sudo Run as another user

START HERE

`sudo <command>`
Run as root

`sudo -e <file>`
Edit a root-owned file

`sudo -u <user> <command>`
Run as another user

`sudo -v`
Refresh the password cache

`sudo -i`
Root login shell

`sudo -k`
Forget cached password

`sudo -l`
What may I run?

`sudo visudo`
Edit the rules safely

You run sudo command + options → `checks rules` → **/etc/sudoers** who, where, as whom, what → `allowed?` → **Password** cached for a few minutes → `runs` → **Command** as target user

CONCEPTS

sudo runs one command with another identity (root by default) if the sudoers rules allow it, and asks for your own password, not root's.

Redirections happen in your shell before sudo starts, so sudo echo x > /etc/f still writes as you.

RUNNING COMMANDS

`sudo <command>`
Run a command as root

`sudo -u <user> <command>`
Run as another user

`sudo -g <group> <command>`
Run with another primary group

`sudo -u <user> -g <group> <command>`
Set user and group

`sudo -u '#<uid>' <command>`
Target a user by numeric UID

`sudo -n <command>`
Fail instead of asking for a password

`sudo -b <command>`
Run in the background

`sudo -T <time> <command>`
Kill the command after a timeout
Needs a sudoers rule that allows user-set timeouts.

`sudo -R <dir> <command>`
Run with another root directory

`sudo -- <command> -x`
Stop sudo parsing options

`sudo !!`
Repeat the last command with sudo
Bash and zsh history expansion, not sudo itself.

SHELLS

`sudo -i`
Root login shell with root's environment

`sudo -s`
Root shell keeping your environment

`sudo -iu <user>`
Login shell as another user

`sudo -i <command>`
Run a command through the login shell

`sudo -H <command>`
Set HOME to the target user's home

SHELLS (cont.)

Prefer one-off sudo commands over a root shell: every command stays logged and a typo does less damage.

PRIVILEGES AND PASSWORD CACHE

`sudo -l`
List what you may run here

`sudo -ll`
List in the long format

`sudo -l <command>`
Check if a command is allowed

`sudo -l -U <user>`
List another user's privileges

`sudo -v`
Refresh the cache without running anything

`sudo -k`
Invalidate the cache for this session

`sudo -K`
Remove all cached credentials

`sudo -k <command>`
Run a command and force a password prompt

`sudo -V`
Show sudo and plugin versions

`sudo -h`
Short help

`sudo -S`
Read the password from stdin

`sudo -A`
Ask for the password via a helper program

ENVIRONMENT

`sudo -E <command>`
Keep your environment variables
The policy may refuse; PATH and similar are often reset anyway.

`sudo <VAR>=<value> <command>`
Set a variable for one command
Only variables the policy allows.

`sudo env <VAR>=<value> <command>`
Set a variable via env

`sudo env | grep -i path`
See the environment sudo passes on

PATH is replaced by secure_path on many systems, so tools in your own PATH may not be found under sudo.

EDITING FILES

`sudo -e <file>`
Edit a copy and write it back as root

`sudoedit <file>`
Same as sudo -e

`sudoedit <file1> <file2>`
Edit several files in one session

`SUDO_EDITOR=<editor> sudoedit <file>`
Pick the editor for sudoedit
Order: SUDO_EDITOR, VISUAL, EDITOR.

sudoedit runs your editor as you, not root, so editor plugins and config are safe to use.

REDIRECTION AND PIPES

`echo '<text>' | sudo tee <file>`
Write a root-owned file

`echo '<text>' | sudo tee -a <file>`
Append to a root-owned file

`echo '<text>' | sudo tee <file> >/dev/null`
Write without echoing to the screen

`sudo sh -c '<command> > <file>'`
Redirect inside a root shell

`sudo bash -c '<command1> && <command2>'`
Run a compound command as root

`<command> | sudo -u <user> <command>`
Pipe into a command run as another user

`sudo cat <file> | <command>`
Read a root-only file into a pipe

SUDOERS RULES

`sudo visudo`
Edit /etc/sudoers with a syntax check

`sudo visudo -f /etc/sudoers.d/<name>`
Edit a drop-in file

`sudo visudo -c`
Check the syntax only

`sudo visudo -cf <file>`
Check one file

SUDOERS RULES (cont.)

Rule: who where = (as whom) what

Part	Example	Meaning
`user or %group`	`alice, %admin`	who may run it
`host`	`ALL`	on which machine
`(runas)`	`(root), (ALL)`	as which user
`tags: commands`	`NOPASSWD: usr/bin/apt`	what, and options

`alice ALL=(ALL:ALL) ALL`
Full rights for a user

`%wheel ALL=(ALL) ALL`
Full rights for a group

`alice ALL=(root) NOPASSWD: /usr/bin/systemctl restart <svc>`
One command without a password

`alice ALL=(root) /usr/bin/apt update, !/usr/bin/apt remove *`
Allow some, deny others

`Cmnd_Alias <NAME> = <cmd1>, <cmd2>`
Name a group of commands

`Defaults timestamp_timeout=<minutes>`
Change the password cache time

Never edit /etc/sudoers with a plain editor: one syntax error can lock out every sudo user. Keep a root shell open while testing.

Later rules win, so put specific exceptions after the general ones.

RECIPES

Edit a system file as yourself

```
sudo -e /etc/hosts
# or choose the editor
SUDO_EDITOR=vim sudoedit /etc/fstab
```

Append a line to a root-only file

```
echo 'vm.swappiness=10' | sudo tee -a /etc/sysctl.d/99-local.conf
sudo sysctl --system
```

Let a user restart one service

```
sudo visudo -f /etc/sudoers.d/deploy
# add:
deploy ALL=(root) NOPASSWD: /usr/bin/systemctl restart myapp
sudo visudo -c
```

Audit what a user may do

```
sudo -l -U alice
sudo -ll
sudo -l /usr/bin/apt
```