链滴

nginx 简介

作者：wenyl

原文链接：https://ld246.com/article/1645779551116

来源网站：链滴

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfile
com/bing/20180515.jpg?imageView2/1/w/960/h/540/interlace/1/q/100">
<h2 id="1-简介">1、简介</h2>
nginx 是一个高性能的 HTTP 和反向代理 web 服务器，同时也提供了 IMAP/POP3/SMTP 服务

<h2 id="2-正向代理和反向代理">2、正向代理和反向代理</h2>

正向代理：对服务端来说，只知道代理服务器访问它，不知道具体的客户端信息，此时客户
和代理在同一局域网下
反向代理：对客户端来说，只知道访问代理服务器，不知道具体访问了那些服务端，此时服
端和代理在同一个局域网下

<img src="https://ld246.com/images/img-loading.svg" alt="image.png" data-src="https:
/b3logfile.com/file/2022/02/image-471c8489.png?imageView2/2/interlace/1/format/jpg"></
>
<img src="https://ld246.com/images/img-loading.svg" alt="image.png" data-src="https:
/b3logfile.com/file/2022/02/image-1fdb575e.png?imageView2/2/interlace/1/format/jpg"></
>
<h2 id="3-常用命令">3、常用命令</h2>

nginx -s stop 不管当前有没有请求在处理，直接停止
nginx -s quit 完成正在处理的请求再停止
nginx -s reload 重新加载（修改了配置文件后执行此命令）

<h2 id="4-动静分离">4、动静分离</h2>
使用 nginx 可以实现动态资源和静态资源的分离
<pre><code class="highlight-chroma"><span class="highlight
cl">#拦截后台请求
location / {
  proxy_pa
s http://localhost:8080;
  proxy_se
_header X-Real-IP $remote_addr;
}

#拦截静态资源
location ~ .*\.(htm
|htm|gif|jpg|jpeg|bmp|png|ico|js|css)$ {
    root /opt/static;
}
</code></pre>
<h2 id="5-负载均衡">5、负载均衡</h2>
nginx 的负载均衡算法主要有轮询，ip_hash，权重，随机等算法，通过 upstream 模块来实现

upstream 定义了一组服务，服务可以监听不同的端口，并且服务可以是 TCP 服务或 unix 域套
字（UNIX-domain sockets）。
下面是官方给的一个定义示例：
<pre><code class="highlight-chroma"><span class="highlight
cl">upstream backend {
  server ba
kend1.example.com weight=5;
  server 1
7.0.0.1:8080 max_fails=3 fail_timeout=30s;
  server un

x:/tmp/backend3;

  server backup1.example.com backup;
}
</code></pre>
nginx 默认使用了轮询算法实现负载均衡，这里发送请求，较多的请求会落在第一个服务上，少分请求会落在第二个或者第三个服务上，第四个服务用 backup 标识，如果前面三个服务都异常，则去请求第四个服务。
<h2 id="5-1-server的参数">5.1、server 的参数</h2>
示例中 server 后面跟了一些参数除了示例中出现的，server 后面还可以添加其他参数，常用的个参数定义如下：

weight=number，给服务分配权重，number 默认为 1；
max_conns=number，标明服务的最大连接数，默认是 0，即不限制；
max_fails=number，fail_timeout=number，这两个参数需要结合使用，fail_timeout 定义一个时间周期，max_fails 表示在这个时间周期内允许的最大失败连接数，如果在这个时间周期内失的连接数超过 max_fails 设置的值，那么在这个时间周期内就不回去请求这个服务,max_fails 默认 1，tmeout 默认 10；
backup，标识这是一个备用服务，其他服务不可用就调用这个服务；
down，标识服务永远不可用；
.....

<h2 id="5-2-hash">5.2、hash</h2>
nginx 内置了基于 ketama 一致性算法的负载均衡方法，该方法基于 Memcached 缓存实现，果新增或者删除一个服务，会导致大部分缓存失效然后重新映射。常用的 url_hash 只需要把 hash 值设为请求的 url 即可，示例如下**。**
<pre><code class="highlight-chroma"> upstream urlhash {
    hash $request_uri;
    server 127.0.0.1:8080;
    server 127.0.0.1:8081;
    server 127.0.0.1:8082;
}
</code></pre>
<h2 id="5-3-ip-hash">5.3、ip_hash</h2>
ip_hash 策略将客户端的 IP 作为 hash 算法的关键字来进行请求的分配，确保从同一个客户端发的请求，会被分配到同一个服务上，除非服务不可用了，如果要删除一个服务，不能直接删除，用 don 标识，如果直接删除，那么就会重新对 ip 进行 hash 操作，之前的请求就不一定能落在原来的服务。
<pre><code class="highlight-chroma">upstream backend {
  ip_hash;
  server backend1.example.com;
  server backend2.example.com;
  server backend3.example.com down;
  server ba

kend4.example.com;
}
</code></pre>
<h2 id="5-4-加权轮询">5.4、加权轮询</h2>
通过 weight=number 实现
<pre><code class="highlight-chroma"><span class="highlight
cl">upstream backend {
    server ba
kend1.example.com weight=5;
    server ba
kend2.example.com weight=3;
    server ba
kend3.example.com weight=2;
}
</code></pre>