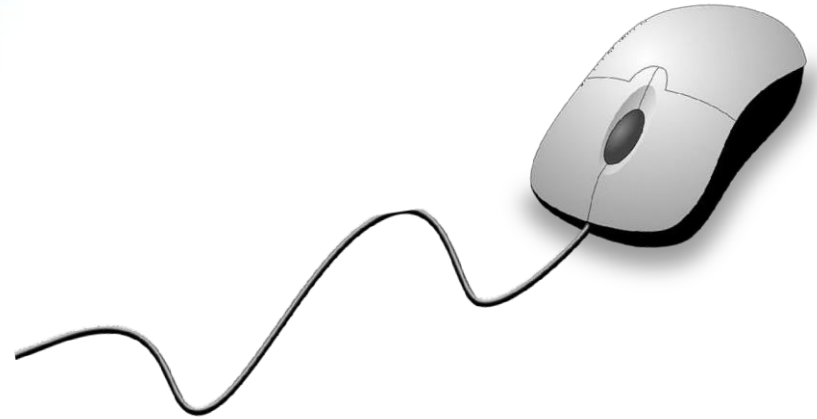


공개SW 솔루션 설치 & 활용 가이드

미들웨어 > WEB서버



# NGINX

## 제대로 배워보자

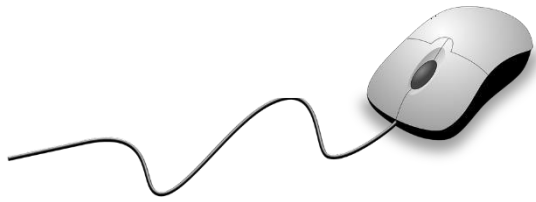
How to Use Open Source Software

---

Open Source Software Installation & Application Guide



오픈소스 소프트웨어 통합지원센터  
Open Source Software Support Center



# CONTENTS

1. 개요
2. 기능요약
3. 실행환경
4. 설치 및 실행
5. 기능소개
6. 활용예제
7. FAQ
8. 용어정리

# 1. 개요

NGINX



<b>소개</b>	<ul style="list-style-type: none"> <li>• Nginx('엔진 엑스'로 읽음)는 러시아의 소프트웨어 엔지니어인 Igor Sysoev가 만든 오픈소스 웹 서버</li> <li>• nginx는 고성능과 높은 동시처리 능력, 그리고 보다 적은 메모리 사용에 집중</li> <li>• 웹서버로의 기능뿐 아니라 로드밸런싱, 캐싱, 접근과 전송량 제어 등 최신의 웹 사이트 아키텍처 구축</li> <li>• 현재 Nginx는 인터넷에서 두번째로 인기있는 오픈소스 웹 서버</li> </ul>		
<b>주요기능</b>	<ul style="list-style-type: none"> <li>• 이벤트 기반의 오픈소스 웹 서버</li> <li>• HTTP, SMTP, POP3, IMAP protocols의 reverse proxy 서버로 사용</li> <li>• 동시성 제어, 성능 관리, low memory 사용 지원</li> </ul>		
<b>대분류</b>	• 미들웨어	<b>소분류</b>	• WEB서버
<b>라이선스형태</b>	• BSD	<b>사전설치 솔루션</b>	• 해당 없음
<b>실행 하드웨어</b>	• OS(Linux or Windows)가 설치될 수 있는 최소 사양	<b>버전</b>	• 1.13.4
<b>특징</b>	<ul style="list-style-type: none"> <li>• 더 적은 자원으로 더 빠르게 데이터를 서비스</li> <li>• 비동기 이벤트 기반의 구조로 작동</li> <li>• 코드가 쉽고 고성능의 프로그램</li> </ul>		
<b>보안취약점</b>	<ul style="list-style-type: none"> <li>• 취약점 ID : CVE-2017-7529</li> <li>• 심각도 : 7.5 HIGH(V3)</li> <li>• 취약점 설명 : 사용자가 제공 한 데이터에 대해 적절한 경계 검사를 수행하지 못하기 때문에 원격 정수 오버 플로우 취약점 발견</li> <li>• 대응방안 : 1.12.1, 1.13.3 이상으로 업데이트</li> <li>• 참고 경로 : <a href="https://www.securityfocus.com/bid/99534/info">https://www.securityfocus.com/bid/99534/info</a></li> </ul>		
<b>개발자/커뮤니티</b>	• Igor Sysoev		
<b>공식 홈페이지</b>	• <a href="https://nginx.org/">https://nginx.org/</a>		



## 2. 기능소개

NGINX



- Nginx 의 주요 기능

주요기능	지원여부
Backend-service 장애 대응 처리	서비스로 연결되어 있는 서버들의 max fails, fail timeout 의 옵션으로 해당 서버들의 상태를 체크하여 특정 서버의 장애 발생 시 백업 서버로 연결해주는 기능이다.
Load Balancing	Nginx 의 로드 밸런싱 기능을 사용하여 여러 대의 웹 서버를 연결하여 분산 처리해주는 기능이다.
Keep alive 제어	Socket에 IN/OUT Access가 마지막으로 종료된 시점부터 지정된 시간까지 Access 가 없더라도 대기하는 구조. 정의된 시간내에 Access 가 이뤄진다면 연결상태를 유지하도록 설정해주는 기능이다.
Sub-domain (Nginx) 관리	여러 도메인을 가상 호스트를 설정하여 사용 할 수 있는 기능이다.
caching 처리	image 및 기타 콘텐츠 데이터에 대해 캐싱해 주는 기능이다.



### 3. 실행환경

NGINX



- Nginx 설치 환경으로 최소, 권장사양으로 분류 할 수 있다.
- 서버 구축 시 성능에 대한 부분을 서비스 진행에 요구되는 사양에 맞춰 OS 설치가 필요하다.

카테고리	최소 사양	권장 사양
CPU	Dual Core	Quad Core
RAM	2 GB	4 GB



# 4. 설치 및 실행

NGINX



## 세부 목차

1. 바이너리 패키지를 이용한 설치
  1. 의존성 라이브러리 설치
  2. Nginx 설치 방법
  - 4.2.3. Nginx 서비스 등록 & 접속 확인
2. 소스 컴파일을 이용한 설치
  1. 의존성 라이브러리 설치
  2. Nginx 설치 파일 다운로드
  3. Nginx 컴파일 설치
  4. Nginx 서비스 실행 방법
  5. Nginx 서비스 등록
  6. Nginx 서비스 접속 확인
3. 컴파일 옵션 및 모듈 설명
  1. Nginx 컴파일 옵션 설명
  2. Nginx 기본 포함 모듈



# 4. 설치 및 실행

NGINX



## 4.1 바이너리 패키지를 이용한 설치

### 4.1.1 의존성 라이브러리 설치

- Nginx를 설치하기 전 Nginx 동작에 필요한 의존성 라이브러리 설치가 필요하다.
- 의존성 라이브러리: pcre, pcre-devel, zlib, zlib-devel, openssl, openssl-devel
- CentOS/Ubuntu를 사용하고 있다면 각 운영체제의 다음 명령을 사용하여 관련 라이브러리 패키지를 설치한다.

#### (1) CentOS

```
$ yum install pcre pcre-devel  
$ yum install zlib*  
$ yum install openssl openssl-devel
```

#### (2) Ubuntu

```
$ apt-get install libpcre3 libpcre3-dev  
$ apt-get install zlib1g zlib1g-dev  
$ apt-get install openssl libssl-dev
```



# 4. 설치 및 실행

NGINX



## 4.1 바이너리 패키지를 이용한 설치

### 4.1.2 Nginx 설치 방법

- Nginx 바이너리 패키지 설치를 위해 먼저 Nginx 다운로드를 위한 온라인 저장소를 지정한다.
- 온라인 저장소 지정은 /etc/yum.repos.d/ 디렉토리 아래와 같이 nginx.repo 파일을 만들면 된다.

```
$ vi /etc/yum.repos.d/nginx.repo  
[nginx]  
Name = nginx_repo baseurl=http://nginx.org/packages/OS/  
OSRELEASE/\$basearch  
gpgcheck=0  
enabled=1
```

- yum 명령어를 이용하여 Nginx 패키지를 설치 한다.

```
$ yum install nginx OR yum install nginx.x86_64
```





# 4. 설치 및 실행

NGINX



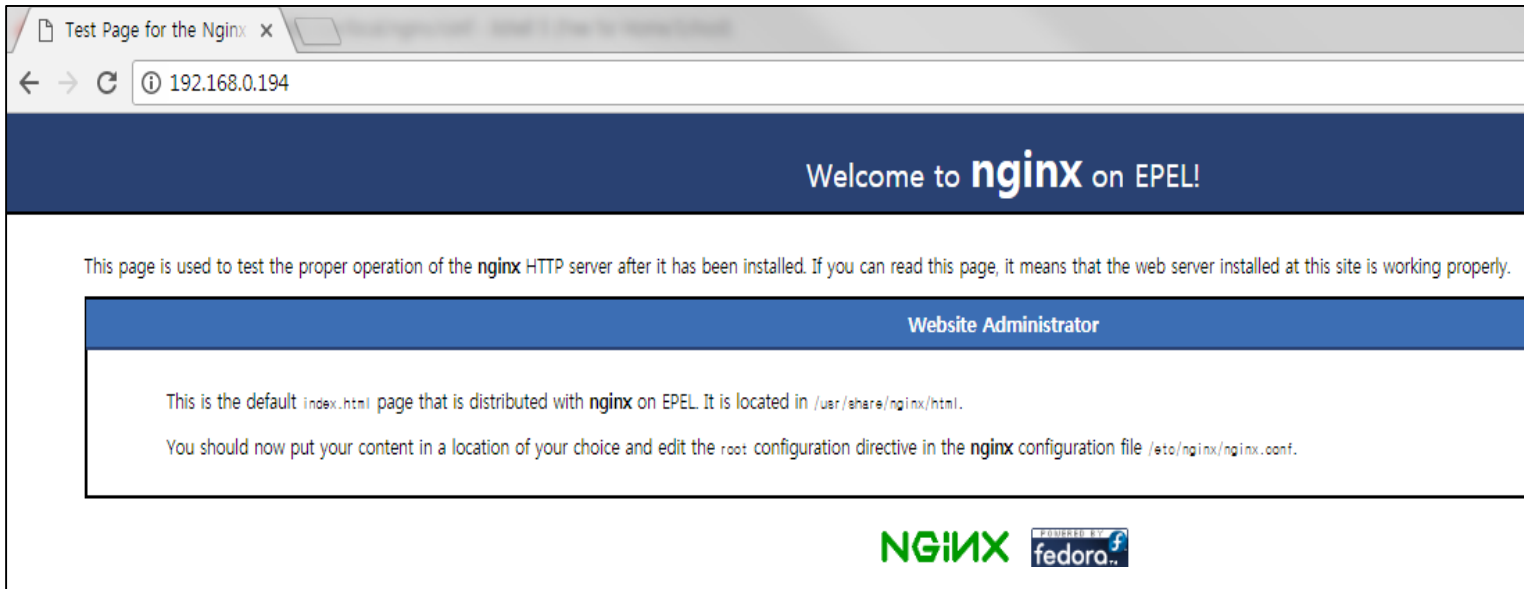
## 4.1 바이너리 패키지를 이용한 설치

### 4.1.3 Nginx 시스템서비스 등록 및 동작 확인

- 시스템 재부팅 시 실행 리스트에 포함하여 자동 시작이 되도록 설정한다.

```
$ systemctl enable nginx    #서비스 등록  
$ systemctl start nginx     #서비스 실행  
$ systemctl stop nginx      #서비스 중지
```

- Nginx 를 설치 후 서비스 실행이 정상적으로 되었는지 웹사이트 접속을 통하여 확인한다.
- 접속 URL : [http://호스트\\_IP/](http://호스트_IP/)



# 4. 설치 및 실행

NGINX



## 4.2 소스 컴파일을 이용한 설치

### 4.2.1 의존성 라이브러리 설치

- 소스 컴파일 설치의 소스파일을 가져와 운영할 서버에서 직접 컴파일하여 설치하는 방법이다.
- 바이너리 패키지 설치와 마찬가지로 Nginx 설치 전 의존성 라이브러리를 설치한다.
- 의존성 라이브러리(pcre, pcre-devel, zlib, zlib-devel, openssl, openssl-devel)는 동일 하다.

```
$ yum install pcre*  
$ yum install zlib zlib-devel  
$ yum install openssl openssl-devel
```

### 4.2.2 Nginx 소스파일 다운로드

- 공식사이트([www.nginx.org](http://www.nginx.org))에서 Nginx 소스파일을 다운로드 한다.  
(※ 2017.9.1 기준 최신 버전은 ngx-1.13.4.tar.gz 임)
- 아래와 같이 wget 명령어를 이용하여 소스파일을 다운로드 한다.
- 외부 통신이 불가능할 경우 미리 다운로드 한 파일을 FTP 업로드 또는 USB 등 외부 저장장치를 이용해 복사 한다.

```
$ wget http://nginx.org/download/nginx-1.13.4.tar.gz
```



# 4. 설치 및 실행

NGINX



## 4.2 소스 컴파일을 이용한 설치

### 4.2.3 Nginx 컴파일 설치

- 다운로드 받은 Nginx를 압축 해제하고 필요한 옵션을 사용하여 컴파일 후 설치한다.
- 본 가이드는 컴파일 설치 시 기본 Nginx 설치를 다루며 서비스 환경에 맞는 옵션을 선택하여 컴파일을 진행한다.  
(※ 기타 컴파일 옵션에 대한 설명은 문서 '15 Page'를 참조)

```
$ tar -xvzf nginx-1.13.4.tar.gz
$ cd nginx-1.13.4
$ ./configure --prefix=/usr/local/nginx --user=daemon --group=daemon
$ make
$ make install
```

### 4.2.4 Nginx 서비스 실행 방법

- 서비스 시작, 재시작, 종료하는 명령어는 각각 아래와 같다.
- 서비스 자동실행 등록은 '12~13' 페이지 참조

```
$ /usr/local/nginx/sbin/nginx           #서비스 시작
$ /usr/local/nginx/sbin/nginx -s reload  #서비스 재시작
$ /usr/local/nginx/sbin/nginx -s quit    #서비스 종료
```



# 4. 설치 및 실행

NGINX



## 4.2 소스 컴파일을 이용한 설치

### 4.2.5 Nginx 서비스 등록 (1/2)

- systemd에 등록을 위한 절차로 아래 내용을 확인한다.
- 아래 설정 경로는 (/lib/systemd/system/nginx.service)를 확인하여 기입한다. 해당 파일이 초기 설정 시 존재하지 않으면 편집기를 이용하여 작성한다.
- [Service] 해당 문구에 기록 시 Nginx 를 설치 한 경로와 동일하게 수정하여야 한다.

```
$ vi /lib/systemd/system/nginx.service
[Unit]
Description=The NGINX HTTP and reverse proxy server After=syslog
.target network.target remote-fs.target nss-lookup.target

[Service] Typ
e=forking
PIDFile=/usr/local/nginx/logs/nginx.pid
ExecStartPre=/usr/local/nginx/sbin/nginx -t
ExecStart=/usr/local/nginx/sbin/nginx ExecSt
Reload=/bin/kill -s HUP $MAINPID ExecSt
op=/bin/kill -s QUIT $MAINPID PrivateTm
p=true

[Install]
WantedBy=multi-user.target
```



# 4. 설치 및 실행

NGINX



## 4.2 소스 컴파일을 이용한 설치

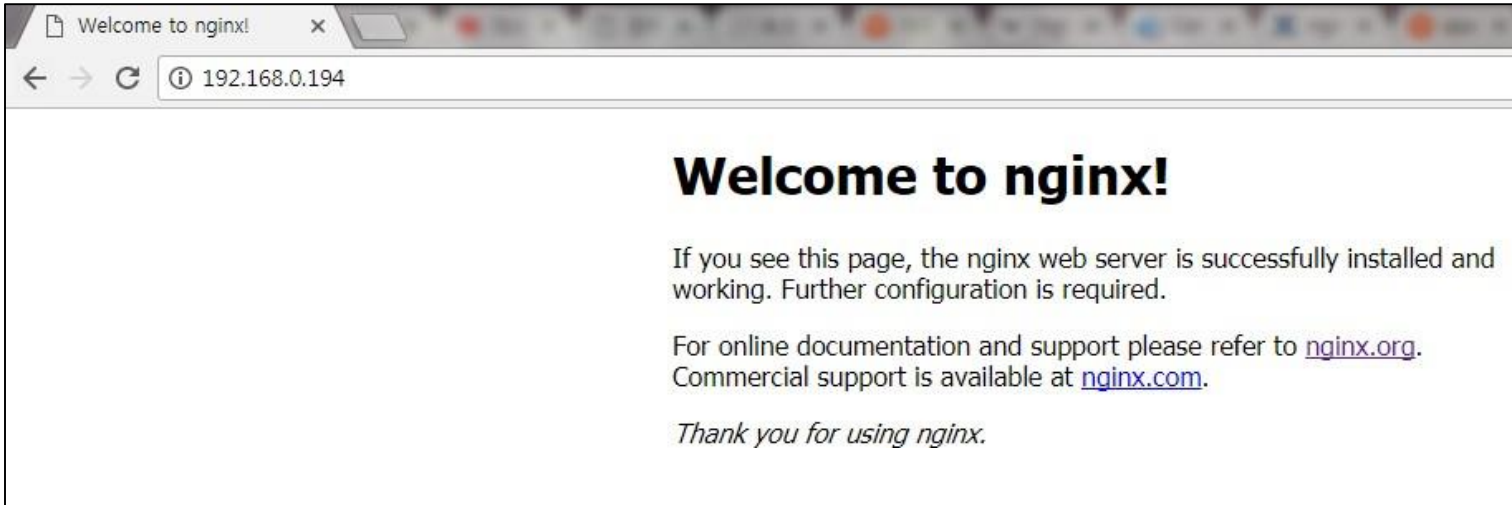
### 4.2.5 Nginx 서비스 등록 (2/2)

- 아래 systemctl 명령어를 사용하여 systemd에 서비스를 등록한다.
- systemd 에 등록 후 start, stop 옵션을 사용하여 Nginx 데몬을 실행 및 중지 할 수 있다.

```
$ systemctl enable nginx.service      #서비스 등록  
$ systemctl start nginx.service       #서비스 실행  
$ systemctl stop nginx.service        #서비스 중지
```

### 4.2.6. Nginx 서비스 접속 확인

- Nginx 데몬을 실행하여 정상적으로 페이지가 열리는지 확인한다.
- 접속 URL : [http://호스트\\_IP/](http://호스트_IP/)



# 4. 설치 및 실행

NGINX



## 4.3 컴파일 옵션 및 모듈 설명

### 4.3.1 Nginx 컴파일 옵션 (1/2)

옵션(스위치)	설 명	기본 값
<code>--prefix=</code>	Nginx가 설치되는 기본 폴더	<code>/usr/local/nginx</code> (※ 다른 스위치의 설정에 사용된 모든 상대 경로는 기본 폴더 밑에 연결된다.)
<code>--sbin-path=</code>	Nginx 바이너리 파일이 설치되는 경로	<code>&lt;prefix&gt;/sbin/nginx</code>
<code>--conf-path=</code>	주 환경설정 파일의 경로	<code>&lt;prefix&gt;/conf/nginx.conf</code>
<code>--error-log-path=</code>	에러로그 파일의 위치 (에러로그 경로는 환경설정 파일 안에서 구체적으로 설정할 수도 있다. 이 경로는 환경설정 파일에서 아무런 에러로그 지시어를 사용하지 않았을 경우에만 적용된다.)	<code>&lt;prefix&gt;/logs/error.log</code>
<code>--pid-path=</code>	Nginx pid 파일의 경로 (pid 파일 경로는 환경설정 파일 안에서 지정할 수도 있지만, 그렇지 않으면 이 스위치에서 지정한 값이 사용된다.)	<code>&lt;prefix&gt;/logs/nginx.pid</code> (※ pid 파일은 프로세스 식별자를 포함하는 단순한 텍스트 파일이다. 다른 애플리케이션들이 현재 실행중인 프로그램의 pid 파일은 잘 알려진 곳에 위치한다.)
<code>--lock-path=</code>	잠금 파일의 경로 (환경설정 파일에서 지정하지 않으면 이 스위치에서 지정한 값이 사용된다.)	<code>&lt;prefix&gt;/logs/nginx.lock</code> (※ 잠금 파일은 다른 애플리케이션들이 프로그램의 실행여부를 판단하는데 사용한다. Nginx의 경우는 프로세스가 두 번 시작되지 않게 보장하는데 사용한다.)



# 4. 설치 및 실행

NGINX



## 4.3 컴파일 옵션 및 모듈 설명

### 4.3.1 Nginx 컴파일 옵션 (2/2)

옵션(스위치)	설 명	기본 값
<code>--with-perl_modules_path=</code>	펄 모듈의 경로 (펄 모듈을 추가하려면 반드시 이 스위치를 정의해야 한다.)	없음
<code>--with-perl=</code>	펄 바이너리 파일 경로 (펄 스크립트를 실행할 때 사용된다. 펄 스크립트의 실행을 허용하려면 반드시 이 경로가 설정되어야 한다.)	없음
<code>--http-log-path=</code>	접근 로그의 위치 (이 경로는 환경설정 파일에서 접근 로그 지시어를 사용하지 않았을 때만 적용된다.)	<prefix>/logs/access.log
<code>--http-client-body-temp-path=</code>	클라이언트 요청에 의한 임시파일의 저장에 사용되는 디렉토리	<prefix>/client_body_temp
<code>--http-proxy-temp-path=</code>	프록시가 사용하는 임시파일의 위치	<prefix>/proxy_temp
<code>--http-fastcgi-temp-path=</code>	HTTP FastCGI 모듈이 사용하는 임시파일의 위치	<prefix>/fastcgi_temp
<code>--builddir=</code>	애플리케이션 빌드 위치	없음



# 4. 설치 및 실행

NGINX



## 4.3 컴파일 옵션 및 모듈 설명

### 4.3.2 Nginx 기본 포함 모듈 (1/2)

모듈 이름	설명	비활성화 하는 법
HTTP-Core	포트, locations, 에러페이지 등의 웹서버 기본 설정 모듈	--without-http
Access	IP 주소에 기반한 Allow/Deny 접근 모듈	--without-http_access_module
Auth Basic	기본 HTTP 인증 모듈	-without-http_auth_basic_module
Auto Index	자동 색인(디렉토리, 파일 리스팅) 모듈	--without-http_autoindex_module
Charset	웹 페이지를 재인코딩 하는데 사용되는 문자세트 모듈	--without-http_charset_module
Empty GIF	메모리로부터 1X1 GIF 이미지를 서비스하는 Empty Gif 모듈	--without-http_empty_gif_module
Geo	IP 주소 범위에 따라 변수를 정의하는 지오 모듈	--without-http_geo_module
Gzip	Gzip 압축 모듈	--without-http_gzip_module
Headers	HTTP 응답 헤더를 설정할 수 있는 모듈	제외할 수 없음
Index	index로 사용되는 파일들을 설정하는 모듈	제외할 수 없음
Limit Requests	사용자당 요청 수를 제한하는 Limit Requests 모듈	--without-http_limit_req_module





# 4. 설치 및 실행

NGINX



## 4.3 컴파일 옵션 및 모듈 설명

### 4.3.2 Nginx 기본 포함 모듈 (2/2)

모듈 이름	설명	비활성화 하는 법
Limit Conn	정의된 지역에 따라 자원의 사용을 제한하는 Limit Zone 모듈	--without-http_limit_zone_module
Log	access log를 설정하는 모듈	제외할 수 없음
Map	map 블록을 선언할 수 있는 맵 모듈	--without-http_map_module
Memcached	메모리 캐시 데몬과 연동되는 Memcached 모듈	--without-http_memcached_module
Proxy	요청을 받아 다른 서버로 전달하는 프록시 모듈	--without-http_proxy_module
Referer	리퍼러 제어 모듈	--without-http_referer_module
Rewrite	재작성 모듈	--without-http_rewrite_module
SCGI	SCGI 프로토콜 지원 모듈	--without-http_scgi_module
Split Client	조건에 따라 클라이언트를 분할	--without-http_split_clients_module
SSI	서버 측 인클루드(SSI) 모듈	--without-http_ssi_module
Upstream	부하 균형 구조를 만드는 업스트림 모듈	--without-http_upstream_ip_hash_module



# 5. 기능소개

NGINX



## 세부 목차

1. Backend-service 장애 대응 처리
2. Load Balancing
3. Keep alive 제어
4. Sub-domain (Nginx) 관리
5. Caching 처리



# 5. 기능소개

NGINX



## 5.1 Backend-service 장애 대응처리

- Upstream Module을 사용하여 max\_fails, fail\_timeout 을 설정하여 서버의 상태를 확인한다
- 서버의 상태를 확인하여 설정 값을 초과 할 시 Backend 백업 서버로 서비스가 활성화 된다.
- max\_fails=n : n으로 지정한 횟수 만큼 연결 실패가 되면 서버가 다운된 것으로 판단한다.
- fail\_timeout=n : max\_fails 가 지정된 상태에서 n값이 설정 만큼 서버가 응답하지 않으면 서버가 다운된 것으로 판단한다.
- Server 192.168.0.188:80은 15초간 timeout 시간이 있으며 5번 실패 시 더 이상 호출 되지 않습니다.

```
upstream service {  
    ip_hash;  
    server 192.168.0.121:80  
    server 192.168.0.188:80 max_fails=5 fail_timeout=15s;  
}
```



# 5. 기능소개

NGINX



## 5.2 Load Balancing

- Cluster에 설정되는 서버 정보와 포트를 upstream 모듈을 사용하여 설정하며 첫 번째 설정되어 있는 서버가 우선적으로 처리한다.
- 자세한 설정 방법은 페이지 '6.2 Nginx Load Balancing' 를 참조 한다.

```
upstream myservice {
    ip_hash;
    server 192.168.0.121:80;
    server 192.168.0.188:80;
}

server {
    listen      80;
    server_name localhost;

    location / {
        proxy_pass http://myservice;
    }
}
```



# 5. 기능소개

NGINX



## 5.3 Keep alive 제어

- Keep-Alive 기능이란 HTTP의 기본 구조로 소켓 연결이 종료된 시점 부터 웹 서버에 설정된 Keep-Alive Timeout 까지 기존 소켓을 유지하는 기능이다.
- Keep-Alive 를 사용 및 미 사용 할 경우 서비스에 따라 소켓을 유지 할 필요가 있다면 활성화를 시키고 사용이 필요하지 않다면 비활성화 시킬 수 있다.
- Keep-Alive 옵션으로 0을 주면 비활성화 1을 주면 활성화 시킬 수 있다.  
하지만 1 이상의 값으로 생성하게 되면 활성화 옵션보다는 캐싱 할 컨넥션 수로 시스템에 맞는 값으로 설정이 필요 하다.
- keepalive\_timeout 설정으로 클라이언트와의 컨넥션을 유지하는 시간을 설정 할 수 있다.

```
upstream myservice {
    server 192.168.0.121:80;
    keepalive 1
}

Server {
    .....
    keepalive_timeout 10
    .....
}
```



# 5. 기능소개

NGINX



## 5.4 Sub-domain (Nginx) 관리

- 하나의 웹 서버에서 2개 이상의 웹 서비스를 운영하는 것을 virtual host라고 한다.
- 경로 '/etc/nginx/conf.d/default.conf' conf 파일이 없을 경우 conf 파일을 임의로 생성한다.
- test1, test2 두 개의 server 구문을 나누어 하나의 웹 서버에 1개 이상의 웹 서비스를 설정 할 수 있다.

```
server {
    listen    80;
    server_name test1.mydomain.com
    location / {
        root /var/www/test1/public_html;
    }
server {
    listen    80;
    server_name test2.mydomain.com
    location / {
        root /var/www/test2/public_html;
    }
}
```



# 5. 기능소개

NGINX



## 5.5 Caching 처리

- 캐싱이란 웹 서비스의 이미지,CSS, 자바스크립트 같은 정적인 데이터를 저장하여 웹 서비스의 응답 속도를 보다 빠르게 처리 가능하다.
- Location : 정적인 데이터로 캐시로 저장할 파일 설정한다.
- Expires : 데이터의 저장 기간을 설정 할 수 있다.  
Expires 1M; 은 한달 동안 정적 데이터를 보관하는 설정이며 1년의 경우 1y; 로 설정한다.

```
location ~* \.(?:jpg|jpeg|gif|png|ico|cur|gz|svg|svgz|mp4|ogg|ogv|webm|htc)$ {  
    expires 1M;  
    access_log off;  
    add_header Cache-Control "public";  
}
```



# 6. 활용예제

NGINX



## 세부 목차

1. Nginx Reverse Proxy
  1. Reverse Proxy 란
  2. Nginx Reverse Proxy 설정
  - 6.2.3. Nginx Reverse Proxy 동작 확인
2. Nginx Load Balancing
  1. Load Balancing 이란
  2. Nginx Load Balancing 설정
  3. Nginx Load Balancing 동작 확인





# 6. 활용예제

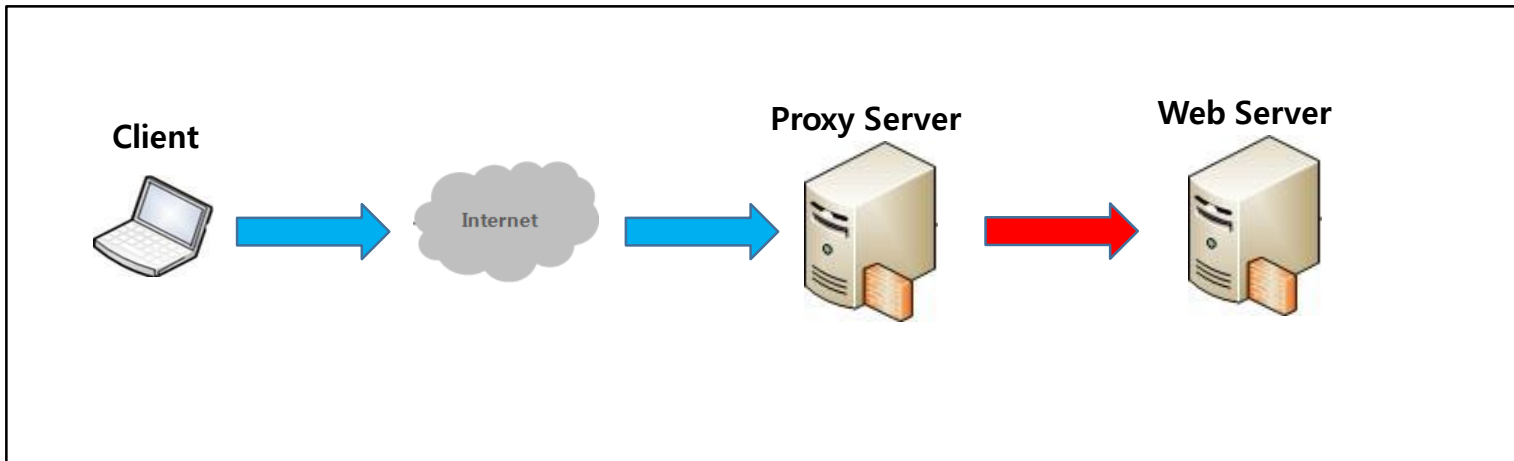
NGINX



## 6.1 Nginx Reverse Proxy

### 6.1.2 Nginx Reverse Proxy 설정

- 클라이언트가 웹 서비스에 데이터를 요청하면 Reverse Proxy는 이 요청을 받아서 웹 서버로 데이터를 전송 한다.
- 보안 : 익명의 사용자가 서버에 직접 접근하는 것을 막는다.
- ACL : 사이트 접근에 대한 정책을 구성 할 수 있다.
- 보안상의 이유로 웹 서비스의 기본 포트인 80번 외의 다른 서비스 포트를 사용하여 웹서버의 포트를 감출 수 있다.
- 웹 서버를 추가 증설하여 분산 처리 시스템을 구성 할 수 있다.



# 6. 활용예제

NGINX



## 6.1 Nginx Reverse Proxy

### 6.1.2 Nginx Reverse Proxy 설정

- 아래 설정은 Nginx의 Reverse Proxy 기본 설정 사항이다.
- Nginx 설정 파일 위치를 확인하여 nginx.conf 파일을 수정한다.
- Nginx server(192.168.0.194)에 Proxy 기능을 사용하여 클라이언트에서 요청한 데이터를 웹 서버(192.168.0.121) 서버로 포워딩한다.

```
server {  
    listen    80 default_server;  
    server_name 192.168.0.194;  
  
    proxy_redirect  p      off;  
    proxy_set_header    X-Real-IP $remote_addr;  
    proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;  
    proxy_set_header    Host $http_host;  
  
    location / {  
        proxy_pass http://192.168.0.121/; # proxy 를 사용하여 포워딩 설정  
    }  
}
```



# 6. 활용예제

NGINX



## 6.1 Nginx Reverse Proxy

### 6.1.3 Nginx Reverse Proxy 동작 확인

- Nginx의 reverse proxy 로 접근하여 포워딩 되어 호스트서버(192.168.0.121)의 Apache 서버로 접근한 것을 확인할 수 있다.



# 6. 활용예제

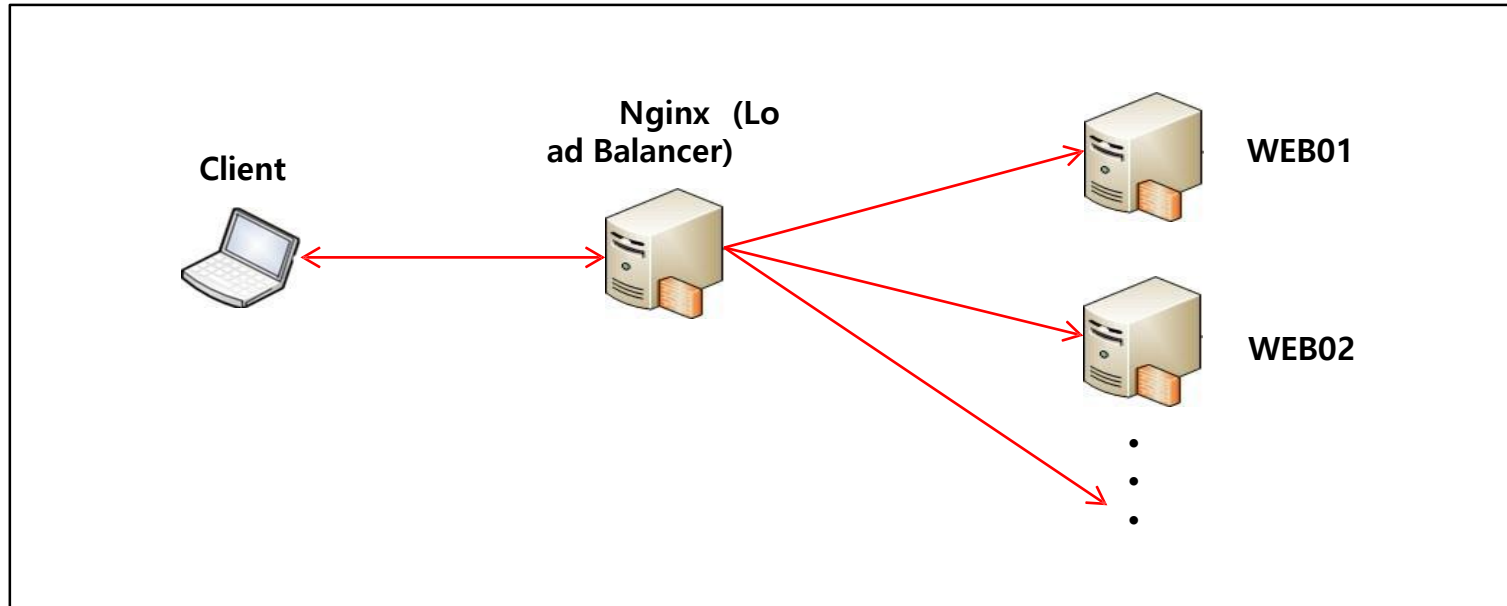
NGINX



## 6.2 Nginx Load Balancing

### 6.2.1 Load Balancing 이란

- 네트워크 트래픽 유입을 뒷단의 서버그룹(서버팜 or 서버풀)에서 효율적으로 분산처리 하는 기술로써, 앞단의 load balancer 를 통해 클라이언트 요청을 처리하여, 성능저하를 일으킬 수 있는 특정 서버의 과부하 방지, 다운된 서버를 제외한 나머지 서버로 요청전달, 추가된 새로운 서버로 요청전달 등의 기능을 수행한다.
  - ✓ 클라이언트 요청 또는 네트워크 부하를 다중서버를 통해 효과적으로 분산
  - ✓ 온라인 상태의 서버로만 요청을 보내 고가용성과 신뢰성 보장
  - ✓ 수요에 따라 서버를 추가하거나 제거할 수 있는 유연성 제공



# 6. 활용예제

NGINX



## 6.2 Nginx Load Balancing

### 6.2.2 Nginx Load Balancing 설정 (1/2)

- upstream의 그룹을 지정하여 밸런싱이 필요한 서버를 설정한다.
  - ✓ server: 밸런싱이 필요한 웹 서버 IP를 입력한다.
  - ✓ weight: 서버 가중치로써 만약 2라면 1로 설정한 서버에 비하여 2배 더 자주 접속하게 된다.
  - ✓ max\_fails: 설정한 값 n 만큼 실패가 발생하면 서버가 다운된 것으로 간주한다.
  - ✓ fail\_timeout: 해당 설정이 된 상태에서는 n 시간만큼 응답하지 않으면 서버가 다운된 것으로 간주한다.

```
$ vi /usr/local/nginx/conf/nginx.conf
upstream service {
    ip_hash;
    server 192.168.0.121:80 weight=2 max_fails=5 fail_timeout=15s;
    server 192.168.0.188:80 weight=1 max_fails=5 fail_timeout=15s;
}
```



# 6. 활용예제

NGINX



## 6.2 Nginx Load Balancing

### 6.2.2 Nginx Load Balancing 설정 (2/2)

- proxy\_set\_header 설정으로 헤더 정보를 추가한다.
  - ✓ proxy\_set\_header Host \$host: 서버에 192.168.0.194를 host로 인식한다.
  - ✓ proxy\_set\_header X-Forwarded-For \$proxy\_add\_x\_forwarded\_for: 서버에 요청한 클라이언트의 IP를 식별한다.
  - ✓ proxy\_pass <http://service>: 해당 설정은 위에 설정한 upstream의 그룹 명이다.

```
server {  
    listen 80;  
    server_name 192.168.0.194;  
  
    proxy_set_header Host $host;  
    proxy_set_header X-Real-IP $remote_addr;  
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
  
    location / {  
        proxy_pass http://service;    }  
}
```





## Q Nginx는 주로 어디에서 사용되고 있나요?

&

A 현재 Nginx를 쓰는 유명한 곳으로는 페이스북, Netflix, WordPress.com, 깃헙 (Github) 사운드 클라우드 (Soundcloud), Zynga, Sourceforge 등이 있으며 한국에서는 네이버 첫페이지, 백과사전, 엔하위키 미리, 카카오톡 공지사항 서버, XpressEngine 공식 홈페이지 등이 있다.

## Q 웹서버의 Apache와 Nginx의 차이는 무엇인가요?

&

A Apache와 Nginx는 서로 웹서버에서 제공하는 기능은 대부분 비슷하다. 하지만 동시접속자 수가 많은 경우 Nginx가 Apache 보다 성능우위에 있을 수 있다. 이유는 Apache의 경우 커넥션 당 하나의 스레드와 프로세스를 생성하지만, Nginx의 경우 Listener가 Request를 처리하기 때문에 각 Request 마다 스레드와 프로세스가 복제되지 않는다. 이 때문에 메모리적인 측면에서 Nginx는 System Resource를 적게 소비한다는 장점이 있다.



# 8. 용어정리

NGINX



용어	설명
Document Root	웹 서버에 요청이 들어오면 해당 URL을 통해서 전달된 웹 서버의 경로를 기준으로 파일을 찾게 된다. 웹 서버 상의 디렉토리를 루트 디렉토리로 정의한다.
server_name	Nginx에 접근되어지는 호스트 이름을 설정 할 수 있다.
listen	Nginx 의 서비스 포트를 지정 할 수 있다.
access_log	Nginx 웹서버로 접근하는 로그를 기록한다.
error_log	Nginx 웹서버의 설정 및 서비스 시의 오류 내용을 기록한다.
upstream	Nginx에 포함된 모듈로서 부하분산, 속도 개선에 필요한 설정을 할 수 있는 역할을 할 수 있다.
ip_hash	동일한 접속자로부터 요청된 데이터를 항상 같은 업스트림 서버가 처리 할 수 있게 한다.





# Open Source Software Installation & Application Guide



이 저작물은 크리에이티브 커먼즈 [저작자표시-비영리-동일조건 변경허락 2.0 대한민국 라이선스]에 따라 이용하실 수 있습니다.