How to Clear Cache Memory on Linux?

written by sysadmin | 16 April 2025

Cache memory is a type of data storage used to store frequently accessed information for faster response time, and it’s used to improve system performance. In Linux, the kernel uses the buff/cache memory to improve system performance by caching frequently accessed data (disk blocks, inodes, etc.) and buffering I/O operations. So if you have an application that is often used and accessible to many people, you will see that the memory cache on the server will often be used. But sometimes, because of necessity, you have to clean the cache memory.

Problem

How to clear cache memory on Linux?

Solution

Before you delete the cache memory on Linux, you must know some of the terms related to the cache memory:

- Buffer: stores disk blocks that have been recently accessed or modified.
- PageCache: It improves file I/O performance by storing often-used file data in RAM.
- Dentries: help speed up file name searches by storing cached directory entries.
- Inodes: store important metadata about files and directories, they are also separate from the content or names.

So if you want to delete cache memory, use the format below:

```
sync; echo 1-3 > /proc/sys/vm/drop_caches
```

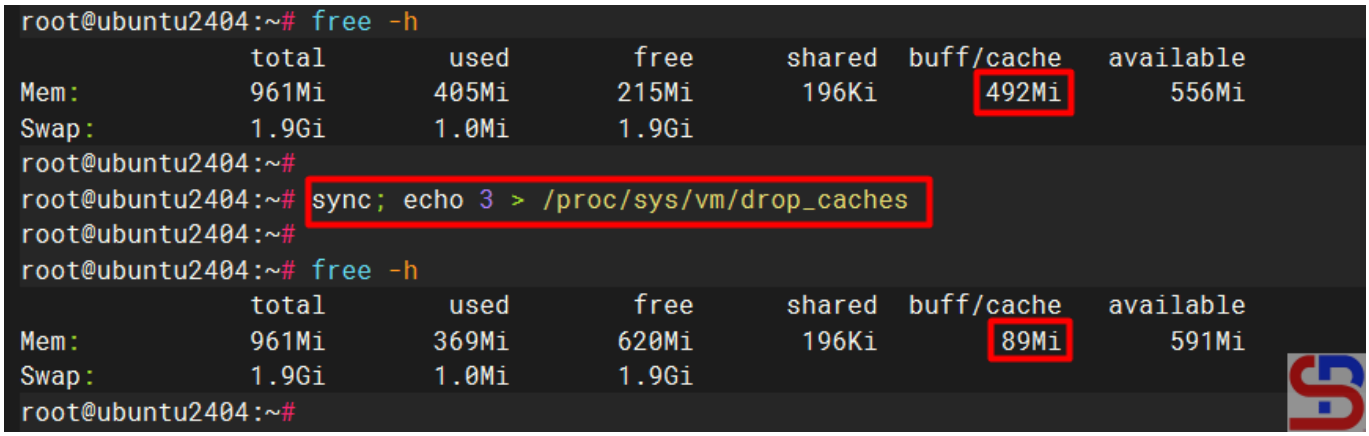
The sync command to clear the buffer and the drop_caches file controls which type of cached data should be cleared and the values are as follows:

- 1 – Clears only the page cache.
- 2 – Clears dentries and inodes.
- 3 – Clears page cache, dentries, and inodes.

So if you want to delete cache memory, use the format below:

```
sync; echo 3 > /proc/sys/vm/drop_caches
```

And the cache memory on your Linux server will be deleted as shown below:



Clear the cache memory

WARNING
You have to be root if you want to run commands to delete cache memory. If you want to use an ordinary user and want to run the command, then use the command below:

sudo sync; sudo sh -c 'echo 3 >/proc/sys/vm/drop_caches'

Note

It is not dangerous if you run the command to delete the memory cache on Linux but this will cause an increase in I/O disk because of the request that is usually handled by the memory cache because the memory cache is cleaned so that the

request will immediately proceed to the hard disk, causing an increase I/O disk. Also on the application side will cause a decrease in performance because the application does not receive a response from the memory cache, but must receive a response from the hard disk, whose response time is slower than from the cache memory.

WARNING
Only clear the cache for **debugging, benchmarking, or emergencies**. For normal usage, **let the kernel manage memory automatically**. Clearing it without a good reason hurts performance.

References

lenovo.com
eukhost.com
scaler.com
tecmint.com
linuxfordevices.com
unix.stackexchange.com
stackoverflow.com

How to Show The Progress Bar In Linux's cp and mv Commands?

written by sysadmin | 16 April 2025

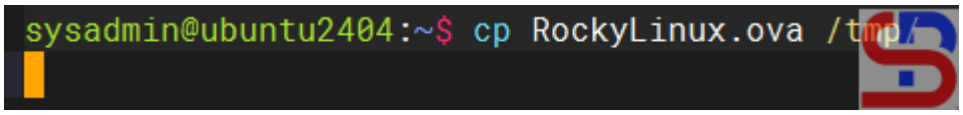
By default, if you do a copy command or move a file in Linux CLI, no progress bar shows how long the commands will take to complete. I think the users will feel bored because they do not know how long the commands will take to complete.

Problem

How to show the progress bar in Linux's cp and mv commands?

Solution

The image below shows no progress bar when you copy a file in Linux CLI:



Copy the file in Linux CLI

To display progress bars on cp and mv commands on Linux, you need a tool made by a GitHub user named jarun, who modified the source code of Florian Zwicke. But before you can install this tool, you must install the required packages:

RockyLinux/AlmaLinux/CentOS

```
yum install -y tar curl patch make coreutils gcc libattr* libpcap* perl libacl*
```

Debian/Ubuntu

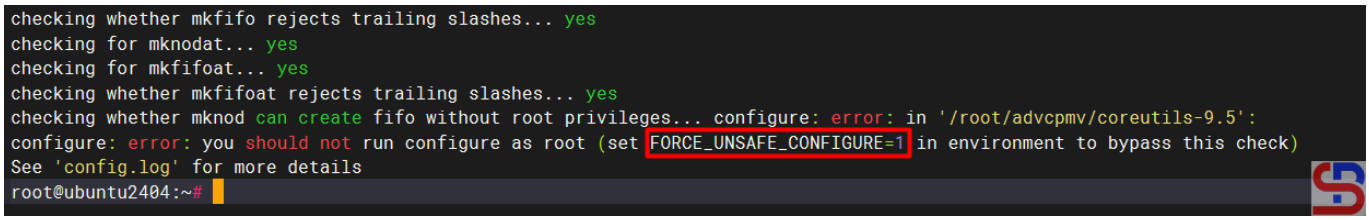
```
apt-get install -y tar curl patch make coreutils gcc libattr* libpcap* perl libacl*
```

OpenSUSE

```
zypper install -y tar curl patch make coreutils gcc libattr* libpcap* perl libacl*
```

After that, install the tool, and it is recommended to use a normal user to install the tool, but if you want to use the root user to install this tool, run the command below:

```
export FORCE_UNSAFE_CONFIGURE=1
```



Error when using the root user

Install the tool on your server by running the command below:

```
curl https://raw.githubusercontent.com/jarun/advcpmv/master/install.sh --create-dirs -o ./advcpmv/install.sh && (cd advcpmv && sh install.sh)
```

After installation completes, it created two new commands under the **advcpmv** folder. You need to replace your original cp and mv commands with these two new commands to get the progress bar while copying files:

```
sudo mv ./advcpmv/advcp /usr/local/bin/cp
sudo mv ./advcpmv/advmv /usr/local/bin/mv
```

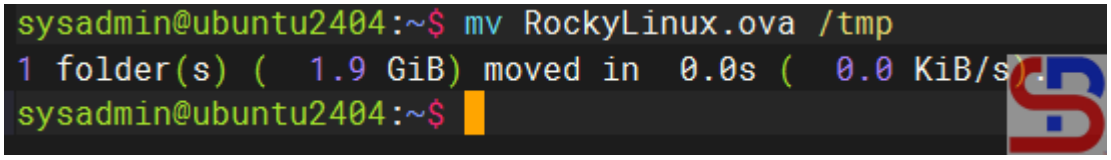
After that, run the commands below:

```
echo "alias cp='/usr/local/bin/cp -g'" >> ~/.bashrc
echo "alias mv='/usr/local/bin/mv -g'" >> ~/.bashrc
source ~/.bashrc
```

Try to copy a file in your Linux CLI, and there should be a progress bar when you copy the file:

The progress bar appears when you copy the file

Likewise, if you are going to move a file on Linux CLI, there will be a notification as below:



The notification appears when you move the file

Note

You should know that if you want to use this tool on a different user from the user where this tool is installed on the server (for example, you installed this tool using another user but you want to use the tool using another user), you need to use the command below using your user so that this tool can be used on that user:

```
echo "alias cp='/usr/local/bin/cp -g'" >> ~/.bashrc
echo "'alias mv='/usr/local/bin/mv -g'" >> ~/.bashrc
source ~/.bashrc
```

References

tecmint.com
ostechnix.com
stackoverflow.com

How to Protect the Linux Server From an Accidental Reboot?

written by sysadmin | 16 April 2025

As a Sysadmin, accessing a Linux server is a normal daily activity. But sometimes we accidentally make mistakes rebooting or shutting down the production server, causing the server to be inaccessible. Therefore, we need a tool to confirm if someone reboots or shuts down a Linux server.

Problem

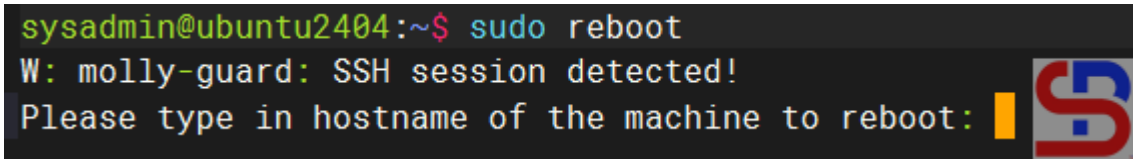
How to protect the Linux server from an accidental reboot or shutdown?

Solution

In the Debian/Ubuntu distribution, the molly-guard tool can be used to protect the Linux server from an accidental reboot or shutdown. Use the two commands below to install molly-guard:

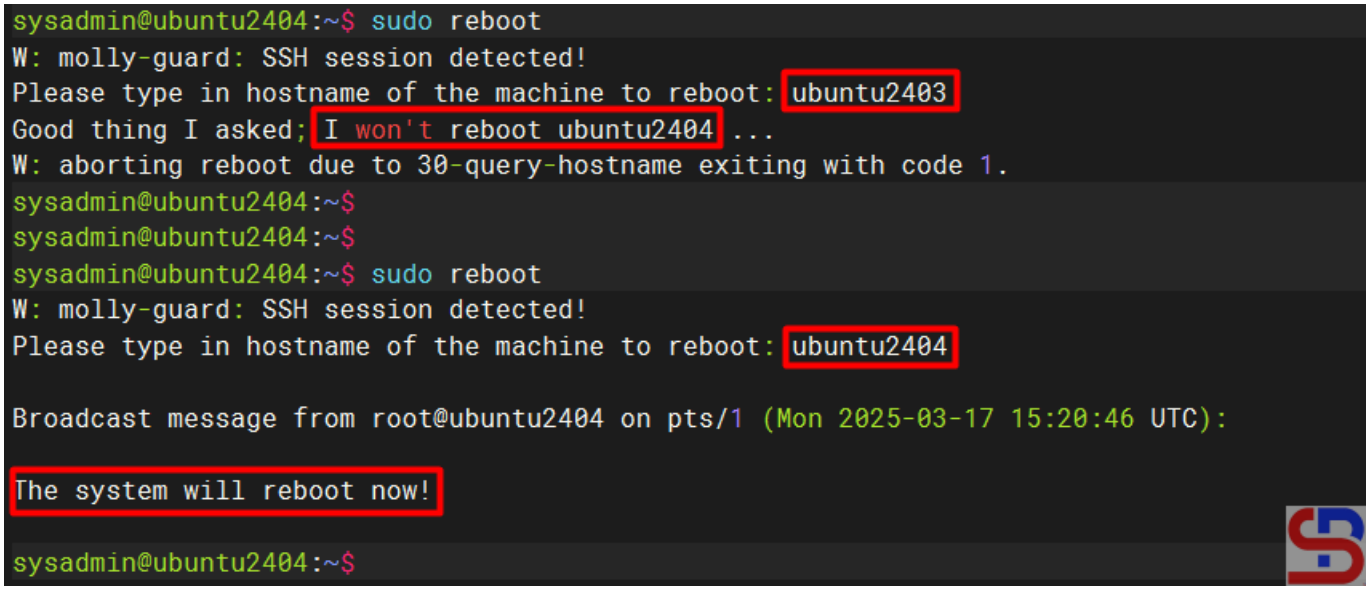
```
sudo apt update
sudo apt-get install molly-guard
```

After that, try to reboot the server, and there should be a notification like the image below:



A notification appears when trying to reboot the server

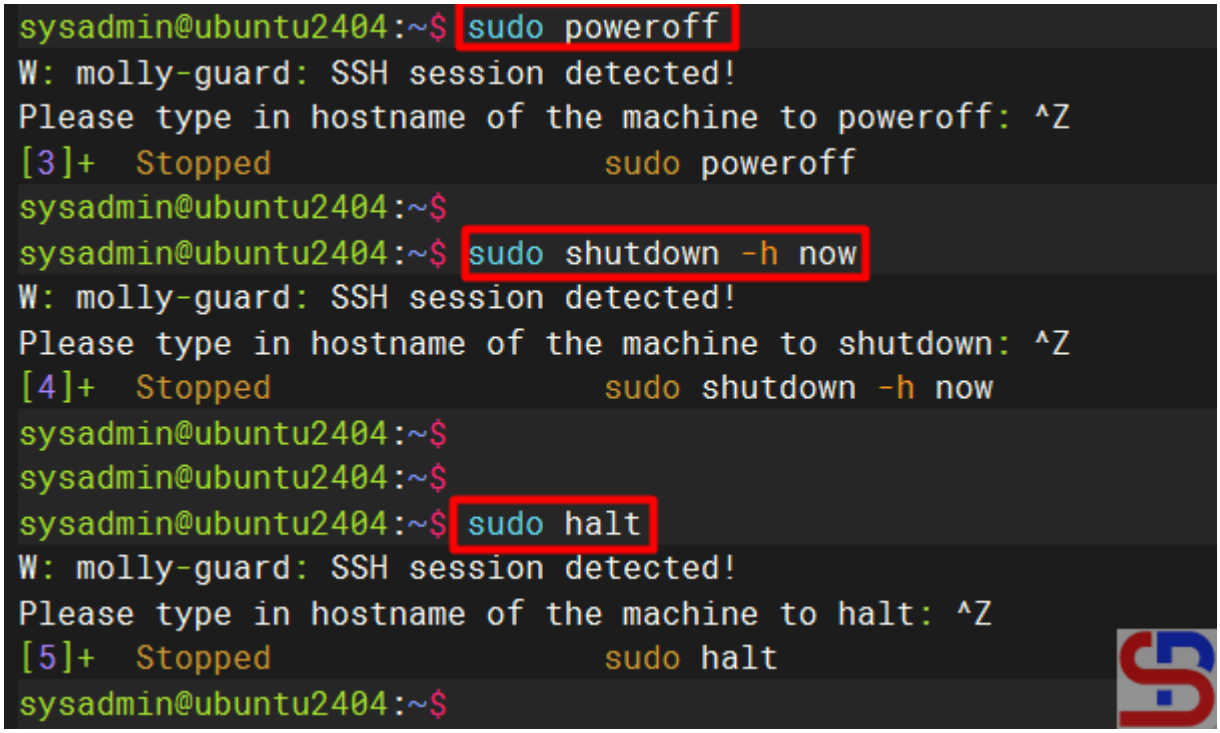
Someone who wants to reboot the server must write the server's hostname. If the nameserver does not match the hostname on the server, the reboot process will not be continued, but if it matches the hostname on the server, the reboot process will be continued.



Try to reboot the server

This is very useful if the sysadmin accidentally types the

reboot command on the server. However, this tool not only protects the server from the reboot command, but also other commands such as the **poweroff**, **shutdown**, **coldreboot**, **pm-hibernate**, **pm-suspend**, and **pm-suspend-hybrid** commands.



Try to turn off the server

Note

Keep in mind that this molly-guard tool can only work in the Debian/Ubuntu distribution and its derivatives, and this tool only works on SSH connections. If you access the Linux server without an SSH connection, for example, by directly connecting the keyboard to the Linux server, this tool will not work, so if you run the reboot command, the Linux server will immediately reboot.

References

manpages.ubuntu.com
launchpad.net
techbits.io

How to Change the Color of Comments in the vi Application?

written by sysadmin | 16 April 2025

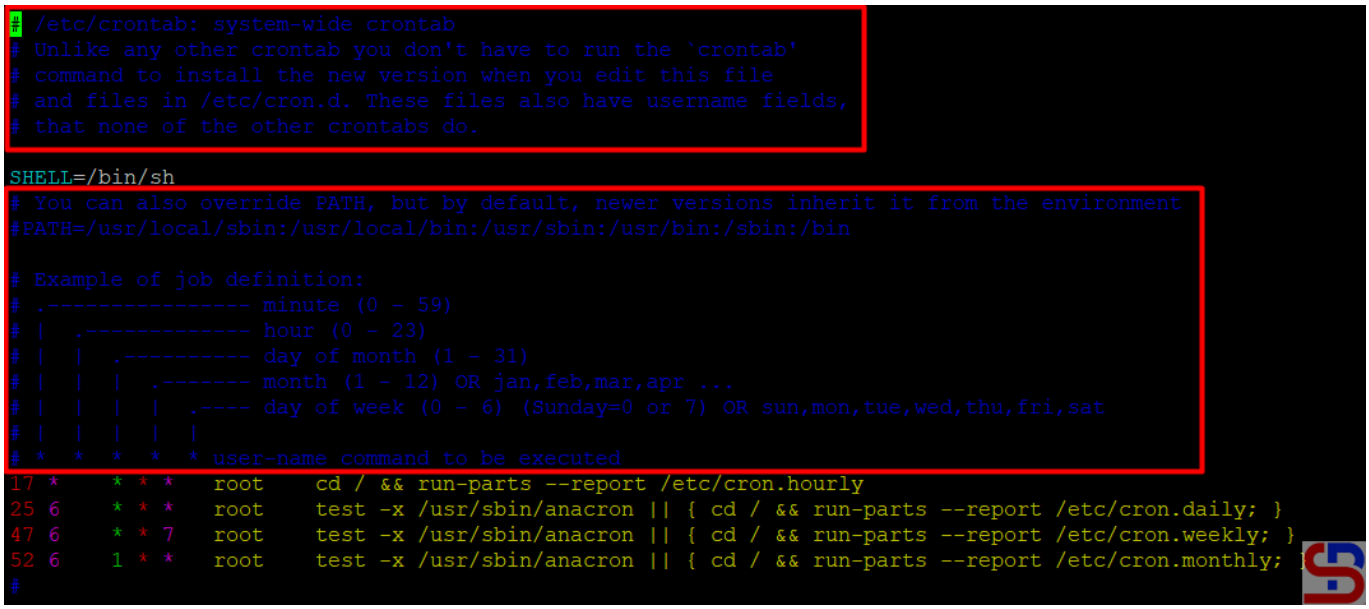
By default, if you open the vi application on Linux (especially using PuTTY), the color for comments is blue. However, sometimes this makes it very difficult for me to read the comments, especially if the background color of the terminal is black.

Problem

How to change the color of comments in the vi application?

Solution

Below is an image of a **/etc/crontab** file opened using Putty:



Comment color in the vi application

For me, it’s very difficult to read the comments in the Linux file if they’re blue like in the image above. So, if

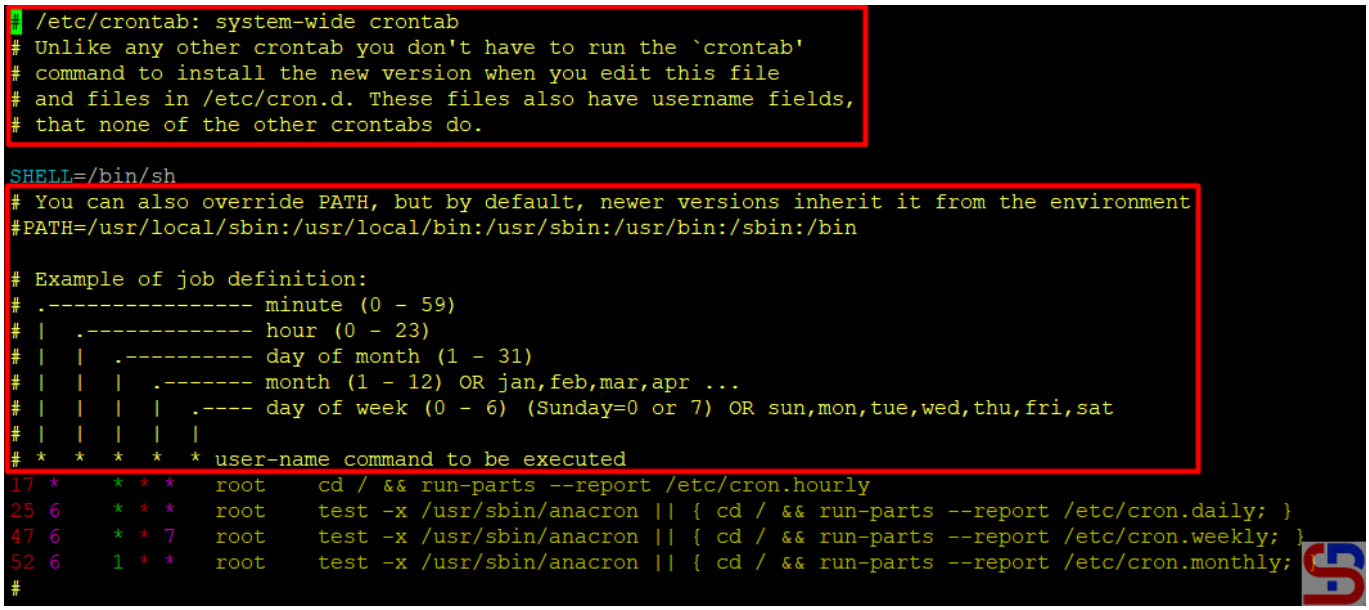
you want to change the color of comments in the vi application, for example, if you want to change the color of comments to yellow, then open the .vimrc file by:

```
vi ~/.vimrc
```

Type the script below into the file:

```
highlight Comment ctermfg=yellow
```

After that, open the file using the vi application, and the comments on the file should change to yellow as in the image below:



Comment color after configuration in the vi application

You can see that the comment color is changed to yellow after you configure **the .vimrc** file.

Info

Please note that the steps above only change per user. If you want all users to change the comment color to yellow then place the above command in the **/etc/vim/vimrc.local** file.

Note

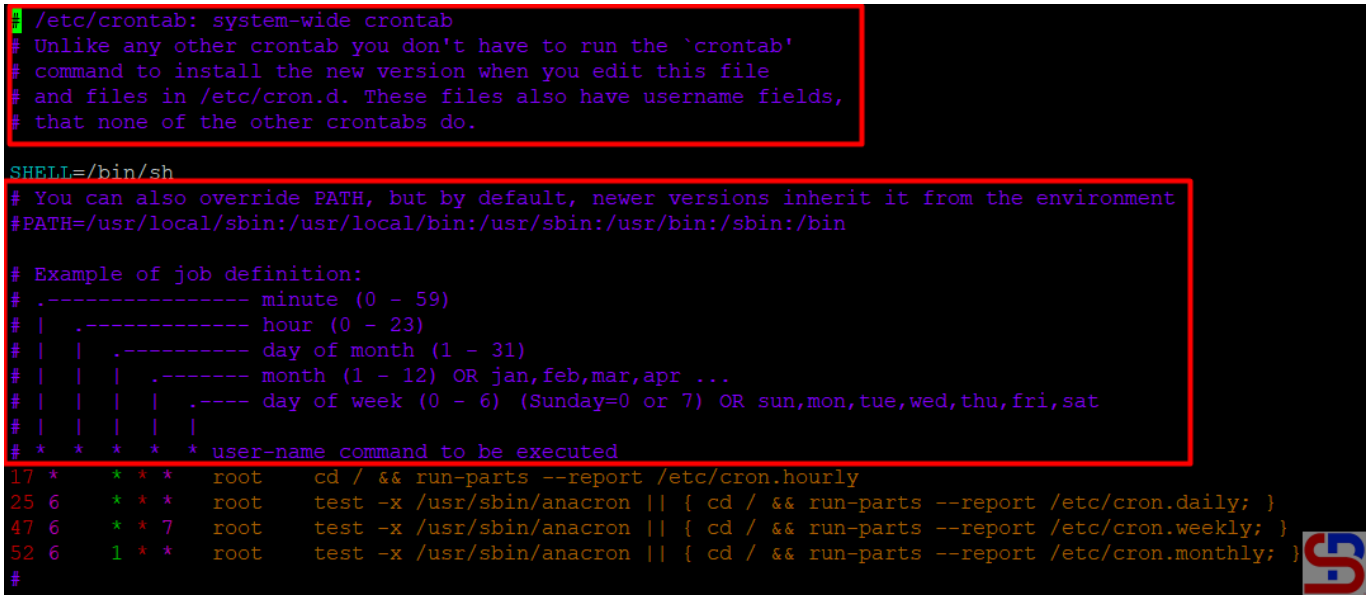
Currently, the vi application can support up to 256 colors that can be used in the vi application. So if you want to use more colors supported by the vi application, in the **.vimrc** file type the script below:

```
set t_Co=256
```

Then you can choose the colors on this page, for example, you want to use purple for comments, then enter the script below in the **.vimrc** file:

```
set t_Co=256
highlight Comment ctermfg=93
```

Then the comments in the vi/vim application should be purple as in the image below:



Change the color of comments in the vi

References

spinspire.com
linode.com

How to Change Crontab Using a Bash Script?

written by sysadmin | 16 April 2025

Crontab, which stands for cron table, is used to run one or more scripts in Linux based on a specific time. Usually, if you want to change something in the crontab, you use the **crontab -e** command and then change the crontab. But I want to change crontab using a shell script.

Problem

How to change crontab using a bash script?

Solution

I create a bash script to execute something on my Linux server, and in my script, I want to change the crontab so the script will add, change, or remove the script in the crontab. Here are ways to change crontab using a script:

1. Add a script in crontab

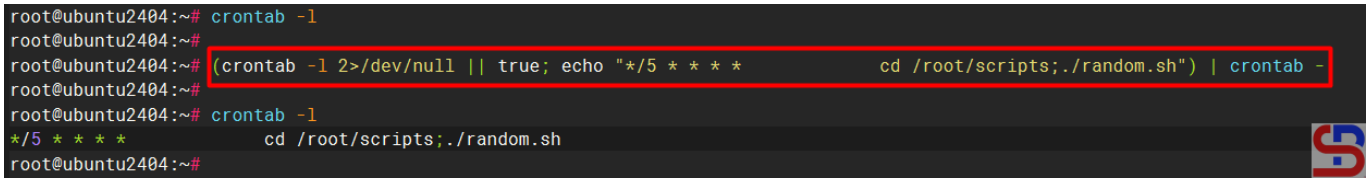
For example, if you want to add a **random.sh** script which is in the **/root/scripts** folder in crontab and will run every 5 minutes, then use the command below:

```
(crontab -l 2>/dev/null || true; echo "*/5 * * * *
/root/scripts/random.sh") | crontab -
```

Add a script to the crontab

Or if you want to add the script to the crontab in another form of writing, then you can use the command below:

```
(crontab -l 2>/dev/null || true; echo "*/5 * * * *             cd
/root/scripts;./random.sh") | crontab -
```



Another method to add the script to the crontab

2. Change the script in crontab

If you want to change the file in crontab to once every 10 minutes (previously every 5 minutes) for the random.sh script in the /root/scripts folder, then use the command below:

```
crontab -l | sed 's/\*\/5 \* \* \* \*
\/root\/scripts\/random.sh/\*\/10 \* \* \* \*
\/root\/scripts\/random.sh/g' | crontab -
```

change crontab using a script

Or, you can execute the command below if your script uses another form of writing in crontab:

```
crontab -l | sed 's/\*\/5 \* \* \* \*             cd\
\/root\/scripts\;\.\/random.sh/\*\/10 \* \* \* \*             cd\
\/root\/scripts\;\.\/random.sh/g' | crontab -
```

Change the script using another form of writing in Crontab using a script

3. Disable and enable the script

If you want to disable the random.sh script in crontab, then

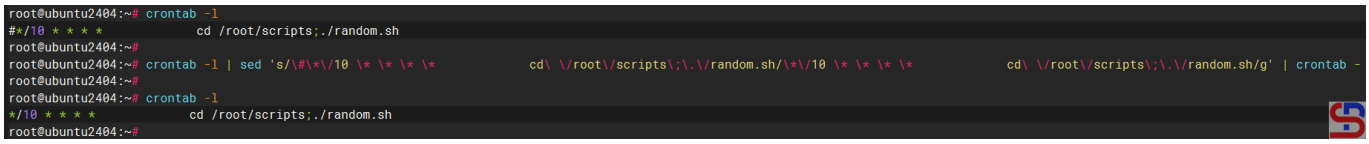
use the command below:

```
crontab -l | sed 's/\*\/10 \* \* \* \*              cd\
\/root\/scripts\;\.\/random.sh/\#\*\/10 \* \* \* \*             cd\
\/root\/scripts\;\.\/random.sh/g' | crontab -
```

Disable the script in the crontab

But if you want to enable it, use the command below:

```
crontab -l | sed 's/\#\*\/10 \* \* \* \*              cd\
\/root\/scripts\;\.\/random.sh/\*\/10 \* \* \* \*             cd\
\/root\/scripts\;\.\/random.sh/g' | crontab -
```



Enable the script in the crontab

4. Deleting the script in crontab

Use the command below if you want to delete the random.sh file in crontab:

```
crontab -l | sed '/\*\/5 \* \* \* \*        \/root\/scripts\/random.sh/d' |
crontab -
```

Delete the script in the crontab

Or, you can execute the command below if your script uses another form of writing in crontab:

```
crontab -l | sed '/\*\/5 \* \* \* \*             cd\
\/root\/scripts\;\.\/random.sh/d' | crontab -
```

Delete the script in crontab using another form

Note

You have to pay attention to whether the script in the crontab uses spaces or tabs because it greatly affects whether the script that you run can change something in the crontab or not. You have to put a backslash(\) if you want to change or delete your script in crontab that uses symbols like an asterisk(*), forward slash(/), hash(#), space, and so on.

References

techtarget.com
stackoverflow.com
webopedia.com

How to Move the Partition to a New Partition in the Linux Server?

written by sysadmin | 16 April 2025

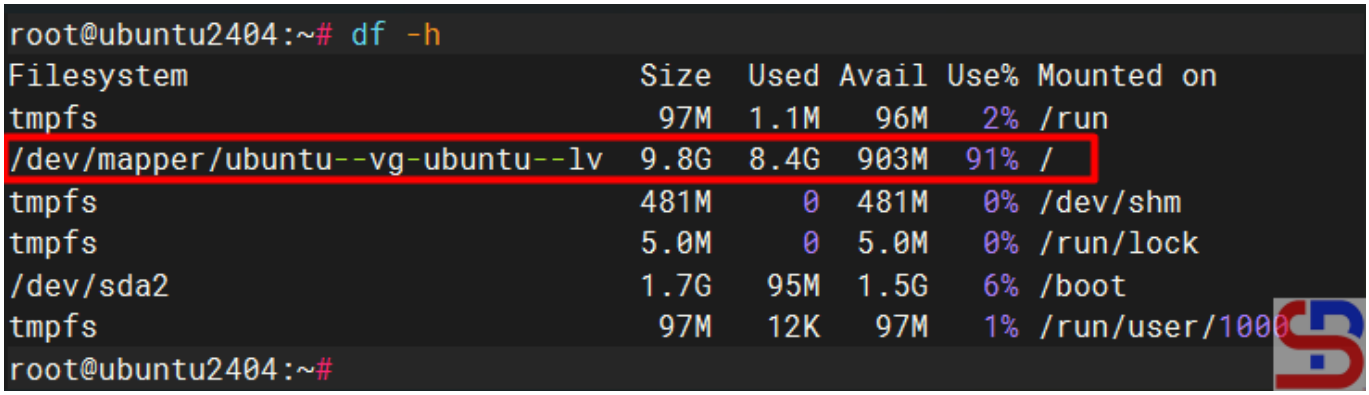
If you install a Linux server, you will usually install it with only one partition and not separate the other partitions. Problems will arise if one of these partitions uses a large enough hard disk, resulting in you running out of HDD space on your Linux server.

Problem

How to move the partition to a new partition in the Linux server?

Solution

In this article, I use the Ubuntu Server OS, and this article should be applied to any Linux distribution. Currently, the condition of the hard disk on my Ubuntu server is like the image below:



Condition of the hard disk in my Ubuntu server

From the image above, the root partition only has a free HDD of 9 percent. After I checked, it turned out that the cause was the /var partition, which took up a lot of hard disk so I want to move the **/var** partition to the new partition.

```
root@ubuntu2404:/# du -sh *
0       bin
4.0K    bin.usr-is-merged
95M     boot
4.0K    cdrom
0       dev
6.1M    etc
32K     home
0       lib
0       lib64
4.0K    lib.usr-is-merged
16K     lost+found
4.0K    media
4.0K    mnt
4.0K    opt
du: cannot access 'proc/1185/task/1185/fd/4': No such file or directory
du: cannot access 'proc/1185/task/1185/fdinfo/4': No such file or directory
du: cannot access 'proc/1185/fd/3': No such file or directory
du: cannot access 'proc/1185/fdinfo/3': No such file or directory
0       proc
32K     root
1.1M    run
0       sbin
4.0K    sbin.usr-is-merged
8.0K    snap
4.0K    srv
2.0G    swap.img
0       sys
64K     tmp
2.0G    usr
4.5G    var
root@ubuntu2404:/#
```

Check the largest partition size

Here are the steps to move the partition to a new partition in the Linux Server:

Info

The steps in this article will make your Linux server **enter maintenance mode** which means that the Linux server cannot be accessed from anywhere results in the application or database that may be in the Linux server also inaccessible. So discuss first with your boss if you want to do the steps in this article

1. Add a new hard drive

I insert a new 10 GB HDD into my Linux server. After that, I check if the new HDD is detected by Linux using the command:

```
fdisk -l
```

```
root@ubuntu2404:/# fdisk -l
Disk /dev/sda: 15 GiB, 16106127360 bytes, 31457280 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 33862586-79A4-4B47-A641-CFC4AB6AF897

Device       Start      End  Sectors  Size Type
/dev/sda1     2048     4095     2048    1M BIOS boot
/dev/sda2     4096  3674111  3670016  1.8G Linux filesystem
/dev/sda3  3674112 31455231 27781120 13.2G Linux filesystem


Disk /dev/sdb: 10 GiB, 10737418240 bytes, 20971520 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/ubuntu--vg-ubuntu--lv: 10 GiB, 10737418240 bytes, 20971520 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
root@ubuntu2404:/#
```

Check the new HDD in the Linux server

From the image above, it can be seen that the new HDD was detected by Linux with a partition in sdb.

2. Create a new partition

Run the command below to create a new partition in Linux (Adjust to the hard disk partition detected on your Linux server after typing the **fdisk -l** command):

```
fdisk /dev/sdb
```

Press the **n** and **p** keys, then the number **,1** and **enter 2x,** then press the **w** button as seen in the image below:

```
root@ubuntu2404:/# fdisk /dev/sdb

Welcome to fdisk (util-linux 2.39.3).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS (MBR) disklabel with disk identifier 0xf65c8ed5.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-20971519, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-20971519, default 20971519):

Created a new partition 1 of type 'Linux' and of size 10 GiB.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

root@ubuntu2404:/#
```

Create a new partition in the new HDD

Then create a filesystem from the new HDD, and I want to use **ext4** for the filesystem of the new HDD using the command:

```
mkfs.ext4 /dev/sdb1
```

```
root@ubuntu2404:/# mkfs.ext4 /dev/sdb1
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 2621184 4k blocks and 655360 inodes
Filesystem UUID: 930db9c1-62d3-49ab-8482-713510fa2604
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

root@ubuntu2404:/#
```

Create a filesystem in the partition of the new HDD

3. Create and mount a new folder

After that, create a new folder using the command:

```
mkdir /mnt/newvar
```

Then, mount the new partition to the new folder using the command:

```
mount /dev/sdb1 /mnt/newvar
```

```
root@ubuntu2404:/# mkdir /mnt/newvar
root@ubuntu2404:/#
root@ubuntu2404:/# mount /dev/sdb1 /mnt/newvar
root@ubuntu2404:/#
root@ubuntu2404:/# df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                               97M  1.1M   96M   2% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  9.8G  8.4G  905M  91% /
tmpfs                              481M     0  481M   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sda2                          1.7G   95M  1.5G   6% /boot
tmpfs                               97M   12K   97M   1% /run/user/1000
/dev/sdb1                          9.8G   24K  9.3G   1% /mnt/newvar
root@ubuntu2404:/#
```

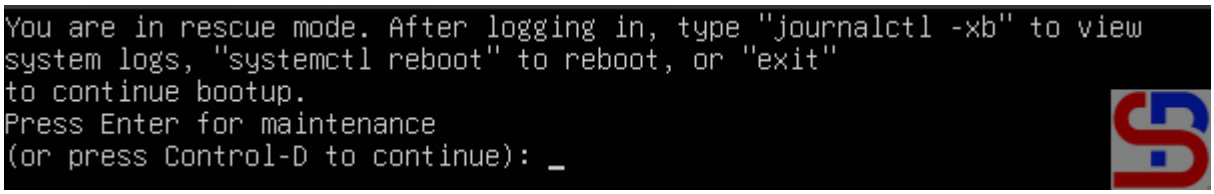
Create and mount a new folder

4. Enter maintenance mode

Type the command below:

```
init 1
```

to enter the rescue mode:



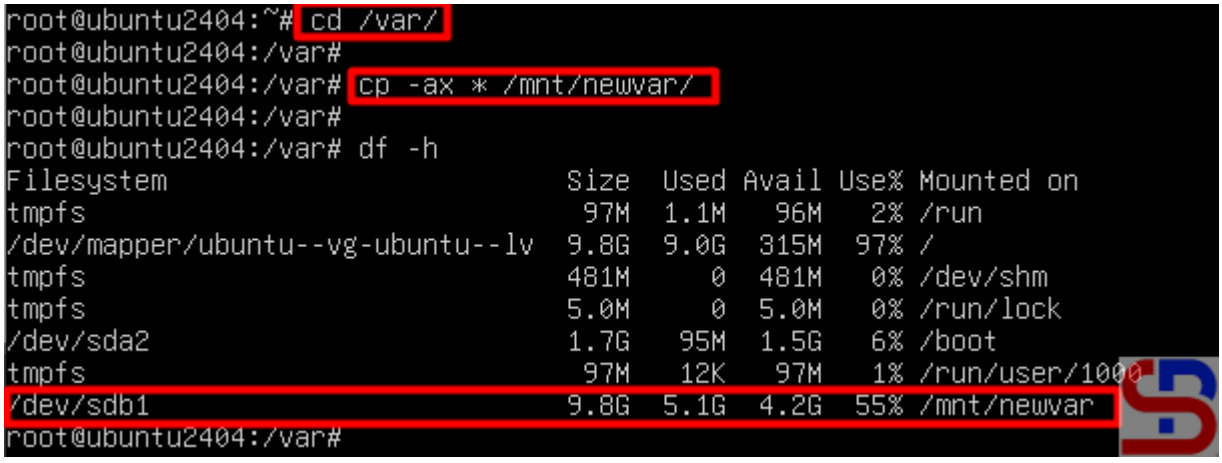
Enter the maintenance mode

After that, press the **Enter button** to enter maintenance mode.

5. Copy the folder

Go to the /var folder and copy all the files and folders in the folder into a new folder by typing the following commands:

```
cd /var
cp -ax * /mnt/newvar
```



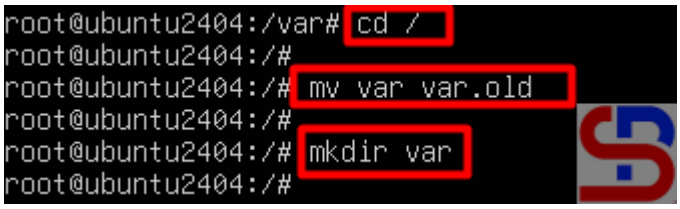
Copy the folder

6. Rename the folder

Once the copy process is complete, change the /var folder to the var.old folder and then create a new /var folder using

the command:

```
cd /
mv var var.old
mkdir /var
```



Rename the folder

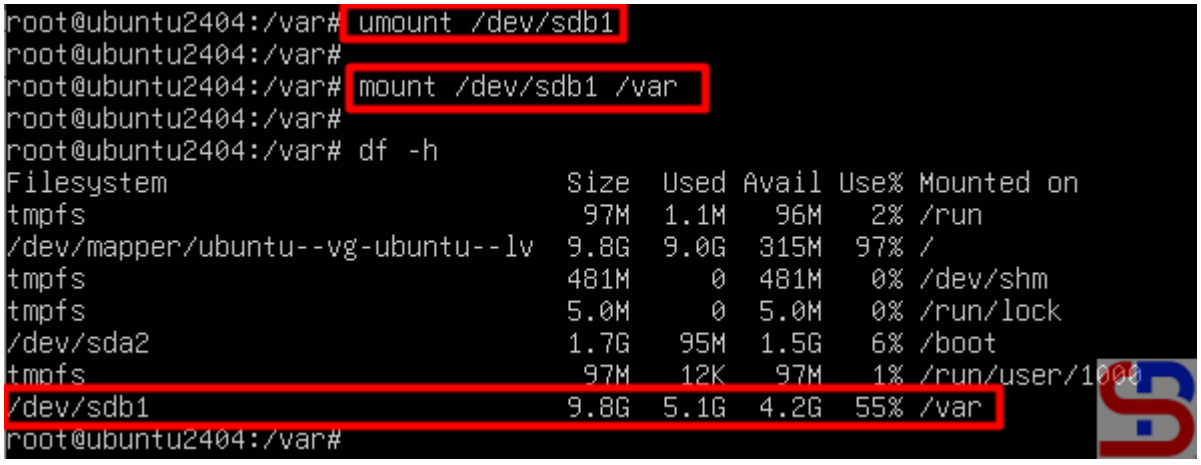
7. Mount the new folder

Next, do umount on the /sdb1 partition by using the command:

```
umount /dev/sdb1
```

And mount the /sdb1 partition to the new /var folder using the command:

```
mount /dev/sdb1 /var
```



Mount the new folder

8. Change the fstab file

Change the **/etc/fstab** file by adding the following script to the file:

```
/dev/sdb1  /var        ext4        defaults        0 0
```

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/ubuntu-vg/ubuntu-lv during curtin installation
/dev/disk/by-id/dm-uuid-LVM-g4VC0MaxbNzT5D0AVCzrf17icg24GBNlPwhuSsXpRZC80pJaXQMcFFctYEhUO0j5 / ext4 defaults 0 1
# /boot was on /dev/sda2 during curtin installation
/dev/disk/by-uuid/c59b0229-fcf2-4f2f-a6c7-e183c8ca6093 /boot ext4 defaults 0 1
/swap.img       none    swap    sw      0       0
/dev/sdb1       /var    ext4    defaults        0 0
```

Script additions in fstab file

9. Restart the server

After that, restart the Linux server and make sure there is no problem when the Linux server reboots.

10. Delete the folder

If the Linux server has finished restarting, then you can delete the var.old folder so that the size of the hard disk of the root partition increases by using the command:

```
cd /
rm -rf var.old
```

```
root@ubuntu2404:~# df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                               97M  1.1M   96M   2% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  9.8G  8.4G  905M  91% /   Before delete var.old folder
tmpfs                              481M     0  481M   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sdb1                          9.8G  4.6G  4.8G  49% /var
/dev/sda2                          1.7G   95M  1.5G   6% /boot
tmpfs                               97M   12K   97M   1% /run/user/1000
root@ubuntu2404:~#
root@ubuntu2404:~# cd /
root@ubuntu2404:/#
root@ubuntu2404:/# rm -rf var.old
root@ubuntu2404:/#
root@ubuntu2404:/# df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                               97M  1.1M   96M   2% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  9.8G  3.9G  5.4G  42% /   After delete var.old folder
tmpfs                              481M     0  481M   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sdb1                          9.8G  4.6G  4.8G  49% /var
/dev/sda2                          1.7G   95M  1.5G   6% /boot
tmpfs                               97M   12K   97M   1% /run/user/1000
root@ubuntu2404:/#
```

Before and after moving the partition

Note

Reboot the server again to make sure there are no problems after you delete the var.old folder. You can use the steps above when you want to move another folder to a new partition in the Linux server.

References

blog.oshim.net
phoenixnap.com

How to Manage a Container in Docker?

written by sysadmin | 16 April 2025

The previous article explained how to install Docker on Linux. This article will explain how to manage a container in Docker.

Problem

How to manage a container in Docker?

Solution

To manage a container in Docker, you have to remember basic Docker commands. And here are the basic Docker commands:

1. Search for container images

To run containers in Docker, we need a Docker image. A Docker image is an immutable (unchangeable) file that

contains the source code, libraries, dependencies, tools, and other files needed for an application to run. The place to store Docker images is known as the Docker registry, which by default uses the Docker Hub located at **hub.docker.com**. If you are looking for a container image in Docker, use the format below:

```
docker search container_name
```

For example, if you want to find an nginx image, then use the command below:

```
docker search nginx
```

```
sysadmin@ubuntu2404:~$ docker search nginx
NAME                                DESCRIPTION                                     STARS     OFFICIAL
nginx                               Official build of Nginx.                        20651     [OK]
nginx/nginx-ingress                 NGINX and  NGINX Plus Ingress Controllers fo…  100
nginx/nginx-prometheus-exporter     NGINX Prometheus Exporter for NGINX and NGIN…  48
nginx/unit                          This repository is retired, use the Docker o…  65
nginx/nginx-ingress-operator        NGINX Ingress Operator for NGINX and NGINX P…  2
nginx/nginx-quic-qns                NGINX QUIC interop                              1
nginx/unit-preview                  Unit preview features                           0
nginx/nginxaas-loadbalancer-kubernetes                                              0
bitnami/nginx                       Bitnami container image for NGINX               196
ubuntu/nginx                        Nginx, a high-performance reverse proxy & we…  127
bitnamicharts/nginx                 Bitnami Helm chart for NGINX Open Source        0
rancher/nginx                                                                       2
kasmweb/nginx                       An Nginx image based off nginx:alpine and in…  8
linuxserver/nginx                   An Nginx container, brought to you by LinuxS…  227
redash/nginx                        Pre-configured nginx to proxy linked contain…  3
dtagdevsec/nginx                    T-Pot Nginx                                     0
vmware/nginx                                                                        2
paketobuildpacks/nginx                                                              0
chainguard/nginx                    Build, ship and run secure software with Cha…  4
gluufederation/nginx                 A customized NGINX image containing a consu…  1
intel/nginx                                                                         0
droidwiki/nginx                                                                     0
circleci/nginx                      This image is for internal use                  2
corpusops/nginx                     https://github.com/corpusops/docker-images/     1
antrea/nginx                        Nginx server used for Antrea e2e testing        0
sysadmin@ubuntu2404:~$
```

Searching the nginx image

2. Download the Docker image

To download the Docker image, use the following format:

```
docker image pull image_name:tag_version
```

where the tag_version is the version of the image, and if

you don’t write the tag, it is considered that you want to install the latest version of the image. For example, if you want to download the newest version of the nginx image, use the command:

```
docker image pull nginx
```

```
sysadmin@ubuntu2404:~$ docker image pull nginx
Using default tag: latest
latest: Pulling from library/nginx
7cf63256a31a: Pull complete
bf9acace214a: Pull complete
513c3649bb14: Pull complete
d014f92d532d: Pull complete
9dd21ad5a4a6: Pull complete
943ea0f0c2e4: Pull complete
103f50cb3e9f: Pull complete
Digest: sha256:9d6b58feebd2dbd3c56ab5853333d627cc6e281011cfd6050fa4bcf2072c9496
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
sysadmin@ubuntu2404:~$
```

Download the nginx image

But if you want to download nginx with a certain version, for example, version 1.27.2, then use the command:

```
docker image pull nginx:1.27.2
```

```
sysadmin@ubuntu2404:~$ docker pull nginx:1.27.2
1.27.2: Pulling from library/nginx
2d429b9e73a6: Pull complete
9b1039c85176: Pull complete
9ad567d3b8a2: Pull complete
773c63cd62e4: Pull complete
1d2712910bdf: Pull complete
4b0adc47c460: Pull complete
171eebbdf235: Pull complete
Digest: sha256:bc5eac5eafc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
Status: Downloaded newer image for nginx:1.27.2
docker.io/library/nginx:1.27.2
sysadmin@ubuntu2404:~$
```

Download the nginx with a certain version

3. List the Docker image(s)

To display the Docker image that you have downloaded, use the command below:

```
docker image ls
```

```
sysadmin@ubuntu2404:~$ docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
nginx        latest    b52e0b094bc0   5 weeks ago   192MB
sysadmin@ubuntu2404:~$
```

List the Docker images

Or you can use the command below:

```
docker images
```

```
sysadmin@ubuntu2404:~$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
nginx        latest    60c8a892f36f   6 months ago   192MB
sysadmin@ubuntu2404:~$
```

List the Docker images

4. Create a container

You can create the container using the format:

```
docker container create –name container_name image_name:tag
```

For example, if you want to create a container with the name webapp1, which contains the nginx application, then use the command below:

```
docker container create --name webapp1 nginx
```

When you use this command, Docker will first check whether the nginx image is on the server. If the image is not on the server, then Docker will download the nginx image, and after that, it will create an nginx container, and the image will remain on your server, as shown in the image below:

```
sysadmin@ubuntu2404:~$ docker container create --name webapp1 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
7cf63256a31a: Pull complete
bf9acace214a: Pull complete
513c3649bb14: Pull complete
d014f92d532d: Pull complete
9dd21ad5a4a6: Pull complete
943ea0f0c2e4: Pull complete
103f50cb3e9f: Pull complete
Digest: sha256:9d6b58feebd2dbd3c56ab5853333d627cc6e281011cfd6050fa4bcf2072c9496
Status: Downloaded newer image for nginx:latest
b6b73bdbb4e39e83bdd8090478868b41b895f847910d65d44b744955562c4cce
sysadmin@ubuntu2404:~$
```

Create the container

You can use an image to create multiple containers as long as the container names are different, as in the image below:

```
sysadmin@ubuntu2404:~$ docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
nginx        latest    b52e0b094bc0   5 weeks ago   192MB
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker container create --name webapp1 nginx
9fbc044864d767f3ffc2d321654e3f66db8ea9be5a1d06f8023bf2e0ffbff6d2
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker container create --name webapp2 nginx
2f22437fcb9d149c542d74ffb6dda54112e88479fecfb16171e819ee0995f006
sysadmin@ubuntu2404:~$
```

Create the containers with 1 image

5. List the status of the container(s)

To display the container status, you can use the command:

```
docker ps
```

```
sysadmin@ubuntu2404:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
sysadmin@ubuntu2404:~$
```

List the status of running Docker

Maybe you are confused about why there is no container status displayed, even though you have made 2 containers

before. Remember that the **docker ps** command only displays the ongoing container status. While the 2 containers you made had not been running, you just made a container. If you want to display all container statuses, use the command below:

```
docker ps -a
```

```
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS     PORTS     NAMES
2f22437fcb9d   nginx     "/docker-entrypoint.…"   6 minutes ago    Created              webapp2
9fbc044864d7   nginx     "/docker-entrypoint.…"   6 minutes ago    Created              webapp1
sysadmin@ubuntu2404:~$
```

List all container statuses in Docker

6. Turn on the container

To turn on a container, you can use the format:

```
docker container start container_id/container_name
```

Usually, I use container_name instead of container_id because it’s easier to memorize, so I run the following command:

```
docker container start webapp1
```

```
sysadmin@ubuntu2404:~$ docker container start webapp1
webapp1
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAMES
b6b73bdbb4e3   nginx     "/docker-entrypoint.…"   31 minutes ago   Up 5 seconds    80/tcp    webapp1
sysadmin@ubuntu2404:~$
```

Turn on the container

7. Pause the container

You can pause a container with the following format:

```
docker container pause container_id/container_name
```

So, you can use the command below to pause the container:

```
docker container pause webapp1
```

```
sysadmin@ubuntu2404:~$ docker container pause webapp1
webapp1
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                     PORTS     NAMES
b6b73bdbb4e3   nginx     "/docker-entrypoint.…"   31 minutes ago   Up 53 seconds (Paused)     80/tcp    webapp1
sysadmin@ubuntu2404:~$
```

Pause the container

To resume the container, use the following format:

```
docker containers unpause container_id/container_name
```

You can use the command below to resume the container:

```
docker container unpause webapp1
```

```
sysadmin@ubuntu2404:~$ docker container unpause webapp1
webapp1
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS              PORTS     NAMES
b6b73bdbb4e3   nginx     "/docker-entrypoint.…"   32 minutes ago   Up About a minute   80/tcp    webapp1
sysadmin@ubuntu2404:~$
```

Resume the container

8. Run a container with a single command

As explained above, if you want to run a container, you have to download the image first, create a container, and turn on the container (see numbers 2, 4, and 6). There is a command that can summarize the three commands above, using the format below:

```
docker run -d --name container_id/container_name image_name:tag
```

where the **-d option** is to run the container in the background. So if you want to run a container with the name webapp2, which contains the nginx application, then use the command below:

```
docker run -d --name webapp3 nginx
```

```
sysadmin@ubuntu2404:~$ docker run -d --name webapp3 nginx
d03087e2d12daf32676144237aef42dfb810568cbf36dadcf0387439ad1c679b
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAMES
d03087e2d12d   nginx     "/docker-entrypoint.…"   6 seconds ago    Up 5 seconds    80/tcp    webapp3
2f22437fcb9d   nginx     "/docker-entrypoint.…"   13 minutes ago   Created                   webapp2
9fbc044864d7   nginx     "/docker-entrypoint.…"   13 minutes ago   Up 12 seconds   80/tcp    webapp1
sysadmin@ubuntu2404:~$
```

Run the container

9. Display the size of Docker

To display how large Docker is installed on your server, use the command below:

```
docker system df
```

```
sysadmin@docker:~$ docker system df
TYPE            TOTAL     ACTIVE    SIZE      RECLAIMABLE
Images          2         1         234.3MB   117MB (49%)
Containers      2         2         2.19kB    0B (0%)
Local Volumes   0         0         0B        0B
Build Cache     0         0         0B        0B
sysadmin@docker:~$
```

Display the size of Docker

To display a Docker size in detail, use the command below:

```
docker system df -v
```

```
sysadmin@docker:~$ docker system df -v
Images space usage:

REPOSITORY   TAG       IMAGE ID       CREATED        SIZE      SHARED SIZE   UNIQUE SIZE   CONTAINERS
nginx        latest    4cad75abc83d   2 months ago   192MB     74.83MB       117.2MB       2
redis        latest    65750d044ac8   3 months ago   117MB     74.83MB       42.21MB       0

Containers space usage:

CONTAINER ID   IMAGE     COMMAND                  LOCAL VOLUMES   SIZE      CREATED          STATUS          NAMES
d8982eca0840   nginx     "/docker-entrypoint.…"   0               1.09kB    4 minutes ago    Up 4 minutes    webapp1
8d609d92bcc7   nginx     "/docker-entrypoint.…"   0               1.09kB    11 minutes ago   Up 11 minutes   nginx

Local Volumes space usage:

VOLUME NAME   LINKS     SIZE

Build cache usage: 0B

CACHE ID   CACHE TYPE   SIZE      CREATED   LAST USED   USAGE     SHARED
sysadmin@docker:~$
```

Display the size of the Docker

10. Display logs

To display logs of the running container to check something, follow the format below:

```
docker container logs container_id/container_name
```

So, run the command below to check the logs of your container:

```
docker container logs webapp1
```

```
sysadmin@ubuntu2404:~$ docker container logs webapp1
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/03/13 15:35:13 [notice] 1#1: using the "epoll" event method
2025/03/13 15:35:13 [notice] 1#1: nginx/1.27.4
2025/03/13 15:35:13 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/03/13 15:35:13 [notice] 1#1: OS: Linux 6.8.0-55-generic
2025/03/13 15:35:13 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/03/13 15:35:13 [notice] 1#1: start worker processes
2025/03/13 15:35:13 [notice] 1#1: start worker process 29
2025/03/13 15:35:13 [notice] 1#1: start worker process 30
sysadmin@ubuntu2404:~$
```

Display logs in the container

If you want to display real-time logs of the container, give an option **-f** like in the below command:

```
docker container logs -f webapp1
```

Press **Ctrl-C** to exit the log.

```
sysadmin@ubuntu2404:~$ docker container logs -f webapp1
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/03/13 15:35:13 [notice] 1#1: using the "epoll" event method
2025/03/13 15:35:13 [notice] 1#1: nginx/1.27.4
2025/03/13 15:35:13 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/03/13 15:35:13 [notice] 1#1: OS: Linux 6.8.0-55-generic
2025/03/13 15:35:13 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/03/13 15:35:13 [notice] 1#1: start worker processes
2025/03/13 15:35:13 [notice] 1#1: start worker process 29
2025/03/13 15:35:13 [notice] 1#1: start worker process 30
^Ccontext canceled
sysadmin@ubuntu2404:~$
```

Display real-time logs in the container

11. Inspect the container

To display detailed information about a container, use the following format:

```
docker inspect container_name/container_id
```

So, if you want to see the detailed information about the container that you created before, use the command below:

```
docker inspect webapp1
```

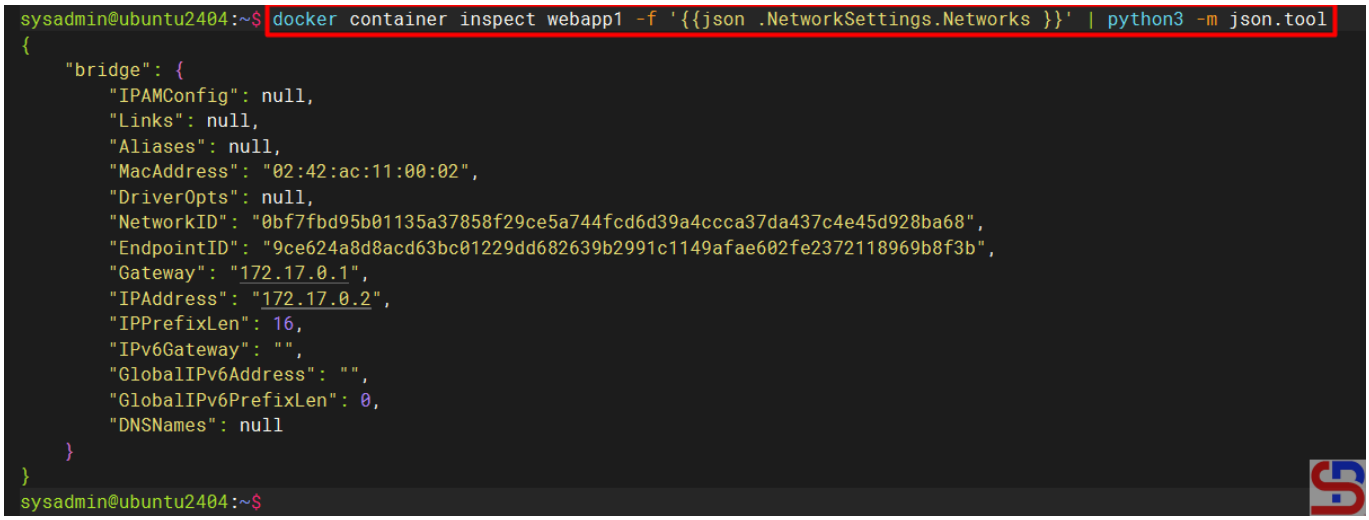
Inspect the container

If you only want to display specific items when running the inspect command, use the following format:

```
docker container inspect container_name/container_id -f '{{json .the item_you_want_to_display<.sub_item> }}' | python -m json.tool
```

So if you want to display only the network section when using the Docker inspect command, use the command below:

```
docker container inspect webapp1 -f '{{json .NetworkSettings.Networks }}' | python3 -m json.tool
```



Inspect the network of the container only

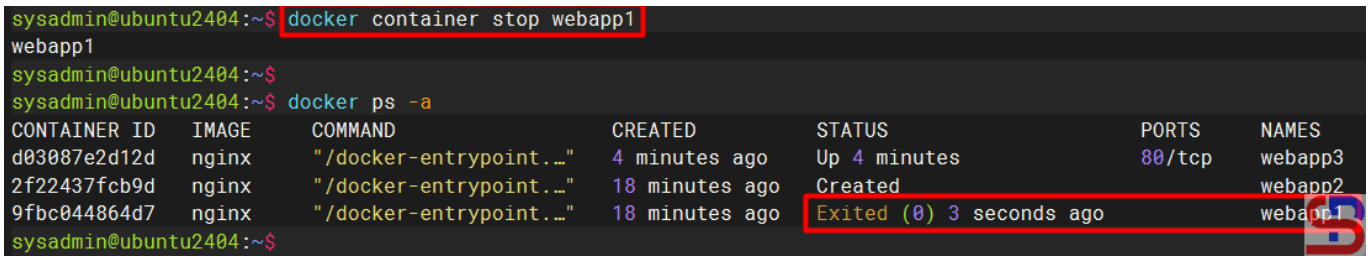
12. Stop the container(s)

To stop the container, use the format below:

```
docker container stop container_id/container_name
```

For example, if I want to stop my container, then use the command below:

```
docker container stop webapp1
```



Stop the container

You can stop all the containers running with the below command:

```
docker stop webapp1 webapp2
```

```
sysadmin@ubuntu2404:~$ docker container stop webapp1 webapp3
webapp1
webapp3
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
d03087e2d12d   nginx     "/docker-entrypoint.…"   6 minutes ago    Exited (0) 3 seconds ago              webapp3
2f22437fcb9d   nginx     "/docker-entrypoint.…"   20 minutes ago   Created                               webapp2
9fbc044864d7   nginx     "/docker-entrypoint.…"   20 minutes ago   Exited (0) 3 seconds ago              webapp1
sysadmin@ubuntu2404:~$
```

Stop more than one container

Or use the below command to stop all the running containers:

```
docker kill $(docker ps -q)
```

```
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAMES
d03087e2d12d   nginx     "/docker-entrypoint.…"   9 minutes ago    Up 4 seconds    80/tcp    webapp3
2f22437fcb9d   nginx     "/docker-entrypoint.…"   23 minutes ago   Created                   webapp2
9fbc044864d7   nginx     "/docker-entrypoint.…"   23 minutes ago   Up 4 seconds    80/tcp    webapp1
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker kill $(docker ps -q)
d03087e2d12d
9fbc044864d7
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                       PORTS     NAMES
d03087e2d12d   nginx     "/docker-entrypoint.…"   9 minutes ago    Exited (137) 3 seconds ago             webapp3
2f22437fcb9d   nginx     "/docker-entrypoint.…"   23 minutes ago   Created                                webapp2
9fbc044864d7   nginx     "/docker-entrypoint.…"   23 minutes ago   Exited (137) 3 seconds ago             webapp1
sysadmin@ubuntu2404:~$
```

Stop all running containers

13. Remove the container(s)

Before you remove the container, you have to **stop the container first**. To delete a container that's already turned off, use the format below:

```
docker container rm container_id/container_name
```

Run the command below to remove the container:

```
docker container rm webapp1
```

```
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
d03087e2d12d   nginx     "/docker-entrypoint.…"   11 minutes ago   Exited (137) 2 minutes ago            webapp3
2f22437fcb9d   nginx     "/docker-entrypoint.…"   25 minutes ago   Created                               webapp2
9fbc044864d7   nginx     "/docker-entrypoint.…"   25 minutes ago   Exited (137) 2 minutes ago            webapp1
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker container rm webapp1
webapp1
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
d03087e2d12d   nginx     "/docker-entrypoint.…"   12 minutes ago   Exited (137) 2 minutes ago            webapp3
2f22437fcb9d   nginx     "/docker-entrypoint.…"   25 minutes ago   Created                               webapp2
sysadmin@ubuntu2404:~$
```

Delete the container

By default, you **can't remove a container if the container is still running**. You can use the command below to delete the container even if the container is still running, but it is not recommended:

```
docker container rm -f webapp2
```

```
sysadmin@ubuntu2404:~$ docker container start webapp2
webapp2
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker container rm webapp2
Error response from daemon: cannot remove container "/webapp2": container is running: stop the container before removing or force remove
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker container rm -f webapp2
webapp2
sysadmin@ubuntu2404:~$
```

Force delete the running container

If you have a lot of containers that are no longer used and you don't want to delete them one by one, you can use the command below to delete all the unused containers:

```
docker rm $(docker ps -a -q)
```

```
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
ba79cff3e0bf   nginx     "/docker-entrypoint.…"   21 seconds ago   Exited (0) 3 seconds ago              webapp2
2edbd2c033d8   nginx     "/docker-entrypoint.…"   26 seconds ago   Exited (0) 3 seconds ago              webapp1
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker rm $(docker ps -a -q)
ba79cff3e0bf
2edbd2c033d8
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
sysadmin@ubuntu2404:~$
```

Delete all the stop containers

You can also use the command below to delete all the stop containers:

```
docker container prune
```

```
sysadmin@docker:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS                     PORTS     NAMES
2fd039880269   mysql     "docker-entrypoint.s…"   5 hours ago    Exited (137) 2 hours ago             db-mysql
2a4eadafffcd   nginx     "/docker-entrypoint.…"   28 hours ago   Exited (0) 2 hours ago               webapp1
e6d61413d2af   nginx     "/docker-entrypoint.…"   29 hours ago   Exited (0) 2 hours ago               nginx
sysadmin@docker:~$
sysadmin@docker:~$ docker container prune
WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] y
Deleted Containers:
2fd039880269153802f303435bf9a197fd1aefed5b96c5df0fe2a8e291266cb3
2a4eadafffcd4899dce3201f8e110489e77d5c0f6d4a9bac8af91f48a06adf35
e6d61413d2afa872ee797369775b0bbaf39d433b29c13c03538383f3bc228888

Total reclaimed space: 22.47MB
sysadmin@docker:~$
sysadmin@docker:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
sysadmin@docker:~$
```

Delete the stop containers using the prune command

14. Delete the image(s)

To delete the Docker image that you have already downloaded, use the format below:

```
docker image rm image_name
```

Run the image below if you want to delete the nginx image:

```
docker image rm nginx
```

```
sysadmin@ubuntu2404:~$ docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
nginx        latest    b52e0b094bc0   5 weeks ago   192MB
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker rmi nginx
Untagged: nginx:latest
Untagged: nginx@sha256:9d6b58feebd2dbd3c56ab5853333d627cc6e281011cfd6050fa4bcf2072c9496
Deleted: sha256:b52e0b094bc0e26c9eddc9e4ab7a64ce0033c3360d8b7ad4ff4132c4e03e8f7b
Deleted: sha256:3c8b88c16794e3082397557e5482f5a04a6c295cec37919c65c234e1a3645e80
Deleted: sha256:d5c83383666c732fcb30d7e25c74c2e0884c262f2e497cc9f2844870980311d8
Deleted: sha256:d62b6301e685a7cdc3bb3b1508a959e4710a707ea2f680f848c19a9ad74ac6a7
Deleted: sha256:d443654bda4a04f31ba6bd39bed82a053a17f2974b401fef552e4e88d6546db8
Deleted: sha256:129409d5d363e5d5af273f0b2a90237f708ed9972f8d58a4dbcd17f1abbabe21
Deleted: sha256:a3a2912e392a24d8c7dde076a3778c6eded8839660963ac2084e051eb6931c13
Deleted: sha256:5f1ee22ffb5e68686db3dcb6584eb1c73b5570615b0f14fabb070b96117e351d
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker image ls
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
sysadmin@ubuntu2404:~$
```

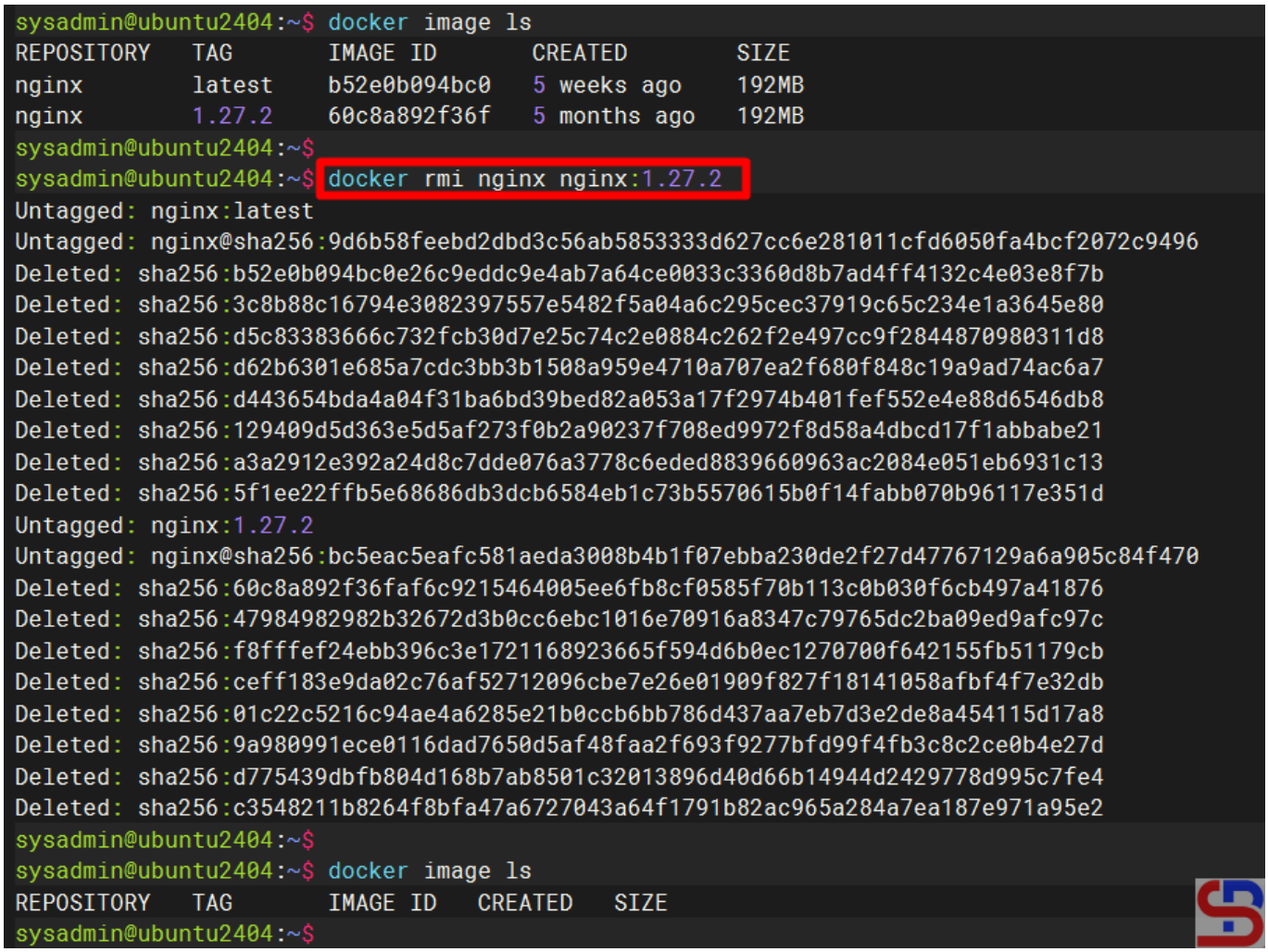
Delete the image

However, you must know that you **can't delete the Docker image if the image is still running in the container**. So you must remove the container first before you delete the image. If you want to delete multiple Docker images, use the following format:

```
docker image rm image_name1 image_name2 ...
```

So if you want to delete the nginx image and nginx:1.27.2 at once, then use the command below:

```
docker image rm nginx nginx:1.27.2
```



Delete more than one image

And if you want to delete all the images, you can use the command below:

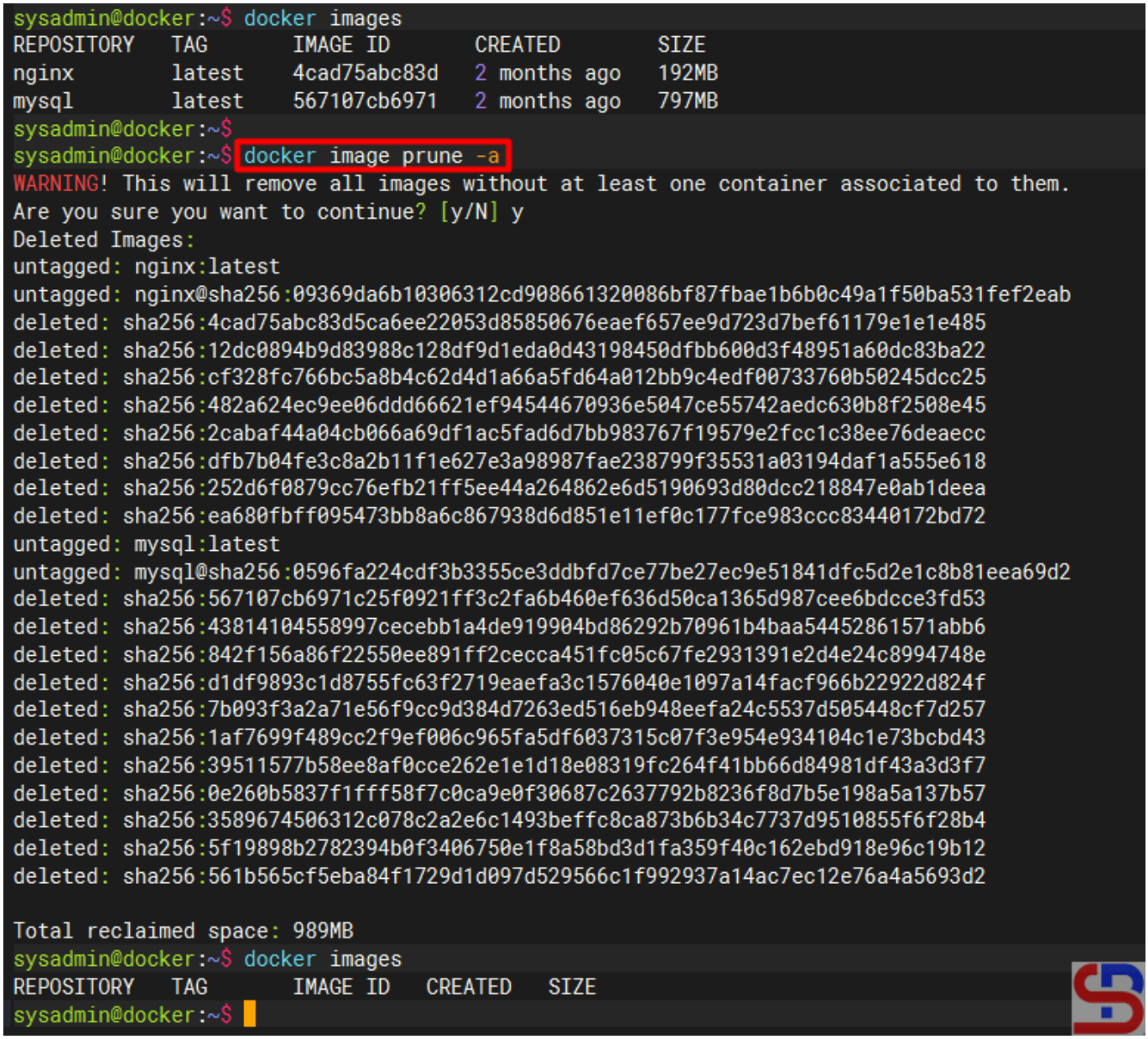
```
docker rmi $(docker images -a -q)
```

```
sysadmin@ubuntu2404:~$ docker image ls
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
nginx        latest    b52e0b094bc0   5 weeks ago    192MB
nginx        1.27.2    60c8a892f36f   5 months ago   192MB
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker rmi $(docker images -a -q)
Untagged: nginx:latest
Untagged: nginx@sha256:9d6b58feebd2dbd3c56ab5853333d627cc6e281011cfd6050fa4bcf2072c9496
Deleted: sha256:b52e0b094bc0e26c9eddc9e4ab7a64ce0033c3360d8b7ad4ff4132c4e03e8f7b
Deleted: sha256:3c8b88c16794e3082397557e5482f5a04a6c295cec37919c65c234e1a3645e80
Deleted: sha256:d5c83383666c732fcb30d7e25c74c2e0884c262f2e497cc9f2844870980311d8
Deleted: sha256:d62b6301e685a7cdc3bb3b1508a959e4710a707ea2f680f848c19a9ad74ac6a7
Deleted: sha256:d443654bda4a04f31ba6bd39bed82a053a17f2974b401fef552e4e88d6546db8
Deleted: sha256:129409d5d363e5d5af273f0b2a90237f708ed9972f8d58a4dbcd17f1abbabe21
Deleted: sha256:a3a2912e392a24d8c7dde076a3778c6eded8839660963ac2084e051eb6931c13
Deleted: sha256:5f1ee22ffb5e68686db3dcb6584eb1c73b5570615b0f14fabb070b96117e351d
Untagged: nginx:1.27.2
Untagged: nginx@sha256:bc5eac5eafc581aeda3008b4b1f07ebba230de2f27d47767129a6a905c84f470
Deleted: sha256:60c8a892f36faf6c9215464005ee6fb8cf0585f70b113c0b030f6cb497a41876
Deleted: sha256:47984982982b32672d3b0cc6ebc1016e70916a8347c79765dc2ba09ed9afc97c
Deleted: sha256:f8fffef24ebb396c3e1721168923665f594d6b0ec1270700f642155fb51179cb
Deleted: sha256:ceff183e9da02c76af52712096cbe7e26e01909f827f18141058afbf4f7e32db
Deleted: sha256:01c22c5216c94ae4a6285e21b0ccb6bb786d437aa7eb7d3e2de8a454115d17a8
Deleted: sha256:9a980991ece0116dad7650d5af48faa2f693f9277bfd99f4fb3c8c2ce0b4e27d
Deleted: sha256:d775439dbfb804d168b7ab8501c32013896d40d66b14944d2429778d995c7fe4
Deleted: sha256:c3548211b8264f8bfa47a6727043a64f1791b82ac965a284a7ea187e971a95e2
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ docker image ls
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
sysadmin@ubuntu2404:~$
```

Delete all the images

Or, you can use the command below to remove the unused images:

```
docker image prune -a
```



Delete all the unused images using the prune command

Note

If you forget or don’t know what command to use in Docker, use the following command:

```
docker --help
```

```
sysadmin@ubuntu2404:~$ docker --help

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
  exec        Execute a command in a running container
  ps          List containers
  build       Build an image from a Dockerfile
  pull        Download an image from a registry
  push        Upload an image to a registry
  images      List images
  login       Authenticate to a registry
  logout      Log out from a registry
  search      Search Docker Hub for images
  version     Show the Docker version information
  info        Display system-wide information

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx
  compose*    Docker Compose
  container   Manage containers
  context     Manage contexts
  image       Manage images
  manifest    Manage Docker image manifests and manifest lists
  network     Manage networks
```

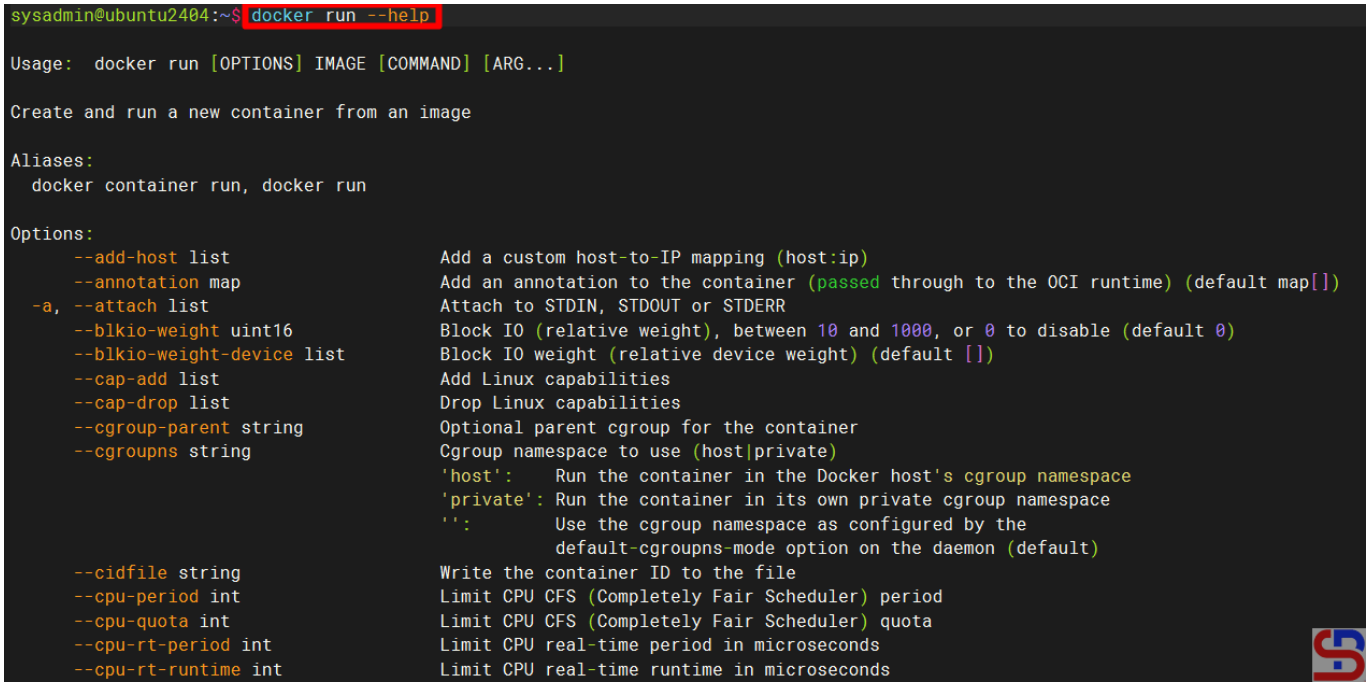
Using the docker help command

After that, if you want to know the options in the Docker command, then use the following format:

```
docker command --help
```

For example, if you want to know the options of the run command in Docker, then type the command below:

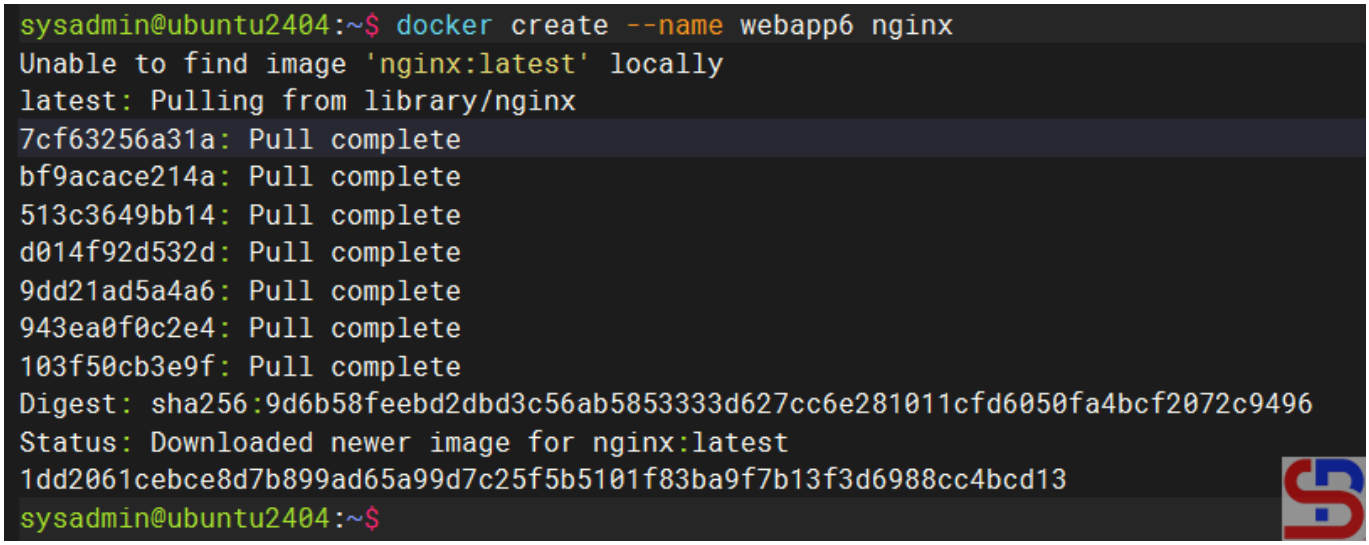
```
docker run --help
```



Using the docker run help command

You can shorten all the **Docker container** commands to just the **docker** command to shorten the typing time. For example, if you want to create a container, you can use the command:

```
docker create --name webapp6 nginx
```



Using the docker create command

References

geeksforgeeks.org
mygreatlearning.com
youtube.com
youtube.dimas-maryanto.com