Week 05

IT-230

- Analyzing and Storing Logs
- Time Synchronization
- Tuning System Performance
- Apache Basics

IT-230 · Linux Administration

Start `workstation` and `servera` on your VDI

IT-230 · Linux Administration

Analyzing and Storing Logs

Where the answers are when something goes wrong

Why Look at Logs?

Security

Who accessed what, when, and were they allowed to?

Troubleshooting

What went wrong, and why?

Logs are your best hope of answering

Usually your only hope

Two Mechanisms, Mostly the Same Messages

Some applications write their own files, such as `httpd` to `/var/log/httpd/`

`rsyslog`

Older, and still doing work systemd does not

Writes plain text under `/var/log/`

Kept across reboots

Can ship logs to a central server

`journald`

Part of systemd, so it knows about units

Writes a binary journal

Read the journal with `journalctl`

Not kept across reboots by default

Common Log Files

File	What lands there
`/var/log/messages`	Most syslog messages
`/var/log/secure`	Security and authentication events
`/var/log/maillog`	Mail server messages
`/var/log/cron`	Scheduled job execution
`/var/log/boot.log`	Console messages from system startup

Every one of these is written by `rsyslog`, following rules you can read and change

IT-230 · Linux Administration

Rsyslog

Sorts messages into files

Every Message Carries Two Labels

`rsyslog` decides where a message goes by reading both of them

Facility

What kind of thing sent it

`cron` , `mail` , `authpriv`

Priority

How bad it is

`emerg` , `err` , `info`

Syslog Facilities

Code	Facility	Messages from
0	`kern`	The kernel
2	`mail`	The mail system
3	`daemon`	System daemons
9	`cron`	The clock daemon
10	`authpriv`	Logins, `sudo`, authentication
16 to 23	`local0` to `local7`	Whatever you assign them to

Syslog Priorities

Eight levels, lowest number most severe

Something is wrong

Code	Priority	Meaning
0	`emerg`	System is unusable
1	`alert`	Act immediately
2	`crit`	Critical condition
3	`err`	Error

Everything else

Code	Priority	Meaning
4	`warning`	Warning
5	`notice`	Normal but significant
6	`info`	Informational
7	`debug`	Debugging detail

Routing Rules

/etc/rsyslog.conf

```
#### RULES ####
...output omitted...
*.info;mail.none;authpriv.none;cron.none                /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                              /var/log/secure

# Log all the mail messages in one place.
mail.*                                                  -/var/log/maillog

# Everybody gets emergency messages
*.emerg                                                 :omusrmsg:*
```

Read each rule as `facility.priority` on the left and a destination on the right

A priority catches that level **and everything above it**, so `info` also catches `err` and `emerg`

Adding Rules in `/etc/rsyslog.d`

Before those rules, `/etc/rsyslog.conf` pulls in a drop-in directory

/etc/rsyslog.conf

```
# Include all config files in /etc/rsyslog.d/
include(file="/etc/rsyslog.d/*.conf" mode="optional")
```

Add your own rule as a file of your own there, named for what it routes

/etc/rsyslog.d/authpriv-info.conf

```
authpriv.info /var/log/authpriv-info
```

> **Warning**
>
> The name has to end in `.conf`, or the include never picks the file up

Writing and Reading

Send a message with `logger`

student@servera:~

```
student@servera:~$ logger -p authpriv.info "Houston, we have a problem."
```

`-p` sets the facility and priority

Read a file with `tail` or `less`

student@servera:~

```
student@servera:~$ sudo tail -n5 /var/log/secure
student@servera:~$ sudo tail -f /var/log/messages
student@servera:~$ sudo less /var/log/cron
```

`tail -f` follows a file, and `less` does the same with `G` to jump to the end or `F` to follow

HANDS-ON EXERCISE

Exercise: Logging authpriv.info Messages

GOAL

Send authentication messages to a file of your own by adding a rule to rsyslog

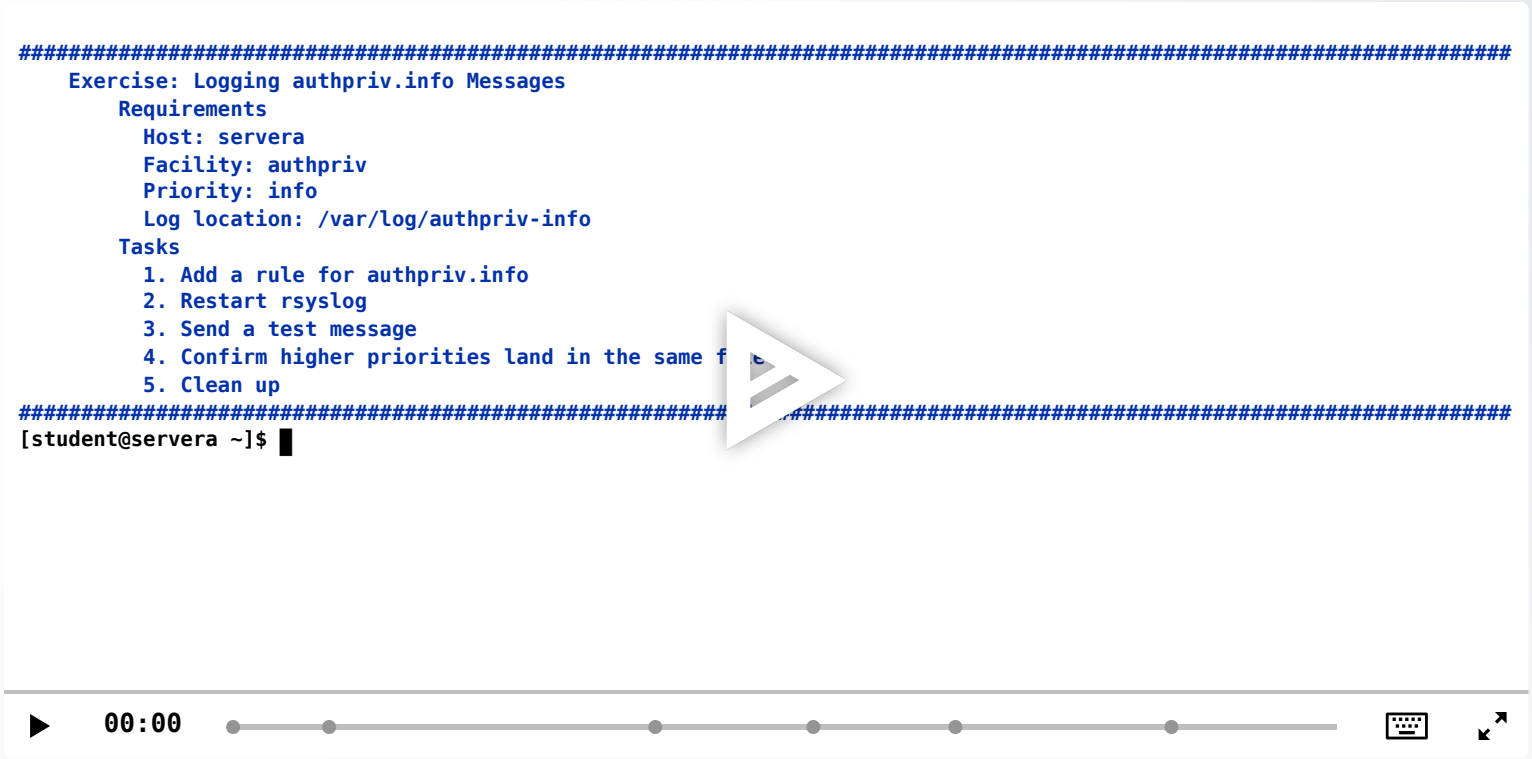
ENVIRONMENT

Host: `servera`

WORKFLOW

1. Add a rule that routes `authpriv.info` to its own file
2. Restart rsyslog so the rule takes effect
3. Send a test message with `logger` and read the file
4. Send higher priorities and confirm the same rule catches them

Exercise: Logging authpriv.info Messages



Written exercise

IT-230 · Linux Administration

Logrotate

Prunes the logs

Logrotate in Action

student@servera:~

```
student@servera:~$ sudo ls -l /var/log/secure*
-rw-------. 1 root root 43073 Sep  6 16:44 /var/log/secure
-rw-------. 1 root root 27744 Sep  5 16:27 /var/log/secure-20260906
```

`logrotate` renames the current file, starts a fresh one, and eventually deletes the oldest

It runs from `logrotate.timer`

Where the Rules Live

`/etc/logrotate.conf` sets the defaults

/etc/logrotate.conf

```
1  weekly
2  rotate 4
3  create
4  dateext
5  include /etc/logrotate.d
```

/etc/logrotate.d/rsyslog

```
1  /var/log/cron
2  /var/log/maillog
3  /var/log/messages
4  /var/log/secure
5  /var/log/spooler
6  {
7      missingok
8      sharedscripts
9      postrotate
10         /usr/bin/systemctl reload rsyslog.service
11     endscript
12 }
```

Options Worth Knowing

What to do

Option	Effect
`rotate N`	Keep `N` old files
`compress`	gzip the old ones
`missingok`	Do not error if absent
`create`	Make a fresh empty file
`postrotate`	Run commands afterward

When to do it

Option	Trigger
`daily`	Every day
`weekly`	Every week
`monthly`	Every month
`size 1k`	When it grows past a size

No schedule given means the default in `/etc/logrotate.conf`, which is `weekly`

HANDS-ON EXERCISE

Exercise: Rotating a Log File by Size

GOAL

Rotate a log file when it outgrows 1 KB, keeping only the two most recent copies

ENVIRONMENT

Host: `servera`

WORKFLOW

1. Write a logrotate configuration for `/var/log/demo.log`
2. Fill the file past the size threshold
3. Run logrotate by hand rather than waiting for the timer
4. Repeat until more than two old copies would exist
5. Confirm the oldest was discarded

Exercise: Rotating a Log File by Size

Written exercise

IT-230 · Linux Administration

Systemd Journal

Searchable record of systemd unit logs

The Journal Is Structured, Not Just Text

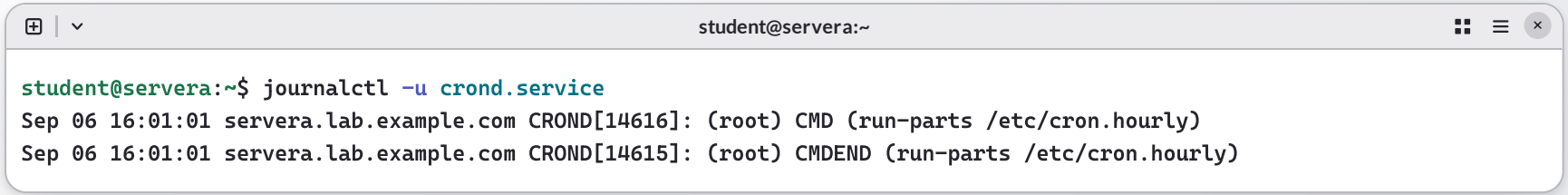
`systemd-journald` writes a binary journal, so every entry carries labelled fields

That is what lets you ask for one unit, one time range, or one priority instead of grepping

> **Warning**
>
> By default the journal lives in `/run/log/journal/`, which is memory. A reboot loses it.

Reading One Unit

`journalctl -u` is the command you will reach for most



It opens in `less`, oldest first, so `G` jumps to the newest entries

Narrowing by Time

```
journalctl --since "30 minutes ago"
journalctl --until "1 hour ago"
journalctl --since "2026-09-01 00:00:00" --until "2026-09-06 23:59:59"
```

Both options take the same human phrasing, documented in `systemd.time(7)`

Combine them with `-u` to ask about one unit during one window

Narrowing by Field

Because entries are structured, you can filter on any field they carry

Ask what exists

```
journalctl --fields
journalctl -F PRIORITY
journalctl -F _SYSTEMD_UNIT
```

Then filter by it

```
journalctl PRIORITY=0
journalctl _SYSTEMD_UNIT=crond.service
journalctl SYSLOG_IDENTIFIER=sshd
```

Making the Journal Survive a Reboot

Create the directory, and journald starts using it

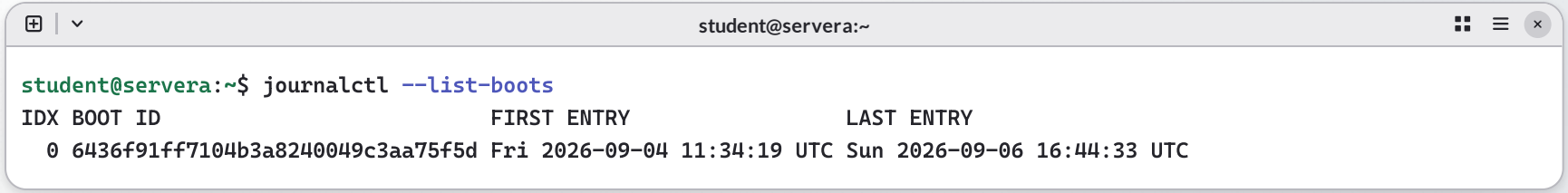
root@servera:~

```
root@servera:~# mkdir /var/log/journal
root@servera:~# journalctl --flush
root@servera:~# ls -d /var/log/journal
/var/log/journal
```

Storage moves from `/run/log/journal/` to `/var/log/journal/`, and past boots are kept

Reading an Earlier Boot

Only possible once the journal is persistent



A volatile journal only ever lists boot `0`, which is the one you are in

`journalctl -b -1` reads the boot before this one, which is where you look after a crash

HANDS-ON EXERCISE

Exercise: Reading the Journal

GOAL

Find entries by unit and by time, then make the journal survive a reboot

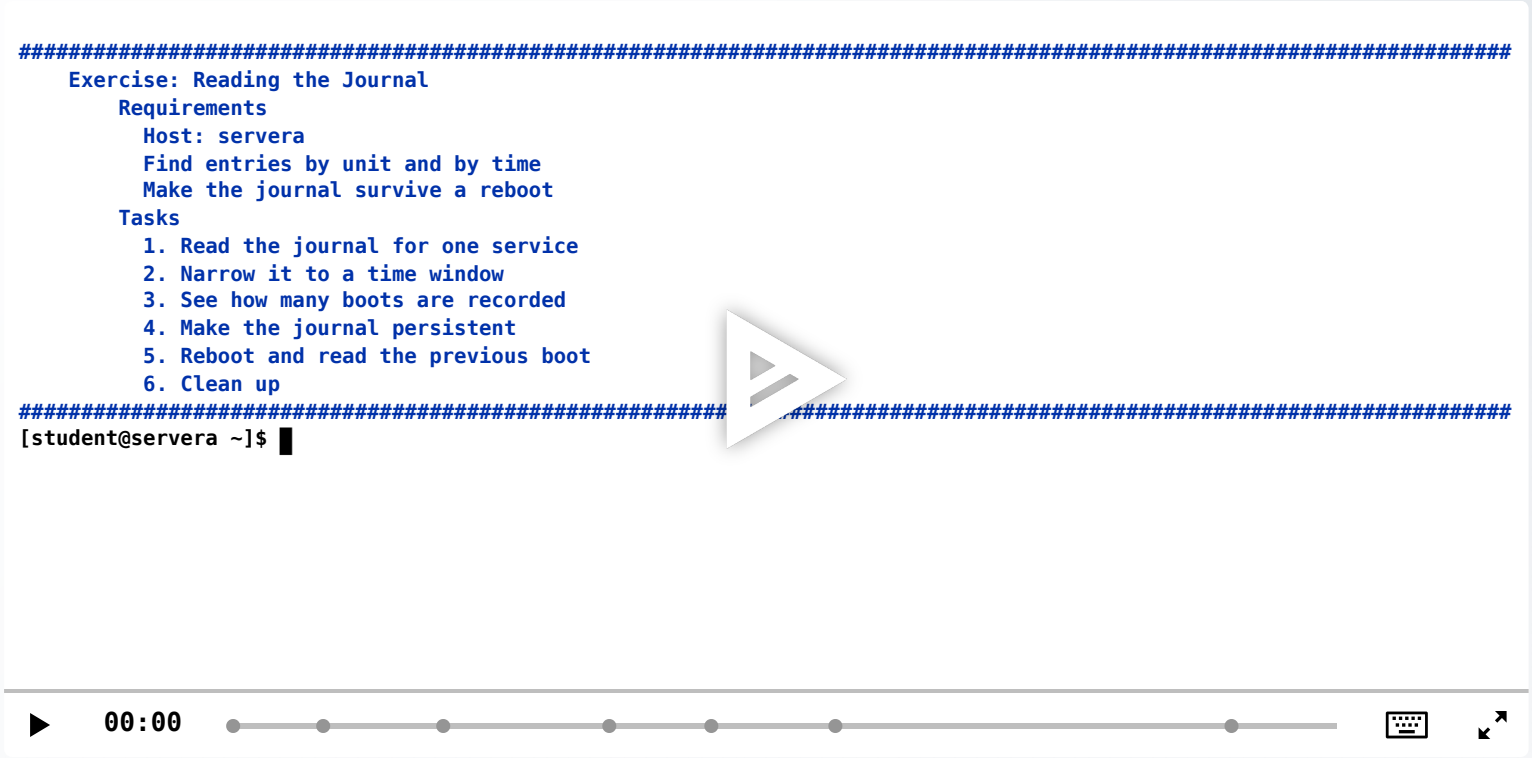
ENVIRONMENT

Host: `servera`

WORKFLOW

1. Read the journal for one service
2. Narrow the same question to a time window
3. Confirm only one boot is listed, and work out why
4. Make the journal persistent and flush it to disk
5. Reboot, then read the boot before the reboot

Exercise: Reading the Journal



Written exercise

IT-230 · Linux Administration

Times and Dates

A programmer’s worst nightmare

Two Clocks

Hardware clock

Also called the real time clock, and it keeps running when the machine is off

Generally set to **UTC**, which is the same everywhere

Windows sets it to local time instead, which is why dual-boot machines argue about the time

Software clock

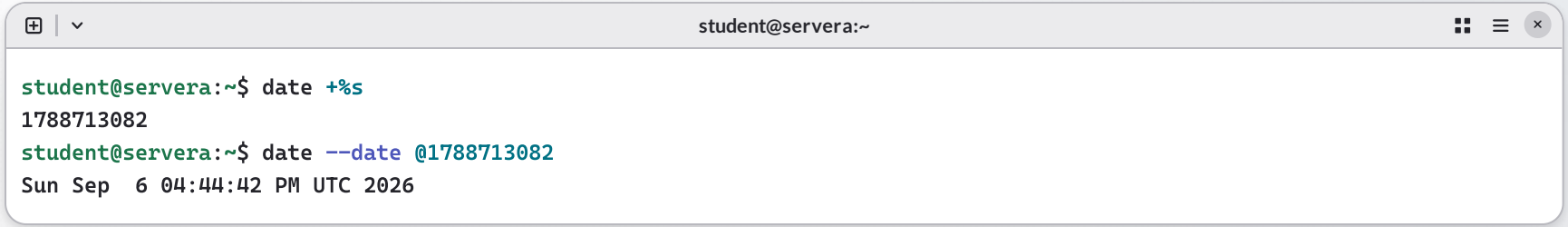
The system time the kernel keeps while running

Also stored as UTC

Converted to your time zone only when something displays it

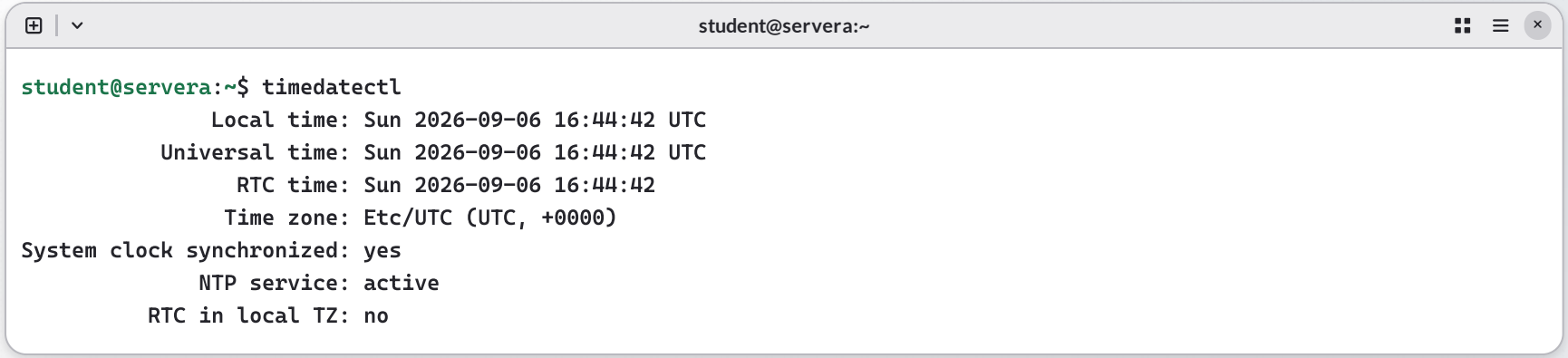
Epoch Time

Linux counts time as seconds since **1 January 1970**, UTC



The `@` is what tells `date` the number is epoch seconds rather than a date

What the System Thinks Right Now



One command answers both questions: what time is it, and is that time trustworthy

Choosing a Time Zone

`tzselect` walks you through continent, country, and region, then prints a name. It changes nothing, it only tells you what to type.

student@servera:~

```
student@servera:~$ timedatectl list-timezones | grep -i amsterdam
Europe/Amsterdam
student@servera:~$ sudo timedatectl set-timezone Europe/Amsterdam
student@servera:~$ date
Sun Sep  6 06:44:42 PM CEST 2026
```

The clock did not move, only the way it is displayed

HANDS-ON EXERCISE

Exercise: Changing the Time Zone

GOAL

Move `servera` to Amsterdam's time zone and confirm only the display changed

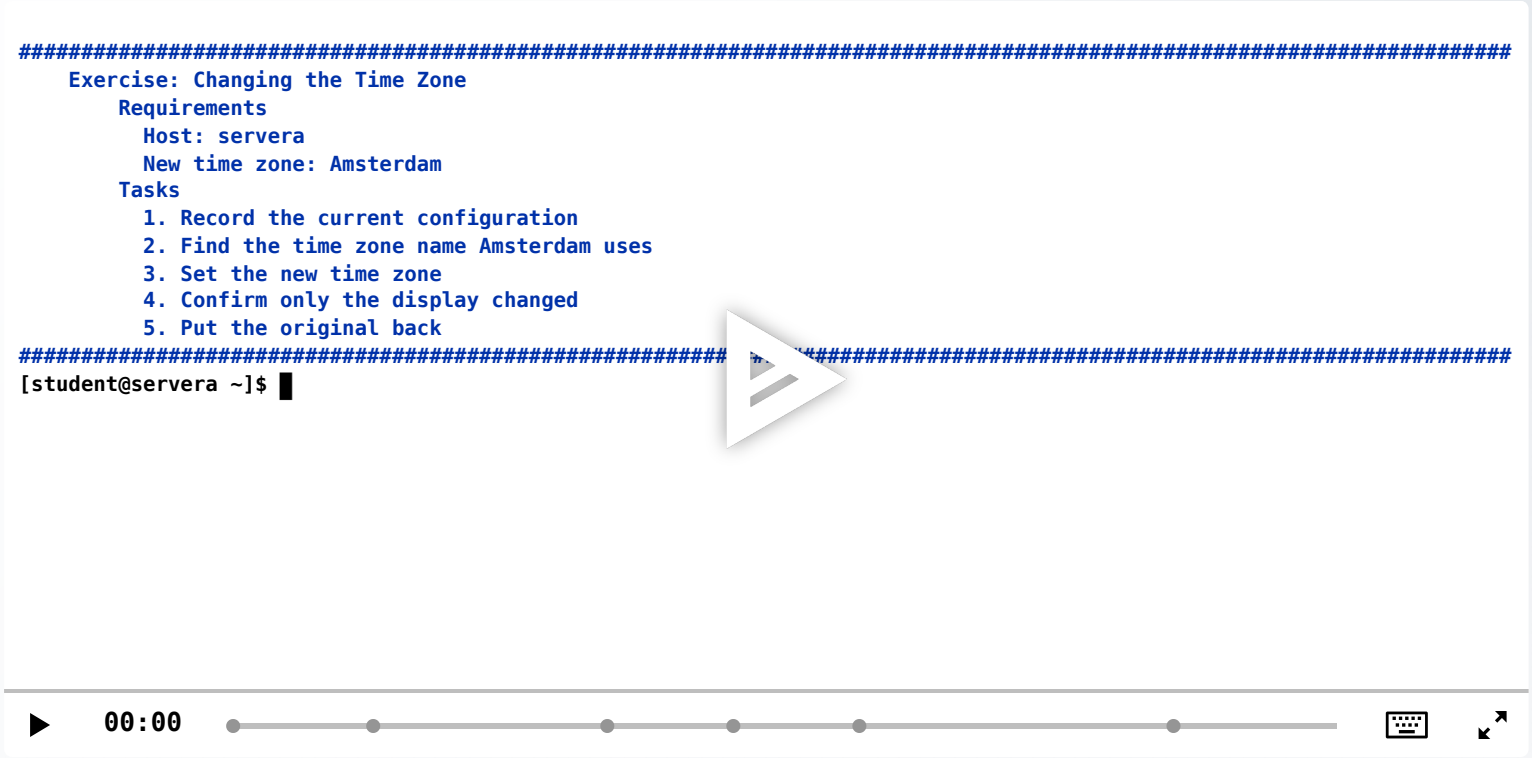
ENVIRONMENT

Host: `servera`

WORKFLOW

1. Record the current time zone and UTC offset
2. Find the time zone name Amsterdam uses
3. Set the new time zone
4. Confirm local time moved while universal time did not
5. Put the original time zone back

Exercise: Changing the Time Zone



Written exercise

IT-230 · Linux Administration

Network Time Protocol

Keeping the clock honest without touching it

Stratum Is Distance From a Real Clock

A **stratum 1** server is attached to an atomic clock or GPS receiver

Each hop away adds one, so stratum 2 asks stratum 1, and so on

Lower is better, and most public internet servers are stratum 1 or 2

> **Note**
>
> Accurate time is not cosmetic. Log correlation, certificate validation, and Kerberos all break when clocks drift.

Chrony Does the Asking

`chronyd` is the NTP client RHEL ships, configured in `/etc/chrony.conf`

/etc/chrony.conf

```
pool 2.rhel.pool.ntp.org iburst
driftfile /var/lib/chrony/drift
makestep 1.0 3
rtcsync
```

A `pool` line names a set of servers, and `iburst` makes the first sync fast

`timedatectl set-ntp true` turns synchronization on if it is off

Checking the Sources

student@servera:~

```
student@servera:~$ chronyc sources -v
^+ wdc1.us.ntp.li                4   9   377   268  +3137us[+3137us] +/-   29ms
^- ntp1.dfw.us.hojmark.net       2  10   377   322  -5606us[-5781us] +/-   42ms
^* time.cloudflare.com           3  10   377   320  -2769us[-2945us] +/-   19ms
```

`^*` is the source currently being used, `^+` is an acceptable alternative, and `^-` was excluded

The number after the name is that source's stratum

HANDS-ON EXERCISE

Exercise: Using a New NTP Source

GOAL

Point `servera` at a specific time source and confirm it synchronizes

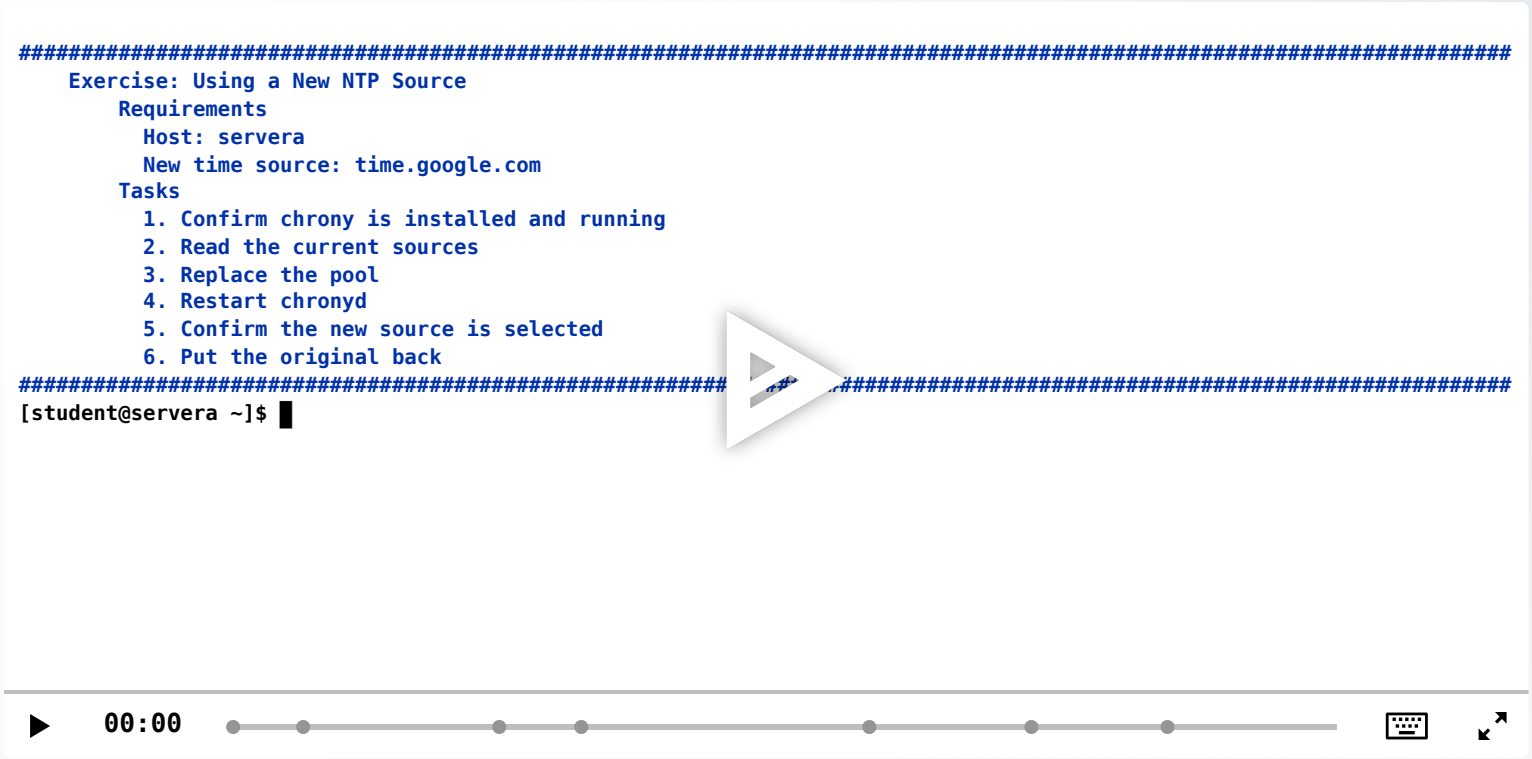
ENVIRONMENT

Host: `servera`

WORKFLOW

1. Confirm `chronyd` is installed, enabled, and running
2. Read the current sources and note which one is selected
3. Replace the pool with `time.google.com`
4. Restart `chronyd` so it reads the change
5. Confirm the new source is selected

Exercise: Using a New NTP Source



Written exercise

IT-230 · Linux Administration

Tuning System Performance

Telling the kernel what this machine is for

What Does Better Performance Mean?

It depends entirely on the job

- A game wants **low latency**, so input feels immediate
- A web server wants **throughput**, so it serves more requests
- A laptop wants **power savings**, so the battery lasts

You cannot maximize all three, so tuning is choosing which one to favor

Tuned Picks the Settings For You

`tuned` runs in the background and applies a profile of kernel settings

Profile	Tuned for
`balanced`	A compromise between power saving and performance
`throughput-performance`	Maximum throughput
`network-latency`	Deterministic low-latency networking
`powersave`	Lowest power consumption
`virtual-guest`	Running as a virtual machine

`virtual-guest` is what the lab machines use, because they are VMs

Working With Profiles

Components

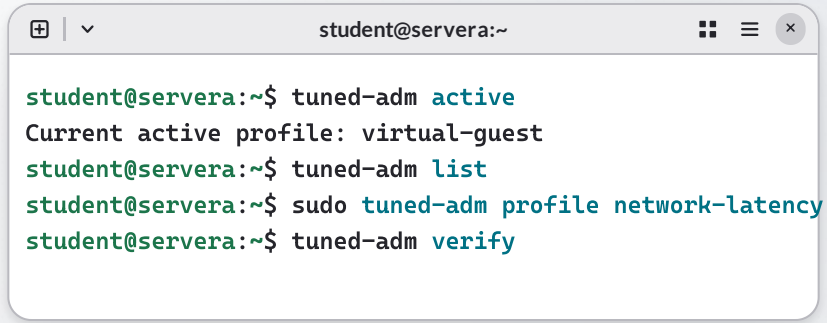
Service: `tuned.service`

Command: `tuned-adm`

Profiles: `/usr/lib/tuned/profiles/`

Log: `/var/log/tuned/tuned.log`

Commands



HANDS-ON EXERCISE

Exercise: Activating a Tuning Profile

GOAL

Switch `servera` to the network-latency profile and confirm it took effect

ENVIRONMENT

Host: `servera`

WORKFLOW

1. Install `tuned` and start its service
2. Find out which profile is active now
3. Read what the `network-latency` profile actually changes
4. Activate it
5. Verify it took effect
6. Put the original profile back

Exercise: Activating a Tuning Profile

Written exercise

IT-230 · Linux Administration

Process Priority

Deciding what waits when the CPU is busy

Not Every Task Is Urgent

Linux runs far more processes than it has CPUs, so the kernel is always choosing

- A nightly backup can afford to wait
- A user typing into a shell cannot

Priority is how you tell the kernel which is which

Nice Values

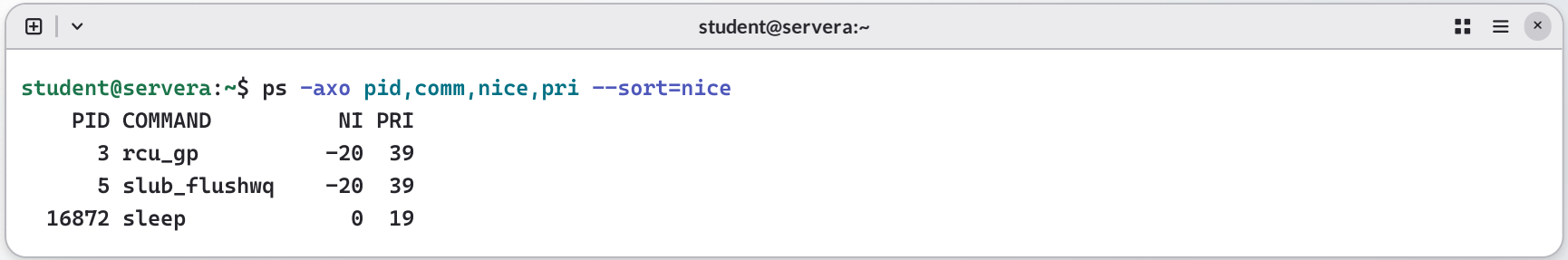
Range from -20 to 19, default `0`

A **lower** number means **less nice** to other processes, so it gets more CPU

A higher number steps aside for everything else

> **Warning**
>
> Raising priority needs `root` . An ordinary user can only be nicer, never greedier, and even to their own processes.

Reading Them



`NI` is the nice value you set, and `PRI` is the priority the kernel derived from it

`top` shows the same two columns and refreshes automatically

Setting Them

```
student@servera:~$ nice -n 15 sleep 300 &
student@servera:~$ ps -o pid,comm,nice -p 16890
    PID COMMAND          NI
  16890 sleep            15
student@servera:~$ sudo renice -n -10 16872
16872 (process ID) old priority 0, new priority -10
student@servera:~$ renice -n -5 16872
renice: failed to set priority for 16872 (process ID): Permission denied
```

`nice` starts a process with a value, `renice` changes one already running

HANDS-ON EXERCISE

Exercise: Managing Nice Values

GOAL

Change a running process's priority in both directions and find the limit

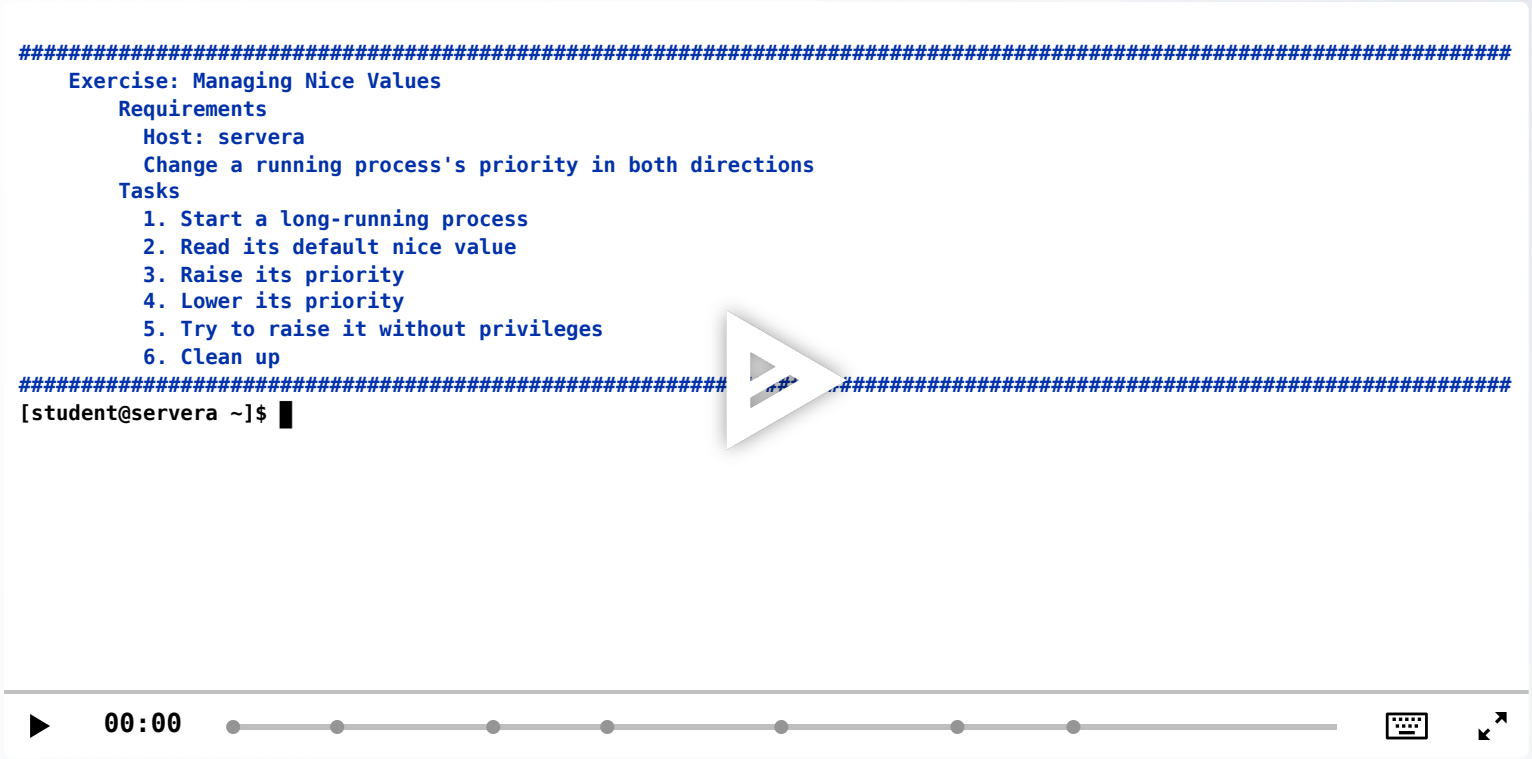
ENVIRONMENT

Host: `servera`

WORKFLOW

1. Start a long-running process in the background
2. Read its default nice value
3. Raise its priority, and notice what that required
4. Lower its priority again
5. Try to raise it without privileges and read the error
6. Stop the process

Exercise: Managing Nice Values



Written exercise

IT-230 · Linux Administration

Apache Web Server

Linux’s killer app

What a Web Server Does

Apache listens on a port, and hands back files when a browser asks for them

Component	Value
Package	`httpd`
Service	`httpd.service`
Service account	`apache` , the `nologin` account from the cron chapter
Configuration	`/etc/httpd/conf/httpd.conf`
Content	`/var/www/html/`
Logs	`/var/log/httpd/`

Two Directives Matter Today

/etc/httpd/conf/httpd.conf

```
Listen 80

DocumentRoot "/var/www/html"

<Directory "/var/www">
    AllowOverride None
    Require all granted
</Directory>
```

`Listen` is the port it answers on

`DocumentRoot` is the directory it serves from, and nothing outside it is reachable

URLs Are Paths Under DocumentRoot

File on disk	URL
`/var/www/html/index.html`	`http://servera/`
`/var/www/html/mission.html`	`http://servera/mission.html`
`/var/www/html/about/index.html`	`http://servera/about/`
`/var/www/html/about/contact-us.html`	`http://servera/about/contact-us.html`

A request for a directory gets that directory's `index.html`

Starting It

`httpd` ships **disabled**, so installing it does not start anything

Everything you learned about `systemctl` last week applies unchanged

Two Things Stand Between You and a Page

student@workstation:~

```
student@workstation:~$ curl http://localhost/
<h1>Welcome to servera</h1>
student@workstation:~$ curl http://servera/
curl: (28) Failed to connect to servera port 80: Connection timed out
```

Locally it works and remotely it does not, because `firewalld` is running

> **Warning**
>
> `firewall-cmd --add-service=http --permanent` then `firewall-cmd --reload` opens port 80. SELinux is the other gatekeeper, and that is next week.

HANDS-ON EXERCISE

Exercise: Serving a Web Page

GOAL

Stand up a web server, publish two pages

ENVIRONMENT

Hosts: `servera`

WORKFLOW

1. Install `httpd` on `servera`
2. Write a home page under the DocumentRoot
3. Enable and start the service, then fetch the page locally
4. Add a page in a subdirectory and fetch it by its URL

Exercise: Serving a Web Page

Written exercise