sudo Cheatsheet

By Dejan Panovski • Updated on Sep 20, 2026

sudo at a glance: running commands as root, credential caching, environment handling, sudoers rule syntax, aliases, Defaults, and logging.

The `sudo` command runs a single command with another user's privileges, usually root, and the `sudoers` file decides who may run what. This cheatsheet covers the command options you use daily, plus the rule syntax, `Defaults` settings, and log locations you need when you configure `sudo` on a server.

Basic Usage

Run commands with elevated privileges.

`sudo command`	Run a command as root
`sudo -u username command`	Run a command as another user
`sudo -g groupname command`	Run a command with another primary group
`sudo -u#1000 command`	Run as a user ID instead of a name
`sudo -i`	Open a root login shell that reads root's profile
`sudo -s`	Open a root shell that keeps the current directory
`sudo -e /etc/hosts`	Edit a root-owned file with an editor running as your user
`sudo -b command`	Run the command in the background
`sudo -- command -l`	Stop option parsing so the flags reach the command
`sudo !!`	Re-run the previous command with `sudo` in Bash or Zsh

Password Caching

Control how often `sudo` asks for a password.

`sudo -v`	Refresh the cached credentials without running a command
`sudo -k`	Invalidate the cache for the current terminal
`sudo -K`	Remove the cached credentials for every terminal
`sudo -n command`	Fail with an error instead of prompting, for cron and scripts
`sudo -A command`	Read the password from the program named in `SUDO_ASKPASS`
`sudo -p "Password for %u: " command`	Set a custom prompt
`Defaults timestamp_timeout=10`	Cache the password for 10 minutes
`Defaults timestamp_timeout=0`	Prompt for every command
`Defaults timestamp_type=global`	Share one timestamp across all terminals

Debian and Ubuntu packages ship a 15 minute timeout, while Fedora and RHEL keep the upstream five minute default.

Listing Privileges

Check what a rule actually grants before you rely on it.

`sudo -l`	List the commands the current user may run
`sudo -ll`	List the same rules in long form
`sudo -l -U username`	List another user's privileges as root or an authorized user
`sudo -l /usr/bin/systemctl`	Check whether one command is permitted
`sudo -V`	Print the version, and as root the full `Defaults` list
`id -nG`	List group membership, which `%group` rules match on
`getent group sudo`	Show the members of the `sudo` group

Environment Handling

`sudo` builds a new environment rather than passing yours through.

`sudo -E command`	Request preservation of the current environment
`sudo --preserve-env=http_proxy command`	Request preservation of only the named variables
`sudo -H command`	Set `HOME` to the target user's home directory
`sudo env`	Print the environment `sudo` actually builds
`sudo env PATH="/opt/tool/bin:/usr/bin" command`	Run with an explicit trusted `PATH` instead of `secure_path`
`Defaults env_reset`	Strip the caller's environment, the default on most distros
`Defaults env_keep += "http_proxy https_proxy"`	Let named variables through `env_reset`
`Defaults secure_path="/usr/sbin:/usr/bin:/sbin:/bin"`	Replace `PATH` for every `sudo` command
`Defaults always_set_home`	Set target `HOME` when `env_reset` does not already do so

`sudo` exports `SUDO_USER`, `SUDO_UID`, `SUDO_GID`, and `SUDO_COMMAND` into the command environment, so a privileged script can recover who called it.

Redirects and Pipes

The shell expands `>` and `|` before `sudo` runs, so redirects need their own root process.

`sudo echo text > /root/file`	Usually fails: the shell opens the file as your user
`sudo sh -c 'echo text > /root/file'`	Run the whole line, redirect included, as root
`echo text | sudo tee /root/file`	Write the file as root
`echo text | sudo tee -a /root/file`	Append to the file as root
`echo text | sudo tee /root/file > /dev/null`	Write without echoing the content back
`sudo sh -c 'cmd1 && cmd2'`	Chain several commands under one `sudo` call
`sudo cat /root/file | grep pattern`	Read as root, filter as your user

Editing the sudoers File

Never open `/etc/sudoers` in a plain editor. A syntax error there can disable `sudo` access.

`sudo visudo`	Edit `/etc/sudoers` with a syntax check on save
`sudo visudo -c`	Check every sudoers file without editing
`sudo visudo -f /etc/sudoers.d/webops`	Edit a drop-in file with the same check
`sudo visudo -s`	Treat undefined aliases and alias cycles as errors
`@includedir /etc/sudoers.d`	Parse eligible files in the drop-in directory
`sudo chown root:root /etc/sudoers.d/webops`	Set the required owner and group
`sudo chmod 0440 /etc/sudoers.d/webops`	Set the permissions a drop-in file requires
`sudo ls /etc/sudoers.d`	List the files in the drop-in directory

On `sudo` older than 1.9.1 the directive is written `#includedir /etc/sudoers.d`. Despite the leading `#`, that line is not a comment. Drop-in files whose names contain a dot or end in `~` are skipped.

`visudo` reads `SUDO_EDITOR`, `VISUAL`, and `EDITOR` when the policy lets those variables through. A command rule that matches `ALL` implies `SETENV`, while a rule granting only `/usr/sbin/visudo` must add `SETENV` or preserve the editor variables. To avoid depending on that policy, open a root shell with `sudo -i`, then run `EDITOR=/usr/bin/vim visudo`.

sudoers Rule Syntax

Each rule reads `user host=(runas:rungroup) tagged commands`.

`username ALL=(ALL:ALL) ALL`	Full access for one user
`%sudo ALL=(ALL:ALL) ALL`	Full access for a group, Debian and Ubuntu
`%wheel ALL=(ALL) ALL`	Full access for a group, Fedora and RHEL
`username ALL=(ALL) NOPASSWD: ALL`	Skip the password prompt for every command
`username ALL=(ALL) NOPASSWD: /usr/bin/systemctl restart nginx`	Skip the prompt for one exact command
`username ALL=(www-data) /usr/bin/php`	Run one command as a service account
`username ALL=(ALL) NOEXEC: /usr/bin/less`	Block child commands where `NOEXEC` is supported
`username ALL=(ALL) PASSWD: /usr/bin/su`	Force a prompt for one command inside a `NOPASSWD` rule

Command paths generally must be absolute. A rule listing `apt` never matches; write `/usr/bin/apt`. Wildcards in command arguments can match whitespace, so exact arguments or anchored regular expressions are safer. Do not use an editor as a security boundary: `/usr/bin/vim` can start a shell even without a wildcard.

Aliases

Aliases keep long rule sets readable and reduce repetition.

`User_Alias ADMINS = alice, bob, %ops`	Name a set of users or groups
`Runas_Alias SUPERUSER = root`	Name a set of target users
`Host_Alias WEB = web01, web02, 10.0.5.0/24`	Name a set of hosts
`Cmnd_Alias NGINX = /usr/bin/systemctl restart nginx, /usr/bin/systemctl reload nginx`	Name a set of exact commands
`ADMINS WEB=(SUPERUSER) NGINX`	Combine aliases in a rule
`ADMINS ALL=(root) NOPASSWD: NGINX`	Apply a tag to a command alias

Alias names start with an uppercase letter and contain uppercase letters, digits, or underscores. The parser resolves aliases across the complete policy, so a definition may appear before or after a rule that uses it.

Defaults Directives

Policy settings that apply to every matching `sudo` call.

`Defaults env_reset`	Run commands with a clean environment
`Defaults secure_path="/usr/sbin:/usr/bin:/sbin:/bin"`	Fixed `PATH` for `sudo` commands
`Defaults timestamp_timeout=15`	Minutes before the password is asked for again
`Defaults passwd_tries=3`	Password attempts before `sudo` gives up
`Defaults targetpw`	Ask for the target user's password, not the caller's
`Defaults lecture=never`	Skip the warning shown on first use
`Defaults mail_badpass`	Mail the administrator after a failed password
`Defaults:username timestamp_timeout=30`	Apply a setting to one user
`Defaults@web01 log_output`	Apply a setting on one host
`Defaults!/usr/sbin/visudo env_keep += "SUDO_EDITOR VISUAL EDITOR"`	Preserve editor variables for `visudo`

Logging and Auditing

By default, `sudoers` logs allowed and denied commands as well as errors.

`sudo journalctl _COMM=sudo`	Show `sudo` events from the systemd journal
`sudo journalctl -t sudo -S today`	Show today's events by syslog tag
`sudo grep sudo /var/log/auth.log`	Read the log on Debian and Ubuntu
`sudo grep sudo /var/log/secure`	Read the log on Fedora, RHEL, and derivatives
`Defaults logfile="/var/log/sudo.log"`	Write a dedicated `sudo` log file
`Defaults log_input, log_output`	Record full sessions for replay
`sudo sudoreplay -l`	List the recorded sessions
`sudo sudoreplay ID`	Replay one recorded session

Troubleshooting

What the common failures mean and where to look.

`username is not in the sudoers file`	Add the user to `sudo` on Debian and Ubuntu or `wheel` on Fedora and RHEL, then log in again
`sudo: unable to resolve host name`	Map the current hostname in `/etc/hosts`, such as `127.0.1.1 name`
`sudo: command not found` or `sudo: name: command not found`	The shell cannot find `sudo`, or the target command is missing or outside `secure_path`
`sorry, you are not allowed to set the following environment variables`	Preserve the variable or grant `SETENV` narrowly
`sorry, you are not allowed to preserve the environment`	Grant `SETENV`; matching `ALL` implies it unless `NOSETENV` overrides it
`>>> /etc/sudoers: syntax error near line N <<<`	Press `e` to re-edit; `Q` force-saves the broken file
`sudo: no tty present and no askpass program specified`	Use `-n` to fail fast, configure `-A`, or grant `NOPASSWD` narrowly
`sudo: effective uid is not 0`	Check root ownership, setuid, `nosuid`, and NFS mounts; repair from root or recovery

Related Guides

Deeper reading on privilege escalation and user administration.

sudo Command in Linux	Full `sudo` guide with examples, sudoedit, and credential caching
How to Add User to Sudoers in Ubuntu	Grant sudo access on Debian and Ubuntu
How to Add User to Sudoers in CentOS	Grant sudo access on RHEL and derivatives
How to Run sudo Command Without Password	Set up a `NOPASSWD` rule safely
su Command in Linux	Switch user accounts instead of running one command
su cheatsheet	Quick reference for `su`