
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

nginx news

Register for **nginx.conf 2016, Sept. 7-9 in Austin, TX and Save Over \$400!**
Join us and industry leaders to learn how to build & deliver applications, flawlessly.
[Register Now! Code: NG16EBB](#)

nginx news

- 2016-07-26 [nginx-1.11.3](#) mainline version has been released.
- 2016-07-05 [nginx-1.11.2](#) mainline version has been released.
- 2016-05-31 [nginx-1.10.1](#) stable and [nginx-1.11.1](#) mainline versions have been released with a fix for the [NULL pointer dereference while writing client request body](#) vulnerability (CVE-2016-4450).
- 2016-05-24 [nginx-1.11.0](#) mainline version has been released.
- 2016-04-26 [nginx-1.10.0](#) stable version has been released, incorporating new features from the 1.9.x mainline branch - including the [stream module](#), [HTTP/2](#), dynamic modules support and more.
- 2016-04-19 [nginx-1.9.15](#) mainline version has been released.
- 2016-04-05 [nginx-1.9.14](#) mainline version has been released.
- 2016-03-29 [nginx-1.9.13](#) mainline version has been released.
- 2016-02-24 [nginx-1.9.12](#) mainline version has been released.
- 2016-02-09 [nginx-1.9.11](#) mainline version has been released, with [dynamic modules](#) and TCP support in [resolver](#).
- 2016-01-26 [nginx-1.8.1](#) stable and [nginx-1.9.10](#) mainline versions have been released, with fixes for [vulnerabilities in resolver](#) (CVE-2016-0742, CVE-2016-0746, CVE-2016-0747).

NGINX

- [english](#)
- [русский](#)
- [news](#)
- [2015](#)
- [2014](#)
- [2013](#)
- [2012](#)
- [2011](#)
- [2010](#)
- [2009](#)
- [about](#)
- [download](#)
- [security](#)
- [documentation](#)
- [faq](#)
- [books](#)
- [support](#)
- [donation](#)
- [trac](#)
- [wiki](#)
- [twitter](#)
- [blog](#)

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

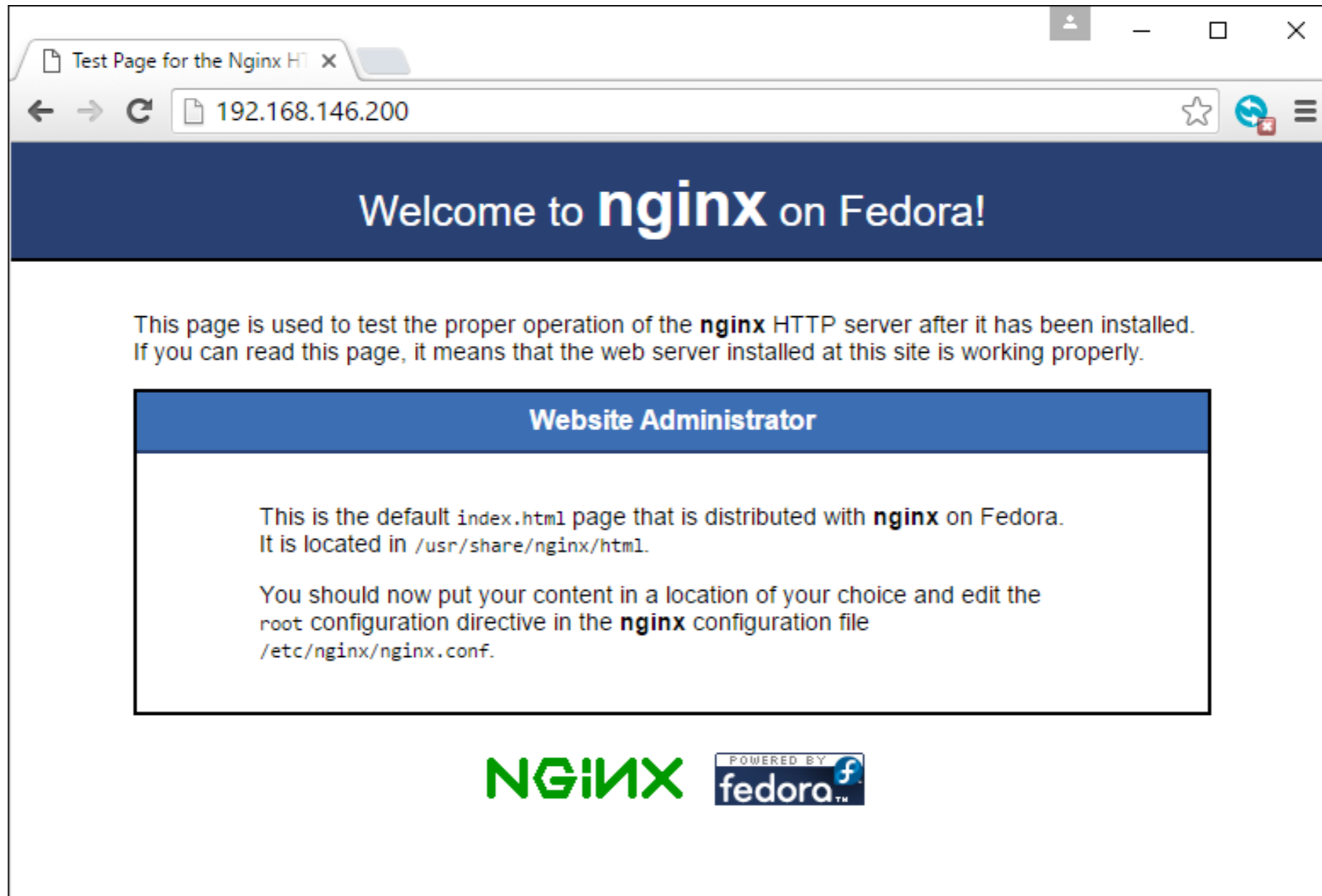
```
[root@server1 ~]# netstat -antp
```

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

ทดสอบการใช้งานผ่าน 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 NGINX

```
[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/intranet 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/www.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 ;
```

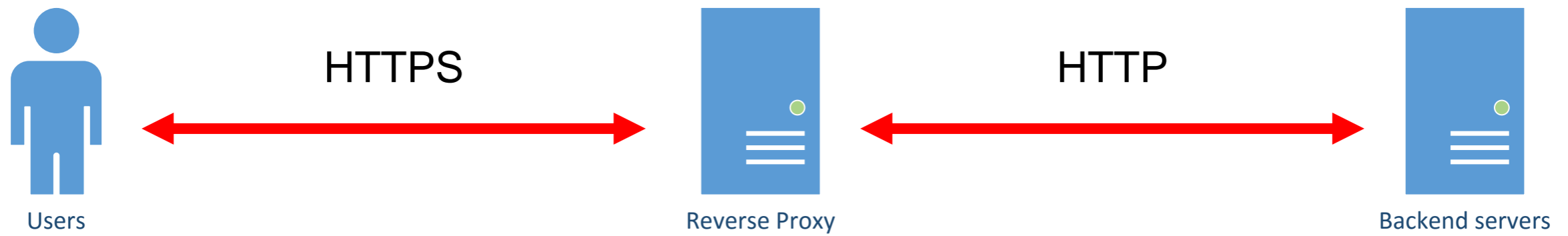
```
}
```

```
}
```



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/www.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

จากนั้นทำการแก้ไขภายใต้ 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 ดัง
ตัวอย่าง

```
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)

GoAccess - Visual Web Log Analyzer

Dashboard - Overall Analyzed Requests (01/Dec/2010 - 18/Dec/2010) [Active Panel: Visitors]

Total Requests	247834	Unique Visitors	22953	Unique Files	14622	Referrers	0
Valid Requests	247834	Processed Time	3	Static Files	10521	Log Size	61.37 MiB
Failed Requests	0	Excl. IP Hits	0	Unique 404	994	Bandwidth	3.39 GiB
Log File	/var/log/apache/access.log						

> 1 - Unique visitors per day - Including spiders Total: 18/18

Hits	Vis.	%	Bandwidth	Avg. T.S.	Cum. T.S.	Max. T.S.	Data
14351	1138	5.79%	188.48 MiB	336.20 ms	1.34 hr	1.07 mn	18/Dec/2010
13949	1245	5.63%	158.14 MiB	285.32 ms	1.11 hr	1.08 mn	17/Dec/2010
10928	1025	4.41%	138.34 MiB	246.56 ms	44.91 mn	31.48 s	16/Dec/2010
8948	1084	3.61%	117.05 MiB	237.29 ms	35.39 mn	43.47 s	15/Dec/2010
12570	1216	5.07%	158.99 MiB	277.96 ms	58.23 mn	45.89 s	14/Dec/2010
16111	1355	6.50%	202.72 MiB	258.03 ms	1.15 hr	41.93 s	13/Dec/2010
15415	1453	6.22%	214.00 MiB	296.97 ms	1.27 hr	1.26 mn	12/Dec/2010

2 - Requested Files (URLs) Total: 366/14622

Hits	Vis.	%	Bandwidth	Avg. T.S.	Cum. T.S.	Max. T.S.	Mtd	Proto	Data
9694	6506	3.91%	34.57 MiB	137.48 ms	22.21 mn	8.29 s	GET	HTTP/1.1	/
4892	1	1.97%	19.67 KiB	20.00 us	100.76 ms	531.00 us	---	---	-
3797	1959	1.53%	11.00 MiB	1.92 ms	7.28 s	2.83 s	GET	HTTP/1.1	/captcha.mod.php
3653	37	1.47%	23.96 MiB	55.95 ms	3.41 mn	1.84 s	POST	HTTP/1.1	/contact.php
3045	292	1.23%	653.07 KiB	757.00 us	2.31 s	6.34 ms	HEAD	HTTP/1.1	/
2414	2163	0.97%	120.82 MiB	892.06 ms	35.89 mn	40.11 s	GET	HTTP/1.1	/mail_interface.html
1720	244	0.69%	27.32 MiB	106.41 ms	3.05 mn	4.86 s	GET	HTTP/1.1	/rss.php

3 - Static Requests Total: 366/10521

What is it?

GoAccess is an open source **real-time web log analyzer** and interactive viewer that runs in a **terminal** in *nix systems or through your **browser**.

It provides **fast** and valuable HTTP statistics for system administrators that require a visual server report on the fly.

[Live Demo](#) [Download](#)

[See the JSON or CSV outputs.](#)

Why GoAccess?

GoAccess was designed to be a fast, terminal-based log analyzer. Its core idea is to quickly analyze and view web server statistics in **real time** without needing to use your browser (*great if you want to do a quick analysis of your access log via SSH, or if you simply love working in the terminal*).

While the terminal output is the default output, it has the capability to generate a complete **real-time HTML** report, as well as a **JSON** and **CSV** report.

Key Features — See Full List

- **Fast, real-time**, millisecond/second updates, written in C
- **No configuration** needed, just run it against your log
- Only ncurses as a **dependency**
- Nearly all web log **formats** (Apache, Nginx, Amazon S3, Elastic Load Balancing, CloudFront, etc)
- Beautiful terminal and bootstrap dashboards (Tailor GoAccess to suit your

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
```

```
root@server1:~
Dashboard - Overall Analyzed Requests (14/Aug/2016 - 14/Aug/2016) [Active Panel: Visitors] ^
Total Requests 187 Unique Visitors 3 Unique Files 5 Referrers 0
Valid Requests 187 Processed Time 0 Static Files 1 Log Size 33.93 KiB
Failed Requests 0 Excl. IP Hits 0 Unique 404 3 Bandwidth 14.58 KiB
Log File /var/log/nginx/access.log

> 1 - Unique visitors per day - Including spiders Total: 1/1

Hits Vis.      %   Bandwidth Data
-----
187      100.00%   14/Aug/2016 |

2 - Requested Files (URLs) Total: 5/5

Hits Vis.      %   Bandwidth Mtd  Proto  Data
-----
151           %           GET     /
7             %           GET     /test/
4             %           HEAD    /
2             %           GET     /test
1             %           GET     /non

[F1]Help [Enter] Exp. Panel 0 - Sun Aug 14 23:57:59 2016 [Q]uit GoAccess 0.9.8
```



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
```

The screenshot displays the GoAccess web dashboard for the date 14/Aug/2016. The dashboard provides a comprehensive overview of server activity, including overall requests, unique visitors, and detailed breakdowns of file requests and errors.

Dashboard Summary (14/Aug/2016 - 14/Aug/2016):

- Valid Requests / Total: 187 / 187
- Failed Requests: 0
- Processed Time: 0 secs
- Unique Visitors: 3
- Unique Files: 5
- Excl. IP Hits: 0
- Referrers: 0
- Unique 404: 3
- Static Files: 1
- Log Size: 33.93 KiB
- Bandwidth: 14.58 KiB
- Log File: /var/log/nginx/a...

Unique visitors per day - Including spiders:

Visitors	Hits	%	Bandwidth	Date
3	187	100.00%	14.58 KiB	14/Aug/2016

Requested Files (URLs):

Visitors	Hits	%	Bandwidth	Protocol	Method	Requests
2	151	80.75%	8.63 KiB	HTTP/1.1	GET	/
1	7	3.74%	1.23 KiB	HTTP/1.1	GET	/test/
1	4	2.14%	0.0 B	HTTP/1.0	HEAD	/
1	2	1.07%	10.0 B	HTTP/1.1	GET	/test
1	1	0.53%	161.0 B	HTTP/1.1	GET	/non

Static Requests:

Visitors	Hits	%	Bandwidth	Protocol	Method	Static Requests
0	2	1.07%	542.0 B	HTTP/1.1	GET	/favicon.ico

Not Found URLs (404s):

Visitors	Hits	%	Bandwidth	Protocol	Method	Not Found
0	16	8.56%	3.22 KiB	HTTP/1.1	GET	/non

Questions?