Google IT Support Certificate: Course 4: Systems Admin. and IT Infrastructure Services

Week 2: Network and Infrastructure Services

Quiz: Managing Services in Linux

Learning Objectives

- Getting familiar with common system services on Linux
- Starting, stopping, and reloading services on Linux
- Editing service configuration

Introduction

As a system administrator, you'll need to know how to look at the status of a running service, and how to stop, start, and restart running services. You'll also need to know how to configure services and fix problems you may encounter while running them. In this lab, we'll focus on how to do this actions on Linux.

What you'll do

You'll get to have a look at the list of services that are running on the machine, practice stopping and starting services such as rsyslog or cups as well as querying their status.
You'll also fix a problem in a service that is failing to start, and edit the configuration of another service.

Linux commands reminder

In this lab, we'll use a number of Linux commands that were already explained during Course 3. Here is a reminder of what these commands do:

- `sudo <command>`: executes a command with administrator rights

- `ls <directory>`: lists the files in a directory
- `mv <old_name> <new_name>`: moves or renames a file from the old name to the new name
- `tail <file>`: shows the last lines of a file
- `cat <file>`: prints the whole contents of a file
- `grep <pattern> <file>`: filters the text of a file according to the pattern
- `less <file>`: lets you browse a file

Additionally, you can combine these commands using the `|` sign. For example:

```
sudo cat /var/log/syslog | grep error | tail
```

Copied!

content_copy

will first print the output of `/var/log/syslog`, then keep only the lines that say "error" and then the last 10 lines of that output.

We will also present a number of new commands such as `service`, `logger`. We will briefly explain what these commands do when they are shown. Remember that you can always read the manual page using `man <command_name>` to learn more about a command.

While you can copy and paste the commands that are presented in this lab, we recommend typing them out manually, to help with understanding and remembering them.

Listing system services

Let's look at the services that are installed in the machine. In order to do this, we will use the `service` command.

If you run `service` with parameters `--status-all`, it lists the state of services controlled by `System V`.

If you are interested in seeing only the services that are running, you can use the following command:

```
sudo service --status-all
```

Copied!

content_copy

Output:

```
[ - ]  avahi-daemon
 [ - ]  cron
 [ - ]  cups
 [ - ]  cups-browsed
 [ - ]  dbus
 [ - ]  exim4
 [ ? ]  hwclock.sh
 [ - ]  procps
 [ + ]  rsyslog
 [ - ]  saned
 [ + ]  ssh
 [ - ]  sudo
 [ + ]  udev
```

Here you will see the following notations with respect to the services.

- `+`: Service is active/running
- `-`: Service is inactive/stopped
- `?`: Can't determine whether service is active or not

Stopping and starting services

Alright, now that we've listed the services let's practice stopping and starting some of them. The first service that we are going to stop is the `rsyslog` service. This service is in charge of writing content to the log files, as in `/var/log/syslog`, `/var/log/kern.log`, `/var/log/auth.log` and others. Processes that generate output will send that output to the rsyslog service and the service will write it to the corresponding log files depending on how the system was configured.

Let's first start by checking the status of the service. We do this by using the `service` command with the `status` action:

```
sudo service rsyslog status
```

Copied!

content_copy

Output:

```
rsyslogd is running.
```

This is showing us a lot of information about the service: it's loaded (which means that the OS has the information about the service in memory), it's enabled (which means that it will start automatically on boot), it's active and running. It also tells us where to find some documentation about the service and more. Finally, it shows us the last log lines that this service generated.

We can see this service in action by using the `logger` command:

```
logger This is a test log entry
```

Copied!

content_copy

The logger command will send the text to the rsyslog service and the service will then write it into `/var/log/syslog`. We can check that this is the case by looking at the last lines in `/var/log/syslog`.

```
sudo tail -1 /var/log/syslog
```

Copied!

content_copy

Output:

```
Aug 13 09:02:52 93edb8c3a3ac student: This is a test log entry
```

Let's now go ahead and stop the rsyslog service:

```
sudo service rsyslog stop
```

Copied!

content_copy

We need to execute the command with sudo, because while all users can query the status of services, only users with administrator rights can stop or start services.

Output:

```
Stopping enhanced syslogd: rsyslogd.
```

To see the current state, we can query the status of the service again:

```
sudo service rsyslog status
```

Copied!

content_copy

Output:

```
rsyslogd is not running ... failed!
```

We see that the service is now stopped. We can also see what the command logged to `/var/log/syslog` when finishing:

```
sudo tail -5 /var/log/syslog
```

Copied!

content_copy

Output:

```
Aug 13 09:00:49 93edb8c3a3ac rsyslogd: imklog: cannot open kernel log
(/proc/kmsg): Operation not permitted.
Aug 13 09:00:49 93edb8c3a3ac rsyslogd: activation of module imklog failed
[v8.1901.0 try https://www.rsyslog.com/e/2145 ]
Aug 13 09:00:49 93edb8c3a3ac rsyslogd:  [origin software="rsyslogd"
swVersion="8.1901.0" x-pid="984" x-info="https://www.rsyslog.com"] start
Aug 13 09:01:28 93edb8c3a3ac example: This is an example log line
Aug 13 09:02:52 93edb8c3a3ac student: This is a test log entry
```

In the last line, we see that the rsyslog service has exited and is no longer running.

We can try sending text with our logger command again:

```
logger This is another test log entry
```

Copied!

content_copy

And then check that the contents of `/var/log/syslog:`

```
sudo tail /var/log/syslog
```

Copied!

content_copy

Output:

```
Aug 13 09:00:49 93edb8c3a3ac rsyslogd: imklog: cannot open kernel log
(/proc/kmsg): Operation not permitted.
Aug 13 09:00:49 93edb8c3a3ac rsyslogd: activation of module imklog failed
[v8.1901.0 try https://www.rsyslog.com/e/2145 ]
Aug 13 09:00:49 93edb8c3a3ac rsyslogd:  [origin software="rsyslogd"
swVersion="8.1901.0" x-pid="984" x-info="https://www.rsyslog.com"] start
Aug 13 09:01:28 93edb8c3a3ac example: This is an example log line
Aug 13 09:02:52 93edb8c3a3ac student: This is a test log entry
```

We can see that nothing was logged, because rsyslog wasn't running.

Let's start it back up:

```
sudo service rsyslog start
```

Copied!

content_copy

Output:

```
Starting enhanced syslogd: rsyslogdrsyslogd: imklog: cannot open kernel
log (/proc/kmsg): Operation not permitted.
rsyslogd: activation of module imklog failed [v8.1901.0 try
https://www.rsyslog.com/e/2145 ]
.
```

```
sudo service rsyslog status
```

Copied!

content_copy

Output:

```
rsyslogd is running.
```

And see that it's running again. Let's try our logger command one more time:

```
logger This is another test log entry
```

Copied!

content_copy

And then check that the contents of `/var/log/syslog:`

```
sudo tail /var/log/syslog
```

Copied!

content_copy

Output:

```
Aug 13 09:00:49 93edb8c3a3ac rsyslogd: imklog: cannot open kernel log
(/proc/kmsg): Operation not permitted.
```

```
Aug 13 09:00:49 93edb8c3a3ac rsyslogd: activation of module imklog failed
[v8.1901.0 try https://www.rsyslog.com/e/2145 ]
Aug 13 09:00:49 93edb8c3a3ac rsyslogd:  [origin software="rsyslogd"
swVersion="8.1901.0" x-pid="984" x-info="https://www.rsyslog.com"] start
Aug 13 09:01:28 93edb8c3a3ac example: This is an example log line
Aug 13 09:02:52 93edb8c3a3ac student: This is a test log entry
Aug 13 09:06:17 93edb8c3a3ac rsyslogd: imklog: cannot open kernel log
(/proc/kmsg): Operation not permitted.
Aug 13 09:06:17 93edb8c3a3ac rsyslogd: activation of module imklog failed
[v8.1901.0 try https://www.rsyslog.com/e/2145 ]
Aug 13 09:06:17 93edb8c3a3ac rsyslogd:  [origin software="rsyslogd"
swVersion="8.1901.0" x-pid="3326" x-info="https://www.rsyslog.com"] start
Aug 13 09:07:47 93edb8c3a3ac student: This is another test log entry
```

The message is now there!

Click *Check my progress* to verify the objective.

Stop and Start rsyslog Service

Check my progress

Fixing a failing service

In order to list the state of services controlled by `System V`, you can use the following command:

```
sudo service --status-all
```

Copied!

content_copy

Output:

```
[ - ]  avahi-daemon
 [ - ]  cron
 [ - ]  cups
 [ - ]  cups-browsed
 [ - ]  dbus
 [ - ]  exim4
 [ ? ]  hwclock.sh
 [ - ]  procps
 [ + ]  rsyslog
 [ - ]  saned
 [ + ]  ssh
 [ - ]  sudo
 [ + ]  udev
```

Here you will find `-` with the `cups` service, which means it is inactive/stopped. This is the service used to manage printers on Linux systems. We can get more information about this service by checking the status:

```
sudo service cups status
```

Copied!

content_copy

Output:

```
cupsd is not running ... failed!
```

We see here that the `cups` service is in a failed state. So, let's look at the contents of that directory:

```
sudo ls -l /etc/cups
```

Copied!

content_copy

Output:

```
total 64
-rw-r--r-- 1 root root 27303 Apr 10  2019 cups-browsed.conf
-rw-r--r-- 1 root root  2923 Nov 28  2020 cups-files.conf
-rw-r--r-- 1 root root  6400 Aug 13 09:01 cupsd.conf.old
drwxr-xr-x 2 root root  4096 Nov 28  2020 interfaces
drwxr-xr-x 2 root root  4096 Nov 28  2020 ppd
-rw-r--r-- 1 root root   240 Aug 13 09:01 raw.convs
-rw-r--r-- 1 root root   211 Aug 13 09:01 raw.types
-rw-r--r-- 1 root root   142 Nov 28  2020 snmp.conf
drwxr-xr-x 2 root root  4096 Nov 28  2020 ssl
```

There's no `cupsd.conf`, but there is `cupsd.conf.old`. Apparently the configuration file was deleted. Good thing we kept a copy! Let's move that file so that cups can find it and start successfully:

```
sudo mv /etc/cups/cupsd.conf.old /etc/cups/cupsd.conf
```

Copied!

content_copy

As with the other commands, we get no output after executing this. We can run `ls` again to see that the file was renamed correctly:

```
sudo ls -l /etc/cups
```

Copied!

content_copy

Output:

```
total 64
-rw-r--r-- 1 root root 27303 Apr 10  2019 cups-browsed.conf
-rw-r--r-- 1 root root  2923 Nov 28  2020 cups-files.conf
-rw-r--r-- 1 root root  6400 Aug 13 09:01 cupsd.conf
drwxr-xr-x 2 root root  4096 Nov 28  2020 interfaces
drwxr-xr-x 2 root root  4096 Nov 28  2020 ppd
-rw-r--r-- 1 root root   240 Aug 13 09:01 raw.convs
-rw-r--r-- 1 root root   211 Aug 13 09:01 raw.types
-rw-r--r-- 1 root root   142 Nov 28  2020 snmp.conf
drwxr-xr-x 2 root root  4096 Nov 28  2020 ssl
```

Now that the file was renamed successfully, we can start cups:

```
sudo service cups start
```

Copied!

content_copy

And then check the status:

```
sudo service cups status
```

Copied!

content_copy

Output:

```
cupsd is running.
```

We've fixed it!

Click *Check my progress* to verify the objective.

Fix Cups Service

Check my progress

Restarting services

Let's go back to the cups service that we just fixed. The logs generated by cups are written into the `/var/log/cups` directory. We can see the contents of the directory using the ls command:

```
sudo ls -l /var/log/cups
```

Copied!

content_copy

Output:

```
total 0
```

The error log file isn't yet created. That's expected because by default cups will only write warning or error messages into that file. If you want cups to log debug messages into that file, you'll need to change the LogLevel parameter in the configuration file. Let's do that.

Let's edit `/etc/cups/cupsd.conf` using the nano editor.

```
sudo nano /etc/cups/cupsd.conf
```

Copied!

content_copy

In one of the first lines of the file you'll see there's a line that says `LogLevel warn`. We want to replace **warn** with **debug**:

```
#
# Configuration file for the CUPS scheduler.  See "man cupsd.conf" for a
# complete description of this file.
#

# Log general information in error_log - change "warn" to "debug"
# for troubleshooting...
LogLevel debug
PageLogFormat

# Deactivate CUPS' internal logrotating, as we provide a better one,
especially
# LogLevel debug2 gets usable now
MaxLogSize 0
```

Once you've done this, press **"Ctrl-X"** to exit the editor. It will ask you if you want to save your changes, press **"Y"** for yes and then **enter** at the filename prompt.

If we now restart cups, the service will notice the change to its configuration file. You could do this by using `sudo service cups stop` and `sudo service cups start`, but there is a shortcut.

```
sudo service cups restart
```

Copied!

content_copy

Restarting the service command is a handy way of stopping a service and then starting it immediately back up. And once we've done that, we can see that there's now a lot of content in `/var/log/cups/error_log.`

```
sudo cat /var/log/cups/error_log
```

Copied!

content_copy

Output:

```
I [13/Aug/2021:09:12:52 +0000] Listening to 0.0.0.0:631 (IPv4)
I [13/Aug/2021:09:12:52 +0000] Listening to /run/cups/cups.sock (Domain)
I [13/Aug/2021:09:12:52 +0000] Remote access is enabled.
D [13/Aug/2021:09:12:52 +0000] Added auto ServerAlias 93edb8c3a3ac
I [13/Aug/2021:09:12:52 +0000] Loaded configuration file
"/etc/cups/cupsd.conf"
D [13/Aug/2021:09:12:52 +0000] Using keychain "/etc/cups/ssl" for server
name "93edb8c3a3ac".
I [13/Aug/2021:09:12:52 +0000] Using default TempDir of
/var/spool/cups/tmp...
I [13/Aug/2021:09:12:52 +0000] Configured for up to 100 clients.
I [13/Aug/2021:09:12:52 +0000] Allowing up to 100 client connections per
host.
I [13/Aug/2021:09:12:52 +0000] Using policy "default" as the default.
I [13/Aug/2021:09:12:52 +0000] Full reload is required.
I [13/Aug/2021:09:12:52 +0000] Loaded MIME database from
"/usr/share/cups/mime" and "/etc/cups": 78 types, 114 filters...
```

```
I [13/Aug/2021:09:12:52 +0000] Loading job cache file
"/var/cache/cups/job.cache"...
I [13/Aug/2021:09:12:52 +0000] Full reload complete.
D [13/Aug/2021:09:12:52 +0000] cupsdCleanFiles(path="/var/spool/cups/tmp",
pattern="(null)")
I [13/Aug/2021:09:12:52 +0000] Cleaning out old files in
"/var/spool/cups/tmp".
D [13/Aug/2021:09:12:52 +0000] cupsdCleanFiles(path="/var/cache/cups",
pattern="*.ipp")
I [13/Aug/2021:09:12:52 +0000] Cleaning out old files in
"/var/cache/cups".
D [13/Aug/2021:09:12:52 +0000] service_checkin: pid=3588
D [13/Aug/2021:09:12:52 +0000] Creating KeepAlive/PID file
"/run/cups/cupsd.pid".
I [13/Aug/2021:09:12:52 +0000] Listening to 0.0.0.0:631 on fd 4...
I [13/Aug/2021:09:12:52 +0000] Listening to /run/cups/cups.sock on fd 6...
I [13/Aug/2021:09:12:52 +0000] Resuming new connection processing...
D [13/Aug/2021:09:12:52 +0000] cupsdSetBusyState: newbusy="Not busy",
busy="Active clients"
D [13/Aug/2021:09:12:52 +0000] cupsdAddCert: Adding certificate for PID 0
D [13/Aug/2021:09:12:52 +0000] Discarding unused server-started event...
D [13/Aug/2021:09:12:52 +0000] cupsdSetBusyState: newbusy="Not busy",
busy="Not busy"
D [13/Aug/2021:09:12:53 +0000] Report: clients=0
D [13/Aug/2021:09:12:53 +0000] Report: jobs=0
D [13/Aug/2021:09:12:53 +0000] Report: jobs-active=0
D [13/Aug/2021:09:12:53 +0000] Report: printers=0
D [13/Aug/2021:09:12:53 +0000] Report: stringpool-string-count=329
D [13/Aug/2021:09:12:53 +0000] Report: stringpool-alloc-bytes=5112
D [13/Aug/2021:09:12:53 +0000] Report: stringpool-total-bytes=5584
```

Click *Check my progress* to verify the objective.

Change the Log Level for Cups

Check my progress

Reloading Services

Finally, let's look at the reload action. Take this action when you want a service to re-read its configuration without actually doing a full stop and start.

Let's return the cups log level back to its default. One more time, let's edit `/etc/cups/cupsd.conf` using the nano editor.

```
sudo nano /etc/cups/cupsd.conf
```

Copied!

content_copy

Let's change `LogLevel debug`, replacing **debug** with **warn**:

```
#
# Configuration file for the CUPS scheduler.  See "man cupsd.conf" for a
# complete description of this file.
#

# Log general information in error_log - change "warn" to "debug"
# for troubleshooting...
LogLevel warn
PageLogFormat

# Deactivate CUPS' internal logrotating, as we provide a better one,
especially
# LogLevel debug2 gets usable now
MaxLogSize 0
```

Once you've done this, press **"Ctrl-X"** to exit the editor. It will ask you if you want to save your changes, press **"Y"** for yes and then **enter** at the filename prompt.

Once we've done this, we can reload cups:

```
sudo service cups reload
```

Copied!

content_copy

Output:

```
Reloading Common Unix Printing System: cupsd.
```

If you check the status of the service, you'll see that it was not restarted (it's been running since we last restarted it).

```
sudo service cups status
```

Copied!

content_copy

Output:

```
cupsd is running.
```

By using the reload action, we caused the service to re-read its configuration without being stopped at any point.

Click *Check my progress* to verify the objective.

Revert the Log Level for Cups

Check my progress

Conclusion

Congrats! You've successfully listed all the services that are running on the machine, practiced stopping and starting some of these services, and queried their status. You also fixed a problem in the service that was failing to start and edited the configuration of another service.

These are important commands and problem-solving skills that you'll use on a daily basis as a system administrator. Keep it up!