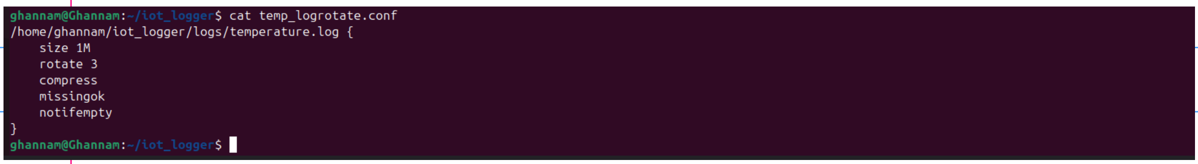
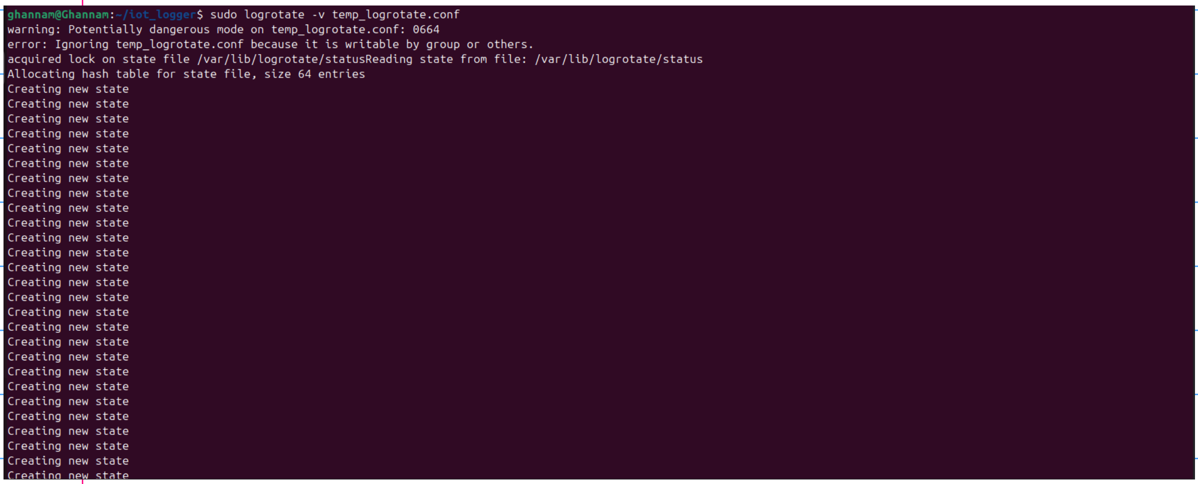
1. Configure log rotation for temperature.log (rotate at 1 MB, compress).

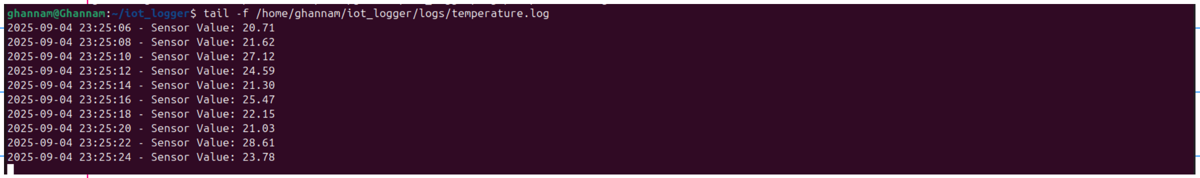


```
ghannam@Ghannam:~/iot_logger$ cat temp_logrotate.conf
/home/ghannam/iot_logger/logs/temperature.log {
    size 1M
    rotate 3
    compress
    missingok
    notifempty
}
ghannam@Ghannam:~/iot_logger$
```

2. Test by forcing a rotation.

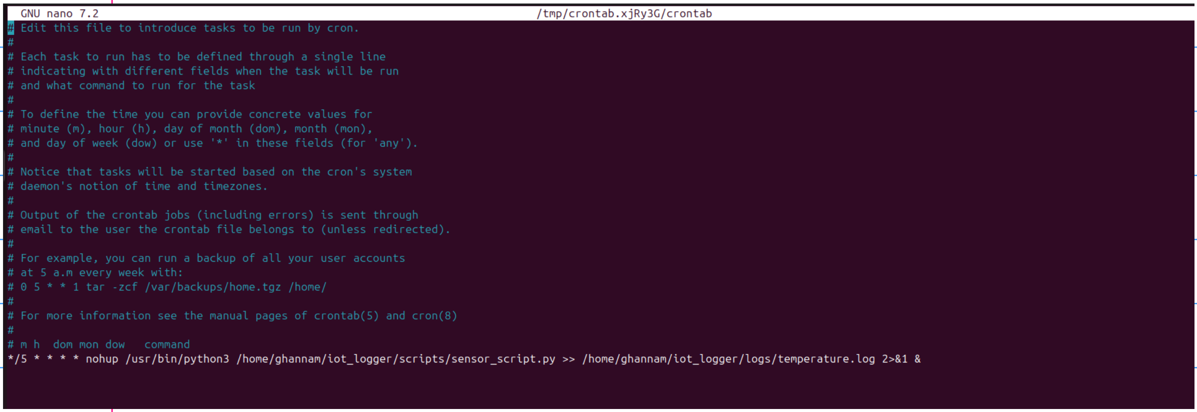


```
ghannam@Ghannam:~/iot_logger$ sudo logrotate -v temp_logrotate.conf
warning: Potentially dangerous mode on temp_logrotate.conf: 0664
error: Ignoring temp_logrotate.conf because it is writable by group or others.
acquired lock on state file /var/lib/logrotate/statusReading state from file: /var/lib/logrotate/status
Allocating hash table for state file, size 64 entries
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
Creating new state
```

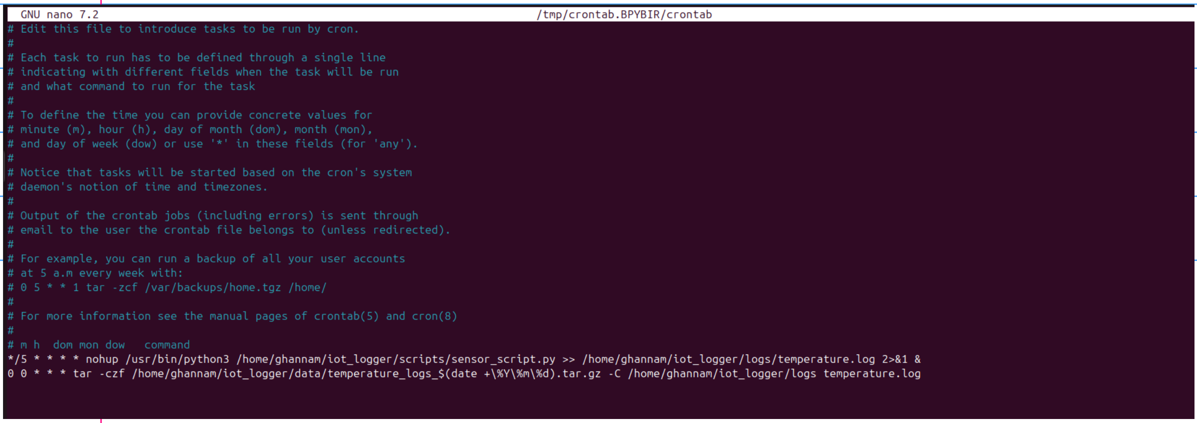


```
ghannam@Ghannam:~/iot_logger$ tail -f /home/ghannam/iot_logger/logs/temperature.log
2025-09-04 23:25:06 - Sensor Value: 20.71
2025-09-04 23:25:08 - Sensor Value: 21.62
2025-09-04 23:25:10 - Sensor Value: 27.12
2025-09-04 23:25:12 - Sensor Value: 24.59
2025-09-04 23:25:14 - Sensor Value: 21.30
2025-09-04 23:25:16 - Sensor Value: 25.47
2025-09-04 23:25:18 - Sensor Value: 22.15
2025-09-04 23:25:20 - Sensor Value: 21.03
2025-09-04 23:25:22 - Sensor Value: 28.61
2025-09-04 23:25:24 - Sensor Value: 23.78
```

3, 4, 5. Schedule the Python script to run every 5 minutes with cron, Verify log growth over time. Compress old logs into .tar.gz in data/.

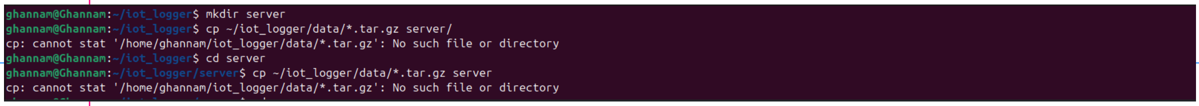


```
  GNU nano 7.2                                   /tmp/crontab.xjRy3G/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
*/5 * * * * nohup /usr/bin/python3 /home/ghannam/iot_logger/scripts/sensor_script.py >> /home/ghannam/iot_logger/logs/temperature.log 2>&1 &
```



```
  GNU nano 7.2                                   /tmp/crontab.BPYBIR/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
*/5 * * * * nohup /usr/bin/python3 /home/ghannam/iot_logger/scripts/sensor_script.py >> /home/ghannam/iot_logger/logs/temperature.log 2>&1 &
0 0 * * * tar -czf /home/ghannam/iot_logger/data/temperature_logs_$(date +\%Y\%m\%d).tar.gz -C /home/ghannam/iot_logger/logs temperature.log
```

6. Simulate sending archives to /home/<username>/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).



```
ghannam@Ghannam:~/iot_logger$ mkdir server
ghannam@Ghannam:~/iot_logger$ cp ~/iot_logger/data/*.tar.gz server/
cp: cannot stat '/home/ghannam/iot_logger/data/*.tar.gz': No such file or directory
ghannam@Ghannam:~/iot_logger$ cd server
ghannam@Ghannam:~/iot_logger/server$ cp ~/iot_logger/data/*.tar.gz server
cp: cannot stat '/home/ghannam/iot_logger/data/*.tar.gz': No such file or directory
```

How cron scheduling works
cron is a background service (daemon) that runs tasks at specific times.
Tasks are defined in a file called a crontab (cron table).
Each line in a crontab has a schedule + the command to run.

Run a script every 5 minutes

***/5 * * * * /home/ghannam/iot_logger/scripts/sensor_script.py >> /home/ghannam/iot_logger/logs/temperature.log 2>&1**

Why do we need log rotation?
Logs grow forever if not managed.
Huge log files → waste disk space, become slow to read, and can even fill the filesystem.
Log rotation solves this by:
Splitting big logs into smaller chunks.
Archiving old logs (with date/number suffix).
Compressing them to save space.
Keeping only a fixed number of old logs.
So rotation keeps logs small, manageable, and prevents disk issues.

Virtual Machine (VM) VS Container

Virtual Machine (VM)
A VM emulates an entire computer.
Needs a hypervisor (like VirtualBox, VMware, KVM).
Each VM runs its own full OS (kernel + system).
Heavy: more memory, CPU, and disk needed.
Good for strong isolation and running completely different operating systems

Container
A container is a lightweight, isolated environment.
Shares the host OS kernel, but has its own filesystem, libraries, and processes.
Managed by tools like Docker or Podman.
Much faster and lighter than VMs (no need to boot a full OS).
Best for packaging apps and their dependencies.

Actions that combined multiple Linux concepts in your project:
Running scripts with cron + logging
You scheduled the Python script using cron (automation).
The script output was handled with stdout/stderr redirection into logs.
→ Combines process scheduling + redirection.
Log rotation + compression
Old logs were compressed (.tar.gz) after rotation.
This required knowledge of filesystem management + archiving utilities.
→ Combines file management + resource efficiency.
Simulated file transfer
Using scp/rsync to move archives combined networking concepts with file permissions and securi
→ Combines system administration + communication

How this applies to real IoT systems:
Automation (cron): IoT devices often collect sensor data periodically. Scheduling ensures data is captured at regular intervals without human intervention.
Logging & Monitoring: Just like you redirected logs, IoT systems must track device health and sensor readings for debugging and auditing.
Log Rotation: Devices have limited storage. Rotation + compression prevents logs from filling up memory.
Data Transfer (scp/rsync): IoT devices need to send collected data to central servers.
Even though you simulated locally, in real life this would be cloud upload.
Concept Integration: Real IoT solutions rarely use one concept in isolation.
They blend automation, networking, security, storage, and process management to keep systems reliable.