دانشگاه صنعتي اميركبير

(پلی تکنیک تهران)

دانشکده‌مهندسی کامپیوتر و فناوری اطلاعات

مبانی رایانش ابري

پروژه پایانی:

Elastic Load Balancing via Haproxy

سارا اصغری (۹۵۱۳۰۰۵)، غزل صادقیان (۹۵۳۳۰۵۴)

گروه ۹

نیمسال اول تحصیلی ۹۹-۴۰۰

جدول تقسیم وظایف

وظایف	سارا اصغری	غزل صادقیان
بخش ۱ (سوال‌های تئوری)	✓	✓
گزارش بخش ۲	✓	✓
بخش 2-1	✓	✓
بخش 2-2	✓	✓
بخش 2-3	✓	✓
بخش 2-4	✓	✓
بخش 2-5	✓	✓
بخش 2-6	✓	✓
بخش 2-7	✓	✓
امتیازی	✓	✓

بخش اول

Load Balancing?

توزیع بار به معنای توزیع کارآمد ترافیک شبکه یا برنامه بین سرورهای متعدد در یک server farm است. هر توزیع کننده بار بین دستگاه های سرویس گیرنده و سرورهای backend قرار می گیرد و درخواست های ورودی را دریافت می کند و سپس به هر سرور موجود که قادر به انجام آنها باشد توزیع می کند.[1]

load balancing، در دسترس بودن ، به روز بودن و عملکرد سرورها ، وب سایت ها و برنامه های را در هنگام افزایش ترافیک تضمین می کند. این امر مخصوصاً برای شرکت هایی با چندین data center و وب سایتها یا برنامه های پر بازدید مهم است.

یک روش بهینه سازی عملکرد که با تقسیم کارهای ورودی در چندین منبع محاسباتی ، تحمل خطا را فراهم می کند ، در نتیجه از بار بیش از حد یک منبع جلوگیری می کند.

برای توضیح بیشتر ، load balancer توزیع کننده بار کاری در چندین منبع محاسباتی است. این منابع می توانند computer CPU، network links، clusters یا disk drives باشند.

توازن بار با هدف دستیابی به استفاده بهینه از منابع ، به حداکثر رساندن stability سرور و جلوگیری از overload شدن اجزای منفرد است. در واقع با تقسیم مقدار کارهایی را که یک سرور باید انجام دهد بین دو یا چند سرور، اجازه می دهد کار بیشتری در همان زمان انجام شود.[4]

load balancer ها به طور کلی به دو دسته طبقه‌بندی می شوند، لایه 4 و لایه 7:

load balancer لایه 4 بر اساس داده های موجود در پروتکل های لایه Transport شبکه عمل می کنند.

load balancer لایه 7 درخواست ها را بر اساس داده های موجود در پروتکل های لایه application مانند HTTP توزیع می کنند.

درخواست ها توسط هر دو نوع تعادل دهنده بار دریافت می شوند و بر اساس الگوریتم پیکربندی شده به یک سرور خاص توزیع می شوند

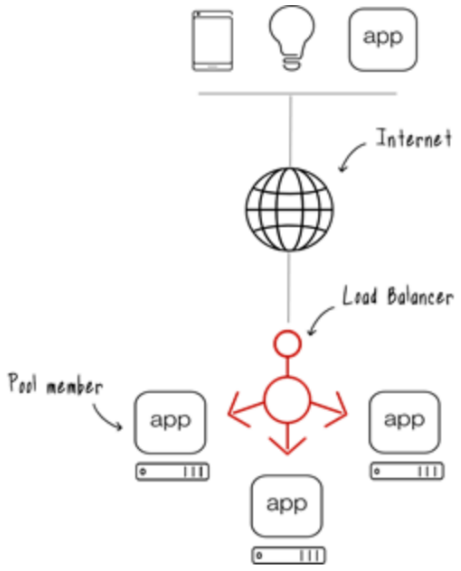
برخی از الگوریتم های استاندارد عبارتند از:[2]

- Round robin
- Weighted round robin
- Least connections
- Least response time

load balancer ها با اطمینان از سلامت سرورها و ارسال درخواست ها تنها به سرورها و برنامه هایی که می توانند به موقع پاسخ دهند، قابلیت اطمینان و در دسترس بودن را تضمین می کنند.[3]

مزایای Load Balancing:

- Reduced downtime
- Scalable
- Redundancy
- Flexibility
- Efficiency



Haproxy چیست؟

Haproxy که کوتاه شده ی عبارت High Availability Proxy است، یک نرم افزار متن باز توزیع بار TCP/HTTP است و یک راهکار پروکسی است که بر روی سیستم های Solaris ،Linux و FreeBSD قابل اجراست. اصلی ترین کاربرد آن در بهبود کارایی و قابلیت اطمینان در حوزه ی کار با سرور از طریق توزیع حجم کار بر روی چندین سرور است. از این نرم افزار در Github، Instagram ،Imger و Twitter استفاده شده است.

الگوریتم های توزیع بار پشتیبانی شده در Haproxy:

بطور کلی الگوریتم های توزیع بار به الگوریتم هایی گفته می شود که در هنگام توزیع بار، مشخص می کنند کدام سرور در backend انتخاب خواهد شد. علاوه بر الگوریتم ها، می توان به هر سرور یک پارامتر وزن نیز اختصاص داد و از این طریق میزان فراوانی انتخاب هر سرور در مقایسه با بقیه ی سرورها تعیین خواهد شد. در Haproxy الگوریتم های زیادی بدین منظور پیاده سازی شده اند. در اینجا چند مورد از پرکاربردترین الگوریتم‌ها بررسی شده اند:

1- Round-Robin:

این الگوریتم معمول ترین الگوریتم موجود و الگوریتم پیش فرض Haproxy است.

می توان این الگوریتم را منصفانه ترین الگوریتم در میان دیگر الگوریتم ها شمرد، زیرا زمان پردازش سرورها به طور مساوی توزیع می شود. در این الگوریتم سرورها یک به یک و به نوبت بر اساس وزنشان انتخاب می شوند. به عنوان یک الگوریتم پویا، Round-Robin اجازه می دهد تا وزن سرورها در حین حرکت تنظیم شود.

2- Static Round Robin:

مشابه Round Robin ، هر سرور به ترتیب طبق وزن خود انتخاب می شود. هر چند برخلاف Round Robin ، تغییر وزن سرور به شکل مداوم و پویا امکان‌پذیر نیست.

:Leastconn -3

این الگوریتم هربار سرور با کمترین تعداد connection ها را انتخاب می کند. این روش برای session های طولانی‌تر مانند LDAP، SQL و TSE پیشنهاد می شود اما برای پروتکل هایی که از session های کوتاه مانند HTTP استفاده می کنند مناسب نیست.

سرورهای موجود در backend یکسان نیز بطور چرخشی (الگوریتم round-robin) انتخاب می شوند. این الگوریتم نیز مانند الگوریتم Round-Robin پویا است.

:Source -4

این الگوریتم IP منبع را hash می‌کند و آن را بر وزن کل سرورهای در حال اجرا تقسیم می‌کند.

.در این رویکرد اطمینان حاصل می شود که هر کاربر هر بار به سرور یکسانی متصل شود؛ به شرط آنکه هیچ سروری up یا down نشود. این الگوریتم به طور کلی در حالت TCP استفاده می شود که در آن کوکی ها نمی توانند وارد شوند.

:URI -5

این الگوریتم یا قسمت چپ URI یا کل URI را hash می کند و مقدار hash را بر وزن کل سرورهای در حال اجرا تقسیم می کند. تا زمانی که هیچ سروری خاموش یا روشن نشود ، یک URI همیشه به یک سرور ثابت هدایت می شود. این الگوریتم نیز یک الگوریتم static است و به همان روش الگوریتم Source کار می کند.

:URL-Parameter -6

این الگوریتم فقط می تواند در backend HTTP استفاده شود. پارامتر URL در رشته ی query هر درخواست HTTP GET جستجو می شود. اگر پس از این پارامتر علامت '=' و پس از آن یک مقدار آمده باشد، آن مقدار هش شده و بر وزن کل سرورهای در حال اجرا تقسیم می شود.

منبع:

[1]https://www.citrix.com/solutions/app-delivery-and-security/load-balancing/what-is-load-balancing.html

[2] https://www.f5.com/services/resources/glossary/load-balancer

[3] https://www.nginx.com/resources/glossary/load-balancing/

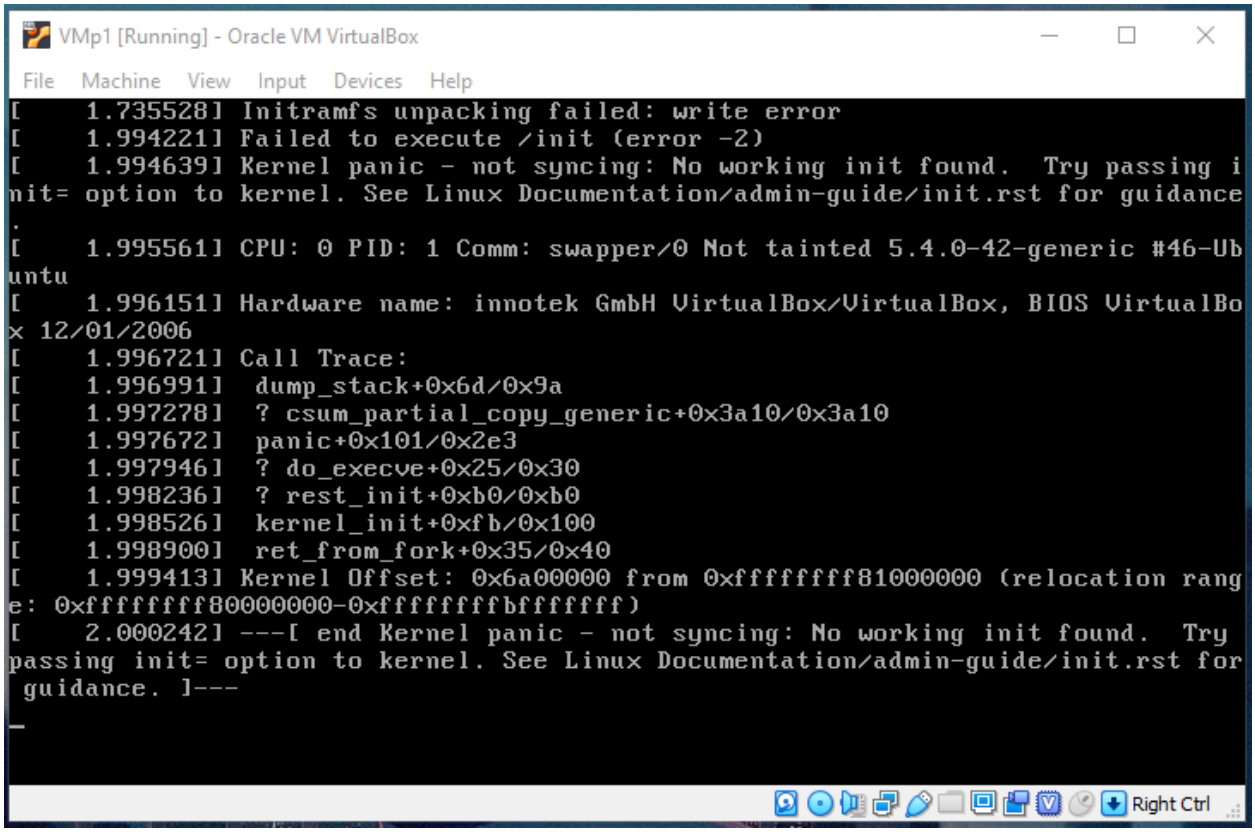
[4] https://blog.stackpath.com/load-balancing-haproxy/

[5]https://www.digitalocean.com/community/tutorials/an-introduction-to-haproxy-and-load-balancing-concepts

بخش دوم

بخش 1-2:

مشکل کمبود ram : (در صورت پروژه ذکر شده است که ram برای vm1 بر روی 512MB تنظیم شود اما ارور زیر به وجود می آید و برای حل آن میزان ram را به 1024MB تغییر دادیم)



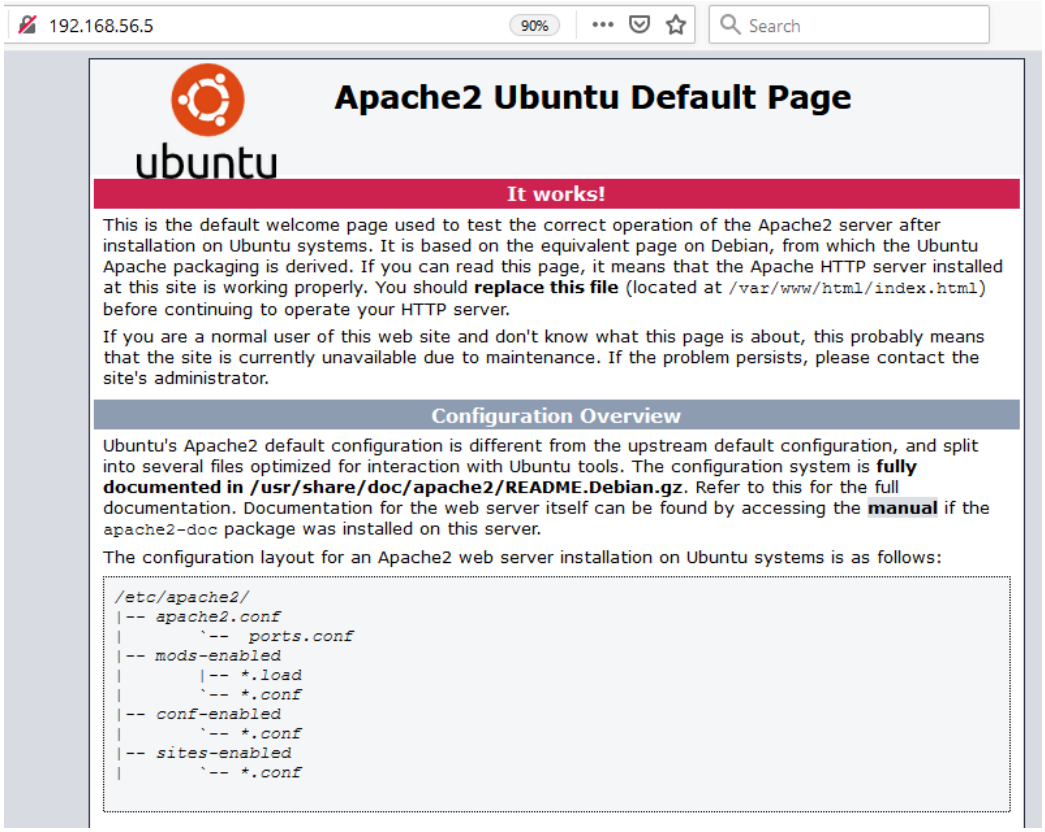
- راه اندازی **apache2** بر روی **vm1**:

```
sara@vmp1: ~
sara@vmp1:~$ sudo apt update
[sudo] password for sara:
Hit:1 http://ir.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ir.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://ir.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://ir.archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
94 packages can be upgraded. Run 'apt list --upgradable' to see them.
sara@vmp1:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap libjansson4 liblua5.2-0 ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser
  openssl-blacklist
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap libjansson4 liblua5.2-0 ssl-cert
0 upgraded, 11 newly installed, 0 to remove and 94 not upgraded.
```

```
sara@vmp1:~$ apache2 -version
Server version: Apache/2.4.41 (Ubuntu)
Server built:   2020-08-12T19:46:17
sara@vmp1:~$
```

```
sara@vmp1: ~
sara@vmp1:~$ sudo ufw app list
Available applications:
  Apache
  Apache Full
  Apache Secure
  OpenSSH
sara@vmp1:~$ sudo ufw allow 'Apache'
Rules updated
Rules updated (v6)
sara@vmp1:~$ sudo ufw status
Status: inactive
sara@vmp1:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese>
     Active: active (running) since Sat 2021-01-30 11:16:27 UTC; 7min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 1866 (apache2)
      Tasks: 55 (limit: 1074)
     Memory: 5.1M
     CGroup: /system.slice/apache2.service
             ├─1866 /usr/sbin/apache2 -k start
             ├─1868 /usr/sbin/apache2 -k start
             └─1869 /usr/sbin/apache2 -k start
```

- دسترسی به صفحه ی وب سرور از مرورگر سیستم میزبان:



- راه اندازی و پیکربندی collectd بر روی vm1:

(شامل plugin های network، memory، cpu، csv)

```
sara@vmp1: ~
sara@vmp1:~$ sudo apt install collectd python build-essential librrds-perl libjs
on-perl libhtml-parser-perl apache2 emboss bioperl ncbi-blast+ gzip libjson-perl
 libtext-csv-perl libfile-slurp-perl liblwp-protocol-https-perl libwww-perl git
libconfig-general-perl libregexp-common-perl
[sudo] password for sara:
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'python-is-python2' instead of 'python'
git is already the newest version (1:2.25.1-1ubuntu3).
git set to manually installed.
gzip is already the newest version (1.10-0ubuntu4).
apache2 is already the newest version (2.4.41-4ubuntu3.1).
The following additional packages will be installed:
  acl adwaita-icon-theme amap-align apg aptdaemon aptdaemon-data aspell
  aspell-en at-spi2-core autoconf automake autotools-dev avahi-daemon
  avahi-utils bedtools binutils binutils-common binutils-x86-64-linux-gnu
  bioperl-run blast2 blends-common bluez bowtie bubblewrap bwa
  ca-certificates-java cheese-common clustalo clustalw collectd-core colord
  colord-data cpp cpp-8 cpp-9 cracklib-runtime cups-pk-helper cwltool dbus-x11
  dconf-cli default-jre-headless desktop-file-utils dialign dialign-tx
  dialign-tx-data dictionaries-common dns-root-data dnsmasq-base docbook-xml
  dpkg-dev emacsen-common emboss-data emboss-lib enchant-2
  evolution-data-server evolution-data-server-common exonerate fakeroot
```

```
  GNU nano 4.8                    /etc/apache2/apache2.conf
# error, crit, alert, emerg.
# It is also possible to configure the log level for particular modules, e.g.
# "LogLevel info ssl:warn"
#
LogLevel warn

# Include module configuration:
IncludeOptional mods-enabled/*.load
IncludeOptional mods-enabled/*.conf

# Include list of ports to listen on
Include ports.conf


# Sets the default security model of the Apache2 HTTPD server. It does
# not allow access to the root filesystem outside of /usr/share and /var/www.
# The former is used by web applications packaged in Debian,
# the latter may be used for local directories served by the web server. If
# your system is serving content from a sub-directory in /srv you must allow
# access here, or in any related virtual host.

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

```
  GNU nano 4.8                    /etc/apache2/ports.conf                Modified
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80
Listen 0.0.0.0:80

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet




^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

```
sara@vmp1:~$ systemctl restart apache2
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to restart 'apache2.service'.
Authenticating as: sara
Password:
==== AUTHENTICATION COMPLETE ===
sara@vmp1:~$
```

```
sara@vmp1:~$ git clone https://github.com/httpdss/collectd-web.git
Cloning into 'collectd-web'...
remote: Enumerating objects: 1430, done.
remote: Total 1430 (delta 0), reused 0 (delta 0), pack-reused 1430
Receiving objects: 100% (1430/1430), 1.19 MiB | 553.00 KiB/s, done.
Resolving deltas: 100% (664/664), done.
sara@vmp1:~$
```

```
  GNU nano 4.8                /etc/collectd/collectd.conf
#       IgnoreSelected false
#</Plugin>

<Plugin cpu>
        ReportByCpu true
        ReportByState true
        ValuesPercentage true
#       ReportNumCpu false
#       ReportGuestState false
#       SubtractGuestState true
</Plugin>

<Plugin csv>
        DataDir "/var/lib/collectd/csv"
        StoreRates true
</Plugin>

#<Plugin curl>
#       <Page "stock_quotes">
#               URL "http://finance.google.com/finance?q=NYSE%3AAMD"

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

```
  GNU nano 4.8                    /etc/collectd/collectd.conf
# or:
#                #Host "memcache.example.com"
#                Address "127.0.0.1"
#                Port "11211"
#        </Instance>
#</Plugin>

<Plugin memory>
        ValuesAbsolute false
        ValuesPercentage true
</Plugin>

#<Plugin modbus>
#        <Data "data_name">
#                RegisterBase 1234
#                RegisterCmd ReadHolding
#                RegisterType float
#                Type gauge
#                Instance "..."
#        </Data>

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

- نمایش خروجی collectd در قالب فایل های csv:

```
sara@vmp1:~$ cd /var/lib/collectd/csv
sara@vmp1:/var/lib/collectd/csv$ cd vmp1
sara@vmp1:/var/lib/collectd/csv/vmp1$ ls
battery-0            df-snap-snapd-10707  disk-loop6        interface-lo
cpu-0                df-snap-snapd-8542   disk-loop7        irq
df-boot              disk-dm-0            disk-sda          load
df-root              disk-loop0           disk-sda1         memory
df-snap-core18-1880  disk-loop1           disk-sda2         processes
df-snap-core18-1944  disk-loop2           disk-sda3         swap
df-snap-hey-1        disk-loop3           entropy           users
df-snap-lxd-16099    disk-loop4           interface-enp0s3
df-snap-lxd-19188    disk-loop5           interface-enp0s8
sara@vmp1:/var/lib/collectd/csv/vmp1$ cd cpu-0
sara@vmp1:/var/lib/collectd/csv/vmp1/cpu-0$ ls
cpu-idle-2021-02-04          percent-nice-2021-02-06
cpu-interrupt-2021-02-04     percent-softirq-2021-02-04
cpu-nice-2021-02-04          percent-softirq-2021-02-05
cpu-softirq-2021-02-04       percent-softirq-2021-02-06
cpu-steal-2021-02-04         percent-steal-2021-02-04
cpu-system-2021-02-04        percent-steal-2021-02-05
cpu-user-2021-02-04          percent-steal-2021-02-06
cpu-wait-2021-02-04          percent-system-2021-02-04
percent-idle-2021-02-04      percent-system-2021-02-05
percent-idle-2021-02-05      percent-system-2021-02-06
percent-idle-2021-02-06      percent-user-2021-02-04
percent-interrupt-2021-02-04 percent-user-2021-02-05
percent-interrupt-2021-02-05 percent-user-2021-02-06
percent-interrupt-2021-02-06 percent-wait-2021-02-04
percent-nice-2021-02-04      percent-wait-2021-02-05
percent-nice-2021-02-05      percent-wait-2021-02-06
sara@vmp1:/var/lib/collectd/csv/vmp1/cpu-0$ cd ..
sara@vmp1:/var/lib/collectd/csv/vmp1$ cd memory
sara@vmp1:/var/lib/collectd/csv/vmp1/memory$ ls
memory-buffered-2021-02-04     memory-used-2021-02-04       percent-cached-2021-02-05  percent-slab_recl-2021-02-04    percent-slab_unrecl-2021-02-06
memory-cached-2021-02-04       percent-buffered-2021-02-04  percent-cached-2021-02-06  percent-slab_recl-2021-02-05    percent-used-2021-02-04
memory-free-2021-02-04         percent-buffered-2021-02-05  percent-free-2021-02-04    percent-slab_recl-2021-02-06    percent-used-2021-02-05
memory-slab_recl-2021-02-04    percent-buffered-2021-02-06  percent-free-2021-02-05    percent-slab_unrecl-2021-02-04  percent-used-2021-02-06
memory-slab_unrecl-2021-02-04  percent-cached-2021-02-04    percent-free-2021-02-06    percent-slab_unrecl-2021-02-05
sara@vmp1:/var/lib/collectd/csv/vmp1/memory$ |
```

بخش 2-2:

- کدهای cpu-intensive.php و memory-intensive.php و cpu-mem-intensive.php:

این کدها به همراه فایل گزارش در فولدر مربوطه قرار داده شده اند.

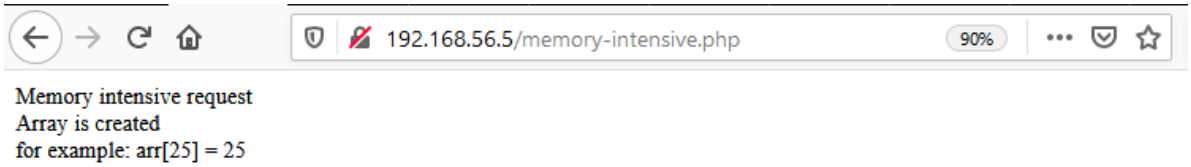
```
  GNU nano 4.8                    cpu-intensive.php
<?php
echo "CPU intensive request <br>";
$a = 1.0001;
$n = 1000000;
$f = 1;
for($i = 1; $i < ($n + 1); $i++){
  $f = $f * $a;
}
echo "$a to the power of $n = $f <br>";
?>
```

```
  GNU nano 4.8                    memory-intensive.php
<?php
echo "Memory intensive request <br>";
$arr = [];
$n = 1000000;
for ($i = 0; $i < $n; $i++)
{
  $arr[$i] = $i;
}
echo "Array is created <br>";
echo "for example: arr[25] = $arr[25]";
?>
```

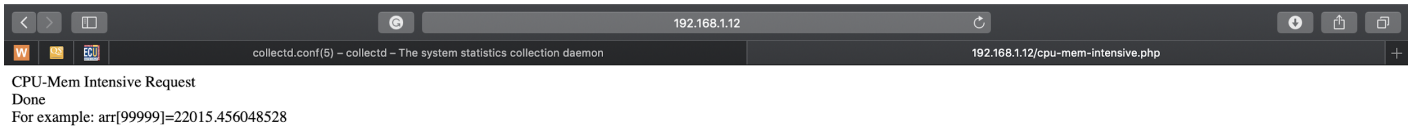
- دسترسی به فایل های مذکور از طریق مرورگر:

: Cpu-intensive.php

: Memory-intensive.php



: Cpu-mem-intensive.php

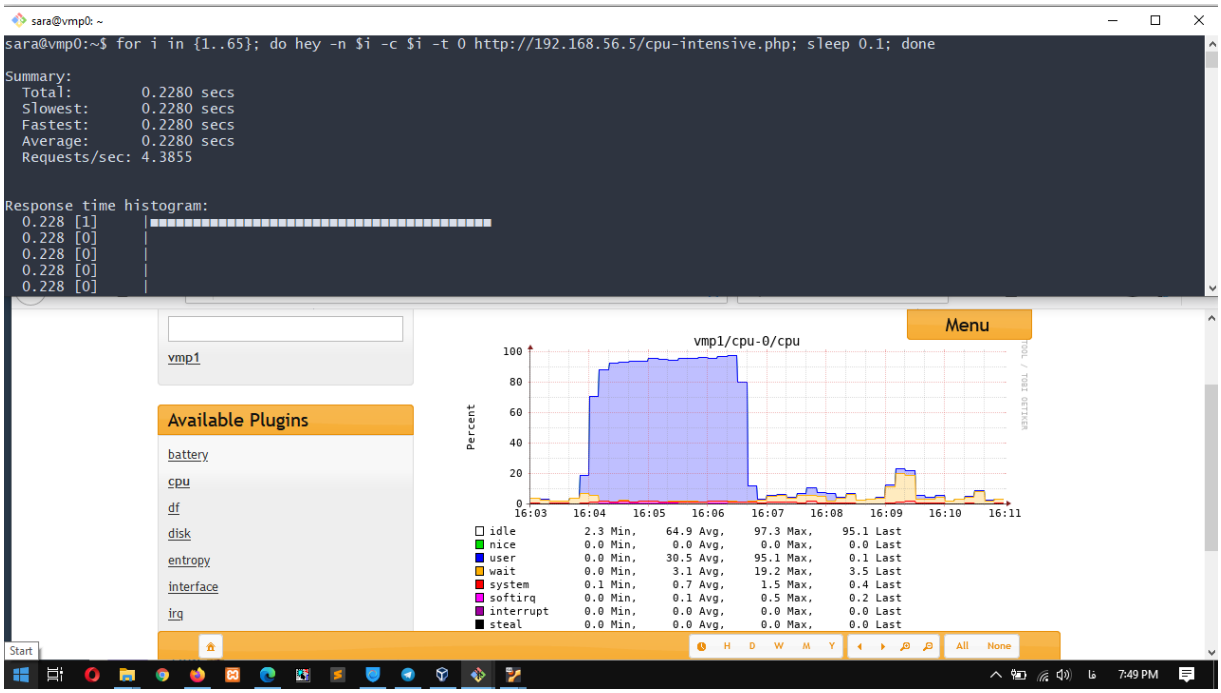


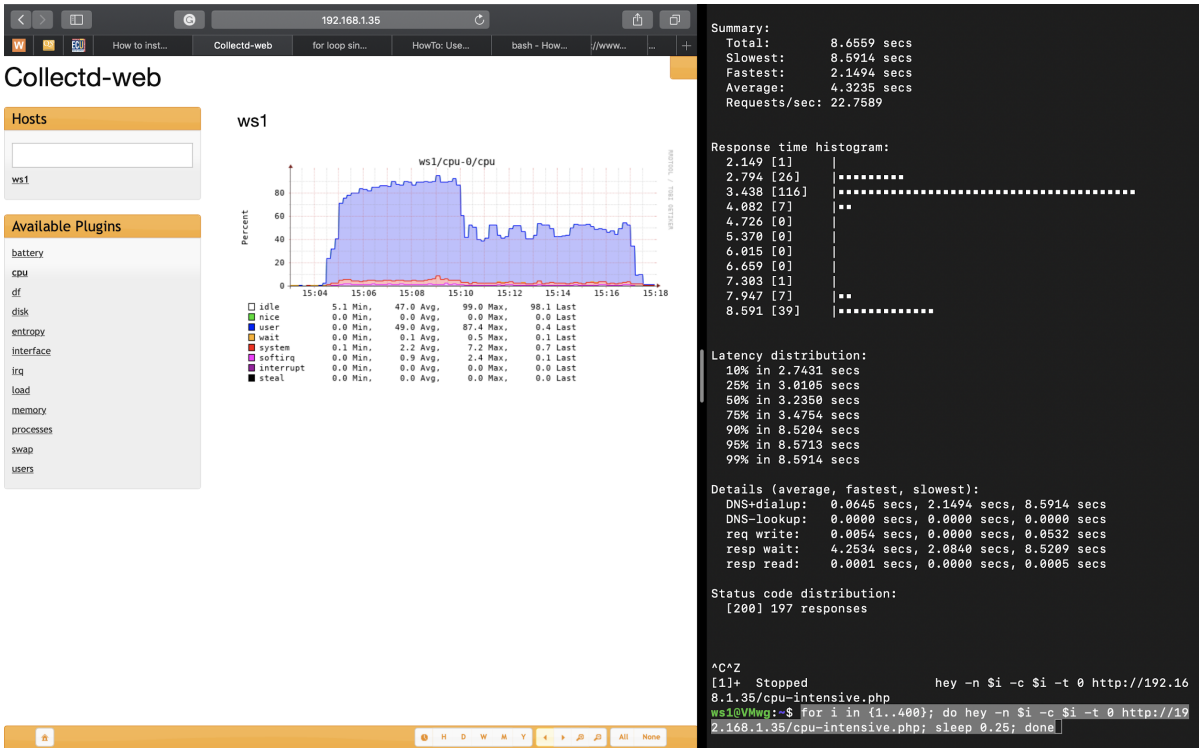
بخش 3-2:

- اجرای workload generator بر روی vm0 و ارسال درخواست به فایل های مذکور: (نمایش نمودارهای cpu و mem در واسط تحت وب collectd)

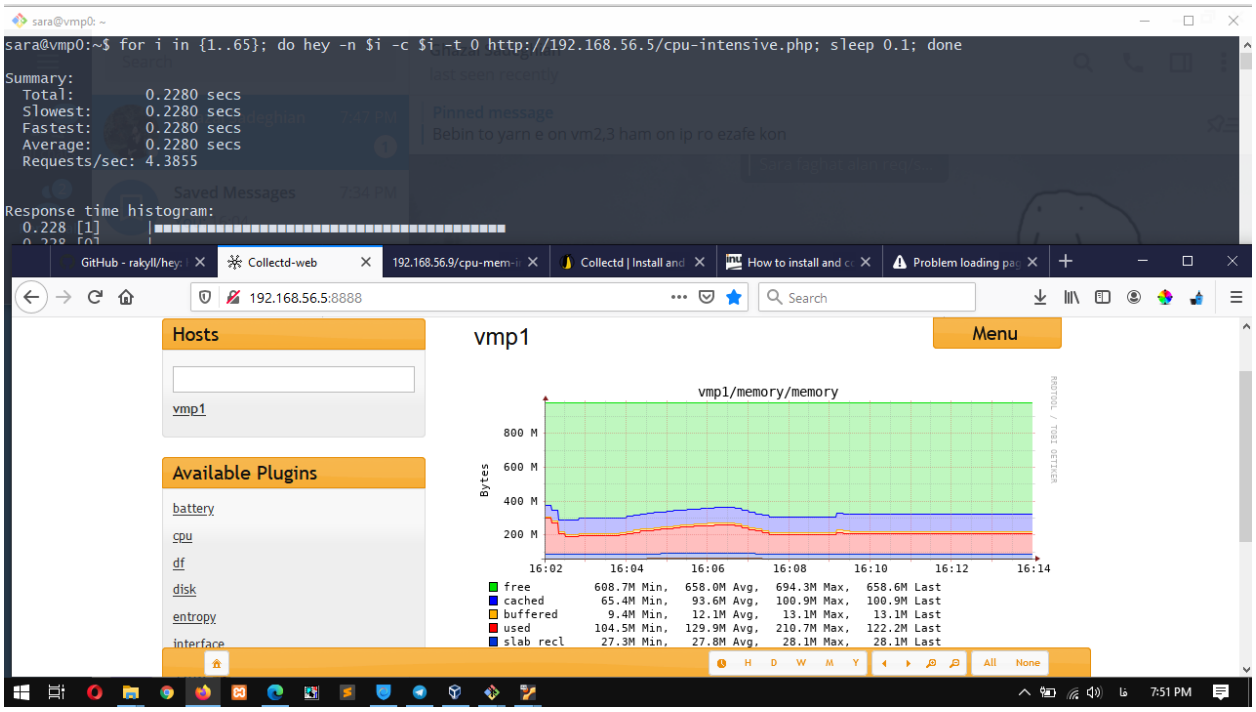
Cpu-intensive.php

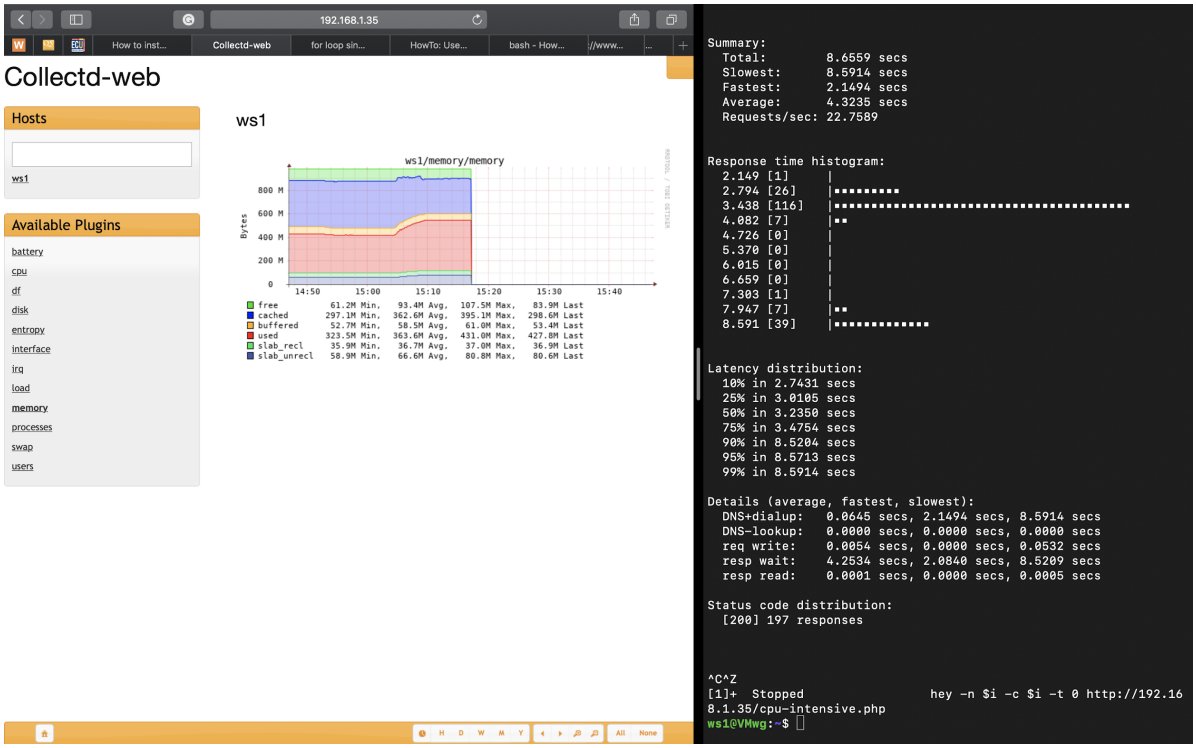
Cpu




192.168.1.35
Collectd-web
Hosts
ws1
Available Plugins
battery
cpu
df
disk
entropy
interface
irq
load
memory
processes
swap
users
ws1
ws1/cpu-0/cpu
Percent
Summary:
Total: 8.6559 secs
Slowest: 8.5914 secs
Fastest: 2.1494 secs
Average: 4.3235 secs
Requests/sec: 22.7589
Response time histogram:
Latency distribution:
10% in 2.7431 secs
25% in 3.0105 secs
50% in 3.2350 secs
75% in 3.4754 secs
90% in 8.5204 secs
95% in 8.5713 secs
99% in 8.5914 secs
Details (average, fastest, slowest):
DNS+dialup: 0.0645 secs, 2.1494 secs, 8.5914 secs
DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0000 secs
req write: 0.0054 secs, 0.0000 secs, 0.0532 secs
resp wait: 4.2534 secs, 2.0840 secs, 8.5209 secs
resp read: 0.0001 secs, 0.0000 secs, 0.0005 secs
Status code distribution:
[200] 197 responses
^C^Z
[1]+ Stopped hey -n $i -c $i -t 0 http://192.168.1.35/cpu-intensive.php
ws1@VMwg:~$ for i in {1..400}; do hey -n $i -c $i -t 0 http://192.168.1.35/cpu-intensive.php; sleep 0.25; done

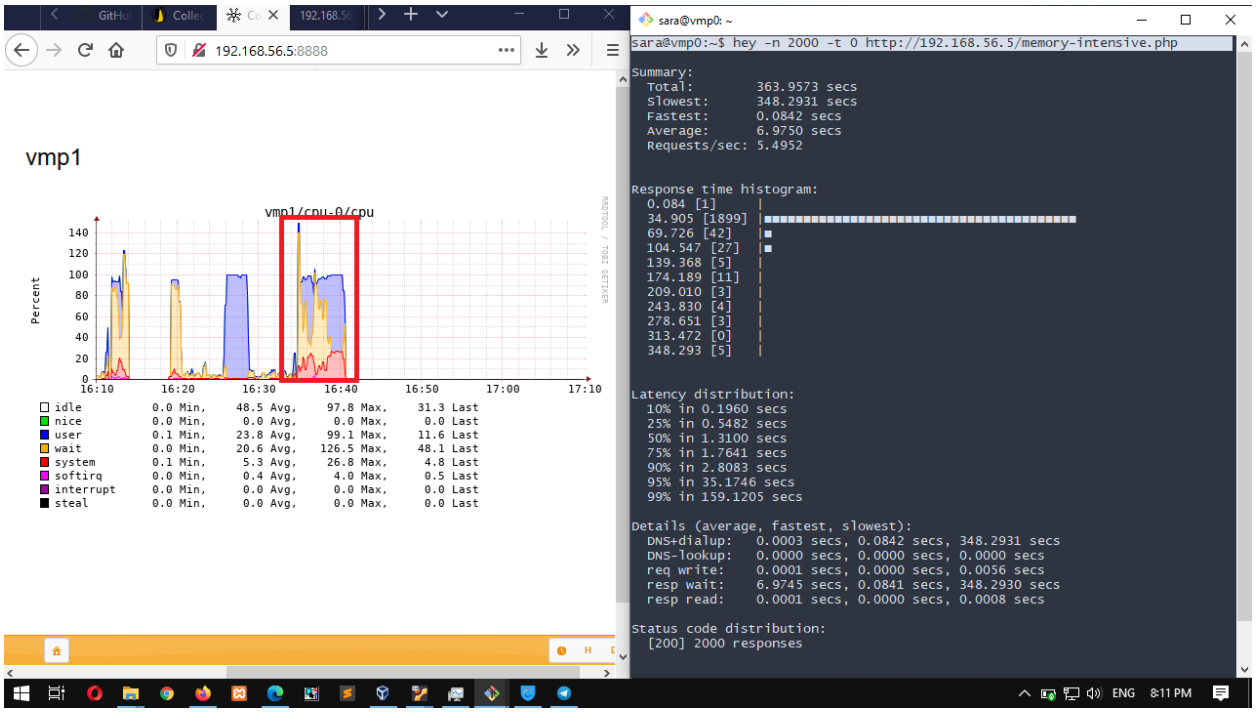
Memory

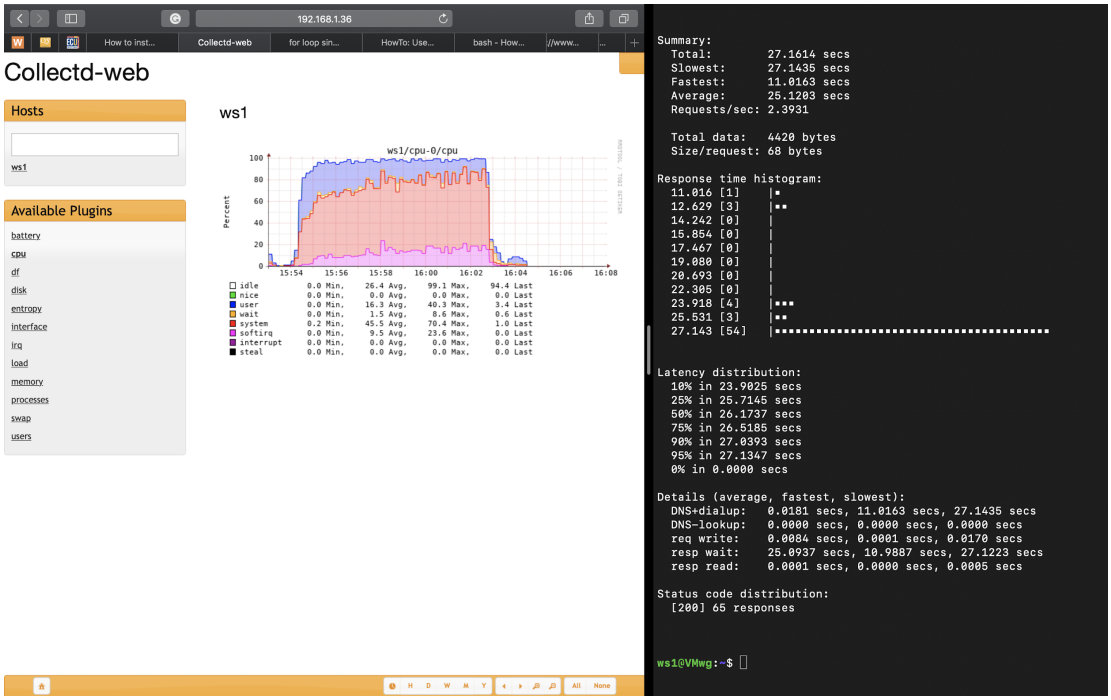

sara@vmp0:~$ for i in {1..65}; do hey -n $i -c $i -t 0 http://192.168.56.5/cpu-intensive.php; sleep 0.1; done
Summary:
Total: 0.2280 secs
Slowest: 0.2280 secs
Fastest: 0.2280 secs
Average: 0.2280 secs
Requests/sec: 4.3855
Response time histogram:
192.168.56.5:8888
Hosts
vmp1
Menu
Available Plugins
battery
cpu
df
disk
entropy
vmp1/memory/memory
Bytes
free 608.7M Min, 658.0M Avg, 694.3M Max, 658.6M Last
cached 65.4M Min, 93.6M Avg, 100.9M Max, 100.9M Last
buffered 9.4M Min, 12.1M Avg, 13.1M Max, 13.1M Last
used 104.5M Min, 129.9M Avg, 210.7M Max, 122.2M Last
slab recl 27.3M Min, 27.8M Avg, 28.1M Max, 28.1M Last



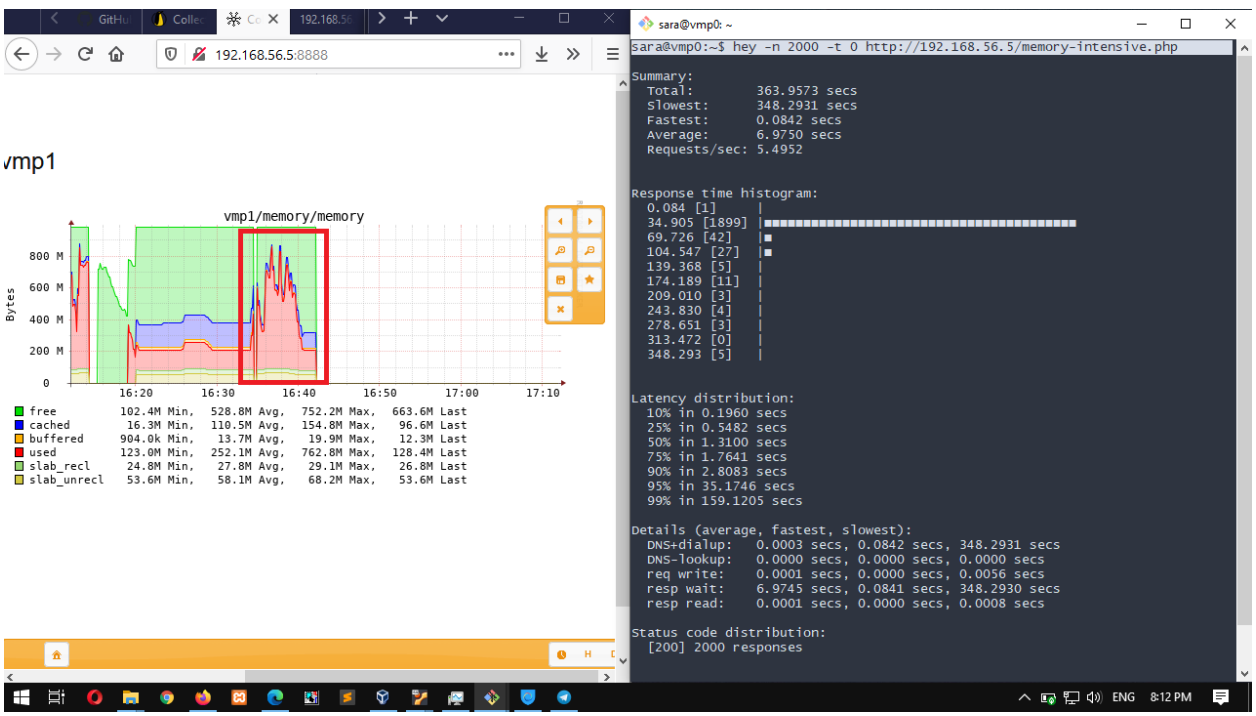
Memory-intensive.php

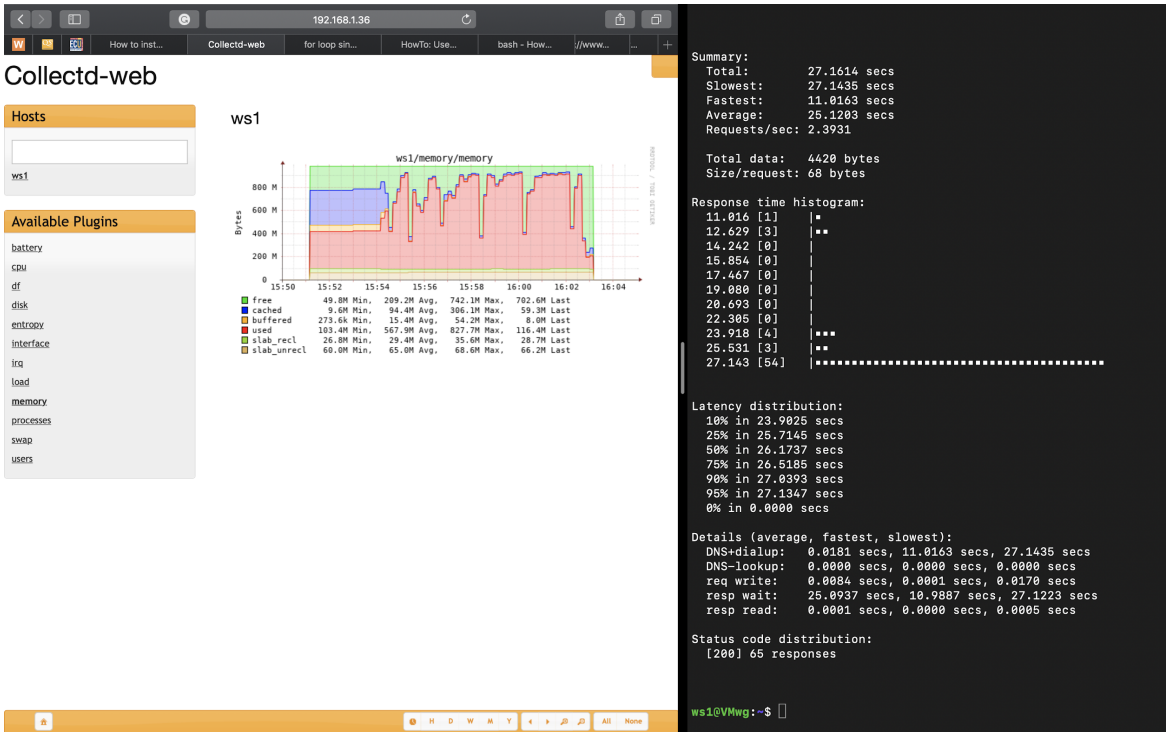
Cpu




192.168.1.36
Collectd-web
Hosts
ws1
Available Plugins
battery
cpu
df
disk
entropy
interface
irq
load
memory
processes
swap
users
ws1
ws1/cpu-0/cpu
Percent
idle 0.0 Min, 26.4 Avg, 99.1 Max, 94.4 Last
nice 0.0 Min, 0.0 Avg, 0.0 Max, 0.0 Last
user 0.0 Min, 16.3 Avg, 40.3 Max, 3.4 Last
wait 0.0 Min, 1.5 Avg, 8.6 Max, 0.6 Last
system 0.2 Min, 45.5 Avg, 70.4 Max, 1.0 Last
softirq 0.0 Min, 9.5 Avg, 23.6 Max, 0.0 Last
interrupt 0.0 Min, 0.0 Avg, 0.0 Max, 0.0 Last
steal 0.0 Min, 0.0 Avg, 0.0 Max, 0.0 Last
Summary:
Total: 27.1614 secs
Slowest: 27.1435 secs
Fastest: 11.0163 secs
Average: 25.1203 secs
Requests/sec: 2.3931
Total data: 4420 bytes
Size/request: 68 bytes
Response time histogram:
11.016 [1]
12.629 [3]
14.242 [0]
15.854 [0]
17.467 [0]
19.080 [0]
20.693 [0]
22.305 [0]
23.918 [4]
25.531 [3]
27.143 [54]
Latency distribution:
10% in 23.9025 secs
25% in 25.7145 secs
50% in 26.1737 secs
75% in 26.5185 secs
90% in 27.0393 secs
95% in 27.1347 secs
0% in 0.0000 secs
Details (average, fastest, slowest):
DNS+dialup: 0.0181 secs, 11.0163 secs, 27.1435 secs
DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0000 secs
req write: 0.0084 secs, 0.0001 secs, 0.0170 secs
resp wait: 25.0937 secs, 10.9887 secs, 27.1223 secs
resp read: 0.0001 secs, 0.0000 secs, 0.0005 secs
Status code distribution:
[200] 65 responses
ws1@VMwg:~$

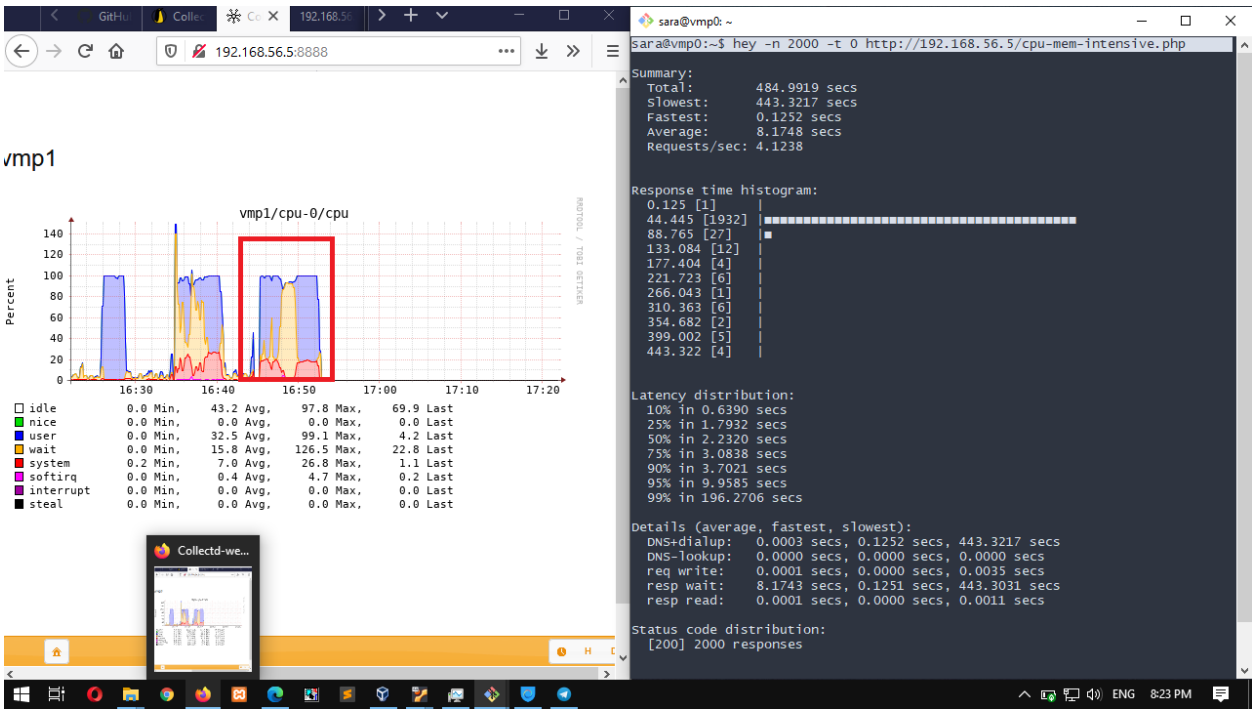
Memory

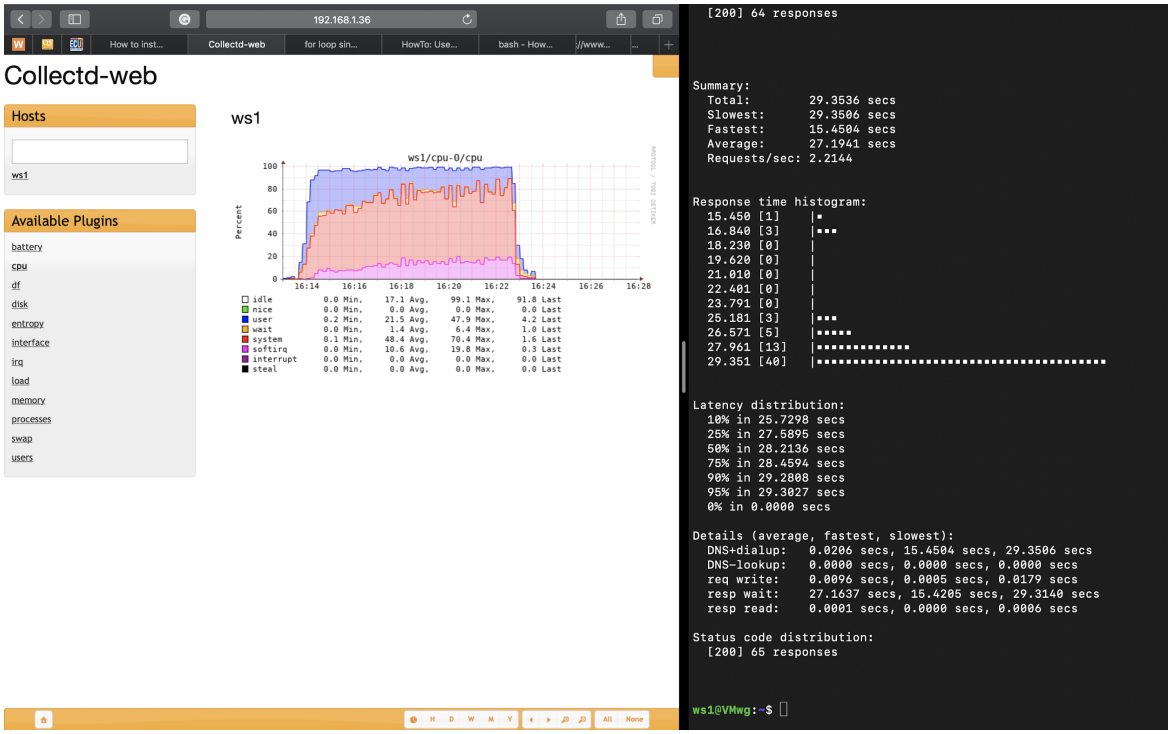

192.168.56.5:8888
vmp1
vmp1/memory/memory
Bytes
free 102.4M Min, 528.8M Avg, 752.2M Max, 663.6M Last
cached 16.3M Min, 110.5M Avg, 154.8M Max, 96.6M Last
buffered 904.0k Min, 13.7M Avg, 19.9M Max, 12.3M Last
used 123.0M Min, 252.1M Avg, 762.8M Max, 128.4M Last
slab_recl 24.8M Min, 27.8M Avg, 29.1M Max, 26.8M Last
slab_unrecl 53.6M Min, 58.1M Avg, 68.2M Max, 53.6M Last
sara@vmp0: ~
sara@vmp0:~$ hey -n 2000 -t 0 http://192.168.56.5/memory-intensive.php
Summary:
Total: 363.9573 secs
Slowest: 348.2931 secs
Fastest: 0.0842 secs
Average: 6.9750 secs
Requests/sec: 5.4952
Response time histogram:
0.084 [1]
34.905 [1899]
69.726 [42]
104.547 [27]
139.368 [5]
174.189 [11]
209.010 [3]
243.830 [4]
278.651 [3]
313.472 [0]
348.293 [5]
Latency distribution:
10% in 0.1960 secs
25% in 0.5482 secs
50% in 1.3100 secs
75% in 1.7641 secs
90% in 2.8083 secs
95% in 35.1746 secs
99% in 159.1205 secs
Details (average, fastest, slowest):
DNS+dialup: 0.0003 secs, 0.0842 secs, 348.2931 secs
DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0000 secs
req write: 0.0001 secs, 0.0000 secs, 0.0056 secs
resp wait: 6.9745 secs, 0.0841 secs, 348.2930 secs
resp read: 0.0001 secs, 0.0000 secs, 0.0008 secs
Status code distribution:
[200] 2000 responses
ENG 8:12 PM



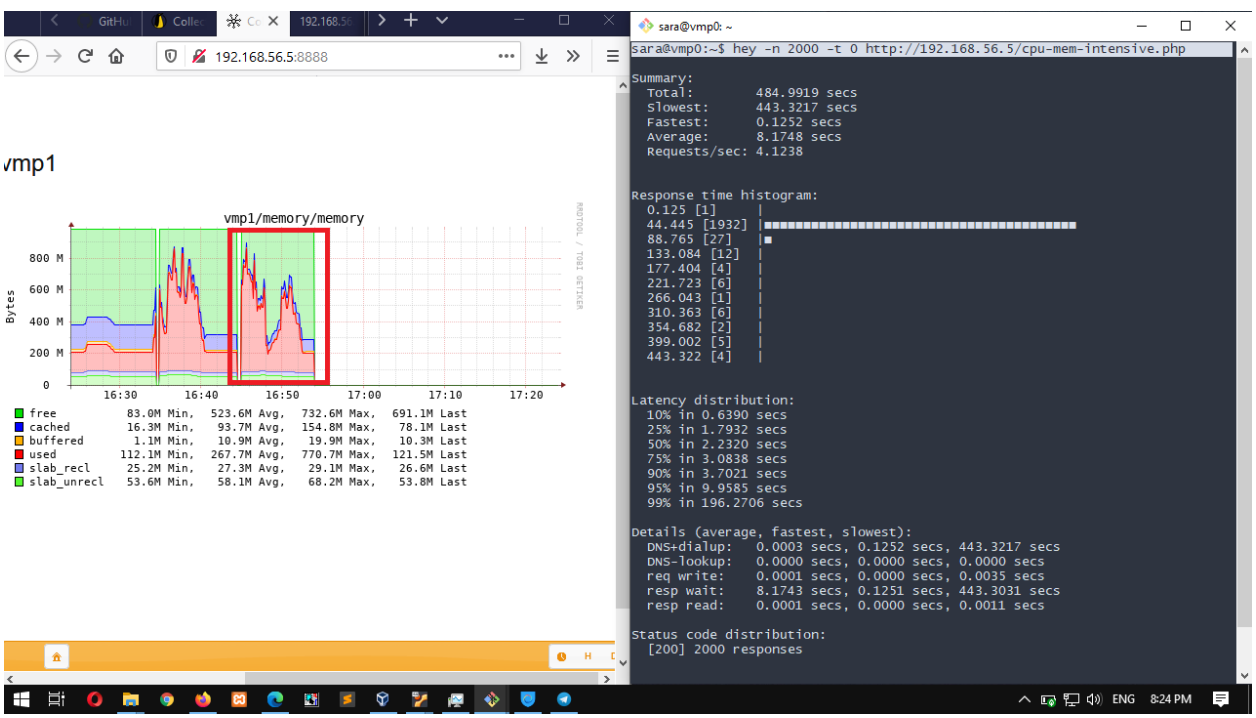
Cpu-mem-intensive.php

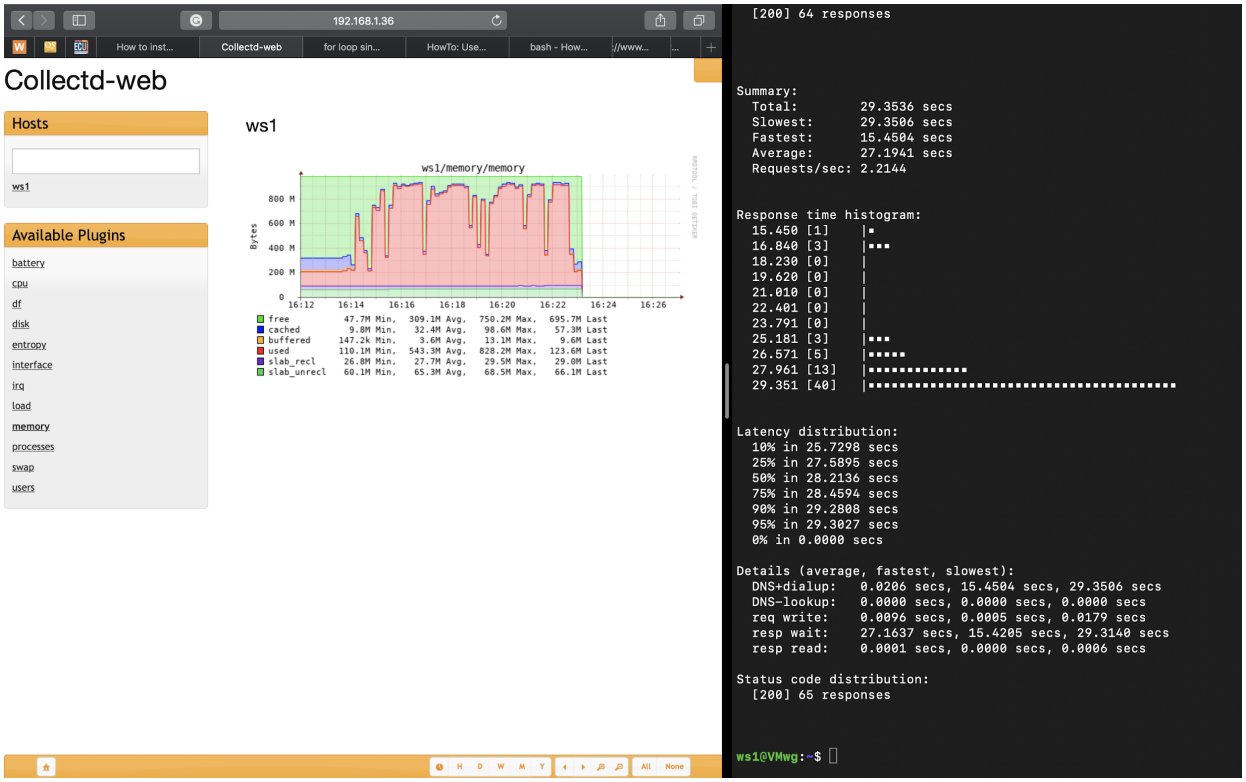
Cpu




192.168.1.36
Collectd-web
Hosts
ws1
Available Plugins
battery
cpu
df
disk
entropy
interface
irq
load
memory
processes
swap
users
ws1
ws1/cpu-0/cpu
Percent
idle 0.0 Min, 17.1 Avg, 99.1 Max, 91.8 Last
nice 0.0 Min, 0.0 Avg, 0.0 Max, 0.0 Last
user 0.2 Min, 21.5 Avg, 47.9 Max, 4.2 Last
wait 0.0 Min, 1.4 Avg, 6.4 Max, 1.0 Last
system 0.1 Min, 48.4 Avg, 70.4 Max, 1.6 Last
softirq 0.0 Min, 10.6 Avg, 19.8 Max, 0.3 Last
interrupt 0.0 Min, 0.0 Avg, 0.0 Max, 0.0 Last
steal 0.0 Min, 0.0 Avg, 0.0 Max, 0.0 Last
[200] 64 responses
Summary:
Total: 29.3536 secs
Slowest: 29.3506 secs
Fastest: 15.4504 secs
Average: 27.1941 secs
Requests/sec: 2.2144
Response time histogram:
15.450 [1]
16.840 [3]
18.230 [0]
19.620 [0]
21.010 [0]
22.401 [0]
23.791 [0]
25.181 [3]
26.571 [5]
27.961 [13]
29.351 [40]
Latency distribution:
10% in 25.7298 secs
25% in 27.5895 secs
50% in 28.2136 secs
75% in 28.4594 secs
90% in 29.2808 secs
95% in 29.3027 secs
0% in 0.0000 secs
Details (average, fastest, slowest):
DNS+dialup: 0.0206 secs, 15.4504 secs, 29.3506 secs
DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0000 secs
req write: 0.0096 secs, 0.0005 secs, 0.0179 secs
resp wait: 27.1637 secs, 15.4205 secs, 29.3140 secs
resp read: 0.0001 secs, 0.0000 secs, 0.0006 secs
Status code distribution:
[200] 65 responses
ws1@VMwg:~$

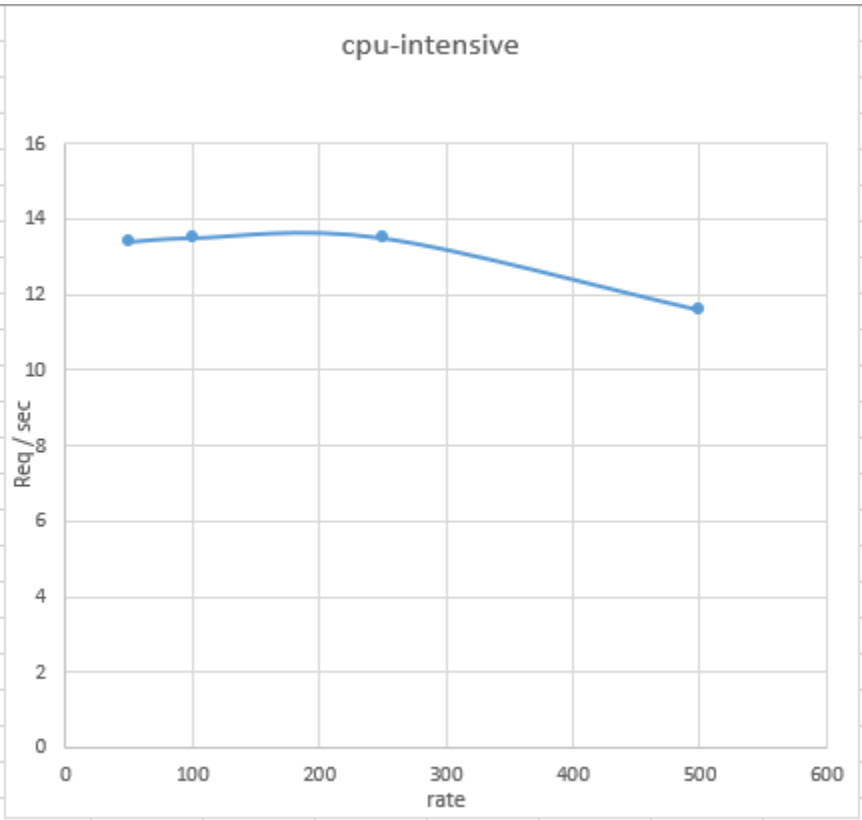
Memory


192.168.56.5:8888
vmp1
vmp1/memory/memory
Bytes
free 83.0M Min, 523.6M Avg, 732.6M Max, 691.1M Last
cached 16.3M Min, 93.7M Avg, 154.8M Max, 78.1M Last
buffered 1.1M Min, 10.9M Avg, 19.9M Max, 10.3M Last
used 112.1M Min, 267.7M Avg, 770.7M Max, 121.5M Last
slab_recl 25.2M Min, 27.3M Avg, 29.1M Max, 26.6M Last
slab_unrecl 53.6M Min, 58.1M Avg, 68.2M Max, 53.8M Last
sara@vmp0: ~
sara@vmp0:~$ hey -n 2000 -t 0 http://192.168.56.5/cpu-mem-intensive.php
Summary:
Total: 484.9919 secs
Slowest: 443.3217 secs
Fastest: 0.1252 secs
Average: 8.1748 secs
Requests/sec: 4.1238
Response time histogram:
0.125 [1]
44.445 [1932]
88.765 [27]
133.084 [12]
177.404 [4]
221.723 [6]
266.043 [1]
310.363 [6]
354.682 [2]
399.002 [5]
443.322 [4]
Latency distribution:
10% in 0.6390 secs
25% in 1.7932 secs
50% in 2.2320 secs
75% in 3.0838 secs
90% in 3.7021 secs
95% in 9.9585 secs
99% in 196.2706 secs
Details (average, fastest, slowest):
DNS+dialup: 0.0003 secs, 0.1252 secs, 443.3217 secs
DNS-lookup: 0.0000 secs, 0.0000 secs, 0.0000 secs
req write: 0.0001 secs, 0.0000 secs, 0.0035 secs
resp wait: 8.1743 secs, 0.1251 secs, 443.3031 secs
resp read: 0.0001 secs, 0.0000 secs, 0.0011 secs
Status code distribution:
[200] 2000 responses
ENG 8:24 PM

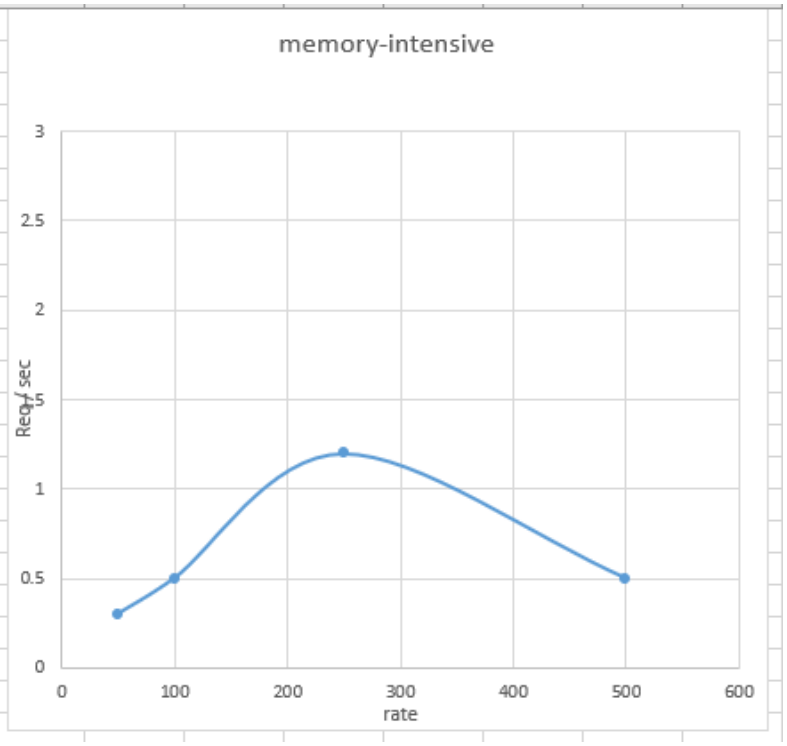

192.168.1.36
How to inst...
Collectd-web
for loop sin...
HowTo: Use...
bash - How...
Collectd-web
Hosts
ws1
Available Plugins
battery
cpu
df
disk
entropy
interface
irq
load
memory
processes
swap
users
ws1
ws1/memory/memory
Bytes
800 M
600 M
400 M
200 M
0
16:12
16:14
16:16
16:18
16:20
16:22
16:24
16:26
free 47.7M Min, 309.1M Avg, 750.2M Max, 695.7M Last
cached 9.8M Min, 32.4M Avg, 98.6M Max, 57.3M Last
buffered 147.2k Min, 3.6M Avg, 13.1M Max, 9.6M Last
used 110.1M Min, 543.3M Avg, 828.2M Max, 123.6M Last
slab_recl 26.8M Min, 27.7M Avg, 29.5M Max, 29.0M Last
slab_unrecl 60.1M Min, 65.3M Avg, 68.5M Max, 66.1M Last
All
None
[200] 64 responses
Summary:
Total: 29.3536 secs
Slowest: 29.3506 secs
Fastest: 15.4504 secs
Average: 27.1941 secs
Requests/sec: 2.2144
Response time histogram:
15.450 [1] |■
16.840 [3] |■■■
18.230 [0] |
19.620 [0] |
21.010 [0] |
22.401 [0] |
23.791 [0] |
25.181 [3] |■■■
26.571 [5] |■■■■■
27.961 [13] |■■■■■■■■■■■■■
29.351 [40] |■■■■■■■■■■■■■

با استفاده از **httperf**:

نرخ اسمی	نرخ واقعی
50	13.4
100	13.5
250	13.5
500	11.6

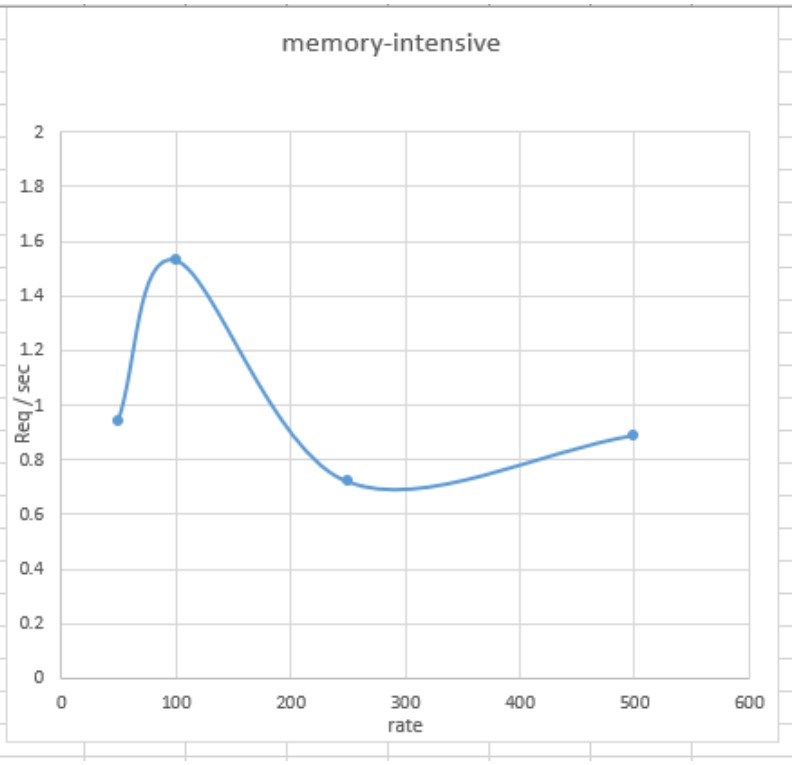


نرخ اسمی	نرخ واقعی
50	0.3
100	0.5
250	1.2
500	0.5

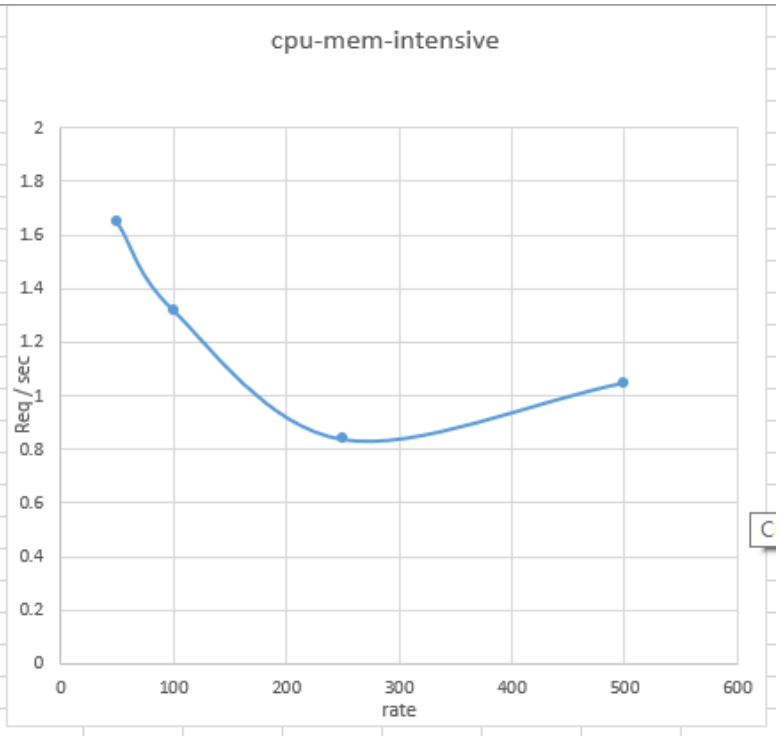


با استفاده از hey:

نرخ اسمی	نرخ واقعی
50	0.94
100	1.53
250	0.72
500	0.89

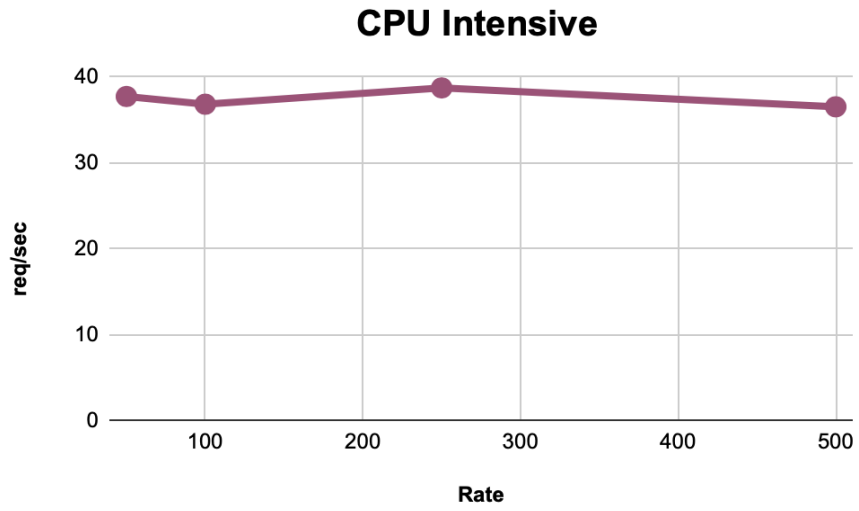


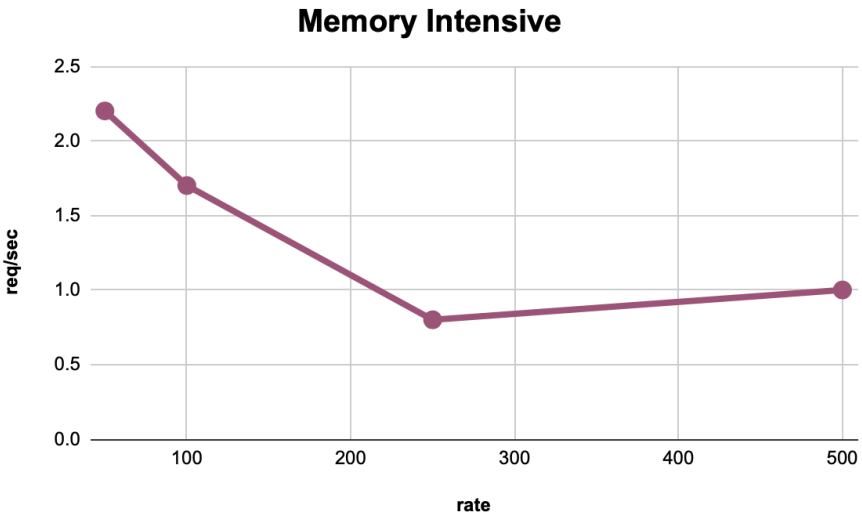
نرخ اسمی	نرخ واقعی
50	1.65
100	1.32
250	0.84
500	1.05

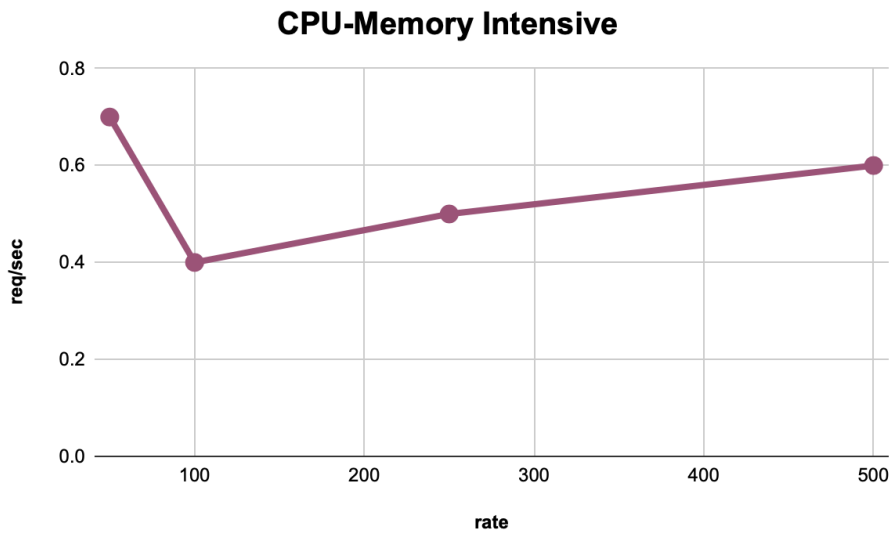


- با استفاده از **httperf**

		Rate	req/sec
CPU			
		50	37.7
		100	36.8
		250	38.7
		500	36.5
Mem			
		50	2.2
		100	1.7
		250	0.8
		500	1.0
CPU-Mem			
		50	0.7
		100	0.4
		250	0.5
		500	0.6


CPU Intensive
40
30
20
10
0
req/sec
100
200
300
400
500
Rate


Memory Intensive
2.5
2.0
1.5
1.0
0.5
0.0
req/sec
100
200
300
400
500
rate



بخش 5-2:

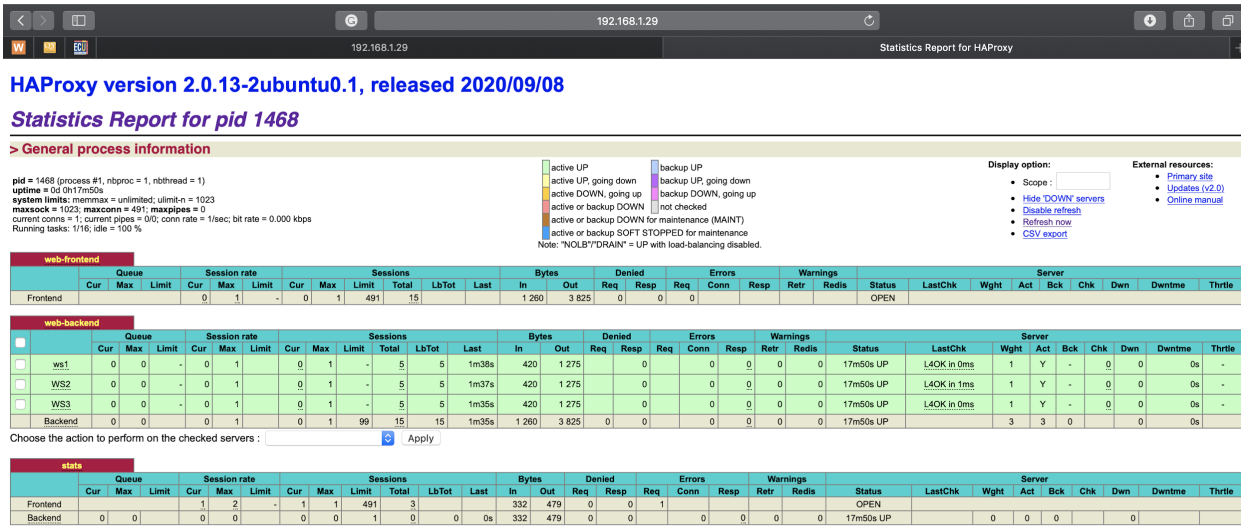
- نصب و پیکربندی haproxy بر روی VMlb:

```
sara@vmp1:~$ sudo nano /etc/hosts
[sudo] password for sara:
sara@vmp1:~$ sudo nano /etc/hosts
sara@vmp1:~$ sudo nano /etc/hosts
sara@vmp1:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /lib/s
temd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
sara@vmp1:~$ sudo systemctl start apache2
sara@vmp1:~$ echo "<H1>Hello! This is webserver1: 192.168.56.5 </H1>" | sud
tee /var/www/html/index.html
<H1>Hello! This is webserver1: 192.168.56.5 </H1>
sara@vmp1:~$ sudo ufw allow 80/tcp
Rules updated
Rules updated (v6)
sara@vmp1:~$ ufw reload
ERROR: You need to be root to run this script
sara@vmp1:~$ sudo ufw reload
Firewall not enabled (skipping reload)
sara@vmp1:~$ curl http://192.168.56.5/
<H1>Hello! This is webserver1: 192.168.56.5 </H1>
sara@vmp1:~$ |
```

```
sara@vmlb:~$ sudo systemctl status haproxy.service
● haproxy.service - HAProxy Load Balancer
     Loaded: loaded (/lib/systemd/system/haproxy.service; enabled; vendor p
     Active: active (running) since Wed 2021-02-03 11:23:20 UTC; 8min ago
       Docs: man:haproxy(1)
             file:/usr/share/doc/haproxy/configuration.txt.gz
    Process: 1457 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q $EXTRAOPT
   Main PID: 1469 (haproxy)
      Tasks: 2 (limit: 1074)
     Memory: 2.0M
     CGroup: /system.slice/haproxy.service
             ├─1469 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /r
             └─1470 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /r

Feb 03 11:23:20 vmlb systemd[1]: Starting HAProxy Load Balancer...
Feb 03 11:23:20 vmlb haproxy[1469]: Proxy web-frontend started.
Feb 03 11:23:20 vmlb haproxy[1469]: Proxy web-frontend started.
Feb 03 11:23:20 vmlb haproxy[1469]: Proxy web-backend started.
Feb 03 11:23:20 vmlb haproxy[1469]: Proxy web-backend started.
Feb 03 11:23:20 vmlb haproxy[1469]: Proxy stats started.
Feb 03 11:23:20 vmlb haproxy[1469]: Proxy stats started.
Feb 03 11:23:20 vmlb haproxy[1469]: [NOTICE] 033/112320 (1469) : New worker
Feb 03 11:23:20 vmlb systemd[1]: Started HAProxy Load Balancer.
lines 1-22/22 (END)
```

```
sara@vmlb:~$ ping 192.168.56.8
PING 192.168.56.8 (192.168.56.8) 56(84) bytes of data.
64 bytes from 192.168.56.8: icmp_seq=1 ttl=64 time=0.344 ms
64 bytes from 192.168.56.8: icmp_seq=2 ttl=64 time=0.910 ms
64 bytes from 192.168.56.8: icmp_seq=3 ttl=64 time=0.717 ms
64 bytes from 192.168.56.8: icmp_seq=4 ttl=64 time=0.737 ms
^C
--- 192.168.56.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3018ms
rtt min/avg/max/mdev = 0.344/0.677/0.910/0.206 ms
sara@vmlb:~$ ping vmp1
PING vmp1 (192.168.56.5) 56(84) bytes of data.
64 bytes from vmp1 (192.168.56.5): icmp_seq=1 ttl=64 time=0.353 ms
64 bytes from vmp1 (192.168.56.5): icmp_seq=2 ttl=64 time=0.629 ms
64 bytes from vmp1 (192.168.56.5): icmp_seq=3 ttl=64 time=0.526 ms
64 bytes from vmp1 (192.168.56.5): icmp_seq=4 ttl=64 time=0.416 ms
64 bytes from vmp1 (192.168.56.5): icmp_seq=5 ttl=64 time=0.620 ms
^C
--- vmp1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4091ms
rtt min/avg/max/mdev = 0.353/0.508/0.629/0.109 ms
sara@vmlb:~$
```



- نمایش درستی عملکرد خط مشی round-robin: (از طریق apache log)

```
sara@vmlb:~$ while true; do curl 192.168.56.9; sleep 1; done
<H1>Hello! This is webserver1: 192.168.56.5 </H1>
<H1>Hello! This is webserver2: 192.168.56.7 </H1>
<H1>Hello! This is webserver3: 192.168.56.8 </H1>
<H1>Hello! This is webserver1: 192.168.56.5 </H1>
<H1>Hello! This is webserver2: 192.168.56.7 </H1>
<H1>Hello! This is webserver3: 192.168.56.8 </H1>
<H1>Hello! This is webserver1: 192.168.56.5 </H1>
<H1>Hello! This is webserver2: 192.168.56.7 </H1>
^C
sara@vmlb:~$
```

```
6 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


Last login: Thu Feb  4 15:14:54 2021 from 192.168.56.1
ws1@VMlb:~$ sudo systemctl restart haproxy.service
[sudo] password for ws1:
ws1@VMlb:~$ sudo systemctl status haproxy.service
● haproxy.service - HAProxy Load Balancer
     Loaded: loaded (/lib/systemd/system/haproxy.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2021-02-04 15:18:03 UTC; 6s ago
       Docs: man:haproxy(1)
             file:/usr/share/doc/haproxy/configuration.txt.gz
    Process: 1273 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q $EXTRAOPTS (code=exited, status=0/SUCCESS)
   Main PID: 1283 (haproxy)
      Tasks: 2 (limit: 1074)
     Memory: 2.1M
     CGroup: /system.slice/haproxy.service
             ├─1283 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -S /run/haproxy-master.sock
             └─1284 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -S /run/haproxy-master.sock

Feb 04 15:18:03 VMlb systemd[1]: Starting HAProxy Load Balancer...
Feb 04 15:18:03 VMlb haproxy[1283]: Proxy web-frontend started.
Feb 04 15:18:03 VMlb haproxy[1283]: Proxy web-frontend started.
Feb 04 15:18:03 VMlb haproxy[1283]: Proxy web-backend started.
Feb 04 15:18:03 VMlb haproxy[1283]: Proxy web-backend started.
Feb 04 15:18:03 VMlb haproxy[1283]: Proxy stats started.
Feb 04 15:18:03 VMlb haproxy[1283]: Proxy stats started.
Feb 04 15:18:03 VMlb haproxy[1283]: [NOTICE] 034/151803 (1283) : New worker #1 (1284) forked
Feb 04 15:18:03 VMlb systemd[1]: Started HAProxy Load Balancer.
ws1@VMlb:~$ while true; do curl 192.168.56.6; sleep 1; done
<H1>Hello! This is webserver1 </H1>
<H1>Hello! This is webserver2 </H1>
<H1>Hello! This is webserver3 </H1>
<H1>Hello! This is webserver1 </H1>
<H1>Hello! This is webserver2 </H1>
<H1>Hello! This is webserver3 </H1>
<H1>Hello! This is webserver1 </H1>
<H1>Hello! This is webserver2 </H1>
<H1>Hello! This is webserver3 </H1>
<H1>Hello! This is webserver1 </H1>
<H1>Hello! This is webserver2 </H1>
<H1>Hello! This is webserver3 </H1>
^C
ws1@VMlb:~$
```

192.168.56.9/cpu-intensive.php

CPU intensive request
1.0001 to the power of 1000000 = 2.6747109931127E+43

```
sara@vmp2: ~
101 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:13:00:13 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/20100101
 Firefox/85.0"
sara@vmp2:~$

sara@vmp3: ~
0101 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:13:00:16 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/2010010
1 Firefox/85.0"
sara@vmp3:~$

sara@vmp1: ~
0101 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:13:00:17 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/2010010
1 Firefox/85.0"
sara@vmp1:~$
```

192.168.56.9/memory-intensive.php

Memory intensive request
Array is created
for example: arr[25] = 25

```
sara@vmp2: ~
1 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:12:53:11 +0000] "GET /memory-intensive.php HTTP/1.1" 200 285 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/2010
0101 Firefox/85.0"
sara@vmp2:~$

sara@vmp3: ~
1 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:12:53:14 +0000] "GET /memory-intensive.php HTTP/1.1" 200 285 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/2010
0101 Firefox/85.0"
sara@vmp3:~$

sara@vmp1: ~
1 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:12:53:16 +0000] "GET /memory-intensive.php HTTP/1.1" 200 285 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/2010
0101 Firefox/85.0"
sara@vmp1:~$
```

192.168.56.9/cpu-mem-intensive.php

CPU and Memory intensive request
Array is created
For example: arr[99999] = 22015.456048528

```
sara@vmp2: ~
 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:13:02:45 +0000] "GET /cpu-mem-intensive.php HTTP/1.1" 200 310 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/2010
0101 Firefox/85.0"
sara@vmp2:~$

sara@vmp3: ~
1 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:13:02:47 +0000] "GET /cpu-mem-intensive.php HTTP/1.1" 200 310 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/201
00101 Firefox/85.0"
sara@vmp3:~$

sara@vmp1: ~
1 Firefox/85.0"
192.168.56.9 - - [03/Feb/2021:13:02:49 +0000] "GET /cpu-mem-intensive.php HTTP/1.1" 200 310 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/201
00101 Firefox/85.0"
sara@vmp1:~$
```

Web Server 1:

```
192.168.1.29 - - [03/Feb/2021:12:38:44 +0000] "GET / HTTP/1.1" 200 263 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0 Sa
fari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:38:45 +0000] "GET / HTTP/1.1" 200 263 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0 Sa
fari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:48:53 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko
) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:52:29 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko
) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:52:55 +0000] "GET /memory-intensive.php HTTP/1.1" 200 216 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Ge
cko) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:53:20 +0000] "GET /cpu-mem-intensive.php HTTP/1.1" 200 289 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like G
ecko) Version/12.0 Safari/605.1.15"
ws1@ws1:~$
```

Web Server 2:

```
fari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:38:50 +0000] "GET / HTTP/1.1" 200 263 "-" "curl/7.68.0"
192.168.1.29 - - [03/Feb/2021:12:48:49 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko
) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:48:56 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko
) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:52:50 +0000] "GET /memory-intensive.php HTTP/1.1" 200 216 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Ge
cko) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:53:15 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko
) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:53:23 +0000] "GET /cpu-mem-intensive.php HTTP/1.1" 200 289 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like G
ecko) Version/12.0 Safari/605.1.15"
ws1@WS2:~$
```

Web Server 3:

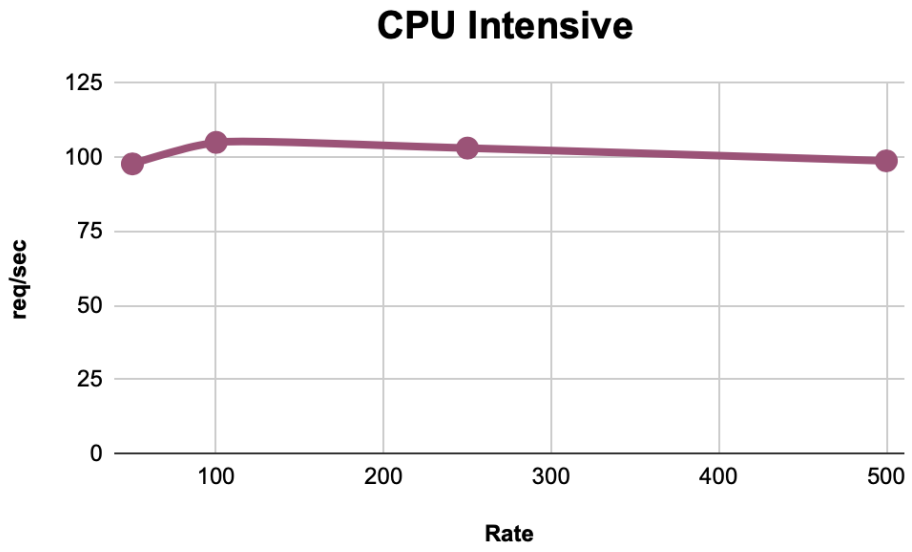
```
192.168.1.29 - - [03/Feb/2021:12:38:44 +0000] "GET / HTTP/1.1" 200 263 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.15 (KHTML, like Ge
cko) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:48:49 +0000] "GET /favicon.ico HTTP/1.1" 404 435 "http://192.168.1.29/cpu-intensive.php" "Mozilla/5.0 (Macintosh; Intel Mac O
S X 10_14) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:48:51 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.1
5 (KHTML, like Gecko) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:48:57 +0000] "GET /cpu-intensive.php HTTP/1.1" 200 291 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.1.1
5 (KHTML, like Gecko) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:52:53 +0000] "GET /memory-intensive.php HTTP/1.1" 200 216 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605.
1.15 (KHTML, like Gecko) Version/12.0 Safari/605.1.15"
192.168.1.29 - - [03/Feb/2021:12:53:18 +0000] "GET /cpu-mem-intensive.php HTTP/1.1" 200 289 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14) AppleWebKit/605
.1.15 (KHTML, like Gecko) Version/12.0 Safari/605.1.15"
ws1@WS3:~$
```

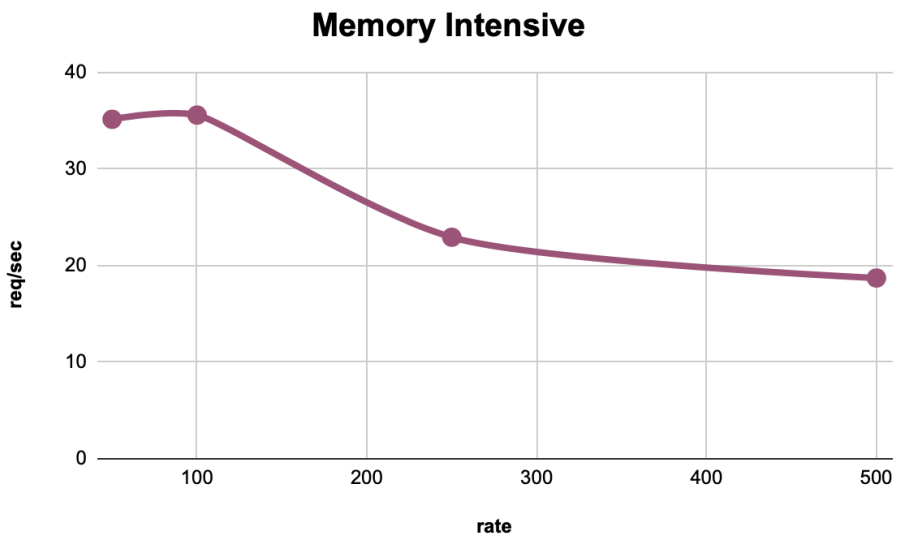
بخش 6-2:

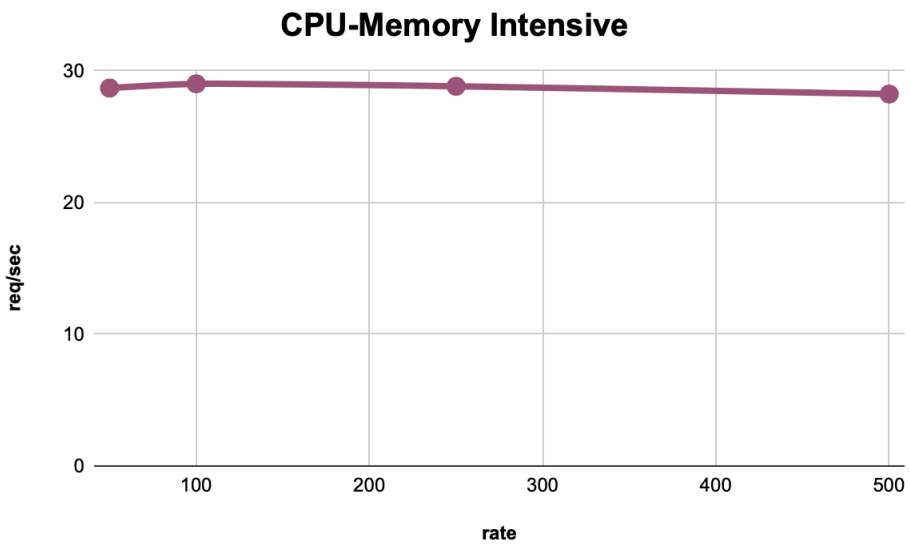
- رسم نمودار throughput برای load balancer:

- با استفاده از hey

		Rate	req/sec
CPU			
		50	97.7
		100	104.95
		250	103.02
		500	98.72
Mem			
		50	35.0872
		100	35.5223
		250	22.8668
		500	18.6458
CPU-Mem			
		50	28.6941
		100	29.0173
		250	28.828
		500	28.2359

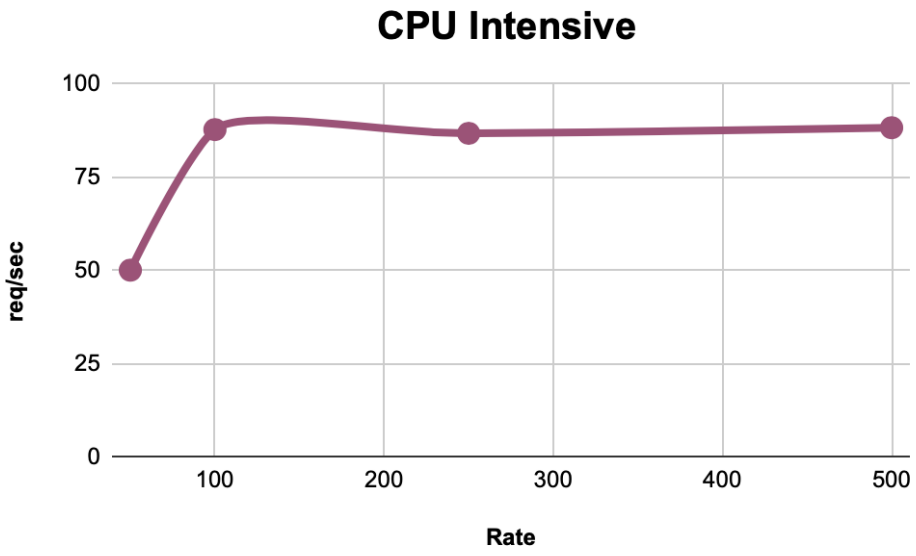


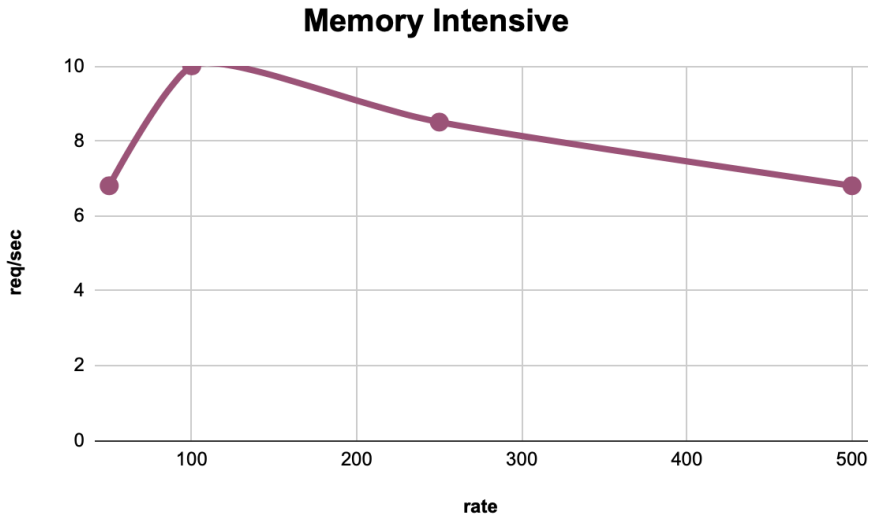

Memory Intensive
40
30
20
10
0
req/sec
100
200
300
400
500
rate

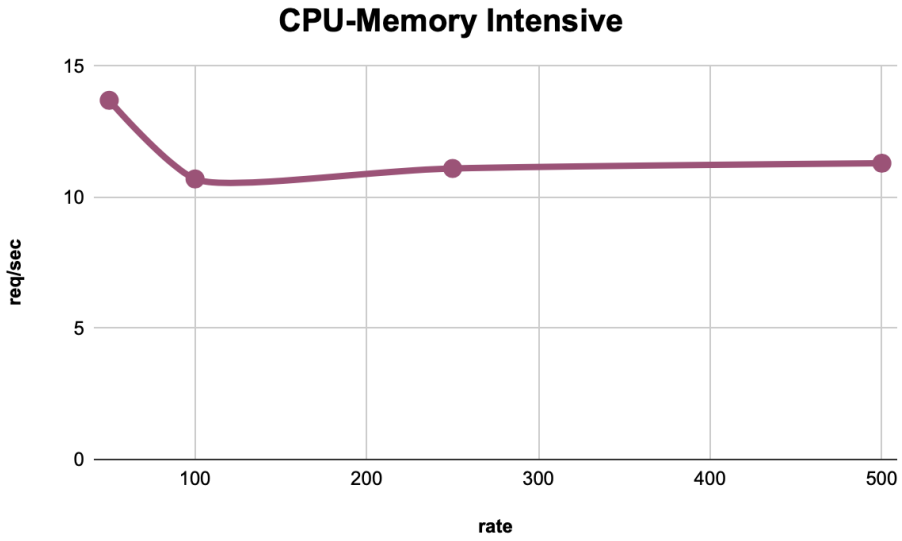

CPU-Memory Intensive
30
20
10
0
req/sec
100
200
300
400
500
rate

- با استفاده از **httperf**

		Rate	req/sec
CPU			
		50	50
		100	87.7
		250	86.7
		500	88.2
Mem			
		50	6.8
		100	10
		250	8.5
		500	6.8
CPU-Mem			
		50	13.7
		100	10.7
		250	11.1
		500	11.3




Memory Intensive
req/sec
10
8
6
4
2
0
100
200
300
400
500
rate


CPU-Memory Intensive
req/sec
15
10
5
0
100
200
300
400
500
rate

بررسی تفاوت این نمودار با نمودار بخش 4-2:

اگر req/sec از حدی بیشتر باشد با یک سرور نمی‌توان آن را هندل کرد و به همین دلیل درخواست‌ها fail می‌شوند و شیب نمودار از یک جا به بعد نزولی می‌شود. ولی هنگامی که از load balancer از تعدادی سرور استفاده می‌کند و درخواست‌ها بین آن‌ها پخش می‌شود در نتیجه بار بین سرورهایی که بار کاری کمتری دارند تقسیم می‌شود و باعث می‌شود که درخواست‌ها fail نشوند پس شیب نمودارها تغییر چندانی نمی‌کند.

طبق تعریف، throughput برابر است با تعداد درخواست های تکمیل شده در برابر تعداد درخواست های ارسال شده. همانطور که مشاهده می شود، نمودارهای بخش 4-2 ابتدا یک شیب صعودی دارند و از یک نقطه به بعد شیب نزولی پیدا می کنند. دلیل این اتفاق این است که بدون وجود load balancer و در صورت استفاده از تنها یک سرور، این امکان وجود دارد که تعداد درخواست ها در هر ثانیه از میزان تحمل بار سرور عبور کند و در نتیجه درخواست ها fail شده و از بین بروند. اما نمودارهای بخش 6-2، همانطور که ملاحظه می کنید، اکثرا دارای یک شیب ثابت صفر می باشند. زیرا با وجود load balancer، درخواست های ارسالی بین چندین سرور تقسیم بندی می شود و بدین ترتیب تعداد درخواست های fail شده کاهش و سرعت رسیدگی به آنها افزایش می یابد و احتمال crash کردن کل سیستم نیز تا حد قابل توجهی پایین می آید.

بخش 7-2:

- پیاده سازی بستر نرم افزاری:

کد مربوط به این قسمت به همراه گزارش در یک فایل جداگانه قرار داده شده است. (شامل کد سمت load balancer و کد سمت host)

برای پیاده‌سازی این بستر نرم‌افزاری ابتدا نیاز بود که اطلاعات مربوط به cpu و memory از backend serverها گرفته‌شود که برای این کار ما از csv plugin مربوط به collectd استفاده‌کردیم.

در مرحله بعد کد مربوط به قواعد load balancing را در ماشین مجازی load balancer نوشتیم.

در سمت host طبق دستور گرفته‌شده از load balancer وب سرورها را با استفاده از vboxmanage خاموش و روشن کردیم.

- نمایش درستی عملکرد بستر نرم افزاری:

```
ws1@VMlb:~/autoscale$ python3 program.py
percent-user-2021-02-05
             100%   54KB  12.1MB/s   00:00
0.1588355
percent-user-2021-02-05
             100%   54KB  10.1MB/s   00:00
15.467391833333334
percent-user-2021-02-05
             100%   54KB  16.4MB/s   00:00
47.341957
percent-user-2021-02-05
             100%   54KB   9.0MB/s   00:00
71.13357241666667
percent-user-2021-02-05
             100%   54KB   6.1MB/s   00:00
94.02174008333333
Turning On WS2
percent-user-2021-02-05
             100%   54KB  34.3MB/s   00:00
94.10908258333333
percent-user-2021-02-05
             100%   26KB 324.2KB/s   00:00
94.10908258333333
7.006809916666667
percent-user-2021-02-05
             100%   55KB  13.2MB/s   00:00
91.50816408333333
percent-user-2021-02-05
             100%   26KB  11.2MB/s   00:00
91.50816408333333
26.75755291666667
percent-user-2021-02-05
             100%   55KB   6.4MB/s   00:00
90.63914225
percent-user-2021-02-05
             100%   26KB  11.5MB/s   00:00
90.63914225
57.402496666666664
percent-user-2021-02-05
             100%   55KB  12.1MB/s   00:00
89.7416315
percent-user-2021-02-05
             100%   26KB  10.2MB/s   00:00
89.7416315
80.67516116666667
Turning ON WS3
percent-user-2021-02-05
             100%   55KB  11.9MB/s   00:00
percent-user-2021-02-05
```

```
percent-user-2021-02-05
            100%   27KB  25.1MB/s   00:00
percent-user-2021-02-05
            100%   17KB   6.3MB/s   00:00
92.45126441666667
85.92887908333333
5.24085175
percent-user-2021-02-05
            100%   55KB   9.7MB/s   00:00
percent-user-2021-02-05
            100%   27KB   5.9MB/s   00:00
percent-user-2021-02-05
            100%   17KB   2.2MB/s   00:00
92.64052316666667
71.120544
24.511741666666666
percent-user-2021-02-05
            100%   55KB  11.3MB/s   00:00
percent-user-2021-02-05
            100%   27KB  17.1MB/s   00:00
percent-user-2021-02-05
            100%   17KB   5.9MB/s   00:00
78.582386
57.15476591666667
40.322657666666665
percent-user-2021-02-05
            100%   55KB  19.0MB/s   00:00
percent-user-2021-02-05
            100%   27KB   6.8MB/s   00:00
percent-user-2021-02-05
            100%   17KB   7.1MB/s   00:00
56.27591691666667
38.47997816666667
41.00156058333334
percent-user-2021-02-05
            100%   55KB   5.3MB/s   00:00
percent-user-2021-02-05
            100%   27KB  14.2MB/s   00:00
```

```
percent-user-2021-02-05
            100%   17KB   6.7MB/s   00:00
25.944797666666666
27.278319583333335
30.265449583333336
Turning OFF WS3
percent-user-2021-02-05
            100%   55KB  26.5MB/s   00:00
percent-user-2021-02-05
            100%   27KB  19.3MB/s   00:00
6.38855575
9.5762375
Turning OFF WS2
percent-user-2021-02-05
            100%   55KB  15.3MB/s   00:00
1.9841886666666666
^Z
[18]+  Stopped                 python3 program.py
ws1@VMlb:~/autoscale$
```

بخش امتیازی:

برای این بخش وزن cpu و memory را برابر با هم و برابر با 0.5 گرفتیم. و درصد مربوط به قواعد را هم مشابه قسمت قبلی در نظرگرفتیم. برای به دست آوردن مقادیر درصد memory هم از فایل‌های csv مربوط به وب‌سرورها استفاده‌کردیم.

- **نمایش درستی عملکرد بستر نرم افزاری:**

```
ws1@VMlb:~/autoscale$ python3 hybridProgram.py
percent-user-2021-02-06
1KB   6.8MB/s   00:00
percent-used-2021-02-06
1KB   3.9MB/s   00:00
17.3739505
67.93143500000001
total percent for VM1:  59.713769850000006
percent-user-2021-02-06
1KB   6.4MB/s   00:00
percent-used-2021-02-06
2KB   7.6MB/s   00:00
28.80100825
63.84060075
total percent for VM1:  64.8491263
percent-user-2021-02-06
1KB   7.7MB/s   00:00
percent-used-2021-02-06
2KB   5.8MB/s   00:00
44.69156625
62.14206866666667
total percent for VM1:  74.78354444166666
percent-user-2021-02-06
2KB   3.4MB/s   00:00
percent-used-2021-02-06
2KB   7.2MB/s   00:00
56.61454008333333
75.65993391666666
total percent for VM1:  92.59213179999999
Turining WS2 ON
percent-user-2021-02-06
2KB   6.5MB/s   00:00
percent-used-2021-02-06
2KB   7.7MB/s   00:00
percent-user-2021-02-06
3     3.0MB/s   00:00
percent-used-2021-02-06
5     1.6MB/s   00:00
63.44343516666667
64.49940683333334
total percent for VM1:  89.5599894
7.40914825
64.49940683333334
total percent for VM2:  50.335988558333334
percent-user-2021-02-06
2KB   3.0MB/s   00:00
percent-used-2021-02-06
2KB   2.6MB/s   00:00
percent-user-2021-02-06
3     4.3MB/s   00:00
percent-used-2021-02-06
5     4.2MB/s   00:00
```

```
percent-user-2021-02-06
3KB   3.9MB/s   00:00
percent-used-2021-02-06
3KB   4.4MB/s   00:00
percent-user-2021-02-06
3     4.2MB/s   00:00
percent-used-2021-02-06
5     1.2MB/s   00:00
91.03432625
23.730686916666667
total percent for VM1:  80.33550921666667
91.37289458333333
23.730686916666667
total percent for VM2:  80.57250705
Turning WS3 ON
percent-user-2021-02-06
3KB   1.3MB/s   00:00
percent-used-2021-02-06
4KB   9.2MB/s   00:00
percent-user-2021-02-06
8     2.0MB/s   00:00
percent-used-2021-02-06
0     1.0MB/s   00:00
percent-user-2021-02-06
2     3.7MB/s   00:00
percent-used-2021-02-06
0     2.6MB/s   00:00
92.87255825
24.199547333333335
total percent for VM1:  81.95047390833332
92.88533016666666
24.199547333333335
total percent for VM2:  81.95941425
7.6520735
23.970889083333333
total percent for VM3:  22.13607380833333
percent-user-2021-02-06
3KB  11.5MB/s   00:00
percent-used-2021-02-06
4KB  11.1MB/s   00:00
percent-user-2021-02-06
8     2.7MB/s   00:00
percent-used-2021-02-06
0     4.8MB/s   00:00
percent-user-2021-02-06
2     4.1MB/s   00:00
percent-used-2021-02-06
0     1.8MB/s   00:00
85.5941975
24.002237166666667
total percent for VM1:  76.71750426666667
88.95613858333333
```

```
17.432421
total percent for VM1:  52.08933905
46.81109416666666
17.432421
total percent for VM2:  44.970460616666664
44.04292
16.747906083333334
total percent for VM3:  42.55357825833333
percent-user-2021-02-06
5KB   3.2MB/s   00:00
percent-used-2021-02-06
5KB  13.9MB/s   00:00
percent-user-2021-02-06
0     9.2MB/s   00:00
percent-used-2021-02-06
0     8.2MB/s   00:00
percent-user-2021-02-06
4     1.6MB/s   00:00
percent-used-2021-02-06
0     3.2MB/s   00:00
24.206063333333333
15.787429166666668
total percent for VM1:  27.995444749999997
17.880252666666667
15.787429166666668
total percent for VM2:  23.567377283333336
17.881054583333334
14.95290825
total percent for VM3:  22.983773983333332
Turning WS3 OFF
percent-user-2021-02-06
5KB   9.3MB/s   00:00
percent-used-2021-02-06
5KB   3.2MB/s   00:00
percent-user-2021-02-06
3     3.1MB/s   00:00
percent-used-2021-02-06
5     2.4MB/s   00:00
5.4545104166666665
14.903879333333332
total percent for VM1:  14.250872824999998
4.40781975
14.903879333333332
total percent for VM2:  13.51818935833333
Turning WS2 OFF
percent-user-2021-02-06
5KB  12.8MB/s   00:00
percent-used-2021-02-06
5KB  13.2MB/s   00:00
5.167450166666667
14.706337
total percent for VM1:  13.911651016666667
```