

Install & Subscription

Installing RHEL and Rocky Linux, subscriptions and repositories, plus general command-line tips.

- [Linux - RHEL Subscription](#)
- [Linux Server - Install RedHat 9 SSH configuration](#)
- [Linux - Commands to Know](#)
- [Linux - Bag of Tricks](#)
- [Linux - Bag of Tricks \(Part 2\): Certificates, Databases, Disks & Boot](#)
- [Command - Cmnd_Alias](#)

Linux - RHEL Subscription

Last verified: October 2026 against Red Hat's registration and Simple Content Access documentation and the RHEL 10.0 release notes.

We encountered the error message *'This system is not registered with an entitlement server, You can use "rhc" or "subscription-manager" to register'*. On CentOS Stream 9 Linux system while trying to perform package installations. For RHEL, this usually is an indication that your system is not registered with Red Hat's subscription management service.

```
$ sudo dnf clean all
```

```
Updating Subscription Management repositories.
```

```
Unable to read consumer identity
```

```
This system is not registered with an entitlement server. You can use "rhc" or "subscription-manager" to register.
```

```
21 files removed
```

Here is how we solved the issue on the system. Open the following file for editing:

```
sudo vi /etc/dnf/plugins/subscription-manager.conf
```

Change from `enabled=1` to `enabled=0`:

```
[main]
```

```
enabled=0
```

```
# When following option is set to 1, then all repositories defined outside redhat.repo will be disabled
```

```
# every time subscription-manager plugin is triggered by dnf or yum
```

```
disable_system_repos=0
```

You can then update package cache and try install your packages.

For RHEL system, run the following command to register your system with the Red Hat Subscription Manager:

```
sudo subscription-manager register --username <your_username> --password <your_password>
```

Remember to replace the username and password with the ones for your Red Hat account. On RHEL 8.8 and later you can also register with `rhc`, using either an activation key or your account:

```
sudo rhc connect --activation-key=<key> --organization=<org_id>
sudo rhc connect --username=<your_username> --password=<your_password>
```

No attach step is needed after registering. Simple Content Access is now the default for all Red Hat accounts, so a registered system can use the content its account is entitled to straight away. `subscription-manager attach --auto` from older guides was deprecated in RHEL 9.5 and the attach commands were removed in RHEL 10.

Check that registration was successful:

```
sudo subscription-manager status
```

To enable a specific repository, run:

```
sudo subscription-manager repos --enable=<repository_name>
```

Listing of available repositories can be done using:

```
sudo subscription-manager repos --list
```

Enjoy using your CentOS Stream or Red Hat Enterprise Linux system!.

Sources

- https://docs.redhat.com/en/documentation/subscription_central/1-latest/html-single/getting_started_with_rhel_system_registration/index
- <https://access.redhat.com/articles/simple-content-access>
- https://access.redhat.com/articles/transition_of_subscription_services_to_the_hybrid_cloud_console
- https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/10/html/10.0_release_notes/removed-features

Linux Server - Install RedHat 9 SSH configuration

Last verified: October 2026 against Red Hat's RHEL 9 system-wide cryptographic policies guide and the sshd_config manual.

On Red Hat Enterprise Linux 9 (RHEL 9 / Rocky Linux 9 / AlmaLinux 9 etc.), SSH configuration is primarily managed through system-wide crypto policies. Direct settings in /etc/ssh/sshd_config for KexAlgorithms, Ciphers, etc., are often ignored unless you properly override the policy.

1. Recommended: Add Curve25519 (preferred) and Diffie-Hellman group exchange

Curve25519-sha256 (and its @libssh.org variant) is already enabled by default in RHEL 9 and is the modern, secure choice.

diffie-hellman-group-exchange-sha256 is also generally available and reasonably secure.

Create a drop-in config. Red Hat's guidance is to use a two-digit prefix smaller than 50 so the file is read before 50-redhat.conf (which pulls in the crypto policy). sshd uses the **first** value it reads for each keyword:

```
sudo mkdir -p /etc/ssh/sshd_config.d
sudo tee /etc/ssh/sshd_config.d/49-custom-kex.conf << EOF
# Only offer these KEX algorithms (the client chooses the order)
KexAlgorithms curve25519-sha256,curve25519-sha256@libssh.org,diffie-hellman-group-exchange-sha256,diffie-
hellman-group14-sha256,diffie-hellman-group16-sha512
EOF
```

2. Test the config

```
sudo sshd -t
```

If no errors, reload SSH:

```
sudo systemctl reload sshd
```

3. Temporarily allow legacy algorithms (if needed for old clients/servers)

Legacy algorithms (e.g., SHA-1 based ones like `diffie-hellman-group1-sha1` or older group-exchange) are disabled for security.

Best temporary option (least impact):

```
# Append legacy KEX to OpenSSH's built-in default list
sudo tee /etc/ssh/sshd_config.d/48-legacy-kex.conf << EOF
KexAlgorithms +diffie-hellman-group-exchange-sha1,diffie-hellman-group1-sha1
EOF
```

Because `48-legacy-kex.conf` is read first, its `KexAlgorithms` line is used instead of the one in `49-custom-kex.conf` while it exists. Test with `sudo sshd -t`, reload `sshd`, and delete the file when the old clients are gone.

Or use the broader (but riskier) **LEGACY** crypto policy:

```
sudo update-crypto-policies --set LEGACY
sudo reboot
```

Warning: LEGACY weakens many other things system-wide (not just SSH). Revert with `sudo update-crypto-policies --set DEFAULT` and reboot when done. Red Hat recommends a reboot after changing the policy so every service picks it up.

Verification commands

```
# Available KEX algorithms
ssh -Q kex

# Effective configuration used by sshd
sudo sshd -T | grep -E '^(kexalgorithms|ciphers|macs)'
```

```
# Test connection  
ssh -v user@yourserver
```

Notes for your homelab / self-hosted setup

- Prefer **drop-in files** under `/etc/ssh/sshd_config.d/` (numbered `< 50`, for example `49-*.conf`, so they are read before Red Hat's `50-redhat.conf`; the first value read wins).
- After any change: `sshd -t` → `systemctl reload sshd`.
- If you have specific old clients (e.g., very old Windows, network devices, or RHEL 6-era), the `+` syntax to append is safest.

Sources

- https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/security_harden/using-the-system-wide-cryptographic-policies_security-hardening
- https://man.openbsd.org/sshd_config

Linux - Commands to Know

Last verified: October 2026 against the current man pages (tar, touch, useradd, userdel, groupadd, groupdel) for RHEL / Rocky Linux.

This is a quick version of commands to be aware of.

System information

`uname -a` : Displays all system information.

`hostnamectl` : Shows current hostname and related details.

`lscpu` : Lists CPU architecture information.

`timedatectl status` : Shows system time.

System monitoring and management

`top` : Displays real-time system processes.

`htop` : An interactive process viewer (needs installation; on RHEL / Rocky Linux it comes from the EPEL repository).

`df -h` : Shows disk usage in a human-readable format.

`free -m` : Displays free and used memory in MB.

`kill [PID]` : Terminates a process (sends SIGTERM by default).

Running commands

`<command> &` : Runs command in the background.

`jobs` : Displays background commands.

`fg %1` : Brings background job 1 (number from jobs) to the foreground.

Service management

`sudo systemctl start [service]` : Starts a service.

`sudo systemctl stop [service]` : Stops a service

`sudo systemctl status [service]` : Checks the status of a service.

`sudo systemctl reload [service]` : Reloads a service's configuration without interrupting its operation.

`journalctl -f` : Follows the journal, showing new log messages in real time.

`journalctl -u [unit]` : Displays logs for a specific systemd unit.

Cron jobs and scheduling

`crontab -e` : Edits cron jobs for the current user.

`crontab -l` : Lists cron jobs for the current user.

File management

`ls` : Lists files and directories.

`touch` : Creates an empty file, or updates the access and modification times of an existing file.

`cp` : Copies files from source to destination.

`mv` : Moves files or renames them.

`rm` : Deletes a file.

Directory navigation

`pwd` : Displays the current directory path.

`cd` : Changes the current directory.

`mkdir` : Creates a new directory.

File permissions and ownership

`chmod [who][+/-][permissions]` : Changes file permissions.

`chmod u+x` : Makes a file executable by its owner.

`chown [user]:[group] [file]` : Changes file owner and group.

Searching and finding

`find [directory] -name` : Finds files and directories.

`grep` : Searches for a pattern in files.

Archiving and compression

`tar -czvf [archive].tar.gz [files]` : Compresses files into a tar.gz archive.

`tar -xvf [archive].tar.gz -C [destination]` : Extracts a compressed tar archive into the destination directory.

Text editing and processing

`nano` : Opens a file in the Nano text editor.

`cat` : Displays the contents of a file.

`less` : Displays the paginated content of a file.

`head` : Shows the first few lines of a file.

`tail` : Shows the last few lines of a file.

`awk '{print}' [file]` : Prints every line in a file.

User management

`w` : Shows which users are logged in.

`sudo useradd [username]` : Creates a new user (Debian / Ubuntu also have the adduser front end).

`sudo userdel [username]` : Deletes a user (add -r to also remove the home directory).

`sudo passwd` : Sets or changes the password for a user.

`su` : Switches user.

`sudo passwd -l` : Locks a user account.

`sudo passwd -u` : Unlocks a user password.

`sudo chage` : Sets user password expiration date.

Group management

`id [username]` : Displays user and group IDs.

`groups [username]` : Shows the groups a user belongs to.

`sudo groupadd [group]` : Creates a new group.

`sudo groupdel [group]` : Deletes a group.

Sources

- <https://man7.org/linux/man-pages/man1/tar.1.html>
- <https://man7.org/linux/man-pages/man1/touch.1.html>
- <https://man7.org/linux/man-pages/man8/useradd.8.html>
- <https://man7.org/linux/man-pages/man8/userdel.8.html>
- <https://man7.org/linux/man-pages/man8/groupadd.8.html>
- <https://man7.org/linux/man-pages/man8/groupdel.8.html>
- <https://packages.fedoraproject.org/pkgs/htop/htop/>

Linux - Bag of Tricks

Introduction

This document has many useful command.

Change Files and Folder Permissions

To change the permissions of files to 655 and subfolders to 755 (which is the common practice for directories to allow execution for navigating into them) within a specified directory and its subdirectories, you can use the `find` command with `chmod`.

Explanation of permissions:

- **655 for files:**
 - Owner: Read (4) + Write (2) = 6
 - Group: Read (4) + Execute (1) = 5
 - Others: Read (4) + Execute (1) = 5
- **755 for directories:**
 - Owner: Read (4) + Write (2) + Execute (1) = 7
 - Group: Read (4) + Execute (1) = 5
 - Others: Read (4) + Execute (1) = 5

Command

```
find /path/to/directory -type f -exec chmod 655 {} +
find /path/to/directory -type d -exec chmod 755 {} +

find /path/to/directory \( -type d -exec chmod 755 {} + \) -o \( -type f -exec chmod 644 {} + \)

chmod -R u+rwX,go+rX,go-w /path/to/directory

find /path/to/directory -type d -print0 | xargs -0 chmod 755
find /path/to/directory -type f -print0 | xargs -0 chmod 644
```

```
find /path/to/directory -print0 \  
\\ -type d -exec chmod 755 {} + \\ \  
-o \\ -type f -exec chmod 644 {} + \\
```

Linux Set Time Examples

You can also simplify format using following syntax:

```
date +%Y%m%d -s "20081128"
```

To set time use the following syntax:

```
date +%T -s "10:13:13"
```

Use the following syntax to set new data and time:

```
date --set="STRING"
```

For example, set new data to 2 Oct 2006 18:00:00, type the following command as root user:

```
date -s "2 OCT 2006 18:00:00"
```

OR

```
date --set="2 OCT 2006 18:00:00"
```

Rclone Copy Examples

Click here for the page [HERE](#)

Rsync Copy Examples

This is to move files from one server to another

Ending the folder WITHOUT a "/" slash means copy that folder everything in that folder

Ending the folder WITH a "/" slash means copy everything within that folder

Example for "remote to local" location

```
rsync -chavzP --stats --progress -e ssh user@remote_host:/remote_folder/dir1/ /local_folder/dir1/
```

Example for “local to remote” location

```
rsync -chavzP --stats --progress -e ssh /local_folder/dir1/ user@remote_host:/remote_folder/dir1/
```

Rsync Auto Login while sending

Example to add a Rsync key on the remote server

On the local server simply login as a given user ex: ROOT or USER

```
ssh-keygen -t rsa
```

If it already exists simply hit "n"

```
Generating public/private rsa key pair.  
Enter file in which to save the key (/root/.ssh/id_rsa):  
/root/.ssh/id_rsa already exists.  
Overwrite (y/n)?
```

If not then simply hit enter through all options

Example: of using ROOT

```
Generating public/private rsa key pair.  
Enter file in which to save the key (/root/.ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /root/.ssh/id_rsa  
Your public key has been saved in /root/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:JoMN/cxvsqZWBHws4eyrU5Q0F0qRe//44qdrriQmbU root@client01  
The key's randomart image is:  
+---[RSA 3072]---+  
| .+=..      |  
| =B.+       |  
| ..=O       |  
| ==+o       |  
| = E=...    |  
| o... oo    |  
| ..o..++ o   |  
| .oo+o==*.   |  
+---[SHA256]-----+  
You have mail in /var/spool/mail/root
```

Run the following to add the key to the remote server, you can also use IP instead of host name

```
ssh-copy-id -i ~/.ssh/id_rsa.pub remuser@server01
```

Example of using a USER you will have to enter yes and the USER password

```
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host 'server02 (192.168.x.x)' can't be established.
ED25519 key fingerprint is SHA256:oZnvrgY+2Xpd2/huaffvzLMBAgI52AMPUmq/LPLIXbE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
remuser@server02's password:
tput: No value for $TERM and no -T specified
tput: No value for $TERM and no -T specified
tput: No value for $TERM and no -T specified
tput: No value for $TERM and no -T specified
tput: No value for $TERM and no -T specified
tput: No value for $TERM and no -T specified
tput: No value for $TERM and no -T specified
Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'remuser@server02'"
and check to make sure that only the key(s) you wanted were added.
```

Optional: If the command cannot be run above you can copy the key to the remote server manually into the “authorized_keys” file

```
cd
cd .ssh
vi authorized_keys
```

Optional: Change the permissions on the local server

```
chmod 600 ~/.ssh/*
chmod 711 ~/.ssh
chmod 711 ~
```

Synology Rsync

```
rsync -aXHmS --syno-acl /volum1/[xxx] /volume2/[xxx]
```

-a, --archive archive mode; equals -rlptgoD (no -H,-A,-X)
-p, --perms preserve permissions

-X, --xattrs preserve extended attributes
-o, --owner preserve owner (super-user only)
-g, --group preserve group
--syno-acl copy Synology ACL data

I use the following options myself:

```
rsync -avhXWog --stats --backup --suffix $OLDSUFFIX --exclude-from=$RSYEXCL --syno-pseudo-root
```

No idea why I list options "og" since they're implied by -a, but it works...

Regards, Arild

PS: "rsync --help" lists all available options for rsync

Find and Replace String with `sed`

There are several versions of `sed`, with some functional differences between them. macOS uses the BSD version, while most Linux distributions come with GNU `sed` pre-installed by default. We'll use the GNU version.

The general form of searching and replacing text using `sed` takes the following form:

```
sed -i 's/SEARCH_REGEX/REPLACEMENT/g' INPUTFILE
```

Cop

- `-i` - By default, `sed` writes its output to the standard output. This option tells `sed` to edit files in place. If an extension is supplied (ex -i.bak), a backup of the original file is created.
- `s` - The substitute command, probably the most used command in `sed`.
- `///` - Delimiter character. It can be any character but usually the slash (`/`) character is used.
- `SEARCH_REGEX` - Normal string or a regular expression to search for.
- `REPLACEMENT` - The replacement string.
- `g` - Global replacement flag. By default, `sed` reads the file line by line and changes only the first occurrence of the `SEARCH_REGEX` on a line. When the replacement flag is provided, all occurrences are replaced.
- `INPUTFILE` - The name of the file on which you want to run the command.

Find and Replace String with `sed` within `vi`

This is to search and replace a file globally withing vi

```
:%s/search_string/replacement_string/g
```

Kill Users in Linux

This is to be used when trying to kill users using the connection, replace the ? with the number of the session.

```
pkill -KILL -t pts/?
```

Continued in [Linux - Bag of Tricks \(Part 2\): Certificates, Databases, Disks & Boot.](#)

Linux - Bag of Tricks (Part 2): Certificates, Databases, Disks & Boot

This page continues [Linux - Bag of Tricks](#).

Create a CERT

First, you need to generate the private key and the Certificate Signing Request (CSR). You can do this via the `openssl` command:

```
openssl req -nodes -newkey rsa:2048 -keyout privatekey.key -out mail.csr
```

Then, generate a signing request

```
openssl x509 -req -days 365 -in mail.csr -signkey privatekey.key -out secure.crt
```

Create a localhost cert on the server

```
openssl req -newkey rsa:2048 -nodes -keyout /etc/pki/tls/private/localhost.key -x509 -days 365 -out  
/etc/pki/tls/certs/localhost.crt
```

Mariadb Log Rotate

If log file is large, try if the logrotate

```
logrotate --force /etc/logrotate.d/mariadb
```

MySQL Fail to Start

If MySQL does not restart, it probably will not as the index of the log files will not be changed

```
:d /var/lib/mysql
mv ib_logfile0 ib_logfile0.old
mv ib_logfile1 ib_logfile1.old
systemctl restart mariadb
```

Configure Rsync

Useful for system migrations

Create a “/etc/rsyncd.conf” containing:

[root]

exclude = /dev /etc/fstab /proc /sys

path = /

read only = yes

list = yes

uid = root

gid = root

Enable and start:

systemctl enable rsyncd.service

systemctl start rsyncd.service

Change Run level

systemctl set-default multi-user.target

To switch from graphical to multi-user:

systemctl isolate multi-user.target;

Change Local settings

timedatectl set-timezone Europe/London

localectl set-locale LANG=en_GB.UTF-8

localectl set-keymap uk

Temporary change

\$ loadkeys us

Configure Alternate Authentication

authconfig-tui

SSD Considerations

Change the value of “issue_discards” option from 0 to 1 in “/etc/lvm/lvm.conf”

```
#  
systemctl enable fstrim.timer
```

Adjust “/etc/fstab”

```
/dev/mapper/xxx /XXX    xfs    defaults,noatime,discard    0 0
```

Optionally set /tmp in RAM

```
# systemctl enable tmp.mount
```

Adding a Disk

```
# parted /dev/sdx
```

```
mklabel gpt
```

```
unit s
```

```
mkpart primary 2048s 100%
```

```
set 1 lvm on
```

```
quit
```

```
# pvcreate /dev/sdx1
```

```
# vgcreate rl_ssd /dev/sdx1
```

```
# lvcreate -L 50GB -n mysql rl_ssd
```

```
# mkfs.xfs /dev/rl-ssd/mysql
```

```
# blkid /dev/sdc
```

```
# chown mysql:mysql /var/lib/mysql
```

Growing a lvm partition

```
# parted /dev/sdc
```

(parted) unit b

(parted) print free

Number	Start	End	Size	Type	File system	Flags
1	31744B	5368709119B	5368677376B	primary		
	5368709120B	21474836479B	16106127360B			Free Space

(parted) resizepart 1 21474836479B

(parted) quit

pvresize /dev/sdc1

Updating Bootloader configuration

/etc/default/grub

grub2-mkconfig -o /boot/grub2/grub.cfg

NMAP Scan for all Open Ports

TCP

```
sudo nmap -sT -p- example.com
```

UDP

```
sudo nmap -sU -p- example.com
```

Look for open Ports

```
nc -vz 203.0.113.10 514
```

Trace route

```
sudo tracepath 203.0.113.10
```

Looking at the Journal, this is an example to look at the mariadb process

```
journalctl -u mariadb -f
```

Search in sub folders

```
grep -r "MYSQL_HOST" . --include="compose.yaml" --include="docker-compose.yaml"
```

Change Lines in KIDSENV and make a backup of it with a .bak extentions

```
find . -type f -name "KIDSENV" -exec sed -i.bak 's/^KWSQL_LOG=query/#KWSQL_LOG=query/' {} +
```

Delete file on the server

```
find . -type f \( -name "SQLQRY*" -o -name "SQLINFO" -o -name "SQLERROR" \) -delete
```

Delete files over 100M

```
find . -type f -size +100M
```

Delete files over 100M and list them

```
find . -type f -size +100M -exec ls -lh {} +
```

Grep files named TX*.DA that are only 1 month back and files lines that have RETDT or RETHD

```
find . -type f -name "TX*.DA" -mtime -30 -exec grep -iE "RETDT|RETHD" {} +
```

Remove certain directory

```
find /path/to/drive/root -type d -name "@eaDir" -print0 | xargs -0 rm -rf
```

Command - Cmnd_Alias

Last verified: October 2026 against the sudoers manual (sudo.ws).

What is Cmnd_Alias?

Cmnd_Alias (Command Alias) is a feature in `/etc/sudoers` (and files in `/etc/sudoers.d/`) that lets you **group multiple commands** under a single, easy-to-read name.

Instead of repeating long command paths many times, you define the group once and then reference the alias name in your user permission rules. This makes the sudoers configuration:

- Much cleaner and more readable
- Easier to maintain (add/remove commands in one place)
- Less error-prone

It is one of four main alias types in sudoers:

- User_Alias — groups of users
- Host_Alias — groups of hosts
- Runas_Alias — groups of users to run as
- **Cmnd_Alias** — groups of commands (this one)

Basic Syntax

```
Cmnd_Alias ALIAS_NAME = /full/path/to/command1, \  
                        /full/path/to/command2 arg1 arg2, \  
                        /full/path/to/command3
```

Rules:

- Alias name **must** start with a capital letter and can contain uppercase letters, numbers, and underscores (e.g., `API_SERVICE`, `SYSTEMCTL_API`).
- Always use **full absolute paths** to commands (never just `systemctl`).
- You can continue long lines with a backslash `\`.
- You can include other Cmnd_Alias names inside another one.

Example for Your api.service (Recommended Version)

Create or edit the file with `sudo visudo -f /etc/sudoers.d/10-deploy-api`:

```
# Command alias for managing the api.service safely
Cmnd_Alias API_SERVICE_CMDS = /usr/bin/systemctl start api.service, \
                               /usr/bin/systemctl stop api.service, \
                               /usr/bin/systemctl restart api.service, \
                               /usr/bin/systemctl status api.service

# Grant the deploy user passwordless access to only these commands
deployuser ALL=(ALL) NOPASSWD: API_SERVICE_CMDS
```

This is cleaner than listing the four commands directly on the user line.

More Flexible Example (Allow Any Action on the Specific Service)

If you want the deploy user to run **several** systemctl actions on api.service, do **not** use a wildcard such as `/usr/bin/systemctl * api.service`. In sudoers, wildcards in command arguments match any characters, including spaces, so that rule does not limit the user to api.service. It also matches, for example, `systemctl stop sshd.service api.service` or `systemctl edit api.service` (which opens an editor as root).

List each allowed command explicitly (as in the example above), or on sudo 1.9.10 and later (check with `sudo -V`) use a regular expression, which must start with `^` and end with `$`:

```
Cmnd_Alias API_SERVICE_CMDS = /usr/bin/systemctl ^(start|stop|restart|reload|status) api\.service$

deployuser ALL=(ALL) NOPASSWD: API_SERVICE_CMDS
```

This allows only the listed actions on api.service.

How to Use It in Your Setup

- Create the file:

```
sudo visudo -f /etc/sudoers.d/10-deploy-api
```

- Paste the Cmnd_Alias + user rule above.
- Set correct permissions:

```
sudo chmod 0440 /etc/sudoers.d/10-deploy-api
```

- Test it:

```
sudo -l -U deployuser
```

1. You should see the allowed commands listed for deployuser.

Then update your PowerShell script to use deployuser (instead of root).

Why This Is Better Than the Original Line

Your original line:

```
deployuser ALL=(ALL) NOPASSWD: /usr/bin/systemctl stop api.service, /usr/bin/systemctl restart api.service,  
/usr/bin/systemctl status api.service
```

Works fine, but using Cmnd_Alias is preferred when:

- You have more than 2-3 commands
- You might want to add more actions later
- You manage multiple services (you can create one alias per service)

Sources

- <https://www.sudo.ws/docs/man/sudoers.man/>