**一、详细介绍加密、解密技术**

现在的加密/解密技术主要有三种：**对称加密，非对称加密，和单向加密**

这三种加密解密技术的组合就是现在电子商务的基础，它们三个有各自最适合的领域，而且所要完成的功能也是不同的，大家都知道，只要我们连上互联网，那么我们就相当于大门洞开，我们的一些隐私，跟其他人对话内容等都有可能会被人窃听，最常见的比如man in the middle（中间人），它主要是因为双方身份无法验证的时候回话被劫持造成的，就是说通信双方都以为是在跟对方交流，其实内容都可能已经被这个“中间人”修改过，一些重要的信息也被这个“中间人”所获得，这对于现在的电子商务来说是致命的，所有我们必须找到一种解决方案来解决这个问题

**对称加密**：指的是加密方和解密方使用的是同一个密钥

优点：加密解密的速度很快

缺点：如果两个从未通信过的用户要进行通信的时候，该如何把解密的密钥传输给对方呢（密钥仍然要在网络上传输，所以密钥还是可能会被“中间人”截获），这是对称加密最大的缺点；

常见的对称加密算法有：

    DES：使用56位的密钥，2000年的时候被人破解了，所以现在基本不再使用

    AES：高级加密标准，可以使用128，129，256三种长度密钥

其他的还有blowfish,Twofish和RC6,IDEA(商业算法)，CAST5等

**非对称加密**：非对称加密方式解决了对称加密的缺陷，它的加密和解密密钥是不同的，比如对一组数字加密，我们可以用公钥对其加密，然后我们想要将其还原，就必须用私钥进行解密，公钥和私钥是配对使用的，常见的非对称加密算法有：

 RSA：既可以用来加密解密，又可以用来实现用户认证

 DSA：只能用来加密解密，所以使用范围没有RSA广

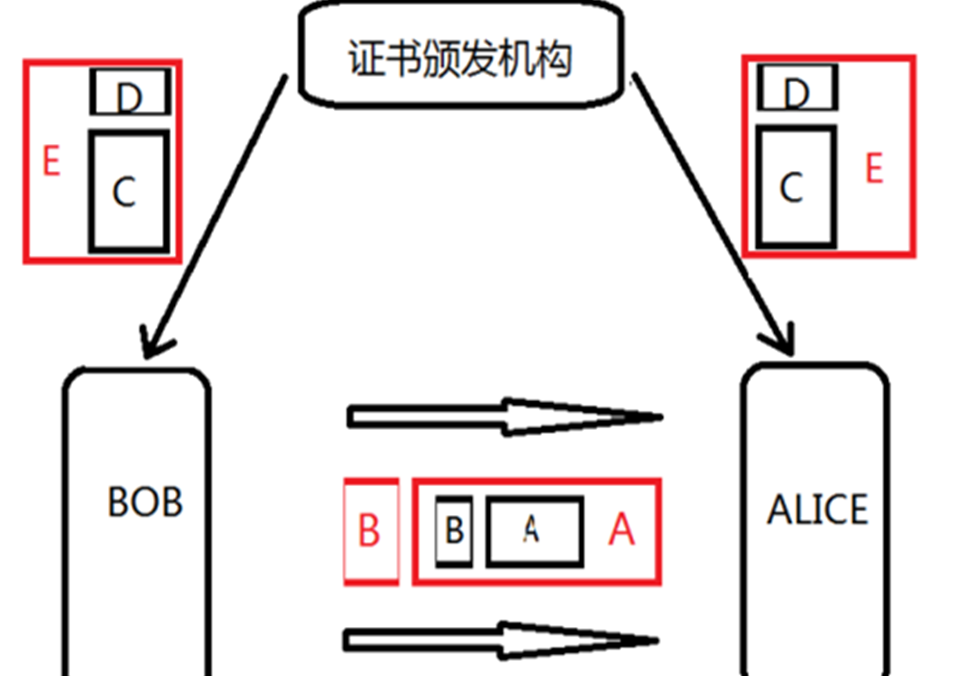
非对称加密长度通常有512，1024，2048，4096位，最常用的就是2048位，长度固然可以增加安全性但是需要花费很长时间来进行加密/解密，和对称加密相比，加密/解密的时间差不多是对称加密的1000倍，所以我们通常用其作为用户认证，用对称加密来实现数据的加密/解密

**单项加密**：单向加密就是用来计算一段数据的特征码的，为了防止用户通过“暴力破解”的方式解密，所以单向加密一般具有“雪崩效应”就是说：只要被加密内容有一点点的不同，加密所得结果就会有很大的变化。单项加密还有一个特点就是无论被加密的内容多长/短，加密的结果（就是提取特征码）是定长的，用途：用于验证数据的完整性，常用的单项加密算法

MD5:这种加密算法固定长度为128位

SHA1：这种加密算法固定长度是160位

下图是帮助我们理解加密算法在Internet上的使用所作的说明：



**随机数+Key值+公钥---->私钥**

**解释如下**，先说BOB和ALICE通信阶段

黑框A：表示要传输的数据  
黑框B：就是单项加密对这段数据提取的特征码，这段特征码同时运用了非对称加密，具体过程是用BOB的私钥加密，传输给ALICE，只要到达后ALICE能解密，表明对方确实是BOB。这一过程同时起到了用户认证和数据完整性的校验。黑框B又称为数字签名  
红框A：这一阶段会生成一段很长的随机数（密钥）然后配合对称加密算法对黑框A和黑框B加密,但是我们如何把加密的密钥传输给ALICE呢？这就要用到红框B了  
红框B：这一阶段是用ALICE的公钥加密这串随机数（对称加密阶段的密钥），ALICE接受到数据后如果能用自己私钥解密，那就证明接受者确实ALICE

**加密过程：**  
第一步：用单向加密算法提取数据（黑框A）的特征值  
第二步：用自己的私钥加密这段特征值形成黑框B  
第三步：用对称加密算法，对黑框A和黑框B来加密，得到红框A  
第四步：用ALICE的公钥来加密第三步所用的密钥，得到红框B

**解密过程：**  
第一步：ALICE用自己的私钥解密红框B得到对称加密的密钥  
第二步：用这个密钥解密红框A内容  
第三步：用BOB的公钥解密黑框B，如果能成功，说明发送方确实是BOB，这就完成了身份验证（解密后会得到一串数据的特征值）  
第四步：用同样的单项加密算法来对这段数据提取特征值，如果和第三步的特征值一样，说明这段数据是完整的，这就完成了数据完整性的校验

进行完上述内容厉害的读者会发现一个问题就是BOB和ALICE如何获得对方的公钥，或者说如何证明获得的公钥就是对方，这就需要引入另一方证书颁发机构CA，下面是对证书颁发机构跟BOB/ALICE之间的解释

黑框C：代表要颁发给BOB/ALICE的公钥，组织，地址等信息  
黑框D：是对黑框C进行单向加密后得到的数字签名，然后用自己的公钥对其加密，传输给BOB和ALICE，拿着这个证书颁发机构的公钥（这些证书颁发机构的公钥一般已经被microsoft事先放在windows里面，当然其他操作系统也是一样的）的BOB和ALICE如果能对这个证书进行解密，说明这个证书颁发机构不是冒充的  
红框E：表示颁发给BOB和ALICE的证书

**二、openssl使用详解**

在讲述之前首先要知道openssl中有如下后缀名的文件

.key格式：私有的密钥

.crt格式：证书文件，certificate的缩写

.csr格式：证书签名请求（证书请求文件），含有公钥信息，certificate signing request的缩写

.crl格式：证书吊销列表，Certificate Revocation List的缩写

.pem格式：用于导出，导入证书时候的证书的格式，有证书开头，结尾的格式

.crt.pem

常用证书协议

x509v3:IETF的证书标准

x.500:目录的标准

SCEP:  简单证书申请协议，用http来进行申请，数据有PKCS#7封装，数据其实格式也是PKCS#10的

PKCS#7:  是封装数据的标准，可以放置证书和一些请求信息

PKCS#10:  用于离线证书申请的证书申请的数据格式，注意数据包是使用PKCS#7封装这个数据

PKCS#12:  用于一个单一文件中交换公共和私有对象，就是公钥，私钥和证书，这些信息进行打包，加密放在存储目录中，CISCO放在NVRAM中，用户可以导出，以防证书服务器挂掉可以进行相应恢复。思科是.p12,微软是.pfx

**openssl主要组成部分**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | OpenSSL:主要有三个部分组成  [root@stu2~]# rpm -ql openssl  /usr/lib64/libcrypto.so.10        #libcrypto主要实现加密解密的工具  /usr/lib64/libssl.so.10           #主要实现ssl功能的  /usr/bin/openssl                  #多用途的命令行程序  [root@stu2~]# openssl version     #查看openssl的版本号  OpenSSL1.0.0-fips 29 Mar 2010 |

**对文件进行加密测试：**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | [root@stu2~]# cp /etc/fstab ./                     #复制文件到当前目录下  [root@stu2~]# ls  anaconda-ks.cfg  boot fstab  install.log  install.log.syslog  ks.cfg  [root@stu2~]# openssl enc -des3 -in fstab -e -out fstab.des3    #对文件进行加密测试输出到fstab.des3  enterdes-ede3-cbc encryption password:              #输入密码  Verifying- enter des-ede3-cbc encryption password:  #再次输入相同密码  [root@stu2~]# ls                                    #查看  anaconda-ks.cfg  fstab      install.log         ks.cfg  boot             fstab.des3  install.log.syslog  [root@stu2~]# cat fstab.des3  Salted\_\_)申.2t~?..??                                #后面的省略掉了 |

说明： openssl enc指定加密的类型，des3表示使用对称加密， -in指定要加密的文件，-e表示加密，-out指定加密后要保存的位置

**对文件进行解密测试**：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | [root@stu2~]# ls  anaconda-ks.cfg  fstab      install.log         ks.cfg  boot             fstab.des3  install.log.syslog  [root@stu2~]# rm -f fstab                   #删除原文件  [root@stu2~]# ls  anaconda-ks.cfg  boot fstab.des3  install.log  install.log.syslog  ks.cfg  [root@stu2~]# openssl enc -des3 -in fstab.des3 -d -out fstab  enterdes-ede3-cbc decryption password:      #输入加密时的密码  [root@stu2~]# ls  anaconda-ks.cfg  fstab      install.log         ks.cfg  boot             fstab.des3  install.log.syslog  [root@stu2~]# cat fstab                    #查看文件明文的列出了一部分  #  #/etc/fstab  #Created by anaconda on Sat Aug 10 08:33:02 2013 |

说明： openssl enc指定加密的类型，des3表示对称加密，-in 表示加密后的文件， -d表示解密， -out表示解密后要保存的位置，实验中为当前目录

**用单向加密算法获取文件的特征码**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | [root@stu2~]# rm -f fstab.des3  [root@stu2~]# ls  anaconda-ks.cfg  boot fstab  install.log  install.log.syslog  ks.cfg  [root@stu2~]# openssl dgst -md5 fstab #用md5单向加密算法获取文件的特征码  MD5(fstab)=70adea925c86b2fa91e6a4900b31508a  [root@stu2~]# openssl dgst -md5 -hex fstab #获取16进制的特征码，默认就是16进制的可以省略  MD5(fstab)=70adea925c86b2fa91e6a4900b31508a  [root@stu2~]# vim fstab #在文件第一行添加一个#  [root@stu2~]# openssl dgst -md5 -hex fstab #重新获取文件的特征码  MD5(fstab)=f8bd8f3c19697c85d4d683b0c4f0169b#和之前的特征码几乎完全不同，这就是所谓的雪崩效应 |

需要知道的是除了用上述命令获取文件的特征码之外，还可以通过md5sum fstal来获取，结果是相同的，

|  |  |
| --- | --- |
| 1  2  3  4 | [root@stu2~]# openssl dgst -md5 -hex fstab  MD5(fstab)=f8bd8f3c19697c85d4d683b0c4f0169b  [root@stu2~]# md5sum fstab  f8bd8f3c19697c85d4d683b0c4f0169b  fstab |

**测试当前主机所支持的加密算法的速率的**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | [root@stu2~]# openssl speed des-ede3  Doingdes ede3 for 3s on 16 size blocks: 3298403 des ede3's in 2.99s  Doingdes ede3 for 3s on 64 size blocks: 828631 des ede3's in 2.99s  Doingdes ede3 for 3s on 256 size blocks: 209735 des ede3's in 2.99s  Doingdes ede3 for 3s on 1024 size blocks: 52521 des ede3's in 3.00s  Doingdes ede3 for 3s on 8192 size blocks: 6691 des ede3's in 2.98s  OpenSSL1.0.0-fips 29 Mar 2010  builton: Thu Feb 21 23:42:57 UTC 2013  options:bn(64,64)md2(int) rc4(16x,int) des(idx,cisc,16,int) aes(partial) blowfish(idx)  compiler:gcc -fPIC -DOPENSSL\_PIC -DZLIB -DOPENSSL\_THREADS -D\_REENTRANT -DDSO\_DLFCN-DHAVE\_DLFCN\_H -DKRB5\_MIT -m64 -DL\_ENDIAN -DTERMIO -Wall -O2 -g -pipe -Wall-Wp,-D\_FORTIFY\_SOURCE=2 -fexceptions -fstack-protector--param=ssp-buffer-size=4 -m64 -mtune=generic -Wa,--noexecstack-DMD32\_REG\_T=int -DOPENSSL\_IA32\_SSE2 -DOPENSSL\_BN\_ASM\_MONT -DSHA1\_ASM-DSHA256\_ASM -DSHA512\_ASM -DMD5\_ASM -DAES\_ASM -DWHIRLPOOL\_ASM  The'numbers' are in 1000s of bytes per second processed.  type             16 bytes     64 bytes   256 bytes   1024 bytes   8192 bytes  desede3         17650.32k    17736.58k   17957.24k    17927.17k    18393.51 |

**计算密码值openssl passwd**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | [root@stu2~]# openssl passwd -1 -salt 12345678  Password:  $1$12345678$85L2ohc0YJ4r/l1LAa6co1  说明：其中-1表示基于md5的加密算法，-salt添加一些额外数（一般为8位），当前最好使用无规律的随机数，通过帮助文档可以进行查看man sslpasswd  [root@stu2~]# openssl rand -hex 4 #随机生产8位随机数  2c58639e  [root@stu2~]# openssl passwd -1 -salt `openssl rand -hex 4`#通过salt后加密计算密码值  Password:       #输入密码  $1$ca64684d$tQyI7a.yLwbQRMulUBMCI/       #生成的密码值           # man sslpasswd           # openssl passwd -1 -salt |

**生成RSA算法的私钥：**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | [root@stu2~]# openssl genrsa;  GeneratingRSA private key, 512 bit long modulus  .++++++++++++  ................++++++++++++  eis 65537 (0x10001)  -----BEGINRSA PRIVATE KEY-----  MIIBOQIBAAJBAJnZzY3oIvGIG8CJFoXp0QHd1VvyFGmSvIvDMAfFgmuzfL8E3uk0  joZDq6HEnNxiAl+KFdheZeB2qluD2lZy9dUCAwEAAQJAEjnvVCRwH1UUlJapCGxO  yn3RTPc829T6aURSaFlJviByWdgKLnOsr/HpBaF4/LGJXKcRdCRLC33CGBrSNFmc  8QIhAMskRtZ0xls0fjmCo8EnYOJBjmGjT5e8in19Ekc0KSy3AiEAweIl9k25q53J  XuFz/rf0ZWtjkH2sC72MgI7iIiGAvdMCIDz9Gzdvv83KKMtjaX6oL/I99F1kBpVr  ybh7QG2YP3RpAiAGRA7CBQUJHNJTNKoRA1TsqinPk6ApyPhvdy6rt7V3bQIgDqQ7  ZquRPTo8QGCGrevqZBUowjeqczOzQaWup9nMlNg=  -----ENDRSA PRIVATE KEY-----  [root@stu2~]# openssl genrsa 2048 > lian.key #可以指定生成私钥的位数，并重定向文件中  GeneratingRSA private key, 2048 bit long modulus  .............................................+++  .....+++  eis 65537 (0x10001)  [root@stu2~]# cat lian.key #可以进行查看 |

需要注意的是，私钥文件是至关重要的，不能让人们轻易的查看，因此我们可以对生成的私钥文件进行加密

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | [root@stu2~]# openssl genrsa -des3 2048 > lian.key #对生成的私钥文件进行单向加密  GeneratingRSA private key, 2048 bit long modulus  ......................................................................................................................................+++  ............................................+++  eis 65537 (0x10001)  Enterpass phrase:  Verifying- Enter pass phrase:  [root@stu2~]# cat lian.key #查看加密后的私钥文件  -----BEGINRSA PRIVATE KEY-----  Proc-Type:4,ENCRYPTED   #表示该私钥已经被加密  DEK-Info:DES-EDE3-CBC,3E3043F0CDBAA02E |

对私钥进行加密，虽然相对比较安全，但是对用户也带来了一定的麻烦，因为用户每次使用私钥时都需要输入密码，因此对私钥文件权限的设置是非常必要的

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | [root@stu2~]# (umask 077;openssl genrsa -out lian2.key 2048)  GeneratingRSA private key, 2048 bit long modulus  ..........................................................+++  ..+++  eis 65537 (0x10001)  [root@stu2~]# ll  total84  -rw-------.1 root root  2801 Aug 10 08:44anaconda-ks.cfg  drwxr-xr-x  3 root root 4096 Aug 11 12:32 boot  -rw-r--r--  1 root root 1051 Aug 21 15:34 fstab  -rw-r--r--.1 root root 43809 Aug 10 08:44 install.log  -rw-r--r--.1 root root  9963 Aug 10 08:41install.log.syslog  -rw-r--r--  1 root root 1388 Aug 11 12:32 ks.cfg  -rw-------  1 root root  1679 Aug 21 16:14 lian2.key  -rw-r--r--  1 root root  1743 Aug 21 16:06 lian.key |

需要注意的是：这里的umask设置权限一定要加上括号不然后面创建所有的文件都会变成此权限的，因为它是一直生效的；使用括号表示此命令在一个子shell中执行，完成子shell退出

**从私钥文件中提出公钥：**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | [root@stu2~]# openssl rsa -in lian2.key -pubout  writingRSA key  -----BEGINPUBLIC KEY-----  MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAuRr/Yy9xPDYowj2Cf+vS  apaSvOHmj33tpm80NAzB7IOvkC8oiulj6oUoowWT1xHEuGbyOJILQEBwN4/Jgm+S  IN6/r8gLssY4UBUj34T9ZHnAD0jEgPxRrQTiWSe1SlCStwkCR3RPlBUNXUAySa7X  St70Fg9gKIlz4HNMZRrXFgxgFSVE0i4sWFCwrDeIbAzTx2/cGJVtQcYSMrdaKDAH  NdgcqKNI0aQSwX9HvTxRtX4sfl7XbqXiyUUJVyuZ4jIxI7yKyii68y/Pk3mIz6mL  g5XOa15Jw9ggKHY5gd7g+nr4iioHMQQ/WI8Bargs1ftwESw5lKddQBnLgn2aTpxe  1QIDAQAB  -----ENDPUBLIC KEY----- |

**三、讲述如何申请证书：**

1)首先要生成一个私钥文件

2)制作一个证书签署请求

3)有CA负责签署证书

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | [root@stu2~]# openssl req -new -key lan.key -out lan.csr  Youare about to be asked to enter information that will be incorporated  intoyour certificate request.  Whatyou are about to enter is what is called a Distinguished Name or a DN.  Thereare quite a few fields but you can leave some blank  Forsome fields there will be a default value,  Ifyou enter '.', the field will be left blank.  -----  CountryName (2 letter code) [XX]:CN   #国家名  Stateor Province Name (full name) []:henan        #省份名称  LocalityName (eg, city) [Default City]:Zhengzhou         #城市名称  OrganizationName (eg, company) [Default Company Ltd]:lanlian   #公司名称  OrganizationalUnit Name (eg, section) []:tech     #部分名称  CommonName (eg, your name or your server's hostname) []:www.lanlian.com      #证书拥有者的名称，若使用域名访问这里使用域名，如用ip地址访问这里一定是Ip地址  EmailAddress []:  #邮箱地址，可以省略不写  Pleaseenter the following 'extra' attributes  tobe sent with your certificate request  Achallenge password []: #将请求加密起来输入密码，不想直接回车即可  Anoptional company name []: |

说明：openssl req 证书签署请求和证书生成工具，-key指定私钥文件的存放路径，-new指定证书申请，当前命令的后面可以加-days明确指定证书申请有效使用期限，-out 指定证书申请保存的位置通常以.csr结尾，要了解更多信息可以通过man req进行查看

**详解自建CA过程：**

首先查看解析配置文件

#cd/etc/pki/tls

#vimopenssl.cnf

列出了重要的一部分：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | ####################################################################  [ca ]  default\_ca      = CA\_default            # The default ca section  ####################################################################  [CA\_default ] #默认CA的配置  dir             = /etc/pki/CA           # Where everything is kept自建CA的工作目录  certs           = $dir/certs            # Where the issued certs are kept指定当前CA的证书存放位置  crl\_dir         = $dir/crl              # Where the issued crl are kept指定证书撤销列表所在的工作目录  database        = $dir/index.txt        # database index file.将签署的证书制作成索引保存下来也就是数据库文件  #unique\_subject= no                    # Set to 'no' toallow creation of                                          #several ctificates with same subject.  new\_certs\_dir   = $dir/newcerts         # default place for new certs.新签证书的位置  certificate     = $dir/cacert.pem       # The CA certificate  CA自己证书位置  serial          = $dir/serial           # The current serial number已签署的证书，序列号  crlnumber       = $dir/crlnumber        # the current crl number已吊销证书的个数                                          # mustbe commented out to leave a V1 CRL  crl             = $dir/crl.pem          # The current CRL当前的证书吊销文件  private\_key     = $dir/private/cakey.pem# The private key证书颁发机构自己的私钥文件  RANDFILE        = $dir/private/.rand    # private random number file |

#cd/etc/pki/CA

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | [root@stu2CA]# (umask 077;openssl genrsa -out private/cakey.pem 2048) #为CA生成一个私钥  [root@stu2CA]# ls private/  cakey.pem   [root@stu2 CA]# openssl req -new -x509 -keyprivate/cakey.pem -out cacert.pem -days 3656  Youare about to be asked to enter information that will be incorporated  intoyour certificate request.  Whatyou are about to enter is what is called a Distinguished Name or a DN.  Thereare quite a few fields but you can leave some blank  Forsome fields there will be a default value,  Ifyou enter '.', the field will be left blank.  -----  CountryName (2 letter code) [XX]:CN  Stateor Province Name (full name) []:henan  LocalityName (eg, city) [Default City]:zhengzhou  OrganizationName (eg, company) [Default Company Ltd]:lanlian  OrganizationalUnit Name (eg, section) []:tech  CommonName (eg, your name or your server's hostname) []:www.magedu.com  EmailAddress []:  [root@stu2CA]# ls  cacert.pem  certs crl  newcerts  private  [root@stu2CA]# touch serial index.txt  [root@stu2CA]# echo 01 > serial  [root@stu2CA]# ls  cacert.pem  certs crl  index.txt  newcerts private  serial |

**进行证书签署：**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | [root@stu2~]# openssl ca -in lan.csr -out lan.crt -days 3656  Usingconfiguration from /etc/pki/tls/openssl.cnf  Checkthat the request matches the signature  Signatureok  CertificateDetails:          Serial Number: 1 (0x1)          Validity              Not Before: Aug 21 09:20:01 2013GMT              Not After : Aug 25 09:20:01 2023GMT          Subject:              countryName               = CN              stateOrProvinceName       = henan              organizationName          = lanlian              organizationalUnitName    = tech              commonName                = www.lanlian.com          X509v3 extensions:              X509v3 Basic Constraints:                  CA:FALSE              Netscape Comment:                  OpenSSL Generated Certificate              X509v3 Subject Key Identifier:                 6C:0F:54:43:9A:73:B2:74:FF:2E:59:16:E6:FD:D7:EE:C3:7D:0A:0B              X509v3 Authority Key Identifier:                  keyid:59:56:89:9D:46:10:65:8B:F0:B4:EB:ED:C2:21:A0:CC:E4:9F:F4:FE  Certificateis to be certified until Aug 25 09:20:01 2023 GMT (3656 days)  Signthe certificate? [y/n]:y  1out of 1 certificate requests certified, commit? [y/n]y  Writeout database with 1 new entries  DataBase Updated |