HTTPS/SSL 2

HTTP/HTTPS 2

Wat ga je leren in dit labo?

- Wireshark installeren en gebruiken.
- Wireshark gebruiken om paketten te sniffen verstuurd via onbeveiligde http.
- Het verschil begrijpen tussen http en https.

Stappenplan

1. We gaan verder op het vorige labo van https/ssl. Hier hebben we een webserver opgezet met https dat gebruik maakt van een self signed certificate. Zorg ervoor dat alle stappen van het vorige labo zijn uitgevoerd.

2. Pas http.js aan zodat we terug TLS 1.3 gebruiken.

   ```
   secureProtocol : 'TLS_method'
   ```

3. Start de webserver op met

   ```
   node http.js
   ```

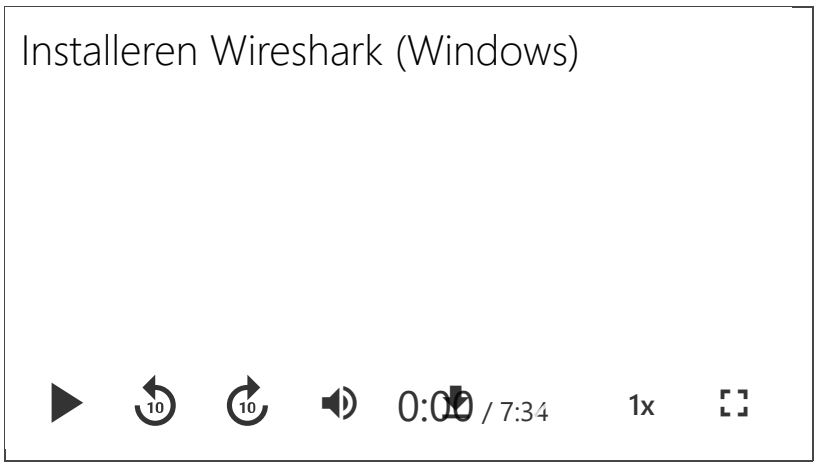
4. Open je browser (liefst chrome) op de adressen die in de terminal zijn gegeven.

   Op http://localhost:3000 zal je een website te zien krijgen die twee links bevat: "Form with GET" en "Form with POST".

   Op https://localhost:3001 krijg je de beveiligde versie van deze website te zien. Als je het labo vorige keer correct hebt uitgevoerd krijg je hier geen errors of warnings meer.

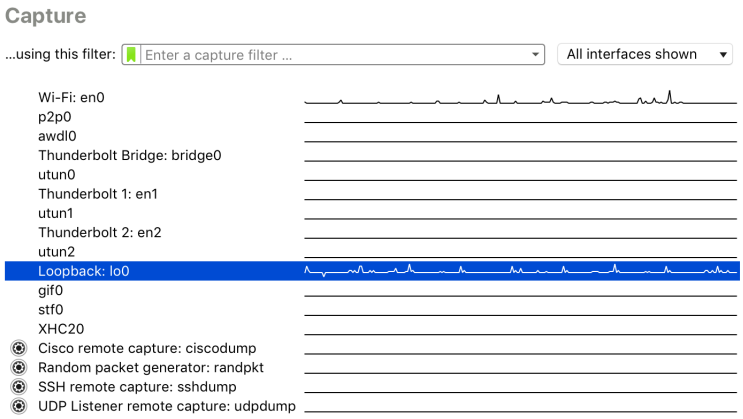
5. Sluit de webserver terug af door CTRL-C te drukken in je terminal venster.

6. Installeer wireshark: https://www.wireshark.org/. Als er gevraagd wordt of je ncap adapter wil installeren doe dit ook anders kan je niet naar je localhost capturen.

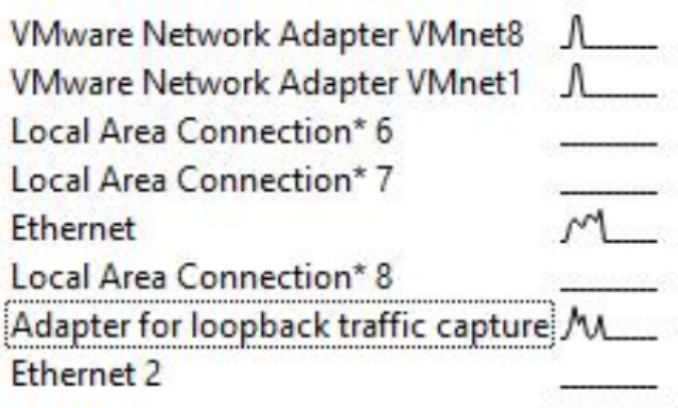


7. Als alles geïnstalleerd is start je wireshark op. Bekijk eerst het filmpje over filteren in wireshark:

8. Dubbelklik op de loopback interface:



In windows heet dit "Adapter for loopback traffic capture" (of iets gelijkaardig)



De loopback interface is een "virtuele netwerkkaart". Omdat onze webserver enkel lokaal draait op localhost, gaan we deze interface moeten gebruiken om paketten te onderscheppen tussen onze browser en de webserver.

9. Wireshark maakt het mogelijk om alle netwerk paketten individueel op te vangen en lezen. Vanaf je de loopback interface hebt gekozen zal je direct zien dat er paketten in de lijst komen te staan.

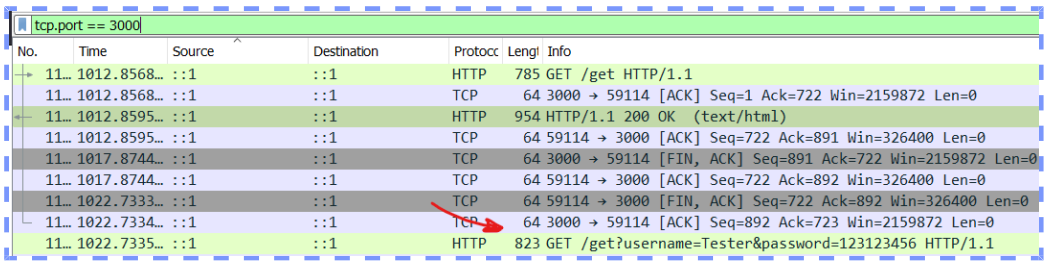
10. Zorg ervoor dat je enkel de paketten te zien krijgt op poort 3000 door in de balk bovenaan: `tcp.port == 3000` in te typen.

11. Ga naar http://localhost:3000/get en probeer met een login en paswoord in te loggen (willekeurig)

12. Zoek het pakket voor deze GET en zoek achter het paswoord dat je hebt ingegeven. Deze paketten zullen niet altijd direct te vinden zijn, dus wat speurwerk kan nodig zijn.

    Je kan ook zoeken achter paketten door Find Packet te doen (ctrl-f).

13. Maak een screenshot van het gevonden pakket en sleep deze hier onder in. Zorg ervoor dat het passwoord en login zichtbaar is.



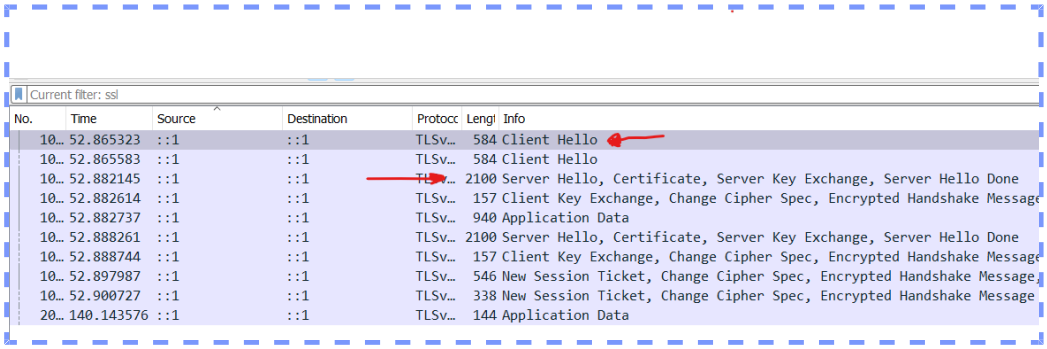
14. Waarom kon je dit paswoord gewoon clear text zien?

```
Omdat een http site is en er dus geen incryptie wordt gebruikt.
```

15. Verander nu de filter in de balk bovenaan zodat je de ssl handshake kan zien.

16. Ga nu naar https://localhost:3001 en kijk nu een wireshark dat je de paketten te zien krijgt. Neem een screenshot waar duidelijk het `Client Hello` en het `Server Hello` pakket op te zien is (gebruik een pijl).

    Sleep deze screenshot hier onder in:



17. Open een `Client Hello` pakket door er op te dubbelklikken. Zoek het `transport layer security` deel van het paket en klap dat open.

    Welke Cipher Suites worden ondersteunt door je browser volgens deze Client Hello?

```
Cipher Suites (16 suites)
    Cipher Suite: Reserved (GREASE) (0x4a4a)
    Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
    Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
    Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
    Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
    Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
    Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
```

    Tip: Je kan dit kopieren door rechter muisknop te klikken en dan Copy all selected tree items te kiezen.

    Wat wordt er bedoeld met deze Cipher Suites en waarom stuurt onze browser deze door met het `Client Hello` bericht:

```
het zijn mogelijke algoritmes voor toekomstig geincripteerd verkeer tussen server en client
ze worden doorgestuurd om de server te laten weten wat de gekende zijn voor de client,
zo kan de server zien of die er ook een van kent die dan gebruikt kan worden
```

    Welke versies van TLS ondersteunt je browser (zoek naar Extension: supported_version)

```
1.2
1.3
```

18. Open nu een `Server Hello` pakket door er op te dubbelklikken. Zoek het `transport layer security`

deel van het paket en klap dat open.

Welke Cipher Suite heeft de server gekozen:

```
Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
```

19. Verander nu de filter in de balk bovenaan naar: `tcp.port == 3001`
20. Ga nu naar https://localhost:3001/get en probeer in te loggen met eender welk login en passwoord.
21. Als je nu gaat zoeken in de paketten die hij gesniffed heeft zal je opmerken dat je deze niet meer vind. Waarom is dit?

```
Omdat het nu over een geincrypteerde https verbinding gaat en er dus niets meer in plain text
wordt doorgestuurd.
```

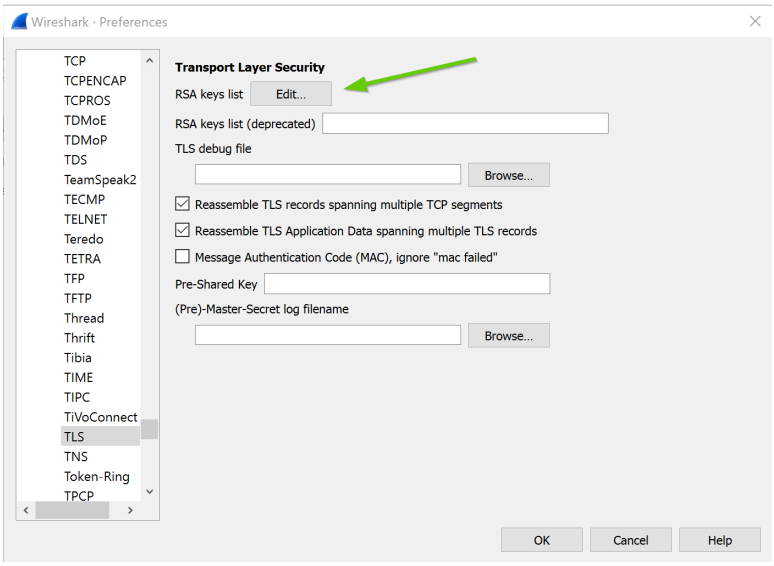
22. Jouw browser en de server spreken normaal gezien altijd de meest veilige cipher suite (encryptie algoritme) af tijdens de handshake. We gaan nu bewust een zwakkere kiezen, zodat we deze toch kunnen uitlezen via wireshark.
23. Pas de options aan in http.js zodat ze deze properties bevat:

````
```
ciphers: 'TLS_RSA_WITH_AES_256_CBC_SHA',
secureProtocol  : 'TLSv1_2_method',
```
````

Als je nog herinnert, volgens de website van ssllabs.com stond TLS_RSA_WITH_AES_256_CBC_SHA gemarkeerd als WEAK:

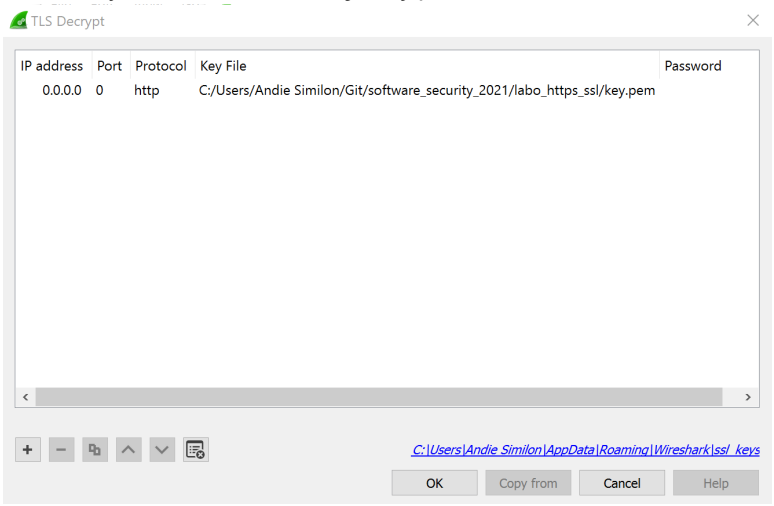
Het ideale voorbeeld dus!

24. Herstart de http server.
25. Ga terug naar wireshark en ga naar preferences (in Edit). Of druk CTRL-SHIFT-P
26. Ga naar Protocols en daarna TLS. Vervolgens ga je naar `RSA keys list` en druk je op Edit.



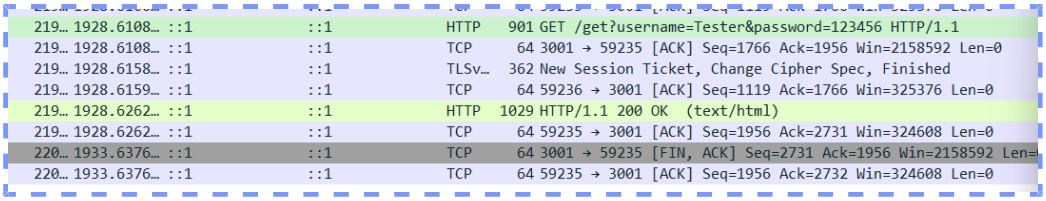
en dan druk je op het + knopje.

27. Daar vul je de volgende waarden in:
    - IP Address: 0.0.0.0
    - Port: 0
    - Protocol: http
    - Key File: *de locatie van je key.pem bestand*



28. Als je nu surft naar https://localhost:3001 en je logt in dan zal je terug in wireshark de paketten kunnen lezen. Dit is omdat de cipher suite die je gebruikt niet veilig is. Als iemand in bezit is van de private sleutel, kan die dus toch alles uitlezen.

29. Maak een screenshot waarmee je bewijst dat je nu terug je login en paswoord kan zien in wireshark als je inlogt in sleep deze hieronder in:



30. Als je nu terug een sterkere cipher suite kiest en de TSL versie terug verhoogt naar 1.3 met

    ```
    ciphers: 'TLS_AES_256_GCM_SHA384',
    secureProtocol  : 'TLS_method',
    ```

    dan zal je zien dat je de berichten zelfs niet meer kan lezen met de private key.

31. Print deze pagina af als PDF en slaag deze op als `naam_voornaam_labo_https_ssl.pdf`.

    Stuur deze vervolgens in via digitap!