

计算机网络安全实验报告

OPENSSL 实验



姓名 田丰瑞

班级 软件 73 班

学号 2172213528

电话 18744296191

Email tianfr@stu.xjtu.edu.cn

日期 2020-5-12

声明

网络安全实验的所有代码已经全部开源于我的 GitHub（仅开源代码部分），项目地址：

<https://github.com/tianfr/Internet-Security-ExpCode/>

OPENSSL 实验

实验要求

实现 openssl 的签发证书功能。

OPENSSL 简介

OPENSSL 简介

openssl 是一个安全套接字层密码库，囊括主要的密码算法、常用密钥、证书封装管理功能及实现 ssl 协议。OpenSSL 整个软件包大概可以分成三个主要的功能部分：SSL 协议库 libssl、应用程序命令工具以及密码算法库 libcrypto。

SSL：Secure Socket Layer，安全套接字层协议，分为 SSLv2 和 SSLv3 两个版本，TLS 在 SSL3.0 基础之上提出的安全通信标准化版。主要是为了加密传输数据而产生的协议，能使用户/服务器应用之间的通信不被攻击者窃听，并且始终对服务器进行认证，还可选择对用户进行认证。

SSL 协议建立在可靠的传输层协议（TCP,UDP,SCTP）之上，应用层协议（HTTP,FTP,TELNET）在 SSL 之上，SSL 协议在应用层协议通信之前就已经完成了加密算法、服务器认证等工作。http 协议调用了 ssl 协议,那么他就变成了 https(http → ssl → https)。主要功能两个：

- a. 加密解密在网络中传输的数据包，同时保护这些数据不被修改和伪造。

应用层——>ssl 解密——>传输层

传输层——>ssl 加密认证——>应用层

- b. 验证网络对话中双方的身份。

SSL 握手协议：SSL 包含两个子协议，一个是包协议，一个是子协议。**包协议**说明 SSL 的数据包是如何封装的；**握手协议**说明通信双方协商通信双方决定使用什么算法及算法使用的 key。

下面事握手协议过程：

- 1 Client 向服务器发送自己支持的协议版本（如 TLS1.2）、client 生成的随机数、自己加密算法的一些配置。
- 2 Server 收到 Client 请求后向客户端 response:确认使用加密通信协议的版本、生成一个随机数、确认使用加密的方法、server certificate（服务器证书）。

- 3 Client 验证服务器证书，在确认无误后取出其公钥，并发送随机数 Pre-Master-Key（用于公钥加密）、编码变更通知（通信双方都用商定好的密钥进行通信;即随后的信息都将用双方商定好的加密方法和密钥发送。）
- 4 Server 验证完 client 的身份之后，用自己的私有密钥解密得到 pre-master-key 后,然后双方利用这个 pre-master key 来共同协商，得到 master secret。返回信息给 client
- 5 双方用 master 一起产生真正的 session key，这就是他们在剩下的过程中的对称加密的 key 了。这个 key 还可以用来验证数据完整性。双方再交换结束信息。握手结束。

证书 Certificate 和证书颁发机构 CA (certification authority)

证书是建立公共密钥和某个实体之间联系的数字化的文件。它包含的内容有：版本信息（X.509 也是有三个版本的）、系列号、证书接受者名称、颁发者名称、证书有效期、公共密钥、一大堆的可选的其他信息、CA 的数字签名。证书由 CA 颁发，由 CA 决定该证书的有效期，由该 CA 签名。每个证书都有唯一的系列号。证书的系列号和证书颁发者来决定某证书的唯一身份。常用的证书是采用 X.509 结构的，这是一个国际标准证书结构，

CA 是第三方机构，被你信任，由它保证证书的确发给了应该得到该证书的人。CA 自己有一个庞大的 public key 数据库，用来颁发给不同的实体。CA 也是一个实体，它也有自己的公共密钥和私有密钥。

加密算法

- 1) 对称加密：指加密和解密使用相同密钥的加密算法。对称加密算法的优点在于加解密的高速度和使用长密钥时的难破解性

常见的对称加密算法：DES、3DES、DESX、AES、RC4、RC5、RC6 等

- 2) 非对称加密：指加密和解密使用不同密钥的加密算法，也称为公私钥加密

常见的非对称加密算法：RSA、DSA（数字签名用）等

- 3) Hash 算法：Hash 算法它是一种单向算法，用户可以通过 Hash 算法对目标信息生成一段特定长度的唯一的 Hash 值，却不能通过这个 Hash 值逆向获得目标信息

常见的 Hash 算法：MD2、MD4、MD5、SHA、SHA-1 等

OPENSSL 实验环境配置

本次实验使用 UBUNTU16.04 环境，并通过 MOBAXTERM 远程连接服务器，实验利用 SCREEN 程序保证网络连接突然中断时的稳定性，并通过 SCRIPT 程序将实验 LOG 本地保存。

下面配置实验环境截图：

建立 screen 会话：

```
root@iZ2zed9y2x6c8rie9x45aaZ:~# screen -S openssl
```

保存实验会话记录：

```
root@iZ2zed9y2x6c8rie9x45aaZ:~# cd /
root@iZ2zed9y2x6c8rie9x45aaZ:~# script exp.log
Script started, file is exp.log
root@iZ2zed9y2x6c8rie9x45aaZ:~#
```

实验过程

使用 OPENSSL 生成 RSA 密钥对

使用 openssl 的私钥产生公钥前，需要了解以下几点：

- **key 算法**：openssl 支持生成 RSA，DSA，ECDSA 的密钥对，但是 RSA 是目前使用最普遍的。
- **Key 长度**：RSA 的 2048 是公认较比较安全的 key 长度。
- **密码 (Passphrase)**：在 key 上使用密码是一个可选值，但是一般都是强烈建议的（官网这样写的，实际项目中很多都没有设置口令），这样每次使用 key 文件时，都需要输入这个密码才能使用，增强了其安全性，但是随之而来的易用性也会变差。

使用 genrsa 命令来生成 RSA key。

生成私钥

使用命令：openssl genrsa -aes128 -out fd.key 2048。以下输入了为这个 key 值设置了密码，且密码使用 aes128 加密保存。

```
root@iZ2zed9y2x6c8rie9x45aaZ:~# cd /
root@iZ2zed9y2x6c8rie9x45aaZ:~# script exp.log
Script started, file is exp.log
root@iZ2zed9y2x6c8rie9x45aaZ:~# openssl genrsa -aes128 -out fd.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
e is 65537 (0x10001)
Enter pass phrase for fd.key:
Verifying - Enter pass phrase for fd.key:
root@iZ2zed9y2x6c8rie9x45aaZ:~#
```

这个 key 文件就是私钥文件。利用 cat 命令可以查看下文件内容：

```
Enter pass phrase for fd.key:
Verifying - Enter pass phrase for fd.key:
root@iZ2zed9y2x6c8rie9x45aaZ:/# cat fd.key
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,407C833BB878723AB895979952CABCA6

zymclSkd5v4xETCb7ZoTfr/C7asuIHxM5P/GXz7Z2kkBh6vnxm85wCEfBTOMQkt0
fCTeWo8y0G01R0/DS8WTfVu/ZsY1QEKKfVK0XAN5Fgo41l05cfb2ermx4D2QLoBs
vLcEuamzV9/0GPLjAAmg41Vgpu8eez42yhrn7gareZKQUV5m26a6PCa+mihij5BK
lULXhWpyS8PZRklncrFKx9mR2pyv1aMdIU0tjQrfnaF26+lox0egjAHqZwsYm3l
mXlb6ABibEBUK7CmK5a1N+SlnxkalvSE54Tu8WyCorwTwljLCWToZkYdw5CHRdG2
HS/jGRppNClj3cMfqsYVKS82kGcbPEmeudKpYlSVtPSAW3fUk9Ww4ZhdM/z91T9
xpjItX+dvWF1Bc3g3vywb1aayUeSF/u50N1RMRCtvSa6M0eJ8kAbQu3byzTlKR2J
KaF5e8x+mdW1X2bjlQdpjXhw1cwa6NhkyoUHUGSeQuJpRG3CrIYG4gYdVeRs5zl
taAZc8UXF8NFRfwer+C6oJ6V1L60p02bQmzWzflHauLSkNsKi7yAXGwLI0Qa+JmT
tDukVGm7+F9CEBTJGJyWk14VjXYSBDd+gEsDAYdq+K5A3F1fM/JULF0euLPyRF5/
OnpLI92GbrIcnnn0nhl27NlkoXn5QwXHeRlj7lkZDJAh1/Q0kfx+hocL+t00a3NT
hQhPSwZilbAF/TxJA8og0r7qPSjofRGoGPLs6nXFmYSyc0gXIinfp6tLR9ilbHU8
0lqM5Tec1qNrxrBGcd70lzquTfmSdTijohXhtGkCfRAoNTwqT/CK35gp4YPjX1lx
/zt+y2hxZiA6ECgdA4eA6oBVLGhfH1I97tnAGDiGqkp/wP5w2eppZVvAnmLn59Q
bxjyAJ8q2BVCeBJy4Vv2DZmibBE/k8dJz5uerDYwElvrLoL2vGPr0sWi4Q60QP4r
noMW6Ll43H9D/D2Ip0Cpxt/z87ovrQ4E0+SW6637p5zHL5BkazXES3JsRlbUnUkt
7EKz4hGX4T2RZTeCaTNPPFIo9hzddEkWx67crC8KTtf96ASkc2Q2sUyV+TLpFBN
ATLYq0SYV7YsK0W059y4KbVjBXvlc6ltl0eB+1W0aFCiIsf8hXk1RNKre1DG9kHL
ADLdeMuvJBZNR4SdWudovGDzMFfhXS1JJPLKnE3x3Fhf1kep9lUHSfZBvvBs5vx
GBV7vI5CBXeddgRF912YBN/kfLYBTjREva1x/jhjCkshpTSJ0H7M3nQGzGc/ra3G
jP1biMfibpgtNP+sQoT92SYSjkiJkKkEtJ0r7UYJvqbMMtgUbJx8VlrxGzBg/Z0j
w1qmmCwz0fXeSkLKQRUCj5XNp10I/PjMdvSCmiRP2jCWS3YolyVmADAZPtXJae0K
FywCjEhcNhoA234EsQgnU3vxInnCbkj6jRHv0PEIDA2eMnIUxABhHGTZmNzqTFD
IzDQR9hqaE3604kP+lFhn/CJq30I4Q2rNhIWl8Q7pbhF6uqqgMk6/kUzZgQbwTS
CHt/ghnHMn4WDac/KLeZqG4klA8otVYHNrmt/YFyAhVxGUvAKPl+N6c0Zu/kJ0HZ
-----END RSA PRIVATE KEY-----
root@iZ2zed9y2x6c8rie9x45aaZ:/#
```

生成公钥

使用命令：`openssl rsa -in fd.key -pubout -out fd-public.key`

```
root@iZ2zed9y2x6c8rie9x45aaZ:/# openssl rsa -in fd.key -pubout -out fd-public.key
Enter pass phrase for fd.key:
writing RSA key
root@iZ2zed9y2x6c8rie9x45aaZ:/#
```

利用 `cat` 查看这个 key 文件，就是公钥：

```
root@iZ2zed9y2x6c8rie9x45aaZ:/# cat fd-public.key
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA00tFdV0i1Aj5eBkAVN4g
K3+LON82U8LXGexx83dg2HRXbeX/pyUfdl0eYLnaH9Gv8F3wL/1NV/m9cJT48sVL
h4K5Voc4sJSXQKQ0Ief1zs1Y4hvXLd53E4AqFVuYs6BCuAneBKRfNFwmaJuggDjP
uaQ6CfIhmUEePGwd0rLWYjJGhXkvsMGyYjP38vp0T9bKDs9puhEWQpKxfWf5vhif
BswHE79Lad2QtCdiZ29I349ruW2rVoCgvvmNxGbXrMVV5V0ke80uXIHzWyyXTDp
uUvhJunEa1j1vZr0qHA03YKiocJ6Ev+H0GrgV+eNu4Cl8VD4Ernu8BZDSE4e71qJ
XQIDAQAB
-----END PUBLIC KEY-----
root@iZ2zed9y2x6c8rie9x45aaZ:/#
```

获取权威机构颁发证书

获取权威机构颁发的证书，需要先得到私钥的 key 文件（.key），然后使用私钥的 key 文件生成 sign req 文件（.csr），最后把 csr 文件发给权威机构，等待权威机构认证，认证成功后，会返回证书文件（.crt）。

生成私钥 KEY

与第二节使用 openssl 生成 RSA 密钥对的步骤 A 一致。使用命令：openssl genrsa -aes128 -out fd.key 2048

```
root@iZ2zed9y2x6c8rie9x45aaZ:/# openssl genrsa -aes128 -out fd.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for fd.key:
Verifying - Enter pass phrase for fd.key:
root@iZ2zed9y2x6c8rie9x45aaZ:/#
```

私钥的 KEY 文件生成 SIGN REQ 文件（.CSR）

生成 csr 文件时，需要填写一些关于待签人或者公司的一些信息，比如国家名，省份名，组织机构名，主机名，email 名，有些信息可以不填写，使用.标识。

使用命令：openssl req -new -key fd.key -out fd.csr。过程如下：

```
root@iZ2zed9y2x6c8rie9x45aaZ:/# openssl genrsa -aes128 -out fd.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for fd.key:
Verifying - Enter pass phrase for fd.key:
root@iZ2zed9y2x6c8rie9x45aaZ:/# openssl req -new -key fd.key -out fd.csr
Enter pass phrase for fd.key:
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) [AU]:GB
State or Province Name (full name) [Some-State]:.
Locality Name (eg, city) []:London
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Feisty Duck Ltd
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:http://www.feistyduck.com/
Email Address []:webmaster@feistyduck.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@iZ2zed9y2x6c8rie9x45aaZ:/#
```

把 csr 文件发给权威机构，等待权威机构认证，交费获取证书即可。

OPENSSL 生成 ROOT CA 及签发证书

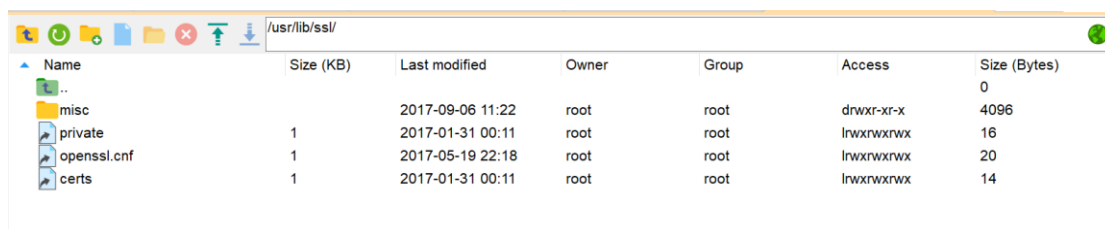
有时候, 使用 SSL 协议是自己内部服务器使用的, 这时可以不必去找第三方权威的 CA 机构做证书, 可以做自签证书 (自己创建 root CA (非权威)) 主要有以下三个步骤。

创建 OPENSSL.CNF 在使用 DEFAULT-CA 时需要使用的 SSL 的工作目录

查看 openssl 的配置文件

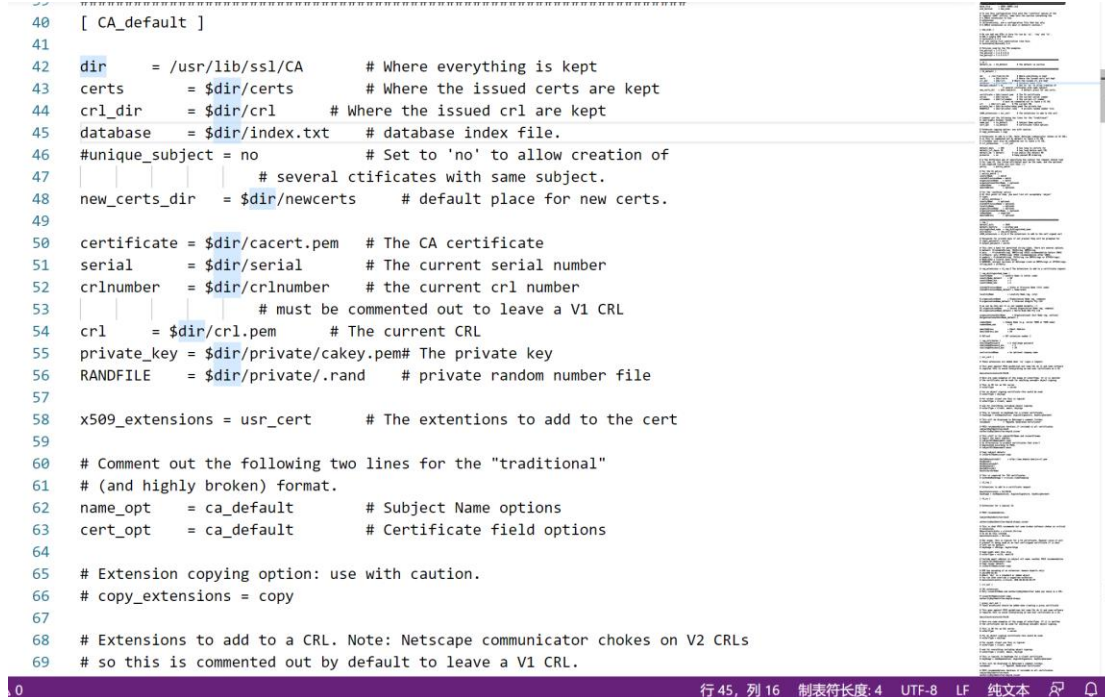
```
root@iZ2zed9y2x6c8rie9x45aaZ:/# openssl version -a
OpenSSL 1.0.2g  1 Mar 2016
built on: reproducible build, date unspecified
platform: debian-amd64
options: bn(64,64) rc4(16x,int) des(idx,cisc,16,int) blowfish(idx)
compiler: cc -I. -I. -I../include -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -m64 -DL_ENDIAN -g
-O2 -fstack-protector-strong -Wformat -Werror=format-security -Wdate-time -D_FORTIFY_SOURCE=2 -Wl,-Bsymbolic-functions -Wl,-z,relro
-Wa,-noexecstack -Wall -DMD32_REG_T=int -DOPENSSL_IA32_SSE2 -DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DS
HA1_ASM -DSHA256_ASM -DSHA512_ASM -DMD5_ASM -DAES_ASM -DVPAES_ASM -DBSAES_ASM -DWHIRLPOOL_ASM -DGHASH_ASM -DECP_NISTZ256_ASM
OPENSSLDIR: "/usr/lib/ssl"
```

找到 OPENSSLDIR: " /usr/lib/ssl/ " 的配置文件 openssl.cnf



Name	Size (KB)	Last modified	Owner	Group	Access	Size (Bytes)
misc		2017-09-06 11:22	root	root	drwxr-xr-x	0
private	1	2017-01-31 00:11	root	root	lrwxrwxrwx	16
openssl.cnf	1	2017-05-19 22:18	root	root	lrwxrwxrwx	20
certs	1	2017-01-31 00:11	root	root	lrwxrwxrwx	14

根据配置文件下的[CA_default]节点默认值, 创建对应文件夹和文件



```
40 [ CA_default ]
41
42 dir = /usr/lib/ssl/CA # Where everything is kept
43 certs = $dir/certs # Where the issued certs are kept
44 crl_dir = $dir/crl # Where the issued crl are kept
45 database = $dir/index.txt # database index file.
46 #unique_subject = no # Set to 'no' to allow creation of
47 # several certificates with same subject.
48 new_certs_dir = $dir/newcerts # default place for new certs.
49
50 certificate = $dir/cacert.pem # The CA certificate
51 serial = $dir/serial # The current serial number
52 crlnumber = $dir/crlnumber # the current crl number
53 # must be commented out to leave a V1 CRL
54 crl = $dir/crl.pem # The current CRL
55 private_key = $dir/private/cakey.pem # The private key
56 RANDFILE = $dir/private/.rand # private random number file
57
58 x509_extensions = usr_cert # The extensions to add to the cert
59
60 # Comment out the following two lines for the "traditional"
61 # (and highly broken) format.
62 name_opt = ca_default # Subject Name options
63 cert_opt = ca_default # Certificate field options
64
65 # Extension copying option: use with caution.
66 # copy_extensions = copy
67
68 # Extensions to add to a CRL. Note: Netscape communicator chokes on V2 CRLs
69 # so this is commented out by default to leave a V1 CRL.
```

```
root@iZ2zed9y2x6c8rie9x45aaZ:~# openssl version -a
OpenSSL 1.0.2g 1 Mar 2016
built on: reproducible build, date unspecified
platform: debian-amd64
options: bn(64,64) rc4(16x,int) des(idx,cisc,16,int) blowfish(idx)
compiler: cc -I. -I.. -I../include -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHAVE_DLFCN_H -m64 -DL_ENDIAN -g
-O2 -fstack-protector-strong -Wformat -Werror=format-security -Wdate-time -D_FORTIFY_SOURCE=2 -Wl,-Bsymbolic-functions -Wl,-z,relro
-Wa,-noexecstack -Wall -DMD32_REG_T=int -DOPENSSL_IA32_SSE2 -DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DS
HA1_ASM -DSHA256_ASM -DSHA512_ASM -DMD5_ASM -DAES_ASM -DVPAES_ASM -DBSAES_ASM -DWHIRLPOOL_ASM -DGHASH_ASM -DECP_NISTZ256_ASM
OPENSSLDIR: "/usr/lib/ssl"
root@iZ2zed9y2x6c8rie9x45aaZ:~# cd /usr/lib/ssl
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl# mkdir CA
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl# cd CA
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# mkdir certs
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# mkdir newcerts
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# mkdir private
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# mkdir crl
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# touch index.txt
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# echo 01>serial
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# ls
certs  crl  index.txt  newcerts  private  serial
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA#
```

其中，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：penSSL 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号参考文件，该文件的序号是以 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号。

执行完后，当前目录为：

```
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# ll
total 28
drwxr-xr-x 6 root root 4096 May 19 20:07 ./
drwxr-xr-x 4 root root 4096 May 19 20:05 ../
drwxr-xr-x 2 root root 4096 May 19 20:06 certs/
drwxr-xr-x 2 root root 4096 May 19 20:07 crl/
-rw-r--r-- 1 root root    0 May 19 20:07 index.txt
drwxr-xr-x 2 root root 4096 May 19 20:06 newcerts/
drwxr-xr-x 2 root root 4096 May 19 20:06 private/
-rw-r--r-- 1 root root    1 May 19 20:07 serial
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA#
```

生成 CA 根证书（ROOT CA 证书）

步骤：生成 CA 私钥（.key）-->生成 CA 证书请求（.csr）-->自签名得到根证书（.crt）（CA 给自己颁发的证书）。

```
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# Generate CA private key -->ca.key
Generate: command not found
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# openssl genrsa -out ca.key 2048
Generating RSA private key, 2048 bit long modulus
.....+++
e is 65537 (0x10001)
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# openssl req -new -key ca.key -out ca.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) [AU]:.
State or Province Name (full name) [Some-State]:.
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:tfr
Email Address []:tfr@tfr.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:openssl
An optional company name []:
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# openssl x509 -req -days 365 -in ca.csr -signkey ca.key -out ca.crt
Signature ok
subject=/O=Internet Widgits Pty Ltd/CN=tfr/emailAddress=tfr@tfr.com
Getting Private key
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA#
```

用自签根证书 CA.CRT 给用户证书签名

步骤：生成私钥（.key）-->生成证书请求（.csr）-->用 CA 根证书签名得到证书（.crt）

修改 openssl.cnf 文件：


```
[ policy_match ]
```

```
countryName      = optional
```

```
stateOrProvinceName = optional
```

```
organizationName  = optional
```

```
organizationalUnitName = optional
```

```
commonName        = supplied
```

```
emailAddress       = optional
```



```
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# openssl genrsa -out server.key 1024
Generating RSA private key, 1024 bit long modulus
.....+++++
.....+++++
e is 65537 (0x10001)
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# openssl req -new -key server.key -out 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) [AU]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA#
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA#
```

```
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key
Using configuration from /usr/lib/ssl/openssl.cnf
unable to load number from /usr/lib/ssl/CA/serial
error while loading serial number
139642581255832:error:0D066096:asn1 encoding routines:a2i_ASN1_INTEGER:short line:f_int.c:212:
```

发现出错，是 ANSI 编码问题。

```
root@iZ2zed9y2x6c8rie9x45aaZ:/usr/lib/ssl/CA# openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key
Using configuration from /usr/lib/ssl/openssl.cnf
Check that the request matches the signature
Signature ok
The commonName field needed to be supplied and was missing
```

至此 openssl 实验全部完成。

把 server.crt 以及 server.key 保存在服务器端等待程序加载使用；把 ca.key 保存在客户端，如果客户端需要验证服务器端发的证书时使用。

总结

本次实验加深了我对 openssl 的理解。Openssl 可以用于网站传输加密。我此次实验让我进一步了解 openssl 的具体结构，了解了 openssl 在 Ubuntu 上的运行具体环境与机制。