Opgaveregning 12 (28-04-2020)

## Question 1

**What is a PreMasterKey? Why is it useful to decrypt TLS streams?**

Det er “hemmeligheden” mellem klienten og serveren, som bliver brugt til at opsætte deres shared keys (sessions keys) som de bruger til at incrypt og decrypt de beskeder de sender til hinanden.

## Question 2

**Why is a self-signed certificate less trusted than one signed by a Certification Authority?**

De er mindre troværdige, da folk med skumle motiver også selv kan lave et certifikat. Derfor er det bedst at have et system hvor dette ikke er muligt og det derfor er sværere at snyde sig til et certifikat.

## Question 3

**How do you find the authorization material (API key) that you have to exchange in messages while using APIkey authorization? What about OAuth2.0 authorization (tokens)?**

Denne API nøgle skal indsættes manuelt, den kan potentielt komme fra en email eller anden form for kommunikation. OAuth2.0 tokens kommer fra en Auth server.

## Question 4

**Having a Javascript page as the client, where do you keep in the authorization material? (e.g.: a file on your computer, a cookie, etc)**

Der er mange muligheder, man kan bruge local storage på sin browser til at gemme API nøglen, dog gør dette det også muligt for folk at stjæle denne nøgle og udnytte den. Derfor vil det være bedst ikke at gemme nøglen på computeren, men at folk husker det inde i hoved sådan at folk ikke kan komme til det.

## Exercise 1

**Set up Wireshark to be able to snoop into your interaction with a HTTPS site.**

**Start Wireshark to collect TLS packets.**

**Open https://www.cs.aau.dk/ and identify the packets related to the interaction with the site. Recognize the messages for the TLS handshake. “Follow” the TLS flow and look into the HTTP verbs / etc.**

**IMPORTANT: remove SSLKEYLOGFILE since it makes your system unsecure.**

Fik det til at virke, er ret god

## Exercise 2

**Create a node.js server that serves a first webpage, and requires API key authentication to serve a second webpage. Then, access the second webpage using Postman. Finally, choosing between:**

**- Creating a node.js program that acts as a client, and dumps the second webpage to a file**

**- Accessing the second webpage directly from the browser**