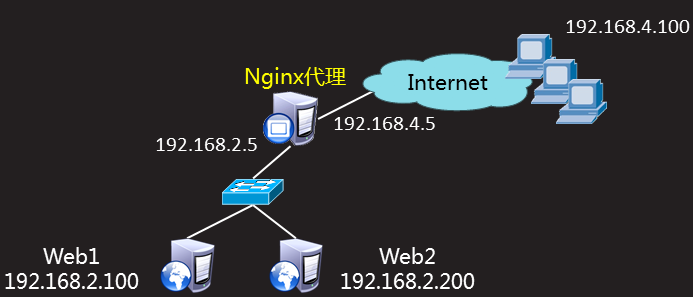
**nginx调度器**

http调度

nginx反向代理架构

实验拓扑



可以实现负载均衡

可以实现web高可用

反向代理语法格式

http {

upstream sergrp {

server 192.168.8.5:80;

server 192.168.8.6:80;

}

server{

listen 80;

server\_name [www.tarena.com;](http://www.tarena.com;)

location / {

#root html;

proxy\_pass http://sergrp;

}

}

}

nginx调度算法

nginx目前支持的调度算法

轮询（默认的）：逐一循环调度

ip\_hash：根据客户端IP分配固定的后端服务器（相同客户端访问同一服务器）

weight：指定轮询几率，权重值和访问比率成正比

fair：按后端服务器的响应时间来分配服务器，响应时间短的优先分配

url\_hash：按访问url的hash结果分配请求，使每个url定向访问到同一个后端服务器

服务器组主机状态

状态类型

down：表示当前server暂时不参与负载

max\_fails：允许请求失败的次数（默认为1，权重为几，则出现几次）

faile\_timeout：max\_fails次失败后，暂停提供服务时间（多少秒后，再次尝试）

backup：备份服务器

weight：指定轮询几率，权重值和访问比率成正比（设置权重，默认为1，权重为几，则出现几次 ）

web案例应用

http{

upstream sergrp {

#ip\_hash；

#server 192.168.8.5:80 weight=2;

server 192.168.8.5:80 down;

server 192.168.8.4:80;

server 192.168.8.6:80 backup;

server 192.168.8.3:80 max\_fails=2 fail\_timeout=30;

}

server {

listen 80;

server\_name [www.aaa.com](http://www.aaa.com)

location / {

proxy\_pass http://sergrp;

}

}

}

TCP/UDP调度

模块

ngx\_stream\_core\_module模块

使用--with-stream开启该模块

注意：

nginx从1.9版本才开始支持该功能

语法格式

stream {

upstream backend {

server backend1.example.com:12345 weight=5;

server 127.0.0.1:22 max\_fails=3 fail\_timeout=30s;

}

server {

listen 12345;

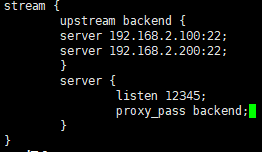
proxy\_connect\_timeout 1s;

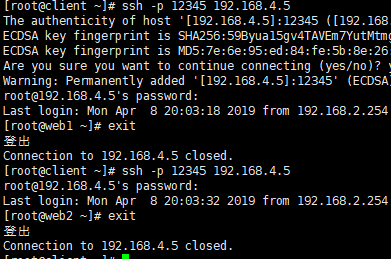
proxy\_timeout 3s;

proxy\_pass backend;

}

}



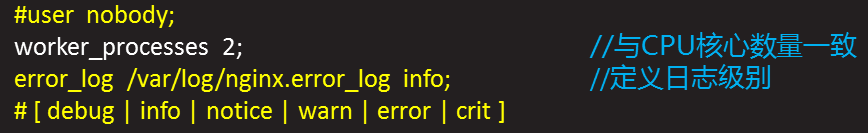


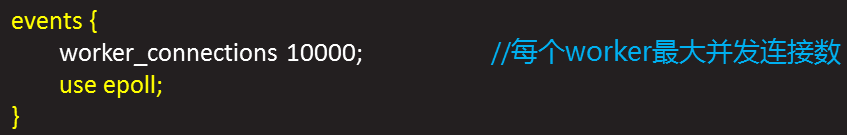
**nginx常见问题**

如何优化

多并发/连接

调整进程数量

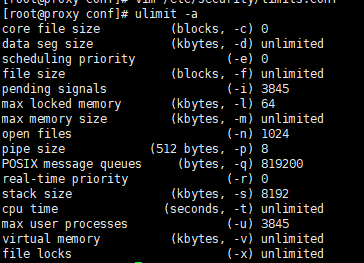




event模块优化

max\_clients=worker\_process\*worker\_connections

注意修改系统ulimit限制/etc/security/limits.conf（需重启生效）



设置软硬限制，上面是硬限制后头是软限制（临时）





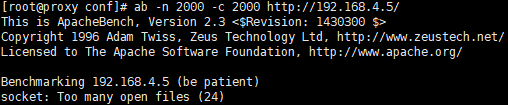
用户或组 硬限制或软限制 需要限制的项目 限制的大小

http模块优化

常见的压力测试工具

ab

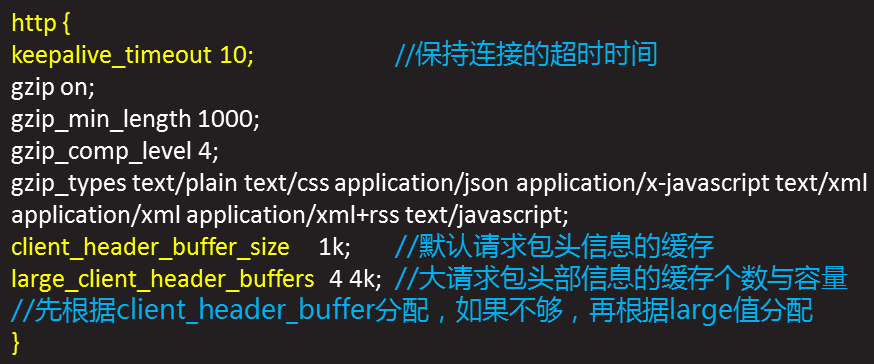
ab -c 并发数 -n 总请求数 URL

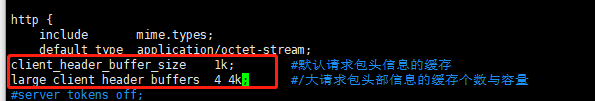


其他常见压力测试工具

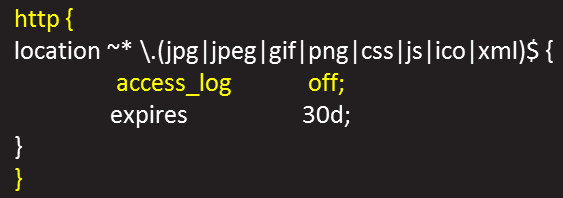
http\_load webbench siege

处理414，包头缓存不够（修改Nginx配置文件，增加数据包头缓存大小）



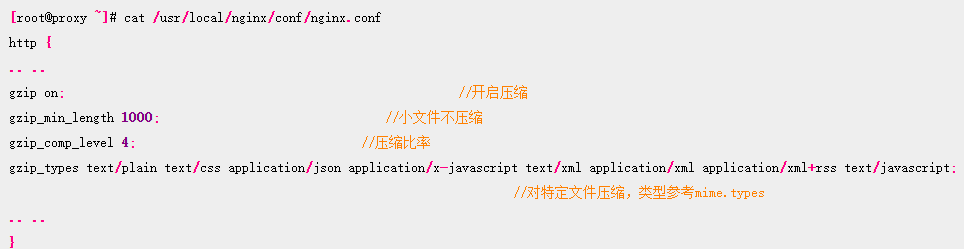


客户端浏览器缓存数据（只适合静态数据、图片、视频、音频）

缓存30天



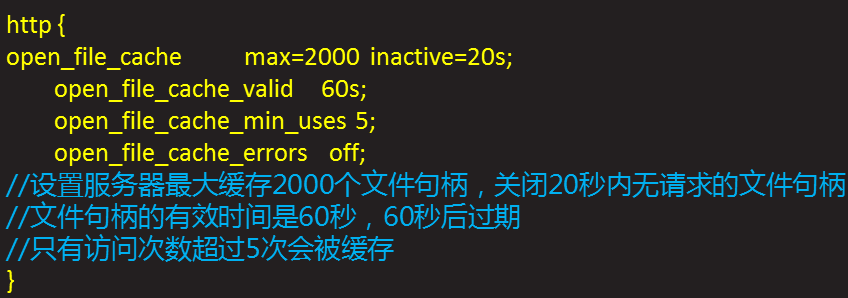
压缩（配置文件启用）



文件大小为字节，等级1-9，等级越大，效果越好，多媒体文件不要压缩，压缩适合于文档

所有浏览器支持解压工作，但不支持rar解压

如果需要处理大量的静态文件，需要保持这些文件句柄为打开状态，避免后续再次打开（打开文件缓存）

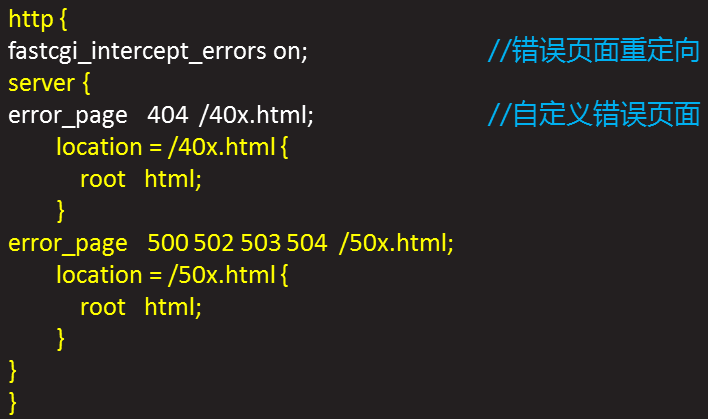


HTTP错误代码

HTTP常见错误代码列表

|  |  |
| --- | --- |
| 返回码 | 描述 |
| 200 | 一切正常 |
| 400 | 请求语法错误 |
| 401 | 访问被拒绝（帐号或密码错误） |
| 403 | 资源不可用，通常由于服务器上文件或目录的权限设置导致 |
| 403 | 禁止访问：客户端的IP地址被拒绝 |
| 404 | 无法找到指定位置的资源 |
| 414 | 请求URI头部太长 |
| 500 | 服务器内部错误 |
| 502 | 服务器作为网关或者代理时，为了完成请求访问下一个服务器，但该服务器返回了非法的应答（Bad gateway） |

nginx如何返回错误页面



修改注释，并写页面

nginx状态页面

status模块

status

--with-http\_stub\_status\_modules开启模块功能

可以查看nginx连接数等

status页面

修改nginx配置文件

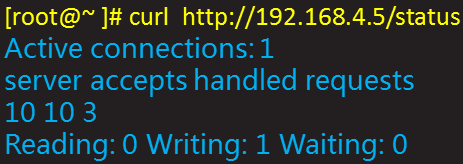
location /status{

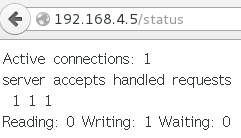
stub\_status on;

}



浏览器访问页面状态





状态信息

active connections

当前活动的连接数量

accepts

已经接受客户端的连接总数量

handled

已经处理客户端的连接总数量（一般与accepts一致，除非服务器限制了连接数量）

requests

客户端发送的请求数量

reading

当前服务器正在读取客户端请求头的数量（当前）

writing

当前服务器正在相应信息的数量（当前回应）

waiting

当前客户端在等待服务器的响应