MIDTERM REPORT: Linux System Security

CENG3544, COMPUTER AND NETWORK SECURITY

Beyza Kurt
beyzakurt@posta.mu.edu.tr

Sunday 7th June, 2020

Our main goal in this task is to improve our scripting skills while improving our knowledge of the system and gaining application experience in this field. We will work with our thoughts on antivirus software, SIEM, and our script targeting system analysis.

1 Introduction

In this lab, again I will work with Ubuntu, which we established as a "victim" in previous labs. I will share my thoughts with my experience related to antivirus software, I will explain why Linux preferred SIEM with my own thoughts and research, and I will share in detail the new version of the changes and additions I made in the script I prepared in the midterm assignment. In addition, I will share the interactive script I wrote to read encrypted reports that prepared by my bash script, "final.sh". The operating system I am working on features:

- VMware WorkStation 15 Player
- Ubuntu 19.10 Desktop
- Number of processor cores: 2
- Hard Disk Capacity: 10 GB
- Memory: 2048 MB

2 Assignments

1) Write your thoughts about Malware (Antivirus etc) tools.
2) Why is Linux preferred in SOC as SIEM?
3) Write and enhance your Linux script in the Midterm Assignment so that it
   a. Gives a summary report of the system to file(s)
   b. Periodically runs
   c. keeps the current status of the system in USB (or cloud?) (BONUS)

d. Uses hashing
e. Encrypts the current status (file(s))with your key)

2.1 Assignment 1 (“Write your thoughts about Malware (Antivirus etc) tools.”)

I used antivirus and system care programs on the high school the last time. One day, after scanning my computer, I noticed that my graphics card drivers were deleted. Then, I did not use any program that I could not see what it was doing in the background. There are open source antivirus programs right now, but I try to not need them by paying attention to my security on my own. Because I suspect that it might not suffice to protect against rapidly changing viruses.

2.2 Assignment 2 (“Why is Linux preferred in SOC as SIEM?”)

SIEM (Security Information Event Management) is a very useful technology to analysis and report to security events. As we know, on an operating system like Linux, we can see all events performed on the system. This gives an opportunity to us for check them easily. SIEM is a perfect match for this situation. In my opinion, Linux preffers SIEM, because SIEM gathers Security Event Management (SEM) and Security Information Management (SIM) under one roof. With SEM, threat monitoring from an incident response can be done and event correlation can be done by analyzing log data and events in real-time. A report that contains analysis of log data can be created with SIM. Becoming more equipped is possible by using SIEM, which can do things that can be done by both of them.

2.3 Assignment 3 (“Write and enhance your Linux script in the Midterm Assignment”)

2.3.1 Linux Script: final.sh

This code, which enables us to obtain information about the system and follow the changes in the system, is included in additional files with the name "final.sh". In this report, I will explain the whole code piece by piece.

Step 1: The Beginning

```
#!/bin/bash
startLoc=$(pwd)
folder_name="Kurt_Beyza_Final"
if [ ! -d "$folder_name" ]; then
	mkdir $folder_name
fi

cd $folder_name
home="$startLoc/Kurt_Beyza_Final"
```

This is the beginning section of the code. Firstly, I am defining that this is a bash script. Then, I am keeping the script's location as "startLoc", I want to open a folder, "Kurt_Beyza_Final", here. First of all, I am checking if the folder exists and if not, I am creating it. Then, I am going into this folder and am keeping this location as "home" like the home folder of the future files.

Step 2: Checking Requirements

```
declare -a arr=("syslog" "auth.log" "kern.log" "messages")
if [ ! -f report.gpg ]; then
	#---first time-------------------------------------------------------
	<some codes>
else
	<some codes>
fi
```

In this part, I am declaring an array as "arr". In this array, I am keeping log files' names to analyze. Then, I have a section to see if this script is running for the first time. To understand this, I am checking if the encrypted report exists. This encrypted report is generated by running this script. If it is not exist, this means that this is the first time for the script. I hide the codes here, because I will explain it in detail below.

A. if [! -f report.gpg]

A.1. run regularly in the background

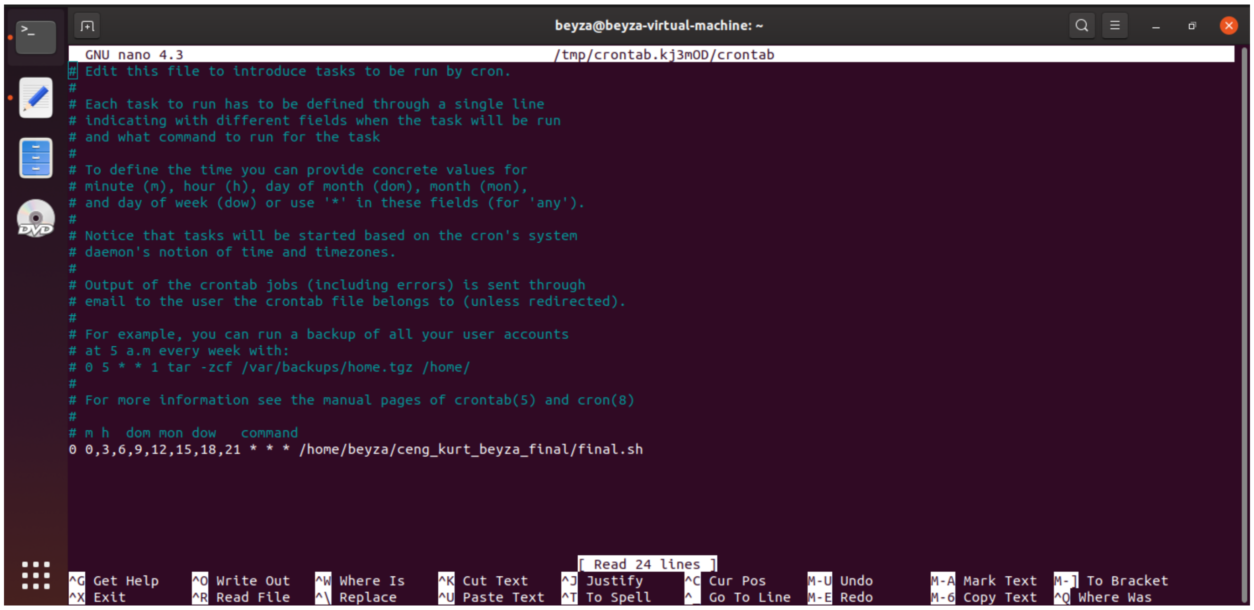
```
echo "I can work in a planned way at certain time intervals!"
read -p "If you want me to work in the background every 3 hours, just say yes: " y
echo ""
if [ ${y,,} == "y" ] || [ ${y,,} == "yes" ]; then
	echo "add this line to the end of the file:"
	echo "0 0,3,6,9,12,15,18,21 * * * $startLoc/final.sh"
	read -p "(if you copied the line, press enter)" enter
	crontab -e
	echo ""
else
	echo "Okey. Okay. If you want it later"
	echo "run 'crontab -e' and add this line to end of the file:"
	echo "0 0,3,6,9,12,15,18,21 * * * $startLoc/final.sh"
	read -p "(press enter to continue)" enter
	echo ""
fi
```

First of all there is an interactive section to make this script **run regularly in the background**. The user can set this up to run <u>every 3 hours</u>. It needs to access the file "crontab" using the "crontab -e" command and enter the line "0 0,3,6,9,12,15,18,21 * * * $ startLoc / final.sh" which I gave there on the screen. If the user tell that "no", then I give the informations about how the user can do this later and I continue.

Below you can see what this section prints by running the script, and also there is the screenshot of the "crontab" file.

- If the answer is “y”:

```
I can work in a planned way at certain time intervals!
If you want me to work in the background every 3 hours, just say yes: y

add this line to the end of the file:
0 0,3,6,9,12,15,18,21 * * * /home/beyza/ceng_kurt_beyza_final/final.sh
(if you copied the line, press enter)
```



```
beyza@beyza-virtual-machine: ~
  GNU nano 4.3                            /tmp/crontab.kj3mOD/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
0 0,3,6,9,12,15,18,21 * * * /home/beyza/ceng_kurt_beyza_final/final.sh

                                              [ Read 24 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo       M-A Mark Text  M-] To Bracket
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line  M-E Redo       M-6 Copy Text  ^Q Where Was
```

Figure 1 the screenshot of the "crontab" file

- If the answer is “n”:

```
I can work in a planned way at certain time intervals!
If you want me to work in the background every 3 hours, just say yes: n

Okay. If you want it later
run 'crontab -e' and add this line to end of the file:
0 0,3,6,9,12,15,18,21 * * * /home/beyza/ceng_kurt_beyza_final/final.sh
(press enter to continue)
```

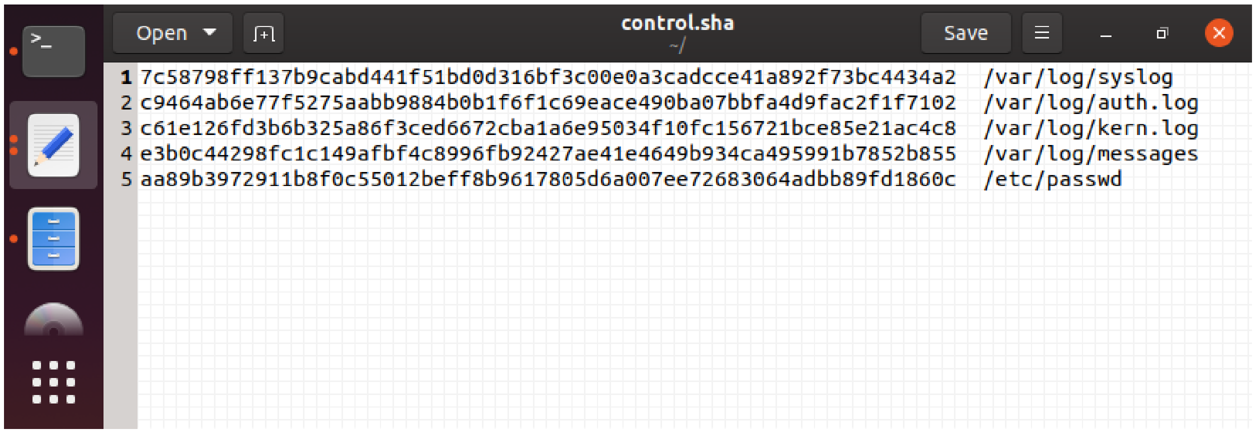
A.2. backup and hashing log files

```
mkdir REPORT
cd REPORT

for x in "${arr[@]}"
do
        sudo cp /var/log/$x $x.last
        sudo chmod +rwx $x.last
        sudo sha256sum /var/log/$x >> control.sha
done

sudo cp /etc/passwd passwd.last
sudo chmod +rwx passwd.last
sudo sha256sum /etc/passwd >> control.sha
```

In this section, first, I create and enter a folder to keep the last records of the files we will look. Then, I return the elements of the "arr" array that I defined earlier and perform the same operation for all elements. I do the same for the "passwd" file. But I do it out of the loop because its path is different.

Operations performed:

1. I am copying the original of the file to the current location as the last record.
2. I am giving the permissions for reading, writing and executing to the file.
3. I am using hashing to understand later if the original file is changed.



```
7c58798ff137b9cabd441f51bd0d316bf3c00e0a3cadcce41a892f73bc4434a2  /var/log/syslog
c9464ab6e77f5275aabb9884b0b1f6f1c69eace490ba07bbfa4d9fac2f1f7102  /var/log/auth.log
c61e126fd3b6b325a86f3ced6672cba1a6e95034f10fc156721bce85e21ac4c8  /var/log/kern.log
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855  /var/log/messages
aa89b3972911b8f0c55012beff8b9617805d6a007ee72683064adbb89fd1860c  /etc/passwd
```

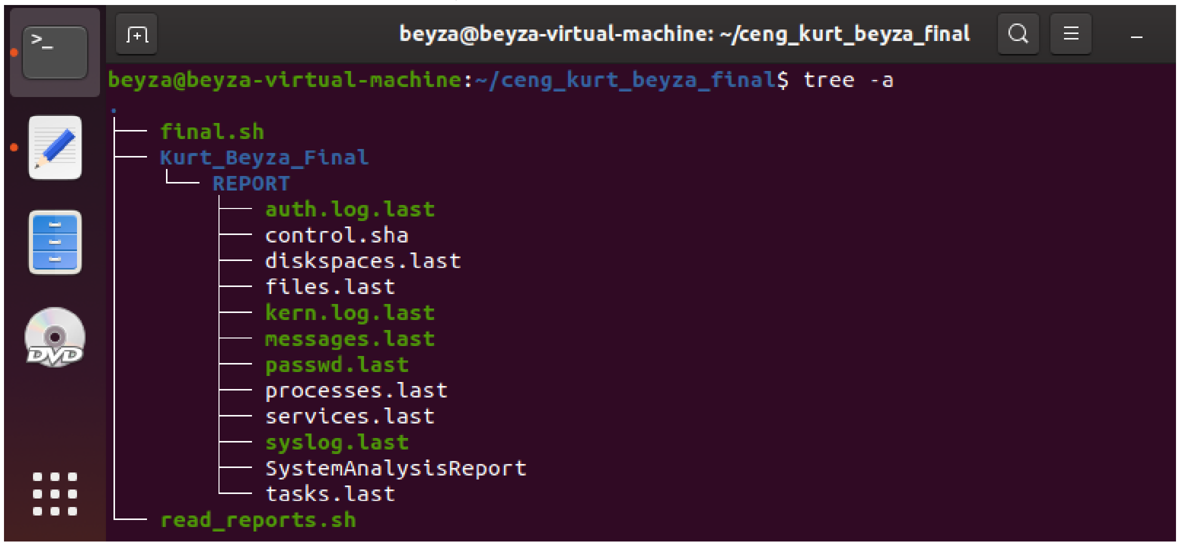
Figure 2 the screenshot of the "control.sha" file

A.3. backup for files, disk spaces, processes, services and tasks

```
cd ~
sudo tree -a > $home/REPORT/files.last
sudo df -h > $home/REPORT/diskspaces.last
sudo ps -aux > $home/REPORT/processes.last
sudo systemctl list-unit-files --type service --all > $home/REPORT/services.last
for user in `cat /etc/passwd | cut -d":" -f1`;
do
        sudo crontab -l -u $user &>> $home/REPORT/tasks.last
done
```

I go to the homepage to get the correct results from the codes to be run in this section. I am keeping last status of files, disk spaces, processes, services and tasks to compare later.

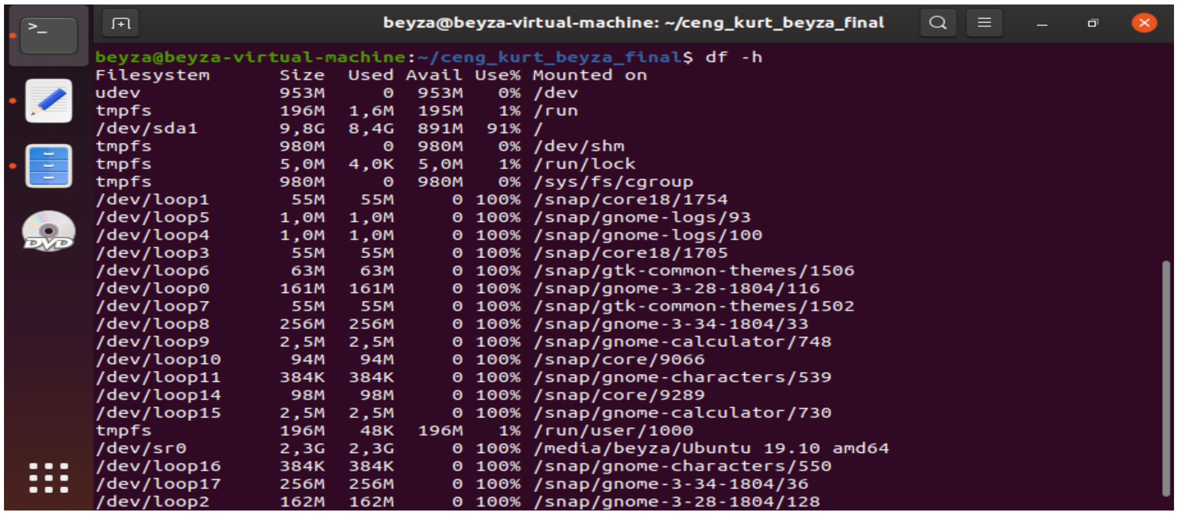
- tree -a: shows all files in a tree shape



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ tree -a
.
├── final.sh
├── Kurt_Beyza_Final
│   └── REPORT
│       ├── auth.log.last
│       ├── control.sha
│       ├── diskspaces.last
│       ├── files.last
│       ├── kern.log.last
│       ├── messages.last
│       ├── passwd.last
│       ├── processes.last
│       ├── services.last
│       ├── syslog.last
│       ├── SystemAnalysisReport
│       └── tasks.last
└── read_reports.sh
```

Figure 3 the screenshot of "tree -a" command

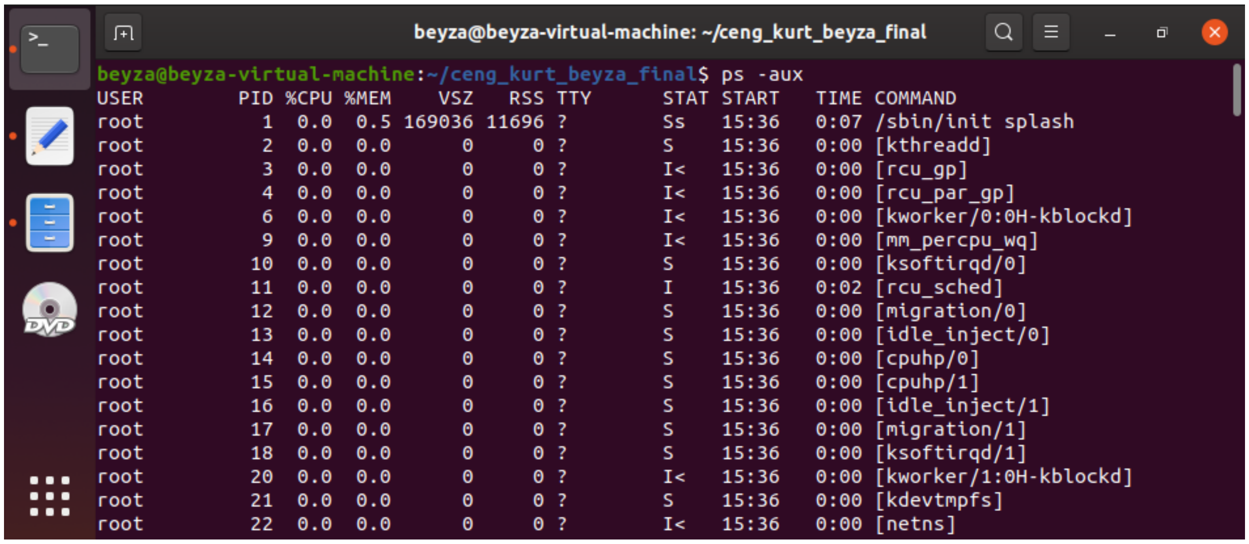
- df -h: shows disk usage with “Filesystem, Size, Used, Avail, Use%, Mounted on” headings



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            953M     0  953M   0% /dev
tmpfs           196M  1,6M  195M   1% /run
/dev/sda1       9,8G  8,4G  891M  91% /
tmpfs           980M     0  980M   0% /dev/shm
tmpfs           5,0M  4,0K  5,0M   1% /run/lock
tmpfs           980M     0  980M   0% /sys/fs/cgroup
/dev/loop1       55M   55M     0 100% /snap/core18/1754
/dev/loop5      1,0M  1,0M     0 100% /snap/gnome-logs/93
/dev/loop4      1,0M  1,0M     0 100% /snap/gnome-logs/100
/dev/loop3       55M   55M     0 100% /snap/core18/1705
/dev/loop6       63M   63M     0 100% /snap/gtk-common-themes/1506
/dev/loop0      161M  161M     0 100% /snap/gnome-3-28-1804/116
/dev/loop7       55M   55M     0 100% /snap/gtk-common-themes/1502
/dev/loop8      256M  256M     0 100% /snap/gnome-3-34-1804/33
/dev/loop9      2,5M  2,5M     0 100% /snap/gnome-calculator/748
/dev/loop10      94M   94M     0 100% /snap/core/9066
/dev/loop11     384K  384K     0 100% /snap/gnome-characters/539
/dev/loop14      98M   98M     0 100% /snap/core/9289
/dev/loop15     2,5M  2,5M     0 100% /snap/gnome-calculator/730
tmpfs           196M   48K  196M   1% /run/user/1000
/dev/sr0        2,3G  2,3G     0 100% /media/beyza/Ubuntu 19.10 amd64
/dev/loop16     384K  384K     0 100% /snap/gnome-characters/550
/dev/loop17     256M  256M     0 100% /snap/gnome-3-34-1804/36
/dev/loop2      162M  162M     0 100% /snap/gnome-3-28-1804/128
```

Figure 4 the screenshot of "df -h" command

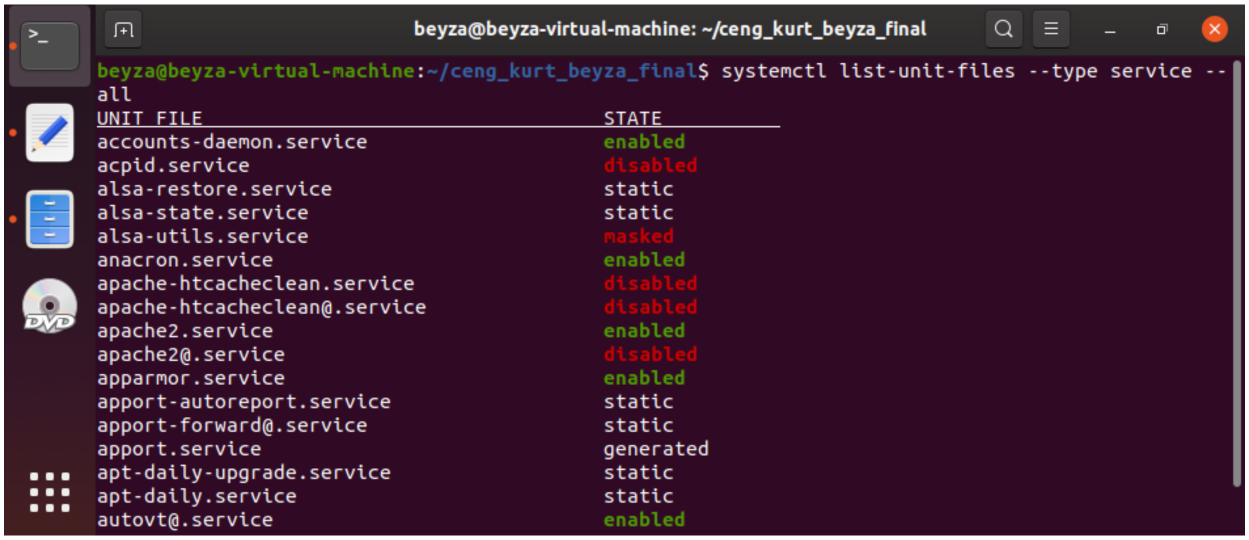
- ps -aux: shows process with "USER, PID,% CPU,% MEM, VSZ, RSS, TTY, STAT, START, TIME, COMMAND" headings



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ ps -aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.5 169036 11696 ?        Ss   15:36   0:07 /sbin/init splash
root           2  0.0  0.0      0     0 ?        S    15:36   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   15:36   0:00 [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   15:36   0:00 [rcu_par_gp]
root           6  0.0  0.0      0     0 ?        I<   15:36   0:00 [kworker/0:0H-kblockd]
root           9  0.0  0.0      0     0 ?        I<   15:36   0:00 [mm_percpu_wq]
root          10  0.0  0.0      0     0 ?        S    15:36   0:00 [ksoftirqd/0]
root          11  0.0  0.0      0     0 ?        I    15:36   0:02 [rcu_sched]
root          12  0.0  0.0      0     0 ?        S    15:36   0:00 [migration/0]
root          13  0.0  0.0      0     0 ?        S    15:36   0:00 [idle_inject/0]
root          14  0.0  0.0      0     0 ?        S    15:36   0:00 [cpuhp/0]
root          15  0.0  0.0      0     0 ?        S    15:36   0:00 [cpuhp/1]
root          16  0.0  0.0      0     0 ?        S    15:36   0:00 [idle_inject/1]
root          17  0.0  0.0      0     0 ?        S    15:36   0:00 [migration/1]
root          18  0.0  0.0      0     0 ?        S    15:36   0:00 [ksoftirqd/1]
root          20  0.0  0.0      0     0 ?        I<   15:36   0:00 [kworker/1:0H-kblockd]
root          21  0.0  0.0      0     0 ?        S    15:36   0:00 [kdevtmpfs]
root          22  0.0  0.0      0     0 ?        I<   15:36   0:00 [netns]
```

Figure 5 the screenshot of "ps -aux" command

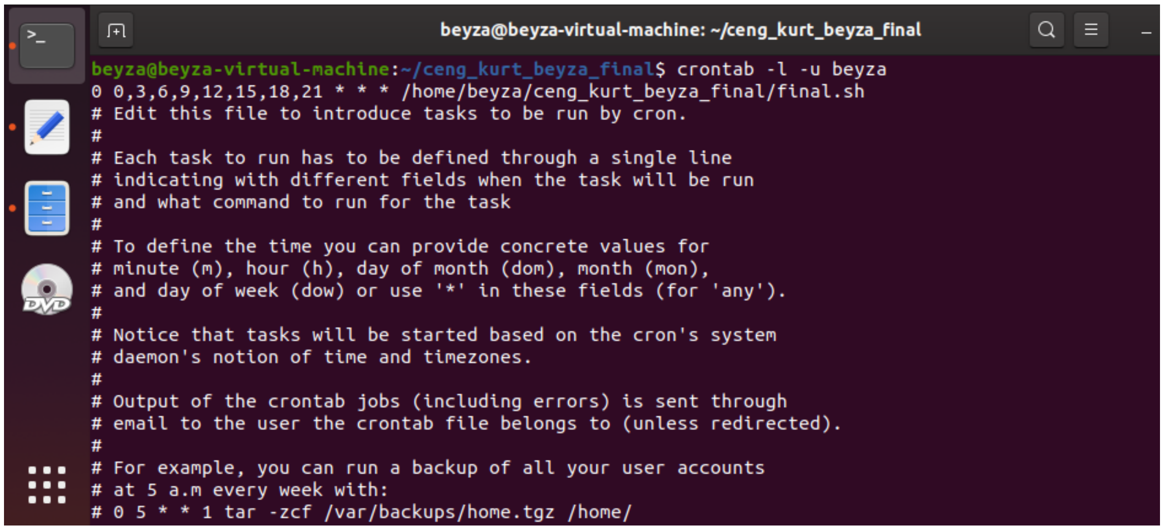
- systemctl list-unit-files --type service --all: shows services with "unit file, state" headings



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ systemctl list-unit-files --type service --all
UNIT FILE                                    STATE
accounts-daemon.service                      enabled
acpid.service                                disabled
alsa-restore.service                         static
alsa-state.service                           static
alsa-utils.service                           masked
anacron.service                              enabled
apache-htcacheclean.service                  disabled
apache-htcacheclean@.service                 disabled
apache2.service                              enabled
apache2@.service                             disabled
apparmor.service                             enabled
apport-autoreport.service                    static
apport-forward@.service                      static
apport.service                               generated
apt-daily-upgrade.service                    static
apt-daily.service                            static
autovt@.service                              enabled
```

Figure 6 the screenshot of "systemctl list-unit-files --type service --all" command

- crontab -l -u $user: shows the user's the jobs that are defined to run in the background



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ crontab -l -u beyza
0 0,3,6,9,12,15,18,21 * * * /home/beyza/ceng_kurt_beyza_final/final.sh
# 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/
```

Figure 7 the screenshot of the "crontab -l -u beyza" command

A.4. Encrypt Files

```
# Encrypt folder
cd $home
tar czf report.tar.gz REPORT/
sudo rm -r REPORT/
#gpg --batch --output report.gpg --symmetric report.tar.gz
gpg --batch --output report.gpg --passphrase mypassword --symmetric report.tar.gz
#echo RELOADAGENT | gpg-connect-agent
rm report.tar.gz
```

This section is about encryption. First of all, I go to the main folder I set. I am compressing the folder "REPORT" which contains the files I created above. I delete the uncompressed version. I encrypt the compressed file using gpg. Since we set this script to run regularly in the background, I use a specific password instead of asking the user for a password in encryption. The commands that I leave as comments must get a password from users every time they run. Finally, I delete the unencrypted compressed file. Now I have only an encrypted report file, “report.gpg”, and I have the latest version of my files.

A.5. Run the script again

```
# Run code again
cd $startLoc
ScriptLoc=$(readlink -f "$0")
exec "$ScriptLoc"
```

After completing all the required files, I restart the script.

B. else

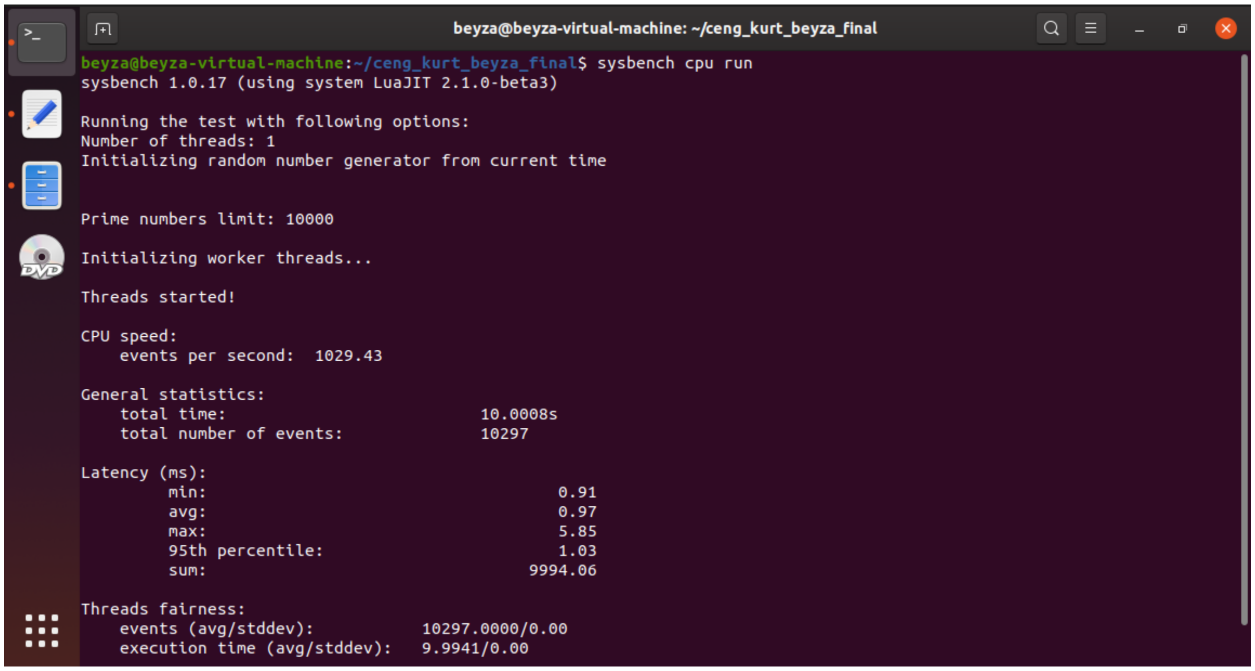
```
#gpg --batch --output report.tar.gz --decrypt report.gpg
gpg --batch --output report.tar.gz --passphrase mypassword --decrypt report.gpg
rm report.gpg
#echo RELOADAGENT | gpg-connect-agent
tar xzf report.tar.gz
rm report.tar.gz
cd REPORT
```

When the script is run again, it first decrypts the encrypted file “report.gpg”. As in "A.5.", I use the same set password to decrypt it. I am deleting the encrypted file for now as I will encrypt the file again after all the processes are finished. I also extract the compressed folder and delete the compressed version. I now have a folder called "REPORT" containing the most recent versions of the files I want.

Step 3: CPU and Memory Benchmark

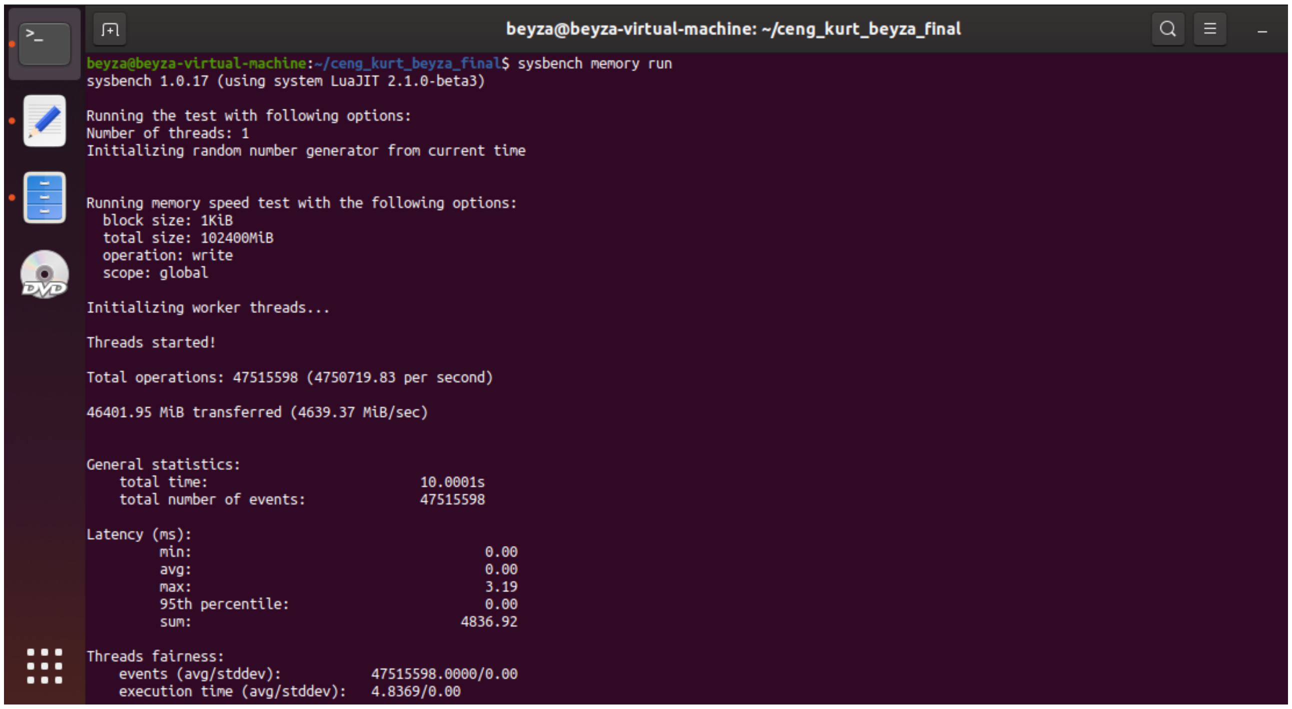
```
echo "SYSTEM ANALYSIS REPORT" > SystemAnalysisReport

echo "***************** BENCHMARKS *****************" >> SystemAnalysisReport
echo "'sysbench' is installing..."
sudo apt install -y sysbench
echo "sysbench' is installed." >> SystemAnalysisReport
echo "***** CPU Benchmark *****" >> SystemAnalysisReport
echo "CPU benchmark is running..."
sysbench cpu run >> SystemAnalysisReport
echo "***** Memory Benchmark *****" >> SystemAnalysisReport
echo "Memory benchmark is running..."
sysbench memory run >> SystemAnalysisReport
```

First of all, I write the string "SYSTEM ANALYSIS REPORT" to the file named "SystemAnalysisReport". I will keep all the future results in this. All strings written in bold above are written to the report. I use the package called "sysbench" to test the performance of CPU and memory. The results of the commands written in bold above are also written to the report.



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ sysbench cpu run
sysbench 1.0.17 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 10000

Initializing worker threads...

Threads started!

CPU speed:
    events per second:  1029.43

General statistics:
    total time:                          10.0008s
    total number of events:              10297

Latency (ms):
         min:                                    0.91
         avg:                                    0.97
         max:                                    5.85
         95th percentile:                        1.03
         sum:                                 9994.06

Threads fairness:
    events (avg/stddev):           10297.0000/0.00
    execution time (avg/stddev):   9.9941/0.00
```

Figure 8 the screenshot of the "sysbench cpu run" command



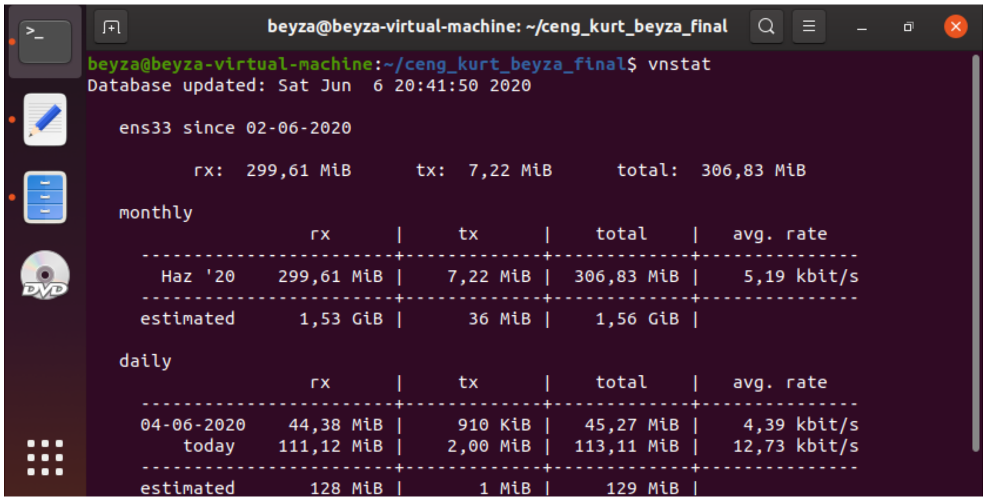
```
beyza@beyza-virtual-machine: ~/ceng_kurt_beyza_final
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ sysbench memory run
sysbench 1.0.17 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Running memory speed test with the following options:
  block size: 1KiB
  total size: 102400MiB
  operation: write
  scope: global

Initializing worker threads...

Threads started!

Total operations: 47515598 (4750719.83 per second)

46401.95 MiB transferred (4639.37 MiB/sec)


General statistics:
    total time:                          10.0001s
    total number of events:              47515598

Latency (ms):
         min:                                    0.00
         avg:                                    0.00
         max:                                    3.19
         95th percentile:                        0.00
         sum:                                 4836.92

Threads fairness:
    events (avg/stddev):           47515598.0000/0.00
    execution time (avg/stddev):   4.8369/0.00
```

Figure 9 the screenshot of the "sysbench memory run" command

Step 4: Network Usage

```
echo "***************** NETWORK *****************" >> SystemAnalysisReport
echo "'vnstat' is installing..."
sudo apt install -y vnstat
echo "'vnstat' is installed." >> SystemAnalysisReport
vnstat >> SystemAnalysisReport
```

I use the "vnstat" package to look at network usage. The bold texts and command's results are written to the report.



```
beyza@beyza-virtual-machine: ~/ceng_kurt_beyza_final
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ vnstat
Database updated: Sat Jun  6 20:41:50 2020

   ens33 since 02-06-2020

          rx:  299,61 MiB      tx:  7,22 MiB      total:  306,83 MiB

   monthly
                     rx      |     tx      |    total    |   avg. rate
     ------------------------+-------------+-------------+---------------
       Haz '20    299,61 MiB |    7,22 MiB |  306,83 MiB |    5,19 kbit/s
     ------------------------+-------------+-------------+---------------
     estimated      1,53 GiB |      36 MiB |    1,56 GiB |

   daily
                     rx      |     tx      |    total    |   avg. rate
     ------------------------+-------------+-------------+---------------
     04-06-2020    44,38 MiB |    910 KiB |   45,27 MiB |    4,39 kbit/s
         today    111,12 MiB |    2,00 MiB |  113,11 MiB |   12,73 kbit/s
     ------------------------+-------------+-------------+---------------
     estimated       128 MiB |      1 MiB |     129 MiB |
```

Figure 10 the screenshot of the "vnstat" command

Step 5: Log files and Passwd file Analysis

```
sudo sha256sum -c control.sha &> results.txt
while IFS= read -r line
do
	s=$(echo "$line" | cut -d':' -f 2)
	if [ $s == "FAILED" ]; then
		f=$(echo "$line" | cut -d':' -f 1)
		if [ $f == "/etc/passwd" ]; then
			f=$(echo $f | cut -d'/' -f 3)
		else
			f=$(echo $f | cut -d'/' -f 4)
		fi
		logAnalysis "$f"

		if [ $f == "passwd" ]; then
			sudo cp /etc/$f $f.last
		else
			sudo cp /var/log/$f $f.last
		fi
	fi
done < "results.txt"
sudo rm results.txt
```

Using the command "sha256sum -c control.sha", I have been checking the original files for changes since the last check and I am printing the results to the "results.txt" file. I am sending the name of the files with the result "FAILED" to the logAnalysis() function. In the next step, I will talk about the content of the function. After the comparison process is completed in the function, I copy the originals that changed to the location I was in, as I did when I first started the script. I delete the "results.txt" file after all the results have been checked and the necessary actions have been taken.

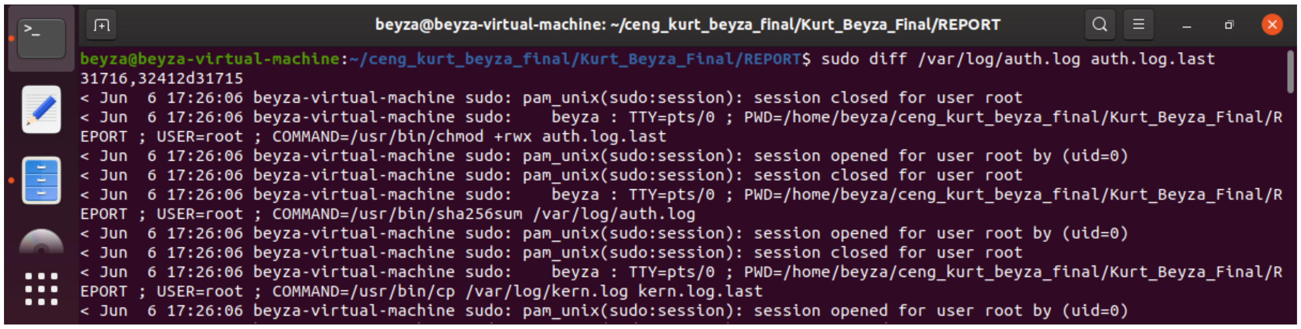
Step 6: logAnalysis()

```
logAnalysis () {
	case $1 in
	"syslog")
		echo "************** SYSLOG **************" >> SystemAnalysisReport
		o=$(grep "Out of memory" /var/log/syslog)
		if [ -z $o ]; then
			echo "Good news! There is no 'Out of memory' error." >>
			SystemAnalysisReport
```

```
                else
                        echo "There is 'Out of memory' error(s):" >> SystemAnalysisReport
                        echo $o >> SystemAnalysisReport
                fi
                ;;
        "auth.log")
                echo "************* AUTH.LOG *************" >> SystemAnalysisReport
                i=$(grep "invalid user" /var/log/auth.log)
                if [ -z $i ]; then
                        echo "Good news! There is no 'Invalid user' error." >>
                        SystemAnalysisReport
                else
                        echo "There is 'Invalid user' error(s):" >> SystemAnalysisReport
                        echo $i >> SystemAnalysisReport
                fi
                s=$(grep "shutdown" /var/log/auth.log)
                if [ -z $s ]; then
                        echo "Good news! There is no 'shutdown' error." >>
                        SystemAnalysisReport
                else
                        echo "There is 'shutdown' error(s):" >> SystemAnalysisReport
                        echo $s >> SystemAnalysisReport
                fi
                ;;
        "kern.log")
                echo "************ KERN.LOG *************" >> SystemAnalysisReport
                ;;
        "messages")
                echo "************* MESSAGES *************" >> SystemAnalysisReport
                ;;
        "passwd")
                echo "************* PASSWD *************" >> SystemAnalysisReport
                echo "Things that have changed since the last check:" >> SystemAnalysisReport
                diff /etc/$1 $1.last >> SystemAnalysisReport
                return 0
                ;;
        esac

        echo "Things that have changed since the last check:" >> SystemAnalysisReport
        diff /var/log/$1 $1.last >> SystemAnalysisReport
}
```

As I said in the previous section, I run this function with the file name. There are cases according to the file name; the function works with the suitable case. There are exceptions for some files, but there are steps for all.

1. First of all, the suitable case will run. Except exceptions I will list below, the title of each file will be written to the report.
    a. I check if there is an "out of memory" error for "**syslog**". If not, I will write the text stating that it does not exist. If there is, I will write the details of the error with a message to the report.
    b. I check whether there are "invalid user" and "shutdown" errors for "**auth.log**". If not, I will write the text stating that it does not exist. If there is, I will write the details of the error with a message to the report.
    c. Since the path of the "**passwd**" file is different from the other files, what others do outside of the 'case' is done here and the function will be completed.
2. Since this function works if the file has changed, the differences of all the changed files are examined with "**diff file1 file1.last**" and written to the report.


beyza@beyza-virtual-machine: ~/ceng_kurt_beyza_final/Kurt_Beyza_Final/REPORT

```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final/Kurt_Beyza_Final/REPORT$ sudo diff /var/log/auth.log auth.log.last
31716,32412d31715
< Jun  6 17:26:06 beyza-virtual-machine sudo: pam_unix(sudo:session): session closed for user root
< Jun  6 17:26:06 beyza-virtual-machine sudo:    beyza : TTY=pts/0 ; PWD=/home/beyza/ceng_kurt_beyza_final/Kurt_Beyza_Final/R
EPORT ; USER=root ; COMMAND=/usr/bin/chmod +rwx auth.log.last
< Jun  6 17:26:06 beyza-virtual-machine sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
< Jun  6 17:26:06 beyza-virtual-machine sudo: pam_unix(sudo:session): session closed for user root
< Jun  6 17:26:06 beyza-virtual-machine sudo:    beyza : TTY=pts/0 ; PWD=/home/beyza/ceng_kurt_beyza_final/Kurt_Beyza_Final/R
EPORT ; USER=root ; COMMAND=/usr/bin/sha256sum /var/log/auth.log
< Jun  6 17:26:06 beyza-virtual-machine sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
< Jun  6 17:26:06 beyza-virtual-machine sudo: pam_unix(sudo:session): session closed for user root
< Jun  6 17:26:06 beyza-virtual-machine sudo:    beyza : TTY=pts/0 ; PWD=/home/beyza/ceng_kurt_beyza_final/Kurt_Beyza_Final/R
EPORT ; USER=root ; COMMAND=/usr/bin/cp /var/log/kern.log kern.log.last
< Jun  6 17:26:06 beyza-virtual-machine sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
```

Figure 11 the screenshot of the "diff file1 file1.last" command

Step 7: Hashing Files Again

```
sudo rm control.sha
for y in "${arr[@]}"
do
        sudo sha256sum /var/log/$y >> control.sha
done
sudo sha256sum /etc/passwd >> control.sha
```

I delete the old control.sha file and hash all files again.

Step 8: Backup for files, disk spaces, processes, services and tasks

```
declare -a arr2=("files" "diskspaces" "processes" "services" "tasks")

for z in "${arr2[@]}"
do
	cd ~
	case $z in
	"files")
		sudo tree -a > $home/REPORT/files.now
		;;
	"diskspaces")
		sudo df -h > $home/REPORT/diskspaces.now
		;;
	"processes")
		sudo ps -aux > $home/REPORT/processes.now
		;;
	"services")
		sudo systemctl list-unit-files --type service --all > $home/REPORT/services.now
		;;
	"tasks")
		for user in `cat /etc/passwd | cut -d":" -f1`;
		do
			sudo crontab -l -u $user &>> $home/REPORT/tasks.now
		done
		;;
	esac

	cd $home/REPORT
	diff $z.now $z.last > diff.temp

	if [ -s diff.temp ]; then
		echo "****************" ${z^^} "*************" >> SystemAnalysisReport

		echo "Things that have changed since the last check:" >> SystemAnalysisReport
		cat diff.temp >> SystemAnalysisReport
	fi

	mv $z.now $z.last

	rm diff.temp
done
```

I define an array named "arr2". I keep the "files, diskspaces, processes, services and tasks" values in it for titles the relevant part of the report. In this section, I use a loop that returns with these values. Case states in the loop perform the same actions as in "A.2. backup and hashing log files" section. The only difference is that in this section I keep the results of the commands with the suffix ".now". Then, I compare the last records I keep with the new ones and write diffirences to the "file.temp "file. If there is a difference, I write the content of the "file.temp" file, the differences, to the report. I change the names of files that the "file.now" type, as "file.last" where I keep the new results. Thus, old "file.last" files are renewed. Finally, I am deleting the file named "file.temp".

Step 9: Final

```
#--FINISH-----------------------------------------------------------------------
if [ -f $home/SystemAnalysisReport.gpg ]; then
        if [ -f $home/SystemAnalysisReport.last.gpg ]; then
                rm $home/SystemAnalysisReport.last.gpg
        fi
        mv $home/SystemAnalysisReport.gpg $home/SystemAnalysisReport.last.gpg
fi
mv SystemAnalysisReport $home

cd $home
gpg --batch --output SystemAnalysisReport.gpg --passphrase mypassword --symmetric
SystemAnalysisReport
rm SystemAnalysisReport

tar czf report.tar.gz REPORT/
rm -r REPORT/
#gpg --batch --output report.gpg --symmetric report.tar.gz
gpg --batch --output report.gpg --passphrase mypassword --symmetric report.tar.gz
#echo RELOADAGENT | gpg-connect-agent
rm report.tar.gz
```

This section is the last part of the script. I encrypt the report I created in this section. I preferred to keep the report I made earlier as a backup along with the report I created last.

First of all, I check the presence of the latest encrypted reports. I am deleting the last backup "SystemAnalysisReport.last.gpg" file. I am converting "SystemAnalysisReport.gpg" file to backup as "SystemAnalysisReport.last.gpg". I encrypt the report I prepared with a specific password using gpg as in the previous sections. I'm deleting the non-encrypted report. I compress and encrypt the "REPORT" folder where the files I hold for comparison exist. I delete the unencrypted version.

2.3.2 Linux Script's Report: SystemAnalysisReport

Below you can see which sections will be included in the report:

```
SYSTEM ANALYSIS REPORT


********************* BENCHMARKS *********************

sysbench' is installed.

***** CPU Benchmark *****
<results>

***** Memory Benchmark *****
<results>


********************* NETWORK *********************

'vnstat' is installed.

<results>


********************* SYSLOG *********************

Good news! There is no 'Out of memory' error.
OR
There is 'Out of memory' error(s)
<results>


Things that have changed since the last check:
<results>
-----------------------------------------------------------------------------------------------

********************* AUTH.LOG *********************

Good news! There is no 'Invalid user' error.
OR
There is 'Invalid user' error(s)
<results>
```

```
Good news! There is no 'shutdown' error.
OR
There is 'shutdown' error(s)
<results>


Things that have changed since the last check:
<results>
-----------------------------------------------------------------------------------------------

********************** KERN.LOG **********************

Things that have changed since the last check:
<results>
-----------------------------------------------------------------------------------------------

********************** MESSAGES **********************

Things that have changed since the last check:
<results>
-----------------------------------------------------------------------------------------------

********************** PASSWD **********************

Things that have changed since the last check:
<results>
-----------------------------------------------------------------------------------------------

********************** FILES **********************

Things that have changed since the last check:
<results>
-----------------------------------------------------------------------------------------------

********************** DISKSPACES **********************

Things that have changed since the last check:
<results>
-----------------------------------------------------------------------------------------------

********************** PROCESSES **********************

Things that have changed since the last check:
<results>
```

```
----------------------------------------------------------------------------------------------------

********************* SERVICES *********************

Things that have changed since the last check:
<results>
----------------------------------------------------------------------------------------------------

********************* TASKS *********************

Things that have changed since the last check:
<results>
----------------------------------------------------------------------------------------------------
```

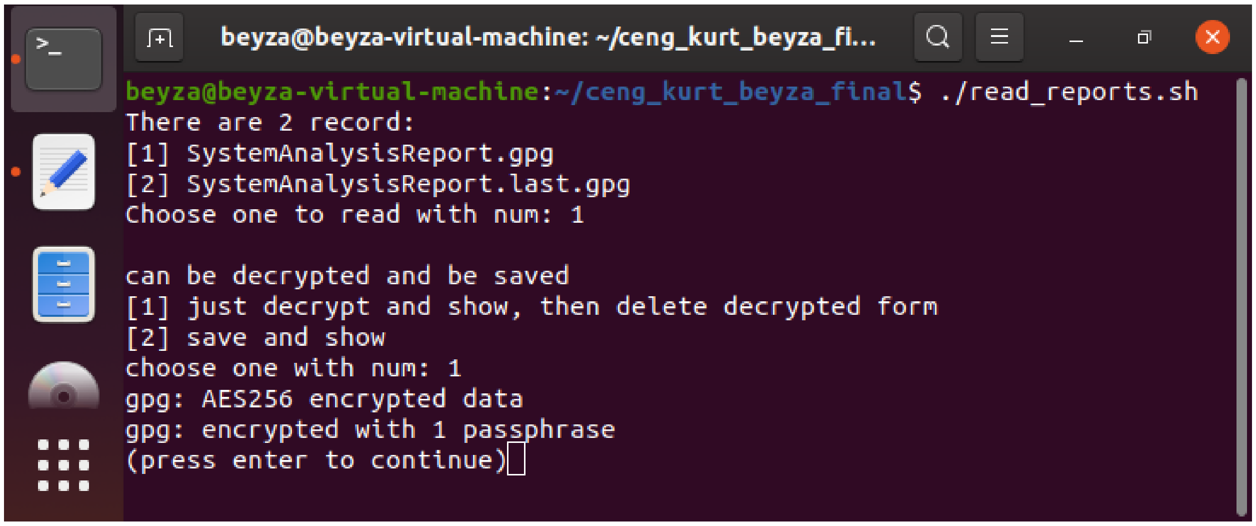
2.3.3 read_reports.sh

I needed to write an interactive script to read encrypted reports.

```
#!/bin/bash
startLoc=$(pwd)
cd Kurt_Beyza_Final

if [ ! -f SystemAnalysisReport.gpg ]; then
        echo "There is no file to read!"
        exit 0;
elif [ ! -f SystemAnalysisReport.last.gpg ]; then
        num="1"
else
        echo "There are 2 record:"
        echo "[1] SystemAnalysisReport.gpg"
        echo "[2] SystemAnalysisReport.last.gpg"
        read -p "Choose one to read with num: " num

        if [ $num != "1" ] && [ $num != "2" ]; then
                echo "This is not an option. Try again!"
                exit 0;
        fi
fi

echo "can be decrypted and be saved"
echo "[1] just decrypt and show, then delete decrypted form"
echo "[2] save and show"
read -p "choose one with num: " ans
```

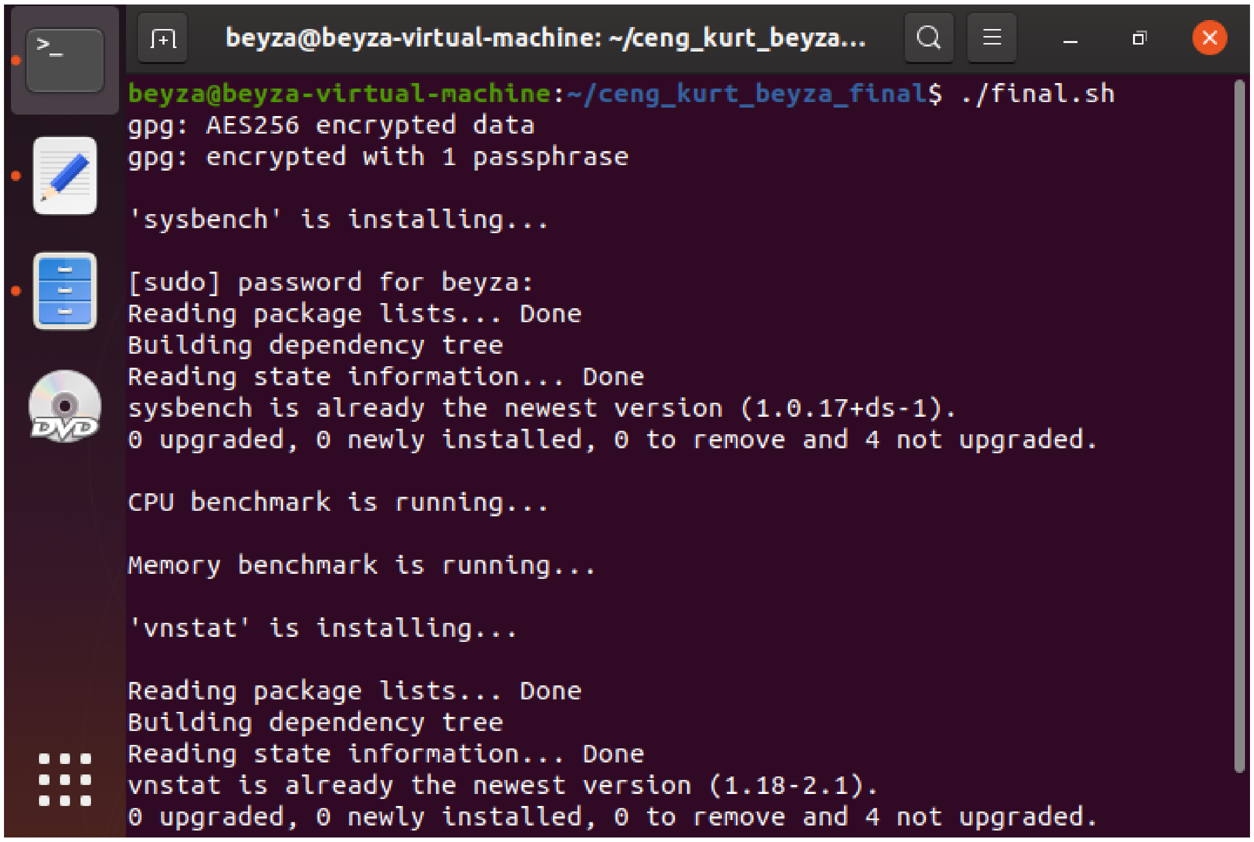
```
if [ $ans != "1" ] && [ $ans != "2" ]; then
        echo "This is not an option. Try again!"
        exit 0;
fi

case $num in
        "1")
                file_name="SystemAnalysisReport"
                ;;
        "2")
                file_name="SystemAnalysisReport.last"
                ;;
esac

gpg --batch --output $file_name --passphrase mypassword --decrypt $file_name.gpg
read -p "(press enter to continue)"

clear
if [ $ans == "1" ]; then
        cat $file_name
        rm $file_name
elif [ $ans == "2" ]; then
        cat $file_name
        echo "$file_name can be found at home directory."
fi
```

First of all, I check whether the reports exist or not. If they exist, I want the user to enter the report they want to read. Then there is a section on whether the report wants to keep the unencrypted version. I perform the actions according to the elections.



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ ./read_reports.sh
There are 2 record:
[1] SystemAnalysisReport.gpg
[2] SystemAnalysisReport.last.gpg
Choose one to read with num: 1

can be decrypted and be saved
[1] just decrypt and show, then delete decrypted form
[2] save and show
choose one with num: 1
gpg: AES256 encrypted data
gpg: encrypted with 1 passphrase
(press enter to continue)
```

Figure 12 the screenshot of the "read_reports.sh" script



```
beyza@beyza-virtual-machine:~/ceng_kurt_beyza_final$ ./final.sh
gpg: AES256 encrypted data
gpg: encrypted with 1 passphrase

'sysbench' is installing...

[sudo] password for beyza:
Reading package lists... Done
Building dependency tree
Reading state information... Done
sysbench is already the newest version (1.0.17+ds-1).
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.

CPU benchmark is running...

Memory benchmark is running...

'vnstat' is installing...

Reading package lists... Done
Building dependency tree
Reading state information... Done
vnstat is already the newest version (1.18-2.1).
0 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.
```

Figure 13 the screenshot of the "final.sh" bash script

3 Results

- There are open source antivirus programs. But they need to be updated frequently, and it's not easy for them to find the budget they need to do.
- Becoming more equipped is possible by using SIEM, which can do things that can be done by both of them, SEM and SIM.
- Hashing is very useful for detecting if the file is changed. If the file that is hashed before is changed, the file's hash will change. We can detect with checking old hash.
- Setting the bash script as a task run in background is too easy. The only thing has to be done is adding the necessary line to crontab file using "crontab -e".
- While encrypting the file, system saves the password that is entered by user. It should be cleared by using "echo RELOADAGENT | gpg-connect-agent" command.
- There are open source cloud services for operating systems like Linux.

4 Conclusion

As I mentioned "Results" section, I learned lots of things about open source antivirüs softwares and SIEM. I couldn't understand which log files can be looked in midterm assignment. In this assignment, I learned about this topic deeply. In addition to these, I found a chance to worked on my 'midterm' bash script. I learned how to make a script run regularly in the background. I learned about hashing, encrypting and cloud services. Most importantly, I gained experience on all these topics.

5 Grade

I trid to do every parts. I worked on cloud services. I wanted to work with an open source cloud service. I found "syncany", but I couldn't handle its usage because of the time or its usage is just not good or clear, I don't know. I am plannig to work on cloud services after my all final exams are done. I believe I will be graded equitable.

6 References

[1] https://static-course-assets.s3.amazonaws.com/CyberOps11/en/index.html#3.1.1.3

[2] https://www.forcepoint.com/cyber-edu/siem

[3] https://www.loggly.com/ultimate-guide/managing-linux-logs/

[4] https://www.loggly.com/ultimate-guide/troubleshooting-with-linux-logs/

[5] https://help.ubuntu.com/community/HowToSHA256SUM

[6] https://www.howtogeek.com/427982/how-to-encrypt-and-decrypt-files-with-gpg-on-linux/

[7] https://www.tecmint.com/linux-network-bandwidth-monitoring-tools/

[8] https://www.tecmint.com/free-open-source-cloud-storage-tools-for-linux/

[9] https://www.syncany.org/

[10] https://syncany.readthedocs.io/en/latest/commands.html

[11] https://community.rapidminer.com/discussion/32891/error-when-installing-rapidminer-stuidio-in-ubuntu-164

[12] https://www.baeldung.com/linux/encrypt-decrypt-files

[13] https://www.howtogeek.com/101288/how-to-schedule-tasks-on-linux-an-introduction-to-crontab-files/