Assignment 1
Team: Khantil Choksi khchoksi
Shubhankar Reddy skatta2

Environment:

Staging Server IP	User	Password	VM IP	User	Password
152.14.83.156	ece792	EcE792net!	192.168.124.15	ece792	EcE792net!

Problem 1

Basic Linux network verification tasks. Using the CLI Utility, show the following default configurations of your machine:

1. Interfaces
There are two popular commands to list down interfaces in linux machine:

Command: $ ifconfig -a
Output:

```
ubuntu@ip-172-31-30-30:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 02:db:0f:72:33:52
          inet addr:172.31.30.30  Bcast:172.31.31.255  Mask:255.255.240.0
          inet6 addr: fe80::db:fff:fe72:3352/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9001  Metric:1
          RX packets:163635 errors:0 dropped:0 overruns:0 frame:0
          TX packets:75478 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:135789933 (135.7 MB)  TX bytes:13638834 (13.6 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:195 errors:0 dropped:0 overruns:0 frame:0
          TX packets:195 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:14603 (14.6 KB)  TX bytes:14603 (14.6 KB)
```

Command: $ ip link show
Output:

```
ubuntu@ip-172-31-30-30:~$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 02:db:0f:72:33:52 brd ff:ff:ff:ff:ff:ff
```

2. Routing table

Command: netstat -rn
Output:

```
ubuntu@ip-172-31-30-30:~$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         172.31.16.1     0.0.0.0         UG        0 0          0 eth0
172.31.16.0     0.0.0.0         255.255.240.0   U         0 0          0 eth0
```

Command: ip route show
Output:

```
ubuntu@ip-172-31-30-30:~$ ip route show
default via 172.31.16.1 dev eth0
172.31.16.0/20 dev eth0  proto kernel  scope link  src 172.31.30.30
```

3. DNS
Command: $ nslookup domain_name
Output: to query the domain name server for “google.com”
e.g. nslookup google.com

```
ubuntu@ip-172-31-30-30:~$ nslookup google.com
Server:         172.31.0.2
Address:        172.31.0.2#53

Non-authoritative answer:
Name:   google.com
Address: 172.217.3.174
```

Command: $ dig domain_name
Output: e.g. dig google.com ; used for a hop by hop dns lookup

```
ubuntu@ip-172-31-30-30:~$ dig google.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64878
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             60      IN      A       172.217.14.238

;; Query time: 1 msec
;; SERVER: 172.31.0.2#53(172.31.0.2)
;; WHEN: Sat Sep 08 19:13:04 UTC 2018
;; MSG SIZE  rcvd: 55
```

4. DHCP (You might need to look at some configurations file)

Command: $ cat /etc/dhcp/dhclient.conf

Output:

```
ubuntu@ip-172-31-30-30:/etc/dhcp$ cat dhclient.conf
# Configuration file for /sbin/dhclient.
#
# This is a sample configuration file for dhclient. See dhclient.conf's
#       man page for more information about the syntax of this file
#       and a more comprehensive list of the parameters understood by
#       dhclient.
#
# Normally, if the DHCP server provides reasonable information and does
#       not leave anything out (like the domain name, for example), then
#       few changes must be made to this file, if any.
#

option rfc3442-classless-static-routes code 121 = array of unsigned integer 8;

send host-name = gethostname();
request subnet-mask, broadcast-address, time-offset, routers,
        domain-name, domain-name-servers, domain-search, host-name,
        dhcp6.name-servers, dhcp6.domain-search, dhcp6.fqdn, dhcp6.sntp-servers,
        netbios-name-servers, netbios-scope, interface-mtu,
        rfc3442-classless-static-routes, ntp-servers;

#send dhcp-client-identifier 1:0:a0:24:ab:fb:9c;
#send dhcp-lease-time 3600;
#supersede domain-name "fugue.com home.vix.com";
#prepend domain-name-servers 127.0.0.1;
#require subnet-mask, domain-name-servers;
timeout 300;
#retry 60;
#reboot 10;
#select-timeout 5;
#initial-interval 2;
#script "/sbin/dhclient-script";
#media "-link0 -link1 -link2", "link0 link1";
#reject 192.33.137.209;

#alias {
#  interface "eth0";
#  fixed-address 192.5.5.213;
#  option subnet-mask 255.255.255.255;
#}

#lease {
#  interface "eth0";
#  fixed-address 192.33.137.200;
#  medium "link0 link1";
#  option host-name "andare.swiftmedia.com";
#  option subnet-mask 255.255.255.0;
#  option broadcast-address 192.33.137.255;
#  option routers 192.33.137.250;
#  option domain-name-servers 127.0.0.1;
```

Problem 2:

Basic Linux performance verification tasks. Using the CLI Utility, show following performance stats of your machine.

1. CPU usage: Display three reports of statistics for all processors at two second intervals. Which CPU is least used (idle most of the time)?

Command: mpstat -P ALL 2 3

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ mpstat -P ALL 2 3
Linux 4.10.0-28-generic (ece792-Standard-PC-i440FX-PIIX-1996)   09/08/2018      _x86_64_        (4 CPU)

03:48:37 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
03:48:39 PM  all    0.38    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.62
03:48:39 PM    0    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
03:48:39 PM    1    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
03:48:39 PM    2    0.50    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.50
03:48:39 PM    3    1.01    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   98.99

03:48:39 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
03:48:41 PM  all    0.25    0.00    0.13    0.00    0.00    0.00    0.00    0.00    0.00   99.62
03:48:41 PM    0    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
03:48:41 PM    1    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
03:48:41 PM    2    0.00    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   99.50
03:48:41 PM    3    1.00    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   98.51

03:48:41 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
03:48:43 PM  all    0.37    0.00    0.25    0.00    0.00    0.00    0.00    0.00    0.00   99.38
03:48:43 PM    0    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
03:48:43 PM    1    0.00    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   99.50
03:48:43 PM    2    0.50    0.00    0.50    0.00    0.00    0.00    0.00    0.00    0.00   99.00
03:48:43 PM    3    1.01    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   98.99

Average:     CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
Average:     all    0.33    0.00    0.12    0.00    0.00    0.00    0.00    0.00    0.00   99.54
Average:       0    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
Average:       1    0.00    0.00    0.17    0.00    0.00    0.00    0.00    0.00    0.00   99.83
Average:       2    0.33    0.00    0.33    0.00    0.00    0.00    0.00    0.00    0.00   99.33
Average:       3    1.00    0.00    0.17    0.00    0.00    0.00    0.00    0.00    0.00   98.83
```

After looking up the average of three reports, we can see that CPU “0” is least used by having most idle time of 100%.

2. Memory usage: Display 3 reports of MEM statistics for every active task in the system at two second intervals. Which one is the most memory intensive task.

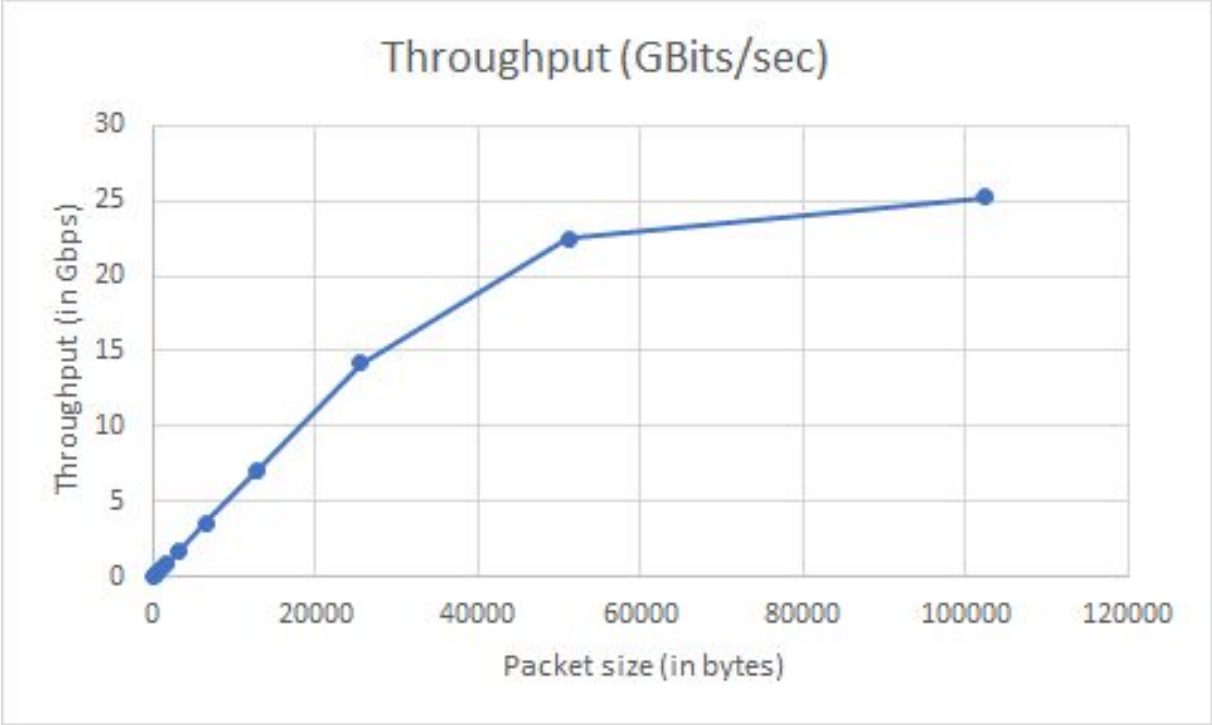
Command: pidstat 2 3 -r

```
ece792@ece792-Standard-PC-i440FX-PIIX-1996:~$ pidstat 2 3 -r
Linux 4.10.0-28-generic (ece792-Standard-PC-i440FX-PIIX-1996)   09/08/2018      _x86_64_        (4 CPU)

04:12:23 PM   UID       PID  minflt/s  majflt/s    VSZ     RSS   %MEM  Command
04:12:25 PM     0       892     19.90      0.00  363968   44624   0.18  Xorg
04:12:25 PM  1000     12783     33.83      0.00    7736    2220   0.01  pidstat

04:12:25 PM   UID       PID  minflt/s  majflt/s    VSZ     RSS   %MEM  Command
04:12:27 PM     0       248      2.50      0.00   43576   13372   0.05  systemd-journal
04:12:27 PM     0       791    153.00      0.00  469292   43592   0.18  NetworkManager
04:12:27 PM     0       892     22.00      0.00  363968   44624   0.18  Xorg
04:12:27 PM   109      1391      1.00      0.00  534140   21040   0.09  whoopsie
04:12:27 PM     0      6515      0.50      0.00  274932    9708   0.04  cups-browsed
04:12:27 PM  1000     12783      7.00      0.00    7736    2484   0.01  pidstat
04:12:27 PM     0     12785    196.00      0.00   16124    3576   0.01  dhclient

04:12:27 PM   UID       PID  minflt/s  majflt/s    VSZ     RSS   %MEM  Command
04:12:29 PM     0       892     19.90      0.00  363968   44624   0.18  Xorg
04:12:29 PM     0     12785      1.00      0.00   16124    3576   0.01  dhclient

Average:      UID       PID  minflt/s  majflt/s    VSZ     RSS   %MEM  Command
Average:        0       248      0.83      0.00   43576   13372   0.05  systemd-journal
Average:        0       791     50.83      0.00  469292   43591   0.18  NetworkManager
Average:        0       892     20.60      0.00  363968   44624   0.18  Xorg
Average:      109      1391      0.33      0.00  534140   21040   0.09  whoopsie
Average:        0      6515      0.17      0.00  274932    9708   0.04  cups-browsed
Average:     1000     12783     13.62      0.00    7736    2396   0.01  pidstat
Average:        0     12785     65.45      0.00   16124    3576   0.01  dhclient
```

As shown in the average, the most memory intensive tasks are “NetworkManager” and “Xorg” with 0.18% of memory usage.

Problem 3. (20 Points) Basic Linux tasks, use of tools. Install iperf traffic generator on your system. Run iperf command (iperf -c < ipofyourVM > -t 10 -l < packetsize(eg100B) >). Keep doubling packet sizes from 100 B to 6400B for different run. What is the average throughput achieved by the iperf data transfef for different packet sizer? Explain your observation. (Note: Before running client you need to start your server iperf -s < ipofyourV M >)

Packet Size(Bytes)	Throughput (GBits/sec)
102400	25.2
51200	22.5
25600	14.2
12800	7.09
6400	3.55
3200	1.77
1600	0.886
800	0.443
400	0.222
200	0.111
100	0.055



- We observe that the throughput decreases as the packet size decreases.
- As, there is more processing overhead for the ip stack traversal and operations which are done on each packet when we have a higher number of packets.
- This leads to a lower throughput.

Problem 4. Slow server Problem.

1. Monitoring Script

Write a shell script to do the following tasks:

(a) Log the CPU load averages in a CSV file with T second granularity. (format of csv: timestamp, 1 min load average, 5 min load average, 15 min load average)

```
$ ./cpuload.sh <T seconds granularity> <TP seconds>
```

```
#!/bin/bash

if [ "$#" -lt 2 ]; then
        echo "Two arguments are required. Granularity and Total Time"
        exit
fi

if [ ! -f ./cpu_load_avg.csv ]; then
        printf "timestamp, 1 min load average, 5 min load average, 15 min load
average\n" >> cpu_load_avg.csv
fi

counter=0
while [ $counter -lt $2 ];
do
   top -b -n  1  | awk '/load average/ { printf "%s, %s %s %s\n", $3, $12, $13, $14
}' >> cpu_load_avg.csv
   counter=$[$counter + $1]
   sleep $1
done
```

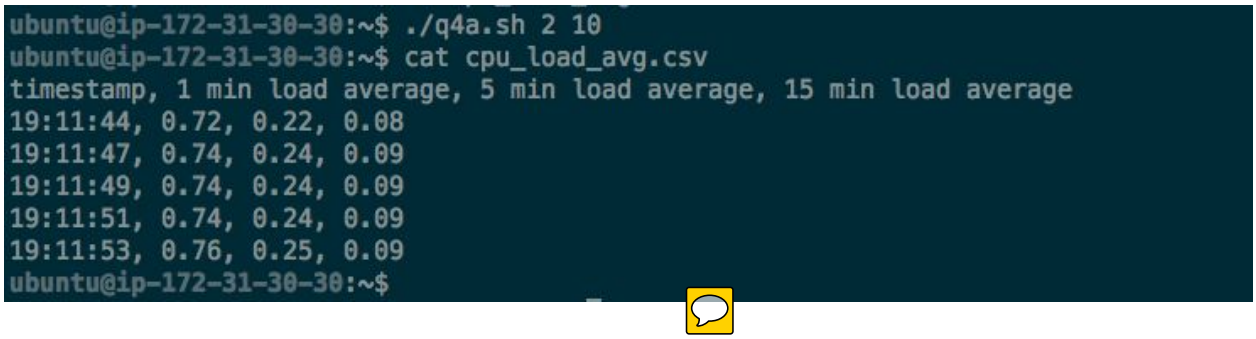
Command:

```
$ ./cpuload.sh 2 10
```

Output:

```
timestamp, 1 min load average, 5 min load average, 15 min load average
19:11:44, 0.72, 0.22, 0.08
19:11:47, 0.74, 0.24, 0.09
19:11:49, 0.74, 0.24, 0.09
19:11:51, 0.74, 0.24, 0.09
19:11:53, 0.76, 0.25, 0.09
```

Screenshot:



(b) Generate alert

i. ”HIGH CPU usage” if CPU usage in last one minute is more than a user defined threshold X.

ii. ”Very HIGH CPU usage” if CPU usage in last 5 minutes is more than a user defined threshold Y and load is increasing.

Log alert messages in a separate CSV file as timestamp, alert String, CPU load Average

Format:

```
$ ./alerts.sh <Granularity> <Total Duration> <X-High CPU Threshold> <Y-Very high CPU Threshold>
```

alerts.sh

```
#!/bin/bash

if [ "$#" -lt 2 ]; then
        echo "Two arguments are required. Granularity and Total Time"
        exit
fi

if [ ! -f ./alert.csv ]; then
        printf "timestamp, alert string, CPU load average \n" >> alert.csv
fi

counter=0

while [ $counter -lt $2 ];
do

        extract_load=`top -b -n  1  | awk '/load average/ { printf "%s, %.2f %.2f %.2f\n", $3, $12, $13, $14 }'`
        echo $extract_load
```

```
        IFS=', ' read -r -a load_avg <<< "$extract_load"
        echo "Timestamp: ${load_avg[0]}"
        echo "1 min load average : ${load_avg[1]}"
        is_high=`printf ${load_avg[1]}'\n'$3'\n' | sort -g | head -1`
        if [ $is_high == $3 ]; then
                echo "${load_avg[0]}, HIGH CPU usage, ${load_avg[1]}" >> alert.csv
        fi

        if [[ ${load_avg[2]} > $4 && ${load_avg[1]} > ${load_avg[2]}  &&  ${load_avg[2]} >
${load_avg[3]} ]]; then
                echo "${load_avg[0]}, Very HIGH CPU usage, ${load_avg[1]}" >> alert.csv
        fi

        counter=$[$counter+$1]
        sleep $1
done
```

Test this script by running a cron job. Submit your script (with readme) and a graph showing one minute load average taken every 10 seconds over 10 minutes duration.

We have used

Command:

A. Generate stress by running following command:

```
$ stress -c 1 -t 100
```

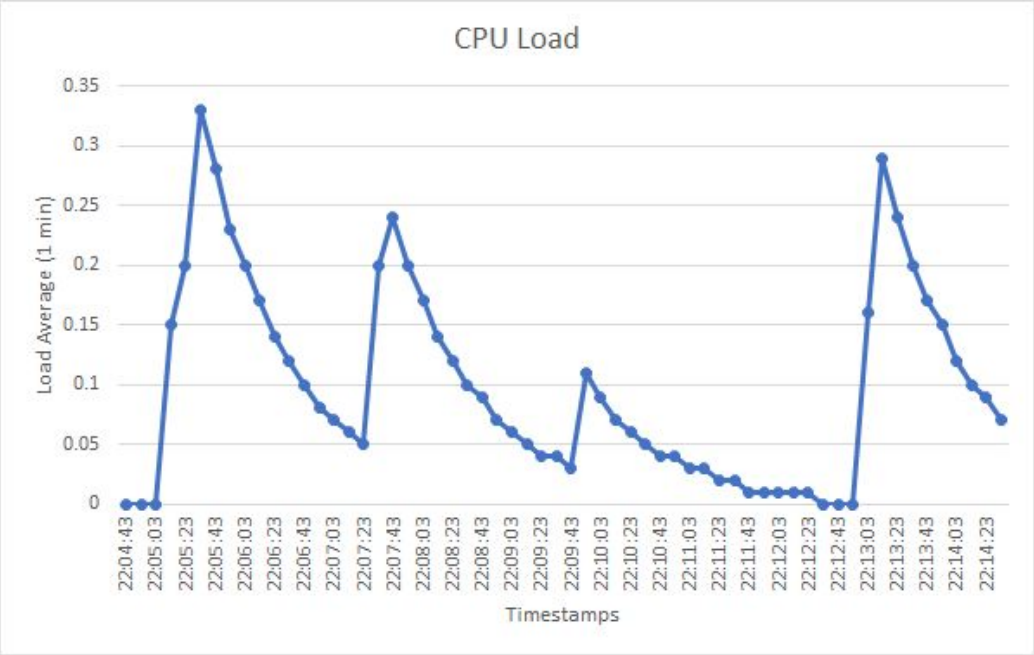
B. Run shell script or call shell script from cron job

```
$ ./alerts.sh 2 20 0.1 0.4
```

Output:

From where you have called alerts.sh file, on that same directory it will generate “alert.csv” file. Which is attached in the zip file.

```
[ec2-user@ip-172-31-44-232 9_12_home]$ cat alert.csv
timestamp, alert string, CPU load average
22:46:29, HIGH CPU usage, 0.15
22:46:31, HIGH CPU usage, 0.15
22:46:34, HIGH CPU usage, 0.15
22:46:36, HIGH CPU usage, 0.22
22:46:38, HIGH CPU usage, 0.22
22:46:40, HIGH CPU usage, 0.28
22:46:42, HIGH CPU usage, 0.28
22:46:44, HIGH CPU usage, 0.34
```



2. Log cleaning scripts A script to clear log files every hour (You can use cron job or log rotation)

Here we assume that, from the above code, alert.csv file is generated at absolute file path “cpu_load_avg.csv” and “alert.csv” .

So, we have written following script to clear log files:
clearlogs.sh

```
#!/bin/bash

rm *.csv
```

To run the above script every hour, we have setup the cron job, using following steps:
Step 1: Run

```
$ crontab -e
```

Step 2: Add the following line: (Make sure add absolute path of shell file)

```
0 * * * * ~/clearlogs.sh
```

Thanks.