NGINX的反向代理与负载均衡

惨绿少年 Linux运维, Web应用, 运维基本功 0评论 来源：本站原创 74℃ 字体： 小 中 大

1.1 集群是什么

简单地说，集群就是指一组（若干个）相互独立的计算机，利用高速通信网络组成的一个较大的计算机服务系统，每个集群节点（即集群中的每台计算机）都是运行各自服务的独立服器。这些服务器之间可以彼此通信，协同向用户提供应用程序、系统资源和数据，并以单一系统的模式加以管理。当用户客户机请求集群系统时，集群给用户的感觉就是一个单一独立的服务器，而实际上用户请求的是一组集群服务器。

打开谷歌、百度的页面，看起来好简单，也许你觉得用几分钟就可以制作出相似的网页，而实际上，这个页面的背后是由成千上万台服务器集群协同工作的结果。而这么多的服务器维护和管理，以及相互协调工作也许就是读者你未来的工作职责了。

若要用一句话描述集群，即一堆服务器合作做同一件事，这些机器可能需要整个技术团队架构、设计和统一协调管理，这些机器可以分布在一个机房，也可以分布在全国全球各个地区的多个机房。

1.2 为什么要有集群

高性能、价格有效性、可伸缩性、高可用性

透明性、可管理性、可编辑性

1.2.1 集群种类

负载均衡集群 LB 解决调度问题

高可用集群 HA 解决单点故障问题（keeplived）

高性能计算集群 HP 、网络计算集群 GC

1.2.2 硬件设备

F5 设备 A10

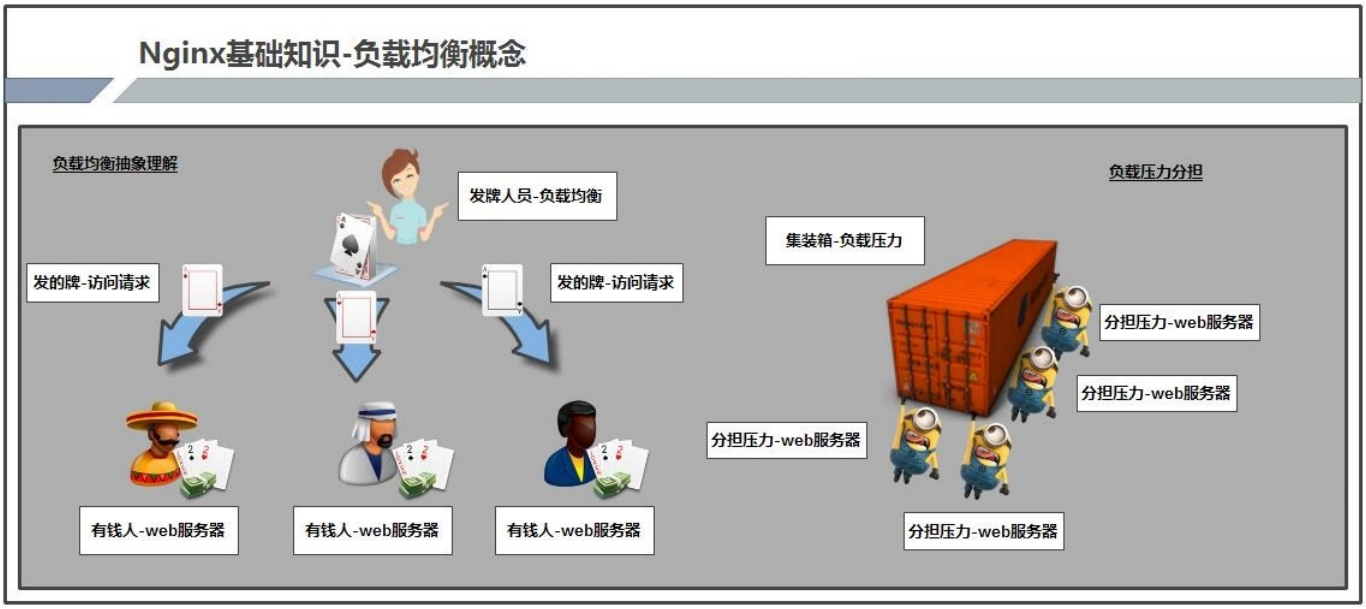
1.2.3 软件

nginx （7层 1.9版本之后支持4层）、LVS （4层）、HAproxy （4层 7层）

1.2.4 负载均衡概念说明

对用户的访问请求进行调度管理

对用户的访问请求进行压力分担



1.2.5 反向代理

接收用户请求代替用户向后端访问

反向代理与数据转发的区别

1.2.6 压力测试的方式

ab （apache里的命令）

通过 yum install httpd-tools 获得

1.3 nginx反向代理实践

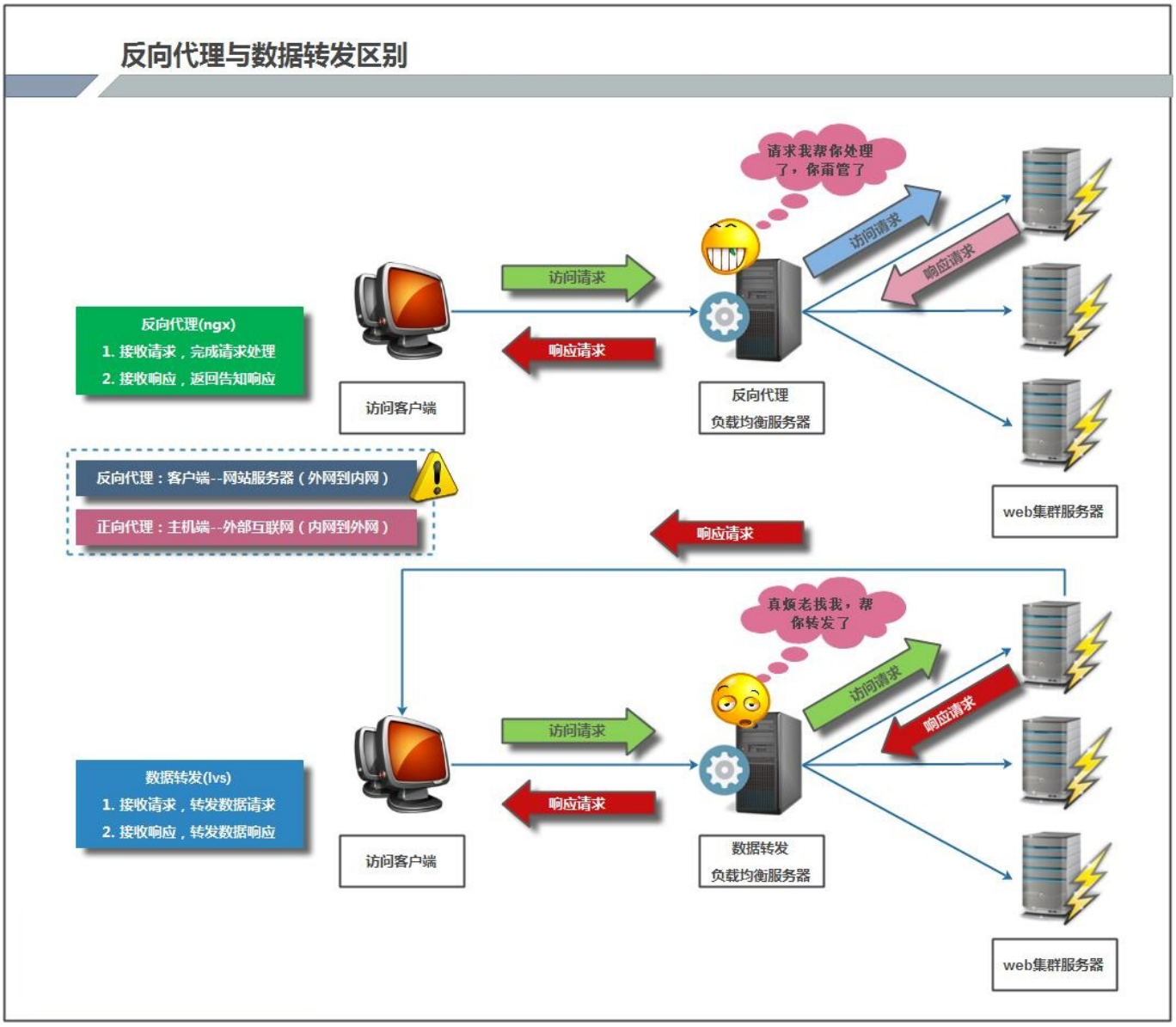
1.3.1 地址规划说明

HOSTNAME	IP	说明
lb01	10.0.0.5	Nginx主负载服务器
lb02	10.0.0.6	nginx辅负载服务器
web01	10.0.0.8	web01服务器
web02	10.0.0.7	web02服务器
web03	10.0.0.9	web03服务器
说明：以上为实际生产架构负载实现规划内容		

ip命令说明

```
ip address show  查看ip地址
ip route show  查看路由信息
```

1.3.2 反向代理与数据转发的区别



1.3.3 安装部署nginx过程（安装命令集）

```
yum install -y pcre-devel openssl-devel
mkdir -p /server/tools
cd /server/tools
wget -q http://nginx.org/download/nginx-1.10.3.tar.gz
ls -l nginx-1.10.3.tar.gz
useradd www -s /sbin/nologin -M
tar xf nginx-1.10.3.tar.gz
cd nginx-1.10.3
./configure  --user=nginx --group=nginx --prefix=/application/nginx-1.10.3 --with-http_stub_stat
make
make install
ln -s /application/nginx-1.10.3 /application/ngin
```

1.3.4 编写nginx配置文件（统一web服务器配置）

```
worker_processes  1;
events {
    worker_connections  1024;
```

```
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';
    server {
        listen       80;
        server_name  bbs.etiantian.org;
        location / {
            root   html/bbs;
            index  index.html index.htm;
        }
      access_log  logs/access_bbs.log  main;
    }
    server {
        listen       80;
        server_name  www.etiantian.org;
        location / {
            root   html/www;
            index  index.html index.htm;
        }
        access_log  logs/access_www.log  main;
    }
}
```

1.3.5 统一nginx测试环境 （web文件）

```
mkdir -p /application/nginx/html/{www,bbs}
for name in www bbs; do echo $name `hostname` >/application/nginx/html/$name/xiaoxinxin.html;dor
for name in www bbs; do cat /application/nginx/html/$name/xiaoxinxin.html;done
```

1.3.6 测试

```
[root@lb01 ~]# curl -H host:bbs.etiantian.org  10.0.0.8/xiaoxinxin.html
bbs web01
[root@lb01 ~]# curl -H host:bbs.etiantian.org  10.0.0.7/xiaoxinxin.html
bbs web02
[root@lb01 ~]# curl -H host:bbs.etiantian.org  10.0.0.9/xiaoxinxin.html
bbs web03
[root@lb01 ~]# curl -H host:www.etiantian.org  10.0.0.8/xiaoxinxin.html
www web01
[root@lb01 ~]# curl -H host:www.etiantian.org  10.0.0.7/xiaoxinxin.html
www web02
[root@lb01 ~]# curl -H host:www.etiantian.org  10.0.0.9/xiaoxinxin.html
www web03
```

1.3.7 配置负载服务文件

```
worker_processes  1;
events {
```

```
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    upstream server_pools {
        server 10.0.0.7:80;
        server 10.0.0.8:80;
        server 10.0.0.9:80;
    }
    server {
        listen 80;
        server_name bbs.etiantian.org;
        location / {
            proxy_pass http://server_pools;
        }
    }
```

1.3.8 测试访问

```
[root@lb01 conf]# curl -H host:bbs.etiantian.org 10.0.0.5/xiaoxinxin.html
bbs web03
[root@lb01 conf]# curl -H host:bbs.etiantian.org 10.0.0.5/xiaoxinxin.html
bbs web02
[root@lb01 conf]# curl -H host:bbs.etiantian.org 10.0.0.5/xiaoxinxin.html
bbs web01
```

1.4 nginx中常用模块说明

```
ngx_http_status_module
ngx_http_ssl_module
ngx_http_log_module
ngx_http_upstream_module
ngx_http_proxy_module
```

1.4.1 模块调度算法

①. 定义轮询调度算法-rr-默认调度算法

②. 定义权重调度算法-wrr

③. 定义静态调度算法-ip_hash

④. 定义最小的连接数-least_conn

1.4.2 nginx反向代理相关两个模块

upstream 模块 类似与一个池塘，将nginx节点放置到池塘中

proxy模块 用池塘里面的nginx节点，利用pr oxy进行调用

1.4.3 upstream模块核心参数简介

weight 权重

max_fails 抛得次数

fail_timeout 失败的超时时间

backup 备份

1.4.4 weight 参数实践 （权重）

upstream 模块只能在http区块里

```
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    upstream server_pools{
         server 10.0.0.7:80 weight=1;
         server 10.0.0.8:80 weight=2;
    }
    server{
       listen 80;
       server_name bbs.etiantian.org;
       location / {
          proxy_pass http://server_pools;
       }
    }
}
```

测试

```
[root@test tools]# curl 10.0.0.5
web01 www
[root@test tools]# curl 10.0.0.5
web01 www
[root@test tools]# curl 10.0.0.5
web02 www
[root@test tools]# curl 10.0.0.5
web01 www
[root@test tools]# curl 10.0.0.5
web01 www
[root@test tools]# curl 10.0.0.5
web02 www
```

1.4.5 其他的参数说明

max_fails 失败的尝试次数

fail_timeout 失败后的再次尝试时间

backup 备份节点：所有的节点都挂掉后数据才会请求web01

```
server 10.0.0.7:80 weight=1 max_fails=3 fail_timeout=10 ;
server 10.0.0.8:80 weight=2 max_fails=3 fail_timeout=10 backup;
```

测试，将web02停掉

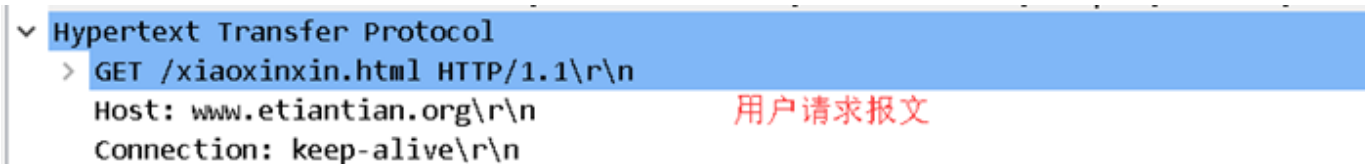
```
[root@test tools]# curl 10.0.0.5
web02 www
[root@test tools]# curl 10.0.0.5
web02 www
```

停掉web02后

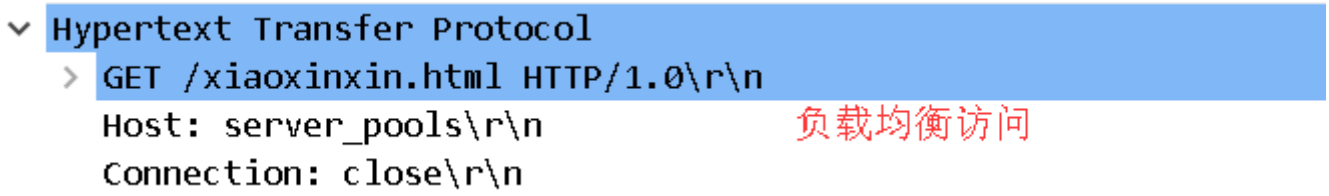
```
[root@test tools]# curl 10.0.0.5
web01 www
[root@test tools]# curl 10.0.0.5
web01 www
[root@test tools]# curl 10.0.0.5
```

1.4.6 访问抓包

用户请求报文



负载均衡请求报文



说明:

hosts 主机头不同，未配置proxy_set_header Host $host 参数，在负载均衡访问的时候会不带hosts信息。

1.4.7 upsrteam参数详细说明

upstream模块内参数	参数说明
server 10.0.10.8:80	负载均衡后面的RS配置，可以是IP或域名，如果端口不写，默认是80端口。高并发场景下， IP可换成域名，通过 DNS做负载均衡。
weigth=1	代表服务器的权重，默认值是1。权重数字越大表示接受的请求比例越大。
max_fails=3	Nginx尝试连接后端主机失败的次数，这个值是配合proxy_next_upstream、fastcgi_next_upstream和memcached_next_upstream这三个参数来使用的。当nginx接收后端服务器返回这三个参数定义的状态码时，会将这个请求转发给正常工作的后端服务器，例如404、502、503、 Max_fails的默认值是1；

	企业场景下建议2-3次。如京东1次，蓝汛10次，根据业务需求去配置
fail_timeout=10s	在max_fails定义的失败次数后，距离下次检查的间隔时间，默认是10s ;如果max_fails是5 ,它就检测5次，如果5次都是502,那么，它就会根据fail_timeout的值，等待10s再去检查，还是只检查一次，如果持续502,在不重新加载 Nginx 配置的情况下，每隔10s都只检查一次。常规业务2~3秒比较合理，比如京东3秒，蓝汛3秒，可根据业务需求去配置。
backup	热备配置（RS节点的高可用），当前面激活的RS都失败后会自动启用热备RS这标志看这个服务器作为备份服务器，若主服务器全部宕机了，就会向它转发请求。 注意：当负载调度算法为ip_hash时，后端服务器在负载均衡调度中的状态不能是weight和backup。
down	这标志着服务器永远不可用，这个参数可配合ip_hash使用；类似与注释。

weight :调节服务器的请求分配权重。1.4.8 上述命令的说明如下：

- check :开启对该服务器健康检查。
- inter:设置连续两次的健康检查间隔时间，单位毫秒，默认值2000。
- rise :指定多少次连续成功的健康检查后，即可认定该服务器处于可用状态。
- fall :指定多少次不成功的健康检查后，即认为服务器为宕机状态，默认值3。
- maxconn :指定可被发送到该服务器的最大并发连接数。

> 虽然Nginx本身不支持一致性hash算法，但Nginx的分支Tengine支持。详细可见 http://tengine.taobao.org/document_cn/http_upstream_consistent_hash_cn.html

1.4.9 ip_hash 参数实践

每个访问的用户都会生成一个hash值。

每个请求按客户端 IP的 hash结果分配，当新的请求到达时，先将其客户端 IP通过哈希算法哈希出一个值，在随后的客户端请求中，客户IP的哈希值只要相同，就会被分配至同一台服务器，该调度算法可以解决动态网页的session共享问题，但有时会导致请求分配不均，即无法保证1:1的负载均衡，因为在国内大多数公司都是NAT上网横式，多个客户端会对应_个外部IP ,所以，这些客户端都会被分配到同一节点服务器，从而导致请求分配不均。

LVS负载均衡的-p参数、Keepalived配置里的 persistence jimeout 50参数都类似这个 Nginx里的ip_hash参数，其功能都可以解决动态网页的session共享问题。

```
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    upstream server_pools{
            ip_hash;
```

```
        server 10.0.0.7:80;
        server 10.0.0.8:80;
    }
    server{
       listen 80;
       server_name bbs.etiantian.org;
       location / {
          proxy_pass http://server_pools;
       }
    }
}
```

1.4.10 least_conn 参数

看谁闲，谁闲发送给谁

least_conn算法会根据后端节点的连接数来决定分配情况，哪个机器连接数少就分发。

1.4.11 fair 参数

看谁响应的快

此算法会根据后端节点服务器的响应时间来分配请求,响应时间短的优先分配。这是更加智能的调度算法。此种算法可以依据页面大小和加载时间长短智能地进行负载均衡，也就是根据后端服务器的响应时间来分配请求，响应时间短的优先分配。Nginx本身不支持fair调度算法，如果需要使用这种调度算法，必须下载Nginx的相关模块upstream_fair。

示例如下:

```
upstream name {
server 192.168.1.1;
server 192.168.1.2;
fair;
}
```

除了上面这些算法外，还有一些第三方调度算法，例如：url_hash、一致性hash算法等.

1.4.12 调度算法

定义轮询调度算法 rr 默认调度算法　平均分配

定义权重调度算法 wrr

定义静态调度算法 ip-hash

定义最小的连接数-least_conn

1.4.13 nginx负载均衡相关重要参数

Nginx反向代理重要参数	解释说明
proxy.pass http://server_pools;	通过proxy_pass功能把用户的清求转向到反向代理定义的upstream服务器池

proxy_set_header Host $host	在代理向后端服务器发送的 http 请求头中加入host字段信息，用于当后端服务器配置有多个虚拟主机时，可以识别代理的是哪个虚拟主机。这是节点服务器多虚拟主机时的关键配置
proxy_set_header X-ForwardedFor $remote_addr;	在代理向后端服务器发送的 http请求头中加入 X-Forward-For字段信息，用于后端服务器程序、日志等接收记录真实用户的 IP ,而不是代理服务器的 IP这是反向代理时，节点服务器获取用户真实IP的必要功能配置

1.4.14 反向代理排错思路

01.先在lb01上访问后端节点进行测试

02.在lb01上访问本地地址进行测试

03.在浏览器上进行测试

缓存、域名解析

1.4.15 proxy_next_uptream 参数

当nginx接收后端服务器返回proxy_next_upstream 参数定义的状态码时，会将这个请求转发给正常工作的后端服务器，例如500，502，503，504，此参数可以提升用户的访问体验。

1.5 定义多个虚拟主机标签信息

1.5.1 proxy_set_header 参数

配置文件

```
[root@lb01 conf]# cat nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    upstream server_pools{
         server 10.0.0.7:80;
         server 10.0.0.8:80;
    }
    server{
       listen 80;
       server_name bbs.etiantian.org;
       location / {
          proxy_pass http://server_pools;
              proxy_set_header Host $host;
       }
    }
    server{
```

```
        listen 80;
        server_name www.etiantian.org;
        location / {
           proxy_pass http://server_pools;
                proxy_set_header Host $host;
        }
    }

}
```

访问抓包

该参数在负载服务器在访问后端服务器的时候 会带上hosts信息。

```
Hypertext Transfer Protocol
  GET /xiaoxinxin.html HTTP/1.0\r\n
  Host: www.etiantian.org\r\n
  Connection: close\r\n
```

1.5.2 X-Forwarded-For 参数

proxy_set_header X-Forwarded-For $remote_addr;

代理的啥时候在后面显示真实的IP地址

```
[root@lb01 conf]# cat nginx.conf
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    upstream server_pools{
         server 10.0.0.8:80;
         server 10.0.0.7:80;
         server 10.0.0.9:80;
    }
    server{
       listen 80;
       server_name bbs.etiantian.org;
       location / {
          proxy_pass http://server_pools;
             proxy_set_header Host $host;
          proxy_set_header X-Forwarded-For $remote_addr;
       }
    }
    server{
       listen 80;
       server_name www.etiantian.org;
       location / {
          proxy_pass http://server_pools;
             proxy_set_header Host $host;
```

```
            proxy_set_header X-Forwarded-For $remote_addr;
        }
    }
    server{
        listen 80;
        server_name blog.etiantian.org;
        location / {
            proxy_pass http://server_pools;
            proxy_set_header Host $host;
            proxy_set_header X-Forwarded-For $remote_addr;
        }
    }

}
```

参看日志信息

```
1 10.0.0.5 - - [30/Oct/2017:12:36:10 +0800] "GET / HTTP/1.0" 200 10 "-" "Mozilla/5.0 (Windows NT
2 10.0.0.5 - - [30/Oct/2017:12:36:10 +0800] "GET /favicon.ico HTTP/1.0" 404 571 "http://www.etia
```

该参数配置，会在访问日志的后面加上真正的访问用户ip

1.5.3 http proxy 模块相关参数说明

http proxy 模块相关参数	说明
proxy_set_header	设置http请求header项传给后端服务器节点，例如：可实现让代理后端的服务器节点获取访问客户端用户的真实IP地址
client_body_buffer_size	用于指定客户端请求主体缓冲区大小
proxy_connect_timeout	表示反向代理后端节点服务器连接的超时时间，即发起握手等候响应的超时时间
proxy_send_timeout	表示代理后端服务器的数据回传时间，即在规定时间内后端服务器必须传完所有数据，否则nginx将断开这个连接
proxy_read_timeout	设置nginx从代理的后端服务器获取信息的时间，表示连接建立成功后，nginx等待后端服务器的响应时间，其实是nginx已经进入后端的排队之中等候处理的时间
proxy_buffer_size	设置缓冲区大小，默认该缓冲区大小等于指令proxy_buffers设置的大小
proxy_buffers	设置缓冲区的数量和大小，nginx从代理的后端服务器获取的响应信息，会设置到缓冲区
proxy_busy_buffers_size	用于设置相同很忙时可以使用的proxy_buffers大小，官方推荐的大小为proxy_buffers * 2
proxy_trmp_file_write_size	指定proxy缓存临时文件的大小

1.6.1 相据URL目录地址转发的应用场景1.6 基于目录（uri）进行转发–网站动静分离

根据HTTP的URL进行转发的应用情况，被称为第7层（应用层）的负载均衡，而LVS的负载均衡一般用于TCP等的转发，因此被称为第4层（传输层）的负载均衡。

在企业中，有时希望只用一个域名对外提供服务，不希望使用多个域名对应同一个产品业务，此时就需要在代理服务器上通过配置规则，使得匹配不同规则的请求会交给不同的服务器池处理。这类业务有：

☑ 业务的域名没有拆分或者不希望拆分，但希望实现动静分离、多业务分离，

☑ 不同的客户端设备（例如：手机和 PC端）使用同一个域名访问同一个业务网站，就需要根 据规则将不同设备的用户请求交给后端不同的服务器处理，以便得到最佳用户体验。

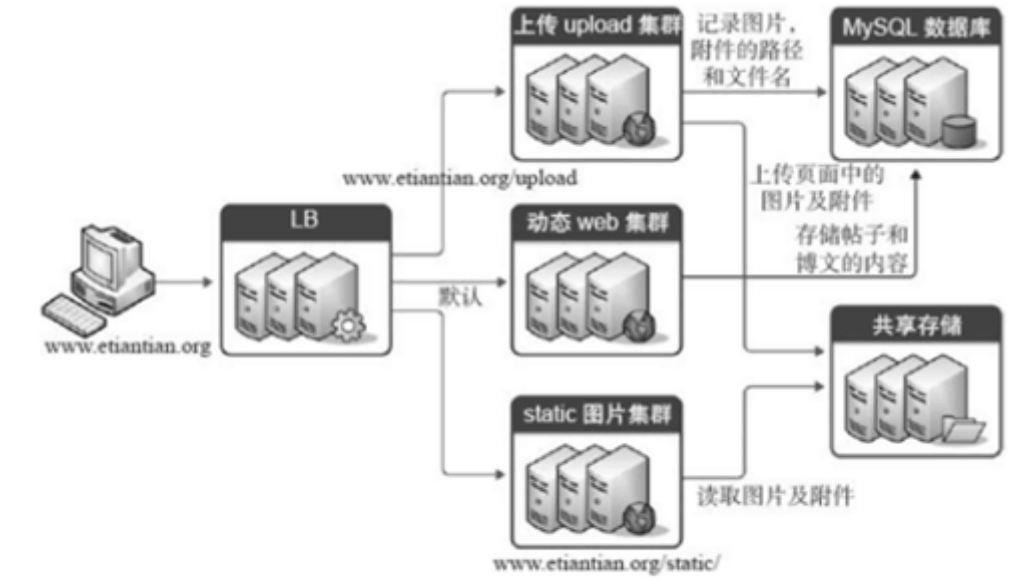


图 9-10 动静分离网站集群架构

1.6.2 第一个里程碑： 服务器规划

目录（uri）	ip	服务器目录	类型
/upload	10.0.0.8:80	html/www/upload	upload服务器
/static	10.0.0.7:80	html/www/static	static静态服务器
/	10.0.0.9:80	html/www	默认

1.6.3 第二个里程碑：创建/设置upstream负载信息

```
upstream upload_pools {
  server 10.0.0.8:80;
}
upstream static_pools {
  server 10.0.0.7:80;
}
upstream default_pools {
  server 10.0.0.9:80;
}
```

1.6.4 第三个里程碑：如何调用upstream信息

```
location /static/ {
    proxy_pass http://static_pools;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $remote_addr;
}
 location /upload/ {
    proxy_pass http://upload_pools;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $remote_addr;
}
 location / {
    proxy_pass http://default_pools;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-For $remote_addr;
}
```

1.6.5 第四个里程碑： 编写配置文件lb01

```
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';
    upstream upload_pools {
      server 10.0.0.8:80;
    }

    upstream static_pools {
      server 10.0.0.7:80;
    }

    upstream default_pools {
      server 10.0.0.9:80;
    }

    server {
        listen 80;
        server_name www.etiantian.org;
    location /static/ {
        proxy_pass http://static_pools;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
    }

        location /upload/ {
            proxy_pass http://upload_pools;
        proxy_set_header Host $host;
```

```
        proxy_set_header X-Forwarded-For $remote_addr;
    }

     location / {
            proxy_pass http://default_pools;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
    }
         access_log  logs/access_www.log  main;
    }
}
```

1.6.6 第五个里程碑： 创建环境

```
www.etiantian.org/xxx.html
www.etiantian.org/upload/xxx.html
www.etiantian.org/static/xxx.html
```

##web01

```
mkdir -p /application/nginx/html/www/upload
echo  "web01 upload" >/application/nginx/html/www/upload/xxx.html
```

##web02

```
mkdir -p /application/nginx/html/www/static
echo  "web02 static" >/application/nginx/html/www/static/xxx.html
```

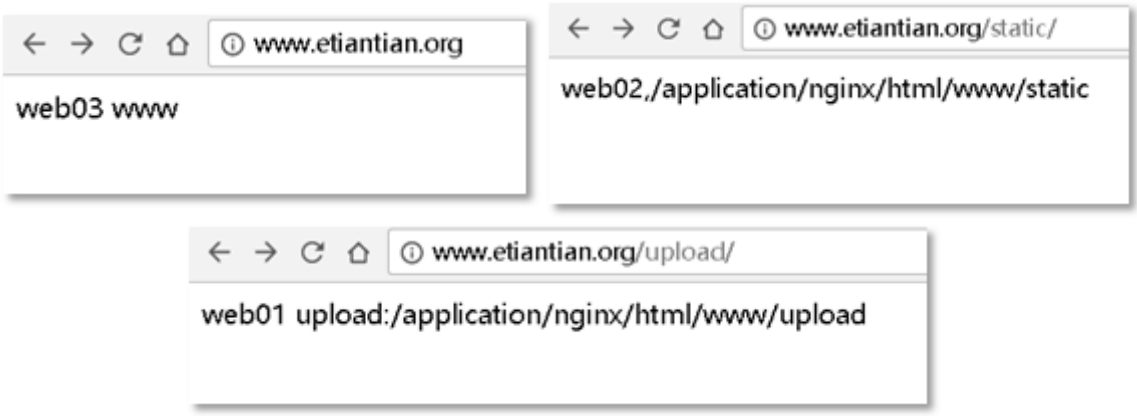
##web03

```
echo  "web03 default" >/application/nginx/html/www/xxx.html
```

1.6.7 第六个里程碑： 进行测试

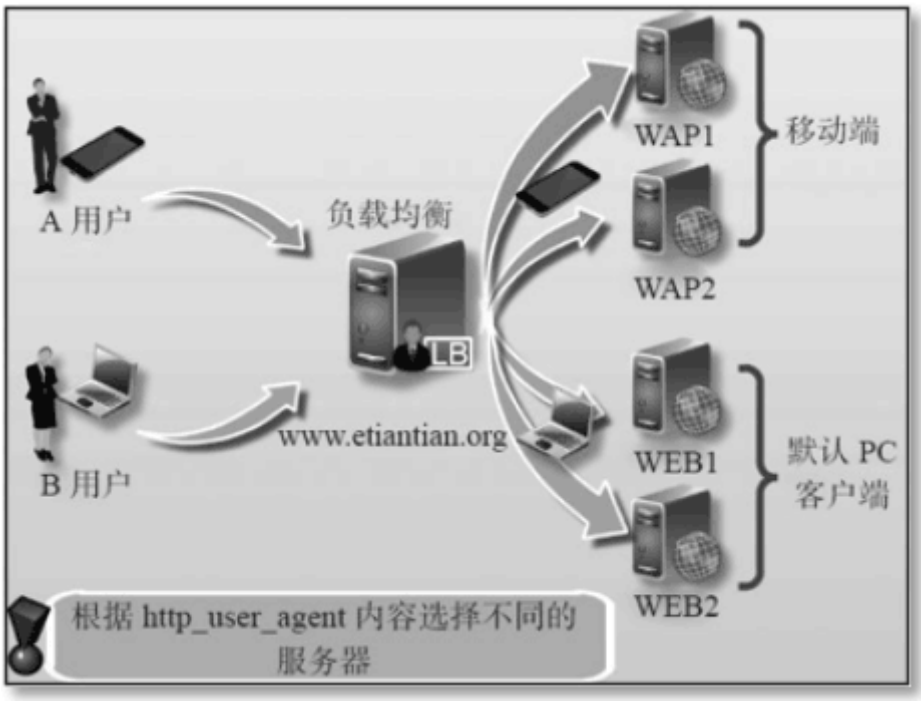
```
[root@lb01 conf]# curl -H  host:www.etiantian.org  10.0.0.5/upload/
web01 upload:/application/nginx/html/www/upload
[root@lb01 conf]# curl -H  host:www.etiantian.org  10.0.0.5/static/
web02,/application/nginx/html/www/static
[root@lb01 conf]# curl -H  host:www.etiantian.org  10.0.0.5/
web03 www
```

浏览器进行访问测试



1.7 根据客户端的设备实现转发（user_agent）

1.7.1 user_agent的应用



1.7.2 修改lb01配置文件

```
worker_processes  1;
events {
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    upstream upload_pools {
      server 10.0.0.8:80;
    }
```

```
17
18     upstream static_pools {
19       server 10.0.0.7:80;
20     }
21
22     upstream default_pools {
23       server 10.0.0.9:80;
24     }
25
26     server {
27         listen 80;
28         server_name www.etiantian.org;
29         location / {
30          if ($http_user_agent ~* "MSIE")
31           {
32             proxy_pass http://static_pools;
33           }
34          if ($http_user_agent ~* "Chrome")
35           {
36             proxy_pass http://upload_pools;
37           }
38         proxy_pass http://default_pools;
39            proxy_set_header Host $host;
40                }
41          access_log  logs/access_www.log  main;
42     }
43 }
```

View Code lb01配置文件

1.7.3 进行测试

基于上一步的操作，现在可以之间的进行访问测试

curl -A 指定访问类型

```
[root@lb01 conf]# curl -A MSIE -H host:www.etiantian.org 10.0.0.5
web02 www
[root@lb01 conf]# curl -A Chrome -H host:www.etiantian.org 10.0.0.5
web01 www
[root@lb01 conf]# curl -A xx -H host:www.etiantian.org 10.0.0.5
web03 www
```

1.8 利用扩展名进行转发

利用后缀名进行转发与nginx中的基于后缀的转跳一样实现。

```
location ~.*.(gif|ipg|jpeg|png|bmp|swf|css|js)$ {
        peoxy_pass http://static_pools;
           include proxy.conf
}
```

赞1

如无特殊说明，文章均为本站原创，转载请注明出处

- 转载请注明来源：Nginx的反向代理与负载均衡
- 本文永久链接地址：https://www.nmtui.com/clsn/lx429.html

该文章由 惨绿少年 发布

惨绿少年Linux www.nmtui.com