



Al-Najah National University
Computer Network and Information Security
Network Administration Lab

Periodic Processes Management & Logging

Instructor: Dr. Ahmed Awad

Dima Bshara
Mohammed Adnan

1 Abstract

The goal of this experiment is to deal with periodic processes to schedule some administrative tasks on a periodic basis. Also, practice managing Syslog messages.

2 Introduction

Sometimes you will need to run commands or scripts on a recurring schedule. Such processes are more used in network and system administration. Such tasks might include periodic checking for network connectivity, periodic updates, and others. Linux has a built-in mechanism for running periodic processes called cron. **Cron** is a standard Unix utility that used to schedule commands for automatic execution at specific intervals. It is a daemon whose objective is to run commands on a predetermined schedule. For instance, you might have a script that produces web statistics that you want to run once a day automatically at 5:00 AM. Users can have their **crontab** files often there is a system-wide crontab file (usually in /etc or a subdirectory of /etc) that only system administrators can edit. Each line of a crontab file represents a job, and looks like figure 1 :

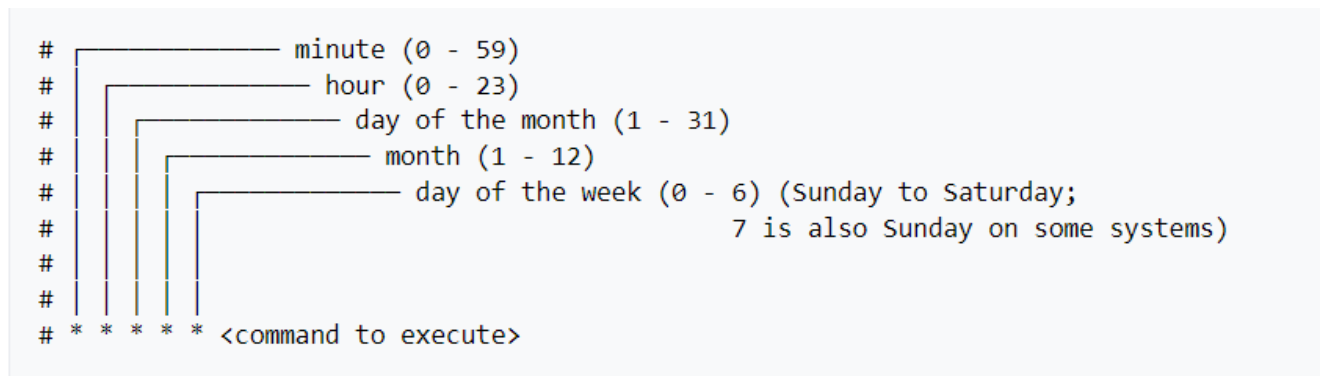


Figure 1: Format of crontab file.

Identifying problems and trends, and trouble-shoot them requires observing events over a period of time (historical monitoring). Since it is generally impossible to observe all events as they occur, most daemons record important events to files known as log files. Log files are used for audits, for evidence in legal actions, for incident response, to reduce liability, and for various legal and regulatory compliance reasons. Email logs can alert you to spam problems. Linux provides a centralized repository of log files that can be located under the `/var/log` directory.

The log files generated in a Linux environment can typically be classified into four different categories: **Application Logs** **Event Logs** **Service Logs** **System Logs**.

Syslog stands for System Logging Protocol and is a standard protocol used to send system log or event messages to a specific server, called a syslog server. It is primarily used to collect various device logs from several different machines in a central location for monitoring and review. The advantage of Syslog over journald is that logs are written in files that can be read using basic text manipulation commands like cat, grep, tail, etc.

3 Procedure

3.1 Crontab Configuration

1. In first, we made sure that crontab is install in our machine, using command `systemctl status cron.service` as shown in figure 2.

```
student@localhost:~/Desktop$ systemctl status cron.service
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: ☐)
   Active: active (running) since Tue 2021-09-28 08:11:33 EEST; 1h 7min ago
     Docs: man:cron(8)
    Main PID: 546 (cron)
      Tasks: 1 (limit: 4651)
     Memory: 480.0K
    CGroup: /system.slice/cron.service
            └─546 /usr/sbin/cron -f
```

Figure 2: Check crontab.

2. Then we open crontab file using command `crontab-e`, as figure 3 show .

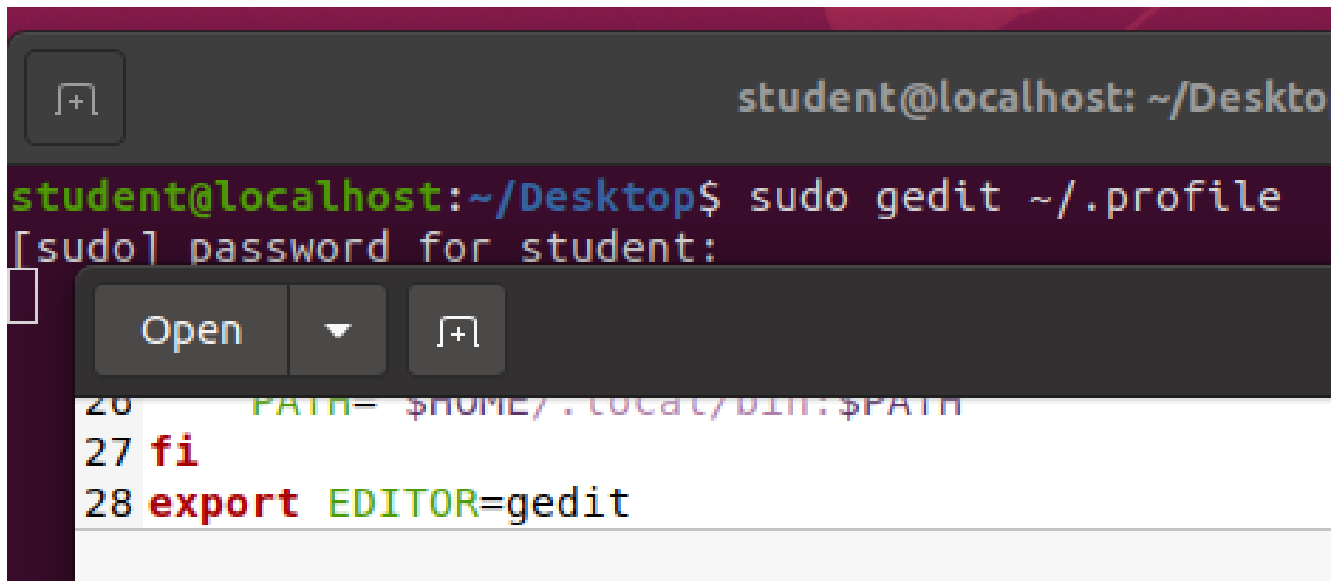
```
student@localhost:~/Desktop$ crontab -e
no crontab for student - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/nano          <---- easiest
 2. /usr/bin/vim.tiny
 3. /bin/ed

Choose 1-3 [1]: 
```

Figure 3: Check crontab.

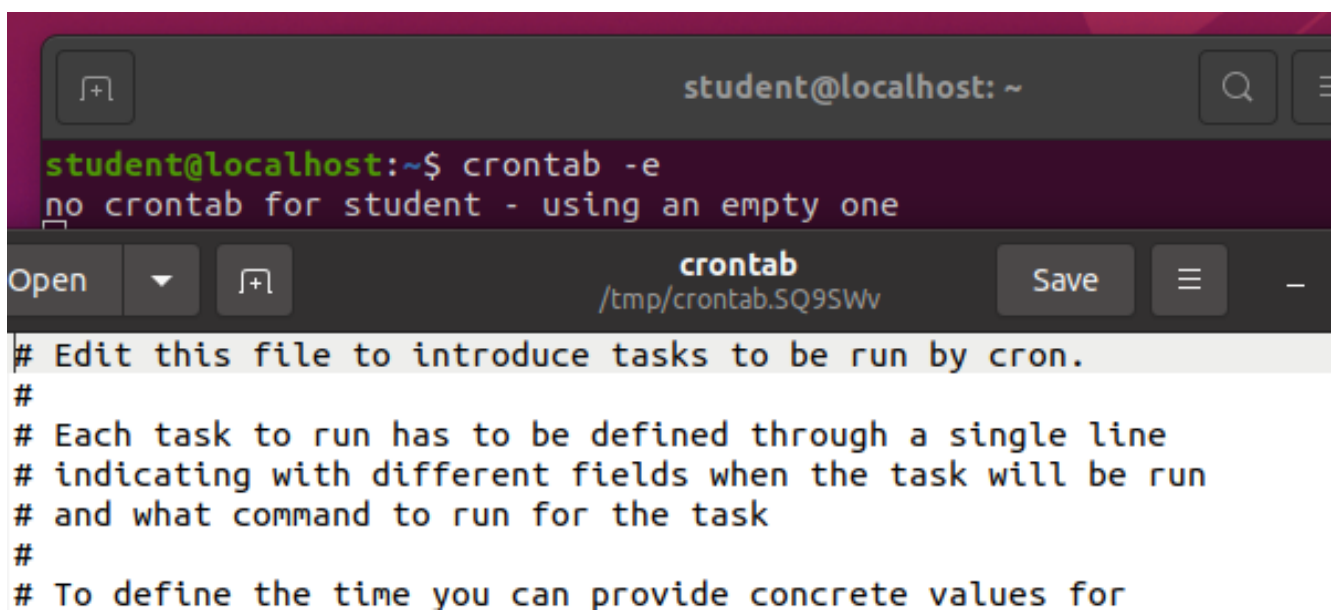
3. After that, open the crontab using gedit by editing the file **.profile** then add this line **:export EDITOR=gedit** to the file and save it, as figure 4 show .



The screenshot shows a terminal window with the prompt `student@localhost: ~/Desktop`. The command `student@localhost:~/Desktop$ sudo gedit ~/.profile` is entered. Below the command line, a password prompt `[sudo] password for student:` is visible. The gedit editor window is open, showing the file `~/.profile`. The content of the file is as follows:

```
26 PATH= $HOME/.local/bin:$PATH
27 fi
28 export EDITOR=gedit
```

Figure 4: Gedit as the default editor in crontab.



The screenshot shows a terminal window with the prompt `student@localhost: ~`. The command `student@localhost:~$ crontab -e` is entered. Below the command line, a message `no crontab for student - using an empty one` is displayed. The gedit editor window is open, showing the file `crontab` located at `/tmp/crontab.SQ9SWv`. The content of the file is as follows:

```
# 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
```

Figure 5: Open crontab in Gedit.

4. To list user's crontab we using command `crontab -l` , as figure 6 show.

```
student@localhost:~$ crontab -l
no crontab for student
student@localhost:~$
```

Figure 6: List user's crontab.

5. After that, we check the permissions of the crontab file, and the permission are read and write by the owner of file, as figure 7 show.

```
student@localhost:~$ ls -l /tmp/crontab.SQ9SWv
total 4
-rw----- 1 student crontab 889 Sep 28 09:24 crontab
```

Figure 7: The permissions of the crontab file.

6. **Questions:** Which users have permissions to schedule cron tasks?
Any user may schedule cron tasks or jobs on a system. The task runs under the user account from which it was created.

3.2 Implementing Periodic Tasks

1. At first, we edit the crontab file to implement the following periodic process. The first process is that creates a file every 1 minute. The filename will be the time of creation which is the first line in figure 8. The second line, as figure 8, does the same command as the first line, but it runs every 2 minutes only on Sunday. The third line will be the same as the second line job, but it will work only on every system reboot. The last line pings the gateway of our machine every 12 am and 12 pm every day of every week. The ping will log into a file named ping.log under the /var/log directory, as shown in the last line in figure 8.

```
24 |
25 1 * * * * touch /home/student/Desktop/${data +%R}
26 2 * * 0 * touch /home/student/Desktop/${data +%R}
27 @reboot touch /home/student/Desktop/${data +%R}
28 * 0,12 * * * ping 127.0.0.1 >> /var/log/ping.log
```

Figure 8: Add periodic process in crontab file

2. We can check if Crontab is Working by using command **service cron status**, as figure 9 show.

```
adminser@adminser-VirtualBox:~/Desktop$ service cron status
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: >
   Active: active (running) since Sat 2021-10-02 10:17:09 EDT; 10min ago
     Docs: man:cron(8)
    Main PID: 605 (cron)
      Tasks: 1 (limit: 2914)
     Memory: 456.0K
    CGroup: /system.slice/cron.service
            └─605 /usr/sbin/cron -f

Oct 02 10:17:09 adminser-VirtualBox systemd[1]: Started Regular background prog>
Oct 02 10:17:09 adminser-VirtualBox cron[605]: (CRON) INFO (pidfile fd = 3)
Oct 02 10:17:10 adminser-VirtualBox cron[605]: (CRON) INFO (Running @reboot job>
lines 1-13/13 (END)
```

Figure 9: Status of crontab

3.3 Log files

1. **Question:** Where can find most of the log files in Linux?
Almost all logfiles are located under **/var/log directory** and its sub-directories on Linux.

2. Linux provides a lot of different types of logs by default. These files are generally located at `/var/log`, the figure 10 show some of example .

```
student@localhost:~/Desktop$ ls /var/log
alternatives.log      auth.log.3.gz        bootstrap.log         dmesg.4.gz           kern.log              syslog.1              vboxadd-install.log
alternatives.log.1    auth.log.4.gz        bttmp                dpkg.log             kern.log.1           syslog.2.gz           vboxadd-setup.log
apache2              boot.log             bttmp.1             dpkg.log.1          kern.log.2.gz        syslog.3.gz           vboxadd-setup.log.1
apport.log           boot.log.1          cups                faillog             kern.log.3.gz        syslog.4.gz           vboxadd-setup.log.2
apport.log.1         boot.log.2          dist-upgrade        fontconfig.log      kern.log.4.gz        syslog.5.gz           vboxadd-setup.log.3
apport.log.2.gz      boot.log.3          dmesg              gdm3               lastlog             syslog.6.gz           vboxadd-setup.log.4
apt                 boot.log.4          dmesg.0            gpu-manager.log     openvpn              syslog.7.gz           wtmp
auth.log             boot.log.5          dmesg.1.gz         hp                 private             ubuntu-advantage.log
auth.log.1           boot.log.6          dmesg.2.gz         installer          speech-dispatcher   ubuntu-advantage.log.1
auth.log.2.gz        boot.log.7          dmesg.3.gz         journal            syslog              unattended-upgrades
```

Figure 10: Type of log file.

3. We can use command **lastlog** to prints the contents of the last login log ex: The login-name, port, and last login time will be printed, as figure 11 show .

```
student@localhost:~/Desktop$ lastlog
Username      Port      From      Latest
root          *Never   logged in**
daemon        *Never   logged in**
bin           *Never   logged in**
sys           *Never   logged in**
sync          *Never   logged in**
games         *Never   logged in**
man           *Never   logged in**
lp            *Never   logged in**
mail          *Never   logged in**
news          *Never   logged in**
uucp          *Never   logged in**
proxy         *Never   logged in**
www-data      *Never   logged in**
backup        *Never   logged in**
list          *Never   logged in**
irc           *Never   logged in**
gnats         *Never   logged in**
nobody        *Never   logged in**
systemd-network *Never   logged in**
systemd-resolve *Never   logged in**
systemd-timesync *Never   logged in**
messagebus    *Never   logged in**
syslog        *Never   logged in**
_apt          *Never   logged in**
tss           *Never   logged in**
```

Figure 11: Type of log file.

4. **Question :** What does log files rotation mean? log rotation is an automated process used in system administration in which log files are compressed, moved (archived), renamed or deleted once they are too old or too big (there can be other metrics that can apply here). New incoming log data is directed into a new fresh file (at the same location).
- Question :** Is it possible to implement it using cron? How? Yes, by edit the crontab file and add for example this line : `*/10 * * * * /etc/cron.daily/logrotate`, which will trigger the rotation after every 10 minutes.
5. **Question:** Write a bash shell script that removes all the log files under the directory `/var/log` that have not been accessed in a week, figure 12 show the bash code.

```
1 #!/bin/bash
2
3 # Usage: cleanup_old_logs <folder> <days>;
4 # Removes all log files in the directory older than a certain number of days
5
6 FOLDER=/var/log
7 N_DAYS=7
8
9 # Validate
10 if [ "$FOLDER" == "" ] || [ "$N_DAYS" == "" ]
11 then
12 echo "Usage: $0 folder number_of_days"
13 exit 1
14 fi
15
16 if [ ! -d "$FOLDER" ]
17 then
18 echo "$FOLDER is not a directory"
19 exit 2
20 fi
21
22 # Remove
23 echo "Deleting files in $FOLDER older than $N_DAYS days"
24 sudo find $FOLDER/* -mtime +$N_DAYS -exec rm -R {} \;
```

Figure 12: Bash code

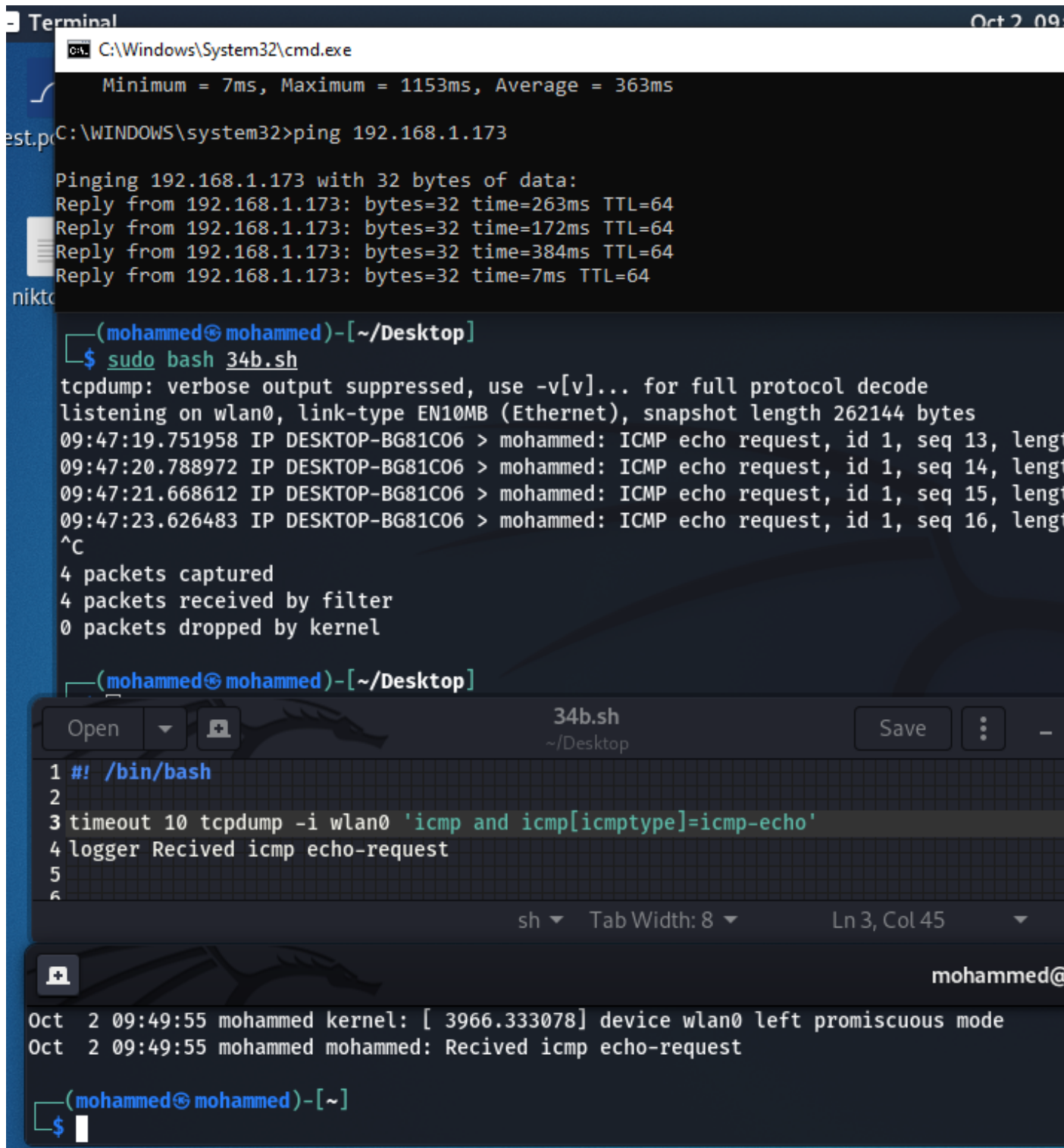
3.4 Syslog messages

1. In first, we display the basic information related to syslog daemon running on your machine by printout the syslog file under etc directory as shown in 13.

```
student@localhost:~$ cat /var/log/syslog
Sep 28 08:11:42 6qCz1bHwwuXwH rsyslogd: [origin software="rsyslogd" swVersion="8.2001.0" x-pid="559"
Sep 28 08:11:42 6qCz1bHwwuXwH networkd-dispatcher[908]: WARNING: systemd-networkd is not running, out
Sep 28 08:11:42 6qCz1bHwwuXwH systemd[1]: Started Dispatcher daemon for systemd-networkd.
Sep 28 08:11:42 6qCz1bHwwuXwH systemd[1]: logrotate.service: Succeeded.
Sep 28 08:11:42 6qCz1bHwwuXwH systemd[1]: Finished Rotate log files.
Sep 28 08:11:43 6qCz1bHwwuXwH systemd[1]: Started Snap Daemon.
Sep 28 08:11:43 6qCz1bHwwuXwH systemd[1]: Starting Wait until snapd is fully seeded...
Sep 28 08:11:43 6qCz1bHwwuXwH systemd[1]: Finished Wait until snapd is fully seeded.
Sep 28 08:11:43 6qCz1bHwwuXwH systemd[1]: Condition check resulted in Auto import assertions from blo
Sep 28 08:11:43 6qCz1bHwwuXwH ModemManager[676]: <info> [base-manager] couldn't check support for de
Sep 28 08:11:43 6qCz1bHwwuXwH systemd[1]: man-db.service: Succeeded.
Sep 28 08:11:43 6qCz1bHwwuXwH systemd[1]: Finished Daily man-db regeneration.
Sep 28 08:11:44 6qCz1bHwwuXwH rc.local[977]: dpkg-query: package 'openssh-server' is not installed an
Sep 28 08:11:44 6qCz1bHwwuXwH rc.local[977]: Use dpkg --info (= dpkg-deb --info) to examine archive f
Sep 28 08:11:44 6qCz1bHwwuXwH rc.local[762]: /usr/sbin/dpkg-reconfigure: openssh-server is not instal
Sep 28 08:11:44 6qCz1bHwwuXwH systemd[1]: rc-local.service: Control process exited, code=exited, stat
Sep 28 08:11:44 6qCz1bHwwuXwH systemd[1]: rc-local.service: Failed with result 'exit-code'.
Sep 28 08:11:44 6qCz1bHwwuXwH systemd[1]: Failed to start /etc/rc.local Compatibility.
Sep 28 08:11:44 6qCz1bHwwuXwH systemd[1]: Starting Hold until boot process finishes up...
Sep 28 08:11:45 6qCz1bHwwuXwH snapd[562]: storehelpers.go:551: cannot refresh: snap has no updates av
Sep 28 08:11:45 6qCz1bHwwuXwH snapd[562]: autorefresh.go:513: auto-refresh: all snaps are up-to-date
Sep 28 08:11:51 6qCz1bHwwuXwH systemd[1]: NetworkManager-dispatcher.service: Succeeded.
Sep 28 08:11:56 6qCz1bHwwuXwH vboxadd[1418]: VirtualBox Guest Additions: Look at /var/log/vboxadd-set
Sep 28 08:11:56 6qCz1bHwwuXwH vboxadd[1418]: went wrong
Sep 28 08:12:02 6qCz1bHwwuXwH systemd[1]: systemd-fsckd.service: Succeeded.
Sep 28 08:12:04 6qCz1bHwwuXwH systemd-timesyncd[506]: Initial synchronization to time server 91.189.9
Sep 28 08:12:04 6qCz1bHwwuXwH systemd[1]: Starting Online ext4 Metadata Check for All Filesystems...
Sep 28 08:12:04 6qCz1bHwwuXwH systemd[1]: e2scrub_all.service: Succeeded.
Sep 28 08:12:04 6qCz1bHwwuXwH systemd[1]: Finished Online ext4 Metadata Check for All Filesystems.
Sep 28 08:12:08 6qCz1bHwwuXwH vboxadd[1436]: VirtualBox Guest Additions: Running kernel modules will
Sep 28 08:12:08 6qCz1bHwwuXwH vboxadd[1436]: the system is restarted
Sep 28 08:12:09 6qCz1bHwwuXwH systemd[1]: systemd-hostnamed.service: Succeeded.
Sep 28 08:12:09 6qCz1bHwwuXwH systemd[1]: Finished vboxadd.service.
Sep 28 08:12:09 6qCz1bHwwuXwH systemd[1]: Starting GNOME Display Manager...
Sep 28 08:12:09 6qCz1bHwwuXwH systemd[1]: Starting vboxadd-service.service...
Sep 28 08:12:09 6qCz1bHwwuXwH vboxadd-service[1487]: vboxadd-service.sh: Starting VirtualBox Guest Ad
Sep 28 08:12:09 6qCz1bHwwuXwH vboxadd-service.sh: Starting VirtualBox Guest Addition service.
Sep 28 08:12:09 6qCz1bHwwuXwH kernel: [ 45.917650] 05:12:09.533410 main VBoxService 6.1.16 r140
Sep 28 08:12:09 6qCz1bHwwuXwH kernel: [ 45.917650] 05:12:09.5334
Sep 28 08:12:09 6qCz1bHwwuXwH kernel: [ 45.918271] 05:12:09.534543 main OS Product: Linux
Sep 28 08:12:09 6qCz1bHwwuXwH kernel: [ 45.918882] 05:12:09.535046 main OS Release: 5.11.0-36-g
Sep 28 08:12:09 6qCz1bHwwuXwH kernel: [ 45.919949] 05:12:09.535900 main OS Version: #40~20.04.1
Sep 28 08:12:09 6qCz1bHwwuXwH kernel: [ 45.923446] 05:12:09.539135 main Executable: /opt/VBoxGu
```

Figure 13: syslog file

2. Then, we write a bash shell script that generates a Syslog message if our machine has pinged as shown in figure 14, in the top, we have the ping machine under that tap, there is the executing of the code. Then we have the code which has in the first line Tcpdump command to know or notice us if this machine has been pinged, or not, the second line will execute and run which is logger, this command sends the statement to the system log file as shown in the last tap of the figure.



```
Terminal
C:\Windows\System32\cmd.exe
Minimum = 7ms, Maximum = 1153ms, Average = 363ms
C:\WINDOWS\system32>ping 192.168.1.173
Pinging 192.168.1.173 with 32 bytes of data:
Reply from 192.168.1.173: bytes=32 time=263ms TTL=64
Reply from 192.168.1.173: bytes=32 time=172ms TTL=64
Reply from 192.168.1.173: bytes=32 time=384ms TTL=64
Reply from 192.168.1.173: bytes=32 time=7ms TTL=64

(mohammed@mohammed)-[~/Desktop]
$ sudo bash 34b.sh
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on wlan0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
09:47:19.751958 IP DESKTOP-BG81C06 > mohammed: ICMP echo request, id 1, seq 13, length 32
09:47:20.788972 IP DESKTOP-BG81C06 > mohammed: ICMP echo request, id 1, seq 14, length 32
09:47:21.668612 IP DESKTOP-BG81C06 > mohammed: ICMP echo request, id 1, seq 15, length 32
09:47:23.626483 IP DESKTOP-BG81C06 > mohammed: ICMP echo request, id 1, seq 16, length 32
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel

(mohammed@mohammed)-[~/Desktop]
34b.sh
~/Desktop
1 #! /bin/bash
2
3 timeout 10 tcpdump -i wlan0 'icmp and icmp[icmptype]=icmp-echo'
4 logger Recived icmp echo-request
5
6
sh Tab Width: 8 Ln 3, Col 45
mohammed@
Oct 2 09:49:55 mohammed kernel: [ 3966.333078] device wlan0 left promiscuous mode
Oct 2 09:49:55 mohammed mohammed: Recived icmp echo-request

(mohammed@mohammed)-[~]
$
```

Figure 14: Bash code for syslog message

3. **Question :** Generate a syslog message by your machine whose destination is your neighboring machine and make sure that this message has been captured by the neighboring machine.

At first, we edit the (Rsyslog. conf) file under (/Etc) directory in server machine and add those lines as shown in figure 15, so we open UDP port 512 to receive from our neighbor, then restart the Rsyslog service to admit the change as shown in figure 16, after that, we edit the Rsyslog file in the neighbor machine as figure 17 shows, we put the IP of the server and his port also @ means that its UDP, in the last we to send Syslog(logger) from the neighbor to the server as figure 18 show , then we display the Syslog file in the server to sure that the Syslog has received figures 19 shows the result of the file.

```
92 *.emerg                                     :oml
93 $ModLoad imudp
94 $UDPServerRun 514
```

Figure 15: edit rsyslog file (server)

```
(mohammed@mohammed)-[~/Desktop]
$ sudo gedit /etc/rsyslog.conf
(mohammed@mohammed)-[~/Desktop]
$ sudo systemctl restart rsyslog.service
(mohammed@mohammed)-[~/Desktop]
$ sudo systemctl enable rsyslog.service
Synchronizing state of rsyslog.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable rsyslog
```

Figure 16: restart syslog service(server)

```
$INCLUDECOMING /etc/rsyslog
*. *@192.168.1.173:514
```

Figure 17: edit rsyslog file(neighbor)

```
user@user-virtualbox:/etc$ sudo systemctl restart rsyslog.service
user@user-virtualbox:/etc$ sudo systemctl enable rsyslog.service
Synchronizing state of rsyslog.service with SysV service script with /lib/systemd/s
Executing: /lib/systemd/systemd-sysv-install enable rsyslog
user@user-virtualbox:/etc$ logger "This message is a syslog message from the client"
```

Figure 18: send syslog from neighbor to server

```

(mohammed@moammed)~[~/Desktop] files
$ sudo tail /var/log/syslog
Oct  2 11:29:28 mohammed systemd[1]: /lib/systemd/system/plymouth-start.service:16: Unit configured
ecycle management for the service. Please update your service to use a safer KillMode=, such as 'm
eventually be removed.
Oct  2 11:29:20 user-virtualbox PackageKit: daemon start
Oct  2 11:29:28 mohammed systemd[1]: Reloading.
Oct  2 11:29:28 mohammed systemd[1]: /lib/systemd/system/plymouth-start.service:16: Unit configured
ecycle management for the service. Please update your service to use a safer KillMode=, such as 'm
eventually be removed.
Oct  2 11:29:28 mohammed systemd[1]: Reloading.
Oct  2 11:29:29 mohammed systemd[1]: /lib/systemd/system/plymouth-start.service:16: Unit configured
ecycle management for the service. Please update your service to use a safer KillMode=, such as 'm
eventually be removed.
Oct  2 11:29:20 user-virtualbox dbus-daemon[376]: [system] Successfully activated service 'org.fre
Oct  2 11:29:20 user-virtualbox systemd[1]: Started PackageKit Daemon.
Oct  2 11:29:53 user-virtualbox user: This message is a syslog message from the client to server
Oct  2 11:30:02 user-virtualbox CRON[2779]: (root) CMD ([ -x /etc/init.d/anacron ] && if [ ! -d /r
) is sent to everybody logged in.

```

Figure 19: display syslog file in server

4 Conclusion

Crontab stands for cron table which contains a list of tasks that are scheduled to run at a regular interval of time. These tasks are then executed by a daemon called cron. All users on the machine can have a separate crontab that helps them schedule tasks. The Syslog protocol is a classic for system administrators or Linux engineers willing to have a deeper understanding of how logging works on a server. In This experiment, we use crontab for many task and we use the Syslog also to send the Syslog from client to server.

5 References

<https://askubuntu.com/questions/1075167/gedit-as-the-default-editor-in-crontab>

<https://techbast.com/2021/05/linux-how-to-install-and-use-crontab-on-ubuntu-server.html> <https://www.redhat.com/en/tech-tips-by-topic/linux-tasks-cron> <https://www.linuxfordevices.com/tutorials/linux/crontabs-in-linux> <https://www.linuxtechi.com/managing-linux-log-files-using-logrotate/:text=Logrotate>