**1 通用字段**

# 使用nginx这个用户

user nginx;

# 错误日志的输出路径

error\_log /var/log/nginx/error.log;

# events模块来用指定nginx的工作模式和工作模式及连接数上限

events {

worker\_connections 1024;

}

# nginx.conf 是主配置文件，加上include配置后，会加重/usr/share/nginx/modules/目录下的配置文件

include /usr/share/nginx/modules/\*.conf;

### 2 tcp协议代理

作TCP协议的代理，例如监听1883端口，代理到 10.8.200.xxx:30000 上面

stream {

server {

listen 1883;

proxy\_pass 10.8.200.xxx:30000;

}

}

### 3 http协议代理

假设服务器ip是 10.8.202.xx3

#### 3.1 请求转发

监听80端口，通过 **10.8.202.xx3/v1/task** 可以跳转到内网地址 http://10.8.200.xx8:8180

**# 举例说明**

**http {**

**server {**

**listen 80 default\_server;**

**listen [::]:80 default\_server;**

**server\_name \_;**

**root /usr/share/nginx/html;**

**# Load configuration files for the default server block.**

**include /etc/nginx/default.d/\*.conf;**

**location /v1/task {**

**proxy\_pass http://10.8.200.xx8:8180;**

**proxy\_set\_header Host $http\_host;**

**}**

**}**

**}**

#### 3.2 静态代理

通过 10.8.202.xx2/sim/ + [文件路径] ，可以访问 /home/xxx/sim/ 目录下的资源

例如：在服务器上，有一张图片，目录是：/home/xxx/sim/feng/1.png，则访问地址是：http://10.8.202.xx2:8011/simrs/feng/1.png

server {

listen 8011;

listen [::]:8011;

charset utf-8;

location /sim/ {

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

add\_header 'Access-Control-Allow-Credentials' 'true';

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

alias /home/xxx/sim/;

index index.html index.htm index.nginx-debian.html;

}

}

#### 3.3 绝对路径与相对路径

关键点在于 proxy\_pass 的值最后是否有 /  
有 / 是绝对路径  
无 / 是相对路径

##### 3.3.1 绝对路径

例子：请求为 10.8.202.xx2/web，那么就会代理到 10.8.202.xx1/ 上

location /web {

proxy\_pass http://10.8.202.xx1/;

proxy\_set\_header Host $http\_host;

}

##### 3.3.2 相对路径

location /web {

proxy\_pass http://10.8.202.xx1;

proxy\_set\_header Host $http\_host;

}

### 4 websocket代理

location /mqtt {

proxy\_pass http://10.8.202.xx2:8083/mqtt;

proxy\_redirect off; # 重定向

proxy\_http\_version 1.1; # 指定http版本

# nginx在拦截时需要识别出这是一个协议升级(upgrade)的请求

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

# 把原http请求的Header中的Host字段也放到转发的请求里。如果不加的话，nginx转发的请求header里就不会有Host字段，而服务器是靠这个Host值来区分你请求的是哪个域名的资源的。

proxy\_set\_header Host $host:$server\_port;

# 将$remote\_addr的值放进变量X-Real-IP中，此变量名可变，$remote\_addr的值为用户真实的ip，也就是客户端的真实ip

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

# 在只有一个代理服务器的转发的情况下，和上面的设置一样，都是用户真实的ip，如果经过两次转发，$remote\_addr部分的值是上一台nginx的ip地址，X-Forwarded-For的值就变成了“用户的真实ip，第一台nginx的ip”，ip用逗号隔开

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

#nginx会等待多长时间来获得请求的响应

proxy\_read\_timeout 7200;

}

### 5 负载均衡

#### 5.1 负载例子

通过浏览器访问 http://IP:80 ，则会轮流得到 code1 code2 code3 下面的index.html页面

//server.conf

server {

listen 8081;

root /soft/code1;

index index.html;

}

server {

listen 8082;

root /soft/code2;

index index.html;

}

server {

listen 8083;

root /soft/code3;

index index.html;

}

、//proxy.conf

upstream node {

server 你的IP:8081;

server 你的IP:8082;

server 你的IP:8083;

}

server {

server\_name localhost;

listen 80;

location / {

proxy\_pass http://node;

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;

}

}

#### 5.2 配置负载状态

状态 概述

down 当前的server暂不参与负载均衡

backup 预留的备份服务器，当其他服务器都挂掉的时候，启用

max\_fails 允许请求失败的次数 ，如果请求失败次数超过限制，则进过fail\_timeout 时间后从虚拟服务池中kill掉该服务器

fail\_timeout 经过max\_fails失败后，服务暂停时间，max\_fails设置后，必须设置fail\_timeout 值

max\_conns 限制最大的连接数，用于服务器硬件配置不同的情况下

upstream node {

server 你的IP:8081 down;

server 你的IP:8082 backup;

server 你的IP:8083 max\_fails=1 fail\_timeout=10s;

}

#### 5.3 配置调度策略

、调度算法 概述

轮询 逐一轮询，默认方式

weight 加权轮询，weight越大，分配的几率越高

ip\_hash 按照访问IP的hash结果分配，会导致来自同一IP的请求访问固定的一个后台服务器

url\_hash 按照访问URL的hash结果分配

least\_conn 最少链接数，那个服务器链接数少就会给分配

hash关键数值 hash自定义的key

————————————————

upstream node {

server 你的IP:8081 weight = 10;

server 你的IP:8082 weight = 20;

server 你的IP:8083 weight = 30;

}

upstream node {

hash $request\_uri;

server 你的IP:8081;

server 你的IP:8082;

server 你的IP:8083;

}

### 6 nginx基础

重启命令  
nginx -s reload

、配置文件地址  
/etc/nginx/nginx.conf