**本次课的内容：**

1、介绍nginx在系统架构中的作用

2、正向代理与反向代理

3、nginx的安装部署方式

4、nginx的运行模式概念，启停命令

5、 核心文件nginx.conf配置介绍

6、日志配置和及切割

**Nginx简介**

**Apache**

Apache仍然是时长占用量最高的web服务器，据最新数据统计，市场占有率目前是50%左右。主要优势在于一个是比较早出现的一个Http静态资源服务器，同时又是开源的。所以在技术上的支持以及市面上的各种解决方案都比较成熟。Apache支持的模块非常丰富。

**Nginx**

Nginx是俄罗斯人编写的一款高性能的HTTP和反向代理服务器，在高连接并发的情况下，它能够支持高达50000个并发连接数的响应，但是内存、CPU等系统资源消耗却很低，运行很稳定。目前Nginx在国内很多大型企业都有应用，据最新统计，Nginx的市场占有率已经到33%左右了。而Apache的市场占有率虽然仍然是最高的，但是是呈下降趋势。而Nginx的势头很明显。选择Nginx的理由也很简单：第一，它可以支持5W高并发连接；第二，内存消耗少；第三，成本低，如果采用F5、NetScaler等硬件负载均衡设备的话，需要大几十万。而Nginx是开源的，可以免费使用并且能用于商业用途

**架构中的作用**

**介绍nginx在系统架构（网关入口）中的作用，总结如下：**

1、路由功能（与微服务对应）：域名/路径，进行路由选择后台服务器

2、负载功能（与高并发高可用对应）：对后台服务器集群进行负载

3、静态服务器（比tomcat性能高很多）：在mvvm模式中，充当文件读取职责

总结：实际使用中，这三项功用，会混合使用。比如先分离动静，再路由服务，再负载机器

**正向代理与反向代理**

1、代理：客户端自己请求出现困难。客户请了一个代理，来代自己做事，就叫代理。

    比如代理律师，代购，政府机关办事的代理人等等。

2、反向代理，服务端推出的一个代理招牌。

**nginx安装：**

**源码编译方式：**

安装make：yum -y install autoconf automake make

安装g++: yum -y install gcc gcc-c++

#一般系统中已经装了了make和g++，无须再装

yum -y install pcre pcre-devel

yum -y install zlib zlib-devel

yum install -y openssl openssl-devel

#安装nginx依赖的库

wget  <http://nginx.org/download/nginx-1.9.15.tar.gz>

tar -zxvf nginx-1.9.0.tar.gz

cd nginx-1.9.0

./configure   --prefix=/usr/local/nginx --with-http\_stub\_status\_module --with-http\_ssl\_module

#配置

#--prefix指定安装目录

#--with-http\_ssl\_module安装https模块

#creating objs/Makefile 代表编译成功

make && make install

#make编译

#make install安装

**yum方式：**

centos6：yum install epel-release -y   #yum扩展源

yum install nginx -y

**目录结构：**

* Conf  配置文件
* Html   网页文件
* Logs   日志文件
* Sbin    二进制程序

**启停命令：**

./nginx -c nginx.conf的文件。如果不指定，默认为NGINX\_HOME/conf/nginx.conf

./nginx -s stop  停止

./nginx -s quit退出

./nginx -s reload 重新加载nginx.conf

**发送信号的方式**

kill -QUIT  进程号 安全停止

kil -TERM  进程号 立即停止

**nginx模型概念：**

**Nginx会按需同时运行多个进程：**

一个主进程(master)和几个工作进程(worker)，配置了缓存时还会有缓存加载器进程(cache loader)和缓存管理器进程(cache manager)等。

所有进程均是仅含有一个线程，并主要通过“共享内存”的机制实现进程间通信。

主进程以root用户身份运行，而worker、cache loader和cache manager均应以非特权用户身份（user配置项）运行。

**主进程主要完成如下工作：**

1. 读取并验正配置信息；

2. 创建、绑定及关闭套接字；

3. 启动、终止及维护worker进程的个数；

4. 无须中止服务而重新配置工作特性；

5. 重新打开日志文件；

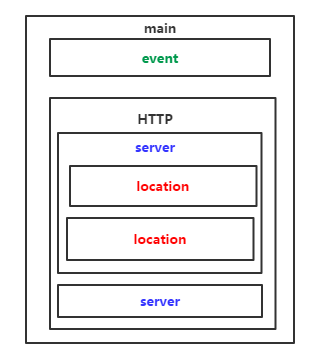
**worker进程主要完成的任务包括：**

1. 接收、传入并处理来自客户端的连接；

2. 提供反向代理及过滤功能；

3. nginx任何能完成的其它任务；

**nginx.conf配置文件结构**



#user nobody; #主模块命令， 指定Nginx的worker进程运行用户以及用户组，默认由nobody账号运行。

worker\_processes 1;#指定Nginx要开启的进程数。

worker\_rlimit\_nofile 100000; #worker进程的最大打开文件数限制

#error\_log logs/error.log;

#error\_log logs/error.log notice;

#error\_log logs/error.log info;

#pid logs/nginx.pid;

events {

use epoll;

worker\_connections 1024;

}

/\*

**以上这块配置代码是对nginx全局属性的配置。**

  user :主模块命令， 指定Nginx的worker进程运行用户以及用户组，默认由nobody账号运行。

  worker\_processes: 指定Nginx要开启的进程数。

  error log:用来定义全局错设日志文件的路径和日志名称。

日志输出级别有debug，info，notice，warn，error，crit 可供选择，其中debug输出日志最为详细，面crit（严重）输出日志最少。默认是error

  pid: 用来指定进程id的存储文件位置。

  event：设定nginx的工作模式及连接数上限，

其中参数use用来指定nginx的工作模式（这里是epoll，epoll是多路复用IO(I/O Multiplexing)中的一种方式）,

nginx支持的工作模式有select ,poll,kqueue,epoll,rtsig,/dev/poll。

其中select和poll都是标准的工作模式，kqueue和epoll是高效的工作模式，对于linux系统，epoll是首选。

    worker\_connection是设置nginx每个进程最大的连接数，默认是1024，所以nginx最大的连接数max\_client=worker\_processes \* worker\_connections。

进程最大连接数受到系统最大打开文件数的限制，需要设置ulimit。

\*/

**#下面部分是nginx对http服务器相关属性的设置**

http {

include mime.types; 主模块命令，对配置文件所包含文件的设定，减少主配置文件的复杂度，相当于把部分设置放在别的地方，然后在包含进来，保持主配置文件的简洁

default\_type application/octet-stream; 默认文件类型，当文件类型未定义时候就使用这类设置的。

#log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" ' 指定nginx日志的格式

# '$status $body\_bytes\_sent "$http\_referer" '

# '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

#access\_log logs/access.log main;

sendfile on; 开启高效文件传输模式（zero copy 方式），避免内核缓冲区数据和用户缓冲区数据之间的拷贝。

#tcp\_nopush on; 开启TCP\_NOPUSH套接字（sendfile开启时有用）

#keepalive\_timeout 0; 客户端连接超时时间

keepalive\_timeout 65;

#gzip on; 设置是否开启gzip模块

**#下面是server段虚拟主机的配置**

server {

listen 80; 虚拟主机的服务端口

server\_name localhost; 用来指定ip或者域名，多个域名用逗号分开

#charset koi8-r;

location / {

#地址匹配设置，支持正则匹配，也支持条件匹配，这里是默认请求地址，用户可以location命令对nginx进行动态和静态网页过滤处理

root html; 虚拟主机的网页根目录

index index.html index.htm; 默认访问首页文件

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

}

**Nginx日志描述**

通过访问日志，你可以得到用户地域来源、跳转来源、使用终端、某个URL访问量等相关信息；通过错误日志，你可以得到系统某个服务或server的性能瓶颈等。因此，将日志好好利用，你可以得到很多有价值的信息。

**日志格式**

打开nginx.conf配置文件：vi /usr/local/nginx/conf/nginx.conf

日志部分内容：

#access\_log  logs/access.log  main;

日志生成的到Nginx根目录logs/access.log文件，默认使用“main”日志格式，也可以自定义格式。

**默认“main”日志格式：** 参数明细表：

|  |  |
| --- | --- |
| $remote\_addr | 客户端的ip地址(代理服务器，显示代理服务ip) |
| $remote\_user | 用于记录远程客户端的用户名称（一般为“-”） |
| $time\_local | 用于记录访问时间和时区 |
| $request | 用于记录请求的url以及请求方法 |
| $status | 响应状态码，例如：200成功、404页面找不到等。 |
| $body\_bytes\_sent | 给客户端发送的文件主体内容字节数 |
| $http\_user\_agent | 用户所使用的代理（一般为浏览器） |
| $http\_x\_forwarded\_for | 可以记录客户端IP，通过代理服务器来记录客户端的ip地址 |
| $http\_referer | 可以记录用户是从哪个链接访问过来的 |

查看日志命令tail -f /usr/local/nginx/logs/access.log

**日志配置和及切割**

/etc/init.d/rsyslog start #系统日志，如不开启，看不到定时任务日志

/etc/rc.d/init.d/crond start #定时任务开启

**编写sh：**

#!/bin/bash

#设置日志文件存放目录

LOG\_HOME="/usr/local/nginx/logs/"

#备分文件名称

LOG\_PATH\_BAK="$(date -d yesterday +%Y%m%d%H%M)"

#重命名日志文件

mv ${LOG\_HOME}/access.log ${LOG\_HOME}/access.${LOG\_PATH\_BAK}.log

mv ${LOG\_HOME}/error.log ${LOG\_HOME}/error.${LOG\_PATH\_BAK}.log

#向nginx主进程发信号重新打开日志

kill -USR1 `cat ${LOG\_HOME}/nginx.pid`

**配置cron：**

\*/1 \* \* \* \* /usr/local/nginx/sbin/logcut.sh