Day 6 – Phase 6: Log Rotation, Scheduling, Archiving

Boss’s Request: Prepare the system for production use.

**Tasks:**

• **Configure log rotation for temperature.log (rotate at 1 MB, compress).**



A computer screen with white text

AI-generated content may be incorrect.

**• Test by forcing a rotation.**

A computer screen with a black background

AI-generated content may be incorrect.

A computer screen shot of a computer code

AI-generated content may be incorrect.

**• Schedule the Python script to run every 5 minutes with cron.**

A screenshot of a computer program

AI-generated content may be incorrect.

**• Verify log growth over time.**

**A computer screen shot of a purple screen

AI-generated content may be incorrect.**

**• Compress old logs into .tar.gz in data/.**

A computer screen with white text

AI-generated content may be incorrect.

**• Simulate sending archives to /home//server/ using cp, scp, or rsync. (hint: use can use scp and copy to destination directory in another path on the same machine just for simulation).**

A computer screen with white text

AI-generated content may be incorrect.

A computer screen with white text

AI-generated content may be incorrect.

**Open-Ended Questions:**

**• How does cron scheduling work? Show a crontab entry to run a script every 5 minutes.**

Cron is a time-based job scheduler in Unix-like operating systems. It works by using a daemon called crond that runs in the background and constantly checks for scheduled tasks. The schedule is defined in a file called a crontab (cron table), which contains commands to be executed at a specific time.

The crontab entry has five fields for specifying the time, followed by the command to be executed. The fields are:

* Minute (0-59)
* Hour (0-23)
* Day of the month (1-31)
* Month (1-12)
* Day of the week (0-6, where 0 and 7 are Sunday)

An asterisk \* acts as a wildcard, matching all possible values for that field.

A crontab entry to run a script every 5 minutes would look like this:

\*/5 \* \* \* \* /path/to/your/script.sh

In this example, \*/5 means "every 5th minute," and the remaining \*s mean "every hour, every day of the month, every month, and every day of the week."

**• Why do we need log rotation? Show an example logrotate config for temperature.log.**

We need log rotation to manage disk space and improve system performance. Log files, especially on busy systems, can grow indefinitely, consuming all available disk space. This can lead to system instability and even crashes. Log rotation automates the process of archiving, compressing, and deleting old log files, ensuring that logs don't take up too much space.

Another important reason is to make log analysis easier. Instead of sifting through one gigantic log file, a series of smaller, date-stamped log files are easier to manage and analyze. This also prevents log files from becoming corrupted due to their large size.

**• Explain the difference between a Virtual Machine and a Container. Must containers use the same OS as the host? Why or why not?**

* **VM**:
  + Runs a full **guest OS** on top of a hypervisor
  + Heavy (GBs of memory, minutes to start)
  + Stronger isolation (separate kernel)
* **Container**:
  + Shares the **host OS kernel**
  + Lightweight (MBs, starts in seconds)
  + Uses namespaces + cgroups for isolation

**Must containers use the same OS as the host?**

* **Yes, same kernel.**
  + Example: Docker on Linux can only run Linux containers.
* But you can run different **distributions** (e.g., Ubuntu container on Fedora host).
* Windows/macOS use **VM layers** (like WSL2 or HyperKit) to run Linux containers.

**• Reflection: Which actions in this project combined multiple Linux concepts (e.g.,**

**redirection + process monitoring)? How does this apply to real IoT systems?**

* **Redirection (>>, 2>&1)** : saving sensor data to log files
* **Process monitoring (cron)** : scheduling periodic execution
* **Compression (gzip, tar)** → archiving logs
* **Permissions (chmod, groups)** : controlling file access

**How this applies to real IoT systems:**

* IoT devices continuously generate data : must **log & rotate** to prevent storage overflow.
* Logs are often **compressed & archived** for analysis in the cloud.
* Automation via **cron/systemd timers** ensures reliability without manual work.
* Using **lightweight containers** instead of VMs saves resources, critical for edge/IoT devices.