IT Technology

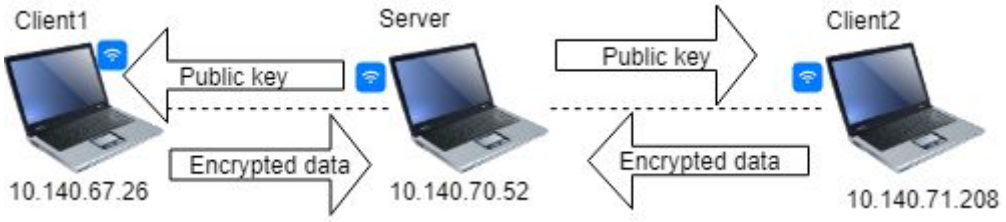
Assignment 23
TLS for socket communication

Authors
Tihamer Biliboc
tiha0006@edu.eal.dk
Sebastian Thomle Mason
seba7286@edu.eal.dk
Anthony James Peak
anth0662@edu.eal.dk

Thursday 20 Dec 2018

Table of Contents

A network diagram with brief explanation



The server sends a public key to the client, then the client encrypts the message and sends it back to server. Once the server receives the encrypted data, it decrypts with the private key which is located on the server machine.

Describe the aim of the software/program in 4 lines

The aim of the program is to send encrypted data from client to server via TCP, they exchange keys and check certificates. The data can be monitored in wireshark, although the content can’t be captured in plain text.

Illustrate graphically and explain the introduce TLS
Explain what it means to encrypt data and to do authentication

The server creates a socket, binds that socket to an address and assigns the socket to a thread. The server does this again for the second connection.

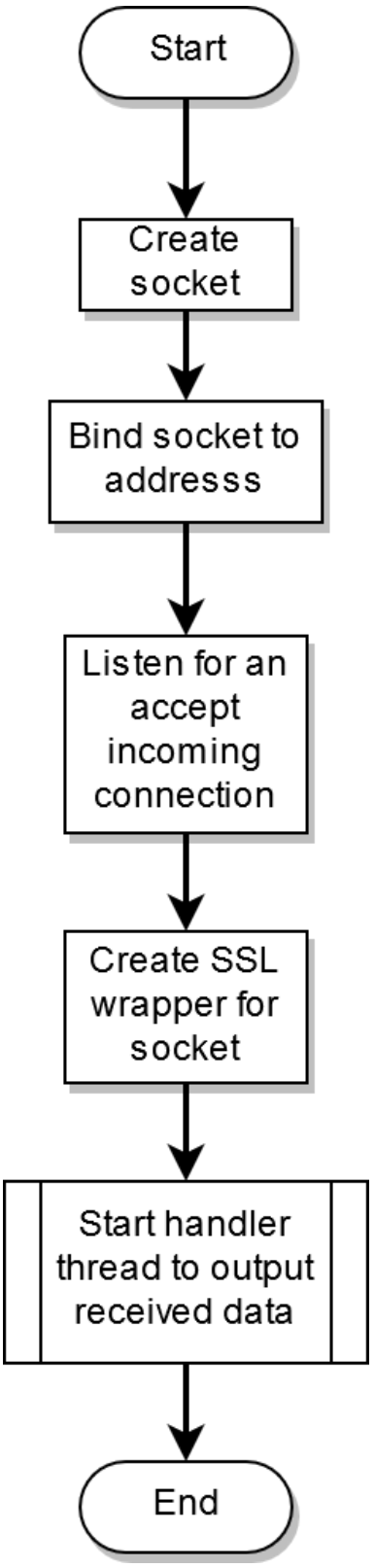
Each thread listens on it's specified port and accepts the first incoming connection, it then continually reads and prints any incoming data.

To get a certificate, you must create a public key on the server. The data that you send to the client contains the public key. The client uses the public key to encrypt data which can later be decrypted by the server using the private key. The client never sees the private key.

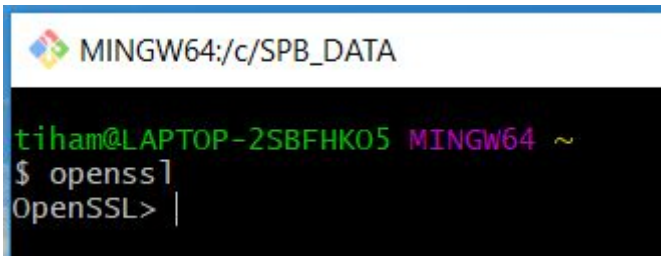
TLS (Transport Layer Security) is just an updated, more secure, version of SSL. We still refer to our security certificates as SSL because it is a more commonly used term.

Encryption allows information to be hidden so that it cannot be read without special knowledge. Encryption is the most effective way to achieve data security. To read an encrypted file, you must have access to a secret key or password that enables you to decrypt it.

See the code here.



Private key and server certificate in OpenSSL




```
OpenSSL> genrsa -des3 -out server.orig.key 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
........+++++
.....................................................................................
.....................................................................+++++
e is 65537 (0x010001)
Enter pass phrase for server.orig.key:
Verifying - Enter pass phrase for server.orig.key:
OpenSSL> rsa -in server.orig.key -out server.key
Enter pass phrase for server.orig.key:
writing RSA key
OpenSSL> req -new -key server.key -out server.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DK
State or Province Name (full name) [Some-State]:Odense
Locality Name (eg, city) []:Odense
Organization Name (eg, company) [Internet Widgits Pty Ltd]:ass23
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
OpenSSL> x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
Signature ok
subject=C = DK, ST = Odense, L = Odense, O = ass23
Getting Private key
OpenSSL> ls
Invalid command 'ls'; type "help" for a list.
error in ls
OpenSSL>
```

OpenSSL comes with Git, so if we have it installed, we can use it from there. We only need to open Git Bash and type in *openssl*, then Type the 4 commands (in the red box), to create the private key and server certificate.

Show and explain the output from the server program when client and server are executed

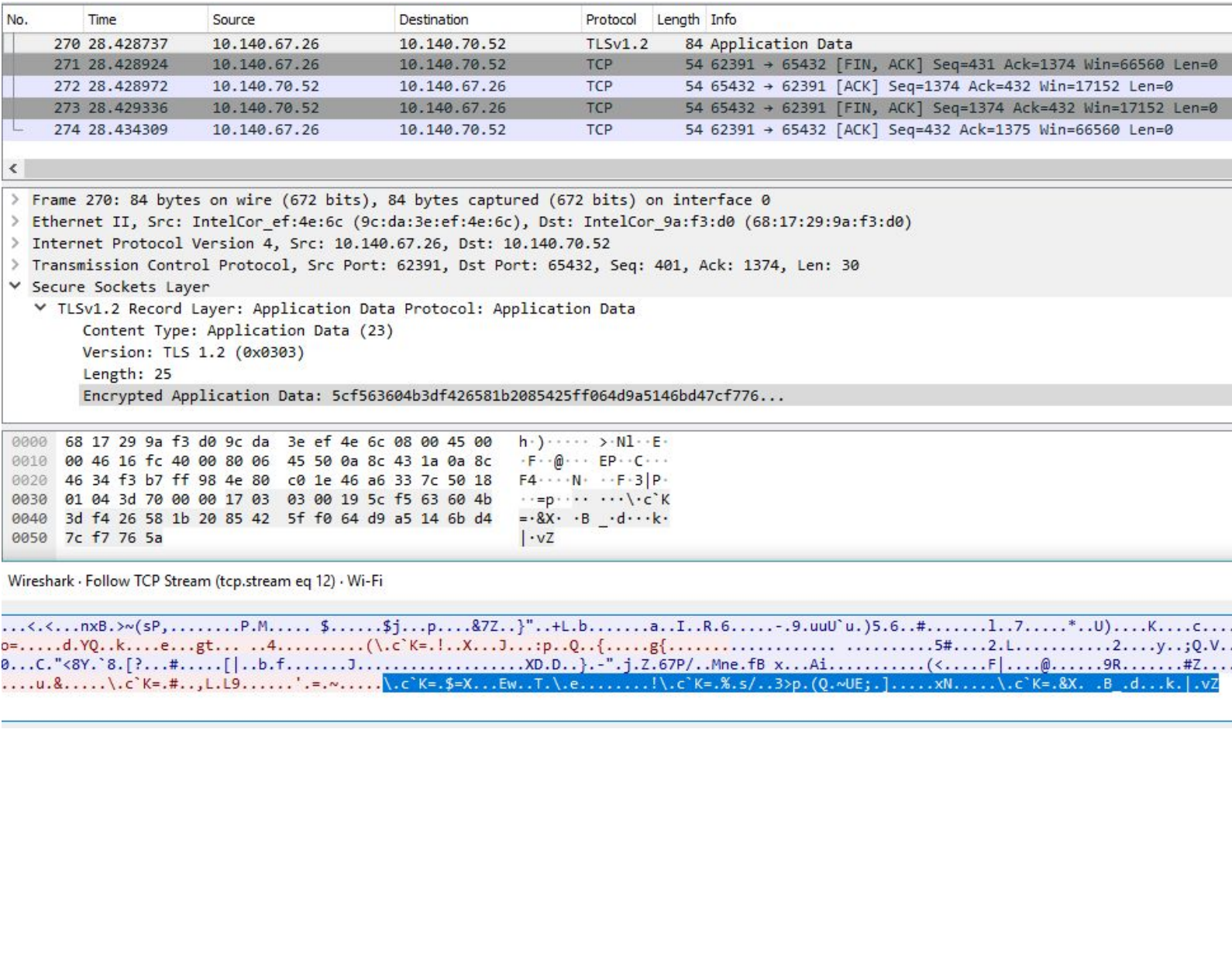
```
PS C:\Users\sebth\Google Drive\EAL\Network\Hand-in's\Group
Hand-in's\Github\Network\ass23TLSForSocketCommunication\Code> Python server2.py

waiting for connection...
('...connected from:', ('10.140.67.26', 63026))
waiting for connection...
('...connected from:', ('10.140.71.208', 56822))
Hello, I am Anthony!
Hello my dude :D This is your Client speaking
```

= Anthony

= Tihamer

The server accepts and monitors connections asynchronously, with a seperate thread running for each open connection to monitor and print received data.

Show in Wireshark that the transmitted data cannot be monitored in plain text



No.	Time	Source	Destination	Protocol	Length	Info
270	28.428737	10.140.67.26	10.140.70.52	TLSv1.2	84	Application Data
271	28.428924	10.140.67.26	10.140.70.52	TCP	54	62391 → 65432 [FIN, ACK] Seq=431 Ack=1374 Win=66560 Len=0
272	28.428972	10.140.70.52	10.140.67.26	TCP	54	65432 → 62391 [ACK] Seq=1374 Ack=432 Win=17152 Len=0
273	28.429336	10.140.70.52	10.140.67.26	TCP	54	65432 → 62391 [FIN, ACK] Seq=1374 Ack=432 Win=17152 Len=0
274	28.434309	10.140.67.26	10.140.70.52	TCP	54	62391 → 65432 [ACK] Seq=432 Ack=1375 Win=66560 Len=0

```
> Frame 270: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface 0
> Ethernet II, Src: IntelCor_ef:4e:6c (9c:da:3e:ef:4e:6c), Dst: IntelCor_9a:f3:d0 (68:17:29:9a:f3:d0)
> Internet Protocol Version 4, Src: 10.140.67.26, Dst: 10.140.70.52
> Transmission Control Protocol, Src Port: 62391, Dst Port: 65432, Seq: 401, Ack: 1374, Len: 30
v Secure Sockets Layer
  v TLSv1.2 Record Layer: Application Data Protocol: Application Data
      Content Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 25
      Encrypted Application Data: 5cf563604b3df426581b2085425ff064d9a5146bd47cf776...
```

```
0000  68 17 29 9a f3 d0 9c da  3e ef 4e 6c 08 00 45 00   h·)····· >·Nl··E·
0010  00 46 16 fc 40 00 80 06  45 50 0a 8c 43 1a 0a 8c   ·F··@··· EP··C···
0020  46 34 f3 b7 ff 98 4e 80  c0 1e 46 a6 33 7c 50 18   F4····N· ··F·3|P·
0030  01 04 3d 70 00 00 17 03  03 00 19 5c f5 63 60 4b   ··=p···· ···\·c`K
0040  3d f4 26 58 1b 20 85 42  5f f0 64 d9 a5 14 6b d4   =·&X· ·B _·d···k·
0050  7c f7 76 5a                                       |·vZ
```

Wireshark · Follow TCP Stream (tcp.stream eq 12) · Wi-Fi

```
...<.<...nxB.>~(sP,........P.M..... $......$j...p....&7Z..}"..+L.b.......a..I..R.6.....-.9.uuU`u.)5.6..#.......l..7.....*..U)....K....c....
o=.....d.YQ..k....e...gt... ..4.........(\.c`K=.!..X...J...:p..Q..{.....g{..................... ..........5#....2.L...........2....y..;Q.V..
0...C."<8Y.`8.[?...#.....[|..b.f.......J...................XD.D..}.-".j.Z.67P/..Mne.fB x...Ai...........(<.....F|....@......9R.......#Z....
....u.&.....\.c`K=.#..,L.L9......'.=.~.....\.c`K=.$=X...Ew..T.\.e........!\.c`K=.%.s/..3>p.(Q.~UE;.]......xN.....\.c`K=.&X. .B_.d...k.|.vZ
```