为什么选择Nginx？

Nginx 是一个高性能的 Web 和反向代理服务器, 它具有有很多非常优越的特性:

**作为 Web 服务器**：相比 Apache，Nginx 使用更少的资源，支持更多的并发连接，体现更高的效率，这点使 Nginx 尤其受到虚拟主机提供商的欢迎。能够支持高达 50,000 个并发连接数的响应，感谢 Nginx 为我们选择了 epoll and kqueue 作为开发模型.

**作为负载均衡服务器**：Nginx 既可以在内部直接支持 Rails 和 PHP，也可以支持作为 HTTP代理服务器 对外进行服务。Nginx 用 C 编写, 不论是系统资源开销还是 CPU 使用效率都比 Perlbal 要好的多。

**作为邮件代理服务器**: Nginx 同时也是一个非常优秀的邮件代理服务器（最早开发这个产品的目的之一也是作为邮件代理服务器），Last.fm 描述了成功并且美妙的使用经验。

**Nginx 安装非常的简单，配置文件 非常简洁（还能够支持perl语法），Bugs非常少的服务器**: Nginx 启动特别容易，并且几乎可以做到7\*24不间断运行，即使运行数个月也不需要重新启动。你还能够在 不间断服务的情况下进行软件版本的升级。

1、热部署

我个人觉得这个很不错。在master管理进程与worker工作进程的分离设计，使的Nginx具有热部署的功能，那么在7×24小时不间断服务的前提下，升级Nginx的可执行文件。也可以在不停止服务的情况下修改配置文件，更换日志文件等功能。

2、可以高并发连接

这是一个很重要的一个特性！在这一个 互联网 快速发展， 互联网 用户数量不断增加，一些大公司、网站都需要面对高并发请求，如果有一个能够在峰值顶住10万以上并发请求的Server，肯定会得到大家的青睐。理论上，Nginx支持的并发连接上限取决于你的内存，10万远未封顶。

3、低的内存消耗

在一般的情况下，10000个非活跃的HTTP Keep-Alive 连接在Nginx中仅消耗2.5M的内存，这也是Nginx支持高并发连接的基础。

4、处理响应请求很快

在正常的情况下，单次请求会得到更快的响应。在高峰期，Nginx可以比其他的Web服务器更快的响应请求。

5、具有很高的可靠性

Nginx是一个高可靠性的Web服务器，这也是我们为什么选择Nginx的基本条件，现在很多的网站都在使用Nginx，足以说明Nginx的可靠性。高可靠性来自其核心框架代码的优秀设计、模块设计的简单性；并且这些模块都非常的稳定。

Linux安装

Nginx 是 C语言 开发，建议在 Linux 上运行，当然，也可以安装 Windows 版本

平台：Linux Centos

安装环境

**1.gcc 安装**   
安装 nginx 需要先将官网下载的源码进行编译，编译依赖 gcc 环境，如果没有 gcc 环境，则需要安装：

yum install gcc-c++

**2. PCRE pcre-devel 安装**   
PCRE(Perl Compatible Regular Expressions) 是一个Perl库，包括 perl 兼容的正则表达式库。nginx 的 http 模块使用 pcre 来解析正则表达式，所以需要在 linux 上安装 pcre 库，pcre-devel 是使用 pcre 开发的一个二次开发库。nginx也需要此库。命令：

yum install -y pcre pcre-devel

**3.zlib 安装**   
zlib 库提供了很多种压缩和解压缩的方式， nginx 使用 zlib 对 http 包的内容进行 gzip ，所以需要在 Centos 上安装 zlib 库。

yum install -y zlib zlib-devel

**4.OpenSSL 安装**   
OpenSSL 是一个强大的安全套接字层密码库，囊括主要的密码算法、常用的密钥和证书封装管理功能及 SSL 协议，并提供丰富的应用程序供测试或其它目的使用。   
nginx 不仅支持 http 协议，还支持 https（即在ssl协议上传输http），所以需要在 Centos 安装 OpenSSL 库。

yum install -y openssl openssl-devel

官方下载

官网地址： <https://nginx.org/en/download.html>

**1.使用wget命令下载安装包（推荐）**

wget -c https://nginx.org/download/nginx-1.13.3.tar.gz

我下载的是1.13.3版本。

**2.解压命令**

tar -zxvf nginx-1.13.3.tar.gz

cd nginx-1.13.3

**3.配置**

其实在 nginx-1.13.3 版本中你就不需要去配置相关东西，默认就可以了。当然，如果你要自己配置目录也是可以的。   
3.1. 使用默认配置

./configure

3.2. 自定义配置（不推荐）

./configure \

--prefix=/usr/local/nginx \

--conf-path=/usr/local/nginx/conf/nginx.conf \

--pid-path=/usr/local/nginx/conf/nginx.pid \

--lock-path=/var/lock/nginx.lock \

--error-log-path=/var/log/nginx/error.log \

--http-log-path=/var/log/nginx/access.log \

--with-http\_gzip\_static\_module \

--http-client-body-temp-path=/var/temp/nginx/client \

--http-proxy-temp-path=/var/temp/nginx/proxy \

--http-fastcgi-temp-path=/var/temp/nginx/fastcgi \

--http-uwsgi-temp-path=/var/temp/nginx/uwsgi \

--http-scgi-temp-path=/var/temp/nginx/scgi

注：将临时文件目录指定为/var/temp/nginx，需要在/var下创建temp及nginx目录

**4.编译安装**

[root@localhost nginx-1.13.3]# make

[root@localhost nginx-1.13.3]# make install

// 查找安装路径

[root@localhost nginx-1.13.3]# whereis nginx

nginx: /usr/local/nginx

**5.启动、停止nginx**

[root@localhost nginx-1.13.3]# cd /usr/local/nginx/sbin/

// 启动 nginx

[root@localhost sbin]# ./nginx

// 停止 nginx(此方式相当于先查出nginx进程id再使用kill命令强制杀掉进程)

[root@localhost sbin]# ./nginx -s stop

// 停止 nginx(此方式停止步骤是待nginx进程处理任务完毕进行停止)

[root@localhost sbin]# ./nginx -s quit

// 从新加载配置文件(修改配置文件 nginx.conf 可以使用该命令使配置生效)

[root@localhost sbin]# ./nginx -s reload

// 查询 nginx 进程

[root@localhost sbin]# ps aux | grep nginx

当安装好之后可通过浏览器中输入 IP, 访问欢迎页, 如果其他设备需要访问, 需要配置防火墙信息。

**6.开机自启动**   
在 rc.local 增加启动代码即可。

vi /etc/rc.local

增加一行 /usr/local/nginx/sbin/nginx

设置执行权限 :

chmod 755 rc.local

[root@localhost sbin]# vi /etc/rc.local

#!/bin/bash

# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES

#

# It is highly advisable to create own systemd services or udev rules

# to run scripts during boot instead of using this file.

#

# In contrast to previous versions due to parallel execution during boot

# this script will NOT be run after all other services.

#

# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure

# that this script will be executed during boot.

touch /var/lock/subsys/local

/usr/local/nginx/sbin/nginx # 增加这一行

到这里， nginx 就安装完毕了， 当然也可以添加系统服务。

配置文件

user root;

worker\_processes auto;

events{

use epoll; #epoll是多路复用IO(I/O Multiplexing)中的一种方式,但是仅用于linux2.6以上内核,可以大大提高nginx的性能

worker\_connections 65535;

multi\_accept on;

}

http{

#媒体类型,include 只是一个在当前文件中包含另一个文件内容的指令

include mime.types;

#默认媒体类型足够

default\_type application/octet-stream;

#limit模块，可防范一定量的DDOS攻击

#用来存储session会话的状态，如下是为session分配一个名为one的10M的内存存储区，限制了每秒只接受一个ip的一次请求 1r/s

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

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

#开启高效文件传输模式，sendfile指令指定nginx是否调用sendfile函数来输出文件，对于普通应用设为 on，如果用来进行下载等应用磁盘IO重负载应用，可设置为off，以平衡磁盘与网络I/O处理速度，降低系统的负载。

sendfile on;

#必须在sendfile开启模式才有效，防止网路阻塞，积极的减少网络报文段的数量(将响应头和正文的开始部分一起发送，而不一个接一个的发送。)

tcp\_nopush on;

#设定请求缓存

server\_names\_hash\_bucket\_size 128;

client\_header\_buffer\_size 512k;

large\_client\_header\_buffers 4 512k;

client\_max\_body\_size 100m;

#FastCGI相关参数：为了改善网站性能：减少资源占用，提高访问速度

fastcgi\_connect\_timeout 300;

fastcgi\_send\_timeout 300;

fastcgi\_read\_timeout 300;

fastcgi\_buffer\_size 64k;

fastcgi\_buffers 4 64k;

fastcgi\_busy\_buffers\_size 128k;

fastcgi\_temp\_file\_write\_size 128k;

#连接超时时间，单位是秒

keepalive\_timeout 60;

#开启gzip压缩功能

gzip on；

#设置允许压缩的页面最小字节数，页面字节数从header头的Content-Length中获取。默认值是0，表示不管页面多大都进行压缩。建议设置成大于1K。如果小于1K可能会越压越大。

gzip\_min\_length 1k;

#压缩缓冲区大小。表示申请4个单位为16K的内存作为压缩结果流缓存，默认值是申请与原始数据大小相同的内存空间来存储gzip压缩结果。

gzip\_buffers 4 16k;

#压缩版本（默认1.1，前端为squid2.5时使用1.0）用于设置识别HTTP协议版本，默认是1.1，目前大部分浏览器已经支持GZIP解压，使用默认即可。

gzip\_http\_version 1.0;

#压缩比率。用来指定GZIP压缩比，1压缩比最小，处理速度最快；9压缩比最大，传输速度快，但处理最慢，也比较消耗cpu资源。

gzip\_comp\_level 9;

#用来指定压缩的类型，“text/html”类型总是会被压缩

gzip\_types text/plain application/x-javascript text/css application/xml;

#vary header支持。该选项可以让前端的缓存服务器缓存经过GZIP压缩的页面，例如用Squid缓存经过Nginx压缩的数据。

gzip\_vary off;

upstream backend\_server {

#此方法可确保始终将客户端请求传输到同一服务器。实现类似于分布式Session的效果

ip\_hash;

#设置由 fail\_timeout 定义的时间段内连接该主机的失败次数，以此来断定 fail\_timeout 定义的时间段内该主机是否可用。默认情况下这个数值设置为 1。零值的话禁用这个数量的尝试。

#设置在指定时间内连接到主机的失败次数，超过该次数该主机被认为不可用。

#这里是在30s内尝试2次失败即认为主机不可用！

server 192.168.43.22 weight=1 max\_fails=2 fail\_timeout=30s;

server 192.168.43.117 weight=1 max\_fails=2 fail\_timeout=30s;

}

server {

listen 80;

server\_name localhost;

location / {

#指定后端服务器地址和端口

proxy\_pass http://backend\_server;

#后端服务器获取用户的主机名或真实IP地址

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

}

}

}