

**Nginx组成部分：**

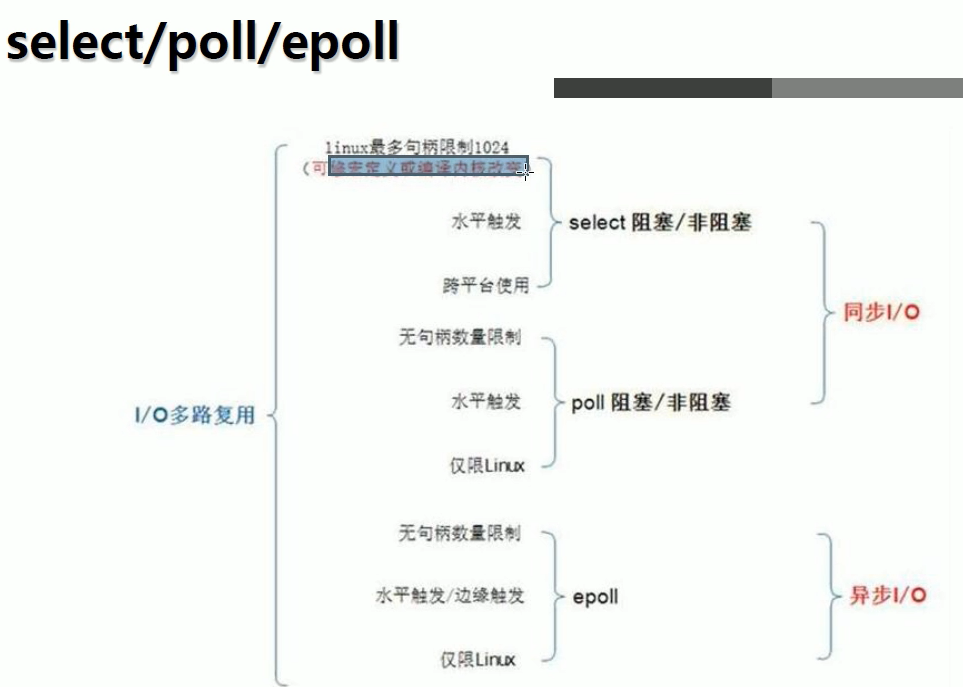
二进制文件：相当于nginx本身，由各模块源码编译出来的一个文件

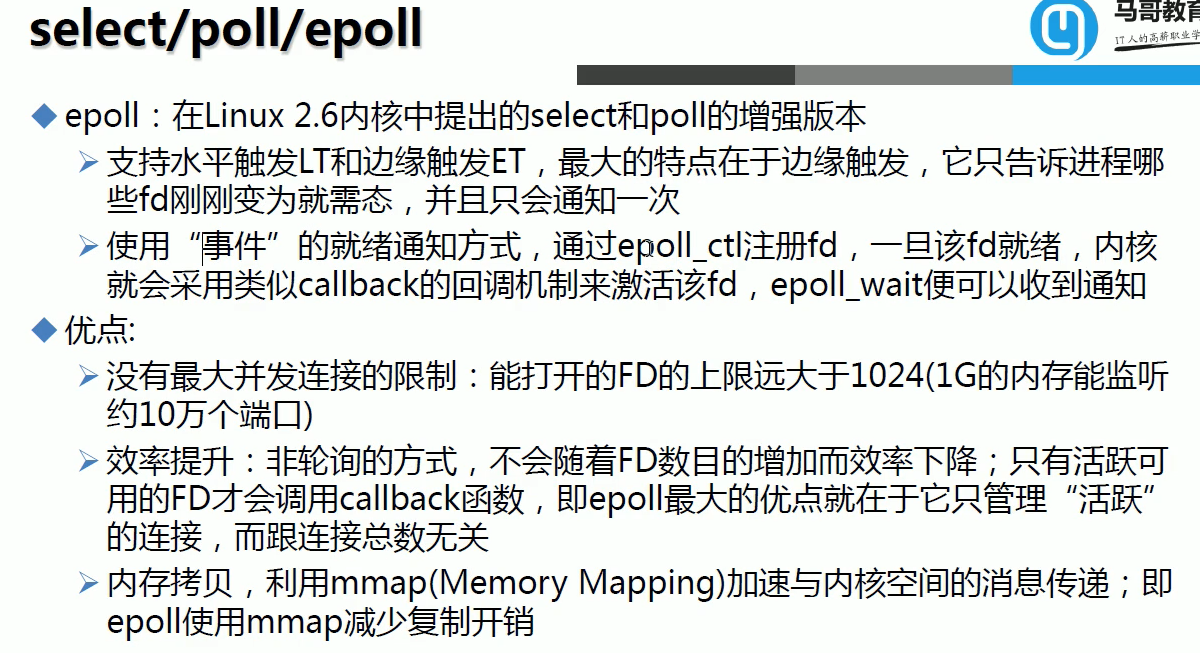
Nginx.conf：nginx的行为

Access.log：记录每一条HTTP请求信息，对运营运维做进一步分析

Error.log：类似黑匣子，定位问题







root@test ~]# /usr/local/nginx/sbin/nginx -V

nginx version: nginx/1.9.3

built by gcc 4.8.5 20150623 (Red Hat 4.8.5-28) (GCC)

built with OpenSSL 1.0.2k-fips 26 Jan 2017

TLS SNI support enabled

configure arguments: --prefix=/usr/local/nginx --with-http\_ssl\_module --with-http\_spdy\_module --with-http\_stub\_status\_module --with-pcre --with-stream

#--prefix=/usr/local/nginx #安装路径

#--with-http\_ssl\_module #安全认证模块

#--with-http\_spdy\_module

#--with-http\_stub\_status\_module #状态模块

#--with-pcre --with-stream #代理模块

192.168.2.242nginx配置说明

user root; #nginx运行的用户

worker\_processes 4; #nginx启动使用的进程数，一般与cpu数量相等

#Nginx进程平均耗费10M~12M内存

#events工作模式以及连接数上限

events {

worker\_connections 4096; #单个后台worker process进程的最大并发链接数

}

stream {

server {

listen 2356;

proxy\_connect\_timeout 10s;

proxy\_timeout 60s;

proxy\_pass 192.168.10.243:2356; #代理192.168.10.243:2356

}

server {

listen 1884; #访问192.168.10.242:1884,轮询到下面app服务器组

proxy\_connect\_timeout 5s;

proxy\_timeout 120s;

proxy\_pass app; #负载均衡

}

upstream app {

hash $remote\_addr consistent; #调度方法为哈希 远程访问一致性

server 192.168.10.242:1883 max\_fails=2 fail\_timeout=2s;

server 192.168.10.243:1883 max\_fails=2 fail\_timeout=2s;

}

}

#max\_fails：允许请求失败的次数，默认为1。当超过最大次数时，返回proxy\_next\_upstream 模块定义的错误；

#fail\_timeout：在经历了max\_fails次失败后，暂停服务的时间。max\_fails可以和fail\_timeout一起使用。

http {

include mime.types; #设定mime类型,类型由mime.type文件定义include是个主模块指令，实现对配置文件所包含的文件的设定，可以减少主配置文件的复杂度。类似于Apache中的include方法。

default\_type application/octet-stream; #default\_type属于HTTP核心模块指令，这里设定默认类型为二进制流，也就是当文件类型未定义时使用这种方式，例如在没有配置PHP环境时，Nginx是不予解析的，此时，用浏览器访问PHP文件就会出现下载窗口。

sendfile on; #sendfile 指令指定 nginx 是否调用 sendfile 函数（zero copy 方式）来输出文件，#普通应用，必须设为 on, 如用来进行下载等应用磁盘IO重负载应用，可设置为 off，#以平衡磁盘与网络I/O处理速度，降低系统的uptime.

keepalive\_timeout 65; #连接超时时间

client\_max\_body\_size 10M; #设置允许客户端请求的最大的单个文件字节数；

server {

listen 80 default backlog=2048;

listen 443 ssl; #支持80、443端口同时访问

server\_name dev.youbasan.com;

add\_header Access-Control-Allow-Origin \*;

add\_header Access-Control-Allow-Headers 'Origin, X-Requested-With, Content-Type, Accept, Token';

add\_header Access-Control-Allow-Methods GET,POST,OPTIONS;

#1. Access-Control-Allow-Origin：服务器默认是不被允许跨域的。给Nginx服务器配置Access-Control-Allow-Origin \*后，表示服务器可以接受所有的请求源（Origin）,即接受所有跨域的请求。

#2. Access-Control-Allow-Headers 是为了防止出现以下错误：Request header field Content-Type is not allowed by Access-Control-Allow-Headers in preflight response. 这个错误表示当前请求Content-Type的值不被支持。其实是我们发起了"application/json"的类型请求导致的。这里涉及到一个概念：预检请求（preflight request）,请看下面"预检请求"的介绍。

#3. Access-Control-Allow-Methods 是为了防止出现以下错误：

Content-Type is not allowed by Access-Control-Allow-Headers in preflight response. 发送"预检请求"时，需要用到方法 OPTIONS ,所以服务器需要允许该方法。

ssl\_certificate ./crt/dev.youbasan.com.pem; #证书位置

#proxy ~]# openssl req -new -x509 -key cert.key > cert.pem #本地生成证书命令

ssl\_certificate\_key ./crt/dev.youbasan.com.key; #私钥位置

#proxy ~]# openssl genrsa > cert.key #本地生成私钥

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2; #指定密码为openssl支持的格式

ssl\_ciphers ECDHE-RSA-AES256-SHA384:AES256-SHA256:RC4:HIGH:!MD5:!aNULL:!eNULL:!NULL:!DH:!EDH:!AESGCM;

# ssl\_ciphers密码加密方式

ssl\_prefer\_server\_ciphers on;

ssl\_session\_cache shared:SSL:10m;

ssl\_session\_timeout 10m;

location /nginx\_status {

stub\_status on; #启用StubStatus的工作状态统计功能

access\_log off; #关闭此页面的访问日志

}

location /file/ {

alias /home/file/; #定义别名

autoindex on;

}

# <https://www.cnblogs.com/zhang-shijie/p/5498623.html> proxy\_pass\_heager proxy\_set\_heager

1. proxy\_pass:用来设置将请求转发给的后端服务器的主机，可以是主机名、IP地址：端口的方式，也可以代理到通过upstream设置的主机组
2. proxy\_hide\_header:用于nginx服务器作为反向代理的时候，在返回给客户端http响应的时候，隐藏后端服务版本（如php版本）的信息
3. proxy\_pass\_header:与proxy\_hide\_header功能相反，proxy\_pass\_header会将原来禁止转发的header设置为允许转发
4. proxy\_pass\_request\_body:是否向后端服务器发送HTTP包体部分,可以设置在http/server或location块,如下
5. proxy\_pass\_request\_headers:是否将客户端的请求头部转发给后端服务器
6. proxy\_set\_header:可以更改或添加客户端的请求头部信息内容，并转发之后端服务器，比如在后端服务器想要获取客户端的真实IP的时候，就要更改每一个报文的头部

location / {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

# if ($remote\_addr ~\* "192.168.1.[1-254]") {

# proxy\_pass http://127.0.0.1:8686;

# }

proxy\_pass http://192.168.10.243:8330;

}

##共享生活

location = /MP\_verify\_HwiQCl3sbOwgVfIz.txt {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.243:8310/static/html/MP\_verify\_HwiQCl3sbOwgVfIz.txt;

}

location = /15TZkgEWws.txt {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.243:8310/static/html/15TZkgEWws.txt;

}

##借宿宝支付回调文件

location = /MP\_verify\_UB3KzRfV5sGkdDio.txt {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.228:8181/static/html/MP\_verify\_UB3KzRfV5sGkdDio.txt;

}

##共享床的回调文件

location = /kXwk9X5y8m.txt {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.243:8500/static/kXwk9X5y8m.txt;

}

}

server {

listen 80 ;

listen 443 ssl; #支持80和443端口同时访问

server\_name test.xjt.kkqb.net;

add\_header Access-Control-Allow-Origin \*;

add\_header Access-Control-Allow-Headers 'Origin, X-Requested-With, Content-Type, Accept, Authorization, Token';

add\_header Access-Control-Allow-Methods GET,POST,OPTIONS;

location / {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.10:8080;

}

}

server {

listen 80 ;

server\_name file.xjt.kkqb.net;

add\_header Access-Control-Allow-Origin \*;

add\_header Access-Control-Allow-Headers 'Origin, X-Requested-With, Content-Type, Accept, Token';

add\_header Access-Control-Allow-Methods GET,POST,OPTIONS;

location / {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.241;

}

}

server {

listen 80 ;

listen 443 ssl;

server\_name vote.kkqb.cn;

add\_header Access-Control-Allow-Origin \*;

add\_header Access-Control-Allow-Headers 'Origin, X-Requested-With, Content-Type, Accept, Authorization, Token';

add\_header Access-Control-Allow-Methods GET,POST,OPTIONS;

location ^~ /(member|player|vote|trade) {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.242:8383;

}

location / {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.242:8383;

}

location /client {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.242:8686;

}

location = /MP\_verify\_JdFfDRR6yAvg5l6l.txt {

proxy\_pass\_header Server;

proxy\_set\_header Host $http\_host;

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

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

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Scheme $scheme;

proxy\_pass http://192.168.10.242:8383/static/html/MP\_verify\_JdFfDRR6yAvg5l6l.txt;

}

}

include vhost/\*.conf;

}