# پیاده سازی Nginx

یکی از بهترین web server هاست.

## کانفیگ فایل

در ابتدای فایل کانفیگ که در مسیر /etc/nginx/nginx.conf قرار دارد باید موارد زیر نوشته شود:

user nginx;

worker\_processes auto;

این مورد تعداد thread هایی که همزمان اجرای این پروسه را انجام می دهند نشان می دهد بنابراین می توان یک عدد برای آن تعیین کرد و یا auto ست شود تا در صورت نیاز تعداد thread ها افزایش یابد.

error\_log /var/log/nginx/error.log;

به این وسیله تعیین می کنیم که error log ها در چه مسیری نوشته شود البته می توان در قسمت های زیر log ها را تخصصی تر نیز تعیین کرد.

pid /run/nginx.pid ;

در این مسیر شماره pid این سرویس ذخیره شده است.

events {

worker\_connections 2048;

این مورد تعیین می کند هر worker چه تعداد کانکشن را پشتیبانی کند

}

http {

در این قسمت تعیین می کنیم سرویسهایی که باید http باشند چه سرورهایی دارند

server {

در این قسمت تعیین می کنیم سروری که می خواهیم چه ویژگی هایی باید داشته باشد

Listen 80;

به این وسیله می گوییم این سرور بر روی چه پورتی به این مسیر پاسخ می دهد

location / {

در این قسمت اعلام کرده ایم که ریشه سایت (اولین صفحه سایت) که با / نمایش داده می شود چه خصوصیاتی (مثل مسیر فایلها در ماشین ما) را داشته باشد

root /home/www/;

به این ترتیب اعلام کرده ایم که اطلاعات مورد نیاز جهت نمایش در صفحه اول سایت از مسیر بالا می آید. بنابراین اگر در این قسمت یک فایل index.htm موجود باشد به صورت خود به خود بعد از وارد کردن آدرس سایت در browser‌ نمایش داده می شود. لازم به ذکر است از آنجایی که در قسمت user اعلام کرده ایم که nginx سایت را باز می کند بنابراین به تمام مسیر هایی که در قسمت location اعلام می کنیم باید owner را nginx تعیین کنیم تا دسترسی به مشکل مواجه نشود.

}

location /image/ {

root /home/img/;

با این دستور اعلام می کنیم اگر کسی به مسیر http://address\_site/image وارد شد فایلهای موجود در مسیر /home/img/image را مشاهده خواهد کرد. نکته مهم آن است که باید در این مسیر حتما یک دایرکتوری به نام image وجود داشته باشد تا محتوی آن نمایش داده شود (در صورتی که این مسیر در ادامه مسیر / بود خودبه خود احتمالا ما یک دایرکتوری image داشتیم بنابراین این مسیر منطقی به نظر می رسید)

در ضمن می توان برای هر location نیز یک error\_log قرار داد و به این ترتیب خطاهای مربوط به این مسیر در آن فضا ذخیره می شود. با این عمل آن directive که در ابتدای فایل نوشته شده است override‌ می شود (در این location )

error\_log /var/log/nginx/error\_image.log;

همین عملیات را نیز برای access\_log می توان انجام داد

access\_log /var/log/nginx/access\_image.log;

}

location /404 {

return 404;

که به این ترتیب اگر کاربری به مسیر <http://address_site/404> برود به آن کاربر کد http برابر با ۴۰۴ برگردانده می شود

}

location /301 {

return 82k.ir;

که به این ترتیب اگر کاربری به مسیر <http://address_site/301> برود آن کاربر به صورت خودکار و کامل به سایت 82k.ir منتقل خواهد شد.

}

location /302 {

return bmi.ir;

که به این ترتیب اگر کاربری به مسیر <http://address_site/302> برود به آن کاربر به صورت موقت به سایت bmi.ir منتقل شده ولی کانتکس آن در سایت اول کماکان باقی است (این مورد برای انتقال به سایت بانک برای پرداخت مناسب است)

}

}

}

در زیر کانفیگ این موضوع به طور کل تقدیم شده است.

user nginx;

worker\_processes auto;

error\_log /var/log/nginx/error.log;

pid /run/nginx.pid ;

events {

worker\_connections 2048;

}

http {

server {

Listen 80;

location / {

root /home/www/;

}

location /image/ {

root /home/img/;

error\_log /var/log/nginx/error\_image.log;

access\_log /var/log/nginx/access\_image.log;

}

location /404 {

return 404;

}

location /301 {

return 82k.ir;

}

location /302 {

return bmi.ir;

}

}

}

## تست syntax کانفیگ فایل

در صورتی کانفیگ فایل را عوض کردیم قبل از restart کردن سرویس nginx بهتر است ابتدا syntax آن را با استفاده از دستور زیر چک کنیم.

#nginx -t

## Load balancing

این امکان وجود دارد تا از nginx‌ به منظور load balancing استفاده شود برای این منظور کانفیگ فایل قسمت http را به فرم زیر تغییر می دهیم.

http {

upstream lb {

server 127.0.0.1:81 weight=1;

server 127.0.0.1:82 weight=1;

به این ترتیب اعلام کرده ایم که دو سرور داریم (که هر دو با عنوان lb دیده می شوند) و وزن load balncing روی آنها برابر است. در صورتی که مثلا وزن دومی را ۲ قرار می دادیم میزان بار دو برابر اولی برای سرور دوم ارسال میشد.

}

server {

Listen 80;

location / {

proxy\_pass <http://lb>;

با این دستور اعلام می کنیم که کسی که به روت سایت ما متصل شده است را به سمت پروکسی هایی که معرفی کرده ایم بفرستد.

}

server {

Listen 81;

location / {

root /home/www1/;

}

server {

Listen 82;

location / {

root /home/www2/;

}

در اینجا دو سرور مقصد پروکسی روی خود سرور ما قرار دارد اما در شرایط واقعی این سرورها می توانستند وب سرورهای واقعی خارج از سرور ما و با IP واقعی باشند.

در ضمن در خط اول زیر upstream می توانیم الگوریتم توزیع را مشخص کنیم که به این ترتیب کانفیگ به فرم زیر خواهد شد.

upstream lb {

least\_conn;

که این مدل باعث می شود درخواست جدید به سمت سروری برود که لود کمتری دارد.

server srv1.com

server srv2.com

server srv3.com

}

به جای least\_conn می شود از مدلهای زیر نیز استفاده کرد:

* Ip\_hash : که ارسال به سمت مقصد بر اساس هش آدرس مبدا انجام می شود

البته مدلهای زیادی وجود دارد که اینجا از آنها صرف نظر شده است.

## امنیت در nginx

در این ماژول می توان از امکانات امنیتی موجود در این وب سرور استفاده کرد. کما کان این دستورات می توانند زیر قسمت location نوشته شوند

location /test {

allow 172.16.0.0/24;

به وسیله این directive اعلام می کنیم که شبکه هایی حق دارند از این سایت استفاده کنند

deny all;

به این وسیله اعلام می کنیم که غیر از IP هایی که در قسمت بالا اعلام کرده ایم بقیه امکان دسترسی به این سایت را ندارند.

نکته بسیار مهم: آن است که ترتیب allow و deny مهم است اگر دو دستور بالا جابجا بودن دسترسی همه قطع می شد.

دستورات پایین باعث می شود که برای وارد شدن به این صفحه ما مجبور باشیم نام کاربری و کلمه عبور را وارد کنیم.

satisfy all;

این باعث می شود تا اجازه دهد از همه user/pass را بپذیرد

auth basic "test of authentication";

این متن هدر dialoge box است که به کاربر نمایش داده می شود

auth\_basic\_url\_file /etc/nginx/.htpasswd;

این آدرس فایلی است که در آن user/pass کاربرانی که حق دارند به سایت لاگین کنند وجود دارد. که برای ساخت کاربر ما از دستور htpasswd که به وسیله ماژول httpd ساخته شده است استفاده می کنیم. برای این دستور به روش زیر عمل می شود.

# htpasswd -c <file-password> *user*

}

این ماژول از htpasswd استفاده می کند که یکی از زیر مجموعه های http-tools‌ است

کانفیگ کلی به صورت زیر است

location /test {

allow 172.16.0.0/24;

deny all;

satisfy all;

auth basic "test of authentication";

auth\_basic\_url\_file /etc/nginx/.htpasswd;

}

## نتظیمات سرعت و تعداد کانکشن در nginx

این امکان وجود دارد که سرعت و تعداد کانکشن را به ازای هر location‌ محدود کرد

http {

limit\_conn\_zone $server\_name zone=nz:5m;

حتما باید این دستورات را زیر قسمت http نوشت و بعدا این ها را زیر location‌ها فراخوانی کرد. دستور بالا اعلام می کند که هر location ای که از این zone‌ (با نام nz) ارث بری داشته باشد می تواند یک فضای 5MB در RAM را اشغال کند.

limit\_conn\_zone $binary\_remote\_addr zone=addr:10m;

این دستور باعث می شود location‌ هایی که از این zone‌ استفاده می کنند در RAM‌ فضای 10MB‌ بگیرند

limit\_req\_zone $binary\_remote\_addr zone=onerps:10m rate=1r/s;

این مورد به ازای کاربرانی که به سایت متصل شده اند تعداد req‌ را محدود کرده است اولا اعلام کرده است که 10MB‌ فضا در RAM‌ بگیرد و ثانیا یک req در ثانیه پاسخ دهد.

حالا این تعریف zone‌ ها را در هر location‌ای که بخواهیم استفاده می کنیم.

location /test-zone {

limit\_rate 50k;

این دستور باعث می شود هر دانلودی از سایت با حداکثر سرعت 50Kbps انجام شود.

limit\_conn nz 1;

این directive‌ از تعریف zone ای که در قسمت http‌ انجام شده استفاده می کند و تعداد کانکشن های به این location‌ را به مقدار 5MB‌ محدود می کند. در ضمن عدد 1‌ اعلام میکند که یک واحد از این zone‌ استفاده شده است

limit\_conn addr 1;

این باعث می شود که هر کاربر با یک ip مشخص فقط بتواند یک کانکشن به این location‌ بزند.

limit\_req zone zone=onerps;

به این ترتیب تعداد req‌ کاربر را محدود به یک req در ثانیه می کنیم.

کانفیگ به صورت کلی معادل زیر است

http {

limit\_conn\_zone $server\_name zone=nz:5m;

limit\_conn\_zone $binary\_remote\_addr zone=addr:10m;

limit\_req\_zone $binary\_remote\_addr zone=onerps:10m rate=1r/s;

location /test-zone {

limit\_rate 50k;

limit\_conn nz 1;

limit\_conn addr 1;

limit\_req zone zone=onerps;

}

}