certificate

Aninda Maulik

November 2020

1 Introduction

A company X has several different geographical locations which are for example Paris, Brest and Marseille. These different sites are connected to the Internet, which allows staff to easily exchange information. However, all exchanges that have a official character do not pass through this network and are made in paper form using the mail laugh. The company finds that this is the only way secure enough to broadcast confidential data. On the other hand, she finds that this method is not sufficient efficient and too expensive.

New company information manager proposes to use World Wide Web to solve these cost and efficiency problems. To the remarks that his management makes to him concerning the confidentiality of exchanges and the authentication of personnel, he suggests using electronic certificates which, according to him, are a flexible and robust solution when they are properly implemented.

We will see in this lab how to create electronic certificates, how to use them and discover all the interest.

2 Environment

To realistically simulate our working environment, you have in the labtainer certificate of 4 machines connected to each other. The www machine will host the web server of the company, the CA machine will be dedicated to the certification authority (all creation and signing of certificates will be carried out there). The brest and marseille machines are will be machines used by goodguy and badboy users respectively. Machines are configured to use the services of a DNS and can therefore be contacted directly by name.

3 Creation of the certificate of the certification authority

Before creating the certificates, company X must first choose an authority to certification which will subsequently issue it with the various certificates it will need. For some reasons of its own, it has decided to create and manage its own certification authority.

A certification authority (CA) must have a particular certificate called a root certificate. This certificate has the particularity of being signed by its own private key, thus this way, everyone can check that it has been signed by the owner of the private key associated with the public key available in the certificate.

If the public key can be widely distributed, the private key is on the contrary confidential and must be very well protected: this is the element on which all the security of the system is based. If a third party becomes aware of the private key of the CA, he can then generate real fake certificates (certificates recognized as valid by all people trusting this CA, but that she will not have signed herself).

- The first task says, generate private and public key pair for Certification Authority computer by running the command which is as below in the CA window

CA $ openssl genrsa -des3 -out private / ca.key.pem 2048

- But in order to do this, we got to get into the root mode of the CA. so basically the first step becomes going to the root mode.
- below figure shows that we are in the root mode. When we are in the root mode, and enter the above mentioned command then it asks to enter a password. I chose the password, 88917071. This password is mentioned here so that it can be used later if requirement comes up. Now, after verifying this password, within the CA terminal; we're back into the root.

```
root@ca:~# openssl genrsa -des3 -out private/ca.key.pem 2048
Generating RSA private key, 2048 bit long modulus
.................................+++
.....................................................+++
e is 65537 (0x10001)
Enter pass phrase for private/ca.key.pem:
```

- The -des3 option means that the key pair is protected by an encryption using the algorithm DES3. To learn about the key, it is necessary to know the "Passphrase" (long password which can even contain spaces) which protects it.
- The -out option specifies the name of the file that contains the key pair.
- The last parameter is the size of the key in bits (2048 bits in our example).

- we would now try to display the different elements of the generated key by typing the following command:

CA $ openssl rsa -in private / ca.key.pem -text

```
root@ca:~# openssl rsa -in private/ca.key.pem -text
Enter pass phrase for private/ca.key.pem:
Private-Key: (2048 bit)
modulus:
    00:f6:53:49:f3:36:98:46:5b:f8:49:e7:47:2f:35:
    a6:89:bc:6b:2a:7b:69:b0:14:c6:8e:10:cb:d8:e8:
    74:c4:fb:54:b0:80:51:89:3c:cf:a8:19:a4:6e:fa:
    e2:1f:ac:28:d3:df:f3:25:02:ee:d7:dc:36:d4:8b:
    40:2a:59:94:d1:a1:fd:43:d5:13:f4:ee:14:33:08:
    d6:38:1a:d5:44:d0:74:b6:be:68:8c:fb:d4:f8:fc:
    c9:db:76:1a:bb:08:1f:21:3e:37:66:e9:3a:b2:56:
    f5:05:e2:39:07:b2:6b:58:18:e8:8e:b3:2c:8f:eb:
    60:2d:92:4c:2d:b4:98:73:34:bf:a9:16:5f:9f:5c:
    41:50:63:86:61:05:9b:5a:a3:77:3d:92:90:84:ed:
    af:af:b0:58:78:67:39:d8:d5:a7:6b:49:da:96:1c:
    64:00:15:64:24:d4:09:e1:f2:60:c2:23:22:de:ab:
    7c:7e:07:bc:f2:f9:c6:c7:af:72:0b:2b:f4:95:92:
    f4:0c:91:0b:38:cc:1e:9c:99:0d:ea:26:ee:d9:36:
    dc:fd:ac:a1:45:58:c8:61:24:ad:b6:c3:dc:85:a2:
    fd:e1:b8:4a:58:06:be:54:43:ba:cd:a1:e8:61:e9:
    a2:31:cb:5d:1b:b7:a8:fd:2c:32:23:ba:c2:b1:cf:
    07:23
publicExponent: 65537 (0x10001)
```

Q1: Is it possible to access this key? Explain.

Yes indeed

b) No

- The answer to the above question is yes, because the above command indeed shows elements of keys that have been generated by us.

Now that we have a couple of keys, we will generate the certificate of our certification authority. This will be a self-signed X.509 certificate (a certification root). To do this, we will sign the information concerning this associated certification authority. ciée to its public key using the private key. This certificate will be valid for 5 years (1825 days).

```
CA $ openssl req -config openssl.cnf -new -x509 -days 1825
-extensions v3_ca -key private / ca.key.pem -out ca.crt.pem
```

- Based on the above instruction and the command, which being executed generates a certificate for Certificate Authority computer.
- We would now attempt to fill in the field names based on the below instructions.

```
Country Name (2 letter code) [FR]:
Sate or Province Name (full name) [FRANCE]:
Organization Name (eg, company) [AA MOOC-IMT]:
Organizational Unit Name (eg, section) [NETWORK SECURITY]:
Common Name (eg server FQDN or YOUR name) []: MOOC_CA
Email Address []: CA@mooc.imt
```

- The above instruction is exactly followed and executed as below.

```
Country Name (2 letter code) [FR]:FR
FR
State or Province Name (full name) [France]:France
France
Organization Name (eg, company) [AA MOOC IMT]:AA MOOC IMT
AA MOOC IMT
Organizational Unit Name (eg, section) [NETWORK SECURITY]:NETWORK SECURITY
NETWORK SECURITY
Common Name (e.g. server FQDN or YOUR name) []:MOOC_CA
MOOC_CA
Email Address []:CA@mooc.imt
CA@mooc.imt
root@ca:~#
```

- The below question now arises

Q2: Why do I have to enter a passphrase when executing this command?

a) To prove that the private key used belongs to us.

b) To protect the generated certificate.

- The answer is to prove that the private key used belongs to us.

4 Creation of certificates of the server

We will start by creating a certificate for the company's web server: www.mooc.imt. The first step is the generation of a key pair of its own:

CA $ openssl genrsa -des3 -out server.key.pem 1024

- We would just attempt to follow the above instruction and run the command. This is being generated for the server by the Certification Authority.

```
root@ca:~# openssl genrsa -des3 -out server.key.pem 1024
Generating RSA private key, 1024 bit long modulus
.................++++++
.........................................++++++
e is 65537 (0x10001)
Enter pass phrase for server.key.pem:
Verifying - Enter pass phrase for server.key.pem:
```

- The below question arises now.

Q3: What does the parameter 1024 mean in the previous command?

a) Period of validity of the key pair.

b) Number of bits in the key pair.

- The answer to this question is the second option which says that the parameter 1024 means in the above command is number of bits in the key pair.

Now that we have this key, we will generate a certification request (Certificate Signing Request - CSR). This operation is done by the web server manager, just like the previous step, the private key of the server does not need to be known to the certification operator.

```
CA $ openssl req -config openssl.cnf -new -key server.key.pem
                 -out server.req.pem
```

- We would try to follow the above instruction and generate the certificate named

"server.cert.pem"

- This certificate is generated for the server computer by executing the above command given in the instruction.

```
req.pem:~# openssl req -config openssl.cnf -new -key server.key.pem -out server.
Enter pass phrase for server.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.
-----
Country Name (2 letter code) [FR]:FR
FR
State or Province Name (full name) [France]:France
France
Organization Name (eg, company) [AA MOOC IMT]:AA MOOC IMT
AA MOOC IMT
Organizational Unit Name (eg, section) [NETWORK SECURITY]:NETWORK SECURITY
NETWORK SECURITY
Common Name (e.g. server FQDN or YOUR name) []:www.mooc.imt
www.mooc.imt
Email Address []:server@mooc.imt
server@mooc.imt
root@ca:~#
```

- The below question arises now

Q4: What are the elements that can be found in the electronic certificate of a machine?

a) The name, URL of the machine.

b) The private key of the machine.

c) The signature of a trusted third party (certification authority).

d) The period of validity of the certificate.

- Now, the above question has not one answer but three and we would choose the answers based on our understanding as below

a) The name, URL of the machine.

c) The signature of a trusted third party (certification authority).

d) The period of validity of the certificate.

- We would now attempt to create the certificate for server www.mooc.int

```
root@ca:~# openssl x509 -inform pem -in ca.crt.pem -noout -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1141904851399649068 9 (0x9e789c1f6db687c1)
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=FR, ST=Rhone-Alphs, O=EMSE, OU=SCIENCE, CN=EMSE_SCIENCE/emailAddress=ca@emse.fr
        Validity
            Not Before: Dec 16 18:39:47 2019 GMT
            Not After : Dec 14 18:39:47 2024 GMT
        Subject: C=FR, ST=Rhone-Alphs, O=EMSE, OU=SCIENCE, CN=EMSE_SCIENCE/emailAddress=ca@emse.fr
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:f6:53:49:f3:36:98:46:5b:f8:49:e7:47:2f:35:
```

- Now, we attempted to copy the server's certificate, private key as well as the CA certificate which can be found in machine CA in the machine hosting to the Server computer.

```
            organizationName          = AA MOOC IMT
            organizationalUnitName    = NETWORK SECURITY
            commonName                = www.mooc.imt
            emailAddress              = server@mooc.imt
        X509v3 extensions:
            X509v3 Key Usage:
                Digital Signature, Key Encipherment
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
            X509v3 Subject Key Identifier:
                4E:21:A8:6E:51:0D:8A:76:E4:82:EF:35:D7:CD:A1:C3:E1:56:8B:E4
            X509v3 Authority Key Identifier:
                keyid:99:BE:1E:5B:4C:74:40:40:38:ED:01:28:C4:04:73:12:85:83:82:36

            Authority Information Access:
                CA Issuers - URI:http://ca.mooc.imt/cacert.pem
                OCSP - URI:http://ocsp.mooc.imt:8888

            X509v3 CRL Distribution Points:

                Full Name:
                  URI:http://ca.mooc.imt/crl.pem

Certificate is to be certified until Dec 27 01:10:00 2020 GMT (365 days)
Sign the certificate? [y/n]:y
y


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

- below shows the immediate next action

```
admin@www:~$ sudo scp authority@CA:server.key.pem /etc/apache2/certs/server.key.pem
authority@ca's password:
server.key.pem                                100%  963     0.9KB/s   00:00
admin@www:~$ sudo scp authority@CA:server.crt.pem /etc/apache2/certs/server.crt.pem
authority@ca's password:
server.crt.pem                                100%    0     0.0KB/s   00:00
admin@www:~$ sudo scp authority@CA:ca.crt.pem /etc/apache2/certs/ca.crt.pem
authority@ca's password:
ca.crt.pem                                    100% 1566     1.5KB/s   00:00
admin@www:~$ cd /etc/apache2/certs/
admin@www:/etc/apache2/certs$ ls
ca.crt.pem  server.crt.pem  server.key.pem
admin@www:/etc/apache2/certs$
```

- The three files are copied perfectly, so we attempted to launch the web service on machine WWW.

```
admin@www:~$ sudo systemctl start apache2
Enter passphrase for SSL/TLS keys for www.mooc.imt:443 (RSA): **********
admin@www:~$
```

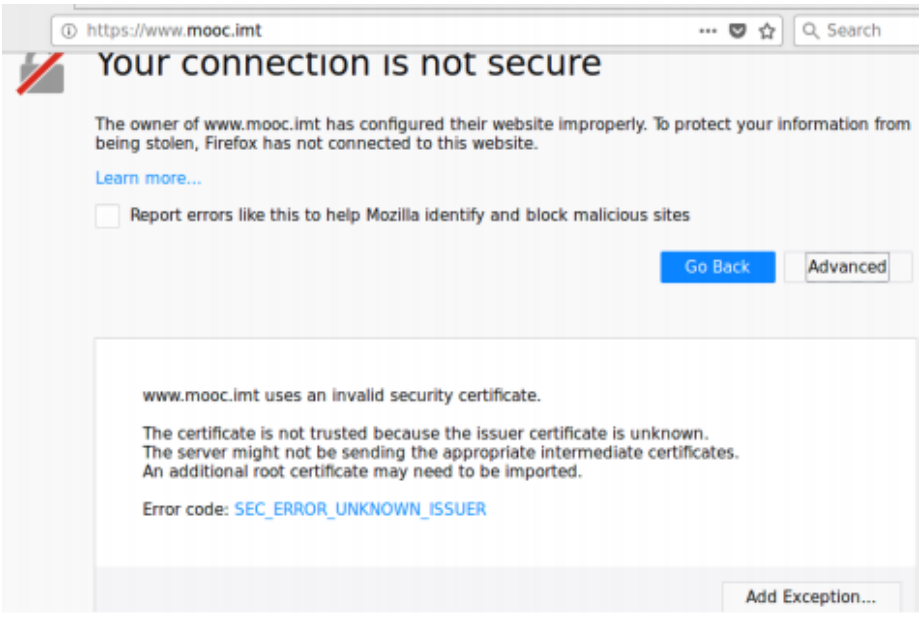
- Now, we would attempt to run the Firefox in brest terminal by typing in the required firefox terminal as below

```
brest $ firefox &
```

- Based ont he instruction we attempt to access the company's web server which is as follows

```
https://www.mooc.imt/
```

- we get the below in the firefox.



- The below questions arises now

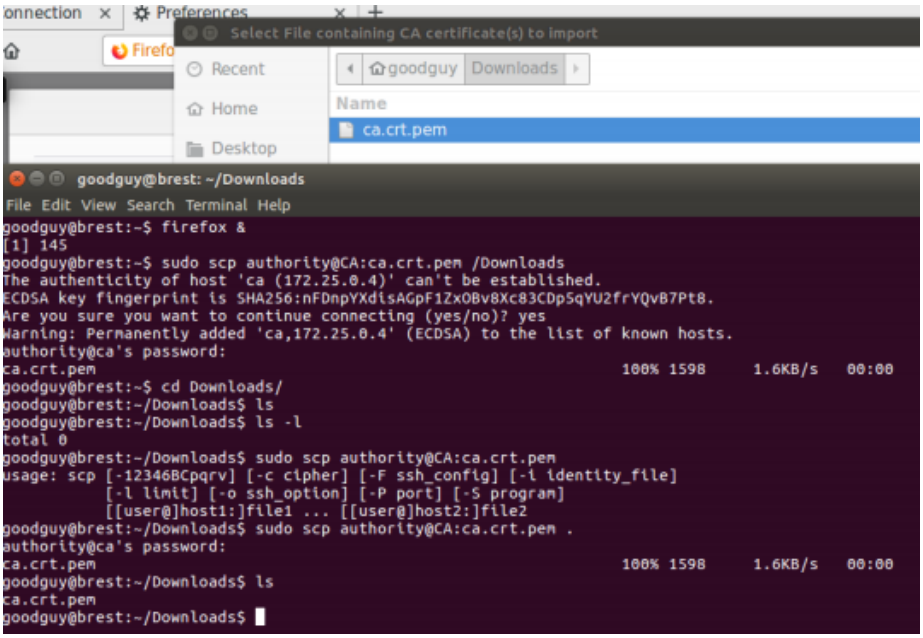
Q5: What do these alerts correspond to?

a) The certificate presented by the server is false;

b) The certificate is not made for that;

c) The certificate cannot be verified.

- The answer to the above question is that the certificate cannot be verified.
- Now, we downloaded the certificate to Downloads folder to Brest computer and imported it into the Firefox browser. Once downloaded the website was accessible. Below snap is the proof. Now, we also added the certificate in the Marseille computer as well.
- The proof



- The below question arises now

Q6: What do you deduce? [1 correct answer]

a) I don't know;

b) My browser is able to verify the server certificate because it has a root of certification that allows it;

c) My browser is able to verify the server certificate because it has the certificate from the server;

d) My browser has switched to a more tolerant mode and accepts all certificates from servers.

- The answer to the above question is that the browser is able to verify the server certificate because it has the certificate from the server.

5 Creation of certificates of the client

Always to meet the specifications of company X, we must provide each user a certificate that must allow him to authenticate on the company's web server.

Client certificates

In the same way as when creating the certificate for the server (see section 2), create a certificate for each of the goodguy and badboy human users (create a key, a CSR correctly entered for the user concerned, then the certificate itself by taking take care to position the -extensions client option).

So that the certificates we have just created allow users to authenticate on the company's web server, the certificate and its associated private key must be available for their web browser (Firefox).

To import a certificate and its private key into a web browser, it is necessary to store these different elements in a secure container: a file in PKCS # 12 format. This file format is an interchange format standardized by RSA. It is used for exchanges of all the material relating to a certificate (private key, public key, certificate and certification chain). It simplifies and secures import operations.

The PKCS # 12 file is created with the following command:

```
CA $ openssl pkcs12 -export -in goodguy.crt.pem -inkey goodguy.key.pem
                    -out goodguy.p12 -name "Good Guy" -certfile ca.crt.pem
```

- the -export option specifies the type of operation to do: export the material in the format PKCS # 12;
- the -in option specifies the name of the file containing the certificate (goodguy.crt.pem)
- the -inkey option designates the file containing the associated private key (goodguy.key.pem);

- We continue to follow the same procedure for the brest and Marseille computers based on the above instructions
- Before proceeding, we'd have to add the certificates and the private keys to the browser in a secure container as a

- The above is done based on the instruction

```
root@ca:~# openssl pkcs12 -export -in brest.crt.pen -inkey brest.key.pen -out brest.p12 -name "Good Guy" -certfile ca.crt.pen
Enter pass phrase for brest.key.pen:
Enter Export Password:
Verifying - Enter Export Password:
root@ca:~# openssl pkcs12 -export -in marseille.crt.pem -inkey marseille.key.pen -out marseille.p12 -name "Bad Boy" -certfile ca.crt.pem
Enter pass phrase for marseille.key.pem:
Enter Export Password:
Verifying - Enter Export Password:
```

- Now, both the files are put into their corresponding computers by the following computers based on the above instruction.

```
badboy@marseille:~$ sudo scp authority@CA:marseille.p12 /home/badboy
authority@ca's password:
marseille.p12                                 100% 3292     3.2KB/s   00:00
```

- Below is the corresponding action

```
goodguy@brest:~$ sudo scp authority@CA:brest.p12 /home/goodguy
authority@ca's password:
brest.p12                                     100% 3286     3.2KB/s   00:00
goodguy@brest:~$ ls
brest.p12  Downloads
```

- Now, we'd be able to view the content of key and certificate

```
badboy@marseille:~$ sudo openssl pkcs12 -in marseille.p12
Enter Import Password:
MAC verified OK
Bag Attributes
    friendlyName: Bad Boy
    localKeyID: 84 C5 E2 1B 32 E2 A5 22 FB 8B AF 2C 8A 2C 99 43 CC D7 6A 4E
subject=/C=FR/ST=France/O=AA MOOC IMT/OU=NETWORK SECURITY/CN=www.marseille.int/emailAddress=marseille@mooc.int
issuer=/C=FR/ST=France/O=AA MOOC IMT/OU=NETWORK SECURITY/CN=MOOC_CA/emailAddress=CA@mooc.int
-----BEGIN CERTIFICATE-----
MIIEAzCCAuugAwIBAgIBAzANBgkqhkiG9w0BAQsFADB9MQswCQYDVQQGEwJGUjEP
```

- The obvious question comes up now as below

Q7: Is the private key in the clear?

Yes indeed

b) No

- The answer is a big NO to the fact that the private key is not in clearText.
- Based on the instruction, we updated

/etc/apache2/sites-enabled/default-ssl.conf

- The configuration file of the www computer is being updated based on the instruction in the root mode. The next action performed is restarting the apache server. Now, we'd attempt to open firefox in both brest and marseille computers to access the website. Error shows up in this attempt.
- Question comes up now.

Q8: What do you see?

a) I cannot connect;

b) I can connect;

- The answer is NO, we are not able to connect.
- Anyway, the certificate revocation for marseille computer is as follows.

```
authority@ca:~$ openssl ca -config openssl.cnf -revoke marseille.crt.pem -crl_reason keyCompromise
Using configuration from openssl.cnf
Enter pass phrase for ./private/ca.key.pem:
Revoking Certificate 03.
Data Base Updated
unable to write 'random state'
authority@ca:~$
```

```
Revoked Certificates:
    Serial Number: 03
        Revocation Date: Dec 28 03:50:56 2019 GMT
        CRL entry extensions:
            X509v3 CRL Reason Code:
                Key Compromise
    Signature Algorithm: sha256WithRSAEncryption
```