**Nginx实现高可用**

# 课程目标

Nginx介绍

Ningx应用场景

Windows环境下安装Nginx

Windows环境下实现反向代理

Windows环境下实现负载均衡

Nginx实现网关接口跨域解决方案

Nginx实现防盗链

Nginx防止DDOS

Linux环境下安装Nginx

Linux环境下实现反向代理

Linux环境下nginx+keepalived实现高可用

Linux环境下Session共享解决方案

高并发解决方案

# 一、nginx入门

## 1.1什么是nginx？

nginx是一款高性能的http 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器。由俄罗斯的程序设计师Igor Sysoev所开发，官方测试nginx能够支支撑5万并发链接，并且cpu、内存等资源消耗却非常低，运行非常稳定，所以现在很多知名的公司都在使用nginx。

## 1.2 nginx应用场景

1、http服务器。Nginx是一个http服务可以独立提供http服务。可以做网页静态服务器。

2、虚拟主机。可以实现在一台服务器虚拟出多个网站。例如个人网站使用的虚拟主机。

3、反向代理，负载均衡。当网站的访问量达到一定程度后，单台服务器不能满足用户的请求时，需要用多台服务器集群可以使用nginx做反向代理。并且多台服务器可以平均分担负载，不会因为某台服务器负载高宕机而某台服务器闲置的情况。



## 1.3 Windows环境下安装Nginx

解压:nginx-windows

双击: nginx.exe



能看到nginx欢迎界面说明,nginx安装成功

演示下 nginx做静态服务器

### 1.3.1 windows常用命令

nginx.exe -s stop –停止

## 1.4 nginx优缺点

占内存小，可以实现高并发连接、处理响应快。

可以实现http服务器、虚拟主机、反向代理、负载均衡。

nginx配置简单

可以不暴露真实服务器IP地址

## 1.4 nginx.conf 介绍

### 1.4.1 nginx.conf文件的结构

nginx的配置由特定的标识符(指令符)分为多个不同的模块。   
指令符分为**简单指令**和**块指令**。

* 简单指令格式：[name parameters;]
* 块指令格式：和简单指令格式有一样的结构，但其结束标识符不是分号，而是大括号{},块指令内部可以包含simple directives 和block directives, 可以称块指令为上下文(e.g. events, http, server, location)

conf文件中，所有不属于块指令的简单指令都属于main上下文的，http块指令属于main上下文，server块指令http上下文。

### 1.4.2 配置静态访问

Web server很重要一部分工作就是提供静态页面的访问，例如images, html page。nginx可以通过不同的配置，根据request请求，从本地的目录提供不同的文件返回给客户端。   
打开安装目录下的nginx.conf文件，默认配置文件已经在http指令块中创建了一个空的server块，在nginx-1.8.0中的http块中已经创建了一个默认的server块。内容如下：

|  |
| --- |
| **server {**  **listen 80;**  **server\_name localhost;**  **location / {**  **root html;**  **index index.html index.htm;**  **}**  **error\_page 500 502 503 504 /50x.html;**  **location = /50x.html {**  **root html;**  **}**  **}** |

## 1.4 nginx实现反向代理

### 1.4.1什么是反向代理?

反向代理（Reverse Proxy）方式是指以[代理服务器](https://baike.baidu.com/item/%E4%BB%A3%E7%90%86%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "_blank)来接受internet上的连接请求，然后将请求转发给内部网络上的服务器，并将从服务器上得到的结果返回给internet上请求连接的客户端，此时代理服务器对外就表现为一个反向代理服务器。

启动一个Tomcat 127.0.0.1:8080

使用nginx反向代理 8080.itmayiedu.com 直接跳转到127.0.0.1:8080

### 1.4.1Host文件新增

|  |
| --- |
| 127.0.0.1 8080.itmayiedu.com  127.0.0.1 b8081.itmayiedu.com |

### 1.4.2 nginx.conf 配置

配置信息:

|  |
| --- |
| server {  listen 80;  server\_name 8080.itmayiedu.com;  location / {  proxy\_pass http://127.0.0.1:8080;  index index.html index.htm;  }  }  server {  listen 80;  server\_name b8081.itmayiedu.com;  location / {  proxy\_pass http://127.0.0.1:8081;  index index.html index.htm;  }  } |

## 1.5 nginx实现负载均衡

### 1.5.1什么是负载均衡

负载均衡 建立在现有网络结构之上，它提供了一种廉价有效透明的方法扩展网络设备和服务器的带宽、增加吞吐量、加强网络数据处理能力、提高网络的灵活性和可用性。

负载均衡，英文名称为Load Balance，其意思就是分摊到多个操作单元上进行执行，例如Web服务器、FTP服务器、企业关键应用服务器和其它关键任务服务器等，从而共同完成工作任务。



### 1.5.3负载均衡策略

|  |
| --- |
| **1、轮询（默认）** 每个请求按时间顺序逐一分配到不同的后端服务器，如果后端服务器down掉，能自动剔除。  upstream backserver {  server 192.168.0.14;  server 192.168.0.15;  }   **2、指定权重** 指定轮询几率，weight和访问比率成正比，用于后端服务器性能不均的情况。  upstream backserver {  server 192.168.0.14 weight=10;  server 192.168.0.15 weight=10;  }   **3、IP绑定 ip\_hash** 每个请求按访问ip的hash结果分配，这样每个访客固定访问一个后端服务器，可以解决session的问题。  upstream backserver {  ip\_hash;  server 192.168.0.14:88;  server 192.168.0.15:80;  } |

### 1.5.4配置代码

|  |
| --- |
| upstream backserver {  server 127.0.0.1:8080;  server 127.0.0.1:8081;  }  server {  listen 80;  server\_name www.itmayiedu.com;  location / {  proxy\_pass http://backserver;  index index.html index.htm;  }  } |

### 1.5.4宕机轮训配置规则

|  |
| --- |
| server {  listen 80;  server\_name www.itmayiedu.com;  location / {  proxy\_pass http://backserver;  index index.html index.htm;  proxy\_connect\_timeout 1;  proxy\_send\_timeout 1;  proxy\_read\_timeout 1;  }    } |

## 1.6 nginx解决网站跨域问题

配置:

|  |
| --- |
| server {  listen 80;  server\_name www.itmayiedu.com;  location /A {  proxy\_pass http://a.a.com:81/A;  index index.html index.htm;  }  location /B {  proxy\_pass http://b.b.com:81/B;  index index.html index.htm;  }  } |

## 1.7 nginx配置防盗链

|  |
| --- |
| location ~ .\*\.(jpg|jpeg|JPG|png|gif|icon)$ {  valid\_referers blocked http://www.itmayiedu.com www.itmayiedu.com;  if ($invalid\_referer) {  return 403;  }  } |

## 1.7 nginx配置DDOS

### 1.7.1限制请求速度

设置Nginx、Nginx Plus的连接请求在一个真实用户请求的合理范围内。比如，如果你觉得一个正常用户每两秒可以请求一次登录页面，你就可以设置Nginx每两秒钟接收一个客户端IP的请求（大约等同于每分钟30个请求）。

|  |
| --- |
| limit\_req\_zone $binary\_remote\_addr zone=one:10m rate=30r/m;  server {  ...  location /login.html {  limit\_req zone=one;  ...  }  } |

`limit\_req\_zone`命令设置了一个叫one的共享内存区来存储请求状态的特定键值，在上面的例子中是客户端IP($binary\_remote\_addr)。location块中的`limit\_req`通过引用one共享内存区来实现限制访问/login.html的目的。

### 1.7.1限制请求速度

设置Nginx、Nginx Plus的连接数在一个真实用户请求的合理范围内。比如，你可以设置每个客户端IP连接/store不可以超过10个。