tomcat配置加密虚拟主机的方法总结

- JSSE （作为Java运行部分）

- JSSE （使用OpenSSL）

- APR （使用OpenSSL）

#安装tomcat9.0+openjdk-1.8.0

[root@server2 ~]# yum -y install java-1.8.0-openjdk-devel.x86\_64

[root@server2 ~]# 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)

[root@server2 ~]# ls apache-tomcat-9.0.6.tar.gz

apache-tomcat-9.0.6.tar.gz

[root@server2 ~]# tar -xf apache-tomcat-9.0.6.tar.gz

[root@server2 ~]# mv apache-tomcat-9.0.6 /usr/local/tomcat

[root@server2 ~]# cd /usr/local/tomcat/

[root@server2 tomcat]# ls

bin conf lib LICENSE logs NOTICE RELEASE-NOTES RUNNING.txt temp webapps work

#备份server.xml文件，便于测试

[root@server2 tomcat]# cp conf/server.xml conf/server.xml.bak

#1:JSSE+KEYSTORE

[root@server2 tomcat]# keytool -genkeypair -alias tomcat -keyalg RSA -keystore /usr/local/tomcat/keystore

输入密钥库口令:

再次输入新口令:

您的名字与姓氏是什么?

[Unknown]: Mark Li

您的组织单位名称是什么?

[Unknown]: Tmooc

您的组织名称是什么?

[Unknown]: Tarena

您所在的城市或区域名称是什么?

[Unknown]: beijing

您所在的省/市/自治区名称是什么?

[Unknown]: Beijing

该单位的双字母国家/地区代码是什么?

[Unknown]: CN

CN=Mark Li, OU=Tmooc, O=Tarena, L=beijing, ST=Beijing, C=CN是否正确?

[否]: y

输入 <tomcat> 的密钥口令

(如果和密钥库口令相同, 按回车):

再次输入新口令:

Warning:

JKS 密钥库使用专用格式。建议使用 "keytool -importkeystore -srckeystore /usr/local/tomcat/keystore -destkeystore /usr/local/tomcat/keystore -deststoretype pkcs12" 迁移到行业标准格式 PKCS12。

[root@server2 tomcat]# ll keystore

-rw-r--r-- 1 root root 2227 2月 21 18:28 keystore

[root@server2 tomcat]# vim conf/server.xml

<Connector port="8080" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

<!-- A "Connector" using the shared thread pool-->

<!--

<Connector executor="tomcatThreadPool"

port="8080" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

-->

<!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443

This connector uses the NIO implementation. The default

SSLImplementation will depend on the presence of the APR/native

library and the useOpenSSL attribute of the

AprLifecycleListener.

Either JSSE or OpenSSL style configuration may be used regardless of

the SSLImplementation selected. JSSE style configuration is used below.

-->

<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" />

#启动服务

[root@server2 tomcat]# ./bin/startup.sh

Using CATALINA\_BASE: /usr/local/tomcat

Using CATALINA\_HOME: /usr/local/tomcat

Using CATALINA\_TMPDIR: /usr/local/tomcat/temp

Using JRE\_HOME: /usr

Using CLASSPATH: /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar

Tomcat started.

[root@server2 tomcat]# ss -ntpul | grep java

tcp LISTEN 0 1 ::ffff:127.0.0.1:8005 :::\* users:(("java",pid=1587,fd=77))

tcp LISTEN 0 100 :::8009 :::\* users:(("java",pid=1587,fd=57))

tcp LISTEN 0 100 :::8080 :::\* users:(("java",pid=1587,fd=48))

tcp LISTEN 0 100 :::8443 :::\* users:(("java",pid=1587,fd=53))

#8443端口启动正常

#2:JSSE+OPENSSL

#还原配置文件

[root@server2 tomcat]# ./bin/shutdown.sh

Using CATALINA\_BASE: /usr/local/tomcat

Using CATALINA\_HOME: /usr/local/tomcat

Using CATALINA\_TMPDIR: /usr/local/tomcat/temp

Using JRE\_HOME: /usr

Using CLASSPATH: /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar

[root@server2 tomcat]# ss -ntplu | grep java

[root@server2 tomcat]# rm -rf conf/server.xml

[root@server2 tomcat]# cp conf/server.xml.bak conf/server.xml

##OpenSSL配置

[root@server2 tomcat]# mkdir certs

[root@server2 tomcat]# cd certs/

##此处可以搭建一台内部的CA服务器，这里为了简单用了命令代替

#生成根秘钥

[root@server2 certs]# openssl genrsa > root.key

Generating RSA private key, 2048 bit long modulus

...................................+++

....+++

e is 65537 (0x10001)

#生成根证书

[root@server2 certs]# openssl req -new -x509 -key root.key > root.crt

You are about to be asked to enter information that will be incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

-----

Country Name (2 letter code) [XX]:CN

State or Province Name (full name) []:Beijing

Locality Name (eg, city) [Default City]:beijing

Organization Name (eg, company) [Default Company Ltd]:Tarena

Organizational Unit Name (eg, section) []:Tmooc

Common Name (eg, your name or your server's hostname) []:mark

Email Address []:mark@tedu.cn

[root@server2 certs]# ll root.\*

-rw-r--r-- 1 root root 1383 2月 21 20:11 root.crt

-rw-r--r-- 1 root root 1675 2月 21 20:08 root.key

#生成服务端秘钥

[root@server2 certs]# openssl genrsa > server.key

Generating RSA private key, 2048 bit long modulus

................................+++

.........................................................+++

e is 65537 (0x10001)

#生成服务端证书请求文件

[root@server2 certs]# openssl req -new -key server.key > server.csr

You are about to be asked to enter information that will be incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

-----

Country Name (2 letter code) [XX]:CN

State or Province Name (full name) []:Beijing

Locality Name (eg, city) [Default City]:beijing

Organization Name (eg, company) [Default Company Ltd]:Tarena

Organizational Unit Name (eg, section) []:Tmooc

Common Name (eg, your name or your server's hostname) []:mark

Email Address []:mark@tedu.cn

Please enter the following 'extra' attributes

to be sent with your certificate request

A challenge password []:123456

An optional company name []:Tarena

#根证书签发服务端证书

[root@server2 certs]# openssl x509 -req -in server.csr -CA root.crt -CAkey root.key -CAcreateserial -days 3650 > server.crt

Signature ok

subject=/C=CN/ST=Beijing/L=beijing/O=Tarena/OU=Tmooc/CN=mark/emailAddress=mark@tedu.cn

Getting CA Private Key

#将证书导出为pkcs12格式

[root@server2 certs]# openssl pkcs12 -export -in server.crt -inkey server.key > server.pkcs12

Enter Export Password:

Verifying - Enter Export Password:

[root@server2 certs]# ls server.\*

server.crt server.csr server.key server.pkcs12

#执行keytool命令生成服务端秘钥库

[root@server2 certs]# keytool -importkeystore -srckeystore server.pkcs12 -destkeystore mykey.keystore -srcstoretype pkcs12

正在将密钥库 server.pkcs12 导入到 mykey.keystore...

输入目标密钥库口令: 123456

再次输入新口令: 123456

输入源密钥库口令: 123456

已成功导入别名 1 的条目。

已完成导入命令: 1 个条目成功导入, 0 个条目失败或取消

Warning:

JKS 密钥库使用专用格式。建议使用 "keytool -importkeystore -srckeystore mykey.keystore -destkeystore mykey.keystore -deststoretype pkcs12" 迁移到行业标准格式 PKCS12。

[root@server2 certs]# ll mykey.keystore

-rw-r--r-- 1 root root 2240 2月 21 20:19 mykey.keystore

[root@server2 certs]# keytool -list -v -keystore mykey.keystore

输入密钥库口令: 123456

密钥库类型: JKS

密钥库提供方: SUN

您的密钥库包含 1 个条目

别名: 1

创建日期: 2019-2-21

条目类型: PrivateKeyEntry

证书链长度: 1

证书[1]:

所有者: EMAILADDRESS=mark@tedu.cn, CN=mark, OU=Tmooc, O=Tarena, L=beijing, ST=Beijing, C=CN

发布者: EMAILADDRESS=mark@tedu.cn, CN=mark, OU=Tmooc, O=Tarena, L=beijing, ST=Beijing, C=CN

序列号: 93c6856d9f717a7b

有效期为 Thu Feb 21 07:15:36 EST 2019 至 Sun Feb 18 07:15:36 EST 2029

证书指纹:

MD5: D1:06:D9:63:BA:32:06:D4:A3:00:A1:3C:03:6F:C7:E7

SHA1: 23:A5:40:CC:ED:88:52:D5:2D:13:DA:E4:46:9C:9E:31:F0:B3:52:BC

SHA256: 32:EA:E5:52:D5:29:1C:53:B8:54:17:8C:C4:98:C8:D1:7F:EA:D4:8B:44:DC:DB:1B:14:49:D9:3B:76:AD:E1:42

签名算法名称: SHA256withRSA

主体公共密钥算法: 2048 位 RSA 密钥

版本: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Warning:

JKS 密钥库使用专用格式。建议使用 "keytool -importkeystore -srckeystore mykey.keystore -destkeystore mykey.keystore -deststoretype pkcs12" 迁移到行业标准格式 PKCS12。

#配置server.xml

[root@server2 certs]# cd ..

[root@server2 tomcat]# vim conf/server.xml

<Connector port="8080" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

<!-- A "Connector" using the shared thread pool-->

<!--

<Connector executor="tomcatThreadPool"

port="8080" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

-->

<!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443

This connector uses the NIO implementation. The default

SSLImplementation will depend on the presence of the APR/native

library and the useOpenSSL attribute of the

AprLifecycleListener.

Either JSSE or OpenSSL style configuration may be used regardless of

the SSLImplementation selected. JSSE style configuration is used below.

-->

<!--

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

maxThreads="150" SSLEnabled="true">

<SSLHostConfig>

<Certificate certificateKeystoreFile="conf/localhost-rsa.jks"

type="RSA" />

</SSLHostConfig>

</Connector>

-->

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

maxThreads="150" SSLEnabled="true"

scheme="https" secure="true" >

<SSLHostConfig>

<!--

< Certificate certificateKeystoreFile="/usr/local/tomcat/certs/mykey.keystore" />

-->

<Certificate certificateKeyFile="/usr/local/tomcat/certs/server.key"

certificateFile="/usr/local/tomcat/certs/server.crt"

type="RSA" />

</SSLHostConfig>

</Connector>

#配置文件中蓝色部分为正常配置部分，黑色部分为注释部分，下边的蓝色的connector部分中有红色和黄色两个部分，两种写法二选一，均可实现效果，其中红色部分与第一种方式一样，只不过keystorefile是由OpenSSL生成，不是采用keytool命令创建，即结合了OpenSSL，黄色部分是OpenSSL配置方法，即指定了服务端的key和crt两个文件

#验证服务

[root@server2 tomcat]# ./bin/startup.sh

Using CATALINA\_BASE: /usr/local/tomcat

Using CATALINA\_HOME: /usr/local/tomcat

Using CATALINA\_TMPDIR: /usr/local/tomcat/temp

Using JRE\_HOME: /usr

Using CLASSPATH: /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar

Tomcat started.

[root@server2 tomcat]# ss -ntplu | grep java

tcp LISTEN 0 1 ::ffff:127.0.0.1:8005 :::\* users:(("java",pid=4728,fd=77))

tcp LISTEN 0 100 :::8009 :::\* users:(("java",pid=4728,fd=57))

tcp LISTEN 0 100 :::8080 :::\* users:(("java",pid=4728,fd=48))

tcp LISTEN 0 100 :::8443 :::\* users:(("java",pid=4728,fd=53))

#8443端口正常启动

#3:JSSE+APR

#该方法即修改一下http的协议方法，启动APR模式

#上述已经完成基础环境部署，不在赘述

#停止tomcat服务，重新部署配置文件

[root@server2 tomcat]# ./bin/shutdown.sh

Using CATALINA\_BASE: /usr/local/tomcat

Using CATALINA\_HOME: /usr/local/tomcat

Using CATALINA\_TMPDIR: /usr/local/tomcat/temp

Using JRE\_HOME: /usr

Using CLASSPATH: /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar

[root@server2 tomcat]# jps

1678 Jps

[root@server2 tomcat]# rm -rf conf/server.xml

[root@server2 tomcat]# cp conf/server.xml.bak conf/server.xml

#开始配置

[root@server2 tomcat]# yum -y install gcc openssl-devel apr apr-util apr-devel

#秘钥文件沿用2中的server.key和server.crt

[root@server2 tomcat]# ls certs/server.\*

certs/server.crt certs/server.csr certs/server.key certs/server.pkcs12

[root@server2 tomcat]# ./bin/shutdown.sh

Using CATALINA\_BASE: /usr/local/tomcat

Using CATALINA\_HOME: /usr/local/tomcat

Using CATALINA\_TMPDIR: /usr/local/tomcat/temp

Using JRE\_HOME: /usr

Using CLASSPATH: /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar

[root@server2 tomcat]# rm -rf conf/server.xml

[root@server2 tomcat]# cp conf/server.xml.bak conf/server.xml

#修改配置文件

[root@server2 tomcat]# vim conf/server.xml

<Connector port="8080" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

<!-- A "Connector" using the shared thread pool-->

<!--

<Connector executor="tomcatThreadPool"

port="8080" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8443" />

-->

<!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443

This connector uses the NIO implementation. The default

SSLImplementation will depend on the presence of the APR/native

library and the useOpenSSL attribute of the

AprLifecycleListener.

Either JSSE or OpenSSL style configuration may be used regardless of

the SSLImplementation selected. JSSE style configuration is used below.

-->

<!--

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

maxThreads="150" SSLEnabled="true">

<SSLHostConfig>

<Certificate certificateKeystoreFile="conf/localhost-rsa.jks"

type="RSA" />

</SSLHostConfig>

</Connector>

-->

<!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443 with HTTP/2

This connector uses the APR/native implementation which always uses

OpenSSL for TLS.

Either JSSE or OpenSSL style configuration may be used. OpenSSL style

configuration is used below.

-->

<!--

<Connector port="8443" protocol="org.apache.coyote.http11.Http11AprProtocol"

maxThreads="150" SSLEnabled="true" >

<UpgradeProtocol className="org.apache.coyote.http2.Http2Protocol" />

<SSLHostConfig>

<Certificate certificateKeyFile="conf/localhost-rsa-key.pem"

certificateFile="conf/localhost-rsa-cert.pem"

certificateChainFile="conf/localhost-rsa-chain.pem"

type="RSA" />

</SSLHostConfig>

</Connector>

-->

<Connector port="8443" protocol="org.apache.coyote.http11.Http11AprProtocol"

maxThreads="200" SSLEnabled="true"

scheme="https" secure="true"

SSLCertificateFile="/usr/local/tomcat/certs/server.crt"

SSLCertificateKeyFile="/usr/local/tomcat/certs/server.key"

SSLVerifyClient="optional"

SSLProtocol="TLSv1+TLSv1.1+TLSv1.2" />

<!-- Define an AJP 1.3 Connector on port 8009 -->

<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />

#配置ARP

[root@server2 tomcat]# cd bin/

[root@server2 bin]# tar -xf tomcat-native.tar.gz

[root@server2 bin]# cd tomcat-native-1.2.16-src/native/

[root@server2 native]# which apr-1-config

/usr/bin/apr-1-config

[root@server2 native]# ls /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-2.b14.el7.x86\_64/

bin include jre lib tapset

[root@server2 native]# ./configure --with-apr=/usr/bin/apr-1-config --with-java-home=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-2.b14.el7.x86\_64/ --with-ssl=yes

configure: creating ./config.status

config.status: creating tcnative.pc

config.status: creating Makefile

config.status: executing default commands

[root@server2 native]# make && make install

Libraries have been installed in:

/usr/local/apr/lib

If you ever happen to want to link against installed libraries

in a given directory, LIBDIR, you must either use libtool, and

specify the full pathname of the library, or use the `-LLIBDIR'

flag during linking and do at least one of the following:

- add LIBDIR to the `LD\_LIBRARY\_PATH' environment variable

during execution

- add LIBDIR to the `LD\_RUN\_PATH' environment variable

during linking

- use the `-Wl,-rpath -Wl,LIBDIR' linker flag

- have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for

more information, such as the ld(1) and ld.so(8) manual pages.

#见到如上信息则安装成功

##一波骚操作，按照提示出来的配置环境变量各种不合适，最后测试出来一个方法

[root@server2 native]# cd /usr/local/tomcat/

[root@server2 tomcat]# cp -r /usr/local/apr/lib/\* /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-2.b14.el7.x86\_64/jre/lib/amd64/

#启动服务

[root@server2 tomcat]# ./bin/startup.sh

Using CATALINA\_BASE: /usr/local/tomcat

Using CATALINA\_HOME: /usr/local/tomcat

Using CATALINA\_TMPDIR: /usr/local/tomcat/temp

Using JRE\_HOME: /usr

Using CLASSPATH: /usr/local/tomcat/bin/bootstrap.jar:/usr/local/tomcat/bin/tomcat-juli.jar

Tomcat started.

[root@server2 tomcat]# ss -ntplu | grep java

tcp LISTEN 0 1 ::ffff:127.0.0.1:8005 :::\* users:(("java",pid=3548,fd=74))

tcp LISTEN 0 100 :::8009 :::\* users:(("java",pid=3548,fd=54))

tcp LISTEN 0 100 :::8080 :::\* users:(("java",pid=3548,fd=48))

tcp LISTEN 0 100 :::8443 :::\* users:(("java",pid=3548,fd=53))

#8443端口正常启动