Backup Strategies

Sumner Evans

October 22, 2020

Mines Linux Users Group

Backup Principles

Why backup? I

Computers were a mistake. But the bigger mistake was to give humans control over the computers.

Sometimes certain humans may write a program along the lines of:

```
with open("~/awesome", "w+") as f:
    f.writeline("Awesome program\n")
```

which doesn't do what you expect because by default Python doesn't expand ~ by default meaning this creates a directory named ~ in your working directory.

Naturally, to delete this directory, you would run `rm -rf ~`, right?

(I may or may not have first-hand experience with this situation.)

Why backup? I

Computers were a mistake. But the bigger mistake was to give humans control over the computers.

Sometimes certain humans may write a program along the lines of:

```
with open("~/awesome", "w+") as f:
    f.writeline("Awesome program\n")
```

which doesn't do what you expect because by default Python doesn't expand ~ by default meaning this creates a directory named ~ in your working directory.

Naturally, to delete this directory, you would run `rm -rf ~`, right?

(I may or may not have first-hand experience with this situation.)

Why backup? I

Computers were a mistake. But the bigger mistake was to give humans control over the computers.

Sometimes certain humans may write a program along the lines of:

```
with open("~/awesome", "w+") as f:
    f.writeline("Awesome program\n")
```

which doesn't do what you expect because by default Python doesn't expand ~ by default meaning this creates a directory named ~ in your working directory.

Naturally, to delete this directory, you would run `rm -rf ~`, right?

(I may or may not have first-hand experience with this situation.)

Why backup? I

Computers were a mistake. But the bigger mistake was to give humans control over the computers.

Sometimes certain humans may write a program along the lines of:

```
with open("~/awesome", "w+") as f:
    f.writeline("Awesome program\n")
```

which doesn't do what you expect because by default Python doesn't expand ~ by default meaning this creates a directory named ~ in your working directory.

Naturally, to delete this directory, you would run `rm -rf ~`, right?

(I may or may not have first-hand experience with this situation.)

Why backup? I

Computers were a mistake. But the bigger mistake was to give humans control over the computers.

Sometimes certain humans may write a program along the lines of:

```
with open("~/awesome", "w+") as f:
    f.writeline("Awesome program\n")
```

which doesn't do what you expect because by default Python doesn't expand `~` by default meaning this creates a directory named `~` in your working directory.

Naturally, to delete this directory, you would run `rm -rf ~`, right?

(I may or may not have first-hand experience with this situation.)

Why backup? I

Computers were a mistake. But the bigger mistake was to give humans control over the computers.

Sometimes certain humans may write a program along the lines of:

```
with open("~/awesome", "w+") as f:
    f.writeline("Awesome program\n")
```

which doesn't do what you expect because by default Python doesn't expand `~` by default meaning this creates a directory named `~` in your working directory.

Naturally, to delete this directory, you would run `rm -rf ~`, right?

(I may or may not have first-hand experience with this situation.)

Why backup? II

Ransomware protection.

If a malicious actor manages to encrypt a bunch of the files on your filesystem and demands money to get the key, you can just restore to a previous backup with minimal loss of productivity.

Why backup? II

Ransomware protection.

If a malicious actor manages to encrypt a bunch of the files on your filesystem and demands money to get the key, you can just restore to a previous backup with minimal loss of productivity.

Why not backup?

If you don't have any data that is important to you, and like setting up your computer over and over again, then you don't need a backup.

I'd bet that you have *something* that you want to backup.

Why not backup?

If you don't have any data that is important to you, and like setting up your computer over and over again, then you don't need a backup.

I'd bet that you have *something* that you want to backup.

Why not backup?

If you don't have any data that is important to you, and like setting up your computer over and over again, then you don't need a backup.

I'd bet that you have *something* that you want to backup.

Why not backup?

If you don't have any data that is important to you, and like setting up your computer over and over again, then you don't need a backup.

I'd bet that you have *something* that you want to backup.

Don't backup everything!

Backups can get bloated if you include too many unimportant files!

- A very small number of your dotfiles are actually useful to be backed up.
- Likewise only a few files in /etc actually matter.
- /var sometimes contains things that are worth backing up.
- On Windows, it's pointless to backup C:\Windows or Program Files.

Don't backup everything!

Backups can get bloated if you include too many unimportant files!

- A very small number of your dotfiles are actually useful to be backed up.
- Likewise only a few files in `/etc` actually matter.
- `/var` sometimes contains things that are worth backing up.
- On Windows, it's pointless to backup `C:\\Program Files`.

Don't backup everything!

Backups can get bloated if you include too many unimportant files!

- A very small number of your dotfiles are actually useful to be backed up.
- Likewise only a few files in `/etc` actually matter.
- `/var` sometimes contains things that are worth backing up.
- On Windows, it's pointless to backup `C:\\Program Files`.

Don't backup everything!

Backups can get bloated if you include too many unimportant files!

- A very small number of your dotfiles are actually useful to be backed up.
- Likewise only a few files in `/etc` actually matter.
- `/var` sometimes contains things that are worth backing up.
- On Windows, it's pointless to backup `C:\\Program Files`.

Don't backup everything!

Backups can get bloated if you include too many unimportant files!

- A very small number of your dotfiles are actually useful to be backed up.
- Likewise only a few files in `/etc` actually matter.
- `/var` sometimes contains things that are worth backing up.
- On Windows, it's pointless to backup `C:\\Program Files`.

The 3-2-1 rule

The **3-2-1 backup rule** states that you should:

Keep at least **three** copies of your data on at least **two** different storage media and store at least **one** of the copies off-site.

This may sound daunting, but keep in mind that **any backup is better than no backup!** You have to start somewhere.

The 3-2-1 rule

The **3-2-1 backup rule** states that you should:

Keep at least **three** copies of your data on at least **two** different storage media and store at least **one** of the copies off-site.

This may sound daunting, but keep in mind that **any backup is better than no backup!** You have to start somewhere.

The 3-2-1 rule

The **3-2-1 backup rule** states that you should:

Keep at least **three** copies of your data on at least **two** different storage media and store at least **one** of the copies off-site.

This may sound daunting, but keep in mind that **any backup is better than no backup!** You have to start somewhere.

The 3-2-1 rule

The **3-2-1 backup rule** states that you should:

Keep at least **three** copies of your data on at least **two** different storage media and store at least **one** of the copies off-site.

This may sound daunting, but keep in mind that **any backup is better than no backup!** You have to start somewhere.

The 3-2-1 rule

The **3-2-1 backup rule** states that you should:

Keep at least **three** copies of your data on at least **two** different storage media and store at least **one** of the copies off-site.

This may sound daunting, but keep in mind that **any backup is better than no backup!** You have to start somewhere.

File sync is not backup

File sync is not a backup strategy! It's an catastrophe distribution system.

File sync is not backup

File sync is not a backup strategy! It's an catastrophe distribution system.

A few other principles

The best backups are automatic, because otherwise you'll always default to "oh, I can do that later".

Tailor your backups to the data you are backing up. For example, don't just backup all of the files that your database uses, rather export your database periodically and backup that export.

Test your backups before you need them! You want to be confident in your backups. And if 2020 has taught us anything, it should be to expect the unexpected.

A few other principles

The best backups are automatic, because otherwise you'll always default to "oh, I can do that later".

Tailor your backups to the data you are backing up. For example, don't just backup all of the files that your database uses, rather export your database periodically and backup that export.

Test your backups before you need them! You want to be confident in your backups. And if 2020 has taught us anything, it should be to expect the unexpected.

A few other principles

The best backups are automatic, because otherwise you'll always default to "oh, I can do that later".

Tailor your backups to the data you are backing up. For example, don't just backup all of the files that your database uses, rather export your database periodically and backup that export.

Test your backups before you need them! You want to be confident in your backups. And if 2020 has taught us anything, it should be to expect the unexpected.

My Backup Strategies

A bit of history

- For a long time, I just copied my photos and other important documents to a hard drive. And I've done this on and off throughout the years.
- I have a fairly old copy of my photos from about 2012 on DVDs.
- I started by using CrashPlan. It allowed P2P backups between devices on the same LAN (as well as their servers).
- I then migrated to Dropbox as a "backup" strategy.
- I then migrated from Dropbox to self-hosted Nextcloud. But this time, I added a Duplicity backup for my VPS.
- Recently, I migrated to Syncthing for file sync and I'm using Restic to backup my VPS.

A bit of history

- For a long time, I just copied my photos and other important documents to a hard drive. And I've done this on and off throughout the years.
- I have a fairly old copy of my photos from about 2012 on DVDs.
- I started by using CrashPlan. It allowed P2P backups between devices on the same LAN (as well as their servers).
- I then migrated to Dropbox as a "backup" strategy.
- I then migrated from Dropbox to self-hosted Nextcloud. But this time, I added a Duplicity backup for my VPS.
- Recently, I migrated to Syncthing for file sync and I'm using Restic to backup my VPS.

A bit of history

- For a long time, I just copied my photos and other important documents to a hard drive. And I've done this on and off throughout the years.
- I have a fairly old copy of my photos from about 2012 on DVDs.
- I started by using CrashPlan. It allowed P2P backups between devices on the same LAN (as well as their servers).
- I then migrated to Dropbox as a "backup" strategy.
- I then migrated from Dropbox to self-hosted Nextcloud. But this time, I added a Duplicity backup for my VPS.
- Recently, I migrated to Syncthing for file sync and I'm using Restic to backup my VPS.

A bit of history

- For a long time, I just copied my photos and other important documents to a hard drive. And I've done this on and off throughout the years.
- I have a fairly old copy of my photos from about 2012 on DVDs.
- I started by using CrashPlan. It allowed P2P backups between devices on the same LAN (as well as their servers).
- I then migrated to Dropbox as a "backup" strategy.
- I then migrated from Dropbox to self-hosted Nextcloud. But this time, I added a Duplicity backup for my VPS.
- Recently, I migrated to Syncthing for file sync and I'm using Restic to backup my VPS.

A bit of history

- For a long time, I just copied my photos and other important documents to a hard drive. And I've done this on and off throughout the years.
- I have a fairly old copy of my photos from about 2012 on DVDs.
- I started by using CrashPlan. It allowed P2P backups between devices on the same LAN (as well as their servers).
- I then migrated to Dropbox as a "backup" strategy.
- I then migrated from Dropbox to self-hosted Nextcloud. But this time, I added a Duplicity backup for my VPS.
- Recently, I migrated to Syncthing for file sync and I'm using Restic to backup my VPS.

A bit of history

- For a long time, I just copied my photos and other important documents to a hard drive. And I've done this on and off throughout the years.
- I have a fairly old copy of my photos from about 2012 on DVDs.
- I started by using CrashPlan. It allowed P2P backups between devices on the same LAN (as well as their servers).
- I then migrated to Dropbox as a "backup" strategy.
- I then migrated from Dropbox to self-hosted Nextcloud. But this time, I added a Duplicity backup for my VPS.
- Recently, I migrated to Syncthing for file sync and I'm using Restic to backup my VPS.

What I backup

- Documents
- Photos
- Projects
- Dotfiles
- System configs
- The data for programs that I self-host

Backup method 1: Git

I use Git to backup a lot of things. It's really good when all you have is text.

- All of my projects (besides some really old ones) are in Git repos. I use either GitHub, GitLab, or Sourcehut to host all of my repos.
- All of the dotfiles that I care about are stored in a Git repo, managed using Chezmoi.
- I'm currently in the process of migrating to NixOS on all of my machines, and everything is *declarative* and *immutable*. I don't have to backup most of my configs because they're already procedurally generated, and I literally can't make changes without updating my NixOS config.

Backup method 1: Git

I use Git to backup a lot of things. It's really good when all you have is text.

- All of my projects (besides some really old ones) are in Git repos. I use either GitHub, GitLab, or Sourcehut to host all of my repos.
- All of the dotfiles that I care about are stored in a Git repo, managed using Chezmoi.
- I'm currently in the process of migrating to NixOS on all of my machines, and everything is *declarative* and *immutable*. I don't have to backup most of my configs because they're already procedurally generated, and I literally can't make changes without updating my NixOS configs.

Backup method 1: Git

I use Git to backup a lot of things. It's really good when all you have is text.

- All of my projects (besides some really old ones) are in Git repos. I use either GitHub, GitLab, or Sourcehut to host all of my repos.
- All of the dotfiles that I care about are stored in a Git repo, managed using Chezmoi.
- I'm currently in the process of migrating to NixOS on all of my machines, and everything is *declarative* and *immutable*. I don't have to backup most of my configs because they are already procedurally generated, and I literally *can't* make changes without updating my NixOS config.

Backup method 1: Git

I use Git to backup a lot of things. It's really good when all you have is text.

- All of my projects (besides some really old ones) are in Git repos. I use either GitHub, GitLab, or Sourcehut to host all of my repos.
- All of the dotfiles that I care about are stored in a Git repo, managed using Chezmoi.
- I'm currently in the process of migrating to NixOS on all of my machines, and everything is *declarative* and *immutable*. I don't have to backup most of my configs because they are already procedurally generated, and I literally *can't* make changes without updating my NixOS config.

Backup method 1: Git

I use Git to backup a lot of things. It's really good when all you have is text.

- All of my projects (besides some really old ones) are in Git repos. I use either GitHub, GitLab, or Sourcehut to host all of my repos.
- All of the dotfiles that I care about are stored in a Git repo, managed using Chezmoi.
- I'm currently in the process of migrating to NixOS on all of my machines, and everything is *declarative* and *immutable*. I don't have to backup most of my configs because they are already procedurally generated, and I literally *can't* make changes without updating my NixOS config.

Backup method 2: Restic

I use Restic to backup everything else. I only backup my Linode VPS, though.

All data gets synced to my VPS, and then from there it gets backed up to BackBlaze B2 using Restic.

Restic encrypts the backups by default. It stores data in a structure very similar to Git.

Backup method 2: Restic

I use Restic to backup everything else. I only backup my Linode VPS, though.

All data gets synced to my VPS, and then from there it gets backed up to BackBlaze B2 using Restic.

Restic encrypts the backups by default. It stores data in a structure very similar to Git.

Backup method 2: Restic

I use Restic to backup everything else. I only backup my Linode VPS, though.

All data gets synced to my VPS, and then from there it gets backed up to BackBlaze B2 using Restic.

Restic encrypts the backups by default. It stores data in a structure very similar to Git.

Future

I want to also figure out a good way to have an on-site non-synchronized Restic backup.

Conclusion

This may be daunting

My backup strategy is kinda complicated. Don't let that deter you!

I've built this up over many years and it's almost become an obsession of mine. And given that the apocalypse is upon us, maybe it's not that bad of an obsession to have.

Start somewhere, even if it's just copying important files to a hard drive, and go from there.

This may be daunting

My backup strategy is kinda complicated. Don't let that deter you!

I've built this up over many years and it's almost become an obsession of mine. And given that the apocalypse is upon us, maybe it's not that bad of an obsession to have.

Start somewhere, even if it's just copying important files to a hard drive, and go from there.

This may be daunting

My backup strategy is kinda complicated. Don't let that deter you!

I've built this up over many years and it's almost become an obsession of mine. And given that the apocalypse is upon us, maybe it's not that bad of an obsession to have.

Start somewhere, even if it's just copying important files to a hard drive, and go from there.

This may be daunting

My backup strategy is kinda complicated. Don't let that deter you!

I've built this up over many years and it's almost become an obsession of mine. And given that the apocalypse is upon us, maybe it's not that bad of an obsession to have.

Start somewhere, even if it's just copying important files to a hard drive, and go from there.

Start by deciding on an “end-state”

Create a backup strategy “ideal state”. Hopefully it integrates the 3-2-1 principle and is automated.

Then, figure out a way to get a quick win that fits into your overall strategy.

Then build more and more infrastructure around it.

Start by deciding on an “end-state”

Create a backup strategy “ideal state”. Hopefully it integrates the 3-2-1 principle and is automated.

Then, figure out a way to get a quick win that fits into your overall strategy.

Then build more and more infrastructure around it.

Start by deciding on an “end-state”

Create a backup strategy “ideal state”. Hopefully it integrates the 3-2-1 principle and is automated.

Then, figure out a way to get a quick win that fits into your overall strategy.

Then build more and more infrastructure around it.

Start by deciding on an “end-state”

Create a backup strategy “ideal state”. Hopefully it integrates the 3-2-1 principle and is automated.

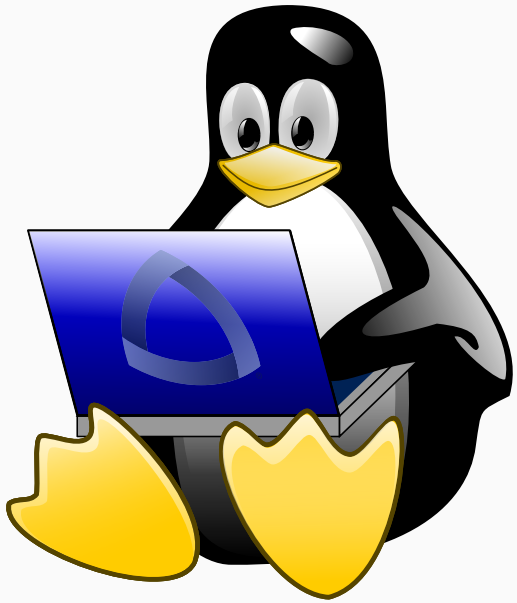
Then, figure out a way to get a quick win that fits into your overall strategy.

Then build more and more infrastructure around it.

Copyright Notice

This presentation was from the **Mines Linux Users Group**. A mostly-complete archive of our presentations can be found online at `https://lug.mines.edu`.

Individual authors may have certain copyright or licensing restrictions on their presentations. Please be certain to contact the original author to obtain permission to reuse or distribute these slides.



Colorado School of Mines
Linux Users Group