Sigurnost transportnog sloja - TLS

Lozić, Luka

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Luka Lozić

Sigurnost transportnog sloja - TLS

ZAVRŠNI RAD

Varaždin, 2019.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Luka Lozić

Matični broj:

Studij: Primjena informacijske tehnologije u poslovanju

Sigurnost transportnog sloja - TLS

ZAVRŠNI RAD

Mentor/Mentorica:

Izv. prof. dr. sc. Ivan Magdalenić

Varaždin, siječanj 2019.

Sadržaj

1. Uvod

Tema ovoga završnog rada jest „Sigurnost transportnog sloja – TLS“. Otkad je interneta postoji online komunikacija putem mreže; koja se koristi u javne, privatne, poslovne, akademske i Vladine potrebe. Svaki od navedenih sektora upotrebe Internet komunikacije vapi za nekim oblikom sigurnosti. Neovisno o sektoru, važnost tajnosti, autentifikacije i očuvanja integriteta uvijek postoji. Komunikacija između računala je u konstantnoj opasnosti od napada treće strane, jer se na relaciji putovanja poruke od pošiljatelja do primatelja može dogoditi napad kojim bi se ugrozila sigurnost informacije. Zbog želje za očuvanjem privatnosti transportnog sloja i osiguranja integriteta same poruke, stvoren je SSL (eng. *Secure Sockets Layer*), odnosno njegov nasljednik TLS (eng. *Transport Layer Security*).

Ovaj završni rad ima za cilj upoznati čitatelja s kriptografijom općenito, kriptografskim sustavima sa simetričnim i asimetričnim ključem, samim TLS-om, poviješću i usporedbom TLS-a i SSL-a; različitim algoritmima koji se koriste unutar TLS-a, koracima protokola rukovanja, te će na kraju biti prikazana implementacija TLS-a uz sažeti zaključak.

2. Kriptografija

Riječ kriptografija opisuje znanstvenu disciplinu koja se bavi kriptiranjem i dekriptiranjem poslanih poruka. Kriptografija je riječ grčkoga porijekla te je sastavljena od riječi (grč. κρυπτός) što u prevedenome znači „skriven“ te riječi (grč. γράφειν) čija je prevedenica glagol „pisati“, što bi se zajedno moglo protumačiti kao „tajnopis“.

Kriptografija je nastala s ciljem očuvanja tajnosti prilikom slanja poruka. Prilikom postupka primjene kriptografije, pošiljateljeva poruka mijenja čitljiv oblik u oblik koji može razumjeti samo primatelj poruke. Promjena oblika poslanih informacija iz čitljivog u nečitljivo stanje zove se kriptiranje, odnosno šifriranje.

Često se u literaturi vezanoj uz kriptografiju spominju Alice, Bob i Eve. Kroz primjer, Alice i Bob su pošiljatelj i primatelj koji razmjenjuju poruke preko komunikacijskog kanala koji nije osiguran, dok je Eve negativac koji prisluškuje poruke unutar tog komunikacijskog kanala prilikom komunikacije Alice i Boba. Da se važne informacije skriju prilikom slanja poruka, poruke se prije slanja kriptiraju u ključ. „Pošiljalac transformira otvoreni tekst koristeći unaprijed dogovoreni ključ. Taj postupak se naziva šifriranjem, a dobiveni rezultat šifratom (engl. ciphertext) ili kriptogramom“ („Klasična kriptografija“, bez dat).

Postoje dvije vrste kriptografskog sustava; to su asimetrični kriptografski sustav te simetrični kriptografski sustav.

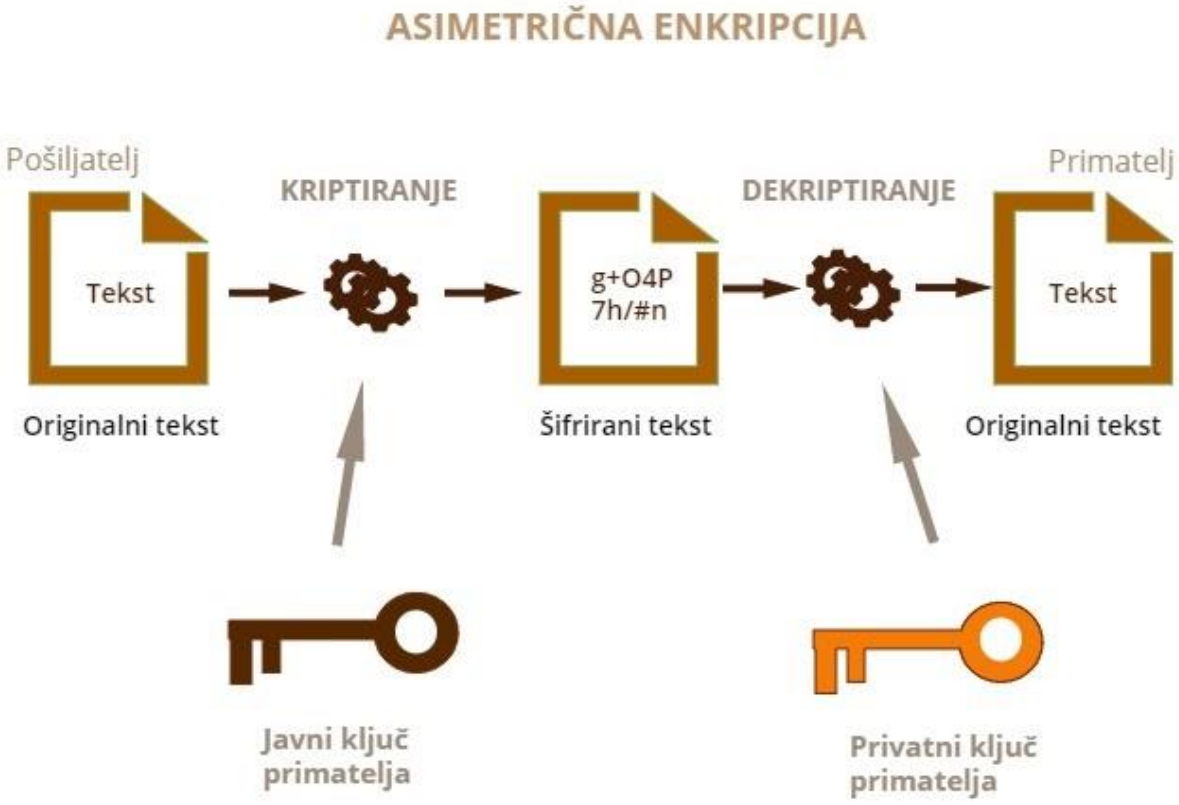
2.1. Asimetrična kriptografija

Jedan od dva poznata oblika kriptografije jest sustav asimetrične kriptografije koji se također naziva enkripcija javnim ključem. Navedeni sustav kriptografije koristi jedinstveni ključ za kriptiranje te poseban ključ za dekriptiranje.

Kod razmjene podataka prisutni su pošiljatelj i primatelj. U ovom slučaju, kod asimetrične kriptografije pošiljatelj i primatelj koriste par ključeva koji je generiran na strani servera - javni i privatni ključ. Javni ključ je vidljiv svima, te on služi za kriptiranje, a privatni ključ je vidljiv samo vlasniku ključa, te on služi za dekriptiranje kriptiranih primljenih poruka.

U slučaju da Jadranka želi poslati poruku Branku, Jadranka uzima javni ključ od Branka te kriptira svoju poruku njegovim javnim ključem. Kriptirana poruka šalje se primatelju, te je primatelj (u ovome slučaju Branko) dekriptira svojim privatnim ključem. Kriptirana poruka je u toku slanja potencijalno bila vidljiva trećoj strani - Milici, ali je u kriptiranom obliku poruka bila nečitljiva i neiskoristiva prisluškivaču.

Neki od poznatijih asimetričnih algoritama koji se aktivno koriste su RSA (Rivest–Shamir–Adleman), Diffie Hellman algoritam razmjene ključeva, DSA (eng. *Digital Signature Algorithm*) te ECDSA (eng. *Elliptic Curve Digital Signature Algorithm*).



Slika 1: Asimetrična enkripcija (*Public key algorithm*, lipanj 2013.)

2.1.1. RSA

Akronim RSA je nastao od prezimena svojih stvoritelja; Rona Rivesta, Adia Shamira te Leonarda Adlemana.

Asimetrični kriptosustav, pod nazivom RSA, jedan je od najpopularnijih asimetričnih sustava za razmjenu ključeva, te je također jedan od prvih kriptosustava koji je korišten za sigurnu razmjenu podataka. Poanta sustava je postojanje dva ključa - jedan ključ za kriptiranje, koji je javni i drugi ključ za dekriptiranje, koji je privatni.

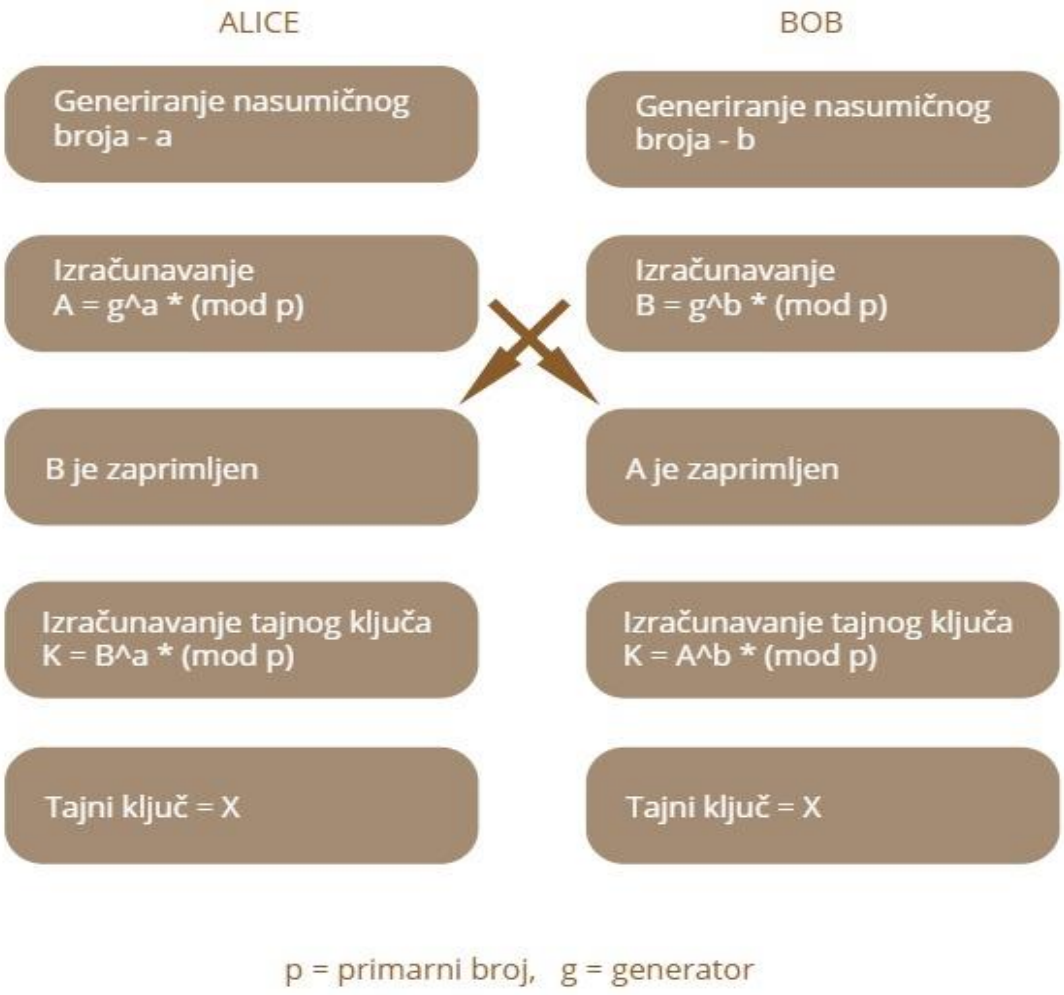
RSA algoritam se temelji na korištenju prostih brojeva, koji u ovome slučaju služe za generiranje ključeva (javnih i privatnih).

2.1.2. Diffie- Hellman

U svijetu poznata metoda kao Diffie-Hellman protokol (eng. *Diffie Hellman secret key exchange*), jest metoda razmjene ključa koja je nazvana po Whitfieldu Diffieu i Martinu Hellmanu.

Ključna činjenica kod asimetrične metode razmjene ključeva Diffie-Hellman je ta, da se tijekom razmjene ključa ne dijele informacije, već se one stvaraju na strani potencijalnih vlasnika ključa. Diffie-Hellman algoritam asimetrične razmjene ključeva, koristi se kako bi se razmijenili ključevi za daljnju simetričnu komunikaciju, odnosno razmjenu podataka enkriptiranim putem.

Algoritam djeluje na sljedeći način; u prostoru između Alice i Boba postoje unaprijed dogovorena dva broja *n* i *g*, ti brojevi su vidljivi svima. Nakon toga, Alice generira nasumični broj *a,* te Bob također stvara nasumični broj, u ovome slučaju *b*. Nasumični brojevi *a* i *b* su privatni i vide ih samo stvaratelji. Tada si Alice i Bob šalju rezultate matematičke operacije kojom je Alice stvorila vrijednost *X* kombinacijom brojeva *n, g* i *a,* te je Bob stvorio broj *Y* kombinacijom vrijednosti *n, g* i *b*. Kada Alice zaprimi vrijednost *Y* i Bob zaprimi broj *X*, oboje miješaju dobivene brojeve sa svojim privatnim generiranim ključem s početka. Alice bi tada pomiješala broj *Y* s vrijednošću *a,* dok bi Bob pomiješao brojeve *X* i *b.* Konačne pomiješane vrijednosti bi trebale vraćati jednaki rezultat, odnosno kod Alice i Boba bi se trebali generirati isti ključevi koji bi služili za daljnju enkripciju.

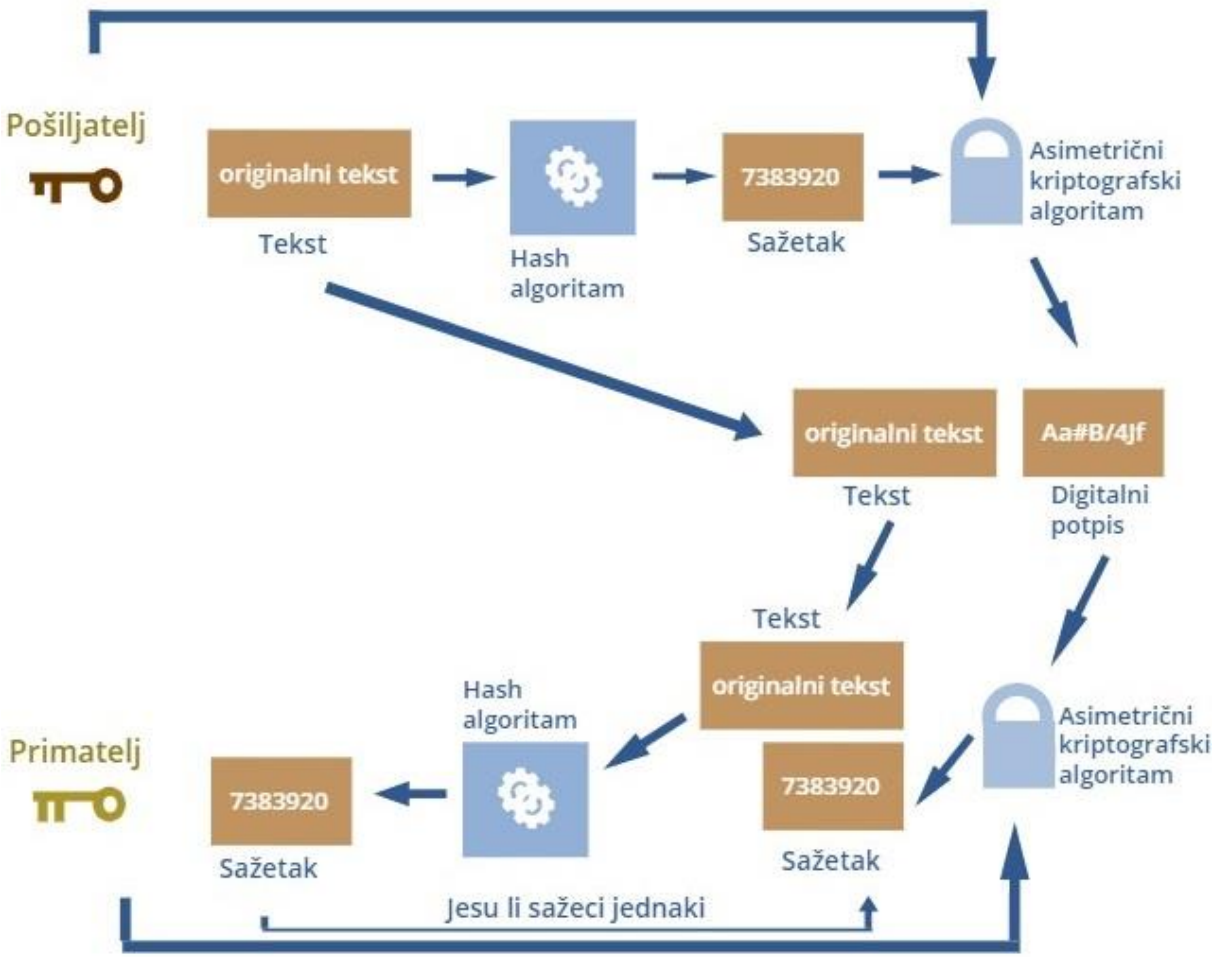


Slika 2: Diffie Hellman razmjena ključeva (*Diffie-Hellman Algorithm*, bez dat.)

2.1.3. DSA

Službeni naziv DSA (eng. *Digital Signature Algorithm*) označava digitalni potpis. Navedeni algoritam koristi sustav asimetrične razmjene ključeva. Iste namjene kao i fizički potpis, digitalni potpis je potvrda o očuvanju integriteta, neporecivosti te autentičnosti pošiljatelja, odnosno poslanog dokumenta. Naspram raznih prednosti, DSA ne jamči tajanstvenost.

DSA radi na jednostavan način; recimo da pošiljatelj želi poslati dokument primatelju, kako DSA koristi asimetričnu kriptografiju, pristupljeni su javni i privatni ključ, koje u ovome slučaju generira pošiljatelj. Pošiljatelj ima privatni, dok primatelj uzima javni ključ. Prije slanja dokumenta, pošiljatelj kreira sažetak istoga, putem *hash* funkcije, te dobiveni sažetak kriptira sa svojim privatnim ključem. Nakon toga, primatelju se šalje kriptirani sažetak uz nekriptirani dokument. Primatelj zaprima kriptirani sažetak i nekriptirani dokument, te dekriptira kriptirani sažetak pošiljateljevim javnim ključem i dobiva dekriptirani sažetak. U isto vrijeme, iz dobivenog nekriptiranog dokumenta izračunava sažetak putem iste *hash* funkcije koja se je koristila na pošiljateljevoj strani prije slanja. Zadnji korak je usporedba izračunatog sažetka i dobivenog sažetka - ukoliko je rezultat usporedbe jednako, tada je sačuvan integritet poruke te je ispunjena obećana neporecivost i autentičnost primatelja.



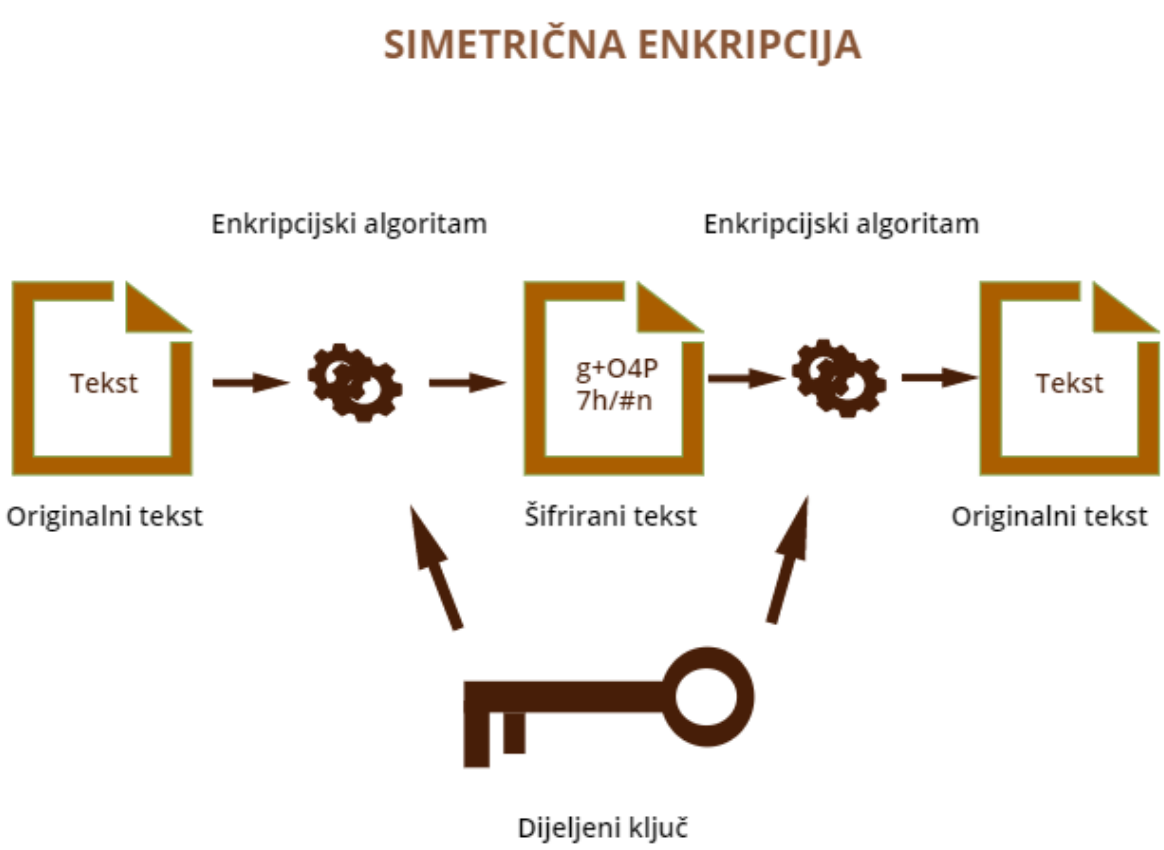
Slika 3: Digitalni potpis (*What is digital signature*, lipanj 2017.)

2.2. Simetrična kriptografija

Drugi poznati oblik kriptografije jest sustav simetrične kriptografije, koja se također naziva enkripcija privatnim ključem. Kod simetrične kriptografije je poznato da se kriptiranje i dekriptiranje vrši s jednim, odnosno istim ključem. Isto kao i kod asimetrične kriptografije, prisutni su pošiljatelj i primatelj koje ćemo radi lakšeg raspoznavanja nazvati Alice, Bob i Eve, kao i do sada.

Alice želi poslati poruku primatelju kroz komunikacijski kanal. Radi potencijalnog prisluškivača potrebno je poruku prije slanja kriptirati. Poruka se kriptira simetričnim ključem kroz jedan od enkripcijskih algoritama. Kako je poruka kriptirana, nitko je ne može pročitati, te se takva šalje Bobu. Bob dekriptira kriptiranu poruku sa simetričnim ključem te se oblik poruke vraća u čitljiv oblik.

Simetričnih algoritama ima više, a najkorišteniji su AES (eng. *Advanced Encryption Standard*), DES (eng. *Data Encryption Standard*) i 3DES (*Triple Data Encryption Algorithm*).



Slika 4: Simetrična enkripcija (*Asymmetric encryption*, bez dat.)

2.2.1.DES

DES je skraćenica od riječi *Data Encryption Standard*. Navedeni pojam predstavlja jedan od najpoznatijih oblika kriptografskog sustava sa simetričnim ključem. Simetrični kriptografski sustav DES je publiciran 1977. godine, a dvije godine nakon je standardiziran. Uobičajeni stari *Data Encryption Standard* se ne koristi kao prije zbog nedovoljne dužine ključa, što ujedno predstavlja nesigurnost.

Data Encryption Standard funkcionira na sljedeći način: prije slanja poruke primatelju ona se dijeli na blokove od 64 bita, te se miješa sa ključem od 54 bita. Mješavina blokova poruke i ključa se kriptira u blokove od po 64 bita svaki. Nakon kombiniranja čitljivog teksta i ključa, dobiveni šifrirani blokovi se šalju primatelju na dekripciju. Kada šifrirani blok prijeđe komunikacijski kanal, on biva dešifriran od strane primatelja simetričnim ključem.

2.2.2. 3DES

Napredniji brat DES-a je 3DES (eng. *Triple Data Encryption Standard*). Navedeni enkripcijski algoritam je također temeljen kao kriptografski sustav sa simetričnim ključem. U principu je 3DES, implementacija DES-a tri puta na jedan blok bitova. Modificirana verzija *Data Encryption Standard-a* je danas u upotrebi zbog dovoljne dužine ključa, što predstavlja zadovoljavajuću razinu zaštite za odgovarajuće područje primjene tj. implementacije enkripcijskog algoritma.

2.2.3. AES

AES, odnosno *Advanced Encryption Standard*, ili originalnim imenom Rijndael, je moderna zamjena za DES. Ovaj oblik enkripcije je oblik kriptografskog sustava sa simetričnim ključem, koji predstavlja viši oblik sigurnosti nad svojim prethodnicima.

Kako se je prilikom DES-a poruka dijelila u blokove od 64 bita, u AES slučaju se poruka dijeli na blokove od 128 bita. U principu je AES sustav rada isti kao i kod DES-a. Prije slanja poruke, ona se tvori u blokove od po 128 bitova te se miješa s ključem duljine 128, 192 ili 256 bita. Duljina primijenjenog ključa je minimalne duljine 128 bita, odnosno dužine samog bloka bita poruke. Ključ također može biti veće duljine od duljine bloka bitova. Nakon kombiniranja bitova poruke i ključa, dobiva se šifrirani blok od 128 bita koji se šalje primatelju, te tako blok po blok.

2.3. Digitalni certifikat

Digitalni certifikat je jedan od korisnijih te potrebnijih elemenata kada se govori o sigurnoj komunikaciji i razmjeni podataka putem „mreže svih mreža“. Ono što digitalni certifikat predstavlja je uvjerenje da je stranica s kojom se komunicira zbilja ta stranica, te da je ona provjerena i da su njeno postojanje, legitimnost i integritet potvrđeni od strane društva za izdavanje certifikata (eng. *certificate authority*) -skraćenicom CA.

Kada je pouzdanost u pitanju, svaka sigurna stranica na internetu bi trebala biti certificirana - bilo od CA-a ili samo-certificirana. *Certificate authority* je tijelo koje izdaje certifikat određenim stranicama nakon provjere stranice koja se želi certificirati. Samo-certificiranje je postupak izdavanja potvrde po načelu „samome sebi“. Posjedovanjem certifikata određena stranica ima potvrdu da ta stranica predstavlja tijelo za koje tvrdi da je.

Provjera digitalnog certifikata se vrši prilikom rukovanja (eng. *Handshake*) klijenta i servera. Server, odnosno traženo web mjesto, daje certifikat na provjeru klijentu u drugom dijelu rukovanja, tada klijent može provjeriti legitimnost traženog web mjesta. Klijent već unaprijed na računalu ima instaliran niz potvrđenih izdavača certifikata. Dobiveni certifikat se uspoređuje s nizom pred-instaliranih certifikata na računalu, ako je certifikat potvrđen, stranica je legitimna.

3. TLS

Glavni dio ovoga završnog rada je TLS, odnosno *(*eng*,Transport Layer Security protocol)* predstavlja sigurnu komunikaciju između dva subjekta, a zadaća mu je zaštita integriteta poslanih poruka, osiguranje legitimnosti međusobnih stranica te očuvanje privatnosti prilikom komunikacije i razmjene podataka.

TLS protokol se u današnje vrijeme koristi svugdje na mreži - rijetke su one stranice koje nemaju TLS certifikat, te se stoga one izbjegavaju zbog sigurnosnih razloga. Osim očitih razloga korištenja potvrde, jedan od razloga važnosti imanja certificirane stranice je izbjegavanje svojevrsne stigme koju nosi neosigurana Internet stranica.

Svakodnevnome korisniku je sigurnost stranice vidljiva po oznaci koja se nalazi u lijevom dijelu mjesta za unos Internet adrese. Ako je početak mrežne adrese (npr. *https)*, tada možemo biti sigurni u legitimnost i sigurnost te web stranice. U suprotnome, ako početni dio URL trake nosi naziv bez ključnog slova „*s*“ (npr.*http*), tada možemo biti sigurni da se radi o stranici koja nema implementirani TLS, odnosno SSL protokol.

Neke od zanimljivih činjenica što se tiče SSL-a; „Pomoću *Secure Socket Layer protokola* se osigurava e-mail (eng. *Simple Mail Transfer Protocol*, SMTP), video poziv (eng. *Voice over IP*, VoIP), protokol za razmjenu datoteka (eng. *File Transfer Protocol,* FTP) i mnogi drugi protokoli, a ne samo Internet pretraživanje (HTTP). Također se mogu osigurati terminal protokoli IBM 3270 ili 5250 iz 1970.-ih preko SSL-a.“ („Jinwoo Hwang“, 2012.).

3.1. TLS povijest

TLS jest pojam usko povezan sa SSL-om (eng. *Secure Socet Layer*), poglavito zbog toga što je TLS izravni nasljednik SSL-a. Potreba za takvim sustavom sigurne komunikacije, odnosno sustavom kao što je SSL, se je pojavila početkom 1990.-ih godina prošloga stoljeća. Razlog potrebe takvog sustava je bio sve veća i ubrzana korištenost Interneta, te samim time i veća izloženost korisnika mreže konstantnoj prijetnji krađe informacija, te njihova zlouporaba od strane hakera. Glavni cilj je bio uspostaviti tzv. „*https*“, mješavinu „*http*“ i SSL protokola, što bi pružalo sigurnu razmjenu podataka kroz „mrežu svih mreža“.

Rezultat rada poznate tvtke Netscape Communications, 1994. godine bio je, dobro nam poznati, (eng. *Secure Socket Layer*), tj. SSL 1.0. Sustav je imao puno potencijala i prednosti, ali zbog svojih sigurnosnih mana nikada nije bio publiciran. Ono što je predstavljalo problem bilo je korištenje nekvalitetnih algoritama. Kako je tada SSL zablistao u centru pozornosti, privukao je pažnju hakera kao njegovu najveću prijetnju, te se od početaka razvoja sustava SSL-a vodi hladni rat između hakera i programera koji se zalažu za internetsku sigurnost.

Bitka za internetsku sigurnost je započela.

Verzija SSL-a 2.0, tj. SSLv2 je puštena u upotrebu u veljači 1995. godine te je bila na sceni tek godinu dana, dok je nije zamijenila bolja verzija SSL 3.0, odnosno SSLv3. Značajno poboljšanje SSL-a 2.0 u odnosu na prethodnika bila je implementacija „*hash*“ MD5 algoritma. Iako poboljšana, prva publicirana verzija SSL-a, *Secure Socket Layer* 2.0 je imala svoje mane, zbog kojih je ubrzo bila zamijenjena. Unatoč novoj verziji, SSLv2 se je aktivno koristio sve do 2011. godine, kada je na snagu stupila „prohibicija“ korištenja od strane IETF-a (eng. *Internet Engineering Task Force-a*).

„Prohibiting Secure Sockets Layer (SSL) Version 2.0

Abstract

This document requires that when Transport Layer Security (TLS) clients and servers establish connections, they never negotiate the use of Secure Sockets Layer (SSL) version 2.0. This document updates the backward compatibility sections found in the Transport Layer Security (TLS).“(„Prohibiting Secure Sockets Layer (SSL) Version 2.0“, Ožujak 2011.)

Davne 1996. godine Paul Kocher, Phil Karlton te Alan Freier su, uz ostatak tima, nadogradili dotadašnji SSL protokol 2.0, te je nova verzija predstavljena pod imenom SSLv3. Mlađi protokol je predstavio određenu ljestvicu koja se je poštivala. Unaprjeđenje u odnosu na prethodnika je bilo postojeću „*hash*“ funkciju zamijeniti kombinacijom MD5“ i „SHA-1 *hash* algoritmima. SSLv3 se je koristio aktivno i u velikoj mjeri sve do lipnja 2015. godine, kada je također predstavljena prohibicija za navedeni protokol.

Uz postojeći SSL 3.0 protokol, u siječnju 1999. godine je predstavljena njegova nadogradnja TLS 1.0 (eng. *Transport Layer Security*), koja se koristila paralelno sa starijim bratom. Protokoli SSLv3 i TLS 1.0 su 1999. godine predstavljali ulaz u sigurnu razmjenu podataka putem interneta. Novi protokol su razvili Tim Dierks i Christopher Allen.

Važno je napomenuti kako je TLS u potpunosti baziran na SSL-u, no nije pokazivao kompatibilnost sa SSL-om. Mana TLS 1.0 je bila ta što nije imao zaštitu protiv CBC (eng. *Cipher Block Chaining*) napada.

Godine 2006. je publicirana nova, poboljšana verzija TLS protokola; to je bio protokol TLS 1.1 kod kojega je bio riješen problem, te primijenjena zaštita protiv CBC napada.

Sljedeća verzija je izašla u kolovozu 2008. godine, te je bila nazvana imenom TLS 1.2. Spomenuta nadogradnja se koristi uspješno i bez većih problema u današnje vrijeme uz algoritam TLS 1.3. Poboljšanja u odnosu na prethodnika su; zamjena spomenute *hash* kombinacije MD5-SHA1 s *hash* algoritmom SHA-256, mogućnost klijenta i servera da odaberu *hash* algoritme te algoritme potpisa, uvođenje AES algoritma i uklanjanje DES algoritma.

Najnovija verzija TLS 1.3 je publicirana u kolovozu 2018. godine, te predstavlja mnogo noviteta u svojem arsenalu. Neke od novih stvari u TLS 1.3 su ukidanje podrške za eliptične krivulje (eng. *Eliptic Curves*), ukidanje podrške za *hash* algoritme MD5 te SHA-224, dopuštena je veća veličina *cookie-a*, potpora 1-RTT rukovanju i dr.

3.2. TLS handshake

TLS sigurna razmjena podataka putem osiguranog kanala započinje rukovanjem, tzv. (eng. *Handshake)*. Poanta rukovanja je uspostavljanje sigurne veze i povjerenja između klijenta i servera. Laički rečeno, glavni cilj prije razmjene podataka je putem asimetrične enkripcije razmijeniti ključeve za simetričnu enkripciju između klijenta i servera.

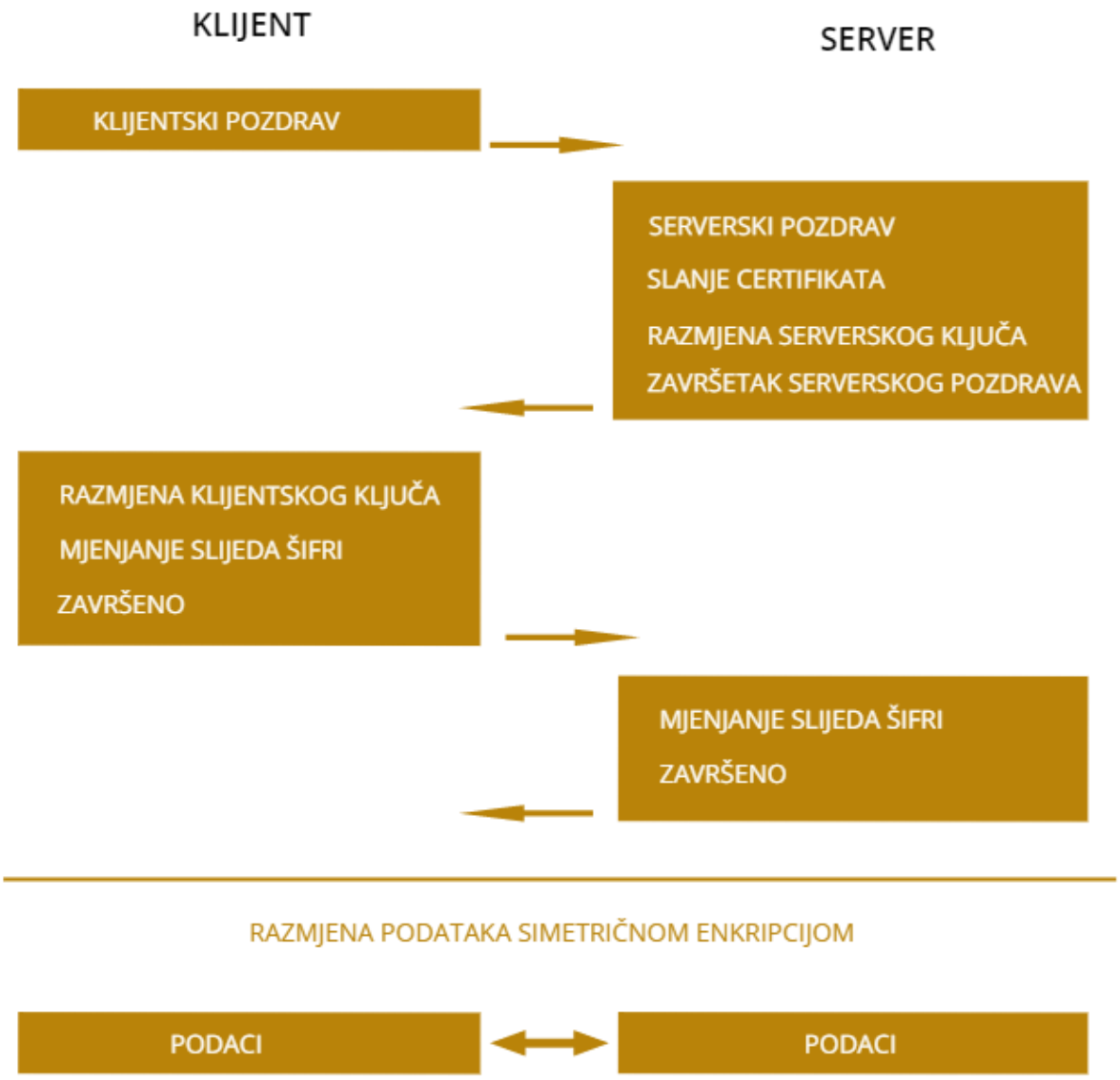
Transport Layer Security rukovanje se odvija između klijenta i servera. Klijent je u ovome slučaju web preglednik, dok neka stranica predstavlja server. *Handshake* se odvija u nekoliko koraka. Prvi korak je od strane klijenta te se tu odvija početno uspostavljanje konekcije, time da klijent šalje *hello* poruku serveru, uz koju šalje popis *cipher suit-eva*, koje podržava klijent strana.

Cipher suite je skup, odnosno kombinacija algoritama koji bi se koristili za enkriptiranu komunikaciju između preglednika i stranice. Server odabire *cipher suite* koji mu odgovara, te se prema odabranome skupu algoritama odvija daljnja komunikacija.

Drugi korak *handshake* protokola je od strane servera. Web stranica zaprima početnu poruku klijenta u kojoj su sadržani *hello* poruka te popis *cipher suiteva*. Serverska strana odabire određeni *cipher suite*, te nakon toga šalje klijentu povratnu *hello* poruku, uz certifikat koji je ujedno i javni ključ. Uz to sve, server šalje zadnju poruku u drugome koraku koja je *hello done,* što označava da je prvi dio rukovanja završen.

Treći korak protokola rukovanja je na klijentu. Klijent zaprima poruke servera od kojih su najvažnije poruke sam certifikat, koji je ujedno i javni ključ. Certifikatom se utvrđuje da je integritet stranice provjeren te da je stranica ona za koju se tvrdi da je. Javni ključ u ovome dijelu služi za asimetrični dio enkripcije, kako bi se preko njega razmijenili podaci bitni za uspostavljanje simetrične enkriptirane razmjene podataka. Klijent generira *pre-master secret* te svoj simetrični ključ. Generirani *pre-master secret* klijent šalje natrag serverskoj strani u kriptiranom obliku, pomoću serverskog javnog ključa. Web stranica također zaprima klijentsku poruku *change cipher spec*, koja govori, u principu, da je klijentska strana stvorila svoj simetrični ključ, te da je spremna za simetričnu razmjenu podataka. Zadnja poruka prema serveru je *client finished*, te ona javlja da je klijentska strana gotova sa svojim dijelom rukovanja.

Četvrti korak, koji je ujedno i zadnji, je serversko primanje *pre master secret*-a, i njegovo dekriptiranje svojim privatnim ključem. Server generira svoj simetrični ključ pomoću *pre master secret*-a. Klijentu se šalje poruka *change cipher spec,* uz poruku *finished,* koja označava da je asimetrični dio enkripcije i rukovanja završen. Od tog trenutka započinje razmjena podataka simetričnom enkripcijom.



Slika 4: Protokol rukovanja

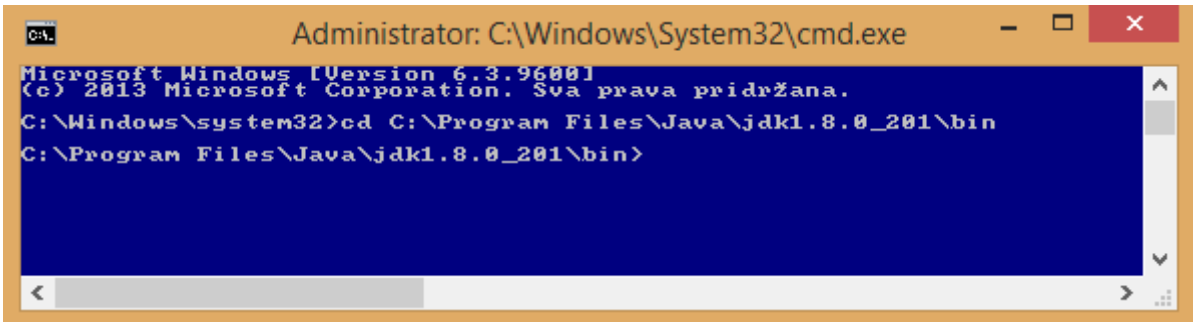
4. Praktična komponenta

4.1. Stvaranje spremnika ključeva

Kao što naslov ove završne radnje govori, tema je sigurnost transportnog sloja, TLS. Kako bismo iz prve ruke uvidjeli o čemu se ovdje radi, u ovome završnome radu će biti opisana i razrađena implementacija TLS-a 1.2, kako bi se osigurala sigurna komunikacija između servera i klijenta. U primjeru će server predstavljati određenu stranicu, dok će klijenta predstavljati *Google Chrome* preglednik.

Za potrebe praktične komponente ovoga završnog rada, koristio sam programski jezik Java (JDK 8), razvojno okruženje odnosno IDE (eng. *Integrated development environment*) Eclipse 4.9.0 te server (eng. *Web container*) WildFly 14.

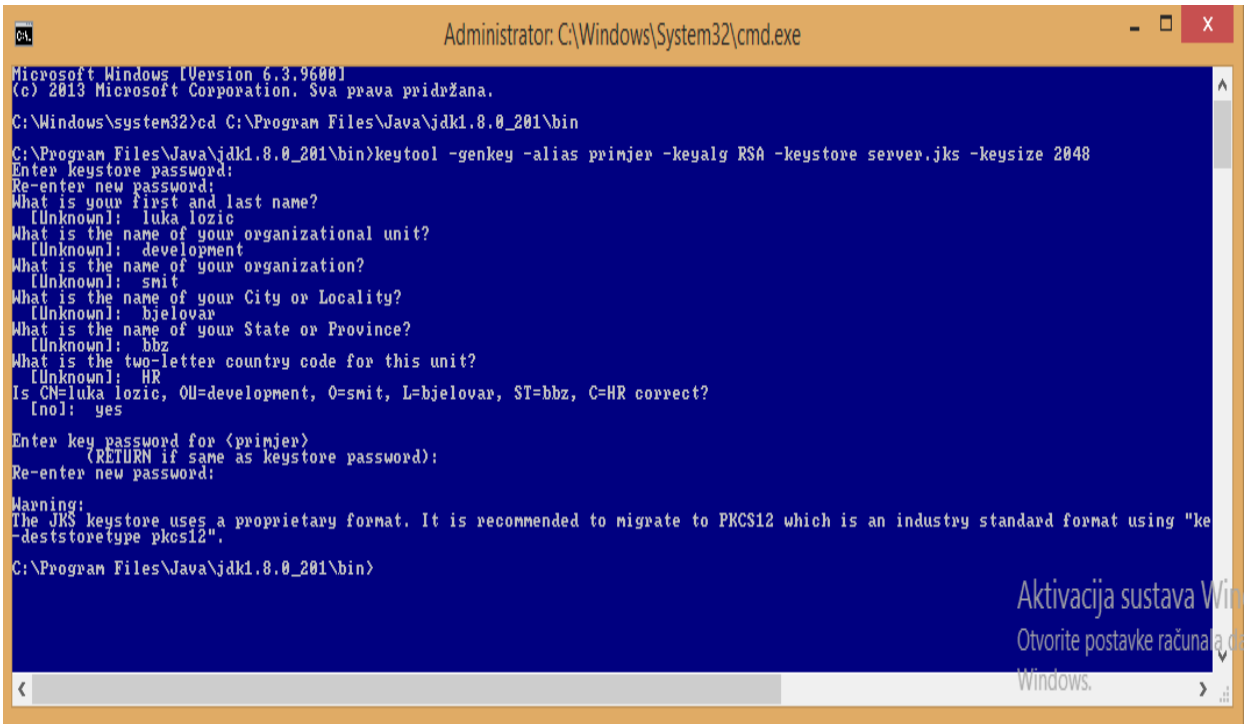
Prije implementacije same sigurnosti neophodno je stvoriti par ključeva, odnosno certifikat koji će nam koristiti kod serverske strane. Novonastali javni i privatni ključ te certifikat su stvoreni putem java ugrađenog modula, zvanog Java *keytool*, preko *Command prompta.* U *Command prompt,* koji mora biti korišten preko administratorskog moda, je ponajprije upisana naredba kojom se mjenja direktorij;



Slika 5: Putanja keytool direktorija

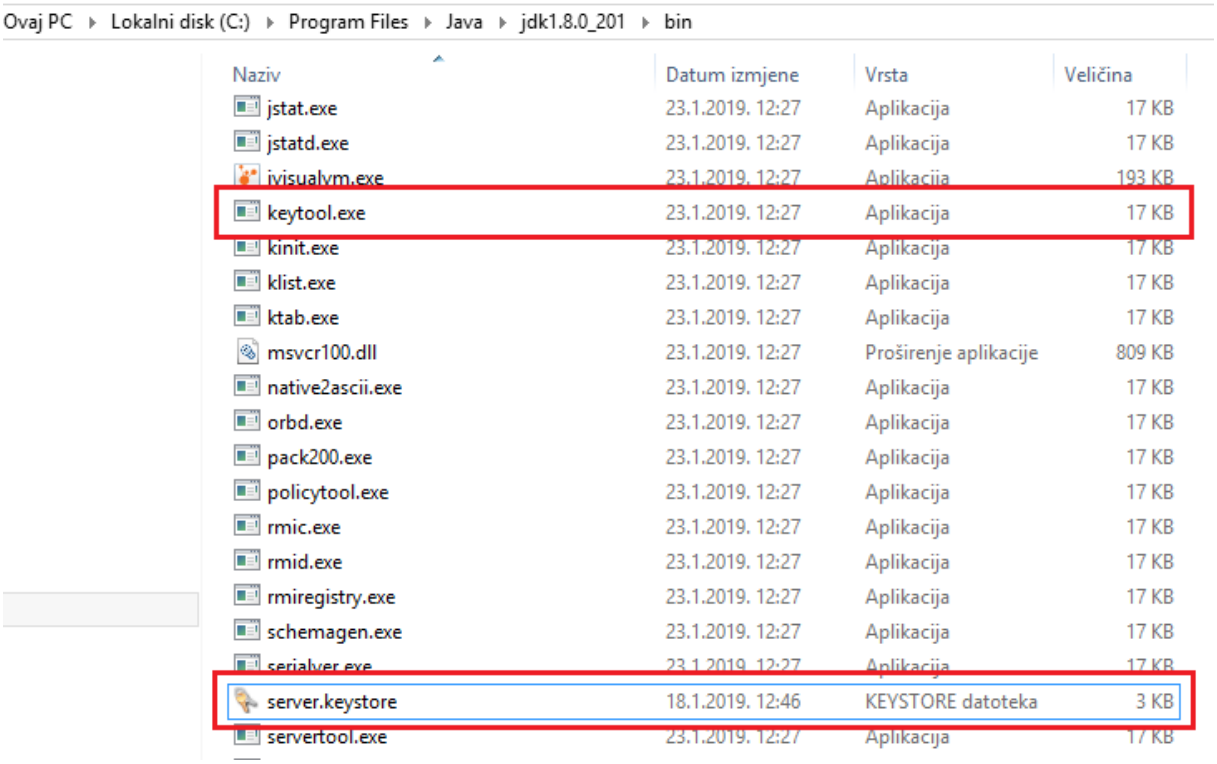
Dolaskom u direktorij gdje se nalazi alat za generiranje ključeva, dobivamo mogućnost kreiranja spremnika za ključeve. Zatim se upisom sljedećih naredbi stvara par ključeva. Prvo je bitno navesti koji izvedbeni modul se želi koristiti iz direktorija *bin*, u ovome slučaju taj modul se zove *keytool*. Potom se unose podaci kojima određujemo *alias* ključa, algoritam za razmjenu ključeva, ime datoteke koju će nositi par ključeva te veličina ključa. Važniji podaci su algoritam asimetrične enkripcije te veličina ključa, u ovome slučaju je algoritam razmjene ključeva RSA, te veličina spremišta ključeva u bitovima tj. njih 2048.

Pritiskom na tipku enter dobivamo sekvencijalni izbor sljedećih opcija; lozinka spremnika za ključeve te informacije stvaraoca ključa.



Slika 6: Kreiranje ključeva

Kreiranjem samih ključeva koji se nalaze u datoteci server.jks, napravljen, on izgleda ovako u direktoriju gdje se nalazi modul za generiranje ključeva;



Slika 7: Kreiranje ključeva

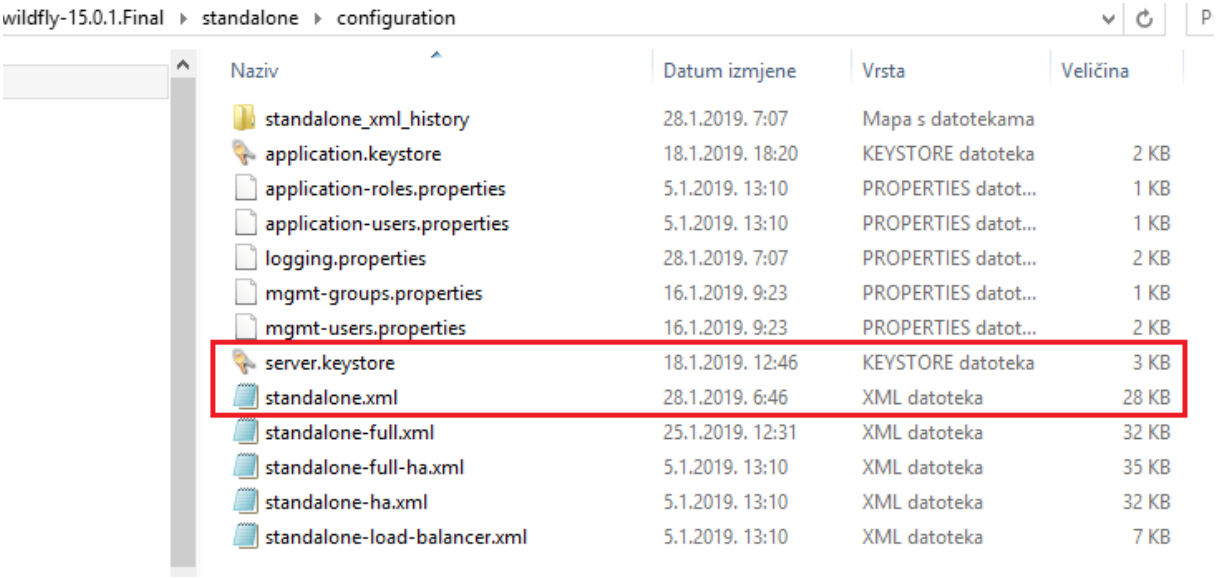
Uokvireno crvenom bojom su izvršna datoteka keytool.exe, kojom smo kreirali dolje označeni spremnik za ključeve server.jks.

4.2. Konfiguracija servera

Podsjetimo se da je sigurna konekcija temeljena i postavljena na serveru. Za serverski dio praktične komponente ovoga završnog rada sam koristio server WildFly verzije 14. Navedeni server sam odabrao ponajviše jer sam već upoznat s osnovnim principima rada tog serverskog modula u paru s Eclipse izvedbenom okolinom.

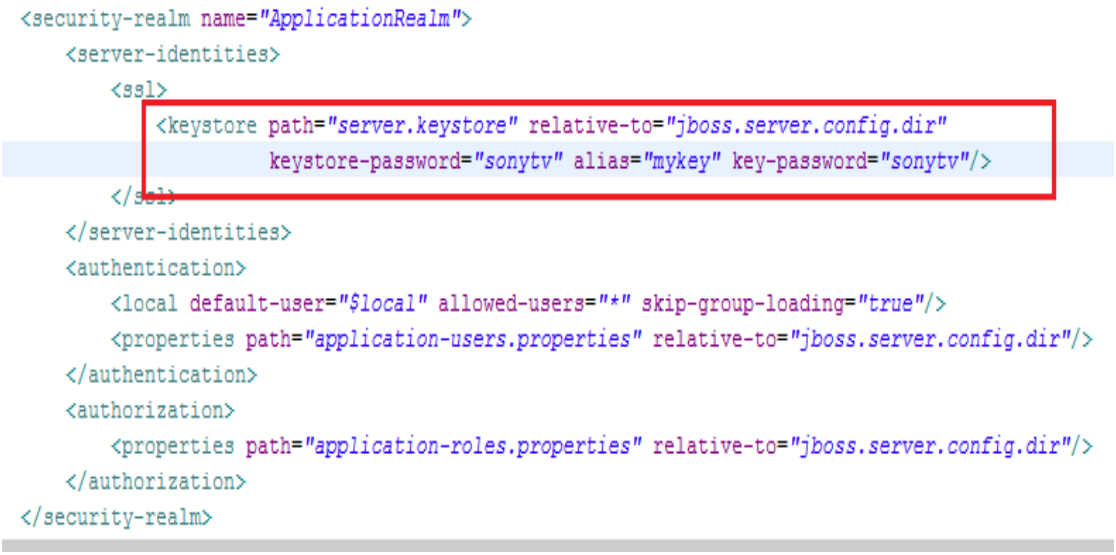
Server ima svoje konfiguracije, podesive opcije.

Prije konfiguriranja opcija servera, potrebno je napravljeno spremište ključeva smjestiti na server;



Slika 8: Putanja spremanja ključeva

Ono što je uokvireno, su preseljena datoteka server.keystore te xml datoteka standalone.xml. Kada su ključevi premješteni na server, moguće ih je implementirati u standalone.xml datoteci, ovisno o željenim potrebama. Osim konfiguracije ključeva, konfiguracijska datoteka standalone nudi mogućnosti upravljanja i drugim serverskim opcijama, kao što su kreiranje *security-realma* te promjena portova. Sljedeće slike će interpretirati kod unutar standalone.xml datoteke, koji koristi praktična komponenta ovoga rada za osiguranu konekciju.



Slika 9: Implementacija spremišta ključeva

Na slici iznad je prikazana implementacija spremišta ključeva kroz kod, unutar standalone.xml datoteke. Korišteni *security-realm* za uporabu „https-a“ je ApplicationRealm. Unutar spomenutog *security-realm-a* se nalazi mjesto za unos željenog spremišta ključeva, te uz sami naziv unosimo i lozinku te alias *keystore*-a.

```
<socket-binding-group name="standard-sockets" default-interface="public"
                      port-offset="${jboss.socket.binding.port-offset:0}">
    <socket-binding name="management-http" interface="management" port="${jboss.management.http.port:9990}"/>
    <socket-binding name="management-https" interface="management" port="${jboss.management.https.port:9993}"/>
    <socket-binding name="ajp" port="${jboss.ajp.port:8009}"/>
    <socket-binding name="http" port="${jboss.http.port:8080}"/>
    <socket-binding name="https" port="${jboss.https.port:8443}"/>
    <socket-binding name="txn-recovery-environment" port="4712"/>
    <socket-binding name="txn-status-manager" port="4713"/>
    <outbound-socket-binding name="mail-smtp">
        <remote-destination host="localhost" port="25"/>
    </outbound-socket-binding>
</socket-binding-group>
```

Slika 10: Portovi za korištenje

Uokvireno pokazuje portove koji se koriste za neosiguranu, „*http*“ te osiguranu,“*https“* konekciju. Valja napomenuti kako je port neosigurane konekcije 8080, a temelj osigurane konekcije se izvršava na portu 8443.

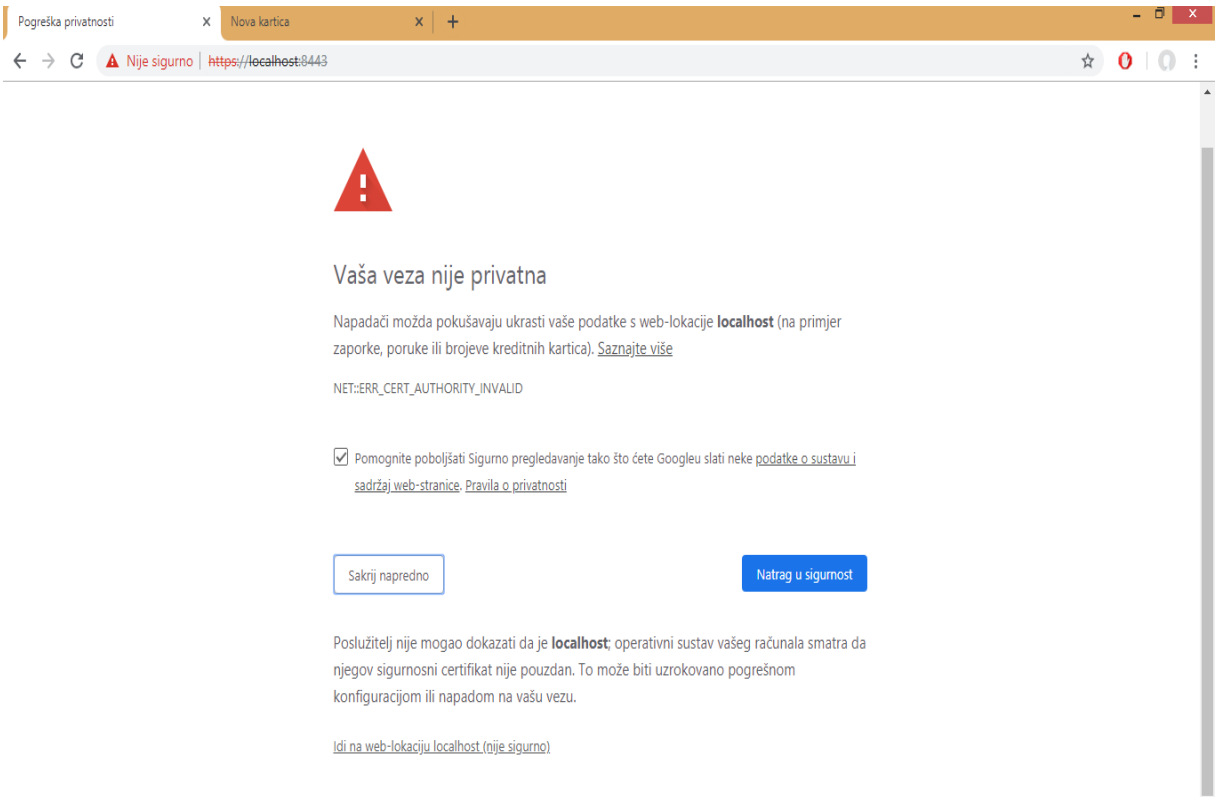
Slika 11: Korišteni security-realm

Na slici iznad se vidi koji security-realm se koristi za sigurnu konekciju.

4.3. TLS na djelu

S certifikatom tj. ključevima implementiranim u server, vrijedi vidjeti kako to izgleda u praksi. Sljedećih nekoliko stranica će opisati gdje se što nalazi te kako što funkcionira na pregledniku, odnosno sa strane klijenta. Osnovna ideja ovoga poglavlja je prikaz sigurne konekcije između preglednika i servera.

Da bi se pristupilo aplikaciji, potrebno je s klijentske strane u URL traku upisati adresu aplikacije, kroz port koji pruža sigurnu konekciju. U ovome praktičnome primjeru je port osigurane konekcije 8443, a cjelokupna adresa aplikacije na serveru je; https://localhost:8443/zavrsniRad/. Prilikom prvog upisa adrese u pregledniku, javlja se upozorenje da pokušaj dohvaćanja željenog web mjesta nudi vezu koja nije privatna. Razlog upozorenju je taj što preglednik ne prepoznaje sigurnosni certifikat implementiran na strani servera. Unatoč upozorenju, potrebno je pritisnuti gumb „idi na web-lokaciju“.

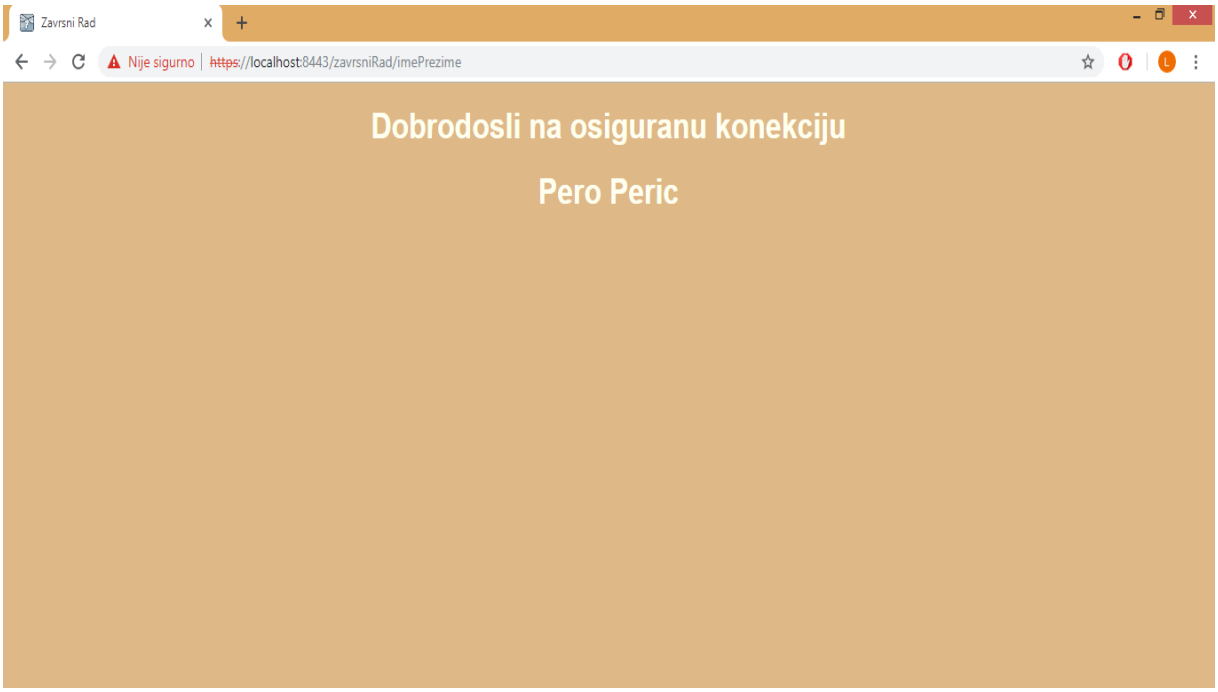


Slika 12: Upozorenje

Nakon toga se javlja početna stranica aplikacije, na kojoj se nalaze polja unosa za ime i prezime korisnika, te gumb unos.

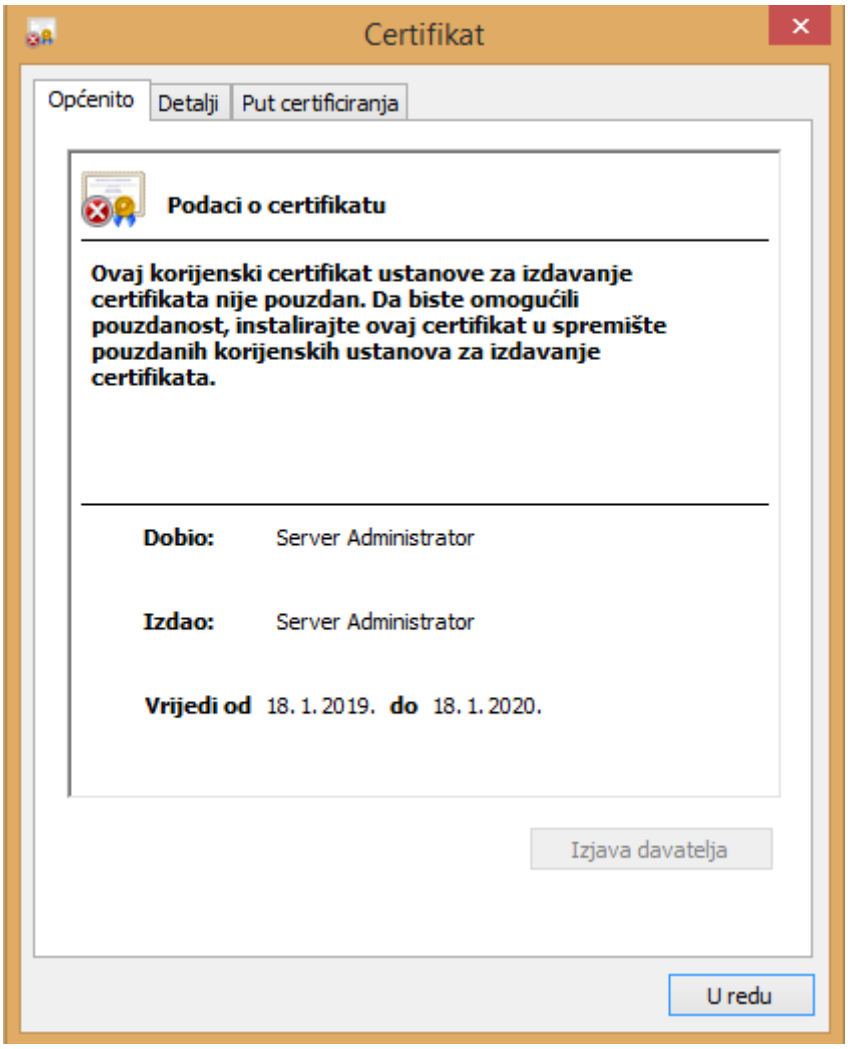
Slika 13: Početna stranica

Unosom argumenata u tekstualna polja, podaci se ispisuju na sljedećoj stranici. Kod forme za upis se koristi POST metoda, te se upisani podaci tako ne vide unutar URL trake kod sljedeće stranice.



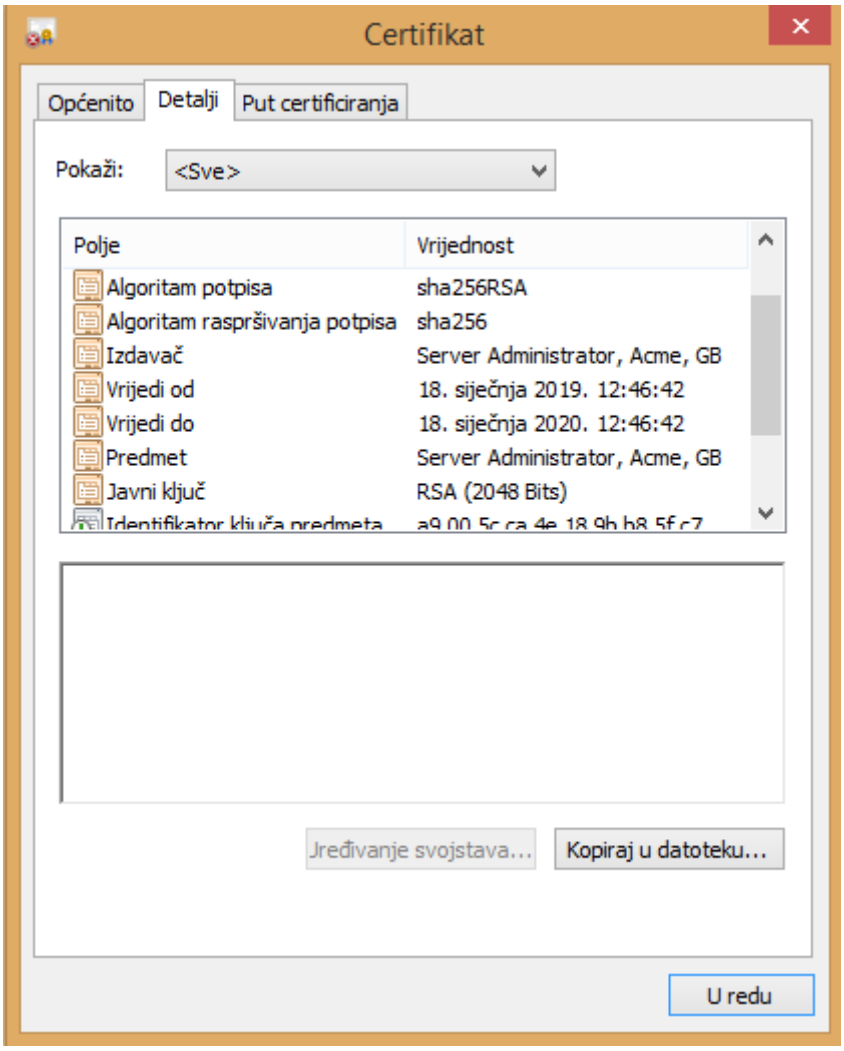
Slika 14: Rezultat stranica

Detalji o primjenjenom certifikatu se mogu vidjeti na dolje navedenoj slici, te se jasno vidi da je certifikat izdan od Server Administratora na datum 18. siječnja 2019. godine te da vrijedi do 18. siječnja 2020. godine.



Slika 15: Certifikat

Za više informacija o certifikatu, prikazani su podaci kartice, detalji. Jasno su vidljive informacije o algoritmu potpisa, izdavaču certifikata, trajanju certifikata te javnome ključu.



Slika 16: Detalji o certifikatu

5. Zaključak

Sigurnost je bitan aspekt razmjene poruka preko interneta. Neovisno o važnosti slanih podataka, važno je imati notu sigurnosti na svim web mjestima jer nam time osim očitih sigurnosnih razloga, stranica ulijeva povjerenje. Tu nastupa osiguranje komunikacije, koja se uspješno omogućava pomoću raznih SSL i TLS protokola više od dvadeset godina.

Protokoli za implementaciju sigurnosti nam predstavljaju osiguranje integriteta, autentičnosti, neporecivosti te tajnosti.

Čar osiguranog web mjesta je ta što je korisniku na jednostavan i ugodan način omogućena sigurna razmjena podataka bez straha da će subjektu netko ukrasti prijeko potrebne informacije tijekom korištenja web mjesta. Sve se to odvija daleko od očiju korisnika, te zapravo ne obraća potrebna pozornost i ne daje dovoljna zahvala implementiranoj sigurnošću.

Unatoč bezbrižnosti koju pružaju razni sigurnosni protokoli, važno je konstantnim naporima raditi na unaprjeđenju postojećih sustava sigurnosti kako bi svakodnevni korisnici bili u mogućnosti biti sigurni bez straha od stalne prijetnje hakera.

Popis literature

[1] https://web.math.pmf.unizg.hr/~duje/kript/osnovni.html, siječanj 2019.

[2] https://www.ibm.com/developerworks/library/ws-ssl-security/index.html, siječanj 2019.

[3 https://datatracker.ietf.org/doc/rfc6176/, siječanj 2019.

[4] Rolf Opplinger, SSL and TLS: Theory and Practice, 2009.

[5] Stephen A. Thomas, SSL and TLS Essentials, 2000.

Popis slika