OpenSSL生成根证书CA及签发子证书

小龙猫 2016-05-09 19098浏览量

简介：

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——存放已吊销的整数

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:
```

该命令含义如下：

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：

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

安全 + 订阅
云安全开发者的大本营
其他文章
- 使用服务器ECS的感受
- ECS使用体验感悟
- ECS的初体验
- ECS云服务器实现24小时监听量化交易
- 服务器使用
展开
最新文章
- 我的ECS使用体验
- 使用服务器ECS的感受
- 云服务器ESC的使用体验
- ECS使用体验感悟
- ECS的初体验
展开
相关文章
- ECS体验感受
- 使用体验
- 飞天加速计划初体验
- 服务器使用
- 第一个博客关于阿里云
展开
近期公开课 全部
- 2021-09-13 19:00:00 云安全阿里云认证重点解读 阿里云
- 2020-04-01 19:00:00 如何将PostgreSQL与MySQL相结合，实现1+1>2的组合拳？ KB小秘书
- 2020-04-01 19:00:00 主流数据库容灾方案 KB小秘书

```
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，有多重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
```

推荐关注：
· 云栖大会开源引力峰会，震撼来袭！
· 问答官 SQL专场，小米行李箱免费领
· 1024八大训练营，致敬程序员，畅谈技术！
· 飒风者计划邀您入驻社区，精彩权益即刻享

分享：

版权声明：本文内容由阿里云实名注册用户自发贡献，版权归原作者所有，阿里云开发者社区不拥有其著作权，亦不承担相应法律责任。具体规则请查看《阿里云开发者社区用户服务协议》和《阿里云开发者社区知识产权保护指引》。如果您发现本社区中有涉嫌抄袭的内容，填写侵权投诉表单进行举报，一经查实，本社区将立刻删除涉嫌侵权内容。

算法　数据安全/隐私保护　Linux　网络安全

生成证书　ssh生成　openssl生成根证书　生成根证书证书　openssl SSL证书

评论

登录后可评论

游客tkrkswzybb5mu
2020-01-05

你好，有以下疑问： 1、https在创建密钥时应该使用的是非对称加密算法，即公钥+私钥，可为什么教程中对于CA、服务器、客户端的密钥创建只有私钥呢？![__20200105095837](https://yqfile.alicdn.com/13ef175da9e6d845a12d7334d3d8b97ba7f036b4.png)

如梦令
2019-09-11

-bash: keytool: 未找到命令

热门产品	短信服务	SSL 证书	CDN	容器服务 ACK	负载均衡 SLB	轻量应用服务器	Web 应用防火墙	日志服务 SLS	DDoS 防护
最新产品	视频直播	Quick BI 数据分析	弹性公网 IP	全站加速 DCDN	DataWorks	机器学习平台 PAI	智能对话机器人	云数据库 HBase 版	消息队列 Kafka 版
解决方案	容器镜像服务 ACR	云防火墙	AnalyticDB PG	DataV 数据可视化	专有云云盾	对象存储 OSS	消息队列 RabbitMQ	视频点播	安全管家服务
更多推荐	企业物联网平台	MaxCompute	RDS MySQL 版	Alibaba Cloud Linux	阿里巴乐 GDB	图像搜索	微消息队列 MQTT 版	智能外呼机器人	云服务器 ECS

关于阿里云2.0　法律声明及隐私权政策　廉正举报　联系我们

阿里巴巴集团　淘宝网　天猫　全球速卖通　阿里巴巴国际交易市场　1688　阿里妈妈　飞猪　阿里云计算　AliOS　万网　高德　UC　友盟　优酷　钉钉　支付宝　达摩院　淘宝海外　阿里云盘

© 2009-2021 Aliyun.com 版权所有 增值电信业务经营许可证：浙B2-20080101 域名注册服务机构许可：浙通网安[2019]94号 信部电函[2005]374号

浙公网安备 33010602009975号　浙B2-20080101-4

乘风者计划重磅来袭！