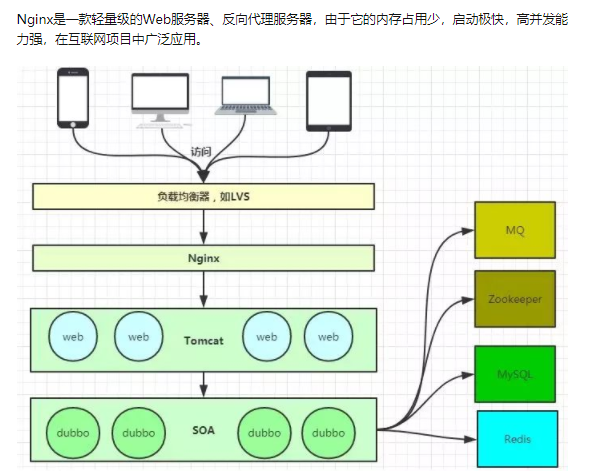
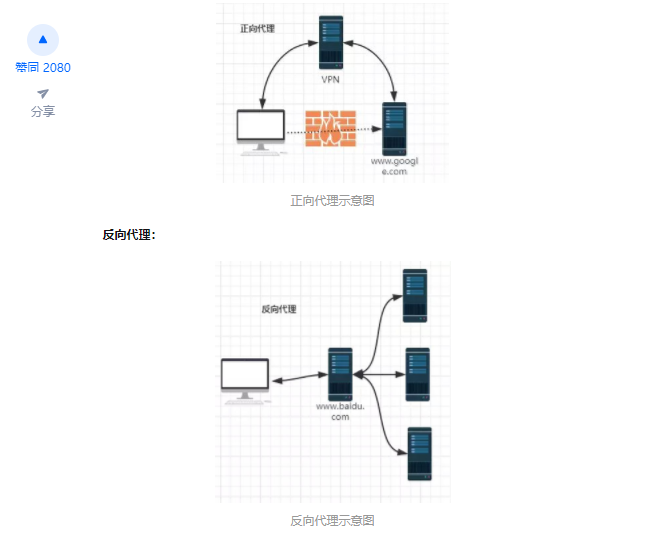
nginx介绍

# 一、nginx简介

## 1、nginx简介



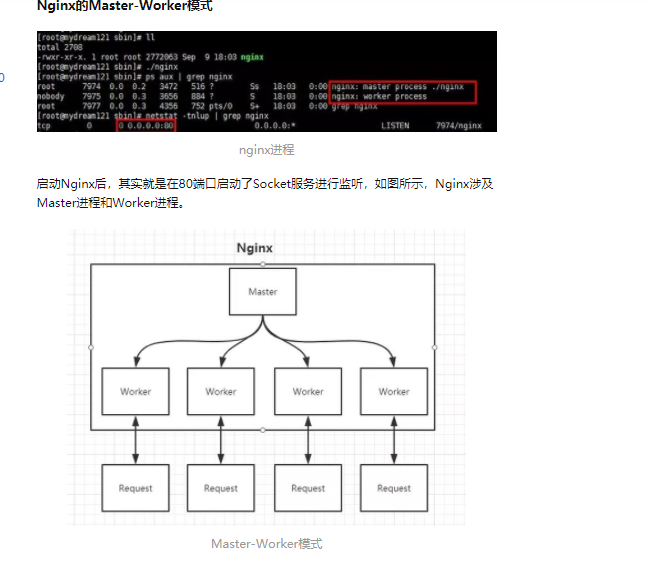
## 2、反向代理：



由于防火墙的原因，我们并不能直接访问谷歌，那么我们可以借助VPN来实现，这就是一个简单的正向代理的例子。这里你能够发现，正向代理“代理”的是客户端，而且客户端是知道目标的，而目标是不知道客户端是通过VPN访问的。

当我们在外网访问百度的时候，其实会进行一个转发，代理到内网去，这就是所谓的反向代理，即反向代理“代理”的是服务器端，而且这一个过程对于客户端而言是透明的。

## 3、nginx的master-worker模式



Master进程的作用是？

**读取并验证配置文件nginx.conf；管理worker进程；**

Worker进程的作用是？

**每一个Worker进程都维护一个线程（避免线程切换），处理连接和请求；注意Worker进程的个数由配置文件决定，一般和CPU个数相关（有利于进程切换），配置几个就有几个Worker进程。**

## 4、思考：Nginx如何做到热部署？

所谓热部署，就是配置文件nginx.conf修改后，不需要stop Nginx，不需要中断请求，就能让配置文件生效！（nginx -s reload 重新加载/nginx -t检查配置/nginx -s stop）

通过上文我们已经知道worker进程负责处理具体的请求，那么如果想达到热部署的效果，可以想象：

方案一：

修改配置文件nginx.conf后，主进程master负责推送给woker进程更新配置信息，woker进程收到信息后，更新进程内部的线程信息。（有点valatile的味道）

方案二：

修改配置文件nginx.conf后，重新生成新的worker进程，当然会以新的配置进行处理请求，而且新的请求必须都交给新的worker进程，至于老的worker进程，等把那些以前的请求处理完毕后，kill掉即可。

Nginx采用的就是方案二来达到热部署的！

## 5、思考：Nginx如何做到高并发下的高效处理？

上文已经提及Nginx的worker进程个数与CPU绑定、worker进程内部包含一个线程高效回环处理请求，这的确有助于效率，但这是不够的。

**作为专业的程序员，我们可以开一下脑洞：BIO/NIO/AIO、异步/同步、阻塞/非阻塞...**

要同时处理那么多的请求，要知道，有的请求需要发生IO，可能需要很长时间，如果等着它，就会拖慢worker的处理速度。

**Nginx采用了Linux的epoll模型，epoll模型基于事件驱动机制，它可以监控多个事件是否准备完毕，如果OK，那么放入epoll队列中，这个过程是异步的。worker只需要从epoll队列循环处理即可。**

## 6、思考：Nginx挂了怎么办？

Nginx既然作为入口网关，很重要，如果出现单点问题，显然是不可接受的。

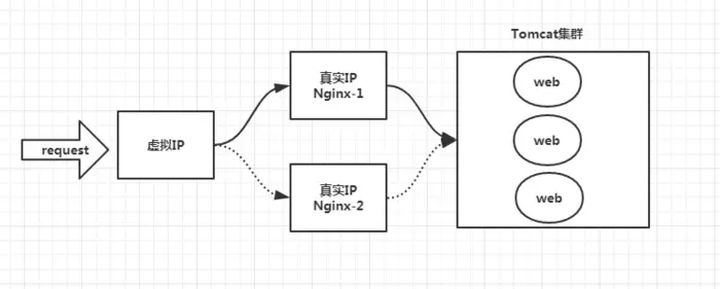
答案是：**Keepalived+Nginx实现高可用**。

Keepalived是一个高可用解决方案，主要是用来防止服务器单点发生故障，可以通过和Nginx配合来实现Web服务的高可用。（其实，Keepalived不仅仅可以和Nginx配合，还可以和很多其他服务配合）

Keepalived+Nginx实现高可用的思路：

第一：请求不要直接打到Nginx上，应该先通过Keepalived（这就是所谓虚拟IP，VIP）

第二：Keepalived应该能监控Nginx的生命状态（提供一个用户自定义的脚本，定期检查Nginx进程状态，进行权重变化,，从而实现Nginx故障切换）

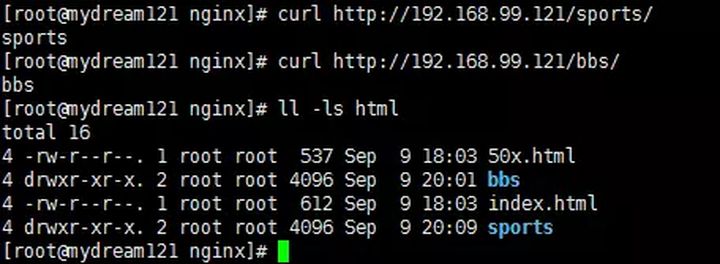
Keepalived+Nginx

## 7、我们的主战场：nginx.conf

很多时候，在开发、测试环境下，我们都得自己去配置Nginx，就是去配置nginx.conf。

nginx.conf是典型的分段配置文件，下面我们来分析下。

### 1）虚拟主机

http的server段访问结果

其实这是把Nginx作为web server来处理静态资源。

第一：location可以进行正则匹配，应该注意正则的几种形式以及优先级。（这里不展开）

第二：Nginx能够提高速度的其中一个特性就是：动静分离，就是把静态资源放到Nginx上，由Nginx管理，动态请求转发给后端。

第三：我们可以在Nginx下把静态资源、日志文件归属到不同域名下（也即是目录），这样方便管理维护。

第四：Nginx可以进行IP访问控制，有些电商平台，就可以在Nginx这一层，做一下处理，内置一个黑名单模块，那么就不必等请求通过Nginx达到后端在进行拦截，而是直接在Nginx这一层就处理掉。

### 2）反向代理【proxy\_pass】

所谓反向代理，很简单，其实就是在location这一段配置中的root替换成**proxy\_pass**即可。root说明是静态资源，可以由Nginx进行返回；而proxy\_pass说明是动态请求，需要进行转发，比如代理到Tomcat上。

反向代理，上面已经说了，过程是透明的，比如说request -> Nginx -> Tomcat，那么对于Tomcat而言，请求的IP地址就是Nginx的地址，而非真实的request地址，这一点需要注意。不过好在Nginx不仅仅可以反向代理请求，还可以由用户**自定义设置HTTP HEADER**。

### 3）负载均衡【upstream】

上面的反向代理中，我们通过proxy\_pass来指定Tomcat的地址，很显然我们只能指定一台Tomcat地址，那么我们如果想指定多台来达到负载均衡呢？

第一，通过**upstream**来定义一组Tomcat，并指定负载策略（IPHASH、加权论调、最少连接），健康检查策略（Nginx可以监控这一组Tomcat的状态）等。

第二，将proxy\_pass替换成upstream指定的值即可。

**负载均衡可能带来的问题？**

负载均衡所带来的明显的问题是，一个请求，可以到A server，也可以到B server，这完全不受我们的控制，当然这也不是什么问题，只是我们得注意的是：**用户状态的保存问题，如Session会话信息，不能在保存到服务器上。**

**负载均衡的方式：**

**轮询：就是将请求挨个发送到每个服务器上。**

**权重：就是按照权重值来向服务器转发请求，不同的权重值分发的请求的数量不同。**

### 4）缓存

缓存，是Nginx提供的，可以加快访问速度的机制，说白了，在配置上就是一个开启，同时指定目录，让缓存可以存储到磁盘上。具体配置，大家可以参考Nginx官方文档，这里就不在展开了。

## 8、[Nginx详解](https://link.zhihu.com/?target=https%3A//mp.weixin.qq.com/s/XoqGvYBabW8YBl9xEeNYZw)：

https://mp.weixin.qq.com/s/XoqGvYBabW8YBl9xEeNYZw

## 9、[一文详解负载均衡和反向代理的真实区别](https://link.zhihu.com/?target=https%3A//mp.weixin.qq.com/s/TYM83F2O-keMvn4ZYa5nqw)

https://mp.weixin.qq.com/s/TYM83F2O-keMvn4ZYa5nqw

# 二、linux安装nginx

## 1、安装gcc--安装在根目录下即可

yum install gcc-c++

如果报错：This system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.

是因为我们没有配置yum仓库。

配置仓库的方法：

1..进入到/etc/yum.repos.d/目录（该目录存放着Yum软件仓库的配置文件）

2.使用vim编辑器创建一个名为rhe17.repo的新配置文件（文件名称可随便，但后缀必须为.repo）

root@2022 ~]# cd /etc/yum.repos.d/

[root@2022 yum.repos.d]# vim rhe17.repo

[rhe17]

name=rhe17

baseurl=file:///media/cdrom

enabled=1

gpgcheck=0

保存退出

3.把光盘设备的系统镜像挂载到/media/cdrom目录  
[root@2022~]*# mkdir -p /media/cdrom*

[root@2022 ~]*# mount /dev/cdrom /media/cdrom*

然后在安装就可以了。

## 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

## 5.下载Nginx

wget https://nginx.org/download/nginx-1.19.9.tar.gz

## 6.解压nginx

tar -zxvf nginx-1.19.9.tar.gz

cd nginx-1.19.9

## 7.执行nginx-configure文件

./configure

## 8.make命令编译

执行完后会有一个MakeFile文件夹

make 是一个命令工具，它解释 Makefile 中的指令（应该说是规则）。在 Makefile文件中描述了整个工程所有文件的编译顺序、编译规则

注：需要先安装make包，否则会报没有make命令的错误，可先检查有没有make包 make -v

yum -y install gcc automake autoconf libtool make

make

make install

## 9.查询nginx 安装目录

whereis nginx

## 10.进入安装目录执行nginx

前往安装目录找到sbin 执行nginx

./nginx

# 三、安装启动过程问题及解决方案~

## 1、yum install 404解决方案

这个解决方案是针对centOS的，如果是redHAT8的话可以参考(二、1.)的解决方案。

1）、进入配置文件内，删除所有的.repo文件（也可以备份）

#进入配置文件夹

cd /etc/yum.repos.d/

#删除旧的配置文件

rm \*.repo

#输入“y”回车确认

ls确保该目录下的.repo文件已完全删除

下载可以用的.repo文件

wget -O /etc/yum.repos.d/CentOS-Base.repo https://mirrors.aliyun.com/repo/Centos-vault-8.5.2111.repo

如果你没有安装wget，也可以用下面命令：

curl -o /etc/yum.repos.d/CentOS-Base.repo https://mirrors.aliyun.com/repo/Centos-vault-8.5.2111.rep

## 2、如何修改nginx监听的端口号？

查找nginx.conf文件的位置：whereis nginx.conf

nginx: /usr/local/nginx

进入到对应的目录

cd /usr/local/nginx

cd conf

vim nginx.conf

将server中的 listen后面的参数修改为想要监听的端口

重新加载文件

source nginx.conf

然后重新启动nginx(见上文)

同样在防火墙中需要将3000端口打开，就可以在浏览器中查看nginx的初始界面了。

# 四、nginx常用命令

## 1、配置环境变量

whereis nginx 找到nginx 安装目录

前往根目录 找到etc文件夹

vim profile 打开配置文件

添加环境变量（前半段node忽略）

export PATH=$PATH:/node-v14.19.1-linux-x64/bin:/usr/local/nginx/sbin

重新执行profile文件

## 2.Nginx常用命令介绍

### 1）查看nginx版本号

nginx -v

nginx -V

### 2）启动nginx

直接执行 nginx 即可

### 3）停止nginx

nginx -s stop

Stop 是立即停止

nginx -s quit

Quit 是一个优雅的关闭方式，Nginx在退出前完成已经接受的请求处理

### 4）重载nginx配置文件

nginx -s reload，改了配置文件不需要重新启动nginx服务

### 5）查看nginx进程

ps -ef | grep nginx

ps -ef

输出标准格式的linux进程命令

grep nginx

grep命令 是查找， 是一种强大的文本搜索工具 我们这儿是查找nginx

### 6）检查nginx配置文件是否有语法错误

nginx -t

# 五、nginx配置文件

## 1、配置文件模块

Nginx的主配置文件是nginx.conf，这个配置文件一共由三部分组成，分别为全局块、events块和http块。

在http块中，又包含http全局块、多个server块。

每个server块中，可以包含server全局块和多个location块。在同一配置块中嵌套的配置块，各个之间不存在次序关系

## 2、全局块

全局块是默认配置文件从开始到events块之间的一部分内容，主要设置一些影响Nginx服务器整体运行的配置指令，因此，这些指令的作用域是Nginx服务器全局。

user [user] [group] 指定可以运行nginx服务的用户和用户组，只能在全局块配置 user指令在Windows上不生效，如果你制定具体用户和用户组会报警告

worker\_processes nginx进程数量worker\_processes 比如设置为2，nginx将会开启一个master进程和2两个worker进程

pid logs/nginx.pid 存放pid文件

error\_log logs/error.log; 全局错误日志类型 debug info warn error 存放地址

## 3.events块

events块涉及的指令主要影响Nginx服务器与用户的网络连接。常用到的设置包括是否开启对多worker process下的网络连接进行序列化，是否允许同时接收多个网络连接，选取哪种事件驱动模型处理连接请求，每个worker process可以同时支持的最大连接数等

accept\_mutex 默认开启-开启之后nginx 的多个worker将会以串行的方式来处理，只会有一个worker将会被唤起，其他的worker继续睡眠，如果不开启将会造成惊群效应多个worker全部唤起不过只有一个Worker能获取新连接，其它的Worker会重新进入休眠状态

worker\_connections 单个进程最大连接数（最大连接数=连接数+进程数）

## 4.http块

http块是Nginx服务器配置中的重要部分，代理、缓存和日志定义等绝大多数的功能和第三方模块的配置都可以放在这个模块中。

**include指令**，用于引入其他的配置文件

**default\_type** 如果Web程序没设置，Nginx也没对应文件的扩展名，就用Nginx 里默认的 default\_type定义的处理方式。default\_type application/octet-stream; #nginx默认文件类型

**log\_forma**t指令，用于定义日志格式，此指令只能在http块中进行配置

**sendfile** 简单来说就是启用sendfile()系统调用来替换read()和write()调用，减少系统上下文切换从而提高性能，当 nginx 是静态文件服务器时，能极大提高nginx的性能表现

**keepalive\_timeout** HTTP 有一个 KeepAlive 模式，它告诉 webserver 在处理完一个请求后保持这个 TCP 连接的打开状态。若接收到来自客户端的其它请求，服务端会利用这个未被关闭的连接，而不需要再建立一个连接。

gzip 开启Gzip压缩功能， 可以使网站的css、js 、xml、html 文件在传输时进行压缩，提高访问速度, 进而优化Nginx性能

## 5.server块

每一个http块都可以包含多个server块，而每个server块就相当于一台虚拟主机，它内部可有多台主机联合提供服务，一起对外提供在逻辑上关系密切的一组服务

**listen**指令的配置非常灵活，可以单独制定ip，单独指定端口或者同时指定ip和端口

listen 127.0.0.1:8000; #只监听来自127.0.0.1这个IP，请求8000端口的请求

listen 127.0.0.1; #只监听来自127.0.0.1这个IP，请求80端口的请求（不指定端口，默认80）

listen 9999; #监听来自所有IP，请求9999端口的请求

listen \*:9999; #和上面效果一样

listen localhost:8000; #和第一种效果一致

**server\_name** nginx 允许一个虚拟主机有一个或多个名字，也可以使用通配符"\*"来设置虚拟主机的名字 支持 ip 域名 通配符 正则等

server\_name localhost;

## 6.location块

每个server块中可以包含多个location块。在整个Nginx配置文档中起着重要的作用，而且Nginx服务器在许多功能上的灵活性往往在location指令的配置中体现出来

location 指令可以分为以下 3 类：

前缀字符串匹配

正则表达式匹配

用于内部跳转的命名location

前缀字符串匹配

精确匹配 =

前缀匹配 ^~（立刻停止后续的正则搜索）

按文件中顺序的正则匹配 ~或~\*

匹配不带任何修饰的前缀匹配。

location root

root 指定目录的上级目录，并且该上级目录要含有locatoin指定名称的同名目录。

location /img/ {

root /var/www/image;

}

若按照这种配置的话，则访问/img/目录下的文件时，nginx会去/var/www/image/img/目录下找文件

# 六、nginx反向代理

## 1、Nginx反向代理的配置语法

反向代理中的常用指令：

proxy\_pass

proxy\_set\_header

proxy\_pass

该指令用来设置被代理服务器地址，可以是主机名称、IP地址加端口号形式。

**案例1代理到哔哩哔哩**

location / {

root html;

index index.html index.htm;

proxy\_pass http://bilibili.com;

}

访问/就会被转到哔哩哔哩

案例2 nginx反向代理解决跨域

前端代码

a.onclick = () => {

let xhr = new XMLHttpRequest()

xhr.open('GET','/api/portal/list')

xhr.onreadystatechange = () => {

if(xhr.readyState == 4 && xhr.status == 200){

console.log(xhr.responseText);

}

}

xhr.send(null)

}

express 服务端代码

const express = require('express')

const app = express()

app.get('/portal/list', (req, res) => {

res.json({

code: 200,

message: "搞咩啊"

})

})

app.listen(9000,()=>{

console.log('success');

})

nginx 配置文件

location /api/ {

proxy\_pass http://localhost:9000/;

}

截取到/api/ 将会转发到 http://localhost:9000/

proxy\_set\_header

该指令可以更改Nginx服务器接收到的客户端请求的请求头信息，然后将新的请求头发送给代理的服务器

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

proxy\_set\_header X-Real-Port $remote\_port;

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

三个header分别表示：

X-Real-IP 客户端或上一级代理ip

X-Real-Port 客户端或上一级端口

X-Forwarded-For 包含了客户端和各级代理ip的完整ip链路

其中X-Real-IP是必需的，后两项选填。当只存在一级nginx代理的时候X-Real-IP和X-Forwarded-For是一致的，而当存在多级代理的时候，X-Forwarded-For 就变成了如下形式

$remote\_addr是前一节点的IP，并不一定是用户的真实IP。

# 七、Vue histroy 模式 跳转路由404 问题

原因是：vue是单页面应用，刷新界面后，nginx的配置文件中没有配置其他的vue-router配置的路由，解决方式就是将其他的路径重定向到根路径

在nginx.conf文件中。

try\_files $uri $uri/ /index.html;

~