Descrierea proiectului

Capitolul 1.1 – Descrierea proiectului

Pentru a putea incepe cu analiza protocolului TLS am decis sa incep proiectul mai intai cu o comunicatie simpla de client-server in java. Am ales sa incep proiectul astfel deoarece am dorit sa arat de ce avem nevoie protocolul de TLS si cum putem sa il aplicam intr-o aplicatie deja existenta dar cu o comunicatie nesigura.

Pentru crearea aplicatiei client-server cu comunicatie nesigura am create 2 clase : EchoServer si EchoClient din pachetele: com.tls.analysis.echo.server si respectiv com.tls.analysis.echo.client.

Capitolul 1.1.1.1 – Descrierea aplicatie EchoServer:

Incepem aplicatia prin a ne importa clasele necesare:

Text

Description automatically generated

In interiorul clasei EchoServer avem definit portul la care dorim sa ascultam pentru conexiuni ca si variabila statica finala de tip int.

In interiorul functiei main incercam sa creem un ServerSocket denumit sugestiv serverSocket pentru a asculta conexiuni dar si o variabila de tip Socket pe care am denumit-o client. Aceasta variabila am asignat-o prin apelul functiei serverSocket.accept() care intoarce un socket nou cu conexiunea creata.

Dupa ce conexiunea a fost creata cu succes, incepem sa citim ceea ce client-ul doreste sa ne comunice cat timp sesiunea este activa.



Mai multe functii din main arunca exceptii de tip IO, aceastea sunt subliniate cu rosu, astfel ca am introdus totul intr-un try catch. :

Text

Description automatically generated

Capitolul 1.1.1.2 – Rularea aplicatiei echoServer:

Pentru rularea aplicatiei in InteliJ, apasam cu click stanga pe iconita verde din stanga clasei si apoi run “Numele\_aplicatiei”.

Graphical user interface, text, application

Description automatically generated

Daca totul a decurs cum trebuie acesta este mesajul pe care am trebuie sa il vedem in consola:

Text

Description automatically generated

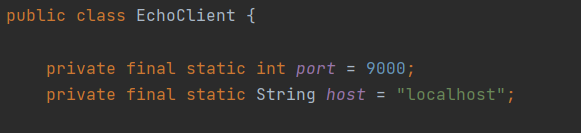
Capitolul 1.1.2.1 – Descrierea aplicatiei EchoClient

Ca si in cazul aplicatiei EchoServer incepem prin a ne importa clasele necesare:

Text

Description automatically generated

In interiorul clasei Echo client avem de aceasta data 2 variabile definite port-ul si adresa la care dorim sa ne conectam:



In interiorul functiei main incercam sa cream un socket cu conexiunea de la adresa si port-ul definite anterior si daca reuseste, acesta trimite un mesaj catre server pana cand este introdus mesajul “exit”. De asemenea, multe dintre functiile apelate in interiorul functiei main arunca exceptii de tip IO asa ca le-am introdus si pe acestea intr-un try-catch:

Text

Description automatically generated

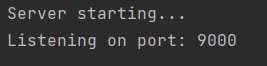
Capitolul 1.1.2.2 – Rularea aplicatiei EchoClient

Pentru a rula aplicatia EchoClient trebuie mai intai rulata aplicatia EchoServer.

Daca totul decurge bine in consola ar trebui sa vedem ca acum avem 2 tab-uri, fiecare consola cu aplicatia sa.



In consola EchoServer ar trebui sa vedem mesasjul acesta:



In consola EchoClient ar trebui sa vedem mesajul acesta:

Text

Description automatically generated

Acest mesaj practice ne confirma ca ne-am conectat la server si ca putem sa incepem sa comunicam cu server-ul.

O data trimis un mesaj, putem verifica pe server daca acesta a fost primit.

Text

Description automatically generated

Text

Description automatically generated

Capitolul 1.1.3 – De ce avem nevoie de protocolul TLS?

Cand initiem o astfel de comunicatie date noastre sunt trimise in “plain-text”, adica datele noastre sunt expuse catre oricine ar dori sa vada comunicatia.

Pentru a demonstra acest lucru am dorit sa interceptez traficul de date folosind aplicatia “WireShark” si ruland aplictia client-server pe o masina virtuala de linux.

Am inceput prin a rula cele 2 aplicatii server si client si apoi am inceput sa interceptez traficul de pe localhost cu WireShark cu un filtru pe portul pe care comunica cele 2 aplicatii.

Graphical user interface, text

Description automatically generated

Graphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Un atacator poate vedea astfel ce mesaje trimite clar client-ul catre server.

Capitolul 1.2 – Adaugarea protocolului TLS asupra aplicatiei Echo Client – Server

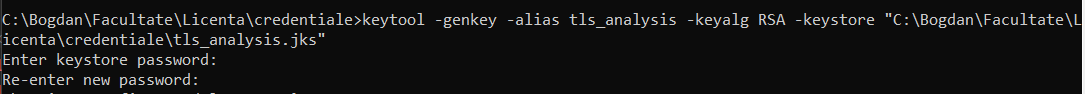
Capitolul 1.2.1 Crearea unui keystore

Pentru a crea un KEYSTORE:

Rulam comanda

Keytool -genkey -alias tls\_analysis -keyalg RSA -keystore “C:\Bogdan\Facultate\Licenta\credentiale\tls\_analysis.jks”

Ne va cere sa ne alegem o parola:



Apoi ni se va cere sa introducem datele de identificare:

Text

Description automatically generated

Apoi ni se va cere sa confirmam corectitudinea datelor introduse si odata confirmata aceasta, cheia va fi generata.

A picture containing text

Description automatically generated

Capitolul 1.2.2 Aplicarea protocolului TLS asupra aplicatiei Echo client-server

Capitolul 1.2.2.1 Introducere

Crearea unui server in Java cu protocolul TLS este similara ca cea a creeri unui server fara acest protocol, diferenta fiind ca in cazul utilizarii protocolului de TLS folosim SSLSockets in loc de Sockets, si bineinteles, un certificat ca cel mentionat anterior.

Capitolul 1.2.2.2 Aplicatia EchoSSLServer

Incepem prin a ne importa clasele folosite

Text

Description automatically generated

Definim port-ul pe care va asculta server-ul si setam proprietatile sistemului astfel incat server-ul nostru sa foloseasca cerificatul creat anterior.

Text

Description automatically generated

Ca si in cazul server-ului anterior trebuie creat un create ServerSocket care sa asculte pe un anumit port si Socket pentru conexiunea cu clientul. Doar ca de aceasta data vom folosi SSLServerSockets si SSLSockets. Un SSLServerSocket se obtine dintr-un SSLServerSocketFactory apeland metoda getDefault(), care creaza un ServerSocketFactory cu configuratiile de baza si apoi apeland metoda createServerSocket(int port\_number) care creaza SSLServerSocket-ul.

Si in acest caz, am introdus totul intr-un try-catch:

Text

Description automatically generated

Capitolul 1.2.2.3 Aplicatia EchoSSLClient

Importam clasele necesare

Text

Description automatically generated

Setam variabilele cu adresa si port ul server ului la care vrem sa ne conctam si setam proprietatile sistemului sa foloseasca certificatul create anterior.

Text

Description automatically generated

Cream un SSLSocket prin apelul functiei SSLSocketFactory.getDefault().createSocket(String:host, int:port) si apoi setam comunicarea cu server-ul, si acestea inconjurate de try-catch.

Text

Description automatically generated

Capitolul 1.2.2.4 Rularea aplicatiilor EchoSSL client-server

Daca totul a functionat cum trebuie, cele 2 console ar trebui sa afiseze urmatoarele mesaje:

EchoSSLServer:

Text

Description automatically generated

EchoSSLClient:

Text

Description automatically generated

Trimitem un mesaj de de EchoSSLClient catre EchoSSLServer incriptat.

Text

Description automatically generated

Text

Description automatically generated

Capitolul 1.2.2.5 Analizarea traficului aplicatiei EchoSSL client-server folosind wireshark

Trimitem un mesaj de pe EchoSSLClient catre EchoSSLServer

Text

Description automatically generated

Text

Description automatically generated

Wireshark a detectat folosirea protocolului TLSv1.3.

Graphical user interface, application

Description automatically generated

Capitolul 1.2.2.6 Negocierea versiunii protocolului folosit

Capitolul 1.2.2.6.1 Negocierea protocolului TLSv1.2

Pentru a seta varianta de protocol TLSv1.2 avem trei optiuni, fie setam negocierea protocolului TLSv1.2 in EchoSSLServer imediat dupa apelul functiei accept(), fie in EchoSSLClient dupa crearea socket-ului cu conexiunea catre server.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

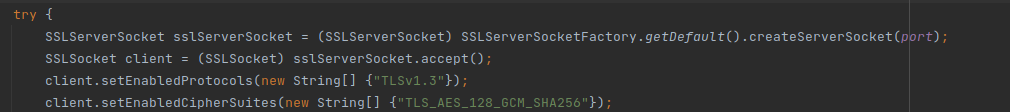
In oricare din cazurile mentionate mai sus, wireshark detecteaza acum ca protocol folosit TLSv1.2

A screenshot of a computer

Description automatically generated with medium confidence

Capitolul 1.2.2.6.2 Negocierea protocolului TLSv1.3

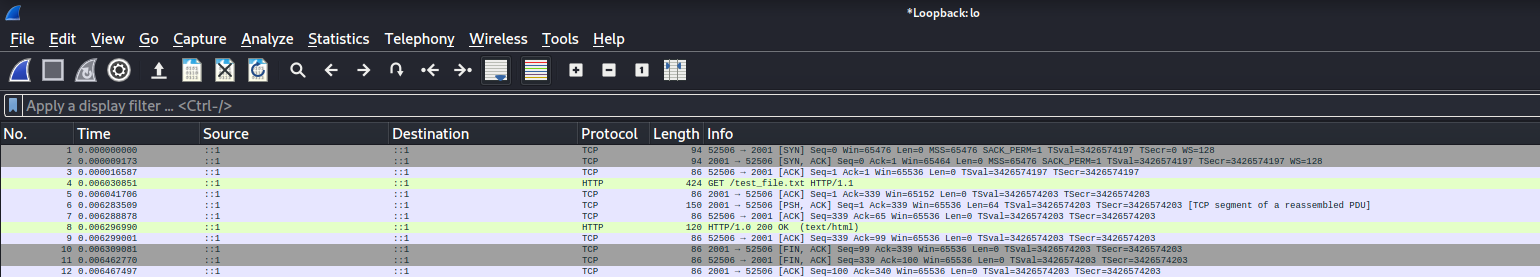
Pe langa faptul ca putem seta varianta de protocol prin apelul functiei setEnabledPretocols(String[] ), putem seta si ce cipher suites sa folosim prin apelul functiei setEnabledCipherSuites(String[] )

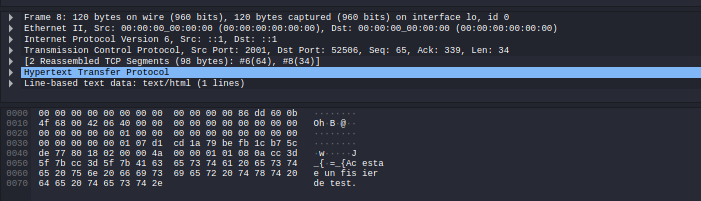


A screenshot of a computer

Description automatically generated with medium confidence

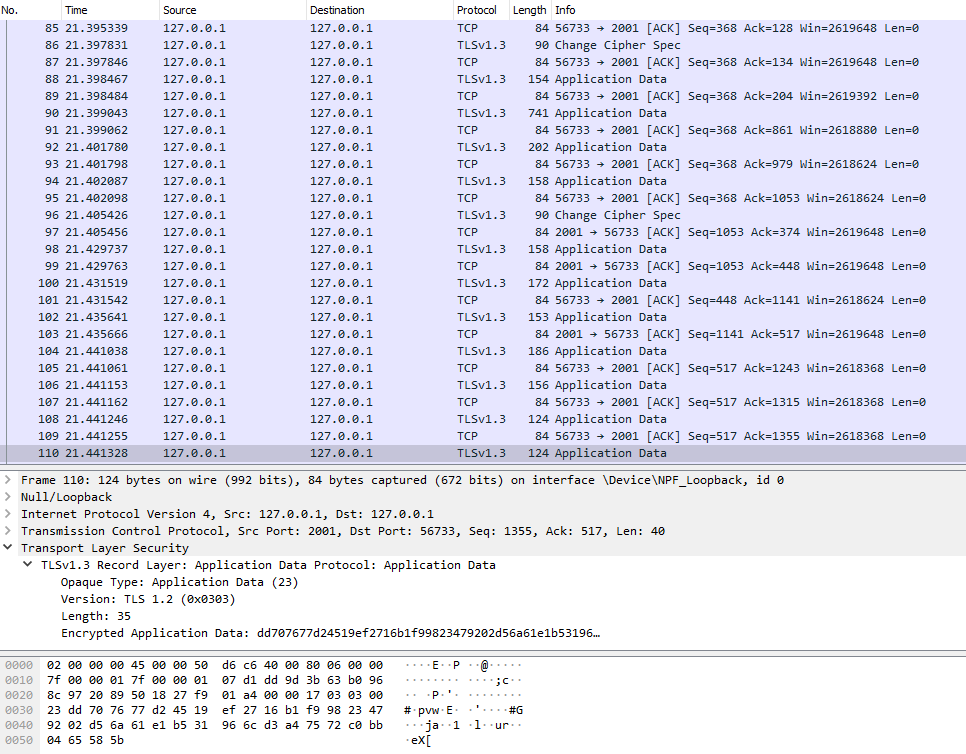
Captura traficului ClassFileServer in plain text:



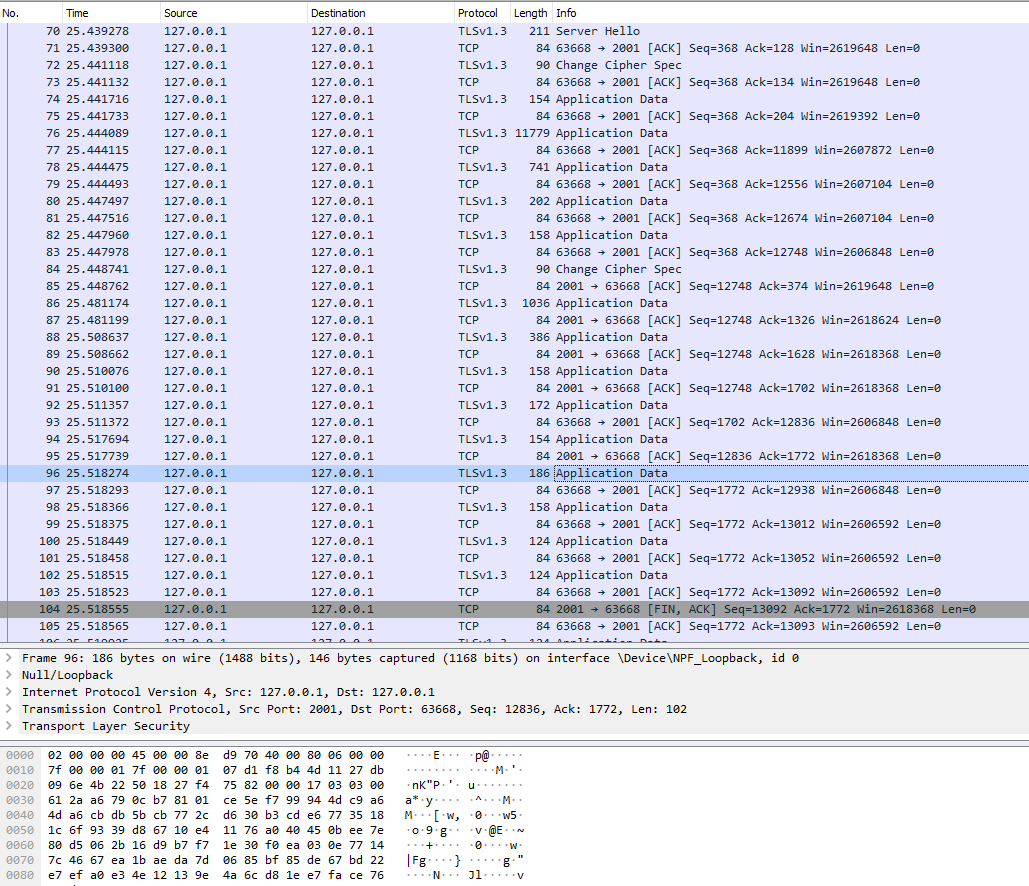


Adaugarea unui certificat in java truststore: 

Captura traficului ClassFileServer cu protocolul TLS fara autentificarea clientului:



Captura traficului ClassFileServer cu ClientAuthentication true:



ClassFileServer cu client authentication true atunci cand clientul nu are certificat: 