1. Nginx介绍

Nginx是一款自由的、开源的、高性能的HTTP服务器和反向代理服务器；同时也是一个IMAP、POP3、SMTP代理服务器；Nginx可以作为一个HTTP服务器进行网站的发布处理，另外Nginx可以作为反向代理进行负载均衡的实现。

1. 主要功能

1.静态web服务器

2.代理服务

3.负载均衡

4.动态缓存

1. 基本命令

nginx // 启动

nginx -s reload -c /etc/nginx/nginx.conf

nginx -t -c /etc/nginx/nginx.conf

-s:

reload - 热重载

quit - 响应已接收的请求后退出

stop - 直接退出

reopen - 重启服务

-t: 检查配置文件

-c: 指定配置文件位置（默认：/etc/nginx/nginx.conf，可省略）

1. 语法配置

http协议配置结构

http {

sendfile on;

...

server {

listen 80;

...

location / {

root /usr/share/nginx/html;

...

}

location = /50x.html {

...

}

}

server {

...

}

}

虚拟主机配置方式

1、基于主机多IP的方式：在一个操作系统上面可以配置多个IP，用同一个nginx，针对不同的IP对外提供服务（示例：listen 192.168.1.8;）

2、基于端口的配置方式：不同的虚拟主机nginx，监听的端口不一样，通过端口区分不同的虚拟主机（示例：listen 80;）

3、基于多host名称方式（多域名方式）：基于不同的主机名称进行虚拟主机的配置（示例：server\_name localhost ;）

日志：

log\_format：日志格式

格式：log\_format name string;

示例：log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" ';

nginx内置变量：http://nginx.org/en/docs/http/ngx\_http\_core\_module.html#var\_status

access\_log：每次http请求的访问状态

格式：access\_log 文件路径 log\_format名称

示例：access\_log /var/log/nginx/access.log main;

内置变量地址：http://nginx.org/en/docs/http/ngx\_http\_core\_module.html#var\_status

error\_log：记录处理http请求的错误状态以及nginx本身服务的错误状态

格式：error\_log 文件路径 日志级别（debug, info, notice, warn, error, crit 默认为crit）

示例：error\_log /var/log/nginx/error.log warn;

location规则：

= 完全匹配 （优先级最高）

^~ 前缀匹配（优先级最高）

~ 和 ~\* 正则匹配 ，前者区分大小写，后者不区分大小写

静态web服务器

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 配置名称 | context | 可选值 | 默认值 | 释义 | 备注 |
| sendfile | http,server,location,if in location | off,on | off | 是否开启静态服务器 |  |
| tcp\_nopush | http,server,location | off,on | off | 数据包达到一定大小时传输，适用于大文件 | 需开启sendfile  与tcp\_nodelay互斥 |
| tch\_nodelay | http,server,location | off,on | on | 数据包立即传输 | 同tcp\_nodelay |
| gzip | http,server,location,if in location | off,on | off | 压缩传输 |  |
| gzip\_comp\_level | http,server,location | 自然数 | 1 | 压缩等级 | 需开启gzip |
| gzip\_http\_version | http,server,location | 1.0,1.1 | 1.1 | http协议版本 | 需开启gzip |
| gzip\_static | http,server,location | off,on | off | gzip预读 | 需要先压缩 |
| expires | http,server,location,if in location | time,epoch,max,off | off | 添加响应头expires、cache-control，以控制缓存过期时间 |  |
| add\_header | http,server,location,if in location | 任意 | - | 新增头信息 | Access-Control-Allow-Origin跨域等 |
| valid\_referers | server,location | none:没有referer信息  blocked:referer不带协议  ip  server\_name:域名，支持正则及\* | - | 配置防盗链  valid\_referers 根据Referer Header头的内容分配一个值为0或1给变量$invalid\_referer | demo: referer |

代理服务器

代理类型：

1. 正向代理：服务对象是客户端，如vpn
2. 反向代理，服务对象是服务器，如代理node服务器接口

支持的协议：HTTP,HTTPS,ICMP/POP/IMAP,RTMP

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 配置名称 | context | 可选值 | 默认值 | 释义 | 备注 |
| proxy\_pass | location,if in location,limit\_except | http协议url  https协议url  socket协议url | - | 代理配置 |  |
| proxy\_buffering | http,server,location | off,on | on | 代理缓冲区 |  |
| proxy\_buffering\_size | http,server,location |  |  |  |  |
| proxy\_buffers | http,server,location |  |  |  |  |
| proxy\_busy\_buffers\_size | http,server,location |  |  |  |  |
| proxy\_redirect | http,server,location | off,default,redirect,replacement |  |  |  |
| proxy\_set\_header | http,server,location | 任意 | Host $proxy\_host;  Connection close; |  |  |
| proxy\_connect\_timeout | http,server,location | time | 60s | 代理超时时间 |  |
| proxy\_read\_timeout | http,server,location |  |  |  | 同上 |
| proxy\_send\_timeout | http,server,location |  |  |  | 同上 |
| proxy\_cache\_path | http |  | - |  |  |
| proxy\_cache | http,server,location | zone,off | off | 代理缓存 |  |
| proxy\_cache\_valid | http,server,location | time | - | 缓存过期周期 |  |
| proxy\_cache\_key | http,server,location | string | $scheme$proxy\_host$request\_uri; | 缓存维度 |  |
| proxy\_no\_cache | http,server,location |  | - |  |  |

$upstream\_cache\_status

|  |  |
| --- | --- |
| 状态 | 意义 |
| MISS | 未命中 |
| HIT | 命中 |
| EXPIRED | 缓存已过期 |
| UPDATING | 正在更新缓存 |
| STALE | 后端得到的应答 |

负载均衡服务器

upstream 配置结构

upstream name {

server backend1.example.com;

server backend2.example.com weight=5;

server backup1.example.com backup;

}

|  |  |
| --- | --- |
| 调度状态 | 释义 |
| down | 不参与负载均衡 |
| backup | 备份服务器 |
| max\_fails | 允许请求失败的次数 |
| fail\_timeout | 经过max\_fails失败后，服务暂停的时间 |
| max\_conns | 最大链接数 |

|  |  |  |
| --- | --- | --- |
| 调度算法 | 释义 | 备注 |
| 轮询 | 逐一分配到后端服务器 |  |
| 加权轮询 | weight越大，权重越高 |  |
| ip\_hash | 根据请求的ip转发的固定服务器 |  |
| url\_hash | 根据url的hash分配到不同服务器 |  |
| least\_conn | 转发到负载最少的服务器 |  |
| hash关键数值 | hash自定义的key |  |

rewrite规则

语法：rewrite regex replacement [flag];

Context: server, location, if

flag选项：

|  |  |
| --- | --- |
| last | 停止处理后续rewrite指令集，然后对当前重写的新URI在rewrite指令集上重新查找 |
| break | 停止处理后续rewrite指令集，并不在重新查找 |
| redirect | 返回302临时重定向 |
| permanent | 返回301永久重定向 |