pass Unix password manager

START HERE

`gpg --list-secret-keys`
Find your key ID

`pass git init`
Track with git

`pass init <gpg-id>`
Create the store

`pass git push`
Sync to a remote

`pass insert <name>`
Save a password

`pass generate <name> 20`
Make a random one

`pass -c <name>`
Copy to clipboard

`pass`
List the tree

Plain text typed or generated → `pass insert` → **GPG file** name.gpg in the store → `pass git push` → **Git remote** optional backup

CONCEPTS

Each password is one GPG-encrypted file under ~/.password-store; the folder tree is your organisation.

The first line of a file is the password; put extra data (user, URL) on later lines.

Back up your GPG secret key: without it the store cannot be decrypted.

SETUP

`pass init <gpg-id>`
Create the store for a GPG key
Re-encrypts existing entries if the key changes. Pass several IDs to share the store.

`pass init -p <subfolder> <gpg-id>`
Use a different key for a folder
Writes a .gpg-id in that folder.

`pass git init`
Turn the store into a git repo
Commits existing contents; later changes are committed automatically.

`pass git remote add origin <url>`
Add a remote to the store

`PASSWORD_STORE_DIR=<dir> pass`
Use another store location
Default is ~/.password-store.

SETUP (cont.)

`PASSWORD_STORE_ENABLE_EXTENSIONS=true`
Enable user extensions

BROWSE

`pass`
List all entries as a tree
Same as pass ls; needs the tree tool.

`pass ls <folder>`
List one folder

`pass find <name>`
List entries whose name matches

`pass grep <text>`
Search inside decrypted entries
Decrypts every file; extra options go to grep, e.g. -i.

`pass grep -i <text>`
Search ignoring case

READ

`pass show <name>`
Print an entry

`pass <name>`
Print an entry (shortcut)

`pass -c <name>`
Copy the first line to the clipboard
Clipboard is cleared after 45 seconds.

`pass -c<N> <name>`
Copy line N instead

READ (cont.)

`pass -q <name>`
Show as a QR code
Needs qrencode.

`PASSWORD_STORE_CLIP_TIME=<sec>`
Change the clipboard timeout

`PASSWORD_STORE_X_SELECTION=primary`
Use the primary selection for -c

ADD AND EDIT

`pass insert <name>`
Add an entry, typed twice hidden

`pass insert -m <name>`
Add a multi-line entry
Finish with Ctrl+D.

`pass insert -e <name>`
Add an entry with visible input

`pass insert -f <name>`
Overwrite without asking

`pass edit <name>`
Edit an entry in $EDITOR
Temp files go to /dev/shm when available.

`pass generate <name> <length>`
Generate a random password

`pass generate -n <name> <length>`
Generate without symbols

ADD AND EDIT (cont.)

`pass generate -c <name> <length>`
Generate and copy, do not print

`pass generate -i <name> <length>`
Replace only the first line
Keeps the other lines of the entry.

`pass generate -f <name> <length>`
Overwrite without asking

`PASSWORD_STORE_GENERATED_LENGTH=<n>`
Default generated length

`PASSWORD_STORE_CHARACTER_SET=<set>`
Characters allowed in generate
Interpreted by tr.

ORGANISE

`pass mv <old> <new>`
Rename or move an entry
Re-encrypts for the destination's keys.

`pass cp <old> <new>`
Copy an entry

`pass rm <name>`
Delete an entry
Asks for confirmation.

`pass rm -r <folder>`
Delete a folder

`pass rm -f <name>`
Delete without asking

ORGANISE (cont.)

`pass mv -f <old> <new>`
Move, overwriting silently

`pass mv <name> <folder>/`
Move into a folder
A trailing slash forces folder treatment.

GIT SYNC

`pass git log`
Show store history

`pass git push`
Push the store

`pass git pull`
Pull changes from the remote

`pass git status`
Check for uncommitted changes

`pass git clone <url> ~/.password-store`
Restore on a new machine
Import your GPG secret key first.

`pass git config pass.signcommits true`
Sign store commits

TEAMS AND SECURITY

`pass init <id1> <id2>`
Encrypt for several recipients

`PASSWORD_STORE_GPG_OPTS=<opts>`
Extra options for every gpg call

TEAMS AND SECURITY (cont.)

`PASSWORD_STORE_SIGNING_KEY=<fingerprint>`
Require signed .gpg-id files
Takes full 40-character fingerprints.

Extensions such as pass-otp live in .extensions/ or system-wide and add commands like pass otp.

Entry names are visible in the file tree and git history; only the contents are encrypted.

RECIPES

First setup

```
gpg --list-secret-keys --keyid-format long
pass init <gpg-id>
pass git init
pass generate email/work 24
```

Sync to a new machine

```
gpg --import <secret-key-file>
pass git clone <url> ~/.password-store
pass -c email/work
```

Store extra fields

```
pass insert -m site/example
# line 1: password
# line 2: user: me@example.com
```

Rotate a password

```
pass generate -i site/example 24
pass git push
```