



Differential Backups with Restic

JaxLUG - Travis Phillips - 08/20/2025

Who Am I

I am a tall man dressed in black
that knows stuff and does things

What is Restic

- Per the documentation:
 - Restic is a fast and secure backup program
- Restic is written in Golang so it's cross platform
- Restic creates encrypted and compressed differential backups using a repository
 - Repos can be local or remote

What is a Differential Backup

- A differential backup is a backup solution that will only backup things that changed between backups
 - This is great for large data volumes
 - Increases the speed of back-ups after the initial backup, allowing for backup strategies that run every hour

Creating a Repo Locally

- Before you can begin, you must create a repo
- You can create a repo locally with the following:
 - `$ restic init --repo [/path/to/repo]`
- This will prompt you to enter a password for the repo.

```
restic init --repo /tmp/backup
enter password for new repository:
enter password again:
created restic repository 71b30fd77b at /tmp/backup

Please note that knowledge of your password is required to access
the repository. Losing your password means that your data is
irrecoverably lost.
```

Contents of the Repo Folder






- Overall, this is for the Restic program to keep track of the data.
- However, with this being local, you can backup and move this directory to external media.

```
❏ ➤ 🏠 ~ ➤ ls -l /tmp/backup
total 4
-r----- 1 owner owner 155 Jul 6 14:00 config
drwx----- 258 owner owner 5160 Jul 6 14:00 data
drwx----- 2 owner owner 40 Jul 6 14:00 index
drwx----- 2 owner owner 60 Jul 6 14:00 keys
drwx----- 2 owner owner 40 Jul 6 14:00 locks
drwx----- 2 owner owner 40 Jul 6 14:00 snapshots
```

Managing Keys

- You can create more than one password for the repo using the `key` command
- There is no permissions model - All keys have full access to the repo
- Follow the `key` command with a subcommand:
- `list`, `add`, `remove`, and `passwd` (changes a password)






Managing Keys

```
  /tmp/work  restic key list -r /tmp/backup  1 x  10s ⌵
```






enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)

ID	User	Host	Created

*ced884ea	owner	Firestorm	2025-07-06 14:00:50

```
  /tmp/work  restic key add -r /tmp/backup  ✓  4s ⌵
```





enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
enter new password:
enter password again:
saved new key with ID 77f47c228bf4a25c06bea9771da59b9305d54a418af715e96282bc6de45aab18

```
  /tmp/work  restic key list -r /tmp/backup  ✓  14s ⌵
```

enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)

ID	User	Host	Created

ced884ea	owner	Firestorm	2025-07-06 14:00:50
*77f47c22	owner	Firestorm	2025-07-06 15:26:09

```
  /tmp/work  ✓  5s ⌵
```


Support for Remote Repos

- Generally, a local backup isn't super useful
- Restic can support remote repos over using various URI protocol handlers:
 - SFTP
 - REST (Requires setting up a Restic REST server)
 - Cloud Storage Services:
 - S3, Minio, Wasabi, Alibaba, Backblaze, OpenStack, Azure, GCP, and many others using `rc1one`

Running a Backup

- Backups can be performed on demand
 - `$ restic backup -r [/path/to/repo]`
`[/path/to/backup]`
 - Note: Your first backup will take the longest!
- Each backup is called a snapshot

Running a Backup

```
❏ ~ ➤ mkdir /tmp/work
❏ ~ ➤ cd /tmp/work
❏ /tmp/work ➤ ls
❏ /tmp/work ➤ echo "Hello World" > test1.txt
❏ /tmp/work ➤ echo "Potato" > shopping_list.txt
❏ /tmp/work ➤ restic backup -r /tmp/backup /tmp/work
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
created new cache in /home/owner/.cache/restic
no parent snapshot found, will read all files
[0:00]          0 index files loaded

Files:          2 new,          0 changed,          0 unmodified
Dirs:           2 new,          0 changed,          0 unmodified
Added to the repository: 1.456 KiB (1.115 KiB stored)

processed 2 files, 19 B in 0:04
snapshot 802bbd1c saved
❏ /tmp/work ➤
```

Running a Backup - Stdin

- You can also have Restic back up from the output of a command using `--stdin-from-command` switch
 - `$ restic backup -r [/path/to/repo] --stdin-from-command mysqldump [...]`
- Or from stdin from a pipe using the `--stdin` flag
 - `$ gzip big.dat | restic backup -r [/path/to/repo] --stdin`




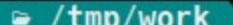
Viewing Snapshots

- snapshots can be viewed with the following:
 - `$ restic snapshots -r [/path/to/repo]`

```
❏ /tmp/work ➤ restic snapshots -r /tmp/backup
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
ID              Time                Host      Tags      Paths      Size
-----
802bbd1c 2025-07-06 14:17:36 Firestorm /tmp/work 19 B
-----
1 snapshots
❏ /tmp/work ➤
```

Viewing the Contents of a Snapshot

- You can use the ls subcommand with the ID
 - `$ restic ls -r [/path/to/repo] [snapshot_id]`

```
ID          Time                Host          Tags          Paths          Size
-----
802bbd1c    2025-07-06 14:17:36    Firestorm          /tmp/work    19 B
-----
1 snapshots
  /tmp/work restic ls -r /tmp/backup 802bbd1c ✓ 4s ⌛
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
[0:00] 100.00% 3 / 3 index files loaded
snapshot 802bbd1c of [/tmp/work] at 2025-07-06 14:17:36.051859883 -0400 EDT by owner@Firestorm filtered by []:
/tmp
/tmp/work
/tmp/work/shopping_list.txt
/tmp/work/test1.txt
  /tmp/work ✓ 4s ⌛
```

Remove a Snapshot

- Use the forget subcommand with the ID
 - `$ restic forget -r [/path/to/repo] [snapshot_id]`

```

/tmp/work ➤ restic snapshots -r /tmp/backup ✓ 4s ⌵
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
ID              Time                Host      Tags      Paths      Size
-----
802bbd1c 2025-07-06 14:17:36 Firestorm /tmp/work 19 B
15a0fbf9 2025-07-06 14:34:33 Firestorm /tmp/work 19 B
-----
2 snapshots

/tmp/work ➤ restic forget -r /tmp/backup 15a0fbf9 ✓ 4s ⌵
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
[0:00] 100.00% 1 / 1 files deleted

/tmp/work ➤ restic snapshots -r /tmp/backup ✓ 4s ⌵
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
ID              Time                Host      Tags      Paths      Size
-----
802bbd1c 2025-07-06 14:17:36 Firestorm /tmp/work 19 B
-----
1 snapshots
```

Restoring from Backup

- There are three different ways to access your backups
 - Directly restoring the file
 - Mounting as a read-only fuse filesystem
 - My personal favorite!
 - Print the contents to stdout

Restoring from Backup - Direct

- Can be used to restore an entire snapshot in place or to a different directory
 - `$ restic restore -r [/path/to/repo] [snapshot_id] --target [/path/to/restore]`

```
❏ /tmp/work ➤ cat /tmp/work/test1.txt ✓
Hello World
❏ /tmp/work ➤ echo "Whoops..." > /tmp/work/test1.txt ✓
❏ /tmp/work ➤ cat /tmp/work/test1.txt ✓
Whoops...
❏ /tmp/work ➤ restic restore -r /tmp/backup 802bbd1c --target /tmp/restore ✓
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
[0:00] 100.00% 4 / 4 index files loaded
restoring snapshot 802bbd1c of [/tmp/work] at 2025-07-06 14:17:36.051859883 -0400 EDT by
owner@Firestorm to /tmp/restore
Summary: Restored 4 files/dirs (19 B) in 0:00
❏ /tmp/work ➤ cat /tmp/restore/tmp/work/test1.txt ✓ 4s ⌛
Hello World
❏ /tmp/work ➤
```

Restoring from Backup - Direct

- You can also use the `latest` instead of a snapshot ID.
 - `$ restic restore latest -r [/path/to/repo] --target [/path/to/restore]`
- There is also the `--include`, `--path`, and `--exclude` flags to target specific files.
 - `$ restic restore latest -r [/path/to/repo] --target [/path/to/restore] --include test1.txt`

Restoring from Backup – Fuse FS

- One of the cooler features is mounting as a read-only Fuse FS.
- Mounts the entire repo and provides snapshot folders based off the timestamp, or a symlink to latest
 - `$ restic mount -r [/path/to/repo] [/mount/dir]`

Restoring from Backup – Fuse FS

```

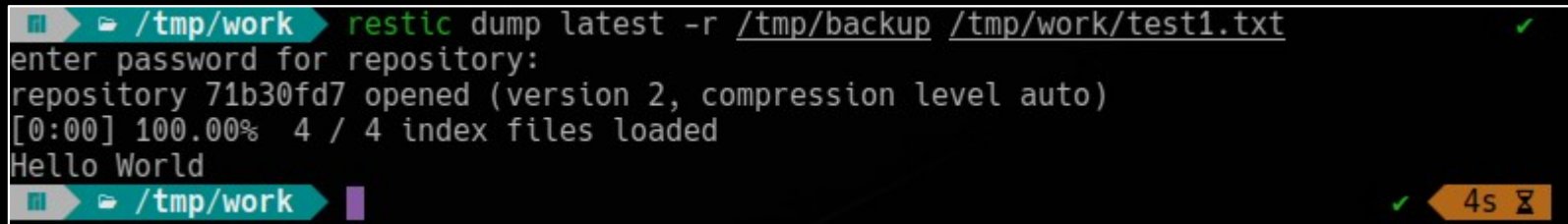
[1] /tmp/work cat /tmp/work/test1.txt 1 x
Whoops...
[1] /tmp/work mkdir /tmp/restore ✓
[1] /tmp/work restic mount -r /tmp/backup /tmp/restore ✓
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
[0:00] 100.00% 4 / 4 index files loaded
Now serving the repository at /tmp/restore
Use another terminal or tool to browse the contents of this folder.
When finished, quit with Ctrl-c here or umount the mountpoint.
[1]

work : zsh [C] +

[1] /tmp/work cd /tmp/restore ✓
[1] /tmp/restore ls ✓
hosts ids snapshots tags
[1] /tmp/restore cd snapshots ✓
[1] /tmp/restore/snapshots ls ✓
2025-07-06T14:17:36-04:00 latest
[1] /tmp/restore/snapshots cat latest/tmp/work/test1.txt ✓
Hello World
[1] /tmp/restore/snapshots cp latest/tmp/work/test1.txt /tmp/work/test1.txt ✓
cp: overwrite '/tmp/work/test1.txt'? y
[1] /tmp/restore/snapshots cd /tmp/work ✓
[1] /tmp/work cat test1.txt ✓
Hello World
[1] /tmp/work [1] ✓
```

Restoring from Backup – Stdout

- Restic can also dump to stdout.
- This is useful to pipe into another command, such as mysql
 - `$ restic dump [snapshot_id] -r [/path/to/repo] [/path/to/file]`

```
/tmp/work $ restic dump latest -r /tmp/backup /tmp/work/test1.txt  
enter password for repository:  
repository 71b30fd7 opened (version 2, compression level auto)  
[0:00] 100.00% 4 / 4 index files loaded  
Hello World  
/tmp/work
```

Tidying Up with Environment Variable

- Restic supports environment variables to reduce switches needed and assist automation
- Examples:
 - `$ export RESTIC_REPOSITORY=/tmp/backup`
 - `$ export RESTIC_PASSWORD=P@ssw0rd1`
- With those in place, you no longer need to provide the repo path, nor the password.

Tidying Up with Environment Variable

```

/tmp/work restic key list -r /tmp/backup 1 x
enter password for repository:
repository 71b30fd7 opened (version 2, compression level auto)
ID      User  Host      Created
-----
*ced884ea owner  Firestorm 2025-07-06 14:00:50
77f47c22 owner  Firestorm 2025-07-06 15:26:09
-----

/tmp/work export RESTIC_REPOSITORY=/tmp/backup ✓ 5s ⌵
/tmp/work read -r -s PASSWORD ✓
/tmp/work export RESTIC_PASSWORD=${PASSWORD} ✓
/tmp/work unset PASSWORD ✓
/tmp/work restic key list ✓

repository 71b30fd7 opened (version 2, compression level auto)
ID      User  Host      Created
-----
*ced884ea owner  Firestorm 2025-07-06 14:00:50
77f47c22 owner  Firestorm 2025-07-06 15:26:09
-----

/tmp/work

```

Backup Exclusions

- Restic supports exclusions via a standalone switch (`--exclude`, `--iexclude`), or as a file list of exclusions (`--exclude-file`, `--iexclude-file`)
- This is useful to avoid backing up cache files!

```
❏ ➤ ⌂ ~ ➤ cat /home/owner/.local/share/restic/exlcudes.txt
/home/*/**/* .pyc
/home/*/**/__pycache__/**
/home/*/**/node_modules/**
/home/*/.cache/**
/home/*/.local/lib/python*/site-packages/**
/home/*/.mozilla/firefox/*/Cache/**
❏ ➤ ⌂ ~ ➤
```


Remove Snapshot Following Policy

- Restic also has several `--keep-*` flags. These can be used to delete snapshots in accordance with a retention policy.
 - `$ restic forget --prune --keep-hourly=24 --keep-weekly=4 --keep-monthly=6 --keep-yearly=1`

Sanity Checking a Repo

- Restic has a check command that can check the integrity and consistency of a repo
 - `$ restic check`
- It's worth doing this following a prune operation or just periodically

Exit Codes

- Restic does attempt to use exit codes that can provide details about how a command failed
- This is useful for scripting

0	Command was successful
1	Command failed, see command help for more details
2	Go runtime error
3	<code>backup</code> command could not read some source data
10	Repository does not exist (since restic 0.17.0)
11	Failed to lock repository (since restic 0.17.0)
12	Wrong password (since restic 0.17.1)
130	Restic was interrupted using SIGINT or SIGSTOP

Automation

- Personally, I use two automations with Restic
 - A manual script I can invoke to run a backup
 - A cron job that runs hourly

```
12 * * * * . /home/owner/.restic-env; /usr/bin/nice -n 19 /usr/bin/restic backup -q /home/owner --tag scheduled --exclude-file "/home/owner/.local/share/restic/exlcudes.txt"; /usr/bin/restic forget --prune --keep-hourly=24 --keep-weekly=4 --keep-monthly=6 --keep-very-early=1
```

Demo and Script Review

```
function main() {  
    # Setup the environment.  
    export RESTIC_REPOSITORY="/backup/${USER}"/;  
    prompt_for_password;  
  
    # Run the backup.  
    run_restic_backup;  
  
    # Dump the snapshots.  
    list_snapshots;  
  
    # Run a check on the latest snapshot.  
    run_repo_check_sample;  
  
    # Prune the back-ups.  
    prune;  
  
    # Remove the exports used for restic.  
    cleanup;  
}
```

Questions

- Travis Phillips
- **Github:**
 - <https://github.com/jaxhax-travis>
- **Restic Docs:**
 - <https://restic.readthedocs.io/en/latest/>

