TASK 1

SHELL SCRIPT

#!/bin/bash

echo $(seq 10) | tr " " "\n" | shuf

Build Instruction:

copy the random.sh file to linux machine and make sure you have changed the permission to executable by running chmod +x random.sh

Usage:

To execute the code run sh random.sh, script will randomly display the numbers from 1 to 10 each time.

Description:

In simple word it will order the set of things in different ways. so here we are echoing the sequence of numbers from 1 to 10 and we are piping the result to translate command it will list down the numbers instead of single line, once this is done using shuf command we are displaying the numbers randomly.

TASK 2

SERVER DETAILS:

OS: Ubuntu 20.04

WORDPRESS: 5.5

MARIA-DB : 10.3.31-MariaDB-0ubuntu0.20.04.1

APACHE: 2.4.41

CPU DETAILS:

```
root@ip-172-31-32-159:~# lscpu
Architecture:              x86_64
CPU op-mode(s):            32-bit, 64-bit
Byte Order:                Little Endian
Address sizes:             46 bits physical, 48 bits virtual
CPU(s):                    16
On-line CPU(s) list:       0-15
Thread(s) per core:        2
Core(s) per socket:        8
Socket(s):                 1
NUMA node(s):              1
Vendor ID:                 GenuineIntel
CPU family:                6
Model:                     85
Model name:                Intel(R) Xeon(R) Platinum 8259CL CPU @ 2.50GHz
Stepping:                  7
CPU MHz:                   2499.998
BogoMIPS:                  4999.99
Hypervisor vendor:         KVM
Virtualization type:       full
L1d cache:                 256 KiB
L1i cache:                 256 KiB
L2 cache:                  8 MiB
L3 cache:                  35.8 MiB
NUMA node0 CPU(s):         0-15
```
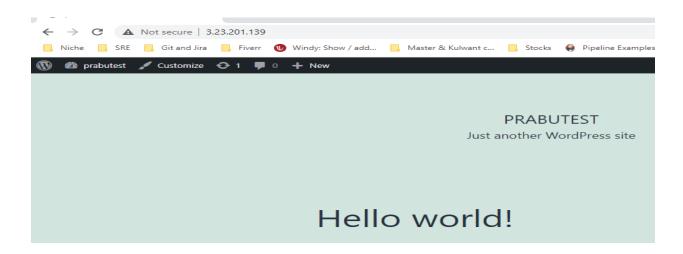
MEMORY DETAILS:

```
root@ip-172-31-32-159:~# free -h
              total        used        free      shared  buff/cache   available
Mem:           62Gi       440Mi        60Gi        23Mi       718Mi        60Gi
Swap:            0B          0B          0B
root@ip-172-31-32-159:~# 
```

DISK DETAILS:

```
root@ip-172-31-32-159:~# df -h
Filesystem       Size  Used Avail Use% Mounted on
/dev/root         97G  2.2G   95G   3% /
devtmpfs          32G     0   32G   0% /dev
tmpfs             32G     0   32G   0% /dev/shm
tmpfs            6.3G  900K  6.3G   1% /run
tmpfs            5.0M     0  5.0M   0% /run/lock
tmpfs             32G     0   32G   0% /sys/fs/cgroup
/dev/nvme1n1     1.9T   77M  1.8T   1% /data
/dev/loop1        34M   34M     0 100% /snap/amazon-ssm-agent/3552
/dev/loop0        56M   56M     0 100% /snap/core18/1997
/dev/loop2        33M   33M     0 100% /snap/snapd/11588
/dev/loop3        71M   71M     0 100% /snap/lxd/19647
tmpfs            6.3G     0  6.3G   0% /run/user/1000
root@ip-172-31-32-159:~#
```

NIC DETAILS:

```
root@ip-172-31-32-159:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
    link/ether 0a:0e:7a:ac:24:12 brd ff:ff:ff:ff:ff:ff
    inet 172.31.32.159/20 brd 172.31.47.255 scope global dynamic ens5
       valid_lft 2519sec preferred_lft 2519sec
    inet6 fe80::80e:7aff:feac:2412/64 scope link
       valid_lft forever preferred_lft forever
root@ip-172-31-32-159:~#
```

WEBSITE for load test

Server Load before starting the load test:

ALL CPU showing 0.0 due to no requests web server and Memory utilization is only 418M out of 64G everything looks fine before starting the load test

I am going to use the Apache benchmark tool for load testing, using this we can figure out the exact load when sending 20000 concurrent request to the server.

NOTE: we can setup only 20000 Concurrent requests in AB tool due to this limitation I am not going to setup 25000 request per second which is mentioned in the TASK Document.

We can do the monitoring by using so many open-source tools like nagios, Zabbix open source but more understanding I am going to do this manually by checking all the possible metrics using Linux tools. So, this will give us brief understanding of what happening in server and how we can troubleshoot the issue in future.

Screenshot of benchmark testing

SERVER METRICS:

UPTIME:

```
root@ip-172-31-32-159:~# uptime
 07:54:17 up  4:10,   2 users,   load average: 1.66, 1.25, 0.53
root@ip-172-31-32-159:~# uptime
 07:54:22 up  4:10,   2 users,   load average: 1.68, 1.27, 0.54
root@ip-172-31-32-159:~# uptime
 07:54:29 up  4:10,   2 users,   load average: 1.97, 1.34, 0.57
root@ip-172-31-32-159:~# uptime
 07:54:31 up  4:10,   2 users,   load average: 1.97, 1.34, 0.57
root@ip-172-31-32-159:~# uptime
 07:54:35 up  4:10,   2 users,   load average: 2.29, 1.42, 0.60
root@ip-172-31-32-159:~# uptime
 07:54:43 up  4:10,   2 users,   load average: 2.57, 1.50, 0.64
root@ip-172-31-32-159:~#
```

```
root@ip-172-31-32-159:~# uptime
 08:23:33 up  4:39,   3 users,   load average: 28.28, 8.91, 4.11
root@ip-172-31-32-159:~# htop
```

This is the quick way to view the load average and processes want to run this will give high level idea of resource load in server

This value counted based on 1, 5, 15 mins interval, so we can assume something is cooking up in CPU.

VMSTAT

```
root@ip-172-31-32-159:~# vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 3  0      0 62551780  45528 1326224    0    0     3    77    42  116  3  0 96  0  0
 5  0      0 62550808  45528 1326248    0    0     0     0 25490 30479 18  2 80  0  0
 2  0      0 62550224  45528 1326268    0    0     0     0 24665 28634 16  2 82  0  0
 1  0      0 62550304  45528 1326288    0    0     0     0 21652 24731 14  2 84  0  0
 6  0      0 62550936  45528 1326308    0    0     0     0 21367 22758 13  1 86  0  0
 3  0      0 62550304  45528 1326324    0    0     0     0 25618 29700 17  2 81  0  0
 1  0      0 62550312  45528 1326352    0    0     0     0 26597 31307 18  2 80  0  0
 1  0      0 62550520  45528 1326372    0    0     0     0 30663 37616 22  3 75  0  0
 3  0      0 62550560  45528 1326392    0    0     0     0 27777 32497 19  3 79  0  0
 3  0      0 62550912  45528 1326420    0    0     0     0 28428 33343 19  3 78  0  0
```

I have printed the vmstat values for every 1 second, there is nothing muching happening on the memory side, you can see free, buff, cache everything, SI(swap in) SO(Swap out) everything looks good, but CPU value, cs,us,sy,id,wa,st looks benchmark process is stressing the CPU

PIDSTAT 1:

This command is similar to top command, but it will print a rolling summary, instead of clearing screen, this will help us to identify which process is taking more cpu, in below example we can see apache2 taking more cpu resources

IOSTAT -XZ 1

This tool will help us to the disks both workload and performance, %utill value will tell us how much disk is busy, more than 60% typically lead to poor performace but here we have only 2% hence we can conclude that there is not much load in the disk.

Free -h

```
root@ip-172-31-32-159:~# free -h
              total        used        free      shared  buff/cache   available
Mem:           62Gi       1.1Gi        59Gi        17Mi       1.4Gi        60Gi
Swap:            0B          0B          0B
root@ip-172-31-32-159:~#
```

Free output looks fine there is no much use in memory side its just only 1.1g and we have to consider buff/cache its almost 1.4Gi higher the value it will affect disk block device IO

sar -n DEV 1

```
root@ip-172-31-32-159:~# sar -n DEV 1
Linux 5.11.0-1020-aws (ip-172-31-32-159)      10/27/21      _x86_64_      (16 CPU)

08:47:54      IFACE    rxpck/s    txpck/s    rxkB/s    txkB/s    rxcmp/s    txcmp/s   rxmcst/s   %ifutil
08:47:55        lo       0.00       0.00      0.00      0.00       0.00       0.00      0.00      0.00
08:47:55      ens5    9874.00    9718.00    866.32  10066.00       0.00       0.00      0.00      0.00

08:47:55      IFACE    rxpck/s    txpck/s    rxkB/s    txkB/s    rxcmp/s    txcmp/s   rxmcst/s   %ifutil
08:47:56        lo       0.00       0.00      0.00      0.00       0.00       0.00      0.00      0.00
08:47:56      ens5   10052.00   10541.00    824.50  10916.60       0.00       0.00      0.00      0.00

08:47:56      IFACE    rxpck/s    txpck/s    rxkB/s    txkB/s    rxcmp/s    txcmp/s   rxmcst/s   %ifutil
08:47:57        lo       0.00       0.00      0.00      0.00       0.00       0.00      0.00      0.00
08:47:57      ens5    9157.00    9505.00    758.24   9643.81       0.00       0.00      0.00      0.00

08:47:57      IFACE    rxpck/s    txpck/s    rxkB/s    txkB/s    rxcmp/s    txcmp/s   rxmcst/s   %ifutil
08:47:58        lo       0.00       0.00      0.00      0.00       0.00       0.00      0.00      0.00
08:47:58      ens5    9035.00    9767.00    717.41  10160.11       0.00       0.00      0.00      0.00

08:47:58      IFACE    rxpck/s    txpck/s    rxkB/s    txkB/s    rxcmp/s    txcmp/s   rxmcst/s   %ifutil
08:47:59        lo       0.00       0.00      0.00      0.00       0.00       0.00      0.00      0.00
08:47:59      ens5   10682.00   10771.00    913.12  10892.07       0.00       0.00      0.00      0.00
```

This will check network interface throughput, rxkb txkb will measure workload

sar -n TCP, ETCP 1

```
root@ip-172-31-32-159:~# sar -n TCP,ETCP 1
Linux 5.11.0-1020-aws (ip-172-31-32-159)        10/27/21        _x86_64_        (16 CPU)

08:52:20    active/s passive/s    iseg/s    oseg/s
08:52:21        0.00    234.00   3851.00   2763.00

08:52:20    atmptf/s  estres/s retrans/s isegerr/s    orsts/s
08:52:21        0.00      0.00   2089.00      0.00       0.00

08:52:21    active/s passive/s    iseg/s    oseg/s
08:52:22        0.00    186.00   2927.00   2286.00

08:52:21    atmptf/s  estres/s retrans/s isegerr/s    orsts/s
08:52:22        0.00      0.00   1973.00      0.00       0.00

08:52:22    active/s passive/s    iseg/s    oseg/s
08:52:23        0.00    223.00   3078.00   2656.00

08:52:22    atmptf/s  estres/s retrans/s isegerr/s    orsts/s
08:52:23        0.00      0.00   2014.00      0.00       0.00

08:52:23    active/s passive/s    iseg/s    oseg/s
08:52:24        0.00    202.00   4285.00   2457.00

08:52:23    atmptf/s  estres/s retrans/s isegerr/s    orsts/s
08:52:24        0.00      0.00   1807.00      0.00       0.00
^C


Average:    active/s passive/s    iseg/s    oseg/s
Average:        0.00    211.25   3535.25   2540.50
```

This will help us to find out the active and passive TCP connections in server which will give rough measure of server load.

HTOP:

Below is output of the htop command when the load was high on CPU side

FOR SSL PROXY We can use nagios for monitoring the below metrics

1.Inbound Traffic

2. Outbound Traffic

3. Open Connections

4. New Connections per second

5. Closed Connections per second

Challenges:

1. Assuming we have proper monitoring tools in place, we can use above commands to figure out the exact problems happening in the server
2. For SSL PROXY we need proper setup of monitoring se we can figure out the issue well in advance
3. Implementing proper SSL handshake failure count script, this will help us to know whether we have any failures between application and Proxy