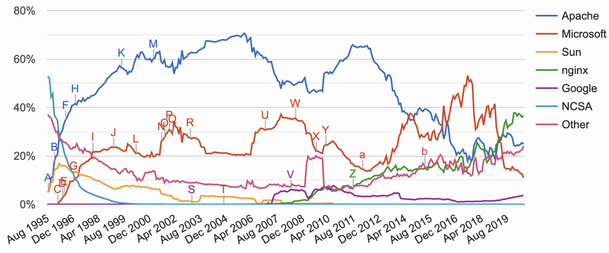
一：Nginx初步了解

- 简介：*Nginx* (engine x) 是一个高性能的HTTP和反向代理web服务器，同时也提供了IMAP/POP3/SMTP服务。Nginx是由伊戈尔·赛索耶夫为俄罗斯访问量第二的Rambler.ru站点（俄文：Рамблер）开发的，第一个公开版本0.1.0发布于2004年10月4日。现在常用来做集群的负载均衡，可配合LVS和Keepalived构造高可用服务集群
- Nginx和Tomcat及Apache的区别：Nginx和Apache都是请求转发服务器（反向代理），用来做请求的转发和静态资源的访问（如动静分离），而Tomcat是作为应用服务器，它像一个容器一样，让Java程序直接运行在其内部
- 什么叫反向代理和正向代理：
  1.正向代理：其实可以通过VPN来理解，我们无法直接访问Google，我们就去访问VPN（代理服务器），VPN将我们的请求转发到Google，返回再将页面返回给我们，这样我们就通过VPN做的一个正向代理访问到了Google。我们可以通俗的理解为正向代理的代理对象是客户端。
  2.反向代理：比如现在一台服务器无法承受用户的访问量，那么就需要水平扩容。假设增加服务器到三台，那么用户请求打过来的时候，该将请求打到哪个服务器呢？这时候就需要一个能够帮我们分发请求的服务，这个服务的作用就是反向代理（比如Nginx）。它帮我们按照一定的路由规则，比如轮询，IP_HASH,最小连接数等去把请求打到集群上，此时对用户是透明（无感知）的。我们可以通俗的理解为反向代理的代理对象是服务器端。
- Nginx的市场份额和走势:根据Netcraft最近趋势图来看，Nginx的市场占有率仍在上升并占据第一的位置



Developer	May 2020	Percent	June 2020	Percent	Change
nginx	445,724,550	36.00%	448,673,487	36.63%	0.63
Apache	315,019,262	25.45%	304,288,405	24.84%	-0.60
Microsoft	155,042,311	12.52%	134,874,928	11.01%	-1.51
Google	44,304,867	3.58%	43,449,240	3.55%	-0.03

二：编译安装Nginx（CentOS7.0）

- 在http://nginx.org/en/download.html下载tar格式的源码，目前最新版本是1.19.0，这里不太推荐用wget命令直接获取安装包，因为特别慢
- 将安装包通过ftp工具（如FlleZilla）推送到服务器，这里要注意FTP工具只是一个FTP Client，服务器上需要安装FTP Server才能成功建立连接，这里我用的是vsftpd，安装之后记得开放防火墙20和21两个用于FTP的端口
- 安装编译所需依赖

```
yum install gcc-c++ pcre pcre-devel zlib zlib-devel openssl openssl-devel
```

- 创建临时目录，也可以先不创建，等后面看到报错了再创建（手动狗头）

```
mkdir /var/temp/nginx -p
```

- 解压nginx安装包

```
tar -zxvf nginx-1.18.0.tar.gz
```

- 进入nginx目录，进行编译前的配置工作

```
./configure --prefix=/usr/local/nginx --pid-path=/var/run/nginx/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
```

各个配置的意义如下
–prefix 指定nginx安装目录
–pid-path 指向nginx的pid
–lock-path 锁定安装文件，防止被恶意篡改或误操作
–error-log 错误日志
–http-log-path http日志
–with-http_gzip_static_module 启用gzip模块，在线实时压缩输出数据流
–http-client-body-temp-path 设定客户端请求的临时目录
–http-proxy-temp-path 设定http代理临时目录
–http-fastcgi-temp-path 设定fastcgi临时目录
–http-uwsgi-temp-path 设定uwsgi临时目录
–http-scgi-temp-path 设定scgi临时目录

- 编译并安装

```
make && make install
```

- 启动，重启和停止

```
/usr/local/nginx/sbin/nginx 启动
/usr/local/nginx/sbin/nginx -s stop 停止
/usr/local/nginx/sbin/nginx -s reload 重新加载
```

- 此时浏览器直接访问ip应该可以看到Welcome Nginx的欢迎页（切记在防火墙中开放80端口，本地调试的话可以直接把防火墙stop和disable掉）

三：Nginx进程模型

3.1 进程模型


```
[root@bogon nginx]# ps -aux | grep nginx
root       6240  0.0  0.0  20568   628 ?        Ss   02:55   0:00 nginx: master process /usr/local/nginx/sbin/nginx
```

可以通过命令看到，Nginx的进程分为两部分，一种为Master，一种为Worker。在Nginx启动时，首先会建立需要监听的socket（listenfd）和Master进程，然后由Master进程根据配置文件Fork出Worker进程

- Master进程：充当整个进程组与用户的交互接口，同时对进程进行监护，管理worker进程来实现重启服务、平滑升级、更换日志文件、配置文件实时生效等功能
- Worker进程：处理客户端请求，每个Worker进程维护有一个连接池，连接池的大小决定了最大连接数，默认1024
- 惊群效应：在类似linux中nginx这种多进程模型中会有一个问题，假设我们现在有4个Work进程，由于它们都监听同一个socket的连接，所以当有请求到的时候，所有Worker进程都会被唤醒，但其实最后只有一个进程会拿到这个请求然后建立TCP连接并进行Handle，其他的进程唤醒都是没有意义并且性能损耗巨大的。就像一颗谷子扔进鸽子群一样，所有鸽子都会过来抢，但是最后只有一只鸽子能抢到，所以叫惊群效应
- 惊群效应的解决方法：在linux2.6版本中，采用等待队列解决了accept的惊群效应，每次都只从队列取一个来处理TCP链接，但这个对Nginx这种采用epllo的服务没有什么作用。于是Nginx早期采用了accept_mutex（锁）的方法，保证每次只有一个进程拿到监听句柄，但是这也导致了性能的损耗。在linux4.5之后为epllo提供了accpet类似的等待队列，nginx转而使用这种方式，可以看一下Linux与Nginx中的惊群效应和"惊群"，看看nginx是怎么解决它的这两篇写的很清晰的文章