Secure and Dependable Systems Jacobs University Bremen April 16th, 2020
Arsenij Percov

Problem Sheet 7

Problem 7.1

Solution:

a)
We are going to use openssl
First, generate the private key using the following command:

```
openssl genrsa -aes256 -out private.key 8912
```

Genrsa is a tool that generates keys. We provide -aes256 as argument in order to encrypt the key (will require pass phrase), then we provide the path for output file (–out) and final number is the bit length of the key.
The openssl asks us to provide pass phrase

```
Enter pass phrase for private.key:
```

Then, it asks to enter it again, to verify it:

```
Verifying - Enter pass phrase for private.key:
```

After, this we have generated private key and stored it in a file private.key. It is 8912 bit long.
Now let's generate the public key, using our private key, for that we are going to use openssl again. Type the following command:

```
 openssl rsa -in private.key -pubout -out public.key
```

rsa defines the program to use, -in provides the private key file path, -pubout indicates that we want to save public key to separate file, and finally –out provides the path for the output file.
Since we have encrypted the private key, the openssl asks us for the pass phrase.

```
Enter pass phrase for private.key:
```

Finally, it saves it to the output file, and we have the pair of keys.

b)
We are going to use openssl req to generate the request.
We will need only private key. Type:

```
openssl req -new -key private.key -out certificate_req.csr
```

-new generate new request, -key takes path to the key we are going to use, and -out provides path to the output file.
As you can remember, we encrypted the private key, so openssl asks us again to provide pass phrase to it:

```
Enter pass phrase for private.key:
Can't load /home/apercov/.rnd into RNG
140092041929152:error:2406F079:random number generator:RAND_load_file:Cannot open file:../
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]:
```

Now we need to provide:[https://www.ssl.com/how-to/manually-generate-a-certificate-signing-request-csr-using-openssl/]
The Country Name - two-letter country code.
The State or Province Name - full name(do not use an abbreviation).
The Locality Name - your city or town.
Organization Name - your company or organization.
Organizational Unit Name -(optional) your department or section.
Common Name - Fully Qualified Domain Name (FQDN) of the website this certificate will protect or your name. Email address - (optional)(You can hit Enter to skip forward.) The challenge password -(optional).
An optional company.

This is example of my input:

```
-----
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:Bremen
Locality Name (eg, city) []:Bremen
Organization Name (eg, company) [Internet Widgits Pty Ltd]:APercov
Organizational Unit Name (eg, section) []:APercov crypto test
Common Name (e.g. server FQDN or YOUR name) []:Arsenij
Email Address []:a.percov@jacobs-university.de

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:1234567890
An optional company name []:
```

Now, that we generated the csr certificate, it's the time to see the content of it. Type:

```
openssl req -in certificate_req.csr -noout -text
```

Again, we are using req utility, we provide -in input file path, so we do not create new certificate, then we provide option -noout so no encoded output is generated(we only want to see text, not to encode it back to csr), and last we provide option text that prints out certificate request in text form.
Now, this is the output of our certificate request:

```
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = DE, ST = Bremen, L = Bremen, O = APercov, OU = APercov crypto test, C
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (8912 bit)
                Modulus:
                    00:bf:2e:ff:6a:20:7f:b2:98:66:ad:46:ee:3a:fa:
                    21:87:43:27:ba:89:af:50:99:b3:33:5b:b8:04:ad:
                    0f:48:a2:58:6d:a2:94:e1:27:b4:54:07:db:c3:e7:
                    c1:28:9c:0d:10:60:40:f3:e2:73:a9:dc:f2:2b:b4:
                    bd:8c:7d:7e:2c:1f:aa:b1:3b:1c:d7:cb:c3:8f:fe:
                    86:bf:5b:81:e0:eb:62:48:35:b8:44:df:59:57:e6:
                    89:26:72:ff:96:aa:74:88:df:77:79:f2:61:db:89:
                    e6:f8:cc:75:d4:60:0a:b1:8c:8e:a7:1f:ab:70:2d:
                    47:cd:7e:c7:90:84:bf:06:65:ca:74:54:bc:48:70:
                    12:87:3b:4f:91:65:4b:ad:c0:7d:15:d4:26:ae:2a:
                    37:80:16:a3:20:2b:e5:7d:28:bd:8c:de:55:77:dc:
                    8c:4c:80:ee:13:85:c5:ce:fe:bb:07:7f:8b:e0:63:
                    f1:46:d5:4f:80:07:5c:c9:58:d1:30:58:f5:68:71:
                    5d:00:b9:c3:0f:f3:41:6a:7d:6f:e6:56:97:3f:22:
                    42:6a:de:ea:33:55:e1:bb:48:6f:d0:46:cb:1d:1f:
                    4b:cc:db:e5:9a:08:c7:b7:4e:20:36:84:3e:c1:f0:
                    82:d2:d0:37:c0:16:fa:94:25:1a:1c:7a:b5:fd:8e:
                    06:77:e1:db:df:18:b7:b3:6c:7b:ff:a6:23:0c:f1:
```

```
                    d7:47:cb:c8:3a:4e:82:8e:78:d3:8a:e7:65:0f:98:
                    86:e5:88:de:46:cf:ca:08:8c:ff:34:00:69:0c:aa:
                    6e:2b:3a:8a:c8:13:15:d3:00:fd:15:c6:83:2a:98:
                    ae:06:86:8c:07:e8:b3:41:0d:61:1c:06:5c:7c:de:
                    35:a3:88:94:6f:e6:d2:35:c9:fb:f4:46:b1:e3:7d:
                    dc:f4:89:b0:f6:18:62:f9:49:61:61:f2:5e:3f:8f:
                    c7:92:34:88:64:14:4e:b0:fb:dc:dd:4e:a3:02:77:
                    11:01:eb:5c:4c:66:5e:22:3f:e4:e3:55:af:2d:e8:
                    34:ec:ff:d9:9d:a2:91:40:44:ed:e9:84:9f:4d:ff:
                    cf:68:cf:ca:15:1f:da:c6:f3:9e:66:f0:f7:10:85:
                    e7:67:35:f9:1f:cb:a1:eb:cf:22:63:7a:32:b5:90:
                    6a:f5:03:3f:32:a8:96:5e:28:15:78:3a:89:44:bf:
                    1b:83:cc:1e:ff:f3:a7:b5:17:14:82:32:88:19:85:
                    c6:38:e9:7d:92:5d:dc:56:b5:5c:c9:f2:d3:6d:d1:
                    f5:36:98:0f:98:cc:4a:c0:91:ef:17:e4:69:e6:c4:
                    bb:d9:bf:5b:e9:6f:1b:6e:11:a5:a8:ee:e1:9f:35:
                    cb:91:b5:5c:09:b4:d3:ed:cf:f9:6c:3b:bb:f8:42:
                    02:94:05:81:90:43:ea:af:e5:3e:59:6e:99:6d:21:
                    02:f3:5f:92:62:e5:35:84:c5:ce:a0:d6:e9:57:14:
                    b6:96:49:b4:79:ac:43:3f:28:fa:26:f2:c5:ff:83:
                    e7:0a:ac:a9:5f:53:b9:a4:01:3a:b0:28:7a:7a:19:
                    11:f6:84:9b:c5:31:41:cf:48:c9:7d:ec:20:12:e8:
                    9e:6a:ff:30:11:59:e3:b6:5f:68:34:69:a7:6c:e0:
                    b2:69:a9:8a:88:bf:5b:64:8d:61:cf:60:c1:fa:d1:
                    61:48:b0:c1:70:77:88:ac:57:af:4b:04:b7:71:f5:
                    34:d3:67:32:36:35:d0:93:8f:f0:cd:bc:23:1f:55:
                    b4:17:6f:e8:c4:7d:cc:04:82:8e:8e:12:87:bb:69:
                    5c:f0:e0:15:07:74:3e:33:97:27:9e:03:ac:30:fd:
                    68:73:06:7c:fe:ac:b4:d4:9e:38:d4:ca:8c:b6:76:
                    47:ea:2e:70:db:1a:69:e0:a4:97:46:f1:88:ec:e5:
                    f9:d9:7c:4a:06:e0:5f:c6:ae:7e:8c:7d:29:88:36:
                    f8:71:c7:ba:6f:4a:20:44:2d:15:da:b4:34:c7:4f:
                    97:57:b1:11:dc:b4:ba:21:1d:07:4f:b2:f9:22:70:
                    9c:37:93:cf:07:00:ba:36:c2:f9:65:1a:1f:9a:c9:
                    7a:51:8e:51:d4:95:cb:34:84:f6:74:89:2d:d5:2f:
                    13:09:59:4d:8e:77:75:9f:d7:13:22:3c:c2:d0:6a:
                    07:87:41:50:9c:57:d5:d1:be:16:51:d0:82:f1:39:
                    5c:41:74:0b:2a:b7:18:d9:48:7d:bc:60:8a:2b:48:
                    12:60:a3:22:8f:ca:1b:ac:81:da:54:10:d8:0f:33:
                    75:56:eb:d0:a2:48:bb:16:8c:ec:a5:52:fb:a8:25:
                    2a:21:1c:8b:a7:e5:a1:8b:37:43:b3:48:09:34:76:
                    2b:d2:46:4f:2b:1d:6a:3f:1f:fb:f2:01:f2:1d:1e:
                    fa:60:a1:e0:57:63:92:a8:5b:e2:1f:78:98:c1:17:
                    b6:41:b0:05:87:25:26:d5:ad:d2:34:7f:cb:6d:b6:
                    46:a6:72:f5:bb:90:e1:ba:35:67:9d:ab:b0:d3:00:
                    41:8c:8d:6c:23:76:82:83:d5:b3:7f:58:42:72:ac:
                    b8:d6:2f:6e:b8:ce:08:23:98:52:69:e3:78:b4:2e:
                    20:cd:c5:ec:1a:e2:18:28:8e:c9:59:82:37:ec:36:
                    b6:a4:cf:a2:c8:07:c4:2c:51:4f:95:39:73:e5:4a:
                    dd:5c:9e:13:b4:b2:d1:10:6e:0a:cb:c7:e8:26:ba:
                    82:ff:56:fa:e6:ad:5b:00:05:17:8a:8a:f2:34:d1:
                    f3:e4:a3:06:9b:f8:7d:7d:3a:62:ef:ff:84:41:e8:
                    33:97:8a:ed:d1:3d:23:41:23:cc:42:1d:ff:14:30:
                    7b:81:ec:8a:b2:b9:59:81:7f:1e:01:e9:16:ed:93:
                    49:dd:cb:7b:e0:10:19:68:1f:4e:f5:98:cb:f4:4d:
                    3c:a3:a8:cd:24:c1:43:4b:04:b7:1b:b6:0a:37:9e:
                    47:a6:3f:2e:6f
                Exponent: 65537 (0x10001)
        Attributes:
            challengePassword        :1234567890
    Signature Algorithm: sha256WithRSAEncryption
         70:bc:d9:58:96:c9:6d:62:3b:bb:29:d0:a6:14:de:52:80:aa:
         21:9b:a8:c1:89:a3:fd:08:44:67:c1:c9:68:15:92:d5:30:c1:
         76:70:16:db:4d:de:d5:64:c7:cc:a8:04:bb:7b:9b:80:a8:87:
```

```
14:c8:8b:3a:cb:3c:78:64:bd:ce:07:81:96:19:c9:8f:f1:26:
32:be:9c:ce:1c:6f:72:8a:e1:81:75:fe:62:45:79:8d:fd:4d:
7c:1d:53:5f:8a:0f:7b:8d:f4:59:18:d3:6c:ee:a0:9a:22:f5:
0b:69:1f:e1:a8:20:3d:8d:f5:26:cc:26:a7:8e:9e:b6:65:06:
96:53:d8:a5:4e:d5:6b:c7:df:c6:99:d3:6c:b7:dc:d9:a3:31:
12:80:ae:51:a5:7c:94:1b:4a:2e:86:58:fc:ac:53:6a:3a:81:
7b:07:db:eb:e9:c2:fe:b5:eb:13:8d:5b:02:15:6f:a1:0f:e2:
f6:84:dd:8a:8a:e0:6a:cd:88:b7:ed:ff:1a:71:a5:48:25:29:
e3:19:5f:98:f9:ac:47:a7:8d:41:21:dc:a3:d3:8b:ea:51:f6:
e9:3a:fa:ac:7b:7c:e1:39:70:ca:a1:18:06:32:2e:f8:21:56:
e1:7e:97:44:f2:74:ea:93:e2:6a:b8:82:09:7b:e4:6b:51:7a:
dc:db:d4:8a:01:20:eb:5d:1f:fc:0d:98:9b:ff:8c:a2:56:ee:
25:8d:10:a9:74:36:e8:65:2d:34:b7:40:d9:af:c1:e9:51:9d:
21:f5:ef:be:d9:04:63:a0:e1:0c:a6:3f:f4:53:b9:cf:55:67:
24:01:ac:e1:16:19:b3:7a:01:a9:25:9b:ca:66:4e:52:51:8e:
11:51:77:48:40:ea:a6:5d:f6:82:4a:f3:eb:03:c8:35:62:61:
49:6f:8a:a4:0a:9f:2d:df:e4:c1:97:9e:64:6c:66:33:e7:a6:
44:88:7f:36:6d:e8:09:80:c6:c9:4c:cc:15:5b:53:6f:43:bf:
4d:98:2a:b3:f6:9e:e8:0f:61:7c:6a:a3:fe:f8:79:0a:5a:07:
16:06:66:bf:09:53:c3:a3:c9:18:3c:1d:de:21:f5:e6:5a:44:
0a:b7:9b:74:e7:94:56:13:85:b8:68:60:33:a9:fb:57:da:ec:
5e:08:da:6c:f6:a4:24:83:52:cc:14:c2:f3:ad:f0:83:2b:55:
f9:e3:f3:17:e1:a6:98:26:8f:43:06:79:73:7a:78:92:1e:4b:
2a:cc:f3:b7:3f:92:6b:14:f6:d2:4c:cf:6b:36:bb:65:37:4d:
7b:c1:14:fc:a0:0b:14:5e:cf:cf:5f:c0:9f:09:b7:44:e1:f9:
40:5b:e8:25:b9:99:30:fe:36:d2:1d:be:5a:d8:4e:17:b7:8b:
54:a9:b2:67:b5:26:3c:16:a0:e7:ae:63:76:41:a7:0f:ed:98:
77:ea:1b:4b:04:b9:8e:23:da:35:22:ec:85:25:97:7d:cb:06:
d7:f2:78:73:03:3b:ed:f9:d8:98:0a:2b:d4:5e:a7:db:84:f8:
19:57:37:11:49:88:96:9f:a6:f9:f0:87:35:d4:c6:a7:ad:93:
1b:c4:9e:50:c8:82:a0:2a:35:28:bd:4f:0c:41:33:90:ed:ed:
1e:1e:c8:31:cb:df:32:8d:fd:7a:26:c0:c3:4a:63:74:b8:28:
bf:af:b3:35:30:eb:2e:d0:a0:59:7a:02:60:c9:db:f5:0c:47:
c7:af:7d:10:e5:a6:bf:3d:c5:b8:3e:52:0f:80:a4:f3:4c:2f:
1d:6d:a1:2d:57:2f:ae:48:8b:c1:61:4a:4b:79:8d:70:88:0f:
cb:c3:d3:1d:3c:34:06:12:10:1c:9f:1f:93:10:b3:ea:0a:63:
90:e4:fb:0c:3d:85:b2:8e:c5:8a:0f:97:ab:c8:23:00:37:ff:
a1:41:e5:9c:f6:93:5c:c3:a2:5a:97:8d:37:52:3f:77:e9:a6:
1f:8d:ce:55:91:e4:6b:04:c2:e6:69:45:14:27:df:db:6c:23:
7b:d6:b8:65:16:1d:03:27:5c:d7:5b:47:2e:a7:6f:be:fa:d8:
b3:9c:b0:1f:b0:a0:0a:d6:28:0c:43:84:d8:21:63:04:99:0e:
30:90:21:1b:3e:bc:cb:c3:bf:20:a2:a7:45:62:61:63:24:47:
f0:18:61:d2:12:0c:67:68:c6:16:87:ad:5e:49:84:a1:6a:bd:
29:28:0d:d5:2c:41:7d:a0:57:ad:37:f4:ae:52:1a:3d:39:7c:
2c:89:3d:cc:fc:c3:b5:00:0f:76:cc:49:b8:5c:89:25:5e:6f:
c8:7e:ac:51:30:d3:fb:3b:bb:88:a7:a8:b3:44:4a:e6:64:c7:
2b:3f:d9:f3:74:d8:bf:87:a1:90:bd:71:a2:95:47:a0:92:1f:
ea:51:59:83:9c:1b:52:a2:18:ca:fa:e3:34:83:f9:49:02:88:
4d:24:ca:ab:bb:d4:4a:66:fa:ef:14:7d:cb:21:4e:79:ad:22:
7d:fc:b2:60:1b:81:73:22:53:66:9d:0d:e2:22:ff:d2:ad:5b:
29:72:c2:5f:be:76:38:88:c3:92:44:d9:d7:a5:70:c3:1a:f1:
c3:f1:b6:bd:8d:96:91:ec:a4:aa:9f:ee:e5:db:86:ab:45:37:
d0:0a:5c:92:a2:5b:69:2b:2f:a3:41:ad:66:e7:f4:81:f3:34:
2b:b5:63:1c:95:a6:aa:8c:28:38:c4:90:3e:19:a6:ce:7f:02:
88:12:60:6a:e1:1b:6d:91:d5:88:75:f9:c2:34:d9:33:da:20:
17:41:a9:c3:b9:c2:0b:cf:f7:df:20:37:e8:df:81:d7:e6:b1:
71:a5:67:a0:e0:3a:62:ba:90:1c:55:4c:61:7b:7b:4f:67:d3:
dd:a7:76:e2:ae:b6:b6:02:d1:9d:6a:e8:e7:4b:14:3b:e0:44:
5b:d5:05:6a:25:88:9e:13:7a:1e:3b:d2:3e:ba:8c:0c
```

c)
This is perl code of the -newca option for /usr/lib/ssl/misc/CA.pl

```
# create the directory hierarchy
    mkdir ${CATOP}, $DIRMODE;
    mkdir "${CATOP}/certs", $DIRMODE;
    mkdir "${CATOP}/crl", $DIRMODE ;
    mkdir "${CATOP}/newcerts", $DIRMODE;
    mkdir "${CATOP}/private", $DIRMODE;
    open OUT, ">${CATOP}/index.txt";
    close OUT;
    open OUT, ">${CATOP}/crlnumber";
    print OUT "01\n";
    close OUT;
    # ask user for existing CA certificate
    print "CA certificate filename (or enter to create)\n";
    $FILE = "" unless defined($FILE = <STDIN>);
    $FILE =~ s{\R$}{};
    if ($FILE ne "") {
        copy_pemfile($FILE,"${CATOP}/private/$CAKEY", "PRIVATE");
        copy_pemfile($FILE,"${CATOP}/$CACERT", "CERTIFICATE");
    } else {
        print "Making CA certificate ...\n";
        $RET = run("$REQ -new -keyout"
                . " ${CATOP}/private/$CAKEY"
                . " -out ${CATOP}/$CAREQ $EXTRA{req}");
        $RET = run("$CA -create_serial"
                . " -out ${CATOP}/$CACERT $CADAYS -batch"
                . " -keyfile ${CATOP}/private/$CAKEY -selfsign"
                . " -extensions v3_ca $EXTRA{ca}"
                . " -infiles ${CATOP}/$CAREQ") if $RET == 0;
        print "CA certificate is in ${CATOP}/$CACERT\n" if $RET == 0;
    }
```

Let’s try using it to create a new ca:

```
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]:DE
State or Province Name (full name) [Some-State]:Bremen
Locality Name (eg, city) []:Bremen
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Arsenij Percov
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:Arsenij
Email Address []:a.percov@jacobs-university.de

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:1234567890
An optional company name []:
==> 0
====
====
openssl ca  -create_serial -out ./demoCA/cacert.pem -days 1095 -batch -keyfile ./demoCA/pr
Using configuration from /usr/lib/ssl/openssl.cnf
Can't load /home/apercov/.rnd into RNG
139868971499968:error:2406F079:random number generator:RAND_load_file:Cannot open file:../
Enter pass phrase for ./demoCA/private/cakey.pem:
Can't open ./demoCA/index.txt.attr for reading, No such file or directory
139868971499968:error:02001002:system library:fopen:No such file or directory:../crypto/bi
139868971499968:error:2006D080:BIO routines:BIO_new_file:no such file:../crypto/bio/bss_fi
```

```
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number:
            77:a6:eb:21:db:ce:ac:29:94:b4:de:aa:72:2d:d8:aa:80:a3:f6:9b
        Validity
            Not Before: Apr 16 19:20:27 2020 GMT
            Not After : Apr 16 19:20:27 2023 GMT
        Subject:
            countryName               = DE
            stateOrProvinceName       = Bremen
            organizationName          = Arsenij Percov
            commonName                = Arsenij
            emailAddress              = a.percov@jacobs-university.de
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                5C:B1:49:75:29:1E:EA:46:B9:34:56:86:24:4D:65:F6:8C:0A:7F:1A
            X509v3 Authority Key Identifier:
                keyid:5C:B1:49:75:29:1E:EA:46:B9:34:56:86:24:4D:65:F6:8C:0A:7F:1A

            X509v3 Basic Constraints: critical
                CA:TRUE
Certificate is to be certified until Apr 16 19:20:27 2023 GMT (1095 days)

Write out database with 1 new entries
Data Base Updated
==> 0
====
CA certificate is in ./demoCA/cacert.pem
```

First, it asks to provide CA certificate file name. We don't have one, so we just press enter.
Then, we are asked to provide the PEM pass phrase, and to verify it. We choose 1234567890 for this example.
Next we are asked about some additional details you can see above.
This create a new CA in the directory demoCA. CA's self-signed sertificate is in demoCA/cacert.pem and its key pair is in demoCA/private/cakey.pem.
d)
To sign the CSR we are going to use CA.pl -sign utility.
But first we need to go to CA.pl script, and chagne the variable NEWREQ to the path of our request, the run the script, enter pass code, and press yes 2 times.

```
sudo /usr/lib/ssl/misc/CA.pl -sign
====
openssl ca  -policy policy_anything -out newcert.pem  -infiles /home/apercov/Documents/Sec
Using configuration from /usr/lib/ssl/openssl.cnf
Can't load /home/apercov/.rnd into RNG
140211945349568:error:2406F079:random number generator:RAND_load_file:Cannot open file:../
Enter pass phrase for ./demoCA/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number:
            77:a6:eb:21:db:ce:ac:29:94:b4:de:aa:72:2d:d8:aa:80:a3:f6:9c
        Validity
            Not Before: Apr 16 19:34:39 2020 GMT
            Not After : Apr 16 19:34:39 2021 GMT
        Subject:
            countryName               = DE
            stateOrProvinceName       = Bremen
            localityName              = Bremen
            organizationName          = APercov
            organizationalUnitName    = APercov crypto test
            commonName                = Arsenij
            emailAddress              = a.percov@jacobs-university.de
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                57:2D:70:F0:AD:C6:62:3C:63:3E:64:5F:2B:48:76:91:6F:75:B3:7D
            X509v3 Authority Key Identifier:
                keyid:5C:B1:49:75:29:1E:EA:46:B9:34:56:86:24:4D:65:F6:8C:0A:7F:1A

Certificate is to be certified until Apr 16 19:34:39 2021 GMT (365 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
==> 0
====
Signed certificate is in newcert.pem
```

Problem 7.2

Solution:

a)
It is valid from:
Wednesday, September 19, 2018 at 3:31:30 PM
Until:
Monday, December 21, 2020 at 2:31:30 PM
It is issued by DFN-Verein Global Issuing CA.
Its certificate is valid from:
5/24/16, 1:38:40 PM GMT+2
until:
2/23/31, 12:59:59 AM GMT+1
It is issued by DFN-Verein Certification Authority 2.
Its certificate is valid from:
2/22/16, 2:38:22 PM GMT+1
until:
2/23/31, 12:59:59 AM GMT+1
And higher than it in the hierarchy is T-TeleSec GlobalRoot Class 2, with its certificate valid from:
10/1/08, 12:40:14 PM GMT+2
until:
10/2/33, 1:59:59 AM GMT+2

b)
OCSP stapling sends the request to CA. CA sends back current status of SSl certificate and time stamp. Then they are binded together by the server, while the user establishes connection with the server.
Then, the browser validates the timestamp and confirms that the certificate signed by the issuer. If the certificate is valid and trusted the page is loaded.
It takes the burden of verification from clients and gives it to the web hosting, and it makes it faster(can download once and cache, so we don't have to check with CA for every user) and more secure. [https://www.leaderssl.com/articl
ocsp-stapling-how-does-this-technology-work].

We can use following website to test it: https://www.ssllabs.com/ssltest/.
Run the test, and under certificate information find a field OCSP Must Staple. For both cnds and beadg it is not enabled, since Must Staple is No.

Problem 7.3

Solution:

a)
p = 191 , g =42, a = 27
Ya = 42**27 mod 191 = 67255970008406099921710928456387385568002048 mod 191 = 143

Ya is the number that Alice send to Bob.

b)
Kab = 143**xmod191 = 178
Trying to crack it with python script:
Too many options, but some of them (for chosen number):

```
 111
301
491
681
871
1061
1251
1441
1631
1821
2011
2201
2391
2581
2771
2961
3151
3341
3531
3721
3911
4101
4291
4481
4671
4861
5051
5241
5431
5621
5811
6001
6191
6381
6571
6761
6951
7141
7331
7521
7711
7901
8091
8281
8471
8661
8851
9041
9231
9421
9611
9801
9991
10181
10371
10561
10751
10941
```

11131
11321
11511
11701
11891
12081
12271
12461
12651
12841
13031
13221
13411
13601
13791
13981
14171
14361
14551