, and now you can copy files into and out of your mountpoint with no fancy syntax needed. There are a couple of caveats to take note of:

- On a FreeNAS array, you have to turn a knob for FUSE to work. Go to System / Tunables, click the Add button, "fuse_load" in the Variable field, "YES" in the Value field, be sure Type is set to "loader" and then either bounce the array or type "kldload fuse" into the shell. "kldstat | grep fuse" to verify it's loaded.
- The mount command has to be amped off as of the time of this writing.

rclone mount example

```
rclone mount b2-crypt:2f5b0f12-e299-43ab-9fa1-4c5b3b2fbf08 /local/mount
/point &  # Note this is amped off. Use 'fg' when you want to dismount.
cp /local/mount/point/*.iso /home/user/iso/  # etc. It's just a regular
(if really slow) mount point now
```

Troubleshooting

- All the filenames are garbled
  - Go back into "rclone config" and edit your crypt entry. Chances are filename encryption was set when the files were uploaded but the crypt entry was setup with filename encryption on (chances are it'll be using "standard" filename encryption so start there)
- I don't see the files I expected to see
  - Are you looking for encrypted files but you mounted the unencrypted remote? b2 = unencrypted, b2-crypt = encrypted in the above example.
  - Did the files expire out past their retention period? Maybe they're actually gone - check the web console to verify