Setting up TLS for a server
In this assignment, we are going to setup and configure a web server that can accept connections over HTTPS (a.k.a. TLS or SSL). To accept a connection over HTTPS, the server needs to have a private key and a corresponding public key signed by some certifying authority (CA). A certifying authority signs (and thereby binds) the the public-key with the fully qualified domain name (FQDN) and some other auxiliary info such as expiry date, provided at the time of registration. The public key, along with the auxiliary info and the signature, is called a certificate. The certificate is stored (and distributed) in a well defined format e.g., X.509.

Certificates are the primary way to distribute trust across the Internet. When a client wants to contact your server, the server will furnish the certificate to the client who, in turn, can validate the server's identity by verifying the certificate. Note that to verify the signature on the certificate the client must be familiar with the CA used to certify your public key. You can look here at the list of CA's preloaded in Firefox. This is a trusted list of CA's identified by Firefox and any certificate signed by them is trusted. Do NOT add a CA to this list unless you are absolutely sure about the identity of the domain you are visiting. A valid certificate is trusted as long as the CA who signed that certificate is trusted.

Assignment
For this assignment, first, you have to obtain a public domain. If you already have one, great; if not, use a free-tier EC2 domain name. Next, you have to generate a pair of public key and private key. You can use `openssl` to generate the key pair.

```bash
$ openssl req -newkey rsa:2048 -nodes -keyout domain.key -out domain.csr
```
After you have generated the key pair, the public key has to be signed by a CA. If **you have a registered top level domain (TLD)** such as example.com or www.example.org (not ristenpart.cornell.edu, which is not a TLD, cornell.edu is the top level), then you can obtain a free signed certificate from the following two websites:

>1: https://letsencrypt.org/
>2: https://www.startssl.com/

You can obviously use other sites, if you want to. Follow the procedure given in the web-sites to obtain the certificate. Often these sites ask for a certificate signing request token or CSR token, which is the file named `domain.csr` in the previous command.

If __you do not own a top level domain__, or you want to use the public domain names provided by EC2, you might not be able to get a certificate from the CAs listed above, because certificates are issued only for the TLD, and you don't own those TLDs. In this case, you can use self-signed certificates. You can refer to this article on how to generate a self-signed certificate. If you are using self-signed certificate then your browser will throw warning indicating that the certificate is not trusted. For your domain and certificate, you can overrule the warning, and add the certificate to the trusted list. (BUT be very cautious in doing so for any random website).

Now you have all the parts required to start a TLS connection, what is missing is putting them in correct places and configuring your server to use HTTPS. You must use an EC2 instance, and we recommend Apache as your web server - this is a common software stack for real websites. Nginx can also be used but Apache has better documentation.

You can test the quality of your TLS connection using online tools, such as Quality SSL Labs. We shall also use this site to test your servers.

Deliverable
You need to submit a text file with the following--

>1. the url of your domain, and the CA you used (write *self-signed* if you are using a self-signed certificate),
>2. brief description of your experience of setting up a server with TLS.

Extra Resources
Here is another good article on how to setup TLS correctly.
This is a very good article on working with certificates using `openssl`.