# About Nginx

<http://wiki.nginx.org/Main> (영문 홈페이지)

<http://nginx.org/> (러시아 개발자 홈페이지)

## - License

BSD License (<http://ko.wikipedia.org/wiki/BSD_%ED%97%88%EA%B0%80%EC%84%9C>)

<http://en.wikipedia.org/wiki/Nginx>

## - OS

Linux, BSD, Win32

## - Powered By

WordPress, Hulu, Github, Ohloh, SourceForge ...

## - Uses (Deployment)

|  |  |  |
| --- | --- | --- |
| **Uses** | **External Module** | **Module Link** |
| **Load Balancer** | - |  |
| **Reverse proxy** | File cache or Memcached | <http://code.google.com/p/memcached/wiki/NewStart> |
| **Web Server** | - |  |
| **Streaming Server (MP4)** | H264 Streaming Module | <http://h264.code-shop.com/trac> |
| **WAS** | MSSQL Driver for PHP | <http://www.freetds.org/> |
| **WAS** | PHP-FPM(PHP) | <http://php-fpm.org/wiki/Documentation> |

# Web servers and Reverse proxy cache servers Benchmark

**- Reference URL**

<http://nbonvin.wordpress.com/2011/03/24/serving-small-static-files-which-server-to-use/>

**- Test Bed**

Intel Core i3 – 370M @ 2.4 Ghz

Hard drive 5400 rpm

Memory: 4GB DDR3 1066MHz)

OS: Ubuntu 10.10 64 bit (kernel 2.6.35)

Web Object: String 100 bytes

**- S/W versions**

Nginx: 0.7.67-3ubuntu1 (64 bit)

Varnish: 2.1.3-7ubuntu0.1 (64 bit)

G-WAN: 2.1.20 (32 bit)

Lighttpd: 1.4.26-3ubuntu2 (64 bit)

Apache Traffic Server: 2.1.7-unstable (64 bit)

**- Test Result**

NginX(<http://wiki.nginx.org/>): 중상의 성능이며, 메모리 사용량이 매우 작음

G-WAN(<http://www.gwan.ch/>): 성능 가장 좋으나 동접자에 따라 CPU사용량이 지속적으로 증가.

Apache Traffic Server(<http://trafficserver.apache.org/>): 중상의 성능

Lighttpd(<http://www.lighttpd.net/>): 중상의 성능

Varnish Cache(<http://www.varnish-cache.org/>): 다른 서버보다 성능이 1/2 수준









# NginX Install

## <http://wiki.nginx.org/Install>

## <http://php-fpm.org/wiki/Documentation> (PHP-FPM 설치)

<http://www.php.net/manual/en/book.memcache.php> (PHP-Memcache Manual)

<http://h264.code-shop.com/trac/wiki/Mod-H264-Streaming-Nginx-Version2> (MP4 Streaming)

<http://www.freetds.org/userguide/php.htm> (PHP MSSQL 연동 모듈)

<http://kr.php.net/manual/en/ref.mssql.php> (PHP MSSQL용 함수)

<https://www.sulinux.net/bbs/board.php?bo_table=success_2&wr_id=182>

<http://xinet.kr/tc/i/entry/84>

<http://curl.haxx.se/download.html> (PHP CURL 모듈)

|  |  |
| --- | --- |
| 설치 경로 | 설명 |
| /usr/local/src | 설치용 파일 다운로드 경로 |
| /usr/local/nginx | nginx 설치 경로 |
| /usr/local/php | PHP-FPM 설치 경로 |
| /usr/local/freetds | FreeTDS(PHP용 MSSQL 모듈) 설치 경로 |
| /etc/init.d/nginx | Nginx 실행 스크립트 |
| /etc/init.d/php-fpm | PHP-FPM 실행 스크립트 |
| /www/nginx/~~~ | 웹루트 상위 디렉토리 |

## - Preparation

yum install pcre-devel # Perl Compatible Regular Expressions (nginx 정규식 처리에 사용)

yum install zlib-devel # Compression Library (nginx 압축 전송에 사용)

## - Install PHP-FastCGI with Memcached, MSSQL module

# CURL 모듈 설치 (PHP로 인증 처리시 이용)

wget http://curl.haxx.se/download/curl-7.21.6.tar.gz

tar zxvf curl-7.21.6.tar.gz

./configure \

--prefix=/usr/local/curl

make && make install

# FreeTDS(PHP용 MSSQL 모듈) 설치

cd /usr/local/src

wget http://ibiblio.org/pub/Linux/ALPHA/freetds/stable/freetds-stable.tgz

tar zxvf freetds-stable.tgz

cd freetds-0.82

./configure \

--prefix=/usr/local/freetds \

--enable-msdblib

make && make install

|  |
| --- |
| # MSSQL 2008버전을 사용하는 경우, 아래 처럼 FreeTDS 설정파일을 수정한다.vi /usr/local/freetds/etc/freetds.conf |
|  tds version = 8.0 |

|  |
| --- |
| # 한글이 깨질 경우, 아래 처럼 FreeTDS 설정파일에 DB접속 정보에 UTF-8을 추가한다.vi /usr/local/freetds/etc/freetds.conf |
| [DEVDB] host = 192.168.0.11 port = 10009 tds version = 8.0 client charset = UTF-8 |

# PHP 설치

cd /usr/local/src

wget http://kr.php.net/get/php-5.3.6.tar.gz/from/this/mirror

tar zxvf php-5.3.6.tar.gz

cd php-5.3.6

./buildconf --force

./configure \

--prefix=**/usr/local/php** \

--with-config-file-path=/usr/local/php/etc \

--with-mssql=/usr/local/freetds \

--with-curl=/usr/local/curl \

--enable-fpm

make && make install

cp php.ini-production /usr/local/php/etc/php.ini

cp sapi/fpm/init.d.php-fpm /etc/rc.d/init.d/php-fpm

# PHP용 memcache모듈 설치

cd /usr/local/src

wget http://pecl.php.net/get/memcache-3.0.6.tgz

tar zxvf memcache-3.0.6.tgz

cd memcache-3.0.6

/usr/local/php/bin/phpize

./configure --with-php-config=/usr/local/php/bin/php-config

make && make install

|  |
| --- |
| vi /usr/local/php/etc/php.ini |
| # 아래 내용을 추가한다.extension\_dir = "/usr/local/php/lib/php/extensions/no-debug-non-zts-20090626/"extension = "memcache.so" |

## - Add PHP-FastCGI for System Service & Run

cp /usr/local/php/etc/php-fpm.conf.default /usr/local/php/etc/php-fpm.conf

|  |
| --- |
| vi /usr/local/php/etc/php-fpm.conf |
| # 아래의 주석들을 모두 제거하고, IP와 포트를 적절히 설정한다.pid = run/php-fpm.piderror\_log = log/php-fpm.log pm.start\_servers = 20pm.min\_spare\_servers = 5 pm.max\_spare\_servers = 35 listen = **127.0.0.1:9000** |

chmod 755 /etc/init.d/php-fpm && chkconfig --add php-fpm && chkconfig php-fpm on

service php-fpm start

## - Download H264 Streaming Module (support for MP4 Streaming)

wget http://h264.code-shop.com/download/nginx\_mod\_h264\_streaming-2.2.7.tar.gz

tar zxvf nginx\_mod\_h264\_streaming-2.2.7.tar.gz

## - Patch Nginx Source for H264 Streaming Module

cd /usr/local/src

|  |
| --- |
| # 아래 처럼 패치 파일을 생성하고, Nginx 소스를 패치한다.vi nginxpatch.diff |
| --- ./nginx\_mod\_h264\_streaming-2.2.7/src/ngx\_http\_streaming\_module.c.orig 2010-05-24 18:04:43.000000000 +0400+++ ./nginx\_mod\_h264\_streaming-2.2.7/src/ngx\_http\_streaming\_module.c 2010-05-24 18:05:02.000000000 +0400@@ -155,10 +155,6 @@ } /\* TODO: Win32 \*/- if (r->zero\_in\_uri)- {- return NGX\_DECLINED;- } rc = ngx\_http\_discard\_request\_body(r); |

patch -p1 < nginxpatch.diff

## - Install Nginx

cd /usr/local/src

wget http://sysoev.ru/nginx/nginx-1.0.0.tar.gz

tar zxvf nginx-1.0.0.tar.gz

cd nginx-1.0.0

./configure \

--prefix**=/usr/local/nginx** \

--add-module**=../nginx\_mod\_h264\_streaming-2.2.7** \

--with-http\_ssl\_module \

--with-http\_flv\_module \

--with-http\_secure\_link\_module

make && make install

wget -O /etc/init.d/nginx "http://wiki.nginx.org/index.php?title=RedHatNginxInitScript&action=raw&anchor=nginx"

|  |
| --- |
| vi /etc/init.d/nginx |
| # 아래처럼 설치 경로를 수정한다.nginx**="/usr/local/nginx/sbin/nginx**"NGINX\_CONF\_FILE**="/usr/local/nginx/conf/nginx.conf**" |

chmod 755 /etc/init.d/nginx && chkconfig --add nginx && chkconfig nginx on

service nginx start

## - Edit Nginx Configuration File

See next section "Nginx Configuration" and make web root directories first.

## - Reload Nginx Configuration File without restart

service nginx reload

# Nginx Configuration

<http://wiki.nginx.org/Configuration>

## - Base Rule

## 명령문 마다 세미 콜론을 붙이며, 묶음은 중괄호를 사용.

## 초는 s, 분은 m, 시간은 h의 키워드를 숫자 뒤에 붙임.

MB는 m, GB는 g 를 숫자 뒤에 붙임.

유닉스소켓의 경우 unix:/tmp/php-fpm.sock 와 같은 형식으로 표시.

구분자는 공백 또는 탭 모두 가능.

## - Default Structure

|  |
| --- |
| 기본 구조이며, http지시자 안에 기능에 따라 적절한 server 지시자를 삽입하여 사용한다. |
| user nobody;worker\_processes 4;error\_log logs/error.log;pid logs/nginx.pid;events { worker\_connections 1024;}http { include mime.types; default\_type application/octet-stream; ########################################## # performance ########################################## sendfile on; #tcp\_nopush on; #keepalive\_timeout 0; keepalive\_timeout 65; #gzip on; ##########################################   server{ .... }} |
| Directive | Example | Description |
| user | user nobody nobody; | nginx가 root권한으로 실행될 때, 웹 계정 및 그룹 |
| worker\_processes | worker\_processes 1;worker\_processes 2; | 작업 프로세스 수. (CPU 코어 수와 맞추는게 좋을 듯.)SMP를 활용하고, disk IO에 의한 지연을 감소 시킴. |
| error\_log | error\_log logs/error.log;error\_log logs/error.log debug;error\_log logs/error.log info;error\_log /dev/null; | 에러 로그 설정.로그 레벨 종류는 debug/info/notice/warn/error/crit 임. |
| pid | pid logs/nginx.pid; | nginx 종료시 사용. |
| worker\_connections | worker\_connections 1024;worker\_connections 4096; | 최대 접속자수(max\_clients)는 아래 처럼 계산됨.max\_clients = worker\_processes \* worker\_connections(단, reverse proxy일 때에는 위 값을 4로 나눠줌.[http://wiki.nginx.org/EventsModule#worker\_connections](http://wiki.nginx.org/EventsModule%23worker_connections)) |
| include | include [file name] | 다른 설정 파일을 포함한다. |
| default\_type | default\_type application/octet-stream; | conf/mime.types에 파일 확장자별 타입 목록이 있는데,여기에 없는 확장자에 대한 타입을 설정한다. |
| sendfile | sendfile off;sendfile on; | read(), write() 대신 커널내부에서 파일복사로 속도향상.sendfile을 설정할 경우 directio 지시자는 취소됨. |
| keepalive\_timeout | keepalive\_timeout 65;keepalive\_timeout 0; | 클라이언트 접속 유지 시간몇 초 후에 클라이언트 접속을 끊을지 결정. |
| tcp\_nopush | tcp\_nopush off;tcp\_nopush on; | 소켓옵션 TCP\_CORK 사용 여부. sendfile on; 일 때에만 사용 가능HTTP Header를 한 패킷에 전송하므로서, 서버 트래픽을 감소 시킴. |
| gzip | gzip off;gzip on; | gzip 압축 사용 여부. |
| directio | directio off;directio 4m; | 운영체제의 버퍼를 사용하지 않고, 직접 IO수행. 메모리 감소 효과. 큰 파일 전송에 유용. (O\_DIRECT 플래그 이용)<http://altistory.net/333> |

## - Load Balancer & Reverse Proxy with File Cache

|  |
| --- |
| 파일로 정적 컨텐츠를 캐싱할 때- http://www.localhost.com:7942로 접속하면, 웹서버 127.0.0.1:9001와 127.0.0.1:9002로 로드밸런싱한다. - 만약 파일 캐싱을 사용하려면, proxy\_cache...로 시작되는 음영처리된 부분의 주석을 해제하면 된다.파일 캐시는 www/nginx/cache디렉토리에 저장된다. |
|  ########################################## # Load Balancer & Reverse Proxy (using file cache) ########################################## upstream backend { server **127.0.0.1:9001** weight=2; server **127.0.0.1:9002**; }# proxy\_cache\_path /www/nginx/cache levels=1:2 keys\_zone=**CACHE1**:10m# inactive=24h max\_size=1g; server { listen **7942**; server\_name **www.localhost.com**;  location / { proxy\_pass http://backend; proxy\_set\_header Host www.localhost.com;# proxy\_cache **CACHE1**;# proxy\_cache\_valid 200 302 10m;# proxy\_cache\_valid 404 1s;# proxy\_cache\_use\_stale error timeout invalid\_header updating# http\_500 http\_502 http\_503 http\_504; } } ########################################## |
| Directive | Example | Description |
| upstream | upstream [이름]{ server [아이피];[포트] weight=[숫자]; server unix:[유닉스소켓경로];} | 로드밸런싱할 서버 목록.weight옆의 숫자는 가중치이며, 정수로 표시 |
| proxy\_set\_header | proxy\_set\_header Host $host;proxy\_set\_header Host www.localhost.com;proxy\_set\_header Connection Close; | 프록시되는(backend) 서버로 전송할 때 포함시킬 HTTP HEADER. |
| proxy\_cache\_path | proxy\_cache\_path [파일캐시경로] levels=[캐시경로레벨] keys\_zone=[키존이름]:[키존용량] inactive=[유효시간] max\_size=[캐시최대용량]; | 파일캐시 설정.[키존이름]은 캐시저장소 이름.[키존용량]은 키 저장 공유메모리 사용량.[유효시간]은 요청이 없을때 삭제할 시간.[캐시최대용량]은 최대 디스크사용량. |
| proxy\_pass | proxy\_pass http://127.0.0.1:9001;proxy\_pass http://backend;proxy\_pass http://unix:/tmp/php-fpm.sock; | 프록시되는(backend) 서버 설정.IP, upstream 이름, Unix 소켓중에서 선택. |
| proxy\_cache | proxy\_cache [키존이름] | 파일캐시를 이용할 때 사용.proxy\_cache\_path 에서 설정한 키존 이름. |
| proxy\_cache\_valid | proxy\_cache\_valid 200 302 10m;proxy\_cache\_valid 404 1s; | 프록시 되는 서버(backend)로부터의 응답에 따른 캐시의 유효 기간 설정. |
| proxy\_cache\_use\_stale | proxy\_cache\_use\_stale error timeout invalid\_header updating http\_500 http\_502 http\_503 http\_504; | 어떤 경우에 과거캐시를 전송할지 여부.옵션의 종류는 proxy\_next\_upstream와 같음.예를 들어, updating 옵션을 붙인 경우 한 스레드가 캐시를 업데이트할 때, 다른 스레드는 과거캐시를 전송한다. |

## - Reverse Proxy with Memcached Server (Static Contents)

|  |
| --- |
| memcached로 정적 컨텐츠를 캐싱할 때- http://mem.localhost.com:7942?abc 형식으로 호출하면 192.168.20.49:11211의 memcached 서버에서 키값이 abc인 데이타를 조회하여 클라이언트에게 전송한다. 만약 없을 경우 웹서버 127.0.0.1:9001과 127.0.0.1:9002에서 읽는다.- 아래에서 PHP FastCGI설정은 memcached에 set을 하기 위한 설정이므로 Reverse Proxy에는 필요없다. (주의) 단, 단일 memcached 서버만 사용가능하며, 클러스터링(upstream지시자)를 사용할 수 없다. 따라서 유용성이 떨어진다.  |
|  ########################################## # Reverse Proxy with Memcached Server) for Image Files ########################################## upstream backend { server **127.0.0.1:9001** weight=2; server **127.0.0.1:9002**; } server { listen **7942**; server\_name **mem.localhost.com**; location / { set $memcached\_key $request\_uri; memcached\_pass **192.168.20.49:11211**; default\_type text/html; error\_page 404 @fallback; } location @fallback { proxy\_pass http://backend; proxy\_set\_header Host www.localhost.com; } ########################################## # PHP FastCGI ########################################## location ~ \.php$ { root /www/nginx/mem.localhost.com; charset utf8; access\_log logs/access.log; fastcgi\_pass 127.0.0.1:9000; fastcgi\_index index.php; fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name; include fastcgi\_params; } ########################################## } ########################################## |
| Directive | Example | Description |
| memcached\_pass | memcached\_pass 192.168.20.49:11211; | memcached 서버 정보 |
| $memcached\_key | set $memcached\_key $request\_uri; | memcached 서버에 요청할 때 키로 어떤 값을 전송할지 여부 $request\_uri, $uri, $args중에 하나를 설정한다. |
| $request\_uri | /foo/bar.html?a=10&b=100 | URL에서 Host이후의 전체경로. |
| $uri | /foo/bar.html | URL에서 Host이후의 파일경로만. |
| $args | a=10&b=100 | URL에서 파라미터만. |

## - Web Server

|  |
| --- |
| HTML만 처리할 때- http://**www2.localhost.com:9002** 로 접속하면, 기본적인 웹 서버로 동작한다. |
|  ########################################## # Sub Web Server #2 ########################################## server { listen **9002**; server\_name **www2.localhost.com**;  root /www/nginx/www2.localhost.com; charset utf8; access\_log logs/access.log; error\_page 404 /404.html; error\_page 500 502 503 504 /50x.html; location / { index index.html index.htm; } } |
| Directive | Example | Description |
| listen | listen 80;listen 7942; | 서버 포트 설정 |
| server\_name | server\_name localhost;server\_name abc.com;server\_name abc.com def.com hij.com; | 가상 호스트용 (HTTP의 Host헤더 값과 비교)server\_name\_in\_redirect가 on일 경우 redirect용으로 사용. |
| root | root html;root /www/nginx/html.localhost.com | 웹 디렉토리를 설정. |
| charset | charset koi8-r;charset utf8;charset off; | 응답 헤더의 "Content-Type"에 charset을 추가. |
| access\_log | access\_log logs/access.log combined;access\_log logs/access.log off;access\_log logs/access.log [format\_name] | 접속 로그 설정. |
| log\_format | log\_format combined;log\_format [format\_name] [format\_ string] | 접속 로그 포맷 설정 |
| error\_page | error\_page 404 /404.html;error\_page 500 502 503 504 /50x.html; | 오류 발생시 표시할 페이지 결정.오른쪽 예제의 경우 404.html, 50x.html 파일 이용. |
| location | location /location ^~ /images/location ~ \.php$ | URI에 대해서 정규식 매칭하여 각각 다른 설정을 한다. |
| index | index index.html index.htm; | 웹 인덱스파일 설정. |

## - PHP FastCGI & HTML

|  |
| --- |
| PHP와 HTML을 모두 처리할 때- http://www1.localhost.com:9001에 접속하면, 확장자가 PHP로 끝나는 경우에는 PHP-FPM 서버(127.0.0.1:9000)에 요청하여 PHP FastCGI를 처리하며, 그 외는 자체적으로 처리한다. |
|  ########################################## # Sub Web Server #1 & WAS ########################################## server { listen **9001**; server\_name **www1.localhost.com**; root /www/nginx/www1.localhost.com; charset utf8; access\_log logs/access.log; error\_page 404 /404.html; error\_page 500 502 503 504 /50x.html; ########################################## # for HTML ########################################## location / { index index.html index.htm; } ########################################## ########################################## # for PHP ########################################## location ~ \.php$ { fastcgi\_pass **127.0.0.1:9000**; fastcgi\_index index.php; fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name; include fastcgi\_params; } ########################################## } ########################################## |
| Directive | Example | Description |
| fastcgi\_pass | fastcgi\_pass 127.0.0.1:9000; | IP:Port 형태의 php fast cgi 서버 접속 경로 |
| fastcgi\_index | fastcgi\_index index.php; | 기본 index 파일 |
| fastcgi\_param |  | PHP-FastCGI 서버에 전송할 정보 목록.예를 들어, "SCRIPT\_FILENAME"는 파일경로를 의미.include fastcgi\_params;를 하여 "conf/ fastcgi\_params"을 인쿠르드 하면 편하다. |

## - MP4 Streaming

<http://h264.code-shop.com/trac/wiki/Mod-H264-Streaming-Nginx-Version2>

|  |
| --- |
| MP4 Streaming Module을 이용하여 MP4 파일을 스트리밍 할 때- http://mp4.localhost.com:7942/abc.mp4와 같이 접속하면 abc.mp4파일을 mp4 지시자에 의하여 Streaming으로 처리한다. 확장자가 mp4가 아닌 경우에는 웹서버처럼 처리한다.- http://mp4.localhost.com:7942/abc.mp4?start=10&end=30 와 같이 접속하면 10초에서 30초 사이의 영상만 출력된다. |
|  ########################################## # MP4 Streaming ########################################## server { listen **7942**; server\_name **mp4.localhost.com**;  root /www/nginx/mp4.localhost.com; charset utf8; access\_log logs/access.log; error\_page 404 /404.html; error\_page 500 502 503 504 /50x.html; location / { index index.html index.htm; } location ~ \.mp4$ { mp4; } } |
| Directive | Example | Description |
| mp4 | mp4; | H246 Streaming Module을 이용하여 스트리밍한다. |

## - Test

|  |
| --- |
| 테스트를 위해, 클라이언트의 hosts 파일에 아래와 같이 등록한다. IP는 Nginx가 설치된 서버로 수정한다. |
| ########################################################## Nginx TEST#########################################################222.231.18.61 www.localhost.com222.231.18.61 mem.localhost.com 222.231.18.61 www1.localhost.com222.231.18.61 www2.localhost.com222.231.18.61 mp4.localhost.com######################################################### |

|  |
| --- |
| 웹 브라우저를 실행하여, 기능별로 테스트해 보자. |
| Uses | URL | Description |
| Load Balancer & Reverse Proxy with File Cache | <http://www.localhost.com:7942/> |  |
| Reverse Proxy with Memcached | <http://mem.localhost.com:7942/index.php> |  |
| WAS (PHP FastCGI) + Memcached, MSSQL | <http://www1.localhost.com:9001/index.php> |  |
| Web Server | <http://www2.localhost.com:9002/> |  |
| MP4 Streaming | <http://mp4.localhost.com:7942/> |  |