OpenSSL生成根证书CA及签发子证书

原创 摆渡者 大型网站 2016/04/01 15:34 阅读数 8.8W

本文被收录于专区
服务端

进入专区参与更多专题讨论

系统：CentOS7 32位

目标：**使用OpenSSL生成一个CA根证书，并用这个根证书颁发两个子证书server和client。**

先确保系统中安装了OpenSSL，若没安装，可以通过以下命令安装：

```
sudo yum install openssl
```

修改OpenSSL的配置

安装好之后，定位一下OpenSSL的配置文件openssl.cnf：

```
locate openssl.cnf
```

```
[user@centos1 ~]$ locate openssl.cnf
/etc/pki/tls/openssl.cnf
/usr/share/man/man5/openssl.cnf.5ssl.gz
[user@centos1 ~]$
```

如图，我这里的目录是/etc/pki/tls/openssl.cnf。

修改配置文件，修改其中的dir变量，重新设置SSL的工作目录：

```
35 ####################################################################
36 [ ca ]
37 default_ca      = CA_default            # The default ca section
38
39 ####################################################################
40 [ CA_default ]
41
42 #dir            = /etc/pki/CA          # Where everything is kept
43 dir             = /home/user/CA        # Where everything is kept
44 certs           = $dir/certs           # Where the issued certs are kept
45 crl_dir         = $dir/crl             # Where the issued crl are kept
46 database        = $dir/index.txt       # database index file.
```

由于配置文件中，dir变量下还有几个子文件夹需要用到，因此在自定义的文件夹下面也创建这几个文件夹或文件，它们是：

```
[user@centos1 CA]$ ls
[user@centos1 CA]$ mkdir certs
[user@centos1 CA]$ mkdir newcerts
[user@centos1 CA]$ mkdir private
[user@centos1 CA]$ mkdir crl
[user@centos1 CA]$ touch index.txt
[user@centos1 CA]$ echo 01>serial
[user@centos1 CA]$ ls
certs  crl  index.txt  newcerts  private  serial
```

certs——存放已颁发的证书

newcerts——存放CA指令生成的新证书

private——存放私钥

crl——存放已吊销的证书

index.txt——OpenSSL定义的已签发证书的文本数据库文件，这个文件通常在初始化的时候是空的

serial——证书签发时使用的序列号参考文件，该文件的序列号是以16进制格式进行存放的，该文件必须提供并且包含一个有效的序列号

生成证书之前，需要先生成一个随机数：

```
openssl rand -out private/.rand 1000
```

该命令含义如下：

rand——生成随机数

-out——指定输出文件

1000——指定随机数长度

生成根证书

a).生成根证书私钥(pem文件)

OpenSSL通常使用PEM（Privacy Enhanced Mail）格式来保存私钥，构建私钥的命令如下：

```
openssl genrsa -aes256 -out private/cakey.pem 1024
```

```
[user@centos1 CA]$ openssl genrsa -aes256 -out private/cakey.pem 1024
Generating RSA private key, 1024 bit long modulus
..++++++
...++++++
e is 65537 (0x10001)
Enter pass phrase for private/cakey.pem:
Verifying - Enter pass phrase for private/cakey.pem:
```

该命含义如下：

genrsa——使用RSA算法产生私钥

-aes256——使用256位密钥的AES算法对私钥进行加密

-out——输出文件的路径

1024——指定私钥长度

b).生成根证书签发申请文件(csr文件)

使用上一步生成的私钥(pem文件)，生成证书请求文件(csr文件)：

```
openssl req -new -key private/cakey.pem -out private/ca.csr -subj \
"/C=CN/ST=myprovince/L=mycity/O=myorganization/OU=mygroup/CN=myname"
```

```
[user@centos1 CA]$ openssl req -new -key private/cakey.pem -out private/ca.csr -subj \
> "/C=CN/ST=myprovince/L=mycity/O=myorganization/OU=mygroup/CN=myname"
Enter pass phrase for private/cakey.pem:
[user@centos1 CA]$
```

该命令含义如下：

req——执行证书签发命令

-new——新证书签发请求

-key——指定私钥路径

-out——输出的csr文件的路径

-subj——证书相关的用户信息(subject的缩写)

c).自签发根证书(cer文件)

csr文件生成以后，可以将其发送给CA认证机构进行签发，当然，这里我们使用OpenSSL对该证书进行自签发：

```
openssl x509 -req -days 365 -sha1 -extensions v3_ca -signkey \
private/cakey.pem -in private/ca.csr -out certs/ca.cer
```

```
[user@centos1 CA]$ openssl x509 -req -days 365 -sha1 -extensions v3_ca -signkey \
> private/cakey.pem -in private/ca.csr -out certs/ca.cer
Signature ok
subject=/C=CN/ST=myprovince/L=mycity/O=myorganization/OU=mygroup/CN=myname
Getting Private key
Enter pass phrase for private/cakey.pem:
[user@centos1 CA]$
```

该命令的含义如下：

关于作者

摆渡者
程序员
关注 私信 提问

文章 179 经验值 1.1K 粉丝 361 关注 8

作者的专辑 全部

数据库 (1)
Mac相关 (1)
前端开发 (1)
Android (10)

x509——生成x509格式证书

-req——输入csr文件

-days——证书的有效期（天）

-sha1——证书摘要采用sha1算法

-extensions——按照openssl.cnf文件中配置的v3_ca项添加扩展

-signkey——签发证书的私钥

-in——要输入的csr文件

-out——输出的cer证书文件

之后看一下certs文件夹里生成的ca.cer证书文件：

```
[user@centos1 CA]$ ls certs
ca.cer
[user@centos1 CA]$
```


用根证书签发server端证书

和生成根证书的步骤类似，这里就不再介绍相同的参数了。

a).生成服务端私钥

```
openssl genrsa -aes256 -out private/server-key.pem 1024
```

b).生成证书请求文件

```
openssl req -new -key private/server-key.pem -out private/server.csr -subj \
"/C=CN/ST=myprovince/L=mycity/O=myorganization/OU=mygroup/CN=myname"
```

c).使用根证书签发服务端证书

```
openssl x509 -req -days 365 -sha1 -extensions v3_req -CA certs/ca.cer -CAkey private/cakey.pem \
-CAserial ca.srl -CAcreateserial -in private/server.csr -out certs/server.cer
```

这里有必要解释一下这几个参数：

-CA——指定CA证书的路径

-CAkey——指定CA证书的私钥路径

-CAserial——指定证书序列号文件的路径

-CAcreateserial——表示创建证书序列号文件(即上方提到的serial文件)，创建的序列号文件默认名称为-CA，指定的证书名称后加上.srl后缀

注意：这里指定的-extensions的值为v3_req，在OpenSSL的配置中，v3_req配置的basicConstraints的值为CA:FALSE，如图：

```
[ v3_req ]

# Extensions to add to a certificate request

basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
```

而前面生成根证书时，使用的-extensions值为v3_ca，v3_ca中指定的basicConstraints的值为CA:TRUE，表示该证书是颁发给CA机构的证书，如图：

```
[ v3_ca ]
# Extensions for a typical CA
# PKIX recommendation.

subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer

# This is what PKIX recommends but some broken software chokes on critical
# extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true
```

在x509指令中，有多种方式可以指定一个将要生成证书的序列号，可以使用set_serial选项来直接指定证书的序列号，也可以使用-CAserial选项来指定一个包含序列号的文件。所谓的序列号是一个包含一个十六进制正整数的文件，在默认情况下，该文件的名称为输入的证书名称加上.srl后缀，比如输入的证书文件为ca.cer，那么指令会试图从ca.srl文件中获取序列号，可以自己创建一个ca.srl文件，也可以通过-CAcreateserial选项来生成一个序列号文件。

用根证书签发client端证书

和签发server端的证书的过程类似，只是稍微改下参数而已。

a).生成客户端私钥

```
openssl genrsa -aes256 -out private/client-key.pem 1024
```

b).生成证书请求文件

```
openssl req -new -key private/client-key.pem -out private/client.csr -subj \
"/C=CN/ST=myprovince/L=mycity/O=myorganization/OU=mygroup/CN=myname"
```

c).使用根证书签发客户端证书

```
openssl x509 -req -days 365 -sha1 -extensions v3_req -CA certs/ca.cer -CAkey private/cakey.pem \
-CAserial ca.srl -in private/client.csr -out certs/client.cer
```

需要注意的是，上方签发服务端证书时已经使用-CAcreateserial生成过ca.srl文件，因此这里不需要带上这个参数了。

至此，我们已经使用OpenSSL自签发了一个CA证书ca.cer，并用这个CA证书签发了server.cer和client.cer两个子证书了：

```
[user@centos1 CA]$ ls certs
ca.cer  client.cer  server.cer
[user@centos1 CA]$
```

导出证书

a).导出客户端证书

```
openssl pkcs12 -export -clcerts -name myclient -inkey \
private/client-key.pem -in certs/client.cer -out certs/client.keystore
```

参数含义如下：

pkcs12——用来处理pkcs#12格式的证书

-export——执行的是导出操作

-clcerts——导出的是客户端证书，-cacerts则表示导出的是ca证书

-name——导出的证书别名

-inkey——证书的私钥路径

-in——要导出的证书的路径

-out——输出的密钥库文件的路径

b).导出服务端证书

```
openssl pkcs12 -export -clcerts -name myserver -inkey \
private/server-key.pem -in certs/server.cer -out certs/server.keystore
```

c).信任证书的导出

```
keytool -importcert -trustcacerts -alias www.mydomain.com \
-file certs/ca.cer -keystore certs/ca-trust.keystore
```

OpenSSL CA 证书

 举报

打赏 5 赞 20 收藏 分享

作者的其它热门文章

ELK(ElasticSearch, Logstash, Kibana)搭建实时日志分析平台
Spring中的AOP（五）——在Advice方法中获取目标方法的参数
很简单的在Ubuntu系统下安装字体和切换默认字体的方法
Spring+Log4j+ActiveMQ实现远程记录日志——实战+分析

osc_42283021
你好，请问 客户端证书和信任证书 有什么区别呢，没怎么懂，两个证书都需要安装在客户端吗？
03/12 09:46 回复 举报

simplesslife
原来是组织名，邮箱等填写的完全一样，导致错误，修改一下就行了
2016/10/24 11:13 回复 举报

更多评论