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

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

Tasks:

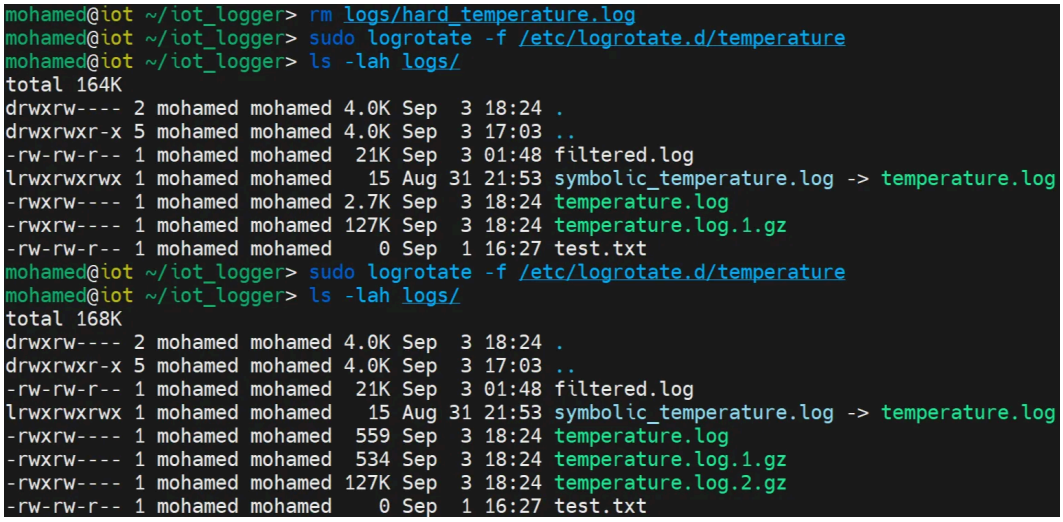
- Configure log rotation for `temperature.log` (rotate at 1 MB, compress).
  - **Log rotation** is the process of automatically **managing log files** so they don't grow too large and fill up your disk.
  - I will create configuration file to temperature log rotation `/etc/logrotate.d/temperature`

```
mohamed@iot ~/iot_logger> cat /etc/logrotate.d/temperature
/home/mohamed/iot_logger/logs/temperature.log {
    size 1M
    rotate 5
    compress
    missingok
    notifempty
    copytruncate
    su mohamed mohamed
}
```

    - **size 1M** → rotate when file > 1M
    - **rotate 2** → keep 5 old log files
    - **compress** → compress rotated logs (`.gz`).
    - **missingok** → don't throw error if log is missing.
    - **notifempty** → don't rotate if empty.
    - **copytruncate** → truncate original log after copying (so process keeps writing) (Don't understand well)
    - **su mohamed mohamed** → which user & group to use

- Test by forcing a rotation.
  - At first `logrotate` refuses to rotate files that have **hard links**, because renaming or truncating one would break consistency (I knew that using `sudo logrotate -d /etc/logrotate.d/temperature`)
  - So we will first remove hard link then test

```
rm logs/hard_temperature.log
sudo logrotate -f /etc/logrotate.d/temperature
ls -lah logs/
```



- Schedule the Python script to run every 5 minutes with `cron`.
  - Crontab is a schedule file where you tell Linux: “Run this command at this time.” automatically
  - First I will rewrite `~/iot_logger/scripts/sensor_script.py` without `while` or `sleep`

```python
import os, time, random

# Get sensor type from environment variable
sensor = os.getenv("SENSOR_TYPE", "unknown")

value = random.randint(15, 45)
print(f"{time.ctime()} | {sensor}: {value}")
```

  - Open crontab for current user → `crontab -e`
  - Add your schedule task
    - Every 5 minutes run `sensor_script.py` and append sensor data to `temperature.log`
  - Make sure every thing is ok → `crontab -l`

```
mohamed@iot ~> crontab -l | tail -n 1
*/5 * * * * SENSOR_TYPE=temperature /usr/bin/python3 /home/mohamed/iot_logger
/scripts/sensor_script.py >> /home/mohamed/iot_logger/logs/temperature.log
```

  - Check `temperature.log` files

```
mohamed@iot ~/iot_logger> tail -n 3 logs/temperature.log
Wed Sep  3 20:35:01 2025 | temperature: 15
Wed Sep  3 20:40:01 2025 | temperature: 38
Wed Sep  3 20:45:01 2025 | temperature: 36
```

- Verify log growth over time.

```
mohamed@iot ~/iot_logger> ls -lah logs/temperature.log
-rwxrw---- 1 mohamed mohamed 190K Sep  3 20:40 logs/temperature.log
```

```
mohamed@iot ~/iot_logger> ls -lah logs/temperature.log
-rwxrw---- 1 mohamed mohamed 191K Sep  3 22:05 logs/temperature.log
```

- Compress old logs into `.tar.gz` in `data/`.

```
tar -czf data/old_logs.tar.gz logs/*.gz
```

```
mohamed@iot ~/iot_logger> ls -lh data/old_logs.tar.gz
-rw-rw-r-- 1 mohamed mohamed 311K Sep  3 20:57 data/old_logs.tar.gz
mohamed@iot ~/iot_logger> tar -tzf data/old_logs.tar.gz
logs/temperature.log.1.gz
logs/temperature.log.2.gz
logs/temperature.log.3.gz
```

- Simulate sending archives to `/home/<username>/server/` using `cp`, `scp`, or `rsync`. (hint: you can use `scp` and `copy` to destination directory in another path on the same machine just for simulation).
  - Create `server` folder for simulation → `mkdir ~/server`
  - Simulate remote copy → `scp ~/iot_logger/data/*.tar.gz ~/server/`

```
mohamed@iot ~> mkdir ~/server
mohamed@iot ~> ls server/
mohamed@iot ~> scp ~/iot_logger/data/*.tar.gz ~/server/
mohamed@iot ~> ls server/
old_logs.tar.gz
```

  - You can also use `cp`, `rsync` instead of `scp`
    - `cp` → Local system only
    - `rsync` → Local and remote
    - `scp` → Local ↔ Remote

Open-Ended Questions:

- How does `cron` scheduling work? Show a crontab entry to run a script every 5 minutes.
  - **Cron** is a background service that runs jobs at scheduled times.
  - **Cron** is a background service that runs jobs at scheduled times.
  - Example:

```
*/5 * * * * /usr/bin/python3 /home/mohamed/iot_logger/scripts/sensor_script.py >> /home/mohamed/iot_logger/logs/temperature.log
```

- Why do we need log rotation? Show an example logrotate config for temperature.log.
  - Logs grow continuously and can fill up disk space.
  - **Log rotation** keeps logs manageable
  - Example:

```
/home/mohamed/iot_logger/logs/temperature.log {
    size 1M
    rotate 5
    compress
    missingok
    notifempty
    copytruncate
    su mohamed mohamed
}
```

- Explain the difference between a Virtual Machine and a Container. Must containers use the same OS as the host? Why or why not?
  - **Virtual Machine (VM):**
    - Runs full OS with its **own kernel**.
    - Heavyweight → more CPU & RAM usage.
    - Strong isolation (hardware-level).
    - Slow startup (minutes).
  - **Container:**
    - **Shares** host's **kernel**.
    - Lightweight → less resource usage.
    - Isolation at process-level.
    - Fast startup (seconds).
  - **Do containers need same OS?**
    - Must share the same kernel family with different distributions.

- Reflection: Which actions in this project combined multiple Linux concepts (e.g., redirection + process monitoring)? How does this apply to real IoT systems?
  - I think `python3 script.py >> logs/temperature.log &`
    - `>>` → redirection
    - `&` → process monitoring
    - And `logrotate` & `cron` → file management , compression, scheduling
  - Real **IoT** systems**:**
    - Need rotation to avoid storage overflow.
    - Scheduling ensures periodic data collection.
    - Archiving + transfer → central monitoring system.