CA6C1 – DevOps - National Institute of Technology, Trichy

HOME-WORK 2

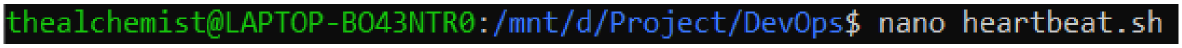
Q1: Write a HEARTBEAT job using BASH using the system log of your choice on your desired operating system.

Introduction

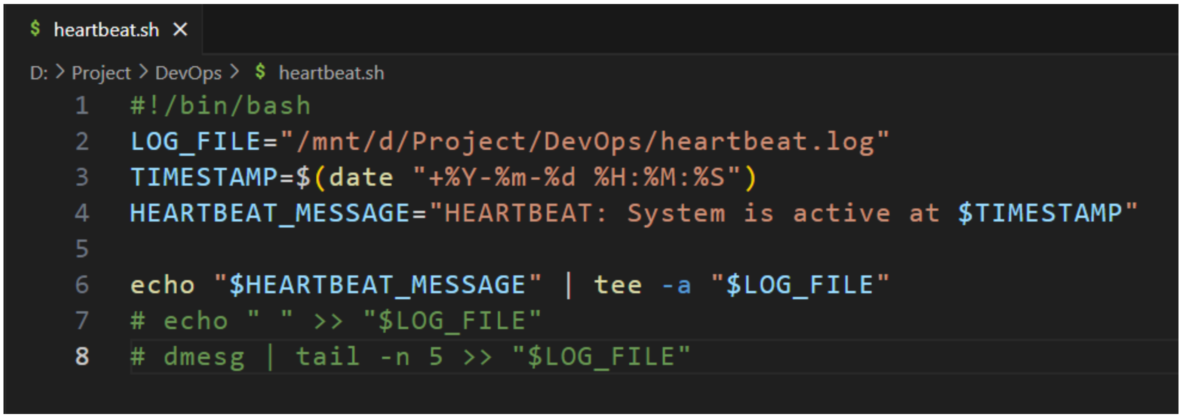
This document describes a heartbeat job implemented in Bash, which logs system status at regular intervals. The job uses system logs to track activity and appends a timestamped heartbeat message to a custom log file.

Steps for the HEARTBEAT Job

- **Creating a Bash Script with name heartbeat.sh**



```
thealchemist@LAPTOP-BO43NTR0:/mnt/d/Project/DevOps$ nano heartbeat.sh
```



```bash
#!/bin/bash
LOG_FILE="/mnt/d/Project/DevOps/heartbeat.log"
TIMESTAMP=$(date "+%Y-%m-%d %H:%M:%S")
HEARTBEAT_MESSAGE="HEARTBEAT: System is active at $TIMESTAMP"

echo "$HEARTBEAT_MESSAGE" | tee -a "$LOG_FILE"
# echo " " >> "$LOG_FILE"
# dmesg | tail -n 5 >> "$LOG_FILE"
```

1. **Defines the Log File Path**: The script sets **LOG_FILE="/mnt/d/Project/DevOps/heartbeat.log"** to specify where the heartbeat messages will be stored.

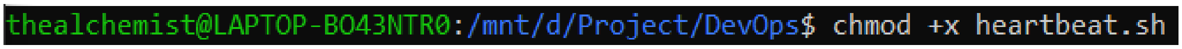
2. **Generates a Timestamp**: It captures the current date and time in the format YYYY-MM-DD HH:MM:SS using date **"+%Y-%m-%d %H:%M:%S"** and stores it in the TIMESTAMP variable.

3. **Creates a Heartbeat Message**: It constructs a message **HEARTBEAT_MESSAGE="HEARTBEAT: System is active at $TIMESTAMP"**, which includes the timestamp to indicate system activity.

4. **Logs the Heartbeat Message**: The **echo "$HEARTBEAT_MESSAGE" | tee -a "$LOG_FILE"** command prints the message to both the terminal and the log file.

5. **Kernel Logs**: The lines **echo " " >> "$LOG_FILE"** and **dmesg | tail -n 5 >> "$LOG_FILE"** suggest that the script was initially intended to append a blank line and log the last five kernel messages using dmesg.

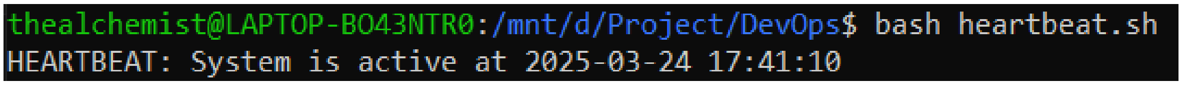
- **Making Script Executable :**



```
thealchemist@LAPTOP-BO43NTR0:/mnt/d/Project/DevOps$ chmod +x heartbeat.sh
```

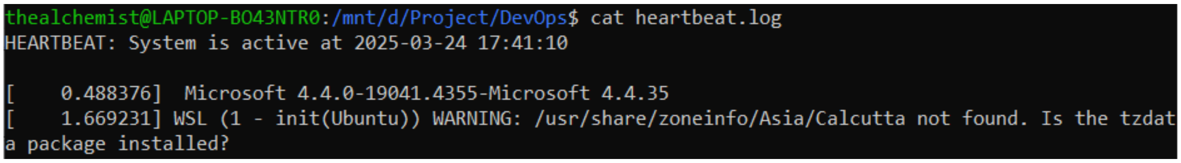
- **Testing the Script:**

Executing heartbeat.sh file:



```
thealchemist@LAPTOP-BO43NTR0:/mnt/d/Project/DevOps$ bash heartbeat.sh
HEARTBEAT: System is active at 2025-03-24 17:41:10
```

Reading hearbeat.log file:

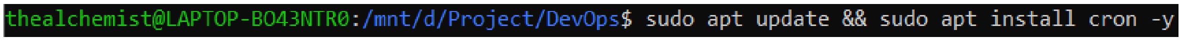


```
thealchemist@LAPTOP-BO43NTR0:/mnt/d/Project/DevOps$ cat heartbeat.log
HEARTBEAT: System is active at 2025-03-24 17:41:10

[    0.488376]  Microsoft 4.4.0-19041.4355-Microsoft 4.4.35
[    1.669231] WSL (1 - init(Ubuntu)) WARNING: /usr/share/zoneinfo/Asia/Calcutta not found. Is the tzdat
a package installed?
```

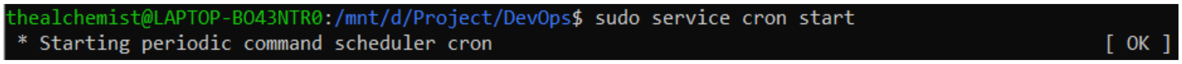
Automating Execution Using Cron in WSL

- **Installing cron service in wsl**



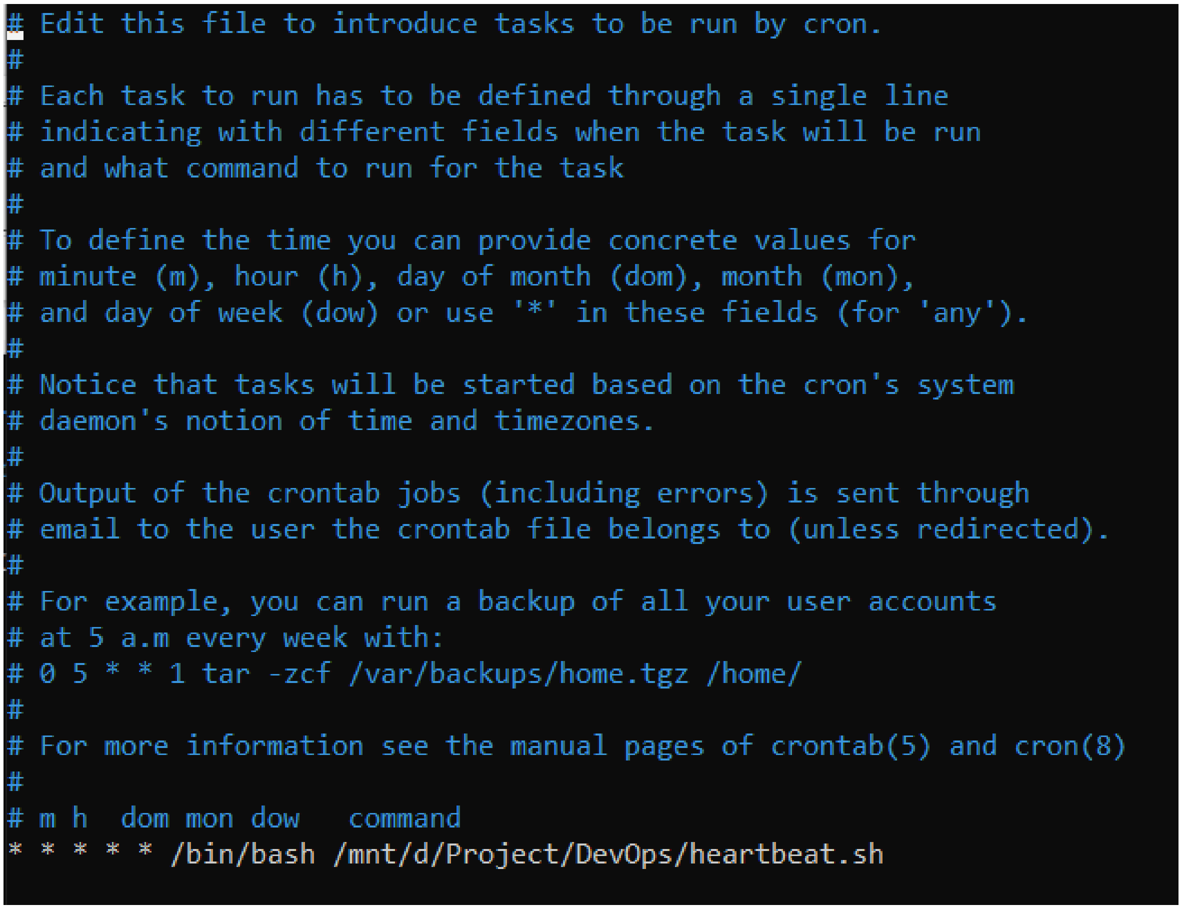
```
thealchemist@LAPTOP-BO43NTR0:/mnt/d/Project/DevOps$ sudo apt update && sudo apt install cron -y
```

- **Running Crontab editor**



```
thealchemist@LAPTOP-BO43NTR0:/mnt/d/Project/DevOps$ sudo service cron start
 * Starting periodic command scheduler cron                                                 [ OK ]
```

- **Adding Job to Crontab scheduler**

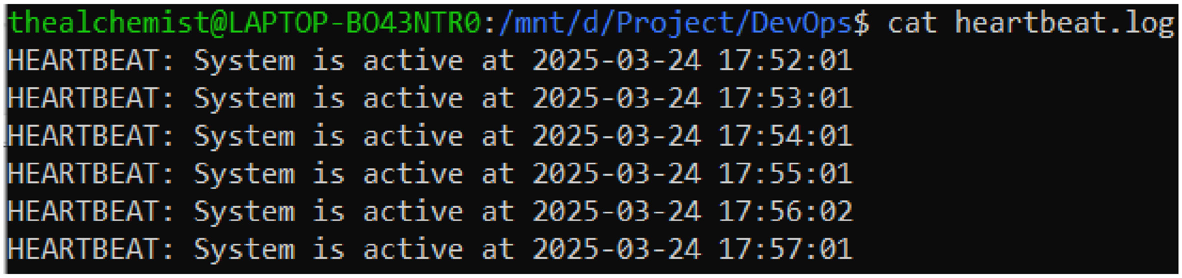


```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), 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
* * * * * /bin/bash /mnt/d/Project/DevOps/heartbeat.sh
```

Note: Crontab Default Interval: The minimum interval for a cron job is 1 minute; it does not support scheduling jobs in seconds.

To execute a script every **30 seconds**, we can use the following workaround:
"* * * * * sleep 30 && /bin/bash /mnt/d/Project/DevOps/heartbeat.sh"
This runs the script **twice per minute**—once at the start of the minute and once after 30 seconds.

Log Output



```
thealchemist@LAPTOP-BO43NTR0:/mnt/d/Project/DevOps$ cat heartbeat.log
HEARTBEAT: System is active at 2025-03-24 17:52:01
HEARTBEAT: System is active at 2025-03-24 17:53:01
HEARTBEAT: System is active at 2025-03-24 17:54:01
HEARTBEAT: System is active at 2025-03-24 17:55:01
HEARTBEAT: System is active at 2025-03-24 17:56:02
HEARTBEAT: System is active at 2025-03-24 17:57:01
```