# **nginx+tomcat会话保持方案**

tomcat做为轻量级的java应用服务器,对静态页面和大并发的处理并不是很好。所以现在流行的做法是使用nginx+tomcat来实现动静分离与负载均衡, 做负载均衡集群就要考虑会话保持。

一、Session概述

1、什么是会话(session)?

答: Session是在无状态的HTTP协议下，服务端记录用户状态时用于标识具体用户的机制。

http协议是无状态的,无状态就表明每一次客户端的访问请求是无关联的。比如张三在一个网站上有多次访问请求，李四也在这个网站上有多次访问请求，他们的请求顺序与时间是无规则的。那么网站服务器是不知道到底哪些是张三的请求，哪些是李四的请求。如果在他们请求时服务器都有一个固定的标签跟踪，就能分辨了。这个标签就可以看做是session.

2、为什么要会话保持?

答: 因为负载均衡集群会将同一个用户的请求通过算法调度给多台服务器，变成了一对多的局面。就好像我要给父母打电话，父母原来使用同一个电话变成了父母各使用一个电话，我说的话，被一半一半的调度给父母各自持有的电话上。那么最终的结果就是父母各听了一半，根本不知道我要说什么。

所以负载均衡和会话保持是矛盾的,但又必须得共存。

二、会话保持的几个方案:

1、nginx的ip\_hash算法

实现原理就是同一个客户端的所有请求只调度给同一个后台tomcat,这样会话就能保持在同一台服务器上。

注：其它的负载均衡软件也有类似算法:如LVS的sh算法,haproxy的source算法等。

优点:配置最简单,在nginx的upstream调度里加一句ip\_hash即可

缺点:后端tomcat宕机,用户session会丢失

2、tomcat的session复制集群

多台tomcat通过组播互相沟通会话信息,以保持不同tomcat之间的会话一致性。

优点:后端tomcat宕机,用户session不丢失

缺点:使用组播将信息复制到多个tomcat节点，网络开销大

3、缓存集中式管理session

session可以保存在文件,数据库和内存中,利用memcached或redis将session信息缓存,以达到会话信息不丢失，也不影响负载均衡的目的。

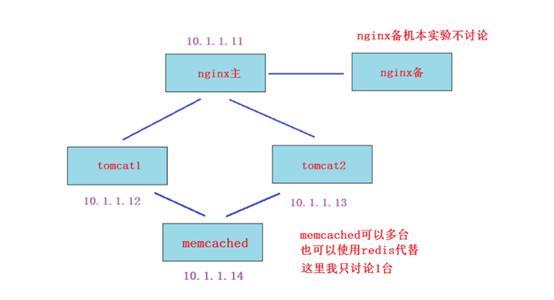
优点:只要缓存服务器没问题,用户session不会丢.也没有额外的网络开销

缺点:太依赖缓存服务器;需要额外的缓存服务器,成本也高;当然要求维护人员技术水平也较高，适合于性能要求高的大型环境。

三、使用MSM实现nginx+tomcat集群的会话保持

1、MSM是什么？

MSM(memcached-session-manager)就是缓存集中式管理session的一种方案 ,它可以把tomcat的会话信息保存在缓存服务器memcached中，实现在负载均衡调度时还能保持会话一致。



2、实验准备:

① 静态ip

10.1.1.11 http://nginx.cluster.com

10.1.1.12 http://tomcat1.cluster.com

10.1.1.13 http://tomcat2.cluster.com

10.1.1.14 http://memcached.cluster.com

② 主机名绑定

③ 关闭防火墙和Selinux

④ 时间同步

⑤ yum源(centos7安装完系统后的默认yum源就OK)

3、实验过程

第1步: 在所有tomcat节点上安装

首先在官网下载二进制版tomcat

下载地址: https://tomcat.apache.org/download-90.cgi

在centos7上确认openjdk已经安装

# java -version

openjdk version "1.8.0\_161"

OpenJDK Runtime Environment (build 1.8.0\_161-b14)

OpenJDK 64-Bit Server VM (build 25.161-b14, mixed mode)

直接解压下载好的二进制版tomcat安装并启动

# tar xf /root/Desktop/apache-tomcat-9.0.14.tar.gz -C /usr/local/

# mv /usr/local/apache-tomcat-9.0.14/ /usr/local/tomcat

# /usr/local/tomcat/bin/startup.sh

第2步:在所有tomcat节点(tomcat1和tomcat2)的家目录里创建一个显示session信息的代码文件

注:文件内容是一模一样的

[root@tomcat1 ~]# vim /usr/local/tomcat/webapps/ROOT/session.jsp

SessionID:<%=session.getId()%> <BR>

SessionIP:<%=request.getServerName()%> <BR>

SessionPort:<%=request.getServerPort()%>

[root@tomcat2 ~]# vim /usr/local/tomcat/webapps/ROOT/session.jsp

SessionID:<%=session.getId()%> <BR>

SessionIP:<%=request.getServerName()%> <BR>

SessionPort:<%=request.getServerPort()%>

第3步: 安装并配置nginx配置文件,实现nginx+tomcat的动静分离与负载均衡。

[root@nginx ~]# yum install epel-release -y

[root@nginx ~]# yum install nginx -y

[root@nginx ~]# vim /etc/nginx/nginx.conf

将下面一段加到http {}配置段里但是不要在server {}配置段里

upstream tomcat {

server 10.1.1.12:8080 weight=1;

server 10.1.1.13:8080 weight=1;

}

把server {}配置段里下面一段修改

location / {

}

修改成

location ~ .\*\.jsp$ {

proxy\_pass http://tomcat;

proxy\_set\_header Host $host;

proxy\_set\_header X-Forwarded-For $remote\_addr;

}

最终结果如下图所示：



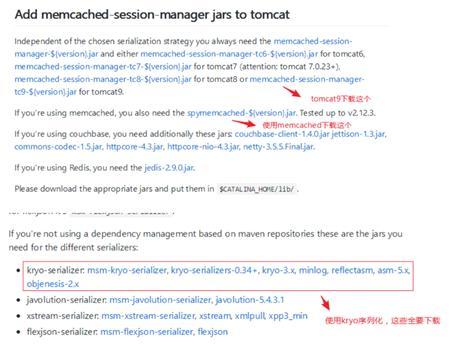
第4步：启动nginx服务

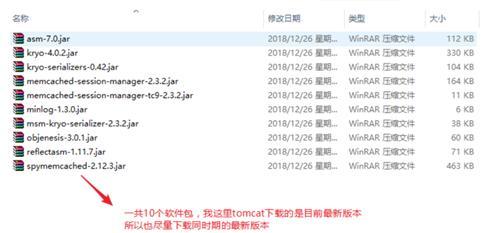
[root@vm1 ~]# systemctl restart nginx

[root@vm1 ~]# systemctl enable nginx

第5步: 下载MSM相关的jar包,并拷贝到所有tomcat节点的/usr/local/tomcat/lib/目录下

下载地址 ：https://github.com/magro/memcached-session-manager/wiki/SetupAndConfiguration







第6步: 确认jar包都拷贝完成后,配置所有tomcat节点(tomcat1和tomcat2都一样配置)

把下面一段加到context.xml配置文件最后一行前面; 也就是<Context> </Context>标签中间;10.1.1.14为memcached服务器IP

# vim /usr/local/tomcat/conf/context.xml

<Manager className="de.javakaffee.web.msm.MemcachedBackupSessionManager"

memcachedNodes="n1:10.1.1.14:11211"

lockingMode="auto"

sticky="false"

requestUriIgnorePattern= ".\*\.(png|gif|jpg|css|js)$"

sessionBackupAsync= "false"

sessionBackupTimeout= "100"

copyCollectionsForSerialization="true"

transcoderFactoryClass="de.javakaffee.web.msm.serializer.kryo.KryoTranscoderFactory" />



第7步: 所有tomcat节点重启tomcat服务

# /usr/local/tomcat/bin/shutdown.sh

# /usr/local/tomcat/bin/startup.sh

第8步: 在memcached服务器上安装,并启动服务

[root@vm4 ~]# yum install memcached -y

[root@vm4 ~]# systemctl restart memcached

[root@vm4 ~]# systemctl enable memcached

[root@vm4 ~]# lsof -i:11211

COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME

memcached 4224 memcached 26u IPv4 44869 0t0 TCP \*:memcache (LISTEN)

memcached 4224 memcached 27u IPv6 44870 0t0 TCP \*:memcache (LISTEN)

memcached 4224 memcached 28u IPv4 44873 0t0 UDP \*:memcache

memcached 4224 memcached 29u IPv6 44874 0t0 UDP \*:memcache

第9步: 使用浏览器访问测试



至此,会话保持就成功了