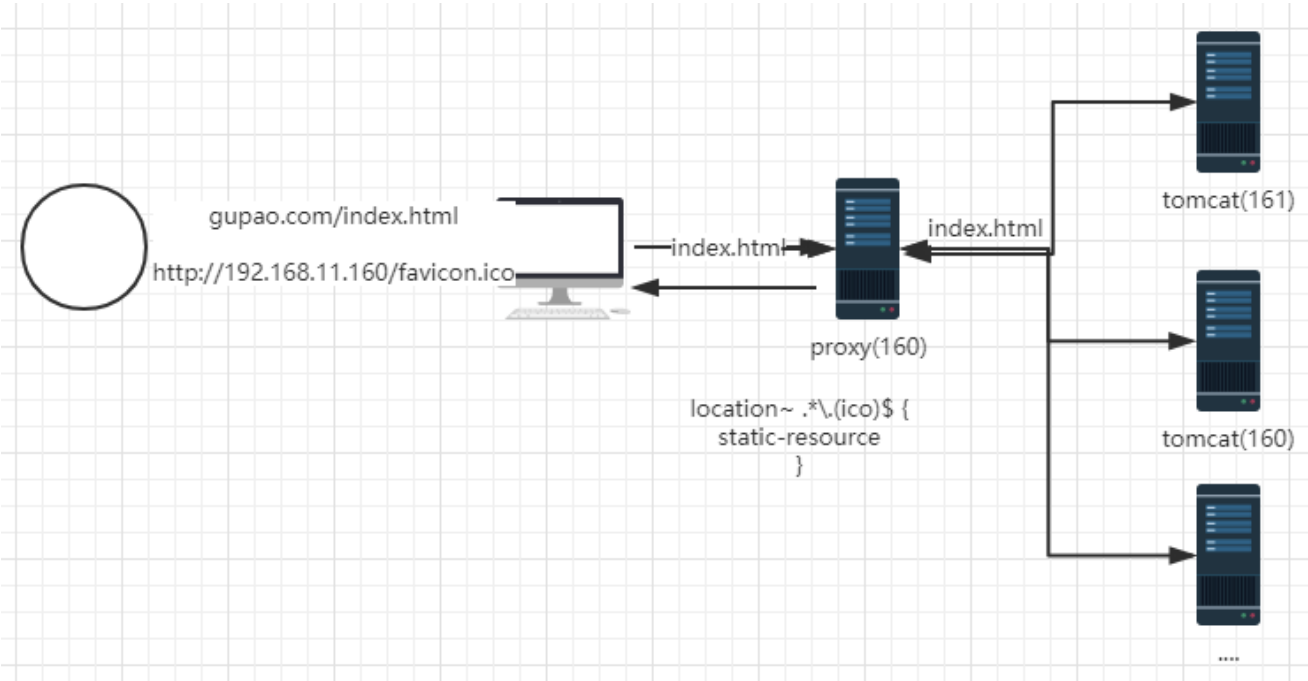
反向代理



nginx反向代理的指令不需要新增额外的模块，默认自带proxy_pass指令，只需要修改配置文件就可以实现反向代理。

proxy_pass 既可以是ip地址，也可以是域名，同时还可以指定端口

```
server {
    listen 80;
    server_name localhost;
    location / {
       proxy_pass http://192.168.11.161:8080;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```

Nginx反向代理实战

\1. 启动tomcat服务器

\2. nginx配置的统一维护，将Nginx.conf文件的内容修改成如下配置

img

\3. 在extra文件夹中添加proxy_demo.conf

server{

listen 80;

server_name localhost;

location / {

proxy_pass http://192.168.11.154:8080;

}

}

\4. ./nginx -s reload 重新加载

负载均衡

网络负载均衡的大致原理是利用一定的分配策略将网络负载平衡地分摊到网络集群的各个操作单元上，使得单个重负载任务能够分担到多个单元上并行处理，使得大量并发访问或数据流量分担到多个单元上分别处理，从而减少用户的等待响应时间

upstream

是Nginx的HTTP Upstream模块，这个模块通过一个简单的调度算法来实现客户端IP到后端服务器的负载均衡

1. Upstream

   语法： server address [paramters]

2. 负载均衡策略或者算法

   轮询算法（默认）， 如果后端服务器宕机以后，会自动踢出

   ip_hash 根据请求的ip地址进行hash

   权重轮询

演示效果

```
upstream tomcat {
  server 192.168.11.161:8080 max_fails=2 fail_timeout=60s;
  server 192.168.11.159:8080;
}

server {
    listen 80;
    server_name localhost;
    location / {
       proxy_pass http://tomcat;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
```

```
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_next_upstream error timeout http_500 http_503;
        proxy_connect_timeout 60s;
        proxy_send_timeout 60s;
        proxy_read_timeout 60s;
        add_header 'Access-Control-Allow-Origin' '*';
        add_header 'Access-Control-Allow-Methods' 'GET,POST,DELETE';
        add_header 'Aceess-Control-Allow-Header' 'Content-Type,*';
    }
    location ~ .*\.(js|css|png|svg|ico|jpg)$ {
        valid_referers none blocked 192.168.11.160 www.gupaoedu.com;
        if ($invalid_referer) {
           return 404;
        }
        root static-resource;
        expires 1d;
    }
}
```

其他配置信息

proxy_next_upstream

语法：proxy_next_upstream [error | timeout | invalid_header | http_500 | http_502 | http_503 | http_504 | http_404 | off];

默认：proxy_next_upstream error timeout;

配置块：http、server、location

这个配置表示当向一台上有服务器转发请求出现错误的时候，继续换一台上后服务器来处理这个请求。

默认情况下，上游服务器一旦开始发送响应数据，Nginx反向代理服务器会立刻把应答包转发给客户端。因此，一旦Nginx开始向客户端发送响应包，如果中途出现错误也不允许切换到下一个上有服务器继续处理的。这样做的目的是保证客户端只收到来自同一个上游服务器的应答。

proxy_connect_timeout

语法: proxy_connect_timeout time;

默认: proxy_connect_timeout 60s;

范围: http, server, location

用于设置nginx与upstream server的连接超时时间，比如我们直接在location中设置proxy_connect_timeout 1ms， 1ms很短，如果无法在指定时间建立连接，就会报错。

proxy_send_timeout

向后端写数据的超时时间，两次写操作的时间间隔如果大于这个值，也就是过了指定时间后端还没有收到数据，连接会被关闭

proxy_read_timeout

从后端读取数据的超时时间，两次读取操作的时间间隔如果大于这个值，那么nginx和后端的链接会被关闭，如果一个请求的处理时间比较长，可以把这个值设置得大一些

proxy_upstream_fail_timeout

设置了某一个upstream后端失败了指定次数（max_fails）后，在fail_timeout时间内不再去请求它,默认为10秒

语法 server address [fail_timeout=30s]

upstream backend { #服务器集群名字

#server 192.168.218.129:8080 weight=1 max_fails=2 fail_timeout=600s;

#server 192.168.218.131:8080 weight=1 max_fails=2 fail_timeout=600s;

}

Nginx动静分离

什么是动静分离

必须依赖服务器生存的我们称为动。不需要依赖容器的比如css/js或者图片等，这类就叫静

静态资源的类型

```
types {
    text/html                                        html htm shtml;
    text/css                                         css;
    text/xml                                         xml;
    image/gif                                        gif;
    image/jpeg                                       jpeg jpg;
    application/javascript                           js;
    application/atom+xml                             atom;
    application/rss+xml                              rss;

    text/mathml                                      mml;
    text/plain                                       txt;
    text/vnd.sun.j2me.app-descriptor                 jad;
    text/vnd.wap.wml                                 wml;
    text/x-component                                 htc;

    image/png                                        png;
    image/svg+xml                                    svg svgz;
    image/tiff                                       tif tiff;
    image/vnd.wap.wbmp                               wbmp;
    image/webp                                       webp;
    image/x-icon                                     ico;
```

```
    image/x-jng                                     jng;
    image/x-ms-bmp                                  bmp;

    application/font-woff                           woff;
    application/java-archive                        jar war ear;
    application/json                                json;
    application/mac-binhex40                        hqx;
    application/msword                              doc;
    application/pdf                                 pdf;
    application/postscript                          ps eps ai;
    application/rtf                                 rtf;
    application/vnd.apple.mpegurl                   m3u8;
    application/vnd.google-earth.kml+xml            kml;
    application/vnd.google-earth.kmz                kmz;
    application/vnd.ms-excel                        xls;
    application/vnd.ms-fontobject                   eot;
    application/vnd.ms-powerpoint                   ppt;
    application/vnd.oasis.opendocument.graphics     odg;
    application/vnd.oasis.opendocument.presentation  odp;
    application/vnd.oasis.opendocument.spreadsheet   ods;
    application/vnd.oasis.opendocument.text         odt;
    application/vnd.openxmlformats-officedocument.presentationml.presentation
                                                    pptx;
    application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
                                                    xlsx;
    application/vnd.openxmlformats-officedocument.wordprocessingml.document
                                                    docx;
    application/vnd.wap.wmlc                        wmlc;
    application/x-7z-compressed                     7z;
    application/x-cocoa                             cco;
    application/x-java-archive-diff                 jardiff;
    application/x-java-jnlp-file                    jnlp;
    application/x-makeself                          run;
    application/x-perl                              pl pm;
    application/x-pilot                             prc pdb;
    application/x-rar-compressed                    rar;
    application/x-redhat-package-manager            rpm;
    application/x-sea                               sea;
    application/x-shockwave-flash                   swf;
    application/x-stuffit                           sit;
    application/x-tcl                               tcl tk;
    application/x-x509-ca-cert                      der pem crt;
    application/x-xpinstall                         xpi;
    application/xhtml+xml                           xhtml;
    application/xspf+xml                            xspf;
    application/zip                                 zip;

    application/octet-stream                        bin exe dll;
    application/octet-stream                        deb;
    application/octet-stream                        dmg;
    application/octet-stream                        iso img;
    application/octet-stream                        msi msp msm;
```

```
    audio/midi                                       mid midi kar;
    audio/mpeg                                       mp3;
    audio/ogg                                        ogg;
    audio/x-m4a                                      m4a;
    audio/x-realaudio                                ra;

    video/3gpp                                       3gpp 3gp;
    video/mp2t                                       ts;
    video/mp4                                        mp4;
    video/mpeg                                       mpeg mpg;
    video/quicktime                                  mov;
    video/webm                                       webm;
    video/x-flv                                      flv;
    video/x-m4v                                      m4v;
    video/x-mng                                      mng;
    video/x-ms-asf                                   asx asf;
    video/x-ms-wmv                                   wmv;
    video/x-msvideo                                  avi;
}
```

在Nginx的conf目录下，有一个mime.types文件

用户访问一个网站，然后从服务器端获取相应的资源通过浏览器进行解析渲染最后展示给用户，而服务端可以返回各种类型的内容，比如xml、jpg、png、gif、flash、MP4、html、css等等，那么浏览器就是根据mime-type来决定用什么形式来展示的

服务器返回的资源给到浏览器时，会把媒体类型告知浏览器，这个告知的标识就是Content-Type，比如Content-Type:text/html。

演示代码

```
location ~ .*\.(js|css|png|svg|ico|jpg)$ {
       valid_referers none blocked 192.168.11.160 www.gupaoedu.com;
       if ($invalid_referer) {
          return 404;
       }
       root static-resource;
       expires 1d;
    }
```

动静分离的好处

第一个，Nginx本身就是一个高性能的静态web服务器；

第二个，其实静态文件有一个特点就是基本上变化不大，所以动静分离以后我们可以对静态文件进行缓存、或者压缩提高网站性能

缓存

当一个客户端请求web服务器, 请求的内容可以从以下几个地方获取：服务器、浏览器缓存中或缓存服务器中。这取决于服务器端输出的页面信息

浏览器缓存将文件保存在客户端，好的缓存策略可以减少对网络带宽的占用，可以提高访问速度，提高用户的体验，还可以减轻服务器的负担nginx缓存配置

Nginx缓存配置

Nginx可以通过expires设置缓存，比如我们可以针对图片做缓存，因为图片这类信息基本上不会改变。

在location中设置expires

格式： expires 30s|m|h|d

location ~ .*.(jpg|jpeg|gif|bmp|png|js|css|ico)$ {

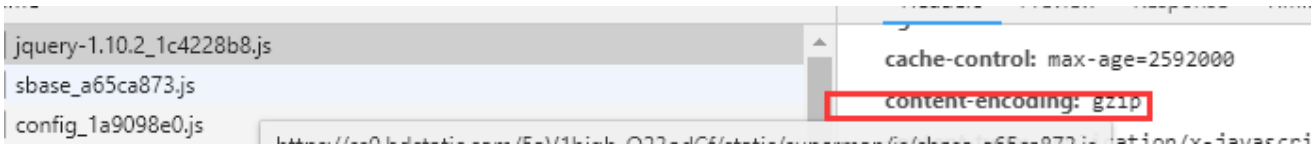
root static;

expires 1d;

}

压缩

Gzip



我们一个网站一定会包含很多的静态文件，比如图片、脚本、样式等等，而这些css/js可能本身会比较大，那么在网络传输的时候就会比较慢，从而导致网站的渲染速度。因此Nginx中提供了一种Gzip的压缩优化手段，可以对后端的文件进行压缩传输，压缩以后的好处在于能够降低文件的大小来提高传输效率 "

配置信息

Gzip on|off 是否开启gzip压缩

Gzip_buffers 4 16k #设置gzip申请内存的大小，作用是按指定大小的倍数申请内存空间。4 16k代表按照原始数据大小以16k为单位的4倍申请内存。

Gzip_comp_level[1-9] 压缩级别， 级别越高，压缩越小，但是会占用CPU资源

Gzip_disable #正则匹配UA 表示什么样的浏览器不进行gzip

Gzip_min_length #开始压缩的最小长度（小于多少就不做压缩），可以指定单位，比如 1k

Gzip_http_version 1.0|1.1 表示开始压缩的http协议版本

Gzip_proxied （nginx 做前端代理时启用该选项，表示无论后端服务器的headers头返回什么信息，都无条件启用压缩）

Gzip_type text/pliain,application/xml 对那些类型的文件做压缩 （conf/mime.conf）

Gzip_vary on|off 是否传输gzip压缩标识; 启用应答头"Vary: Accept-Encoding";给代理服务器用的，有的浏览器支持压缩，有的不支持，所以避免浪费不支持的也压缩，所以根据客户端的HTTP头来判断，是否需要压缩

演示效果

```
http {
    include       mime.types;
    default_type  application/octet-stream;

    #log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
    #                  '$status $body_bytes_sent "$http_referer" '
    #                  '"$http_user_agent" "$http_x_forwarded_for"';

    #access_log  logs/access.log  main;

    sendfile        on;
    #tcp_nopush     on;

    keepalive_timeout  60;

    include extra/*.conf;

    gzip  on;
    gzip_min_length 5k;
    gzip_comp_level 3;
    gzip_types application/javascript image/jpeg image/svg+xml;
    gzip_buffers 4 32k;
    gzip_vary on;
}
```

防盗链

一个网站上会有很多的图片，如果你不希望其他网站直接用你的图片地址访问自己的图片，或者希望对图片有版权保护。再或者不希望被第三方调用造成服务器的负载以及消耗比较多的流量问题，那么防盗链就是你必须要做的

防盗链配置

在Nginx中配置防盗链其实很简单，

语法: valid_referers none | blocked | server_names | string ...;

默认值: —

上下文: server, location

“Referer”请求头为指定值时，内嵌变量$invalid_referer被设置为空字符串，否则这个变量会被置成“1”。查找匹配时不区分大小写，其中none表示缺少referer请求头、blocked表示请求头存在，但是它的值被防火墙或者代理服务器删除、server_names表示referer请求头包含指定的虚拟主机名

\1. 配置如下

location ~ .*.(gif|jpg|ico|png|css|svg|js)$ {

valid_referers none blocked 192.168.11.153;

if ($invalid_referer) {

return 404;

}

root static;

}

需要注意的是伪造一个有效的“Referer”请求头是相当容易的，因此这个模块的预期目的不在于彻底地阻止这些非法请求，而是为了阻止由正常浏览器发出的大规模此类请求。还有一点需要注意，即使正常浏览器发送的合法请求，也可能没有“Referer”请求头。

跨域访问

什么叫跨域呢？如果两个节点的协议、域名、端口、子域名不同，那么进行的操作都是跨域的，浏览器为了安全问题都是限制跨域访问，所以跨域其实是浏览器本身的限制。

解决办法

```
修改proxy_demo.conf配置
server{
   listen 80;
   server_name localhost;
   location / {
       proxy_pass http://192.168.11.154:8080;
       proxy_set_header Host $host;
       proxy_set_header X-Real-IP $remote_addr;
       proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
       proxy_send_timeout 60s;
       proxy_read_timeout 60s;
       proxy_connect_timeout 60s;
       add_header 'Access-Control-Allow-Origin' '*';  // #允许来自所有的访问地址
       add_header 'Access-Control-Allow-Methods' 'GET,PUT,POST,DELETE,OPTIONS'; //支持的
请求方式
       add_header 'Access-Control-Allow-Header' 'Content-Type,*'; //支持的媒体类型
   }
   location ~ .*\.(gif|jpg|ico|png|css|svg|js)$ {
       root static;
   }
}
```