Secure chatserver

Naam: Jelle Groot
Klas: C203
Studentnummer: 500902844
Docent: M. Bonsema

Table of Contents

1. Ontwerp

1.1 Handshake

De handshake wordt volgens de methode van TLS_1.2. Hiervoor is geen module gebruikt, maar eigen code die hetzelfde principe in zekere zin mogelijk maakt. Er wordt gebruik gemaakt van “Client Hello” en “Server Hello”, waarna aan de hand van Diffie-Hellman de elliptic-curve-cerfticaten worden uitgewisseld in de vorm van public keys om een shared-key te bepalen. Het bepalen van deze shared-key wordt gezien als een verificatie van een veilige verbinding. Vervolgens wordt de handshake voltooid.

1.2 Encryptie

De berichten worden verseluteld met AES256-CFB, omdat AES256 momenteel de sterkste beveiliging is en momenteel nog niet gekraakt is. Het is een symmetrische encryptie, wat betekent dat beide kanten dezelfde sleutel gebruiken voor het versleutelen en ontsleutelen van de berichten. Verder wordt de key met de Diffie-Hellman methode gedeeld, waardoor deze nooit openbaar wordt gemaakt en dus veilig is. Tot slot wordt er ook bij elk bericht een initialization-vector(iv) toegevoegd. AES256 is niet gevoelig voor brute-force aanvallen.

1.3 Man in the Middle Attack

Indien er een man in the middle attack plaatsvindt kan alleen het met AES256-CFB versleutelde bericht onderschept worden. Het zal een volledig versleuteld bericht zijn, waardoor de hackers er niks mee kunnen. Dit komt ook doordat de shared-key nooit openbaar is gemaakt en dus niet afgeluisterd kan zijn.

1.4 Functies

1.4.1 Server:

De code van de server is onderverdeeld in twee classes.

Functie: server

Deze functie staat buiten de classes. Het is de functie die een paar variabelen bevat voor het programma. Verder regelt deze functie het verzenden en ontvangen van berichten, door functies aan te roepen.

Class Connection:

De eerste class bevat alle functies voor de connectie met de client, het verzenden van berichten, het versleutelen van berichten, het ontvangen van berichten en het ontsleutelen van ontvangen berichten.

Functie: def __init__

In deze functie staan de variabelen die specifiek bij de class horen. In dit geval zorgt het er ook voor dat bepaalde variabelen vanuit de startfunctie (def_server) geimporteerd worden.

Functie: connect

In deze functie wordt door middel van de methode “socket” een verbinding opgezet. Vervolgens wordt op deze verbinding geluisterd of er een client verbindt met de socket. Mocht dit het geval zijn, dan wordt de handshake in gang gezet. Ook wordt de functie aangeroepen om de shared-key te maken.

Functie: send_message

In deze functie wordt door middel van de methode “socket” de socket gebruikt om een bericht te versturen. Verder wordt dit bericht versleuteld. Deze versleuteling wordt mogelijk gemaakt door de “Cipher”, “algoritms” en “modes” methodes uit de “cryptography” library.

Er wordt padding toegepast over het versleutelde bericht, om ervoor te zorgen dat het verstuurde bericht precies uit blokken van 16 bytes bestaat. De methode die hierbij gebruikt wordt is “padding” uit de “cryptography” library.

Tot slot wordt er een initialization vector (iv) aangemaakt met de methode “secrets”. Zo kan er precies een iv gemaakt worden van 16 bytes.

Functie: receive_message

In deze functie wordt door middel van de methode “socket” de socket gebruikt om een bericht te ontvangen. Verder wordt dit bericht ontsleuteld. Deze versleuteling wordt mogelijk gemaakt door de “Cipher”, “algoritms” en “modes” methodes uit de “cryptography” library.

Er wordt padding toegepast over het ontsleutelde bericht, om ervoor te zorgen dat de eventueel toegepaste padding wordt verwijderd. De methode die hierbij gebruikt wordt is “padding” uit de “cryptography” library.

Class Diffie_Hellman:

De tweede class is voor de Diffie Hellman key-exchange. Hierbij wordt gebruik gemaakt van certificaten en private-keys/public-keys. In deze class is een functie aanwezig om de private-key en het certificaat in te laden. Verder is er een functie aanwezig om de shared_key te vormen.

Functie: def __init__:

In deze functie staan de variabelen die specifiek bij de class horen. In dit geval zorgt het ervoor dat vanuit de eerste class de locaties van de private-key en het certificaat geimporteerd worden.

Functie: open_privatekey_and_cert:

Hier worden de private-key van de server en het certificaat van de client ingeladen. Vervolgens wordt de private key uit de PEM-file gehaald met de methode “serialization”. Ook het certificaat wordt uit de PEM-file gehaald, maar deze met de methode “x509”. Hierna wordt van het certificaat een public-key gemaakt.

Functie: Shared_key:

Deze functie laat eerst de functie “open_privatekey_and_cert” uitvoeren. Vervolgens wisselt deze functie de public key uit en maakt een shared-key. Het uitwisselen van de public-key maakt gebruik van de methode “ec” uit de “cryptography” library.

1.4.2 Client:

Ook de code van de server is onderverdeeld in twee classes. De eerste class bevat alle functies voor de connectie met de client, het verzenden van berichten, het versleutelen van berichten, het ontvangen van berichten en het ontsleutelen van ontvangen berichten.

De tweede class is voor de Diffie Hellman key-exchange. Hierbij wordt gebruik gemaakt van certificaten en private-keys/public-keys. In deze class is een functie aanwezig om de private-key en het certificaat in te laden. Verder is er een functie aanwezig om de shared_key te vormen.

Functie: server:

Deze functie staat buiten de classes. Het is de functie die een paar variabelen bevat voor het programma. Verder regelt deze functie het verzenden en ontvangen van berichten, door functies aan te roepen.

Class Connection:

Functie: def __init__

In deze functie staan de variabelen die specifiek bij de class horen. In dit geval zorgt het er ook voor dat bepaalde variabelen vanuit de startfunctie (def_server) geimporteerd worden.

Functie: connect

In deze functie wordt door middel van de methode “socket” een verbinding gemaakt met de socket van de server. Mocht dit het geval zijn, dan wordt de handshake in gang gezet. Ook wordt de functie aangeroepen om de shared-key te maken.

Functie: send_message

In deze functie wordt door middel van de methode “socket” de socket gebruikt om een bericht te versturen. Verder wordt dit bericht versleuteld. Deze versleuteling wordt mogelijk gemaakt door de “Cipher”, “algoritms” en “modes” methodes uit de “cryptography” library.

Er wordt padding toegepast over het versleutelde bericht, om ervoor te zorgen dat het verstuurde bericht precies uit blokken van 16 bytes bestaat. De methode die hierbij gebruikt wordt is “padding” uit de “cryptography” library.

Tot slot wordt er een initialization vector (iv) aangemaakt met de methode “secrets”. Zo kan er precies een iv gemaakt worden van 16 bytes.

Functie: receive_message

In deze functie wordt door middel van de methode “socket” de socket gebruikt om een bericht te ontvangen. Verder wordt dit bericht ontsleuteld. Deze versleuteling wordt mogelijk gemaakt door de “Cipher”, “algoritms” en “modes” methodes uit de “cryptography” library.

Er wordt padding toegepast over het ontsleutelde bericht, om ervoor te zorgen dat de eventueel toegepaste padding wordt verwijderd. De methode die hierbij gebruikt wordt is “padding” uit de “cryptography” library.

Class Diffie_Hellman:

De tweede class is voor de Diffie Hellman key-exchange. Hierbij wordt gebruik gemaakt van certificaten en private-keys/public-keys. In deze class is een functie aanwezig om de private-key en het certificaat in te laden. Verder is er een functie aanwezig om de shared_key te vormen.

Functie: def __init__:

In deze functie staan de variabelen die specifiek bij de class horen. In dit geval zorgt het ervoor dat vanuit de eerste class de locaties van de private-key en het certificaat geimporteerd worden.

Functie: open_privatekey_and_cert:

Hier worden de private-key van de client en het certificaat van de server ingeladen. Vervolgens wordt de private key uit de PEM-file gehaald met de methode “serialization”. Ook het certificaat wordt uit de PEM-file gehaald, maar deze met de methode “x509”. Hierna wordt van het certificaat een public-key gemaakt.

Functie: Shared_key:

Deze functie laat eerst de functie “open_privatekey_and_cert” uitvoeren. Vervolgens wisselt deze functie de public key uit en maakt een shared-key. Het uitwisselen van de public-key maakt gebruik van de methode “ec” uit de “cryptography” library.