การใช้งาน Reverse Proxy โดย NGINX บน CentOS7

คมกริช คำสวัสดิ์
วิศวกรอาวุโส
สำนักงานรัฐบาลอิเล็กทรอนิกส์ (องค์การมหาชน)

การใช้งาน Reverse Proxy โดย NGINX บน CentOS7

- Reverse Proxy WIKIpedia
- ประโยชน์ของ Reverse Proxy
- www.nginx.org
- ข้อยกเว้นสำหรับเอกสารนี้
- การติดตั้ง NGINX
- การใช้งาน Reverse Proxy ผ่าน NGINX
- NGINX Logging
- Load Balance by NGINX
- การกำหนดให้ Request Path ไปยัง Backend ต่างเครื่อง
- HTTPS Offloading
- Redirect Users to HTTPS
- NGINX Caching
- การปรับแต่งค่า Configurations อื่นๆ
- แนะนำ Report สำหรับ NGINX (GoAccess)

Reverse Proxy

Ref. **https://en.wikipedia.org/wiki/Reverse_proxy**

In computer networks, a reverse proxy is a type of proxy server that retrieves resources on behalf of a client from one or more servers. These resources are then returned to the client as if they originated from the proxy server itself. While a forward proxy acts as an intermediary for its associated clients to contact any server, a reverse proxy acts as an intermediary for its associated servers to be contacted by any client.

Quite often, popular web servers utilize reverse-proxying functionality, acting as shields for application frameworks with weaker HTTP capabilities.

ประโยชน์ของ Reverse Proxy

Ref. **http://www.jscape.com/blog/bid/87841/Top-8-Benefits-of-a-Reverse-Proxy**

1. Creates a single point of access to your file transfer servers

2. Simplifies access control tasks

3. Moves user credentials to a safer place

4. Reduces risks to sensitive data

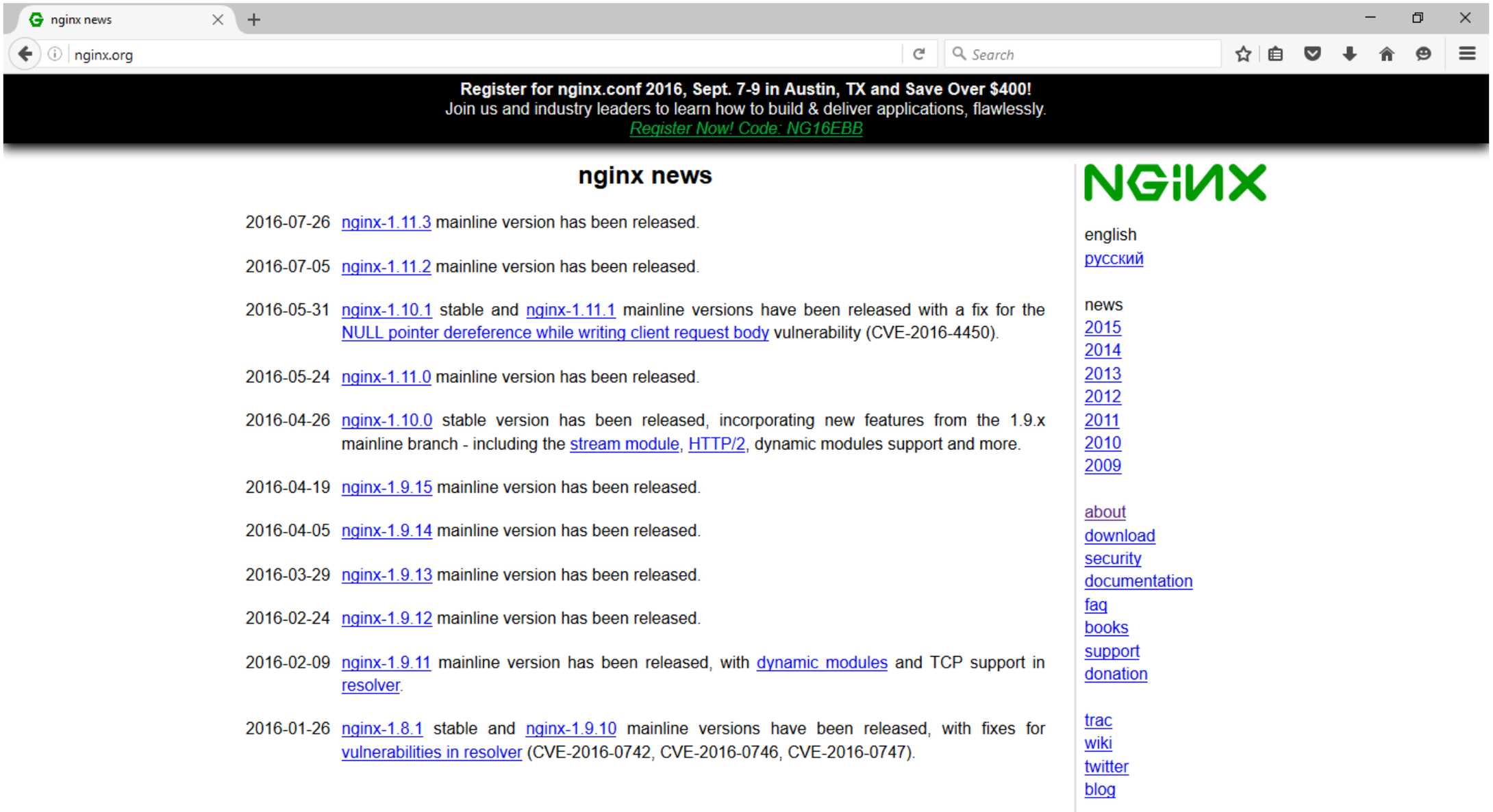
5. Helps achieve regulatory compliance

6. Brings down capital and operational expenses

7. Allows transparent maintenance of backend servers

8. Enables load balancing and failover

www.nginx.org



ข้อยกเว้นสำหรับเอกสารนี้ !!!!!!!!

เพื่อให้ง่ายในการทดสอบปฏิบัติใน Class นี้ จึงขอให้ทำการ Disable SELINUX ดังนี้

- ทำการแก้ไขไฟล์ /etc/selinux/config ดังนี้

SELINUX=disabled

- ทำการแก้ไขโหมดการทำงานของ SELINUX เป็น Permissive

[root@server1 ~]# **setenforce 0**

- ตรวจสอบการทำงานของ SELINUX

[root@server1 ~]# **getenforce**

Permissive

การติดตั้ง NGINX

ติดตั้ง NGINX

[root@server1 ~]# **yum -y install epel-release**

[root@server1 ~]# **yum -y install nginx**

กำหนดให้ NGINX ทำงานทุกครั้งที่มีการ Reboot เครื่อง

[root@server1 ~]# **systemctl enable nginx**

ทำการเปิดใช้งาน NGINX

[root@server1 ~]# **systemctl start nginx**

การติดตั้ง NGINX

ตรวจสอบสถานะของ NGINX

[root@server1 ~]# **systemctl status nginx**

● nginx.service - The nginx HTTP and reverse proxy server
Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; vendor preset: disabled)
Active: active (running) since Sat 2016-08-13 21:30:07 ICT; 1h 14min ago
Process: 11149 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
Process: 11147 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
Process: 11145 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
Main PID: 11152 (nginx)
CGroup: /system.slice/nginx.service
├─11152 nginx: master process /usr/sbin/nginx
└─11153 nginx: worker process

Aug 13 21:30:07 reverseproxy systemd[1]: Starting The nginx HTTP and reverse proxy server...
Aug 13 21:30:07 reverseproxy nginx[11147]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Aug 13 21:30:07 reverseproxy nginx[11147]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Aug 13 21:30:07 reverseproxy systemd[1]: Failed to read PID from file /run/nginx.pid: Invalid argument
Aug 13 21:30:07 reverseproxy systemd[1]: Started The nginx HTTP and reverse proxy server.

การติดตั้ง NGINX

ตรวจสอบสถานะของ nginx

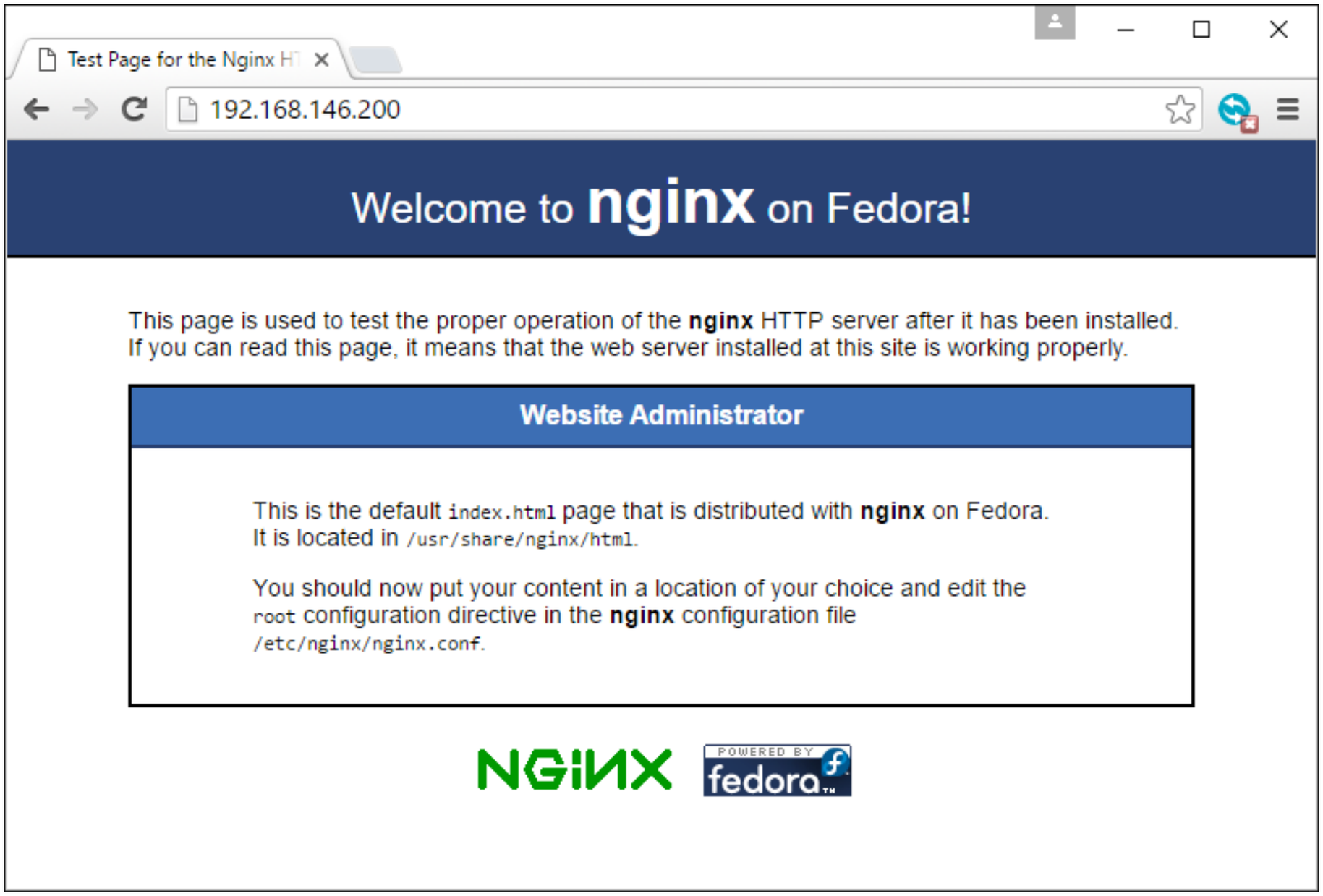
[root@server1 ~]# **netstat -antp**

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	PID/Program name
tcp	0	0	0.0.0.0:80	0.0.0.0:*	LISTEN	11152/nginx: master
tcp6	0	0	:::80	:::*	LISTEN	11152/nginx: master

การติดตั้ง NGINX

ทดสอบเข้าใช้งานผ่าน Web browser



การใช้งาน Reverse Proxy ผ่าน NGINX

ทำการสร้างไฟล์ Configuration ที่ Directory /etc/nginx/conf.d โดยชื่อไฟล์ต้องตามด้วย **.conf** ตัวอย่างเช่น **/etc/nginx/conf.d/www.mydomain.com.conf** จากนั้นเพิ่มบรรทัดต่อไปนี้

```
upstream BackendServer {
        server 192.168.146.100:80 ;
}
server {
        listen              80 ;
        server_name         www.mydomain.com ;
        location  /  {
                proxy_pass      http:// BackendServer ;
        }
}
```

Web server ที่เก็บ Content จริงๆ

กำหนด Port ที่ให้บริการบนเครื่อง Reverse Proxy

กำหนด Domain ที่สนใจ

กำหนดให้ Request ส่งไปที่ Backend server

การใช้งาน Reverse Proxy ผ่าน NGINX

เมื่อสร้างไฟล์ Configuration เสร็จแล้ว ให้ทำการทดสอบ Configuration syntax ด้วยคำสั่งต่อไปนี้

[root@server1 ~]# **nginx -t**

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok

nginx: configuration file /etc/nginx/nginx.conf test is successful

จากนั้นให้ทำการ restart NGXIN

[root@server1 ~]# **systemctl restart nginx**

ทดสอบการใช้งาน

NGINX Logging

ทำการแก้ไขไฟล์ Configuration ของ Reverse Proxy ในส่วนของ Server tag ดังตัวอย่าง

```
server {

        listen                  80 ;

        server_name             www.mydomain.com ;

        access_log              /var/log/nginx/www.mydomain.com_access.log main ;

        error_log               /var/log/nginx/www.mydomain.com_error.log ;
```

เมื่อเสร็จแล้วให้ทำการทดสอบ Configuration และทำการ Restart NGINX service

Load Balance by NGINX

ให้ทำการเพิ่ม Upstream server ที่ **Tag upstream** ดังตัวอย่าง

```
upstream  BackendServer  {

        server  192.168.146.100:80 ;

        server  192.168.146.200:80 ;

}
```

โดยรูปแบบของการกำหนดค่า Configure แบบนี้ NGINX จะทำการกระจาย Request ไปยัง Backend server แบบ **Round-robin** และจะทำการ Disable Backend server เครื่องที่ Down หรือใช้งานไม่ได้โดยอัตโนมัติ

เมื่อเสร็จแล้วให้ทำการทดสอบ Configuration และทำการ Restart NGINX service

Load Balance by NGINX

การกำหนดให้ Backend server อยู่ในสถานะ Standby

```
upstream  BackendServer  {

        server  192.168.146.100:80 ;

        server  192.168.146.200:80  backup ;

}
```

โดยรูปแบบของการกำหนดค่า Configure แบบนี้ เครื่อง Backend server ที่เป็น Backup จะไม่ถูกใช้งานจนกว่าเครื่องหลักจะ Down หรือใช้งานไม่ได้

เมื่อเสร็จแล้วให้ทำการทดสอบ Configuration และทำการ Restart NGINX service

Load Balance by NGINX

การ Disable การใช้งาน Backend server บางเครื่อง

```
upstream  BackendServer  {

        server  192.168.146.100:80 ;

        server  192.168.146.200:80  down ;

}
```

โดยรูปแบบของการกำหนดค่า Configure แบบนี้ เครื่อง Backend server ที่ถูกกำหนดเป็น **down** จะไม่ถูกใช้งานเลย

เมื่อเสร็จแล้วให้ทำการทดสอบ Configuration และทำการ Restart NGINX service

Load Balance by NGINX

การกำหนดการใช้งาน Backend server บางเครื่องให้มากกว่าเครื่องอื่น ดังตัวอย่างนี้เป็นการกำหนดในอัตราส่วนแบบ **4:1**

```
upstream   BackendServer   {

            server   192.168.146.100:80    weight=4 ;

            server   192.168.146.200:80 ;

}
```

เมื่อเสร็จแล้วให้ทำการทดสอบ Configuration และทำการ Restart NGINX service

การกำหนดให้ Request Path ไปยัง Backend ต่างเครื่อง

ตัวอย่างเช่น

- หากเรียก **www.mydomain.com** ให้ส่ง Request ไปยัง Backend server IP **192.168.146.100**
- หากเรียก **www.mydomain.com/instranet** ให้ส่ง Request ไปยัง Backend server IP **192.168.146.200**

จะสามารถกำหนดได้ดังนี้

การกำหนดให้ Request Path ไปยัง Backend ต่างเครื่อง

```
### Backend server.

upstream BackendServer  {

      server 192.168.146.100:80 ;

}

upstream BackendServer_Intranet  {

      server 192.168.146.200:80 ;

}
```

การกำหนดให้ Request Path ไปยัง Backend ต่างเครื่อง

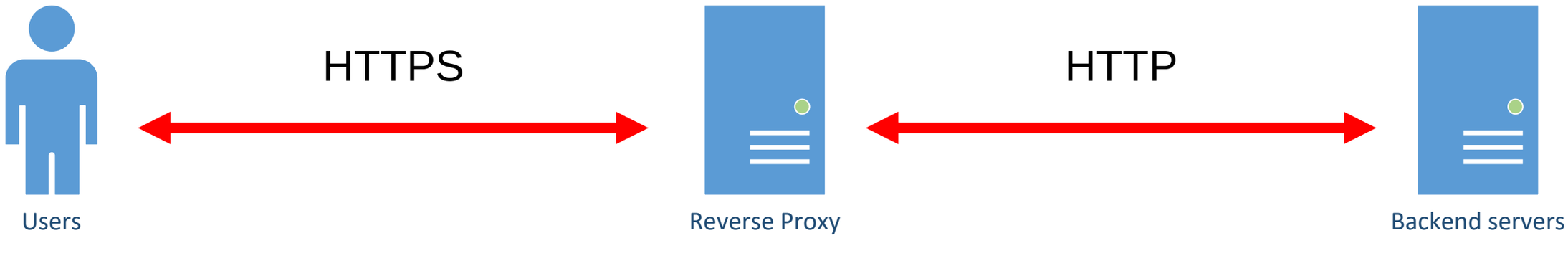
```
### Server define.

server {

        listen  80 ;

        server_name      www.mydomain.com ;

        access_log       /var/log/nginx/wwww.mydomain.com_access.log main ;

        error_log        /var/log/nginx/www.mydomain.com_error.log ;
```

การกำหนดให้ Request Path ไปยัง Backend ต่างเครื่อง

```
### Location define.

    location /intranet {

            proxy_pass      http://BackendServer_Intranet ;

     }

        location / {

            proxy_pass      http://BackendServer ;

        }

}
```

HTTPS Offloading

แนวคิด



HTTPS Offloading

ทำการสร้างไฟล์ SSL configure ที่ **/etc/nginx/conf.d/ssl.conf**

```
ssl                     on ;

ssl_certificate         /etc/nginx/ssl/server.pem ;

ssl_certificate_key     /etc/nginx/ssl/server.pem ;
```

HTTPS Offloading

แก้ไขไฟล์ **/etc/nginx/conf.d/www.mydomain.com.conf** ในส่วนของ **Server tag**

```
server {
        listen   443 ssl ;
        server_name     www.mydomain.com ;
        include         /etc/nginx/conf.d/ssl.conf ;
        location / {
                proxy_pass      http://BackendServer ;
        }
}
```

Redirect Users to HTTPS

เมื่อทำการ Configure ให้ใช้งาน HTTPS ได้แล้ว ให้ทำการเพิ่ม Configure ต่อไปนี้เข้าไปในไฟล์ (**/etc/nginx/conf.d/www.mydomain.com.conf**)

```
server {

        listen              80 ;

        ssl                 off ;

        server_name         www.mydomain.com ;

        access_log          /var/log/nginx/wwww.mydomain.com_access.log main ;

        return 301          https://$server_name$request_uri ;

}
```

NGINX Caching

ทำการสร้าง RAM disk เพื่อใช้งานเป็น Caching เนื่องจาก RAM มีความเร็วในการใช้งานสูงกว่า Hard disk ดังนี้ (ในกรณีนี้ต้องการ Ramdisk ขนาด 128MB)

ทำการสร้าง Directory เพื่อใช้เป็น Ramdisk

[root@server1 ~]# **mkdir /ramdisk**

จากนั้นทำการแก้ไขไฟล์ /etc/fstab โดยเพิ่มบรรทัดต่อไปนี้เข้าไป

```
tmpfs   /ramdisk     tmpfs   defaults,size=128M   0 0
```

NGINX Caching

จากนั้นทำการสั่งให้ทำการ Remount partitions

[root@server1 ~]# **mount -a**

ตรวจสอบผลของการสร้าง Ramdisk

[root@server1 ramdisk]# **df -h /ramdisk**

Filesystem	Size	Used	Avail	Use%	Mounted on
tmpfs	128M	0	128M	0%	/ramdisk

NGINX Caching

ทำการแก้ไขไฟล์ Reverse Proxy **/etc/nginx/conf.d/www.mydomain.com.conf**

```
upstream BackendServer  {

        server 192.168.146.100:80 ;

}
```

```
proxy_cache_path  /ramdisk/cache_www.mydomain.com  levels=1:2
keys_zone=cache_www.mydomain.com:8m  max_size=32m  inactive=10m
use_temp_path=off ;
```

NGINX Caching

จากนั้นทำการแก้แขภายใต้ **Tag location** ดังตัวอย่าง

```
server {

        listen   443 ssl ;

        server_name      www.mydomain.com ;

        include          /etc/nginx/conf.d/ssl.conf ;

location / {

        proxy_pass       http://BackendServer ;

        proxy_cache      cache_www.mydomain.com ;

        proxy_cache_valid       200 302       5m;

        proxy_cache_valid       404           1m;

}
```

NGINX Caching

วิธี Clear cache ของ NGINX ทำได้โดยการลบข้อมูลใน RAM disk ได้เลย **ยกเว้น path temp ที่อยู่ใน proxy_cache_path** เช่น

[root@server1 ~]# **ls -l /ramdisk/cache_www.mydomain.com/**

drwx------ 3 nginx nginx 60 Aug 14 07:55 5

drwx------ 3 nginx nginx 60 Aug 14 07:54 d

Cache directory : ลบได้

drwx------ 4 nginx root 80 Aug 14 07:55 temp

temp directory : ห้ามลบ

[root@server1 ~]# **rm -rf /ramdisk/cache_www.mydomain.com/5**

[root@server1 ~]# **rm -rf /ramdisk/cache_www.mydomain.com/d**

***** หากเผลอลบ temp ไป หรือเผลอลบหมด ให้แก้ไขโดยการ Restart service ของ NGINX**

การปรับแต่งค่า Configurations อื่นๆ

การปิดค่า Server Tokens สามารถทำได้โดยการแก้ไขไฟล์ **/etc/nginx/nginx.conf**

แล้วแก้ไขภายใต้ Tag http โดยเพิ่ม Configure ต่อไปนี้เข้าไป ดังตัวอย่าง

```
http {

    server_tokens  off ;
```

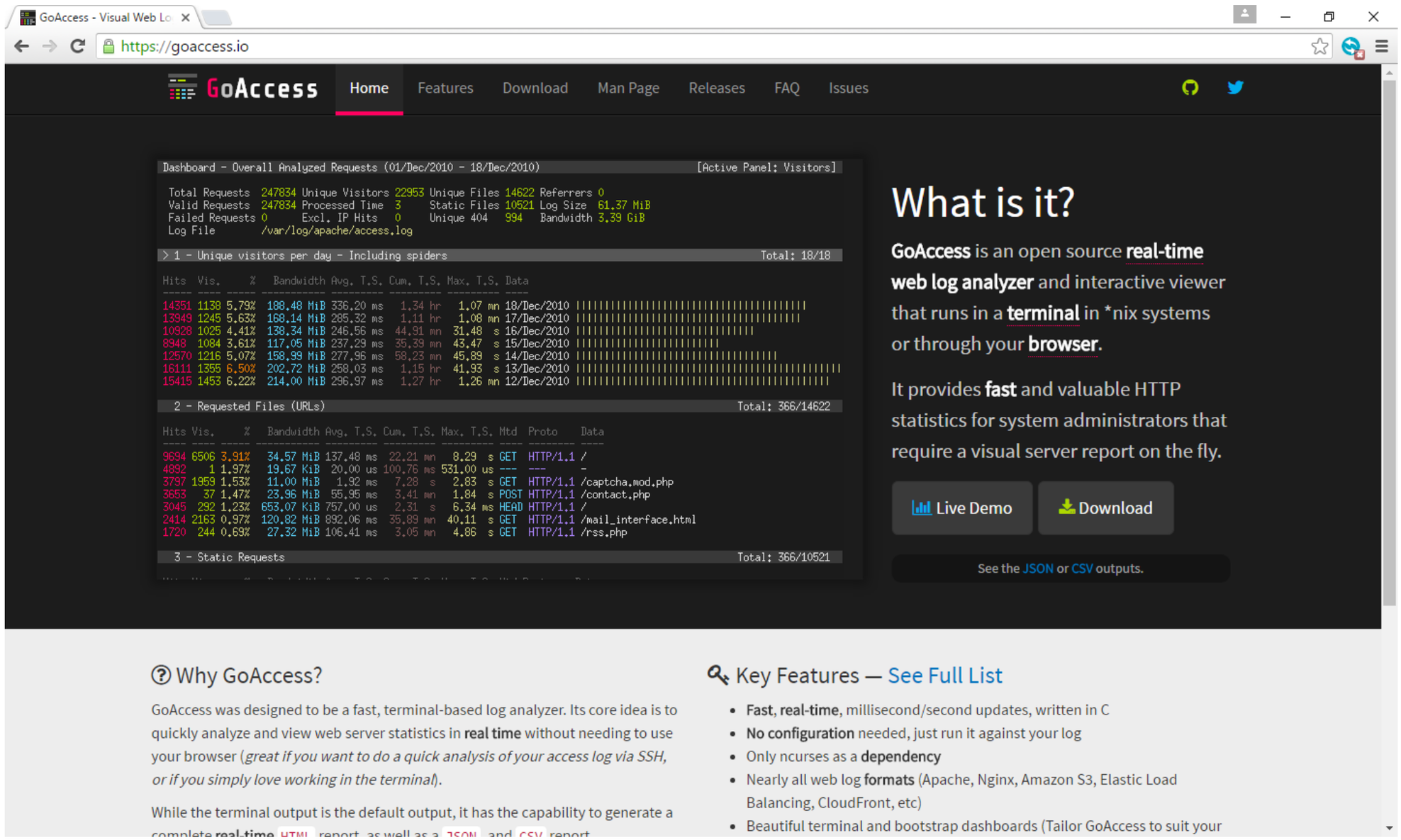
การปรับแต่งค่า Configurations อื่นๆ

การแก้ไข Request Header ของ NGINX โดยการแก้ไขในส่วนของ Tag location ดังตัวอย่าง

```
location / {

        proxy_set_header    X-Real-IP       $remote_addr ;

        proxy_set_header    X-Forwarded-For   $proxy_add_x_forwarded_for ;
```

แนะนำ Report สำหรับ NGINX

GoAccess (https://goaccess.io)



แนะนำ Report สำหรับ NGINX

GoAccess (https://goaccess.io)

[root@server1 ~]# **yum -y install goaccess**

จากนั้นทำการแก้ไขไฟล์ /etc/goaccess.conf โดยเอา # จากแถวต่อไปนี้

time-format %H:%M:%S

date-format %d/%b/%Y

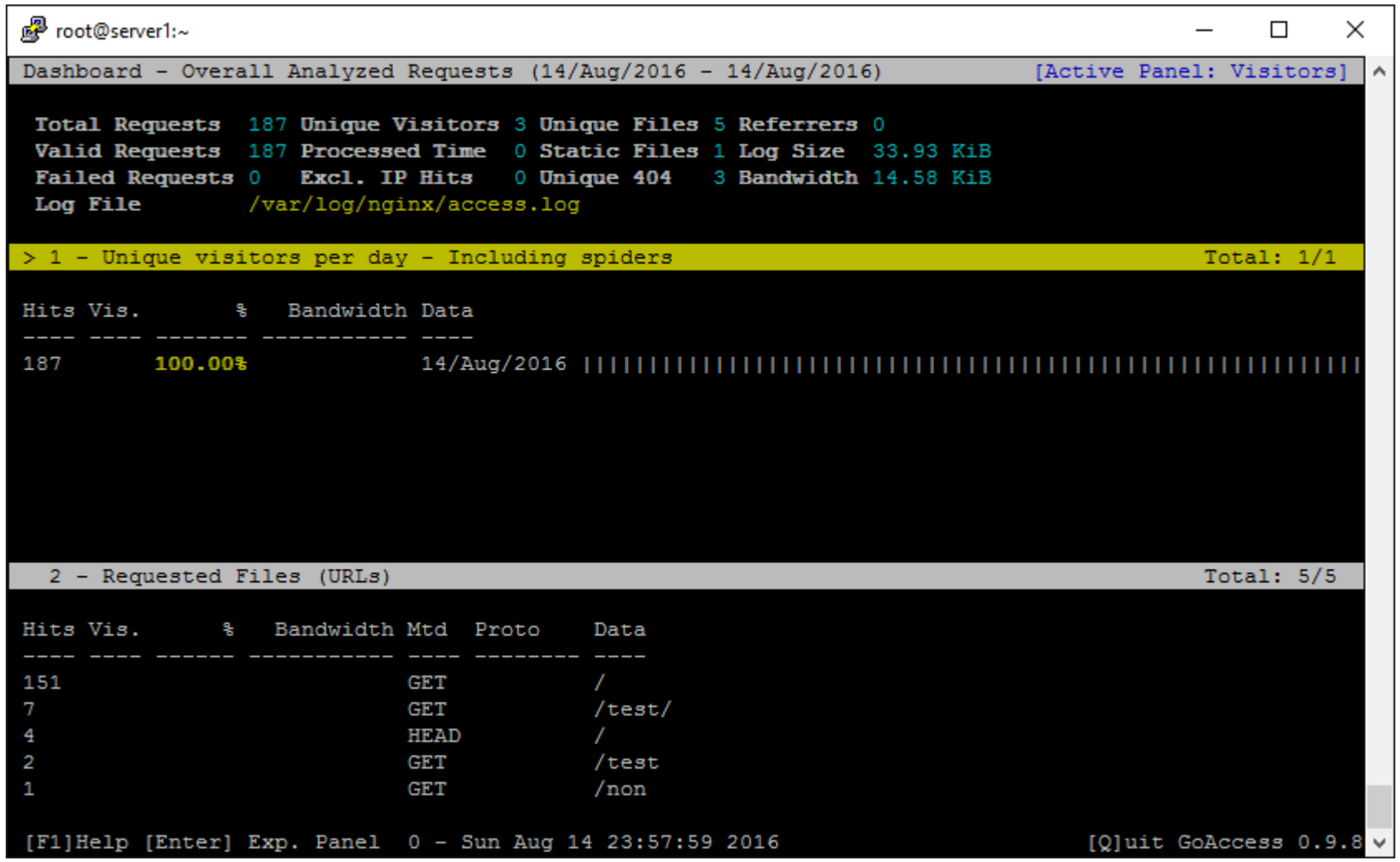
log-format %h %^[%d:%t %^] "%r" %s %b "%R" "%u"

แนะนำ Report สำหรับ NGINX

GoAccess (https://goaccess.io)

ทดสอบใช้งาน goaccess ดังตัวอย่าง

[root@server1 ~]# **goaccess -f /var/log/nginx/access.log**

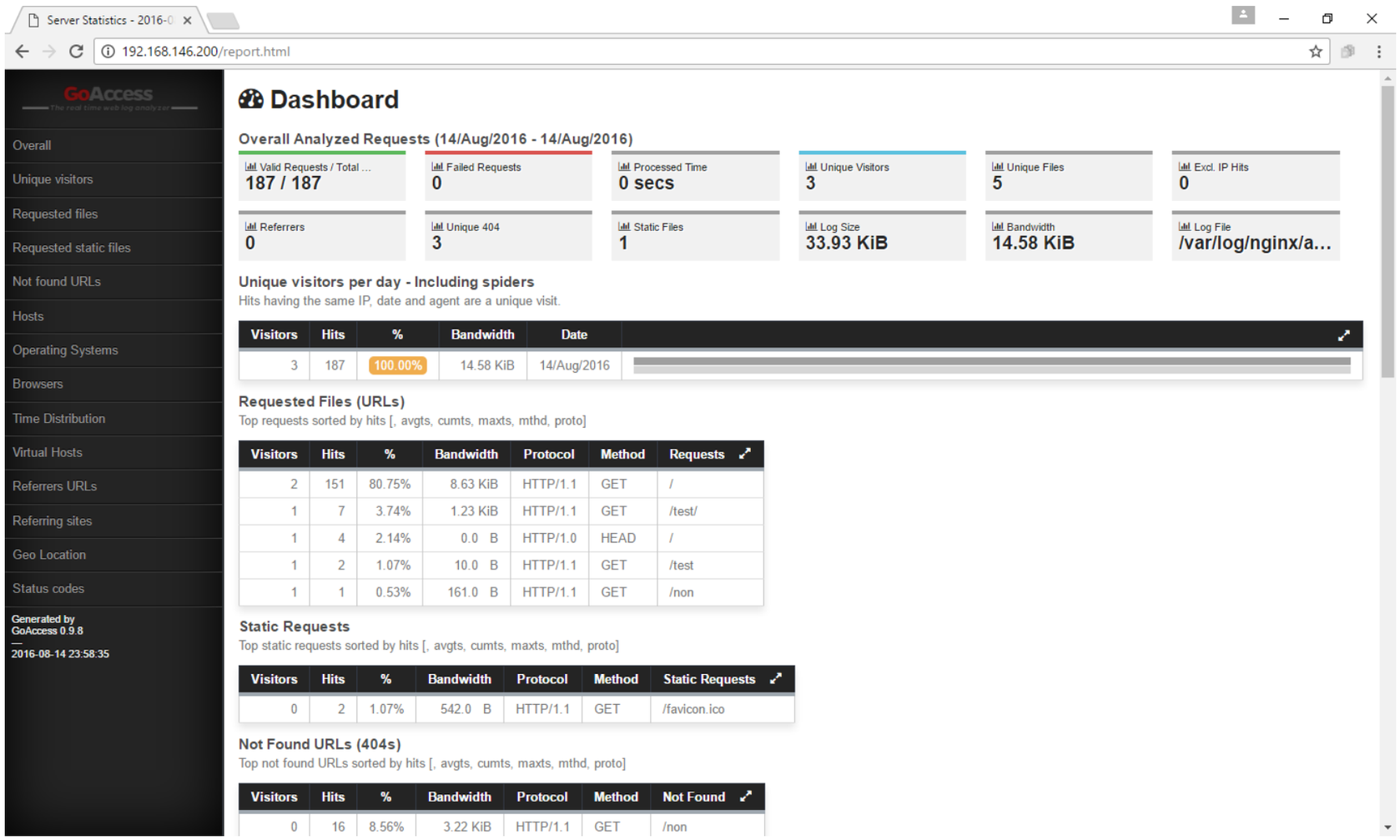


แนะนำ Report สำหรับ NGINX

GoAccess (https://goaccess.io)

ทดสอบใช้งาน goaccess ดังตัวอย่าง

[root@server1 ~]# **goaccess -f /var/log/nginx/access.log > /usr/share/nginx/html/report.html**



Questions?