1Password CLI

Secrets from the shell

Reference op://vault/item/field → `op run` / `inject` → **1Password CLI** signed-in account → resolves → **Secret value** env var or file

CONCEPTS

Keep secrets out of files: store op:// references instead, and let op run or op inject resolve them at the last moment.

With the desktop app integration on (Settings > Developer), op unlocks via Touch ID or system auth and needs no manual signin.

Use --account <shorthand> or OP_ACCOUNT when you have several accounts; use --format json to script output.

Command arguments can be seen by other processes. Pass sensitive values via --template or stdin, not as assignment arguments.

SIGN IN

`op signin`
Sign in (prompts only if needed)

`eval $(op signin --account <shorthand>)`
Sign in, set session in this shell

SIGN IN (cont.)

`op whoami`
Show the signed-in account

`op signout`
Sign out

`op account list`
List configured accounts

`op account add --address <host>`
Add an account

VAULTS

`op vault list`
List readable vaults

`op vault get <vault>`
Show a vault

`op vault create <name>`
Create a vault

ITEMS

`op item list`
List items

`op item list --vault <vault>`
List items in one vault

`op item list --categories Login`
Filter by category

`op item list --tags <tag>`
Filter by tag

ITEMS (cont.)

`op item get <item>`
Show item details

`op item get <item> --fields label=password`
Print one field

`op item get <item> --otp`
Current one-time password

`op item get <item> --format json`
Full item as JSON

`op item create --category login --title <title>`
Create a login item

`op item create ... --generate-password=20,letters,digits`
Create with a generated password

`op item create ... <field>=<value>`
Set a field on create

`op item create --template <file.json>`
Create from a JSON template

`op item edit <item> <field>=<value>`
Change a field

`op item edit <item> --generate-password`
Regenerate the password

ITEMS (cont.)

`op item delete <item>`
Delete an item

`op item delete <item> --archive`
Archive instead of delete

`op item move <item> --current-vault <a> --destination-vault <b>`
Move to another vault

SECRET REFERENCES

`op read op://<vault>/<item>/<field>`
Print a secret

`op read op://<vault>/<item>/<section>/<field>`
Read a field in a section

`op read "op://<vault>/<item>/one-time password?attribute=otp"`
Read the current OTP

`op read "op://<vault>/<item>/private key?ssh-format=openssh"`
SSH key in OpenSSH format

`op read -o <file> op://<vault>/<item>/<field>`
Write a secret to a file

References use names or IDs. Quote them if a name contains spaces.

RUN AND INJECT

`op run -- <cmd>`
Run with env var references resolved

`op run --env-file=<file> -- <cmd>`
Resolve references from a .env file

`op run --no-masking -- <cmd>`
Show secrets in command output

`op inject -i <tpl> -o <out>`
Fill a template into a file

`cat <tpl> | op inject`
Fill a template from stdin

Templates use {{ op://vault/item/field }}. Env vars inside references, like op://$APP_ENV/db/password, switch vaults per environment.

DOCUMENTS

`op document list`
List documents

`op document create <file> --title <title>`
Upload a file

`op document get <doc> -o <file>`
Download a document