Extra Encryption Lab: Web Authentication 1 – Set up HTTPS in Localhost

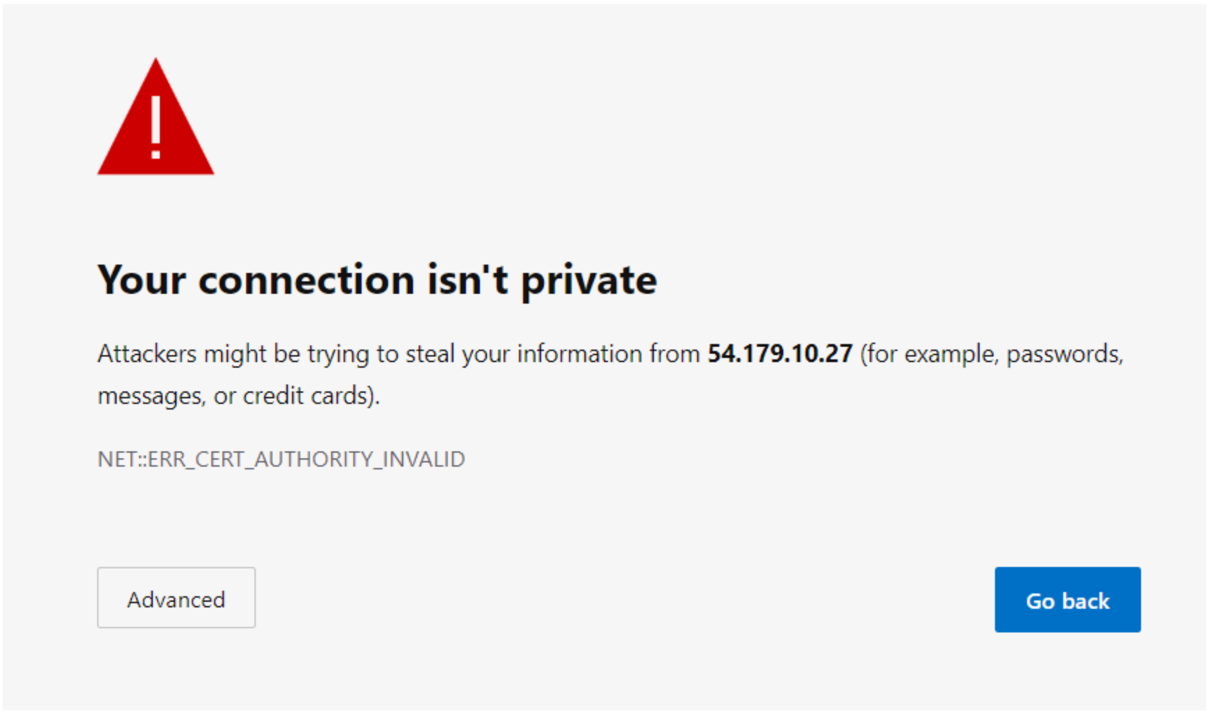
HTTPS has been pre-installed on the cloud system. Let's look at how it was made.

Task:

In any web browser, go to

https://1.remotelabs.me
or your own lab url

You might see something like this



Your connection isn't private

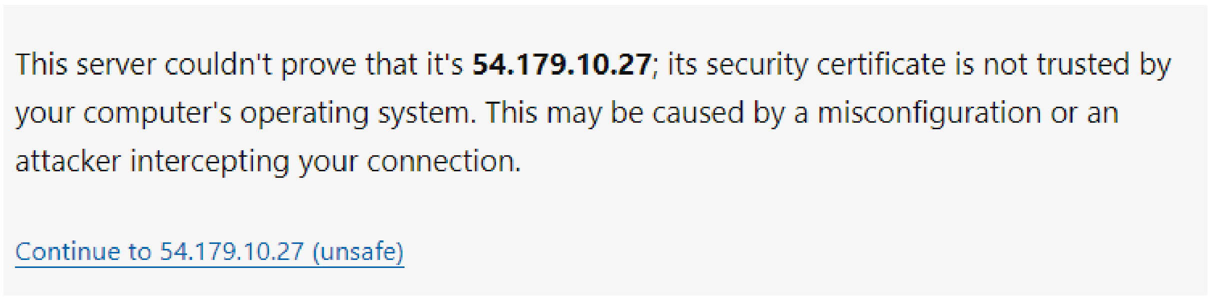
Attackers might be trying to steal your information from **54.179.10.27** (for example, passwords, messages, or credit cards).

NET::ERR_CERT_AUTHORITY_INVALID

This is a warning that shows the HTTPS certificate is invalid. In this case, the cert authority is invalid, meaning it is might be a self-signed certificate

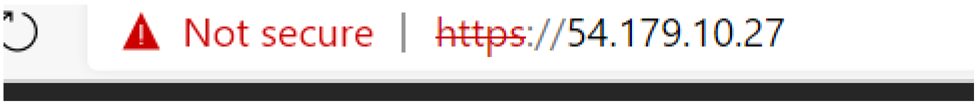
Click on "Advanced" and then "continue"

Copyright Blockcyber. Written by Wong Ding Jie


This server couldn't prove that it's **54.179.10.27**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.

Continue to 54.179.10.27 (unsafe)

Even though it says insecure, traffic between the website and browser is safe.

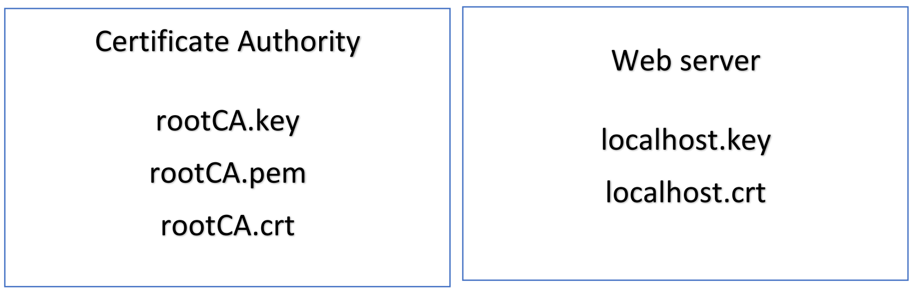

Not secure | ~~https~~://54.179.10.27

It is insecure because the site may not be who you think it is, that is why it is untrusted.

Steps to setting up HTTPS in localhost.

1. Use OpenSSL to generate a CA Root SSL Certificate Key(Certificate Authority) called **rootCA.key**. This is the CA's private key。
2. Generate a CA PEM file called **rootCA.pem**.
3. Generate a CA certificate file called **rootCA.crt**. This is the CA's public key.
4. Use OpenSSL to generate a private key and Certificate Signing Request(CSR) called **localhost.key** and **localhost.csr** for local server**.**
5. Use the CSR, rootCA.key and rootCA.pem to sign a domain level cert, generating a **localhost.crt.**
6. Install CA certificate(**rootCA.pem**) into browser to accept certificates from your new CA.
7. Instruct your server to use the new server certs. (done for you).

*notice: a few changes have been made to enable this setup to work. If you try it at home, additional steps are required.



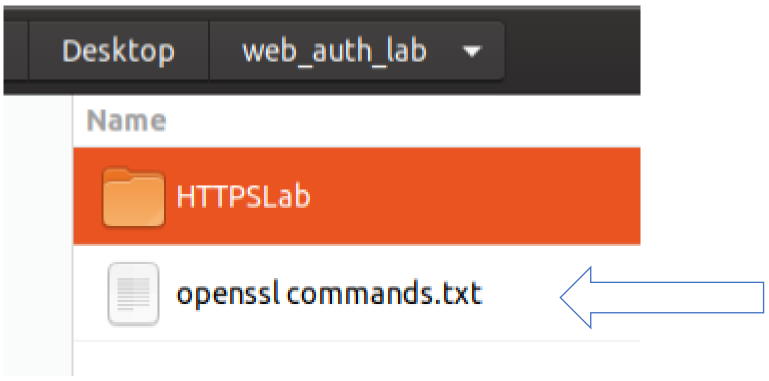
CSR is only used once to sign the certificate.

Copyright Blockcyber. Written by Wong Ding Jie

Open all the files with a text editor to explore.

Open command file

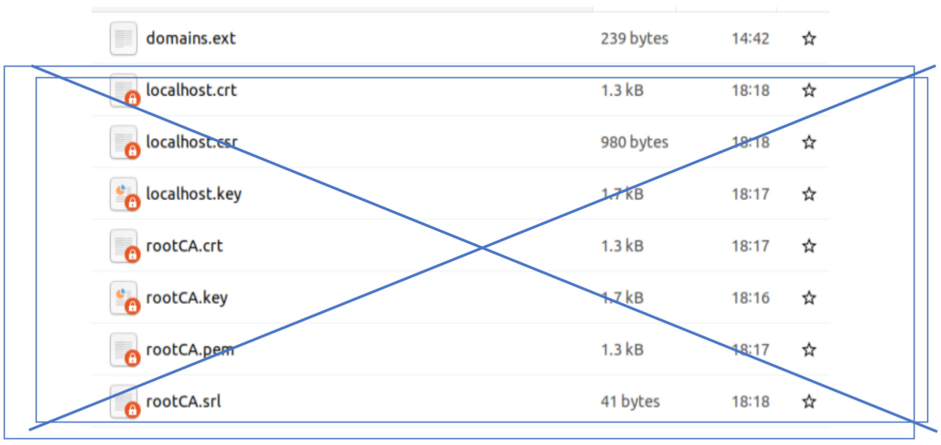
Go to web auth lab folder and open the OpenSSL commands file. Commands have been written for you to copy and paste.



Lab4 Exercise 1: Set up the Root SSL certificate (our own certificate authority)

Tasks:

0. Delete these items in the HTTPSLab. **Do not delete domains.ext.**
   If these are not present, skip to next step.



domains.ext	239 bytes	14:42	☆
localhost.crt	1.3 kB	18:18	☆
localhost.csr	980 bytes	18:18	☆
localhost.key	1.7 kB	18:17	☆
rootCA.crt	1.3 kB	18:17	☆
rootCA.key	1.7 kB	18:16	☆
rootCA.pem	1.3 kB	18:17	☆
rootCA.srl	41 bytes	18:18	☆

1. Open a terminal **inside** the HTTPSLab folder.

We need to run the commands in the same folder as domains.ext

Copy and paste the first line in the command file. Press enter to run the command.

```
sudo openssl genrsa -out rootCA.key 2048
```

Copyright Blockcyber. Written by Wong Ding Jie

This is to generate an RSA-2048 private key file for our Certificate Authority.

2. Copy and paste the second line in the command file. Press enter to run the command.

```
sudo openssl req -x509 -new -nodes -key -rootCA.key -sha256 -days 1024 -out
rootCA.pem
```

X509 is the public key cert standard that we are using.
sha256 to hash data.
Choose how many days for the cert to expire. In this case 1024 is chosen. You can choose any other number of days.
Save the output as **rootCA.pem**.

Options 选择:
- Organization name is used to identify the **Certificate Authority(CA) name**.
- other options can be anything random inputs.
- no need to fill in email. 不需输入 email。

3. Copy and paste the third line in the command file. Press enter to run the command.

```
sudo openssl x509 -in rootCA.pem -out rootCA.crt
```

This is to generate a certificate from the PEM file.

4. Copy and paste the fourth line in the command file. Press enter to run the command.

```
sudo openssl req -new -nodes -newkey rsa:2048 -keyout localhost.key -out
localhost.csr
```

This is to generate a new private key for our server. We name it **localhost.key**, and using that to generate the **Certificate Signing Request(CSR)**.

Options:
- Organization name is used to identify the **server name**.
- other options can be anything random inputs.
- no need to fill in email.

5. Copy and paste the fifth line in the command file. Press enter to run the command.

Copyright Blockcyber. Written by Wong Ding Jie

```
sudo openssl x509 -req -sha256 -days 1024 -in localhost.csr -CA rootCA.pem -
CAkey rootCA.key -CAcreateserial -extfile domains.ext -out localhost.crt
```

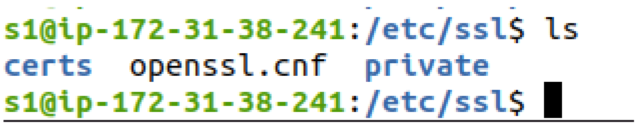
We are generating the server public key, called localhost.crt, by signing CSR with the CA keys, and also a server file configuration called domains.ext.

Installing the certs

The certs and keys will then be installed into the web server. Different webservers might have different ways of installing.

In this instance:

Go to /etc/ssl



```
s1@ip-172-31-38-241:/etc/ssl$ ls
certs   openssl.cnf   private
s1@ip-172-31-38-241:/etc/ssl$
```

Generally, the .CRT files go into the "certs" folder, and the private keys go to the "private" folder.

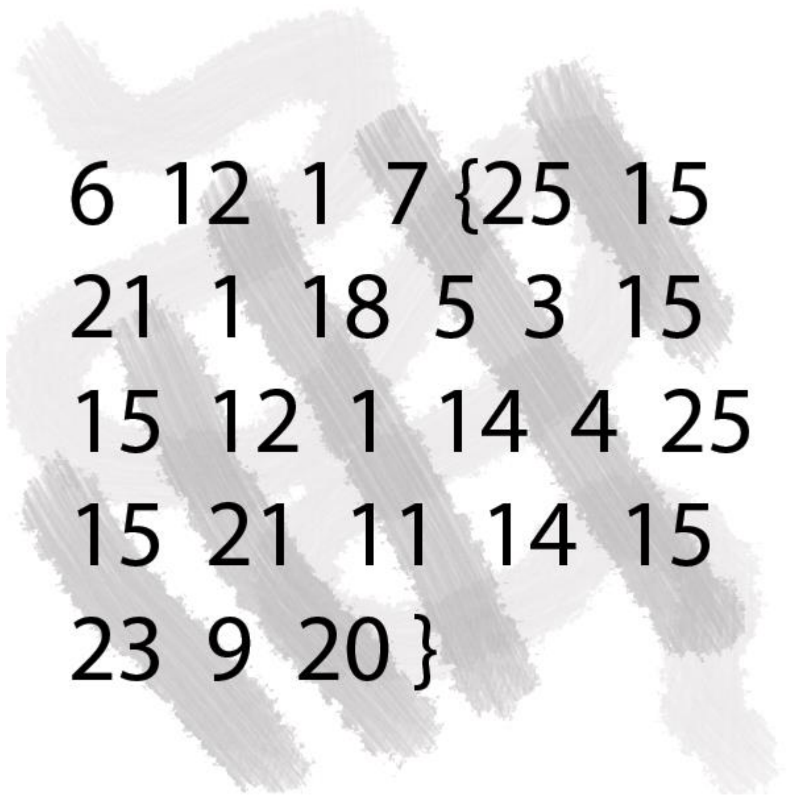
Tasks:

1. Go to the /etc/ssl directory
2. look at the items inside "certs" and "private" folder.
3. Do you see the pre-installed certificates?
4. Can sudo get you into the private folder? Find a way in to the private folder. Hint: who has the highest privilege in a linux system?
5. What is the name of the cert used by our server for HTTPS?

Copyright Blockcyber. Written by Wong Ding Jie

Cryptography teaser challenge

Decode this!



6 12 1 7 {25 15
21 1 18 5 3 15
15 12 1 14 4 25
15 21 11 14 15
23 9 20 }

If you need a hint, ask your instructor :p

Copyright Blockcyber. Written by Wong Ding Jie