秋明分享平台之

**CentOS下配置apache双向https认证负载均衡代理+tomcat集群**

撰稿人：秋明

撰稿人邮箱：tyumen@zhoufengjie.cn

QQ群：90122290、68610841【朋友建的一个群，包括你有什么比较不错的群也可以联系我放到这里】

本文撰稿日期： 2015年3月26日星期四

本站口号：开源、分享、共进

注解：任何想通过本平台分享文档的，可以随时联系，文档里面注明出稿人和邮箱，方便文档使用者同撰稿的兄弟技术咨询沟通和交流；

目录

1 前言 4

2 apache及tomcat安装 4

3 ssl双向证书生成 4

3.1 创建一个新的CA根证书 4

3.1.1 在conf 目录中新建 openssl.conf 文件 5

3.1.2 生成私钥 key 文件 6

3.1.3 生成证书请求 csr 文件 7

3.1.4 生成凭证 crt 文件 8

3.1.5 为我们的 key 设置起始序列号 8

3.1.6 创建 CA 键库 9

3.2 为 "用户证书" 的移除创建一个证书撤销列表 9

3.3 服务器证书的生成 9

3.3.1 创建一个 key 9

3.3.2 为我们的 key 创建一个证书签名请求 csr 文件 9

3.3.3 使用我们私有的 CA key 为刚才的 key 签名 11

3.4 客户端证书的生成 11

3.4.1 创建存放 key 的目录 users 11

3.4.2 为用户创建一个 key 11

3.4.3 为 key 创建一个证书签名请求 csr 文件 12

3.4.4 使用我们私有的 CA key 为刚才的 key 签名 13

2.3.5 将证书转换为大多数浏览器都能识别的 PKCS12 文件 14

4 apache配置 15

5 Tomcat 配置 18

6 配置验证 19

6.1 Tomcat 重启验证 19

6.2 Nginx 重启验证 19

6.3 客户访问 ssl 验证 19

7 双向 SSL 验证的集群配置 22

7.1 Nginx 配置 22

7.2 Tomcat 配置 23

# 前言

这里做集群，nginx做负载均衡，tomcat做server进行服务；在nginx上面做https双向认证；

# apache及tomcat安装

在此忽略apache和tomcat的安装步骤，安装目录：/etc/httpd;和/opt/apache-tomcat-version

# ssl双向证书生成

首先自己造ssl证书，用于双向认证使用；

* 1. 创建一个新的CA根证书

这里不重新做证书了，就用我使用nginx做负载均衡做的证书，所以把在nginx下做证书的步骤写在这里（其实你可以把证书做在一个公共的位置，这样就都可以用了）；

在 nginx 安装目录下新建 ca 文件夹，进入 ca，创建几个子文件夹：
**# mkdir ca**
**# cd ca**
**# mkdir newcerts private conf server**
newcerts 子目录将用于存放 CA 签署过的数字证书(证书备份目录)；private 用于存放 CA 的私钥；conf 目录用于存放一些简化参数用的配置文件；server 存放服务器证书文件。

* + 1. 在conf 目录中新建 openssl.conf 文件

编辑其内容如下：vi /usr/local/nginx/ca/conf/openssl.conf

[ ca ]

default\_ca = foo # The default ca section

[ foo ]

dir = /usr/local/nginx/ca # top dir

database = /usr/local/nginx/ca/index.txt # index file.

new\_certs\_dir = /usr/local/nginx/ca/newcerts # new certs dir

certificate = /usr/local/nginx/ca/private/ca.crt # The CA cert

serial = /usr/local/nginx/ca/serial # serial no file

private\_key = /usr/local/nginx/ca/private/ca.key # CA private key

RANDFILE = /usr/local/nginx/ca/private/.rand # random number file

default\_days = 365 # how long to certify for

default\_crl\_days= 30 # how long before next CRL

default\_md = sha1 # message digest method to use

unique\_subject = no # Set to 'no' to allow creation of

 # several ctificates with same subject.

policy = policy\_any # default policy

[ policy\_any ]

countryName = match

stateOrProvinceName = match

organizationName = match

organizationalUnitName = match

localityName = optional

commonName = supplied

emailAddress = optional

* + 1. 生成私钥 key 文件

# cd /usr/local/nginx/ca# openssl genrsa -out private/ca.key
输出
Generating RSA private key, 512 bit long modulus
..++++++++++++
.++++++++++++
e is 65537 (0x10001)
private 目录下有 ca.key 文件生成。

* + 1. 生成证书请求 csr 文件

**# openssl req -new -key private/ca.key -out private/ca.csr**

输入内容示例：

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]:SinoCache

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

Common Name (eg, your name or your server's hostname) []:\*.zhoufengjie.cn

Email Address []:tyumen@zhoufengjie.cn

Please enter the following 'extra' attributes

to be sent with your certificate request

A challenge password []:

An optional company name []:

****

* + 1. 生成凭证 crt 文件

# openssl x509 -req -days 365 -in private/ca.csr -signkey private/ca.key -out **private/ca.crt**
控制台输出
Signature ok

subject=/C=cn/ST=BeiJing/L=BeiJing/O=SinoCache/OU=DevOpt/CN=\*.zhoufengjie.cn/emailAddress=tyumen@zhoufengjie.cn

Getting Private key
private 目录下有 ca.crt 文件生成。

* + 1. 为我们的 key 设置起始序列号

**# echo FACE > serial**
可以是任意四个字符

* + 1. 创建 CA 键库

# touch index.txt

* 1. 为 "用户证书" 的移除创建一个证书撤销列表

**#openssl ca -gencrl -out /usr/local/nginx/ca/private/ca.crl -crldays 7 -config** "/usr/local/nginx/ca/conf/openssl.conf"
输出
Using configuration from /usr/local/nginx/ca/conf/openssl.conf
private 目录下有 ca.crl 文件生成。

* 1. 服务器证书的生成
		1. 创建一个 key

# openssl genrsa -out server/server.key输出
Generating RSA private key, 512 bit long modulus
...........................++++++++++++
.................++++++++++++
e is 65537 (0x10001)
server 目录下有 server.key 文件生成。

* + 1. 为我们的 key 创建一个证书签名请求 csr 文件

**#openssl req -new -key server/server.key -out server/server.csr**
这时会要求你输入和上面一样的那些问题，所有输入需要和那一步一致。但 A challenge password 是服务器证书口令，可以与根证书口令一致。这里：

输入示例：

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]:SinoCache

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

Common Name (eg, your name or your server's hostname) []:\*.zhoufengjie.cn

Email Address []:tyumen@zhoufengjie.cn

Please enter the following 'extra' attributes

to be sent with your certificate request

A challenge password []:

An optional company name []:


server 目录下有 server.csr 文件生成。

* + 1. 使用我们私有的 CA key 为刚才的 key 签名

**# openssl ca -in server/server.csr -cert private/ca.crt -keyfile private/ca.key** -out server/server.crt -config "/usr/local/nginx/ca/conf/openssl.conf"输出

两次都输入 y，server 目录下有 server.crt 文件生成。

* 1. 客户端证书的生成
		1. 创建存放 key 的目录 users

# mkdir users位置 /usr/local/nginx/ca/users。

* + 1. 为用户创建一个 key

**# openssl genrsa -des3 -out /usr/local/nginx/ca/users/client.key 1024**

要求输入 pass phrase，这个是当前 key 的口令，以防止本密钥泄漏后被人盗用。两次输入同一个密码(比如我这里输入 Password)，users 目录下有 client.key 文件生成。

* + 1. 为 key 创建一个证书签名请求 csr 文件

#openssl req -new -key /usr/local/nginx/ca/users/client.key -out /usr/local/nginx/ca/users/client.csr
提示输入 pass phrase，即 client.key 的口令。将pass phrase 口令输入后并回车：

要求你输入和 前面一样的那些问题。输入需要和那一步一致。但 A challenge password 是客户端证书口令(请注意将它和 client.key 的口令区分开！)，可以与服务器端证书或者根证书口令一致：

输入示例：

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]:SinoCache

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

Common Name (eg, your name or your server's hostname) []:\*.zhoufengjie.cn

Email Address []:tyumen@zhoufengjie.cn

Please enter the following 'extra' attributes

to be sent with your certificate request

A challenge password []:

An optional company name []:

users 目录下有 client.csr 文件生成。

* + 1. 使用我们私有的 CA key 为刚才的 key 签名

**# openssl ca -in /usr/local/nginx/ca/users/client.csr -cert /usr/local/nginx/ca/private/ca.crt -keyfile /usr/local/nginx/ca/private/ca.key -out /usr/local/nginx/ca/users/client.crt -config "/usr/local/nginx/ca/conf/openssl.conf"**
输出
Using configuration from /usr/local/nginx/ca/conf/openssl.conf

Check that the request matches the signature

Signature ok

The Subject's Distinguished Name is as follows

countryName :PRINTABLE:'cn'

stateOrProvinceName :ASN.1 12:'BeiJing'

localityName :ASN.1 12:'BeiJing'

organizationName :ASN.1 12:'SinoCache'

organizationalUnitName:ASN.1 12:'DevOpt'

commonName :ASN.1 12:'\*.zhoufengjie.cn'

emailAddress :IA5STRING:'tyumen@zhoufengjie.cn'

Certificate is to be certified until Mar 24 23:37:35 2016 GMT (365 days)

Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y

Write out database with 1 new entries

Data Base Updated


两次都输入 y，users 目录下有 client.crt 文件生成。

### 2.3.5 将证书转换为大多数浏览器都能识别的 PKCS12 文件

**#openssl pkcs12 -export -clcerts -in /usr/local/nginx/ca/users/client.crt -inkey /usr/local/nginx/ca/users/client.key -out /usr/local/nginx/ca/users/client.p12**
要求输入 client.key 的 pass phrase，输入 2.3.2 步输入的 pass phrase 并回车后：


要求输入 Export Password，这个是客户端证书的保护密码，在客户端安装证书的时候需要输入这个密码。我还是输入 Password。users 目录下有 client.p12 文件生成。

# apache配置

SSL 的目的是为了保证网络通信的安全以及数据完整性。所以，如果 tomcat 前面有了 apache 作前端代理，那就没有理由再在 apache 和 tomcat 之间进行无加密传输，毕竟二者处于同一内网。

如上图所示，客户端通过 SSL 请求过来的访问被反向代理 apache 接收，apache 结束了 SSL 并将请求以纯 HTTP 提交 tomcat。apache 配置 ssl.conf修改增加如下内容：

**ServerName www.zhoufengjie.cn**

**LogLevel info**

**SSLEngine on**

**SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL**

**SSLCertificateFile /usr/local/nginx/ca/server/server.crt**

#SSLCertificateFile 指定服务器使用的证书。该指令用于指定服务器持有的X.509证书(PEM编码)，其中还可以包含对应的RSA或DSA私钥。如果其中包含的私钥已经使用密语加密，那么在 Apache启动的时候将会提示输入密语。如果服务器同时使用了RSA和DSA两种证书，那么该指令可以使用两次来分别指定两种证书的位置。

**SSLCertificateKeyFile /usr/local/nginx/ca/server/server.key**

#SSLCertificateKeyFile 指定了服务器私钥文件。如果SSLCertificateFile指定的服务器证书文件中不包含相应的私钥，那么就必须使用该指令，否则就不需要使用。我们反对在服务器证书中包含私钥，正确的做法应该是将证书和私钥分开在不同的文件中。如果私钥文件已经使用密语加密，那么在Apache启动的时候将会提示输入密语。如果服务器同时使用了RSA和DSA两种证书，那么该指令可以使用两次来分别指定两种私钥的位置。

**SSLCertificateChainFile /usr/local/nginx/ca/private/ca.crt** #如果有很久级证书就好好配这个。

#SSLCertificateChainFile指定了一个多合一的CA证书：由直接签发服务器证书的CA证书开始，按证书链顺序回溯，一直到根CA的证书结束，这一系列的CA证书(PEM格式)就构成了服务器的证书链。这个指令也可以由SSLCACertificatePath指令代替，或者两个一起使用，用于明确的创建服务器的证书链。这个证书链将被与服务器证书一起发送给客户端。这有利于避免在执行客户端认证时多个CA证书之间出现混淆或冲突。因为虽然将服务器证书链上的某个CA证书放到SSLCACertificatePath目录中对于证书链结构没什么影响，但是由这个CA签发的客户端证书也会在执行客户端认证的时候同时被认可，这通常不是你期望的结果。

**SSLCACertificateFile /usr/local/nginx/ca/private/ca.crt**

#SSLCACertificateFile 该指令用于客户端认证。这个指令指定了一个多合一的CA证书，只有持有这些CA所签发证书的客户端才允许访问。这个所谓"多合一"证书文件其实就是将多个PEM格式的证书按照优先级顺序放置在同一个文件中而已。这个指令也可以由SSLCACertificatePath指令代替，或者两个一起使用。

**SSLVerifyClient require**

#要求进行客户端认证。

**SSLVerifyDepth 1**

**SSLOptions +StdEnvVars**

**ProxyRequests Off**

**<Proxy balancer://mycluster>**

 **BalancerMember http://192.168.231.224:8080**

 **BalancerMember http://192.168.231.224:8080**

**</Proxy>**

**ProxyPass / balancer://mycluster/**

# Tomcat 配置

Nginx 反向代理 HTTP 不需要更改 Tomcat 配置。与 HTTP 代理不同的是，这里需要通过更改 tomcat 的配置文件来告诉它前面的 HTTPS 代理。将 %tomcat%/conf/ 以下部分：

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

 connectionTimeout="20000"

 redirectPort="8443"/>

修改为

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

 connectionTimeout="20000"

 redirectPort="8443"

 scheme="https"

 proxyName="www.zhoufengjie.cn"

 proxyPort="443" />

# 配置验证

* 1. Tomcat 重启验证

重启 tomcat，后台日志没问题，也可以看到小猫界面。

* 1. Nginx 重启验证

先关闭运行中的 nginx，如果你已经开启了的话。

* 1. 客户访问 ssl 验证

谷歌浏览器使用 https 访问原有项目 https://www.zhoufengjie.cn，通过host固定是 Nginx 所在服务器，提示 400 Bad Request(No required SSL certificate was sent)：

这是因为 https 双向验证需要客户端安装证书。windows os 下拿到生成的证书 client.p12，直接双击它，进入 "证书导入向导"：

一步一步下一步，最后直接点击 "完成" 按钮完成证书导入。


重启谷歌浏览器，再次访问 https://www.zhoufengjie.cn，浏览器要求我们选择证书：
 
选中刚才安装好的那个证书(www.zhoufengjie.cn)，点击 "确定"，提示 "隐私设置错误"：

这是因为我们服务器用的是自己签发的证书。选择继续访问，守得云开见月明，终于看到久违了的项目登录页面，成功了：

可以点击浏览器输入框左侧的小锁图标查看我们导入的客户端证书相关信息：


# 双向 SSL 验证的集群配置

Nginx 做负载均衡器，多台 Tomcat 进行集群，在双向 SSL 验证的环境下的配置，和纯 HTTP 没有多少差异。只需注意以下细节即可。

* 1. Nginx 配置

调整upstreame即可。具体负载算法以及服务器列表依照实际情况自行调整。

* 1. Tomcat 配置

所有设备，一样CentOS 下对 Nginx + Tomcat 配置 SSL 实现服务器 / 客户端双向认证，至此结束。