Disabling unnecessary services for faster boot Linux

More and more operating systems become 'cumbersome' with lots of services running in the background. Although they may not use too much CPU resources, they increase boot time and RAM usage.

More and more operating systems become 'cumbersome' with lots of services running in the background. Although they may not use too much CPU resources, they increase boot time and RAM usage.

Some Linux distributions include everything, with the aim of making user use easier. This makes things like file sharing or printing increasingly creative. Unfortunately, the only way to do it is to create hundreds of utilities for people to find what they need.

Fortunately, open source operating systems give you the ability to disable or delete anything that isn't needed. The first option, how to disable services - is what will be covered in this article. When disabled instead of removing components, the risk of everything being permanently broken is less likely to occur. And you can reactivate the service if you find something useful has stopped working.

Make Linux boot faster by disabling unnecessary services

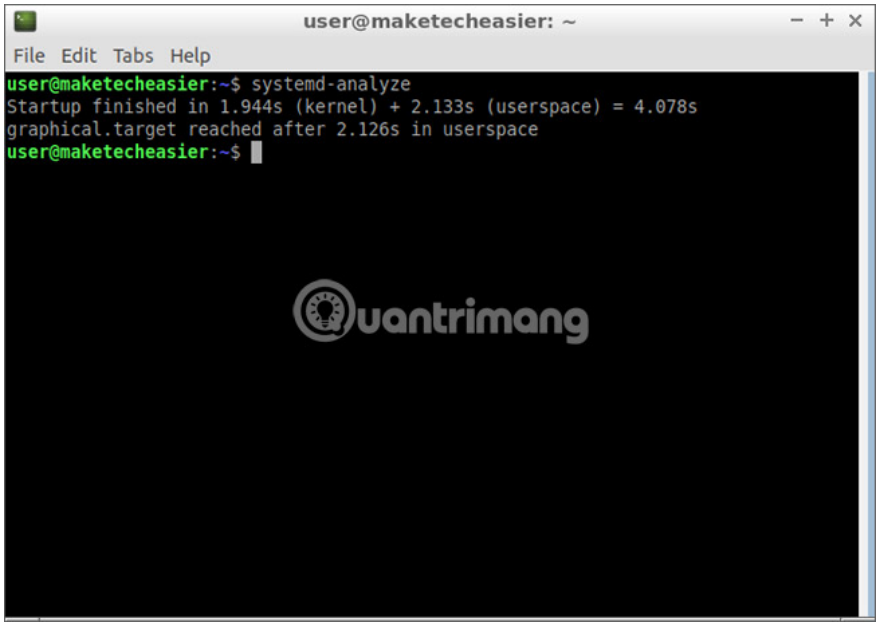
1. Analyze the time needed to load a service
2. Use systemctl to disable unnecessary services
3. Use systemctl to hide a service

Analyze the time needed to load a service

Most Linux-based operating systems have switched to Systemd. Among Systemd's utilities, there is a program that allows users to analyze the system boot speed. Specifically, it shows the user the total time needed to boot and the time required to load each service. Note that some services load in parallel. So if the first service takes 2 seconds and another service needs 3 seconds to load, it doesn't mean it takes 5 seconds to load those 2 services.

Open a terminal and enter this command:

```
systemd-analyze
```



This shows how long it takes to load the basic system services and the Linux kernel, not counting the boot time of the graphical interface. In other words, it is a necessary time, since the kernel is loaded until the user reaches the Linux control panel, as described below:

If you have not installed a graphical interface, this is what you will see on your screen.

However, you can also see the time that the graphical interface requires initialization with the following command:

```
systemd-analyze critical-chain graphical.target
```

```
user@maketecheasier: ~
File  Edit  Tabs  Help
user@maketecheasier:~$ systemd-analyze critical-chain graphical.target
The time after the unit is active or started is printed after the "@" character.
The time the unit takes to start is printed after the "+" character.

graphical.target @2.126s
└multi-user.target @2.126s
  └snapd.seeded.service @2.102s +23ms
    └snapd.service @805ms +1.295s
      └basic.target @640ms
        └sockets.target @640ms
          └snapd.socket @639ms +1ms
            └sysinit.target @636ms
              └apparmor.service @311ms +324ms
                └local-fs.target @307ms
                  └run-user-1000.mount @1.986s
                    └local-fs-pre.target @284ms
                      └keyboard-setup.service @169ms +114ms
                        └systemd-journald.socket @167ms
                          └system.slice @167ms
                            └-.slice @165ms
user@maketecheasier:~$
```

This command shows that the graphical user interface is loaded in 2,126 seconds, not counting the time needed to load desktop utilities. To enable or disable those utilities, launch the desktop environment startup manager.

Finally, perhaps the most useful command to help Linux boot faster is:

```
systemd-analyze blame
```

```
user@maketecheasier: ~
File  Edit  Tabs  Help
          1.927s dev-vda1.device
          1.295s snapd.service
           638ms dev-loop0.device
           628ms dev-loop1.device
           614ms dev-loop2.device
           598ms dev-loop4.device
           598ms dev-loop3.device
           589ms dev-loop5.device
           574ms dev-loop7.device
           570ms dev-loop6.device
           561ms dev-loop8.device
           548ms dev-loop9.device
           539ms dev-loop10.device
           524ms dev-loop11.device
           521ms udisks2.service
           516ms dev-loop15.device
           514ms dev-loop13.device
           512ms dev-loop12.device
           506ms dev-loop14.device
           466ms networkd-dispatcher.service
           391ms NetworkManager-wait-online.service
           367ms NetworkManager.service
           324ms apparmor.service
lines 1-23
```

You can navigate the list with the arrow keys or the PAGE UP and PAGE DOWN keys. Press q to exit.

Use systemctl to disable unnecessary services

As you can see in the previous image, service snapd takes 1.295 seconds to load. On an SSD, this time period is negligible. But on a hard drive, these services will in turn lose certain load times and the total time needed to load them is not small. In addition, on SSDs, many services are loaded in parallel, resulting in amazing results. On the contrary, on a hard drive, parallel service loads are difficult to implement. Readers place on the platter (the platter is a circular disk where the magnetic data stored in the hard drive) must switch from one sector to another, so parallel data reading is almost impossible. .

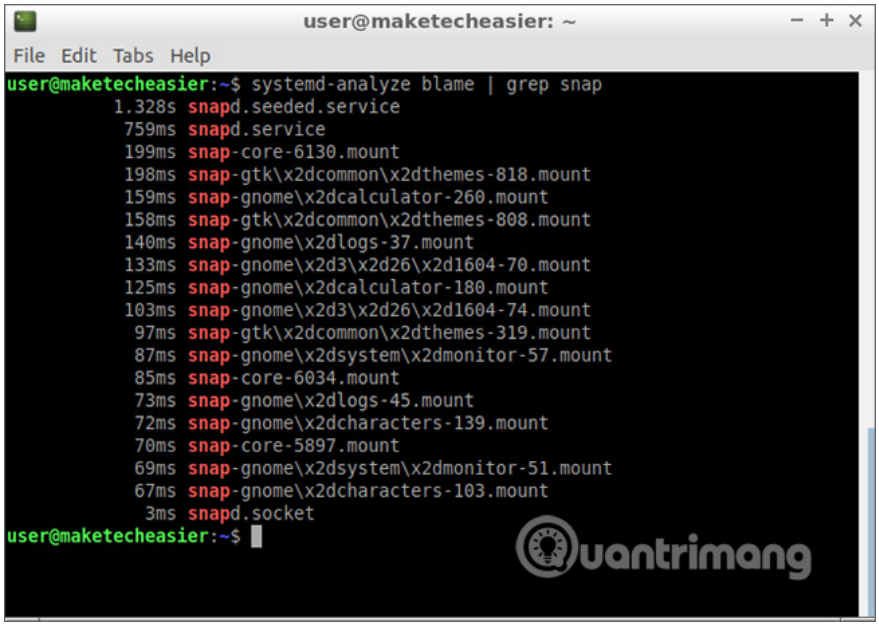
To make it easier to understand, imagine the following: If on SSD there are 5 services, each service needs a second to boot, then the total time needed to load all these services is less than 1.2 seconds. But on a hard drive, with 5 similar services, the total load time will be more than 5 seconds.

Suppose you do not need service snapd, can disable it with this command:

```
sudo systemctl disable snapd.service
```

But, if you reboot, you will see the daemon snap is still running. The reason may be because other dependencies have started it, even if it is disabled. Try to see what could be the cause of this problem:

```
systemd-analyze blame | grep snap
```



snapd.seeded.service and snapd.socket are the culprits here. A child service may require its original service to be launched. When disabling a service, it simply requires that it not automatically start when you boot. But there is an effective method to solve this problem.

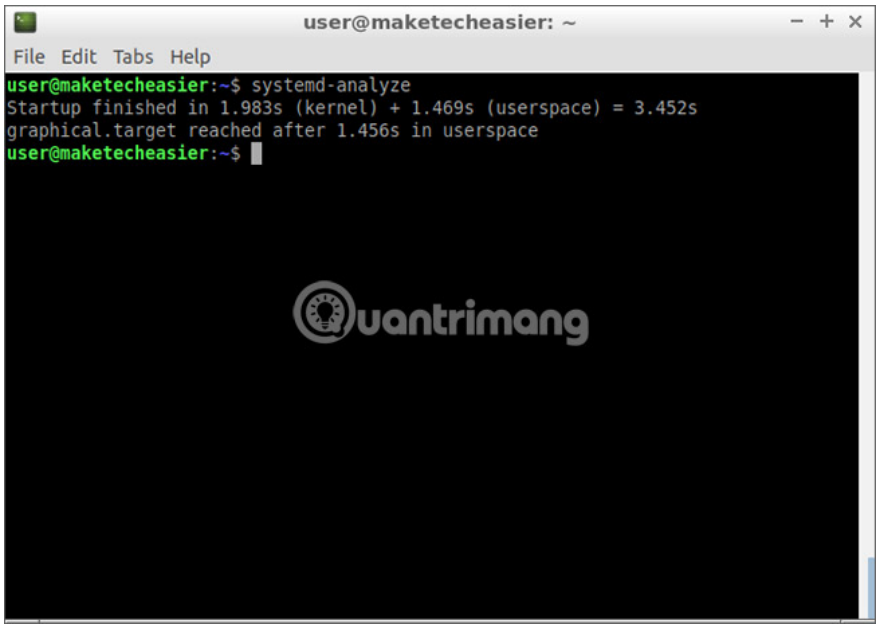
Use systemctl to hide a service

```
sudo systemctl mask snapd.service
```

This basically makes the service file empty so that the application has no way to launch the service.

In most cases, disabling a service is not too complicated. The example below is chosen to show you how to handle more difficult situations. In this case, snapd.seeded.service and snapd.socket should be disabled or hidden.

The following figure shows the boot time has improved.



Boot time has decreased from 4,078 seconds to 0.52 seconds, down to 15%. Not too bad. And this only considers a service disabled on SSD. On a hard drive, it's easy to get much better results.

In this particular example, many services will be disabled with a command like:

```
sudo systemctl mask avahi-daemon.service ModemManager.service thermald.service p
```

If you like a graphical application to manage your services, you can consider Systemd Manager. However, what you can do will be limited to this tool, because the command line offers much more flexibility.

Good luck!

See more:

1. 6 ways to speed up the Ubuntu system
2. 10 ways to help Linux boot faster
3. 10 Linux tools for rescue, restore Linux, Window, Mac

You finished reading the article "**Disabling unnecessary services for faster boot Linux**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.

© 2019 TipsMake.com