[**使用OpenSSL生成证书**](http://www.cnblogs.com/AloneSword/p/3809002.html)

**Step 1. Create key (password protected)**

**openssl genrsa -out prvtkey.pem 1024/2038                     (with out password protected)**

**openssl genrsa -des3 -out prvtkey.pem 1024/2048    (password protected)**

这个命令会生成一个1024/2048位的密钥。

**Step 2. Create certification request**

openssl req -new -key prvtkey.pem -out cert.csr

openssl req -new -nodes -key prvtkey.pem -out cert.csr  
  
这个命令将会生成一个证书请求，当然，用到了前面生成的密钥prvtkey.pem文件  
这里将生成一个新的文件cert.csr，即一个证书请求文件，你可以拿着这个文件去数字证书颁发机构（即CA）申请一个数字证书。CA会给你一个新的文件cacert.pem，那才是你的数字证书。  
  
**Step 3: Send certificate request to Certification Authority (CA)**

如果是自己做测试，那么证书的申请机构和颁发机构都是自己。就可以用下面这个命令来生成证书：  
openssl req -new -x509 -key prvtkey.pem -out cacert.pem -days 1095  
这个命令将用上面生成的密钥privkey.pem生成一个数字证书cacert.pem

cacert.pem 生成过程见“**OpenSSL建立自己的CA**”

有了privkey.pem和cacert.pem文件后就可以在自己的程序中使用了，比如做一个加密通讯的服务器  
  
-------------  
**OpenSSL建立自己的CA**

(1) 环境准备

首先，需要准备一个目录放置CA文件，包括颁发的证书和CRL(Certificate Revoke List)。  
这里我们选择目录 /var/MyCA。

然后我们在/var/MyCA下建立两个目录，certs用来保存我们的CA颁发的所有的证书的副本；private用来保存CA证书的私钥匙。

除了生成钥匙，在我们的CA体系中还需要创建三个文件。第一个文件用来跟踪最后一次颁发的证书的序列号，我们把它命名为serial，初始化为01。第二个文件是一个排序数据库，用来跟踪已经颁发的证书。我们把它命名为index.txt，文件内容为空。

$ mkdir /var/MyCA  
$ cd /var/MyCA  
$ mkdir certs private  
$ chmod g-rwx,o-rwx private  
$ echo "01" > serial  
$ touch index.txt

第三个文件是OpenSSL的配置文件，创建起来要棘手点。示例如下：

$ touch openssl.cnf

文件内容如下：

[ ca ]  
default\_ca = myca

[ myca ]  
dir = /var/MyCA  
certificate = $dir/cacert.pem  
database = $dir/index.txt  
new\_certs\_dir = $dir/certs  
private\_key = $dir/private/cakey.pem  
serial = $dir/serial

default\_crl\_days= 7  
default\_days = 365  
default\_md = md5

policy = myca\_policy  
x509\_extensions = certificate\_extensions

[ myca\_policy ]  
commonName = supplied  
stateOrProvinceName = supplied  
countryName = supplied  
emailAddress = supplied  
organizationName= supplied  
organizationalUnitName = optional

[ certificate\_extensions ]  
basicConstraints= CA:false

我们需要告诉OpenSSL配置文件的路径，有两种方法可以达成目的：通过config命令选项；通过环境变量OPENSSL\_CONF。这里我们选择环境变量的方式。

$ OPENSSL\_CONF=/var/MyCA/openssl.cnf"  
$ export OPENSSL\_CONF

(2) 生成根证书 (Root Certificate)

我们需要一个证书来为自己颁发的证书签名，这个证书可从其他CA获取，或者是自签名的根证书。这里我们生成一个自签名的根证书。

首先我们需要往配置文件里面添加一些信息，如下所示，节名和命令行工具的命令req一样。我们把所有必要的信息都写进配置，而不是在命令行输入，这是唯一指定X.509v3扩展的方式，也能让我们对如何创建根证书有个清晰的把握。

[ req ]  
default\_bits = 2048  
default\_keyfile = /var/MyCA/private/cakey.pem  
default\_md = md5  
prompt = no  
distinguished\_name = root\_ca\_distinguished\_name  
x509\_extensions = root\_ca\_extensions  
[ root\_ca\_distinguished\_name ]  
commonName = My Test CA  
stateOrProvinceName = HZ  
countryName = CN  
emailAddress = test@cert.com   
organizationName = Root Certification Authority  
[ root\_ca\_extensions ]  
basicConstraints = CA:true

万事俱备，我们可以生成根证书了。注意设置好环境变量OPENSSL\_CONF。

$ openssl req -x509 -newkey rsa -out cacert.pem -outform PEM -days 356

注：“-days 356“控制有效期限为365天，默认为30天。

验证一下我们生成的文件。

$ openssl x509 -in cacert.pem -text -noout

(3) 给客户颁发证书

在给客户颁发证书之前，需要客户提供证书的基本信息。我们另外开启一个终端窗口，使用默认的OpenSSL配置文件(不要让之前的OPENSSL\_CONF干扰我们，那个配置是专门用来生成根证书的)。

命令和我们生成根证书的类似，都是req，不过需要提供一些额外的信息。如下：

$ openssl req -newkey rsa:1024 -keyout testkey.pem -keyform PEM -out testreq.pem -outform PEM

有两次提示要口令，第一次的口令用来加密私钥匙testkey.pem，第二次口令一般被OpenSSL忽略。  
结果生成两个文件：testkey.pem，私钥匙；testreq.pem，请求信息，其中包括公钥匙。

我们来看看testreq.pem都有哪些信息？

$ openssl req -in testreq.pem -text -noout

现在，我们可以把testreq.pem提交给我们的CA生成证书了。  
为了方便起见，我们假定testreq.pem在//var/MyCA/private/中。

$ openssl ca -in testreq.pem

有三次提示，一次是问你CA的私钥匙密码，两次是确认，输出的结果就是为客户颁发的证书。  
可以通过batch选项取消命令提示，可通过notext选项取消证书的输出显示。  
此外，还可以一次给多个客户颁发证书，方法是用 infiles选项替换in选项，不过这个选项必须放在最后，因为此后的任何字符均被处理为文件名称列表。

生成的证书放在certs目录，同时index.txt和serial的内容都发生了改变。