**Nginx+Tomcat实现反向代理及动静分离**

**通常tomcat部署结构**

通常tomcat前端是nginx或apache，后端都为tomcat，也就意味着无论前端是什么角色都是以代理的方式进行工作的

但是要注意的是如果基于nginx做反向代理，转发请求到tomcat的时候是基于http协议进行转发的

但注意的是tomcat的连接器有httpajp jk2 jserv

而如果基于nginx做转发的话只支持http做转发

而如果apache做代理转发的话，几乎常用协议都支持

但常用的连接类型都是ajp协议，因为ajp协议可以工作在二进制模式下，而http协议都是工作在文本模式下，所以ajp的效率要比http要高效

因此要部署一个tomcat前端web服务器很多时候可以使用apache

**使用nginx反向代理用户请求至tomcat**

规划如下：

|  |  |
| --- | --- |
| 服务器角色 | 服务器IP |
| Nginx | 10.0.10.62 |
| Tomcat | 10.0.10.61 |

**配置Tomcat**

还原配置文件并修改配置文件

[root@node1 conf]#catalina.sh stop

[root@node1 conf]#mv server.xml server.xml.bak2  
[root@node1 conf]# cp server.xml.bak server.xml

加入用户

[root@node1 conf]#vim tomcat-users.xml

  <rolerolename="manager-gui"/>                                                     #定义管理组，分别是信息页面以及管理页面  
  <role rolename="admin-gui"/>

  <userusername="tomcat" password="tomcat"roles="manager-gui,admin-gui"/>           #将用户tomcat加入至这两个组内

创建网站目录

[root@node1 conf]#mkdir /tomcat/app1

创建host主机

[root@node1 conf]#vim server.xml

加入参数

     <Host name="www.test.com"  appBase="/tomcat/app1"  
           unpackWARs="true"autoDeploy="true">

              <ValveclassName="org.apache.catalina.valves.AccessLogValve"directory="logs"

              <ValveclassName="org.apache.catalina.valves.AccessLogValve"directory="logs"

     <Context path="/"docBase="/tomcat/app1" />

         <ValveclassName="org.apache.catalina.valves.AccessLogValve"directory="logs"

          prefix="www\_access\_log." suffix=".txt"  
            pattern="%h %l %u %t &quot;%r&quot; %s %b"/>

     </Host>

#path相当于uri的路径

更改默认主机

   <Engine name="Catalina" defaultHost="www.test.com">

更改端口

   <Connector port="80" protocol="HTTP/1.1"

启动服务

[root@node1 conf]#catalina.sh start

部署网站程序，步骤略，访问ip后返回如下页面

[](http://s3.51cto.com/wyfs02/M00/39/80/wKioL1O6SmnT-9blAASiu6_pSzE799.jpg)

访问测试无误，继续之后的配置

**配置Nginx**

[root@node2nginx]# pwd  
/etc/nginx

编辑配置文件，将所有的请求都代理至后端tomcat

[root@node2nginx]# vim nginx.conf

       location / {  
            proxy\_pass [http://10.0.10.61:80](http://10.0.10.61/);

       }

保存退出，并检查语法

[root@node2nginx]# /usr/local/nginx/sbin/nginx -t

nginx: theconfiguration file /etc/nginx/nginx.conf syntax is ok

nginx:configuration file /etc/nginx/nginx.conf test is successful

访问测试

[](http://s3.51cto.com/wyfs02/M01/39/80/wKioL1O6SoDjQJ_gAAOCzXFIzr4122.jpg)

确保代理无误

**将动态内容重定向至tomcat**

再次编辑配置文件

[root@node2nginx]# vim nginx.conf

改为以下参数，凡是以jsp等动态页面的请求统统转发至10.0.10.61的80端口上

        location / {

            root /usr/shar/nginx/html;

            index index.html;

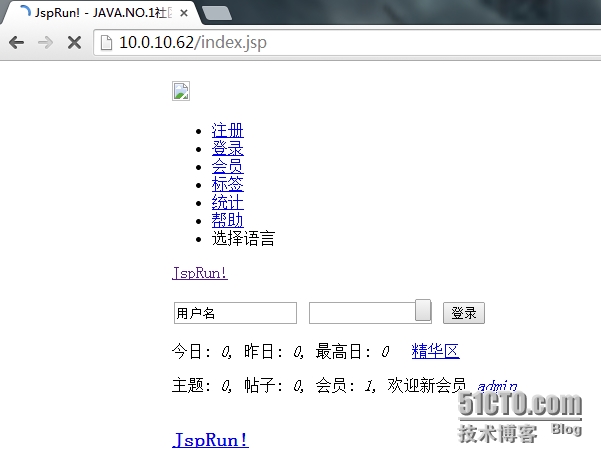
        }

        location ~ \.(jsp|do)$ {

            proxy\_pass http://10.0.10.61:80;

        }

访问测试，结果如下：

[](http://s3.51cto.com/wyfs02/M00/39/80/wKiom1O6SsKBneL8AAEldm1MpCQ989.jpg)

因为只将jsp跳转过来但是图片找不到，这时我们需要使用nginx来响应用户的静态请求

**实现动静分离**

将图片复制到nginx本地目录，这里我们为了方便并且为了目录的一致性，将其源码全部拷贝至nginx节点

[root@node2nginx]# mkdir -p /web/html

[root@node1tomcat]# scp -r app1 root@node2:/web/html

编辑nginx配置文件，修改如下：

server {  
       listen       8080;  
        server\_name  localhost;  
           index index.jsp;  
        #charset koi8-r;  
        #access\_log logs/host.access.log  main;  
        location / {  
        root /web/html;  
           index index.html;  
        }  
}  
    server {  
       listen       80;  
        server\_name  localhost;  
           index index.jsp;  
        #charset koi8-r;  
        #access\_log logs/host.access.log  main;

   location/ {  
         proxy\_pass http://10.0.10.61:80;  
        }  
  
     location ~\*\.(html|gif|jpg|jpeg|ico|swf|htm|xml)$ {  
         proxy\_pass http://10.0.10.62:8080;  
     }  
  
     location ~\* \.(jsp|do)$ {  
         proxy\_pass http://10.0.10.61:80;  
     }

}

访问测试，一切正常，如下所示

[](http://s3.51cto.com/wyfs02/M01/39/80/wKiom1O6StXx3ZPqAAOCzXFIzr4120.jpg)