**Java + Tomcat**

**java是一个跨平台的、面向对象的程序设计语言，java技术具有卓越的通用性、高效性、平台移植性和安全性**。

Java体系

Java SE（标准版） Java EE（企业版） Java ME (移动版)

JDK是Sun针对Java开发者推出的Java语言的软件开发工具包，是核心

Servlet是java扩展Web服务器功能的主讲规范

常见Servlet容器

IBM websphere

Oracle weblogic

Apacge tomcat

RedHat Jboss

**步骤一：部署Tomcat服务器软件(**192.168.2.100/24)

**1）使用RPM安装JDK环境;已经自带**

[root@web1 ~]# yum -y install java-1.8.0-openjdk //安装JDK

[root@web1 ~]# yum -y install java-1.8.0-openjdk-headless //安装JDK

[root@web1 ~]# java -version //查看JAVA版本

1. **安装Tomcat（apache-tomcat-8.0.30.tar.gz软件包，在lnmp\_soft中有提供）**

[root@web1 ~]# tar -xf apache-tomcat-8.0.30.tar.gz

[root@web1 ~]# mv apache-tomcat-8.0.30 /usr/local/tomcat

[root@web1 ~]# ls /usr/local/tomcat

bin/ //主程序目录

lib/ //库文件目录

logs/ //日志目录

temp/ //临时目录

work/ //自动编译目录jsp代码转换servlet

conf/ //配置文件目录

webapps/ //页面目录

1. **启动服务**

[root@web1 ~]# /usr/local/tomcat/bin/startup.sh

[root@web1 ~]# firewall-cmd --set-default-zone=trusted

[root@web1 ~]# setenforce 0

1. **服务器验证端口信息 8080 8005 8009**

[root@web1 ~]# netstat -nutlp |grep java //查看java监听的端口

tcp LISTEN 0 100 :::8080

tcp LISTEN 0 1 ::ffff:127.0.0.1:8005

tcp LISTEN 0 100 :::8009

提示：如果检查端口时，8005端口启动非常慢，可用使用下面的命令用urandom替换random（非必须操作）。寻找乱码

[root@web1 ~]# mv /dev/random /dev/random.bak

[root@web1 ~]# ln -s /dev/urandom /dev/random

#这三个端口都启了，tomcat才能用

测试页面： firefox 192.168.2.100:8080

**步骤二：修改Tomcat配置文件**

**1）创建测试JSP页面**

[root@web1 ~]# vim /usr/local/tomcat/webapps/ROOT/test.jsp

**2）重启服务**

[root@web1 ~]# /usr/local/tomcat/bin/shutdown.sh

[root@web1 ~]# /usr/local/tomcat/bin/startup.sh

**2）客户端浏览测试页面(proxy充当客户端角色)**

[root@proxy ~]# firefox http://192.168.2.100:8080/test.jsp

**案例2：使用Tomcat部署虚拟主机**

**修改server.xml配置文件，创建两个域名的虚拟主机，修改如下两个参数块**：

[root@web1 ~]# vim /usr/local/tomcat/conf/server.xml

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

</Host>

#复制模版另外添加到原模版上面，不要添加在<Host name>里面，记住结尾</Host>

**2）创建虚拟主机对应的页面根路径**

[root@web1 ~]# mkdir -p /usr/local/tomcat/{a,b}/ROOT

[root@web1 ~]# echo "AAA" > /usr/local/tomcat/a/ROOT/index.html

[root@web1 ~]# echo "BBB" > /usr/local/tomcat/b/ROOT/index.html

**3）客户端设置host文件，并浏览测试页面进行测试(proxy充当客户端角色)**

[root@proxy ~]# vim /etc/hosts

… …

192.168.2.100 www.a.com www.b.com

[root@proxy ~]# firefox http://www.a.com:8080/ //注意访问的端口为8080

[root@proxy ~]# firefox http://www.b.com:8080/

**步骤二：修改www.b.com网站的首页目录为base**

1）使用docBase参数可以修改默认网站首页路径

[root@web1 ~]# vim /usr/local/tomcat/conf/server.xml

… …

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

<Context path="" docBase="base" reloadable="true"/>

</Host>

[root@web1 ~]# mkdir /usr/local/tomcat/b/base

[root@web1 ~]# echo "BASE" > /usr/local/tomcat/b/base/index.html

[root@web1 ~]# /usr/local/tomcat/bin/shutdown.sh

[root@web1 ~]# /usr/local/tomcat/bin/startup.sh

**2）测试查看页面是否正确(proxy充当客户端角色)**

[root@proxy ~]# firefox http://www.b.com:8080/ //结果为base目录下的页面内容

**步骤三：跳转**

**1）当用户访问http://www.a.com/test打开/var/www/html目录下的页面**

[root@web1 ~]# vim /usr/local/tomcat/conf/server.xml

… …

<Host name="www.a.com" appBase="a" unpackWARS="true" autoDeploy="true">

<Context path="/test" docBase="/var/www/html/" />

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

<Context path="" docBase="base" />

</Host>

… …

[root@web1 ~]# echo "Test" > /var/www/html/index.html

[root@web1 ~]# /usr/local/tomcat/bin/shutdown.sh

[root@web1 ~]# /usr/local/tomcat/bin/startup.sh

2）测试查看页面是否正确(proxy充当客户端角色)

[root@proxy ~]# firefox http://www.a.com:8080/test

**步骤四：配置Tomcat支持SSL加密网站**

**1）创建加密用的私钥和证书文件**

[root@web1 ~]# keytool -genkeypair -alias tomcat -keyalg RSA -keystore /usr/local/tomcat/keystore

//-genkeypair 生成密钥对

//-alias tomcat 密钥别名

//-keyalg RSA 定义密钥算法为RSA算法

//-keystore 定义密钥文件存储在:/usr/local/tomcat/keystore

**2)再次修改server.xml配置文件，创建支持加密连接的Connect**or

[root@web1 ~]# vim /usr/local/tomcat/conf/server.xml

… …

<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"

maxThreads="150" SSLEnabled="true" scheme="https" secure="true"

keystoreFile="/usr/local/tomcat/keystore" keystorePass="123456" clientAuth="false" sslProtocol="TLS" />

//备注，默认这段Connector被注释掉了，打开注释，添加密钥信息即可

1. **重启Tomcat服务器**

**[root@web1 ~]# /usr/local/tomcat/bin/shutdown.sh**

**[root@web1 ~]# /usr/local/tomcat/bin/startup.sh**

1. **客户端设置host文件，并浏览测试页面进行测试(proxy充当客户端角色)**

**[root@proxy ~]# vim /etc/hosts**

**… …**

**192.168.2.100 www.a.com www.b.com**

**[root@proxy ~]# firefox https://www.a.com:8443/**

**[root@proxy ~]# firefox https://www.b.com:8443/**

**[root@proxy ~]# firefox <https://192.168.2.100:8443/>**

**步骤五：配置Tomcat日志**

**1)为每个虚拟主机设置不同的日志文件**

[root@web1 ~]# vim /usr/local/tomcat/conf/server.xml

.. ..

<Context path="/test" docBase="/var/www/html/" />

#从默认localhost虚拟主机中把Valve这段复制过来，适当修改下即可

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

prefix=" a\_access" suffix=".txt"

pattern="%h %l %u %t &quot;%r&quot; %s %b" />

</Host>

<Host name="www.b.com" appBase="b" unpackWARS="true" autoDeploy="true">

<Context path="" docBase="base" />

<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"

prefix=" b\_access" suffix=".txt"

pattern="%h %l %u %t &quot;%r&quot; %s %b" />

</Host>

#prefix=”命名” suffix=”后缀”

**2）重启Tomcat服务器**

**3）查看服务器日志文件**

[root@web1 ~]# ls /usr/local/tomcat/logs/

**步骤六：扩展实验(配置Tomcat集群)**

**1) 在192.168.4.5主机上配置Nginx调度器（具体安装步骤参考前面的章节**）

[root@proxy ~]# vim /usr/local/nginx/conf/nginx.conf

http{

upstream toms {

server 192.168.2.100:8080;

server 192.168.2.200:8080;

}

server {

listen 80;

server\_name localhost;

location / {

proxy\_pass http://toms;

}

**2) 在192.168.2.100和192.168.2.200主机上配置Tomcat调度器**

**以下以Web1为例：**

[root@web1 ~]# yum -y install java-1.8.0-openjdk //安装JDK

[root@web1 ~]# yum -y install java-1.8.0-openjdk-headless //安装JDK

[root@web1 ~]# tar -xzf apache-tomcat-8.0.30.tar.gz

[root@web1 ~]# mv apache-tomcat-8.0.30 /usr/local/tomcat

**3）启动服务**

[root@web1 ~]# /usr/local/tomcat/bin/startup.sh

**使用Varnish加速Web**

**步骤一：构建Web服务器**

1）使用yum安装web软件包

[root@web1 ~]# yum -y install httpd

2）启用httpd服务（注意需要关闭nginx，否则端口冲突）

3）为Web访问建立测试文件

在网站根目录/var/www/html下创建一个名为index.html的首页文件：

[root@web1 ~]# cat /var/www/html/index.html

192.168.2.100

4）测试页面是否正常

[root@proxy ~]# firefox http://192.168.2.100

**步骤二：部署Varnish缓存服务器(192.168.4.5)**

1）编译安装软件(python-docutils默认光盘中没有，需要在lnmp\_soft中找)

[root@proxy ~]# yum -y install gcc readline-devel //安装软件依赖包

[root@proxy ~]# yum -y install ncurses-devel //安装软件依赖包

[root@proxy ~]# yum -y install pcre-devel //安装软件依赖包

[root@proxy ~]# yum -y install \

python-docutils-0.11-0.2.20130715svn7687.el7.noarch.rpm //安装软件依赖包

[root@proxy ~]# useradd -s /sbin/nologin varnish //创建账户

[root@proxy ~]# tar -xf varnish-5.2.1.tar.gz

[root@proxy ~]# cd varnish-5.2.1

[root@proxy varnish-5.2.1]# ./configure

[root@proxy varnish-5.2.1]# make && make install

**2）复制启动脚本及配置文件**

[root@proxy varnish-5.2.1]# cp etc/example.vcl /usr/local/etc/default.vcl

3）修改代理配置文件

[root@proxy ~]# vim /usr/local/etc/default.vcl

backend default {

.host = "192.168.2.100";

.port = "80";

}

**4）启动服务**

[root@proxy ~]# varnishd -f /usr/local/etc/default.vcl

//varnishd命令的其他选项说明如下：

//varnishd –s malloc,128M 定义varnish使用内存作为缓存，空间为128M

//varnishd –s file,/var/lib/varnish\_storage.bin,1G 定义varnish使用文件作为缓存

**5.客户端开启浏览器访问**

[root@client ~]# curl http://192.168.4.5

步骤四：其他操作

**1）查看varnish日志**

[root@proxy ~]# varnishlog //varnish日志

[root@proxy ~]# varnishncsa //访问日志

**2）更新缓存数据，在后台web服务器更新页面内容后，用户访问代理服务器看到的还是之前的数据，说明缓存中的数据过期了需要更新（默认也会自动更新，但非实时更新）。**

[root@proxy ~]# varnishadm

varnish> ban req.url ~ .\*

//清空缓存数据，支持正则表达式

**总结：**

**一．部署Tomcat服务器**

1. 部署JDK环境(后端服务器192.168.2.100/24)

java-1.8.0-openjdk java-1.8.0.openjdk-headless

java-version #查看JAVA版本

1. 安装tomcat

解包 tar -xf apache-tomcat-8.0.30.tar.gz

安装程序 mv apache-tomcat-8.0.30 /usr/local/tomcat #tomcat是自己定义

1. 启动服务

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

1. 验证端口信息

ss -anputl | grep java #8080 8005 8009三个端口都开启

#若8005端口启动慢，可以使用下面命令

mv /dev/random /dev/random.bak

ln -s /dev/urandom /dev/random

1. 测试

firefox 192.168.2.100:8080

网站默认根目录是 webapps

创建测试页面

/usr/local/tomcat/webapps/ROOT/test.jsp

**二．使用tomcat部署虚拟主机**

1.修改server.xml配置文件，创建域名虚拟主机

vim /usr/local/tomcat/conf/server.xml

<Host name=”www.a.com” appBase=”a” unpackWARS=”true” autoDeploy=”true” />

<Host name=”www.b.com” appBase=”b” unpackWARS=”true” autoDeploy=”true” />

1. **创建虚拟主机对应的页面根路径**

mkdir -p /usr/local/tomcat/{a,b}/ROOT

echo “AA” > /usr/local/tomcat/a/ROOT/index.html

echo “BB” > /usr/local/tomcat/b/ROOT/index.html

1. **重启**

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

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

1. 测试 设置 /etc/hosts

firefox [www.a.com:8080](http://www.a.com:8080)

1. **修改www.b.com网站的首页目录为base**

1) /usr/local/tomcat/conf/server.xml

<Host name=”www.b.com” .... >

<Contest path=”” docBase=”base” reloadable=”true” />

</Host>

**2)创建首页目录**

mkdir /usr/locatl/tomcat/b/base

**3）创建页面**

echo “hello world” > /usr/local/tomcat/b/base/index.html

**跳转**

<Host name=”www.b.com” .... >

<Contest path=”test” docBase=”/var/www/html” />

</Host>

**2)创建测试页面**

echo “test” > /var/www/html/index.html

**3)重启，测试** firefox www.b.com:8080/test