Nginx

尚硅谷 JAVA 研究院

版本：V1.1

第 1 章 Nginx 简介

1. 是什么

Nginx("engine x") 是一个高性能的 HTTP 和反向代理服务器,特点是占有内存少，并发能力强，事实上 nginx 的并发能力确实在同类型的网页服务器中表现较好，中国大陆使用 nginx 网站用户有：百度、京东、新浪、网易、腾讯、淘宝等。

1.1 WEB 服务器

web 服务器不是 Tomcat 么？

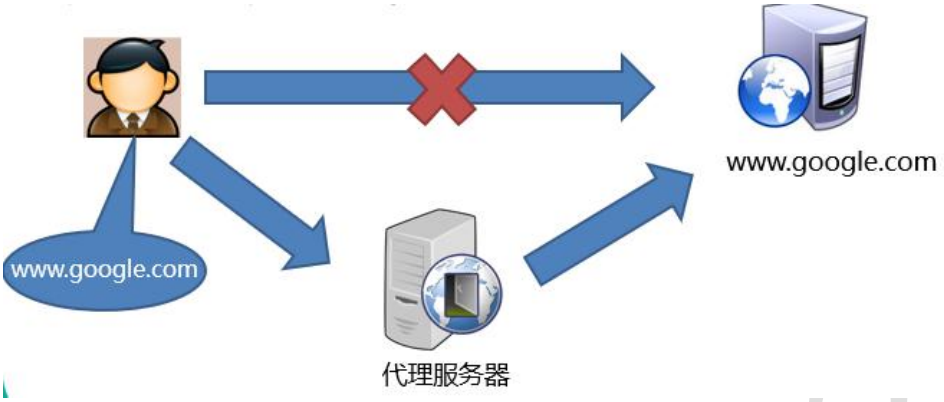
Nginx 和 Tomcat 是什么关系？

Nginx 可以作为静态页面的 web 服务器，同时还支持 CGI 协议的动态语言，比如 perl、php 等。但是不支持 java。Nginx 和 apache 是竞争对手，和 tomcat 是合作关系。

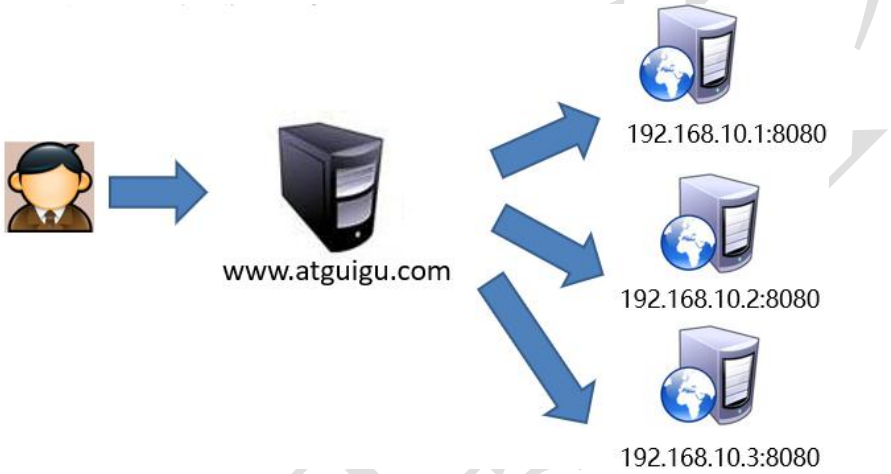
Java 程序只能通过与 tomcat 配合完成。

1.2 反向代理

（1） 正向代理，代理客户端

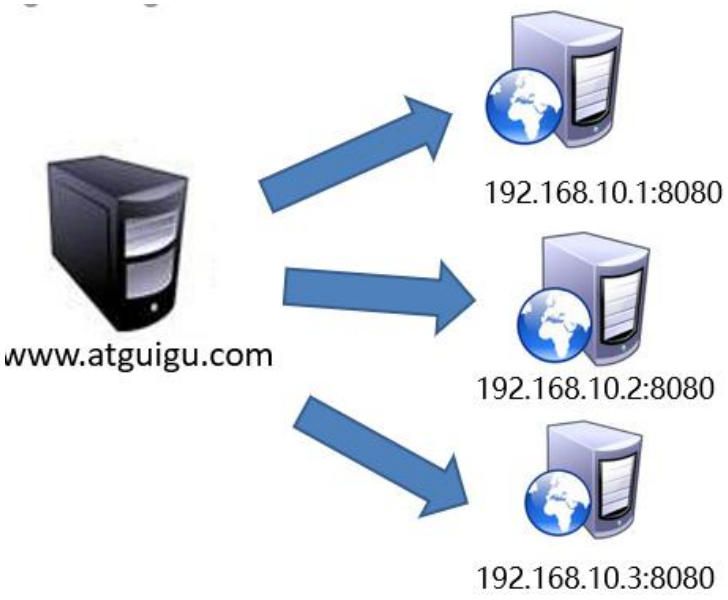


（2） 反向代理，代理服务端



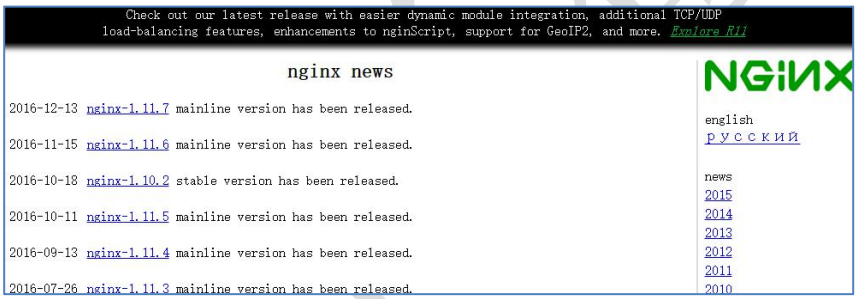
1.3 负载均衡

Nginx 的异步框架可以处理很大的并发请求，把这些并发请求 hold 住之后就可以分发给后台服务端(backend servers，也叫做服务池， 后面简称 backend)来做复杂的计算、处理和响应，这种模式的好处是相当多的：隐藏业务主机更安全，节约了公网 IP 地址，并且在业务量增加的时候可以方便地扩容后台服务器。



第 2 章 Nginx 的安装、启动

Nginx 官网：http://nginx.org/



1. 相关安装包

- pcre-8.37.tar.gz
- openssl-1.0.1t.tar.gz
- zlib-1.2.8.tar.gz
- nginx-1.11.1.tar.gz

2. 安装流程

（1） 安装 pcre

解压缩 pcre-xx.tar.gz 包

进入解压缩目录，执行./configure

如果提示，需要提前安装 gcc++，进入安装光盘目录的软件包(/media/CentOSXX/Package)执行

rpm -ivh libstdc++-devel-4.4.7-17.el6.x86_64.rpm

rpm -ivh gcc-c++-4.4.7-17.el6.x86_64.rpm

./configure 完成后，回到 pcre 目录下执行 make，再执行 make install

（2） 安装 openssl

解压缩 openssl-xx.tar.gz 包。

进入解压缩目录，执行./config

make && make install

（3） 安装 zlib

解压缩 zlib-xx.tar.gz 包。

进入解压缩目录，执行./configure。

make && make install

（4） 安装 nginx

解压缩 nginx-xx.tar.gz 包。

进入解压缩目录，执行./configure。

make && make install

3. Nginx 启动

（1） 启动问题

进入/usr/local/nginx/sbin 目录，运行命令./nginx 即可启动 nginx

nginx 无法启动: libpcre.so.1/libpcre.so.0: cannot open shared object file 解决办法

解决方法：

ln -s /usr/local/lib/libpcre.so.1 /lib64

32 位系统则：

ln -s /usr/local/lib/libpcre.so.1 /lib

（2） 启动命令

启动命令：在/usr/local/nginx/sbin 目录下执行 ./nginx

关闭命令：在/usr/local/nginx/sbin 目录下执行 ./nginx -s stop

重新加载命令：在/usr/local/nginx/sbin 目录下执行 ./nginx -s reload

（3） 设置 nginx 为自启动服务

修改 linux 启动脚本/etc/rc.d/rc

加入 :/usr/local/nginx/sbin/nginx

```
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
/usr/local/nginx/sbin/nginx
```

（4） 进入首页

第 3 章 启动结合 redis 配置负载均衡

1. 首先准备两个同时启动的 Tomcat

修改 tomcat2 的配置文件：server.xml 修改端口号，使两个 tomcat 可以同时启动

2. 拷贝对应 jar 包到 tomcat 下 lib 包中

commons-pool2-2.0.jar

jedis-2.5.2

tomcat-redis-session-manager1.2.jar

3. 修改 tomcat 的下 content.xml(加到最下方)

```
<Valve className="com.orangefunction.tomcat.redissessions.RedisSessionHandlerValve" />
  <Manager className="com.orangefunction.tomcat.redissessions.RedisSessionManager"
    host="127.0.0.1"
    port="6379"
    database="0"
    maxInactiveInterval="60" />
```

4. 启动 redis

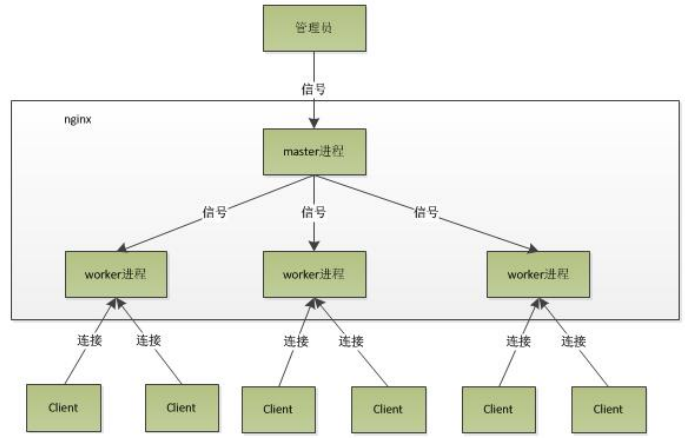
5. 配置 nginx.conf

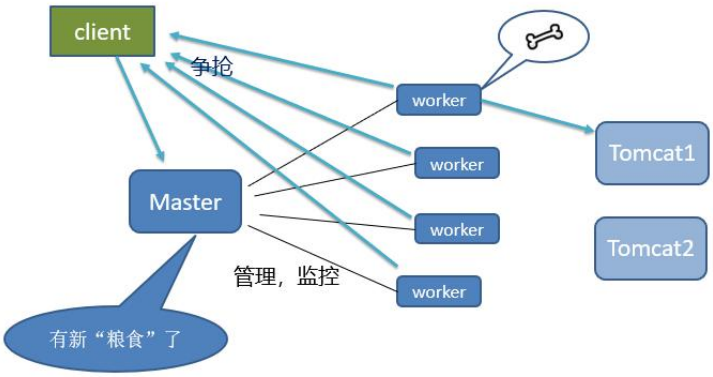
```
http {
.....
        upstream myserver{
```

```
            ip_hash;
            server 115.28.52.63:8080 weight=1;
            server 115.28.52.63:8180 weight=1;
        }
.....
        server{
            location / {
                                .........
                proxy_pass http://myserver;
                    proxy_connect_timeout 10;
            }
                        .........
        }
}
```

第 4 章 nginx 的原理与配置

1. master&worker





2. master-workers 的机制的好处

首先，对于每个 worker 进程来说，独立的进程，不需要加锁，所以省掉了锁带来的开销，同时在编程以及问题查找时，也会方便很多。

其次，采用独立的进程，可以让互相之间不会影响，一个进程退出后，其它进程还在工作，服务不会中断，master 进程则很快启动新的 worker 进程。当然，worker 进程的异常退出，肯定是程序有 bug 了，异常退出，会导致当前 worker 上的所有请求失败，不过不会影响到所有请求，所以降低了风险。

```
#设置 worker 数量。
worker_processes    4
```

```
#work 绑定 cpu(4 work 绑定 4cpu)。
worker_cpu_affinity 0001 0010 0100 1000
#work 绑定 cpu (4 work 绑定 8cpu 中的 4 个) 。
worker_cpu_affinity 0000001 00000010 00000100 00001000
```

3.需要设置多少个 worker

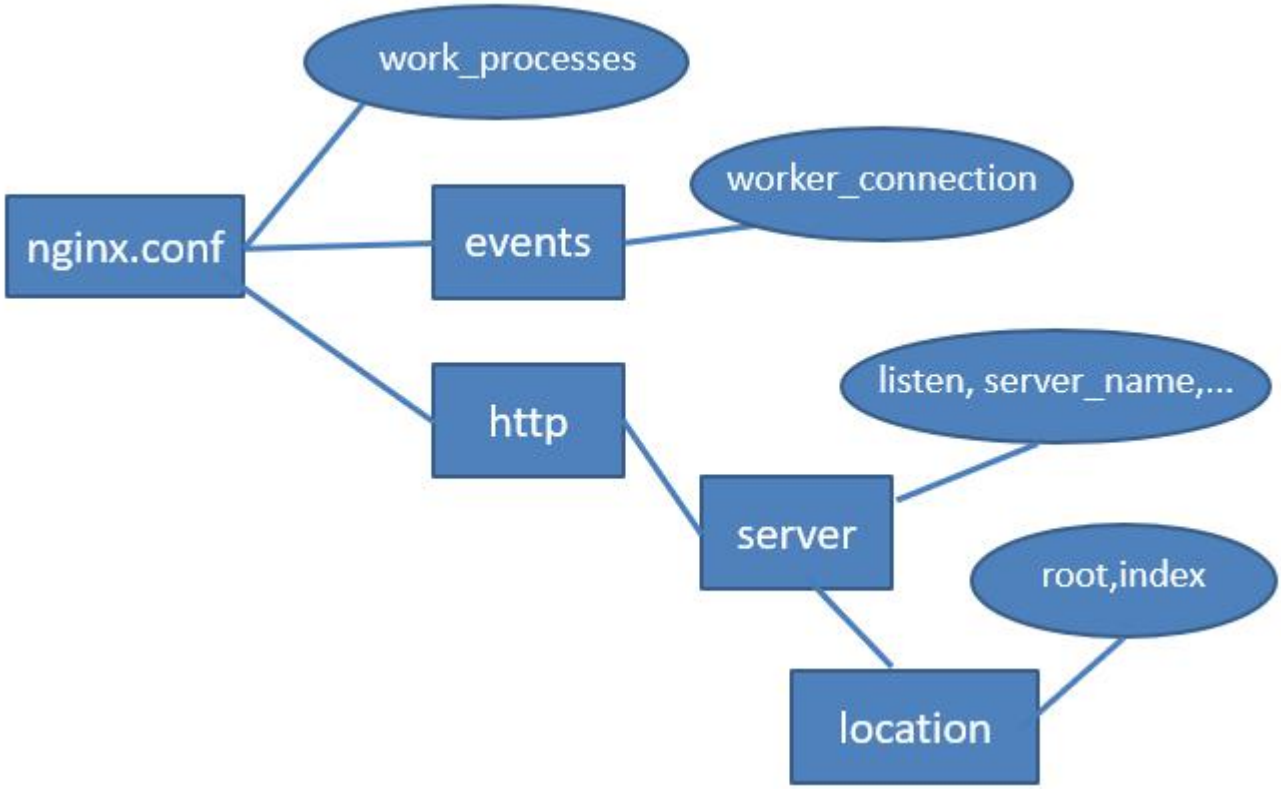
Nginx 同 redis 类似都采用了 io 多路复用机制，每个 worker 都是一个独立的进程，但每个进程里只有一个主线程，通过异步非阻塞的方式来处理请求， 即使是千上万个请求也不在话下。每个 worker 的线程可以把一个 cpu 的性能发挥到极致。

所以 worker 数和服务器的 cpu 数相等是最为适宜的。设少了会浪费 cpu，设多了会造成 cpu 频繁切换上下文带来的损耗。

4.连接数 worker_connection

这个值是表示每个 worker 进程所能建立连接的最大值，所以，一个 nginx 能建立的最大连接数，应该是 worker_connections * worker_processes。当然，这里说的是最大连接数，对于 HTTP 请求本地资源来说，能够支持的最大并发数量是 worker_connections * worker_processes，如果是支持 http1.1 的浏览器每次访问要占两个连接，所以普通的静态访问最大并发数是： worker_connections * worker_processes /2，而如果是 HTTP 作为反向代理来说，最大并发数量应该是 worker_connections * worker_processes/4。因为作为反向代理服务器，每个并发会建立与客户端的连接和与后端服务的连接，会占用两个连接。

5.nginx.conf 结构




nginx.conf