# Fifth Assignment: Public Key Infrastructure

## Purpose

The purpose of this assignment is to enable you to be familiar with symmetric and asymmetric encryption as well as authentication of systems when addressing cybersecurity.

## Problem

To prevent tempering or eavesdrop of information between systems, cybersecurity include encryption and authentication.

## Form

You are to generate a certificate for your system, and upload it to Canvas in plain text format.

## Execution

For this assignment, you will be using OpenSSL. If you are using Windows, you will have to download the free application Cygwin, which provide functionality similar to a Linux distribution on Windows. If you use Linux or MacOS, you just have to run the terminal application.

If you are lazy, you may cut and paste the provided commands. However, you would lose 80% of the experience, so typing each command is recommended.

## Symmetric encryption

Within the terminal type `touch original.txt` to create an empty file. Type then `echo Industrial IoT is interesting > original.txt` to insert the text *Industrial IoT is interesting* into the file. If you want to see what is in the original.txt file, type `cat original.txt` To encode the file original.txt, you can use

```
openssl enc -aes-256-cbc -in original.txt -out encrypted.bin
```

It will ask you for a password from which a 256-bit secret key will be computed. Pick one that you can remember 5 minutes, *myPassword* is used here as the example. You can use the `cat` command to look at *encrypted.bin* file.

To decode a file the decrypt option (-d) has to be used

```
openssl enc -aes-256-cbc -d -in encrypted.bin -pass
pass:myPassword
```

This is an example using symmetric encryption since it uses the same key to encrypt and decrypt.

## Public Key Infrastructure

The problem of symmetric encryption is that the key must be somehow delivered to both parties (the sender and the receiver) without any other system or person being aware of it.

Although more computationally demanding, and thereby slower, asymmetric keys resolves this issue. The sender encrypts the payload with the public key of the receiver, which everybody can have. The encrypted payload can only be decrypted using the receiver's private key, which only the receiver has. To authenticate the public key of the receiver, certificate authority (CA) must sign a certificate signing request or a CSR. A CSR includes a public key as well as some extra information that gets inserted into the certificate when signed.

Since this assignment does not require a CA, you are asked to generate a self signed certificate and private key. You generate them signed certificate using the x509 standard valid for 365 days

```
openssl req -newkey rsa:2048 -nodes -keyout kul4iot.key -x509 -
days 365 -out kul4iot.crt
```

The *req* generates an interim certificate signing request (CSR) to get your details and then generate two output files: *kul4iot.crt* and *kul4iot.key*. The *req* needs some information about yourself and your organization. In the field "Common Name," or CN, you must provide the fully qualified domain name of the host, use your company's or *myhost.example.com*.

Use the `cat` command to look at *kul4iot.crt* and *kul4iot.key*. (Next week, you will be getting some signed ones from Amazon.)

You are to upload the *kul4iot.crt* onto Canvas. I can review it with

```
openssl x509 -text -noout -in kul4iot.crt
```

#kul4 #pki