Primjena Verisign digitalnih certifikata u identifikaciji web mjesta

Vinko, Cerovečki

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Vinko Cerovečki

PRIMJENA VERISIGN DIGITALNIH CERTIFIKATA U IDENTIFIKACIJI WEB MJESTA

DIPLOMSKI RAD

Varaždin, 2020.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Vinko Cerovečki

Matični broj: 44087/15–R

Studij: Informacijsko i programsko inženjerstvo

PRIMJENA VERISIGN DIGITALNIH CERTIFIKATA U IDENTIFIKACIJI WEB MJESTA

DIPLOMSKI RAD

Mentor/Mentorica:

izv.prof.dr.sc. Sandro Gerić

Varaždin, kolovoz 2020.

Vinko Cerovečki

Izjava o izvornosti

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Kao što se može zaključiti iz naslova ovog diplomskog rada, njegova središnja tema je implementacija identifikacije web mjesta certifikatom. Kroz teoretski dio rada, najprije su opisani protokoli koji se danas koriste za omogućavanje sigurne komunikacije više sudionika putem nesigurne mreže. Nadalje, ukratko je opisana kriptografija, njezina povijest, te najvažniji kriptografski algoritmi današnjice. Također, napravljena je usporedba prednosti i nedostataka simetričnog i asimetričnog kriptiranja, te su opisani pojmovi vezani uz infrastrukturu javnog ključa, digitalne certifikate i hijerarhijski model povjerenja. U praktičnom primjeru implementiran je sustav koji se sastoji od baze podataka i tri različite aplikacije. Implementiran je mehanizam identifikacije poslužitelja sigurnosnim certifikatom, te je u teoretskom dijelu ukratko opisan postupak implementacije.

Ključne riječi: SSL/TLS; identifikacija; kriptografija; PKI; certifikati; javni ključ; privatni ključ; HTTPS;

Sadržaj

Sadržaj iii
1. Uvod 1
2. Metode i tehnike rada 2
3. Secure Socket Layer/Transport Layer Security 3
3.1. Princip rada SSL/TLS protokola 6
4. Osnove kriptografije 11
4.1. Što je to kriptografija? 11
4.2. Simetrično kriptiranje 11
4.3. Asimetrično kriptiranje 14
4.4. Usporedba prednosti i nedostataka simetričnog i asimetričnog kriptiranja 17
5. PKI infrastruktura 20
5.1. Struktura PKI 20
5.2. Slabosti PKI infrastrukture 22
5.3. Digitalni certifikati 25
5.4. Hijerarhijski model povjerenja 28
5.5. VeriSign 28
5.6. PKI u Hrvatskoj 29
6. Praktični primjer 31
6.1. Korištene tehnologije 31
6.2. Prikaz korištenja aplikacije 36
7. Zaključak 40
Popis literature 41
Popis slika 42
Popis tablica 43

1. Uvod

Živimo na svijetu na kojem su ljudi iz dana u dan sve više povezani internetom. Od kraja 20. stoljeća popularnost interneta je porasla dramatično i promijenila način života ljudima na cijelom svijetu. Ljudi danas korištenjem mobilnih uređaja i računala komuniciraju, kupuju, plaćaju račune, planiraju putovanja.

Na početku razvoja interneta, malo pažnje je bilo posvećivano sigurnosti. U tim danima, nije ni postojala pretjerana potreba za time, s obzirom na to da je malo subjekata bilo povezano internetom i to su većinom bila sveučilišta. Međutim danas, kada su svi ili gotovo svi povezani Internetom stvari se mijenjaju. Postoji mnogo onih koji bi dobre strane interneta mogli iskoristiti u loše svrhe, pa se javlja potreba za povećavanjem razine sigurnosti na internetu.

U ovom diplomskom radu objašnjeni su protokoli koji služe za sprječavanje prisluškivanja, izmjene i lažiranja poruka. Također, objašnjene su osnove kriptografije, te dvije vrste kriptiranja uz usporedbu prednosti i nedostataka svake od tih dviju vrsta kriptiranja. Objašnjena je i infrastruktura javnog ključa, prednosti i nedostaci te infrastrukture, te digitalni certifikati. Ovaj rad ukratko opisuje VeriSign digitalne certifikate, ali i ulogu FINE u Republici Hrvatskoj kao pružatelja usluga certifikacije. U sklopu diplomskog rada napravljen je i praktični primjer.

2. Metode i tehnike rada

U izradi ovog diplomskog rada korišteno je nekoliko alata. Za uređivanje tekstualnog dokumenta korišten je Microsoft Office Word, a u izradi praktičnog primjera koji pokazuje primjenu digitalnih certifikata korišteno je više tehnologija. U izradi back-end aplikacije korišten je Spring Boot razvojni okvir. Front-end aplikacija izrađena je uz pomoć React biblioteke, a za bazu podataka korišten je MySQL. Aplikacija se vrti iza Nginx proxy poslužitelja, na kojem je i konfigurirana identifikacija.

3. Secure Socket Layer/Transport Layer Security

U ovom poglavlju objašnjen je SSL (eng. Secure Socket Layer), odnosno TLS (eng. Transport Layer Security) protokol. Ukratko je opisana povijest verzija SSL-a od prve verzije do TLS-a, te je objašnjena važnost tog protokola.

Da bi se lakše shvatio ostatak ovog, ali i poglavlja koja slijede, potrebno je najprije razumjeti što je to uopće Internet. Internet je svjetska računalna mreža koja je nastala šezdesetih godina 20. stoljeća u Sjedinjenim Američkim Državama. U početku su u toj mreži bila uključena samo pojedina sveučilišta, ali s vremenom su se počela uključivati i razna poduzeća, a na kraju i privatni korisnici. Internet danas nudi nezamislive mogućnosti korisnicima, od raznih edukacijskih sadržaja preko usluga kupnje i prodaje do zabavnih sadržaja. [2]

Kao što sam već napisao u uvodu ovog rada, svjesni smo da je od kraja 20. stoljeća do danas popularnost i korištenje interneta dramatično porasla. Uz sve dobre strane koje na internet nudi, postoje i njegove slabosti. Naime, osim ljudi koji internet koriste u pozitivne svrhe kao što je olakšavanje i ubrzavanje obavljanja različitih poslova, postoje i oni koji slabosti interneta iskorištavaju u negativne svrhe. Zbog toga se svi uređaji koji su povezani na Internet oslanjaju na SSL i TLS protokole.

Na početku stvaranja Interneta jako malo pažnje posvećivalo se sigurnosti. Točnije, sigurnost Interneta bila je na dnu liste prioriteta. U to vrijeme stvari su tako možda i mogle funkcionirati. Međutim, danas, kad su gotovo svi ljudi povezani Internetom, takav pristup više nije moguć. SSL i TLS protokoli su kriptografski protokoli koji su osmišljeni sa svrhom osiguravanja sigurne komunikacije putem nesigurne infrastrukture. U prijevodu, ti protokoli omogućuju da svi korisnik nekog servisa na Internetu može biti siguran da komunicira s poslužiteljem koji mu je predstavljen, te da podaci koje razmjenjuje s poslužiteljem neće biti promijenjeni i neće završiti u tuđim rukama.

Osim omogućavanja sigurne komunikacije putem Interneta, SSL/TLS protokol služi i u svrhe interoperabilnosti (mogućnost programera da razvijaju programe i biblioteke koje međusobno mogu komunicirati korištenjem zajedničkih kriptografskih parametara), proširivosti (neovisnost o kriptografskim značajkama, mogućnost migracija bez potrebe za stvaranjem novih protokola) i učinkovitosti (ostvarenje prethodno navedenih ciljeva uz prihvatljiv „trošak“). [1, str. 1]

SSL protokol razvijen je u tvrtki Netscape (Netscape Communications Corporation), te je dostupan u verziji 2.0 od ožujka 1995. Verzija 1.0 SSL protokola nikada nije bila javno

dostupna iz nekih razloga. Međutim, verzija 2.0 SSL protokola je sadržavala mnogo sigurnosnih propusta i ranjivosti, pa je do 1996. kao odgovor na te ranjivosti razvijena i objavljena verzija 3.0 koja je ujedno i temelj kasnije razvijenog TLS protokola. Iz tog razloga se SSL i TLS protokoli najčešće navode jedan uz drugog. [1, str. 3]

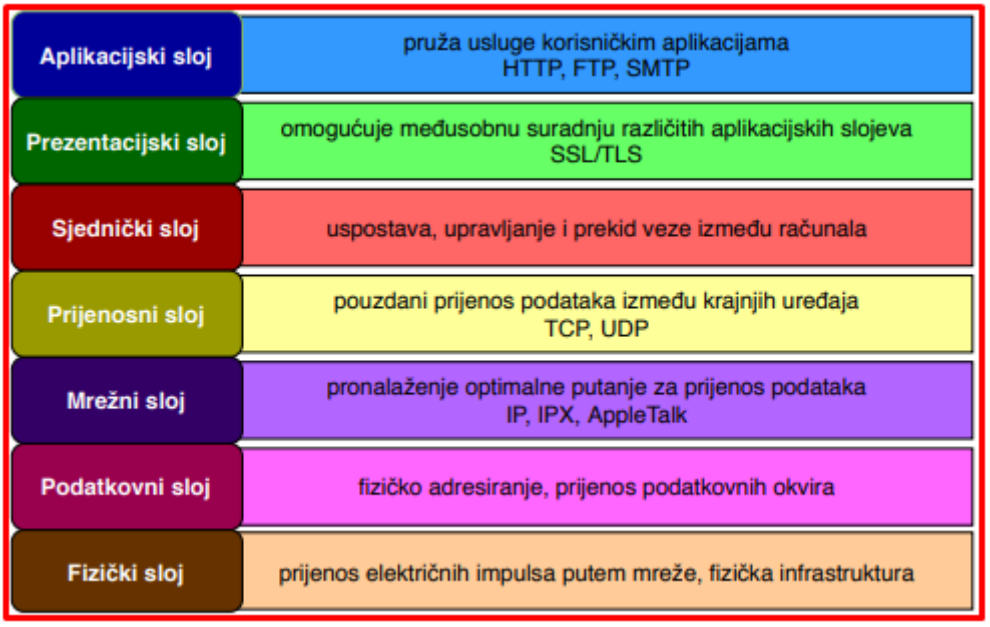
Za kvalitetno razumijevanje TLS protokola i njegove svrhe, važno je razumjeti OSI (eng. Open Systems Interconnection) model. OSI model je model podijeljen u sedam slojeva koji stručnjaci koriste u razvoju računalnih mreža i protokola. Slojevi OSI modela služe za prijenos podataka od izvorišta do odredišta pri čemu su donji slojevi redom sve bliži fizičkim komunikacijskim vodovima, a gornji su redom na višoj razini apstrakcije od nižeg sloja. Slojevi od kojih se sastoji OSI model su sljedeći (redom od najvišeg do najnižeg, odnosno od najviše razine apstrakcije prema najnižoj):

- Aplikacijski sloj – 7. sloj OSI modela. Aplikacijski sloj, za razliku od ostalih slojeva OSI modela, ne pruža usluge ni jednom drugom sloju OSI modela, već korisničkim aplikacijama koje nisu dio OSI modela. Aplikacijski sloj zapravo osigurava aplikacijskim servisima pristup mreži, te olakšava korisničkim aplikacijama prikaz dohvaćenih podataka korisniku. Primjeri protokola na ovom sloju su HTTP (eng. Hypertext Transfer Protocol), FTP (eng. File Transfer Protocol), SMTP (eng. Simple Mail Transfer Protocol) i sl.
- Prezentacijski sloj – 6. sloj OSI modela. Prezentacijski sloj omogućuje aplikacijskim slojevima različitih sustava da međusobno surađuju i to radi na način da prevodi aplikacijski format podataka u mrežni. Naime, prezentacijski sloj enkapsulira podatkovne jedinice u SPDU (eng. Session Protocol Data Units) blokove, te takve blokove šalje nižim slojevima. Na taj način podaci mogu putovati mrežom bez problema s kompatibilnošću. Također, prezentacijski sloj i kriptira podatke u nizove bitova. Na ovom sloju nalazi se SSL/TLS protokol.
- Sjednički sloj – 5. sloj OSI modela. Sjednički sloj služi za uspostavu, upravljanje i prekid veze između računala. Osim toga, sjednički sloj omogućuje dvosmjerne (eng. full-duplex) i jednosmjerne (eng. half duplex) operacije, te postavljanje kontrolnih točaka (eng. checkpoint). Kontrolne točke služe za ispravnu ponovnu sinkronizaciju podataka u slučaju da dođe do greške, te omogućavaju da se izbjegne nepovratni gubitak podataka.
- Prijenosni sloj – 4. sloj OSI modela. Prijenosni sloj zadužen je za pouzdani prijenos podataka između krajnjih uređaja. U tom sloju postoje protokoli koji imaju mogućnost otkrivanja i ispravljanja eventualnih pogrešaka u prijenosu

podataka. Najznačajniji protokoli prijenosnog sloja su TCP (eng. Transmission Control Protocol) i UDP (eng. User Datagram Protocol).

- Mrežni sloj – 3. sloj OSI modela. Mrežni sloj pruža uslugu povezivanja, odnosno odabira optimalne putanje za prijenos podataka. U tom poslu koristi logičko adresiranje putem IP (eng. Internet Protocol) adresa. Zadaća ovog sloja je prijenos podataka između dviju krajnjih točaka, ali ne i briga oko ispravljanja grešaka. Taj posao, kao što je već rečeno, obavlja prijenosni sloj. Na mrežnom sloju najznačajniji je IP protokol, a od ostalih postoje još IPX (eng. Internetwork Packet Exchange) protokol, AppleTalk i ostali.
- Podatkovni sloj – 2. sloj OSI modela. Podatkovni sloj služi za pouzdani prijenos podataka putem fizičkog medija. U sklopu te zadaće podatkovni sloj obavlja posao fizičkog adresiranja, prijenosa podatkovnih okvira, kontrole protoka i sl.
- Fizički sloj – 1. sloj OSI modela. Fizički sloj služi za prijenos bita, odnosno električnog impulsa putem mreže. U sklopu te zadaće fizički sloj vodi brigu o fizičkim medijima koji služe za prijenos podataka što obuhvaća vodove i konektore, te upravlja jačinom napona/signala, brzinom prijenosa podataka i sl. [3, str. 8]

Slika 1. prikazuje OSI model koji sam napravio kao vizualnu sistematizaciju onog što je opisano u prethodnom odlomku.



Slika 1. Osi model (vlastita izrada prema opisu iznad)

U sljedećem potpoglavlju opisan je način rada SSL/TLS protokola.

3.1. Princip rada SSL/TLS protokola

Najprije ćemo ugrubo objasniti korake koji se izvode u komunikaciji klijenta i poslužitelja kod SSL/TLS protokola. Osnovni koraci te komunikacije su:

- Rukovanje i dogovor oko šifriranja
- Autentifikacija
- Izmjena informacija o ključu

Tek nakon što se izvrše prethodno nabrojeni koraci započinje sigurna komunikacija, odnosno sigurna razmjena podataka između klijenta i poslužitelja. Slika 2. prikazuje osnovne korake u komunikaciji klijenta i poslužitelja kod SSL/TLS protokola.

Slika 2. Komunikacija klijenta i poslužitelja kod SSL/TLS protokola [3, str. 11]

U prvom koraku obavlja se rukovanje (eng. SSL Handshake) koje će biti detaljnije objašnjeno kasnije, te razmjena informacija o šifriranju, odnosno dogovor oko toga koje će se šifriranje koristiti u daljnjoj komunikaciji. Slijedi autentifikacija poslužitelja i klijenta u sklopu kojeg se razmjenjuju informacije o identitetu. U tom se koraku zapravo obavlja razmjena javnih ključeva. Sljedeći korak je razmjena informacija vezanih uz ključ. U ovom se koraku radi o generiranju pseudoslučajnih vrijednosti na strani klijenta i servera. Nakon generiranja, te

vrijednosti se kombiniraju s dodatnim podacima koji omogućuju poslužitelju i klijentu generiranje zajedničke tajne koja se još zove i *Master Secret*. [3, str. 11]

Da bismo shvatili način rada SSL/TLS protokola, važno je da razumijemo da se SSL protokol sastoji od dva manja protokola, protokola zapisa (eng. Record Protocol) i protokola rukovanja (eng. Handshake Protocol).

Protokol rukovanja je najkompliciraniji dio TLS protokola. U koracima rukovanja klijent i poslužitelj vrše pregovore o parametrima veze, te vrše autentifikaciju. U protokolu rukovanja obično se razmijeni između 6 i 10 poruka između klijenta i poslužitelja što ovisi o značajkama koje se koriste. Ovisno o konfiguraciji i mogućim proširenjima protokola, u razmjeni poruka kod rukovanja može biti mnogo različitih varijacija. No, u praksi postoje 3 uobičajena moguća slijeda:

- Potpuno rukovanje (eng. Full Handshake) s provjerom autentičnosti poslužitelja
- Skraćeno rukovanje (eng. Half Handshake) koje nastavlja raniju sjednicu
- Rukovanje s provjerom autentičnosti klijenta i poslužitelja [1, str. 26]

U slučaju da klijent već ranije nije uspostavio sjednicu s poslužiteljem, prilikom uspostavljanja TLS veze klijent i poslužitelj izvršavaju potpuno rukovanje. U tom postupku klijent i poslužitelj dogovaraju parametre TLS sjednice i to kroz četiri osnovna koraka koji su opisani u nastavku.

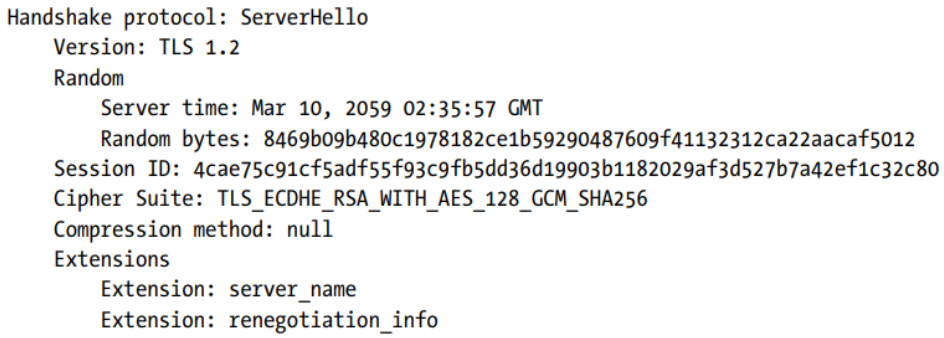
U svakom rukovanju prvi korak napravi klijent. U tom koraku klijent šalje ClientHello poruku serveru. U toj poruci klijent šalje serveru podatke o svojim mogućnostima i postavkama. Tu poruku klijent šalje svaki put kada se prvi puta želi povezati sa poslužiteljem, kada želi ponovno uspostaviti (obnoviti) vezu sa poslužiteljem ili kada odgovara na poslužiteljev zahtjev za obnovom veze. Slika 3. prikazuje osnovne podatke koje klijent šalje poslužitelju u ClientHello poruci. Na slici su prikazani samo ključni podaci koji se šalju. Sadržaj te poruke je jednostavan i razumljiv. Najprije je navedena verzija protokola koju klijent podržava. Uvijek se navodi najbolja (najnovija) verzija koju je klijent u stanju podržati. Zatim slijedi polje koje sadrži 32 bajta. Od ta 32 bajta, 28 bajtova je pseudoslučajna vrijednost, a preostala 4 bajta se odnose na dodatne podatke koji se dobiju uz pomoć sistemskog vremena klijenta. Taj dio od 4 bajta primjenjuje se od 1994. godine kao zakrpa za „slabe“ generatore pseudoslučajnih vrijednosti. U početku se zapisivalo stvarno vrijeme, ali iz sigurnosnih razloga danas neki preglednici dodaju vremenski pomak. To pseudoslučajno polje igra ključnu ulogu u autentifikaciji. Sljedeći dio klijentske ClientHello poruke je identifikacijska oznaka sjednice. Kod uspostavljanja nove veze, to polje ostaje prazno, te označava da klijent ne pokušava obnoviti postojeću sjednicu, nego uspostaviti novu vezu. Kod obnavljanja postojeće veze, to polje sadrži jedinstveni

identifikator sjednice, tako da poslužitelj može prema tom identifikatoru pronaći odgovarajuće stanje sjednice u predmemoriji. Identifikator se obično sastoji od 32 bajta pseudoslučajne vrijednosti. „Cipher suite“ blok sadrži listu podržanih šifriranja i to poredanih po prioritetu od strane klijenta. „Compression methods“ polje sadrži popis podržanih metoda sažimanja, a ako se kao metoda kompresije pošalje *null* vrijednost, to označava da nema kompresije. U polju „Extensions“ obično se navodi proizvoljan broj dodataka koji služe za prijenos nekih dodatnih podataka.

```
Handshake protocol: ClientHello
    Version: TLS 1.2
    Random
        Client time: May 22, 2030 02:43:46 GMT
        Random bytes: b76b0e61829557eb4c611adfd2d36eb232dc1332fe29802e321ee871
    Session ID: (empty)
    Cipher Suites
        Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
        Suite: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
        Suite: TLS_RSA_WITH_AES_128_GCM_SHA256
        Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
        Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA
        Suite: TLS_RSA_WITH_AES_128_CBC_SHA
        Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA
        Suite: TLS_RSA_WITH_RC4_128_SHA
    Compression methods
        Method: null
    Extensions
        Extension: server_name
            Hostname: www.feistyduck.com
        Extension: renegotiation_info
        Extension: elliptic_curves
            Named curve: secp256r1
            Named curve: secp384r1
        Extension: signature_algorithms
            Algorithm: sha1/rsa
            Algorithm: sha256/rsa
            Algorithm: sha1/ecdsa
            Algorithm: sha256/ecdsa
```

Slika 3. Primjer ClientHello poruke s ključnim podacima [1, str. 28]

Na klijentsku ClientHello poruku poslužitelj odgovara sa ServerHello porukom. U toj poruci poslužitelj šalje klijentu odabrane postavke veze. Struktura poruke je identična ClientHello poruci, samo što poslužitelj klijentu šalje samo jednu odabranu postavku po pojedinom polju. Slika 4. prikazuje primjer ServerHello poruke.



Slika 4. Primjer „ServerHello“ poruke [1, str. 30]

Nakon ServerHello poruke, u slučaju da je provjera autentičnosti na strani poslužitelja potrebna, poslužitelj šalje klijentu i certifikat, odnosno lanac certifikata (eng. Certificate Chain). S obzirom na to da provjera autentičnosti poslužitelja može, ali i ne mora biti potrebna, slanje te poruke nije u svakom slučaju obavezno.

Sljedeća poruka je također opcionalna, odnosno ovisi o nekim drugim parametrima, a šalje ju opet poslužitelj. Radi se o „ServerKeyExchange“ poruci, a ta poruka se šalje u ovisnosti o odabranoj razmjeni ključeva. „ServerKeyExchange“ poruka nosi informacije potrebne za kreiranje *Master Secret* vrijednosti o kojoj je već bilo riječi u jednom od prethodnih odlomaka.

Nakon svih prethodnih poruka, od kojih su neke opcionalne, poslužitelj šalje *ServerHelloDone* poruku kojom javlja klijentu da je pregovaranje oko parametara veze na strani poslužitelja završeno.

Nakon zaprimljene *ServerHelloDone* poruke, klijent šalje sljedeću poruku poslužitelju. Radi se o *ClientKeyExchange* poruci u kojoj klijent šalje poslužitelju podatke potrebne za kreiranje *Master Secret* vrijednosti. Ta poruka sadrži javni ključ, posebnu PMS (eng. Pre-Master Secret) vrijednost ili nikakve podatke.

Sljedeća poruka je poruka klijenta i za nju se koristi naziv *ChangeCipherSpec*. To je poruka nakon koje klijent prelazi na kriptiranje komunikacije simetričnim kriptiranjem. Sve dosadašnje razmijenjene poruke kriptiraju se asimetričnim kriptiranjem. Osnovna razlika između simetričnog i asimetričnog kriptiranja je u tome da se poruke kod simetričnog kriptiranja kriptiraju i dekriptiraju identičnim ključevima, dok se kod asimetričnog kriptiranja za kriptiranje koristi jedan ključ, a za dekriptiranje drugi. No, o simetričnom i asimetričnom kriptiranju će biti više u sljedećem poglavlju.

Nakon poslane *ChangeCipherSpec* poruke, klijent šalje *Finished* poruku. U toj poruci klijent šalje poslužitelju MAC (eng. Message Authentication Code) vrijednosti prethodno

primljenih i poslanih poruka tijekom rukovanja. MAC vrijednost dobiva se pomoću MAC kriptografske funkcije. MAC kriptografska funkcija samom sažimanju dodaje mogućnost provjere autentičnosti. MAC vrijednost se uvijek šalje odvojeno od originalne vrijednosti, a za to postoji i dobar razlog. Naime, kada bi se originalna i MAC vrijednosti slale zajedno, napadači bi obje vrijednosti mogli bez problema modificirati, bez da to bude primijećeno. Slanje MAC vrijednosti omogućuje primatelju da provjeri je li odgovarajuća originalna poruka bila mijenjana putem od pošiljatelja ili nije. Na taj način osigurava se integritet podataka.

U prethodnim odlomcima opisan je protokol rukovanja s provjerom autentičnosti poslužitelja, odnosno razmjena poruka u tom protokolu. Ako se pak radi o rukovanju s provjerom autentičnosti klijenta i poslužitelja, onda se razmjenjuju još neke dodatne poruke. Naime, prije *ServerHelloDone* poruke poslužitelj šalje *CertificateRequest* poruku u kojoj poslužitelj od klijenta zahtijeva da se autentificira. U toj poruci poslužitelj klijentu šalje i prihvatljive javne ključeve certifikata i algoritme potpisa. Također, opcionalno u toj poruci poslužitelj klijentu može poslati i listu imena prihvatljivih tijela za izdavanje certifikata (eng. Certification Authority). Nadalje, kod klijenta se prije *ClientKeyExchange* poruke obavlja slanje *Certificate* poruke. U toj poruci klijent poslužitelju šalje certifikat u identičnom formatu kao što to prethodno radi i poslužitelj, a to je već opisano u prethodnim odlomcima. Također, nakon *ClientKeyExchange* poruke klijent šalje *CertificateVerify* poruku. Tom porukom klijent dokazuje da posjeduje privatni ključ koji odgovara javnom ključu kojeg je prethodno poslao u *Certificate* poruci. *CertificateVerify* poruka sadrži potpis svih prethodno razmijenjenih poruka u rukovanju.

Ostalo je još za objasniti protokol skraćenog rukovanja kojim se nastavlja, odnosno obnavlja ranija sjednica. U tom protokolu se razmjenjuju sve već ranije opisane poruke, pa ću ih ovdje samo navesti. S klijentske strane to su *ClientHello, ChangeCipherSpec* i *Finished* poruka, dok poslužitelj šalje *ServerHello*, *ChangeCipherSpec* i *Finished* poruke. Cijeli mehanizam za obnovu sjednice temelji se na sigurnosnim parametrima koje klijent i poslužitelj čuvaju neko vrijeme nakon što ispregovarana veza završi. Klijent koji želi obnoviti sjednicu šalje poslužitelju *ClientHello* poruku u kojoj prosljeđuje identifikacijski broj sjednice koju želi obnoviti. Ako poslužitelj želi obnoviti sjednicu, vraća klijentu isti identifikacijski broj koji šalje u *ServerHello* poruci, te generira novi set ključeva uz pomoć *Master Secret* vrijednosti, odabire vrstu kriptiranja i šalje te podatke u *Finished* poruci. [1, str. 34]

4. Osnove kriptografije

U ovom poglavlju kroz potpoglavlja je objašnjeno što je to kriptografija, koja je njezina uloga, te njezin razvoj kroz povijest. Također, objašnjene su dvije osnovne vrste kriptiranja (simetrično kriptiranje i asimetrično kriptiranje), te koje su prednosti, a koji nedostaci svake od tih dviju vrsta kriptiranja.

4.1. Što je to kriptografija?

Postoji više različitih definicija kriptografije, a jedna od njih je i ona koja kriptografiju definira kao znanost, odnosno umjetnost sigurne komunikacije. Koliko god se kriptografija doživljava kao nešto novo, odnosno nešto što je nastalo u novije vrijeme, stvari su zapravo drugačije. Ljudi kriptografiju koriste već tisućama godina, samo u različitim oblicima. Prvi poznati oblik kriptografije datira još u 7. stoljeću prije Krista, a radi se o *scytale,* alatu koji se koristio za kreiranje transpozicijske šifre. *Scytale* se sastojao od cilindra s trakom pergamenta koji je bio namotan oko cilindra, te na kojem je bila napisana poruka. Takav oblik šifriranja navodno su koristili Grci u vojnim pohodima.[4]

Kriptografiju kao znanost danas dijelimo na kriptoanalizu i kriptografiju. Kriptografija kakvu danas poznajemo i koristimo nastala je u najvećem dijelu u 20. stoljeću, a prvotno je također korištena samo u vojne svrhe. Danas je ta kriptografija svakodnevno prisutna u životima prosječnih ljudi. Osnovne svrhe kriptografije danas su očuvanje tajnosti informacija, potvrda identiteta, te omogućavanje sigurnog transporta podataka u smislu očuvanja integriteta podataka.

Kriptoanaliza odnosni se na suprotni posao od kriptografije, odnosno na dešifriranje kriptiranog sadržaja u izvorni sadržaj. Drugim riječima, kriptoanaliza daje rješenje za slučajeve kada nemamo ključ, šifru ili jedno i drugo. [5, str. 4]

4.2. Simetrično kriptiranje

Simetrični kriptografski algoritmi poznati su po tome što koriste jedinstveni ključ za kriptiranje i dekriptiranje sadržaja. Općenito, simetrični algoritmi kriptiranja temelje se na XOR (eng. Exclusive or) operaciji čija je tablica prikazana na Slici 5., ali i nekim dodatnim složenijim funkcijama koje povećavaju njihovu pouzdanost.

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

Tablica 1. XOR operator

Kroz sljedeći primjer vidjet ćemo kako funkcionira kriptiranje, odnosno dekriptiranje sadržaja simetričnim algoritmima kriptiranja, te koja je uloga XOR operatora u tim operacijama. Zbog jednostavnosti primjera, kriptirat ćemo i dekriptirati tekst „FOI“. Ako tu riječ pretvorimo u binarni zapis, dobivamo sljedeći niz nula i jedinica:

FOI = 01000110 01001111 01001001

Za šifru ćemo koristiti binarni zapis riječi „key“ koji izgleda ovako: 01101011 01100101 01111001. Tablica 2. prikazuje rezultat XOR operacije nad te dvije vrijednosti. Prvi redak prikazuje ključ, drugi redak tekst koji kriptiramo, a treći redak prikazuje kriptirani tekst.

X	0	1	1	0	1	0	1	1	0	1	1	0	0	1	0	1	0	1	1	1	1	0	0	1
Y	0	1	0	0	0	1	1	0	0	1	0	0	1	1	1	1	0	1	0	0	1	0	0	1
X XOR Y	0	0	1	0	1	1	0	1	0	0	1	0	1	0	1	0	0	0	1	1	0	0	0	0

Tablica 2. Prikaz XOR operacije

Dekriptiranje je isti postupak, samo što u tom slučaju XOR operaciju radimo nad ključem i kriptiranim sadržajem, a rezultat je dekriptirani tekst.

Sustav simetričnog kriptiranja sastoji se od sljedeća tri osnovna elementa:

- Sadržaj za kriptiranje (eng. *plaintext, M)*
- Kriptirani sadržaj (eng. *ciphertext, C)*
- Ključ za kriptiranje, odnosno dekriptiranje (eng. *key, K)*

Simetrična kriptografija se vrlo lako može prikazati i matematički i to sljedećim izrazima:

- Kriptiranje: $C = E_k(M)$
- Dekriptiranje: $M = D_k(C)$

Jedna od osnovnih pozitivnih karakteristika simetričnih algoritama je njihova brzina kriptiranja zbog čega su pogodni za kriptiranje velikih količina podataka. Međutim, problem simetričnih algoritama kriptiranja je razmjena ključeva za kriptiranje. Naime, kao što je već rečeno, simetrični algoritmi za kriptiranje i dekriptiranje koriste isti ključ, pa je npr. kod razmjene poruka potrebno osigurati da taj ključ imaju i pošiljatelj i primatelj. Danas najpoznatiji korišteni simetrični algoritmi su DES, 3DES, BLOWFISH, RC2, RC5, IDEA i ostali. Simetričnim algoritmima se kriptiraju podaci u blokovima. Ti mogu biti veličine od po nekoliko stotina bitova.

Duljine ključeva navedenih simetričnih algoritama iznose od 64 do 1024 bita. Tablica 1. prikazuje popis danas najpoznatijih simetričnih algoritama, te njihove duljine ključeva za kriptiranje i veličine blokova koji se kriptiraju. [6, str. 35]

ALGORITAM	VELIČINA KLJUČA (bit)	VELIČINA BLOKA (bit)
DES	56	64
3DES	112/168	64
IDEA	128	64
BLOWFISH	448	64
RC5	832	64
RC2	1024	64

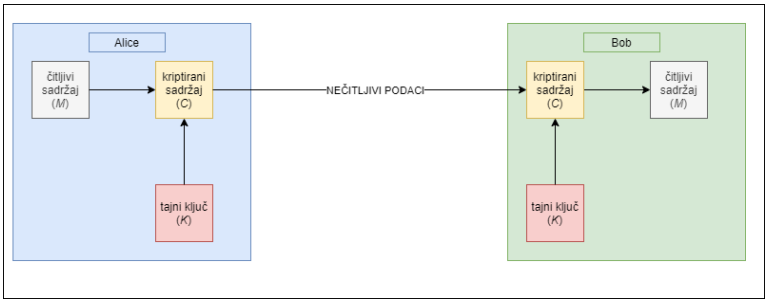
Tablica 3. Neki od simetričnih algoritmi

Koliko su određeni simetrični algoritmi sigurni uvelike ovisi o duljini ključa kojeg koristi u kriptiranju. Naime, ukoliko ne postoji način da se do ključa dođe preko informacija koje se prenose, jedini način koji preostaje napadačima je pronalazak ključa. Vrijeme potrebno da se dođe do ključa eksponencijalno raste sa samom duljinom ključa. Ako bacimo opet pogled na prethodnu tablicu, lako možemo zaključiti da DES algoritam baš nije siguran jer ima ključ od svega 56 bita, a potvrđeno je i nekoliko uspješno izvedenih napada kriptoanalizom.

Danas se simetrični algoritmi kriptiranja uvelike koriste, ali tek nakon što primatelj i pošiljatelj prethodno razmijene informacije o algoritmu kojeg će koristiti u komunikaciji, te ključu. Ta razmjena ključeva i informacije o kriptografskom algoritmu koji će se koristiti najčešće se izvršava upotrebom asimetričnih algoritama. Jedan takav način opisan je u prethodnom poglavlju. Nakon što sudionici dogovore koji će algoritam koristiti, te razmijene ključeve, komunikacija se odvija na sljedeći način:

- Alice prije slanja poruke sadržaj kriptira korištenjem tajnog ključa *K*
- Kriptirana poruka se šalje komunikacijskim kanalom do Boba
- Bob nakon što je primio poruku dekriptira sadržaj korištenjem tajnog ključa *K* (istog kakvim je pošiljatelj kriptirao sadržaj)

Slika 5. prikazuje proces komunikacije u kojoj se koristi simetrično kriptiranje.



Slika 5. Simetrična kriptografija

4.3. Asimetrično kriptiranje

Slično kao što to vrijedi za simetrični sustav kriptiranja, asimetrični sustav kriptiranja također služi za kriptiranje, odnosno dekriptiranje podataka. Međutim, postoji jedna velika razlika između te dvije vrste kriptiranja. Naime, dok se kod simetričnog kriptiranja koristi tajni ključ, asimetrični sustav kriptiranja podrazumijeva korištenje javnog ključa i pripadajućih tehnika kriptiranja. Upravo iz tog razloga asimetrična kriptografija često se još naziva i kriptografijom javnog ključa (eng. *public-key cryptography)*. Kod asimetričnog kriptiranja podrazumijeva se da bilo tko može na siguran način javno ustupiti svoj javni ključ. Taj posao vezan je uz PKI (eng. *Public Key Infrastructure)* što će detaljnije biti objašnjeno u narednim poglavljima. Osim sigurnog dijeljenja javnog ključa, asimetrično kriptiranje podrazumijeva da bilo tko može korištenjem javnog ključa neke osobe kriptirati poruku za koju je sigurno da ju može dekriptirati isključivo osoba kojoj je namijenjena.

Osnovni elementi kod asimetričnog sustava kriptiranja su privatni i javni ključ. Pritom javni ključ predstavlja ključ kojim se sadržaj kriptira, dok privatni ključ predstavlja ključ kojim se kriptirani sadržaj dekriptira. Parovi tajnih i javnih ključeva su takvi da se sadržaj kriptiran nekim javnim ključem može dekriptirati isključivo pripadajućim privatnim ključem i ni jednim drugim. Pošiljatelj da bi poslao primatelju poruku mora prvo znati primateljev javni ključ. Privatni ključ poznat je samo primatelju. Pošiljatelj uz pomoć javnog ključa kriptira sadržaj koji šalje, te tako kriptirani sadržaj šalje primatelju. Primatelj zatim dobiveni kriptirani sadržaj dekriptira svojim privatnim ključem. Svaki asimetrični algoritam sastoji se od 3 osnovna manja algoritma:

- Algoritam za generiranje pseudoslučajne vrijednosti

- Algoritam za kriptiranje sadržaja
- Algoritam za dekriptiranje sadržaja [6, str. 46]

Danas najkorišteniji asimetrični algoritmi temelje se na problemu faktorizacije velikih brojeva dobivenih množenjem dva prosta broja. Matematičke funkcije koje se koriste u asimetričnom kriptiranju imaju svojstvo da im se inverz može lako izračunati ako se posjeduje neki posebni parametar. U protivnom, izračun inverza je vrlo težak. Radi se o tzv. jednosmjernim funkcijama s tajnim vratima (eng. *trapdoor*). [6, str. 46]

Tablica 4. prikazuje neke od poznatijih asimetričnih algoritama kriptiranja, te veličine ključeva tih algoritama.

ALGORITAM	VELIČINA KLJUČA (bit)
Diffie-Hellman	1024
DSA	1024
ElGamal	1024
RSA	1024

Tablica 4. Asimetrični algoritmi

Definitivno jedan od najpoznatijih asimetričnih algoritama današnjice je RSA algoritam. RSA algoritam dobio je naziv prema inicijalima znanstvenika koji su sudjelovali u njegovom stvaranju, a to su bili Ron Rivest, Adi Shamir i Leonard Adleman. Objavili su ga 1978. godine. RSA kriptosustav značajan je jer predstavlja sustav kriptiranja, ali i sustav digitalnog potpisa. Drugim riječima, omogućuje da se isti skup algoritama može koristiti za kriptiranje, odnosno dekriptiranje poruka, ali i za digitalno potpisivanje, te provjeru digitalnog potpisa. Funkcija koja se pruža ovisi o kriptografskom ključu koji se koristi:

- Kod korištenja javnog ključa primatelja za kriptiranje sadržaja poruke RSA kriptosustav pruža asimetrični sustav kriptiranja. Privatni ključ primatelja koristi se za dekriptiranje sadržaja, a u idealnom slučaju to može učiniti samo primatelj poruke.
- Kod korištenja privatnog ključa pošiljatelja za kriptiranje sadržaja poruke ili njegovog sažetka RSA kriptosustav pruža sustav digitalnog potpisa. Javni ključ pošiljatelja pritom se koristi za provjeru digitalnog potpisa, te to može učiniti svatko. [6, str.47]

Koraci RSA algoritma kao sustava kriptiranja su sljedeći

1. Odabir dva velika prosta broja p i q

2. Izračun $n = p \cdot q$ i $z = (p-1) \cdot (q-1)$
3. Odabir *d* pri čemu je *d* relativno prost broj s obzirom na *z*.
4. Pronalazak broja *e* takvog da je $e \cdot d = 1 \, mod \, z$
5. Dijeljenje sadržaja za kriptiranje u blokove duljine P na način da svaki od njih bude u intervalu $0 \leq P < n$

Za kriptiranje sada vrijedi formula

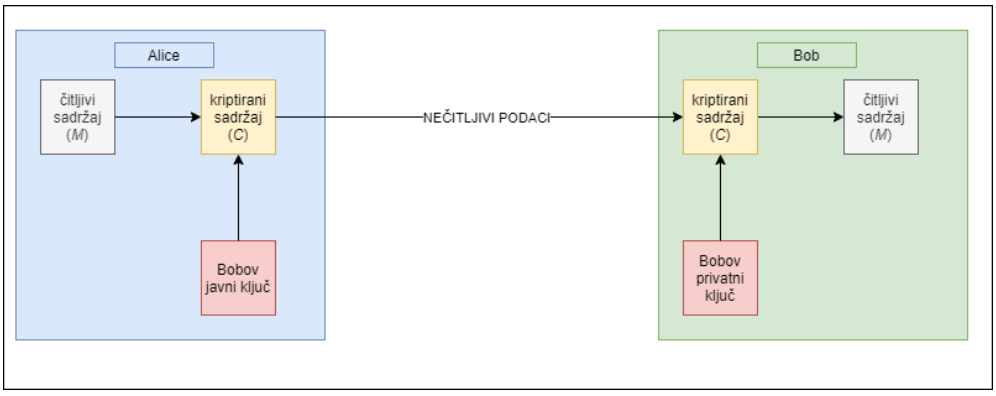
$$C = M^e \, mod \, n,$$

dok za dekriptiranje vrijedi formula

$$M = C^d \, mod \, n$$

pri čemu je *C* kriptirani sadržaj, a *M* sadržaj prije kriptiranja, odnosno poslije dekriptiranja.

Pogledajmo sada još kako bi izgledala komunikacija Alice i Boba kada bi koristili asimetrični kriptosustav. Prije svega, da bi mogla poslati kriptiranu poruku Bobu, Alice mora znati njegov javni ključ. Alice kriptira sadržaj poruke koju šalje Bobovim javnim ključem. Rekli smo već da je kod asimetričnog sustava sadržaj kriptiran javnim ključem moguće dekriptirati isključivo pripadajućim privatnim ključem. Drugim riječima, sadržaj koji je Alice kriptirala Bobovim javnim ključem, moguće je dekriptirati isključivo Bobovim privatnim ključem kojeg u idealnom slučaju zna samo Bob. Slika 6. prikazuje razmjenu poruka štićenu asimetričnim kriptiranjem.



Slika 6. Asimetrična kriptografija

4.4. Usporedba prednosti i nedostataka simetričnog i asimetričnog kriptiranja

U prethodna dva potpoglavlja opisana su neka obilježja simetričnog, odnosno asimetričnog kriptiranja. U ovom poglavlju opisat ću osnovne prednosti i nedostatke korištenja svakog od te dvije vrste kriptiranja, te navesti u kojim se situacijama koja vrsta kriptiranja koristi kako bi se izvuklo najbolje iz jedne i druge vrste kriptiranja.

Krenimo od prednosti simetričnog sustava kriptiranja. Kao što je već bilo spomenuto u prethodnim poglavljima, jedna od karakteristika simetričnog kriptiranja je brzina kriptiranja, te je to i najveća prednost tog sustava. Simetrični sustav kriptiranja omogućava da kriptiranje velikih količina podataka bude brzo i kvalitetno. Nadalje, kriptiranje tajnim ključem zahtijeva daleko manje računalnih resursa u odnosu na asimetrično kriptiranje. Sljedeća jako važna prednost simetričnog sustava kriptiranja je korištenje različitih ključeva za kriptiranje u komunikaciji s različitim sudionicima. Naime, ukoliko napadač uspije doći do jednog tajnog ključa, kompromitiranim ključem može dekriptirati samo poruke koje se razmjenjuju s jednim jedinim sudionikom. Poruke koje se razmjenjuju sa ostalim sudionicima kriptirane su drugim različitim tajnim ključevima, te je njihova razmjena i dalje sigurna.

Osim nekih od prednosti simetričnog kriptiranja opisanih u prethodnom odlomku, taj sustav kriptiranja ima i svojih nedostataka. Osnovni i najveći nedostatak simetričnog sustava kriptiranja je problem razmjene tajnih ključeva. Naime, sigurna razmjena kriptografskih ključeva prije početka njihovog korištenja u komunikaciji ne može se napraviti putem nesigurnog komunikacijskog kanala, već se za taj posao koriste različite tehnike. Danas je najkorištenija tehnika razmjene tajnih ključeva tehnika asimetričnog kriptiranja, ali o tome će nešto više biti u narednim odlomcima. Jedno od obilježja simetričnog sustava kriptiranja je stvaranje velike količine različitih kriptografskih ključeva, što pretvara jednu od prednosti opisanih u prethodnom odlomku u manu. Naime, kao što smo već spomenuli, za komunikaciju s različitim sudionicima koriste se različiti kriptografski ključevi. Zbog toga dolazi do problema upravljanja svim ključevima i očuvanja njihove sigurnosti. Posljednji nedostatak simetričnog kriptiranja koji ću istaknuti je nemogućnost očuvanja autentičnosti razmijenjenih poruka. S obzirom na to da pošiljatelj i primatelj koriste isti ključ za kriptiranje, nemoguće je utvrditi od kojeg konkretnog sudionika poruka dolazi.

Tablica 5. sadrži popis osnovnih prednosti i nedostataka simetričnog sustava kriptiranja.

PREDNOSTI	NEDOSTACI
Jednostavnost	Kompliciranа razmjena tajnih ključeva
Brzina kriptiranja i dekriptiranja	Velik broj tajnih ključeva
Niska razina korištenja računalnih resursa	Nemogućnost utvrđivanja autentičnosti poruke
Kriptiranje i dekriptiranje vlastitih podataka	Nemogućnost očuvanja integriteta poruke

Tablica 5. Prednosti i nedostaci simetričnog sustava kriptiranja

Kod asimetričnog sustava kriptiranja također postoje brojne prednosti, ali i nedostaci. Osnovna prednost asimetričnog sustava kriptiranja je njegova praktičnost. Asimetričnim kriptiranjem riješen je problem razmjene kriptografskih ključeva. Javni ključevi kojima se poruke kriptiraju su svima dostupne, a privatni ključevi za dekriptiranje poruka dostupni su samo pojedincima koji posjeduju par ključeva. Nadalje, asimetrični sustav kriptiranja omogućava provjeru autentičnosti poruka. Kriptiranje javnim ključem omogućuje korištenje digitalnih potpisa čime je primatelju omogućena provjera je li poruka doista od pošiljatelja koji se tako predstavlja. Digitalni potpis uz provjeru pošiljatelja nudi još jednu prednost, a to je provjera integriteta poruke. Naime, digitalno potpisanu poruku nije moguće izmijeniti, a bez da se pritom ne poništi potpis, pa je tako vrlo lako moguće utvrditi je li primljena poruka originalna ili je putem promijenjena. Još jedna prednost je i nemogućnost poricanja poslane poruke. Naime, digitalno potpisivanje poruke je slično fizičkom potpisivanju dokumenata. Digitalni potpis je potvrda poruke i pošiljalac ne može negirati poslanu poruku.

Kao i kod simetričnog kriptiranja, asimetrični sustav također ima svojih nedostataka. Osnovni nedostatak u usporedbi sa simetričnim sustavom kriptiranja je brzina. Kriptiranje javnim ključem daleko je sporije i dugotrajnije od kriptiranja tajnim ključem. Također, dekriptiranje velikih količina podataka u asimetričnom sustavu kriptiranja oduzima puno vremena i računalnih resursa. Sljedeći nedostatak asimetričnog kriptiranja je količina štete koja može nastati ako napadač uspije doći do privatnog ključa. U tom slučaju, napadaču su otvorene sve poruke žrtve. Također, gubitak privatnog ključa često uzrokuje veliku štetu jer nije moguće dekriptirati primljene poruke. Jedan od nedostataka asimetričnog sustava kriptiranja je i taj što je nemoguće sa sigurnošću tvrditi, odnosno vjerovati da javni ključ pripada osobi koja se njime predstavlja, pa je potrebno raditi provjeru autentičnosti.

Tablica 6. sadrži popis osnovnih nedostataka i prednosti asimetričnog sustava kriptiranja.

PREDNOSTI	NEDOSTACI
Praktičnost	Mala brzina kriptiranja i dekriptiranja
Omogućava provjeru autentičnosti poruka	Iskorištavanje računalnih resursa
Omogućava provjeru integriteta poruka	Velika šteta kod gubitka privatnog ključa
Onemogućava negiranje pošiljatelja	Velika šteta u slučaju kompromitacije privatnog ključa

Tablica 6. Prednosti i nedostaci asimetričnog sustava kriptiranja

Sada kada smo naveli neke od prednosti i nedostataka jednog i drugog sustava kriptiranja, možemo pogledati kako je moguće iskoristiti najbolje iz svakog od njih. Zbog nedostatka asimetričnog sustava kriptiranja koji se odnosi na brzinu kriptiranja većih količina podataka, taj se sustav praktički i ne koristi u takve svrhe. Kako bi se nadišao taj problem, javila se ideja hibridnog kriptiranja. Smisao hibridnog kriptiranja je u korištenju prednosti simetričnog kriptiranja tajnim ključem zajedno s prednošću visoke razine sigurnosti koju pruža asimetrični sustav kriptiranja. Na taj način zadržava se brzina kriptiranja koju pruža simetrično kriptiranje, ali i visok stupanj zaštite koju omogućava asimetrični sustav. Asimetrično kriptiranje se u hibridnom kriptiranju koristi samo kako bi se na siguran način razmijenile informacije o tajnim ključevima simetričnog sustava. Nakon što se razmijene tajni ključevi, prelazi se na simetrično kriptiranu komunikaciju. Na taj način sigurnosni ključevi simetričnog sustava kriptiranja se razmijene na siguran način, a kasnije kriptiranje poruka se odvija na brz i efikasan način. Takvi, hibridni kriptosustavi se danas jako često upotrebljavaju u praksi, a možda najbolji primjer takvog sustava je upravo SSL/TLS protokol. S obzirom na to da asimetrični kriptosustavi koriste tajne parametre koje subjekti uključeni u komunikaciji ne dijele međusobno, podrazumijevaju se odgovarajući algoritmi koje provode različiti subjekti. Iz tog razloga se ti kriptosustavi definiraju kao skupovi algoritama koje provode različiti subjekti o čemu će više biti nešto kasnije. Primjeri o kojima ovdje govorimo uključuju asimetrične kriptografske sustave, sustave digitalnog potpisa, te protokole ključnih sporazuma.

5. PKI infrastruktura

U ovom poglavlju kroz nekoliko potpoglavlja razjasnit ćemo pojam PKI infrastrukture, njezine osnovne značajke, arhitekturu, te pojam certifikata. Također, vidjet ćemo kakve uloge danas imaju algoritmi koje smo opisali u prethodnim poglavljima u sigurnosti komunikacije.

5.1. Struktura PKI

Da bismo mogli ispravno shvatiti neke stvari u narednim poglavljima, važno je da razjasnimo osnovne pojmove koji se tiču PKI infrastrukture, te ostalih stvari koje se tiču same PKI.

U opisu načina rada SSL/TLS protokola, spomenuli smo da taj protokol radi s certifikatima javnog ključa. Međutim, upravljanje tim certifikatima nije dio tog protokola. Upravljanje javnim ključevima izdvojena je aktivnost koju, kao osnovni zadatak obavlja PKI.

Temelj PKI infrastrukture je asimetrična kriptografija. Do prije 40-50 godina aktivno se koristila kriptografija tajnog ključa. Međutim, kao što smo to već spomenuli, problem simetrične kriptografije bio je razmjena tajnih ključeva. 1976. godine, Whitfield Diffie i Martin Hellman su predstavili ideju o razmjeni tajnih ključeva korištenjem asimetrične kriptografije. Asimetrična kriptografija pružila je i mogućnost digitalnog potpisivanja podataka, a uveden je i pojam digitalnog certifikata koji povezuje javni ključ i entitet kojem on pripada. Sve te činjenice dovele su do potrebe za specifičnim sustavom koji će služiti za omogućavanje sigurne komunikacije putem nesigurnog komunikacijskog kanala.

PKI predstavlja cjelokupnost ljudi, sklopova, algoritama, sigurnosnih politika i procedura koje se upotrebljavaju za stvaranje, upravljanje, pohranjivanje i povlačenje digitalnih certifikata. Kod konkretne reprezentacije kriptografskog sustava, PKI ima ulogu poveznice između javih ključeva i njihovih pojedinačnih vlasnika. Za tu svrhu postoje certifikacijski centri, odnosno CA (eng. *Certification Authority*) organizacije koje funkcioniraju na način da pružaju procese registracije i izdavanja certifikata. Drugi naziv CA organizacija je još i TTP (eng. *trusted third party*). Taj posao, u ovisnosti o tome o kojoj se razini sigurnosti radi, može biti izveden od strane fizičke osobe ili programa namijenjenog za to. Web preglednici i operacijski sustavi dolaze s predefiniranom listom CA organizacijama kojima vjeruju. Takve organizacije se nazivaju korijenskim CA organizacijama (eng. *root CA*). Korijenske CA organizacije pak mogu ovlaštenje za kreiranje i provjeru certifikata koje posjeduju proslijediti, odnosno dodijeliti nekim drugim CA organizacijama za koje su odlučile da im vjeruju. U stručnoj terminologiji,

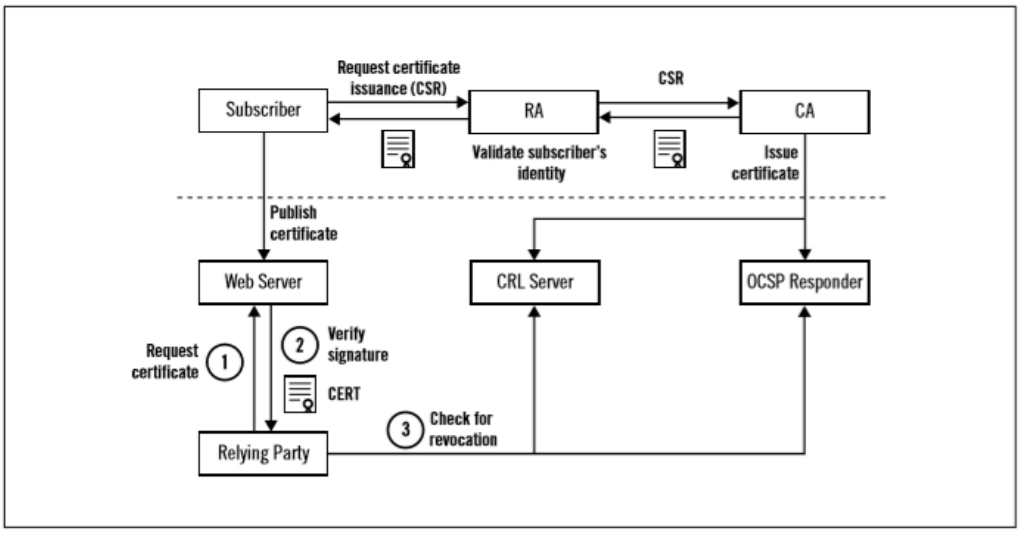
prethodno opisano povezivanje javnog ključa s njegovim vlasnikom ima registracijski centar, odnosno RA (eng. *Registration Authority*). [7, str. 10]

PKI zapravo pruža sigurnosnu osnovu na kojoj se grade druge aplikacije, te implementiraju sigurnosni sustavi. Temeljni zadatak PKI infrastrukture je suradnja s drugim sigurnosnim mehanizmima, kojima je zadatak praćenje i ostvarenje ciljeva sigurnosnih politika. Tim mehanizmima PKI služi kao potpora. Sektor telekomunikacijske standardizacije koji je dio međunarodne telekomunikacijske unije izdao je i kontinuirano nadograđuje preporuku ITU-T X.509 (kraće: X.509) koju su prihvatile mnoge ustanove koje se bave normizacijom, te mnoge od njih rade na prilagodbi te preporuke za specifična aplikacijska okruženja. [6, str. 211]

Struktura PKI sustava sastoji se od mnogo povezanih elemenata kao što su:

- certifikacijski centar
    - potpisuje i izdaje certifikate, obnavlja ih i po potrebi opoziva
    - jamči ispravnost podataka u potpisanom certifikatu
- registracijski centar
    - često kao dio certifikacijskog centra
    - registrira krajnje korisnike PKI sustava
    - često ima ulogu posrednika između krajnjeg korisnika i certifikacijskog centra u slučaju da dođe do kompromitiranja privatnog ključa
    - ako ne postoji registracijski centar, njegove funkcije obavlja certifikacijski centar
- baza izdanih certifikata
    - lista izdanih certifikata
    - lista opozvanih certifikata, odnosno CRL (eng. *Certification Revocation List*)
- izdavač opozvanih certifikata
    - ima ulogu izdavanja opozvanih certifikata
    - certifikat iz različitih razloga može postati nevažeći
- korisnički certifikati
    - povezuju javni ključ i samog korisnika
- korisničke aplikacije koje koriste PKI
    - sustavi koji kod implementacije sigurnosnih mehanizama koriste PKI infrastrukturu

Slika 7. predstavlja vizualni prikaz prethodno objašnjene PKI strukture.



Slika 7. Struktura PKI infrastrukture

Kako bi se uspostavili standardi izdavanja i obrade certifikata, te omogućilo njihovo provođenje, osnovana je dobrovoljna skupina koja se naziva CA/Browser Forum ili CAB Forum. Na početku je osnovni razlog postojanja CAB Foruma bilo definiranje standarda za izdavanje certifikata s proširenom validacijom. Međutim, 2012. godine CAB Forum je restrukturiran. Iste godine izdali su *Baseline Requirements* u kojem su opisani zahtjevi za izdavanje i upravljanje certifikatima javnog ključa. Iako CAB Forum broji tek četrdesetak CA članova, *Baseline Requirements* prihvaćen je od strane svih certifikacijskih centara.

5.2. Slabosti PKI infrastrukture

Da bismo lakše shvatili temu ovog poglavlja, najprije ću objediniti sadržaj prethodnih poglavlja, na način da ću opisati kako naš preglednik ili operacijski sustav odlučuje kojem web mjestu će vjerovati, odnosno kojem neće.

Danas u svijetu postoje stotine tisuća ljudi koji se bave izradom web stranica i programskih proizvoda razne namjene, ali u svakom trenutku postoje stotine onih koji to rade sa zlom namjerom. Radi se o hakerima koji dizajniraju svoje aplikacije i web stranice da izgledaju identično kao originalne aplikacije ili web stranice banaka ili društvenih mreža, a s ciljem da od korisnika na prevaru dođu do autentifikacijskih podataka ili pak kriptiraju podatke pohranjene u računalu i traže novac za vraćanje istih. Zbog takvih pojava, preglednici svaki put kad se preko njih pokuša pristupiti nekoj web stranici ili aplikaciji provjeravaju je li to upravo ona aplikacija ili web stranica kojom se predstavlja ili se radi o nekoj lažnoj stranici na koju nas preusmjerava neki haker. Također, istu stvar rade i operacijski sustavi kad pokušamo preuzeti neki softver s interneta. Međutim, proizvođači preglednika i operacijskih sustava ne žele biti odgovorni za provjeru svakog pojedinog web mjesta ili aplikacije, nego vjeruju CA organizacijama. CA organizacije, kao što je već bilo opisano u nekom od prethodnih poglavlja, obavljaju taj posao izdavanjem certifikata. Certifikati su, kao što smo već imali prilike uočiti u prethodnom poglavlju, osnovni faktor koji omogućava sigurno korištenje interneta, bilo kad koristimo Internet bankarstvo ili kad preuzimamo softver s interneta. CA organizacije obavljanjem posla provjere i izdavanja certifikata zarađuju novac. Danas na svijetu postoje stotine takvih organizacija, ali većina preglednika i operacijskih sustava, ako što smo već rekli, dolazi s navedenim samo nekih od tih organizacija koje nazivamo korijenskim CA organizacijama. Korijenske CA organizacije pak mogu ovlaštenje koje posjeduju proslijediti, odnosno dodijeliti nekim drugim CA organizacijama za koje su odlučile da im vjeruju. Bilo koja od spomenutih CA organizacija može izdati certifikat bilo kojem web mjestu, pa čak i web mjestima za koje su certifikati već izdani od strane neke druge CA organizacije. Kompleksnost čitavog sustava je jedan od razloga zašto stvari često mogu poći po zlu. Jednostavno je previše uključenih faktora kod kojih može doći do određenih sigurnosnih propusta.

Jedan od najvećih problema današnje PKI infrastrukture je u tome što za izdavanje certifikata nije potrebna suglasnost vlasnika domene na koju se certifikat izdaje. Drugim riječima, bilo koja CA organizacija može izdati certifikat za bilo koju domenu bez traženja nekog posebnog odobrenja. Ne postoje tehničke mjere koje bi onemogućile da se dogodi propust tijela za izdavanje certifikata ili bilo koji drugi sigurnosni problem. Prije nekoliko desetaka godina, kad je na cijelom svijetu postojalo tek nekoliko certifikacijskih centara, to se nije činilo kao velik problem. Međutim, danas takvih tijela ima na stotine, pa to počinje predstavljati problem. Sve CA organizacije su podložne raznim revizijama, ali prolazne ocjene na tim revizijama ne moraju značiti apsolutnu sigurnost. Nizozemska organizacija DigiNotar je primjer CA organizacije koja je prošla revizije, međutim 2011. godine njezina sigurnost je kompromitirana u potpunosti, pa su tako napadači izdavali certifikate za domene aol.com, microsoft.com, gmail.com, pa čak i cia.gov. Iz razloga što se kod izdavanja certifikata ne traži

odobrenje vlasnika domene, napadači su, nakon što su pronašli sigurnosni propust kod DigiNotar organizacije, moli bez ikakvih poteškoća izdavati certifikate za različite domene.

Sljedeća slabost PKI infrastrukture je slaba razina validacije domena. Slanje informacija potrebnih za izdavanje DV (eng. *Domain Validated*) certifikata koji se danas najviše koriste, a o kojima će više biti u sljedećem poglavlju, obavlja se putem nesigurnog WHOIS („who is“) protokola. Interakcija s krajnjim korisnikom se najčešće odvija putem elektroničke pošte, pa je to samo po sebi nesigurno. Napadač bez problema može dobiti lažni certifikat ako ukrade naziv domene ili uspije dobiti pristup poštanskom sandučiću. Također, moguće je napasti postupak validacijskog procesa kod CA organizacije na način da se mrežni promet presiječe na njezinom kraju.

Jedna od najvećih slabosti PKI infrastrukture je loš sustav opoziva certifikata. U 2011. godini i u godinama iza 2011. dogodilo se nekoliko velikih grešaka kod opoziva certifikata. Kod svake od grešaka krajnji korisnici su bili primorani koristiti vlastite liste opozvanih certifikata ili su morali raditi na vlastitim nadogradnjama kako bi pouzdano opozvali ugrožene certifikate. Postoji više izvora tog problema. Prvi od njih je taj što postoji veliko kašnjenje u širenju podataka o opozivu certifikata do svakog sustava. Preduvjeti koje je PKI osigurao omogućavaju da podaci CRL-a ostanu valjani do 10 dana. Drugim riječima, potrebno je najmanje 10 dana da se informacije o opozivu certifikata u potpunosti objave. Sljedeći problem je sigurnosna politika implementirana u većini današnjih preglednika, a prema kojoj preglednik pokuša dobiti informacije o opozivu certifikata, ali zanemaruje sve pogreške. Na primjer, napadač može onemogućiti zahtjeve prema OCSP (eng. *Online Certificate Status Protocol*) serveru, te si na taj način omogućiti da koristi lažni certifikat. Moguće rješenje tog problema je usvajanje certifikata koji mogu biti korišteni isključivo u kombinaciji sa svježim OCSP odgovorom.

Vjerojatno najveći problem PKI sustava je labavi pristup validaciji certifikata. Mnoge aplikacije zaobilaze validaciju certifikata. Preglednici provjeravaju certifikate, ali u slučaju da certifikat nije valjan samo obavještavaju korisnika o tome i nude mu alternativni put do web mjesta koji nije siguran. Prema nekim istraživanjima, čak do 70% korisnika nakon pročitanog upozorenja da web mjesto nije sigurno ipak zanemari upozorenje, te na takvim mjestima unosi osjetljive podatke kao što su lozinke. Time je svrha kriptiranja u potpunosti narušena. Zbog svega toga, osmišljen je HTTP Strict Transport Security, standard politike web sigurnosti koji pomaže u zaštiti web stranica od napada. Naime, taj standard upućuje sve preglednike da zamijene upozorenja o nevažećim certifikatima greškama, te da onemoguće pristup web mjestima koja nisu sigurna.

5.3. Digitalni certifikati

U ovom poglavlju će biti opisani digitalni certifikati, njihove podvrste, svrha digitalnih certifikata, te prednosti koje oni donose.

S tehničke strane gledano, digitalni certifikat je digitalni dokument koji se sastoji od javnog ključa, informacija o entitetu uz kojeg je vezan, te digitalnog potpisa izdavatelja. Digitalni certifikati zapravo predstavljaju temeljnu komponentu potrebnu za sigurno korištenje javnih ključeva. Kako bi uopće mogli početi komunikaciju putem interneta, sudionici prvo moraju znati javni ključ ostalih sudionika s kojima žele komunicirati. Međutim, kod razmjene javnih ključeva postoji mogućnost lažnog predstavljanja, pa se tako uljezi svojim javnim ključevima mogu predstaviti kao netko drugi. Kako bi se taj sigurnosni problem uklonio, Loren M. Kohnfelder je 1978. godine predstavio ideju digitalnih certifikata. Integritet certifikata osigurava njegov potpisnik svojim digitalnim potpisom.

Digitalni certifikati se sastoje od sljedećih atributa, odnosno vrijednosti:

- Verzija (eng. *Version*)
    - postoje 3 verzije certifikata
    - 1. verzija sadrži samo osnovne atribute
    - 2. verzija sadrži osnovne atribute i jedinstvene identifikatore
    - 3. verzija sadrži isto što i 2. verzija, te niz dodatnih proširenja
    - Danas je većina certifikata 3. verzije
- Serijski broj (eng. *Serial number*)
    - U početku su to bili pozitivni cijeli brojevi koji su na jedinstven način identificirali certifikat izdan od strane neke CA organizacije
    - Danas za serijske brojeve postoje dodatni zahtjevi
        - Moraju biti nesekvencijalni
        - Moraju se sastojati od najmanje 20 bitova
- Algoritam potpisa (eng. *Signature Algorithm*)
    - Nosi informaciju o algoritmu koji se koristi za potpis certifikata
- Izdavač (eng. *Issuer*)
    - DN (eng. *Distinguished Name*), odnosno razlikovni naziv izdavača certifikata
    - Sastoji se od više komponenata (ovisno o izdavaču)
- Valjanost (eng. *Validity*)
    - Vremenski interval u kojem je certifikat važeći
    - Sastoji se od datuma početka i datuma kraja valjanosti

- Vlasnik (eng. *Subject*)
    - Razlikovni naziv vlasnika certifikata, odnosno entiteta čiji koji je u posjedu javnog ključa za koji je izdan certifikat
    - Danas se ne preporučuje korištenje ovog atributa, nego se preporučuje korištenje atributa *Subject Alternative Name*
- Javni ključ (eng. *Public key*)
    - Sadrži javni ključ vlasnika certifikata uz ID algoritma i opcionalne parametre

Navedeni atributi se odnose na certifikate 1. verzije. 2. verzija certifikata sadržavala je još sljedeće atribute:

- Jedinstveni ID vlasnika (eng. *Subject Unique ID*)
    - U verziji 3 zamijenjen proširenjem *Subject Key Identifier*
- Jedinstveni ID izdavača
    - U verziji 3 zamijenjen proširenjem *Authority Key Identifier*

Danas se najčešće koriste certifikati verzije 3. Certifikati verzije 3 sadrže još neka dodatna proširenja:

- *Subject Alternative Name*
    - Zamjenjuje atribut *Subject* koji nije pružao dovoljno fleksibilnosti
- *Name constraints*
    - Omogućuje organizacijama da dobiju ulogu CA organizacije koja može izdavati certifikate samo za nazive dome koje su u vlasništvu te organizacije
- *Basic Contraints*
    - Služi za kontrolu dubine podređenih certifikata (određuje može li CA certifikat izdati ugniježđene certifikate)
- *Key Usage*
    - Definira za što je moguće koristiti ključ koji je u certifikatu
    - Postoji ograničen broj mogućnosti
- *Extended Key Usage*
    - Služi za fleksibilnije specificiranje mogućnosti korištenja ključa
    - Dozvoljava proizvoljne dodatne vrijednosti koje se mogu specificirati
- *Certificate Policies*
    - Sadrži listu pravila

    - Obično sadrži poveznicu na stranice na kojima se mogu dobiti cjeloviti tekstovi pravilnika
- *CRL Distribution Points*
    - Sadrži informaciju o lokaciji liste opozvanih certifikata
- *Authority Information Access*
    - Označava kako pristupiti informacijama i uslugama koje pruža izdavač certifikata
    - Sadrži URI (eng. *Uniform Resource Identifier*) OCSP servera [1, str.66]

Važno je razlikovati DV (eng. *Domain validated*), OV (eng. *Organization validated*) i EV (eng. *Extended validation*) certifikate. DV certifikati su od spomenute tri kategorije certifikati koji se najčešće koriste danas, a provjeravaju se preko domene. Također, interakcija sa CA organizacijom se najčešće provodi putem elektroničke pošte. DV certifikat se relativno lako može nabaviti, te se kod ove vrste certifikata zahtijeva najslabiji oblik validacije od strane CA organizacije, što nije pozitivna karakteristika. Još jedna negativna karakteristika je i jamstvo koje CA organizacije daju za DV certifikate. Šteta koja je pokrivena jamstvom je najčešće između 10000$ i 20000$.

OV certifikati su stepenica više od DV certifikata što se tiče sigurnosti. Naime, CA organizacije prije izdavanja certifikata traže potvrdu vlasništva nad tvrtkom. Prednost OV certifikata je i lakše praćenje certifikata u poduzeću. Certifikati se izdaju na me tvrtke, a mogu biti dodijeljeni i specifičnim odjelima tvrtke. Također, pozitivna strana je i jamstvo. Naknade u slučaju štete prouzrokovane CA organizacijom su višestruko veće u odnosu na DV certifikate.

EV certifikati se smatraju najsigurnijim certifikatima. Upravo zbog toga ih često koriste velike organizacije, banke i web trgovine. Time postižu povjerenje kod klijenata. EV certifikati dolaze s jamstvima vrijednim po više milijuna dolara, te nude neke stvari koje ostale vrste certifikata ne nude (npr. zelena traka s nazivom organizacije pored adresne trake). No, s takvim „premium“ obilježjima dolazi i takva cijena, pa su EV certifikati daleko najskuplja vrsta certifikata. Zbog toga je potrebno najprije dobro proučiti treba li nam EV certifikat ili je dovoljan OV certifikat. [1, str. 250]

5.4. Hijerarhijski model povjerenja

U ovom poglavlju objasnit ću hijerarhijski model povjerenja, odnosno lanac certifikata. Također, objasnit ću kako stvari funkcioniraju kod danas najkorištenijih preglednika, te koji od njih podržavaju takav model povjerenja, a koji ne.

U hijerarhijskom modelu povjerenja korijenski CA u sebi nosi certifikat koji je sam izdao i potpisao, te predstavlja takozvano „sidro povjerenja“. Kod klijenta je u postavkama zadano da vjeruje korijenskom CA. Inače, samopotpisani certifikati (oni kod kojih je izdavač i subjekt za kojeg se izdaje certifikat jedan te isti subjekt) nisu od velike koristi. Razlog tome je taj što bilo koji subjekt može sam sebi izdati certifikat i potpisati ga. Međutim, u hijerarhijskom modelu povjerenja su samopotpisani certifikati neizbježna pojava jer je hijerarhija strogo definirana i zbog toga mora imati početnu razinu od koje sve kreće. Za razliku od korijenskog CA, certifikatima niže razine u hijerarhiji se ne vjeruje bezuvjetno. Njima se vjeruje iz razloga jer su izdani od strane korijenskog CA kojem se vjeruje, te su od iste strane i potpisani. Bilo koji od CA-ova niže razine može izdati certifikat nekom krajnjem korisniku, ali krajnji korisnik ne može izdati ni potpisati certifikat. Klijent s takvim postavkama može pokušati pronaći put od korijenskog certifikata do krajnjeg certifikata izdanog nekom korisniku ili sustavu. Put, odnosno lanac certificiranja definira se u DIT (eng. *Directory Information Tree*) stablu koje sadrži niz od korijenskog preko posrednih do krajnjeg korisničkog certifikata. U tom nizu, svaki certifikat iz više razine ovjerava javni ključ certifikata niže razine.

Danas postoje preglednici koji podržavaju hijerarhijski model povjerenja u smislu da razlikuju korijenski od certifikata niže razine, ali postoje i oni koji ga ne podržavaju. Preglednici koji podržavaju hijerarhijski model oslanjaju se na funkcionalnosti upravljanja certifikatima koje pruža operacijski sustav. Primjer preglednika koji podržavaju taj model su Microsoftov Internet Explorer i Google Chrome, dok s druge strane Mozilla Firefox ne podržava taj model.

5.5. VeriSign

U ovom potpoglavlju opisat ću VeriSing certifikate, povijest VeriSigna, te će biti nekoliko riječi i o Symantec-u, s obzirom na to da da je VeriSign u vlasništvu Symantec-a.

Kako kaže opis na službenim stranicama Verisign-a, oni su globalni pružatelj usluga registra domena i internetske infrastrukture. Sjedište im je u Sjedinjenim Američkim Državama. VerSign ima ulogu osiguravatelja stabilnosti, otpornosti i sigurnosti internetske infrastrukture i

usluga, što uključuje i upravljanje dvama od trinaest globalnih korijenskih poslužitelja interneta. VeriSign pruža usluge registracije i mjerodavnih rješenja za domene .com i .net na koje je registrirana većina globalne e-trgovine.

VeriSign je nastao 1995.godine kao tvrtka koja, između ostalih usluga, pruža i usluge certificiranja. Primarna djelatnost tvrtke bilo je izdavanje certifikata. 2010. godine VeriSign je prodao svoje poslovne jedinice za autentifikacijska rješenja Symantec-u. Do tada je VeriSign izdao više od 3 milijuna certifikata što za vojne, što za financijske usluge, maloprodajne aplikacije, itd. Danas je osnovna djelatnost VeriSign-a registracija domena.

Kroz povijest svojeg postojanja VeriSign je imao nekoliko ozbiljnijih incidenata, što poslovnih, što pravnih. Tako je 2001. godine izdano nekoliko certifikata pojedincima koji su se lažno predstavili kao Microsoft-ovi djelatnici. Greška je otkrivena tek dva tjedna kasnije i to u rutinskoj kontroli iz razloga što VeriSign nije podržavao CRLDP (eng. *Certificate Revocation List Distribution Point*) protokol koji omogućava automatsku detekciju opozvanih certifikata. To je Microsoft-ove klijente dovelo u rizik, pa je Microsoft morao čak izdati sigurnosnu zakrpu kako bi riješio taj problem. Već 2002. godine, imali su problema sa zakonom jer su klijentima drugih tvrtki za registraciju domena sugerirali da im domena ističe, te pod krinkom obnove domene preuzimali te klijente od drugih tvrtki. Ubrzo im je to zabranjeno. Kroz povijest je VeriSign imao još nekoliko zakonskih incidenata manje važnosti.

5.6. PKI u Hrvatskoj

U ovom potpoglavlju objasnit ću ulogu FINE kao pružatelja usluga certificiranja u Republici Hrvatskoj.

Financijska agencija predstavlja vodeću hrvatsku tvrtku koja se bavi pružanjem financijskih i elektroničkih usluga. Tvrtka je u stopostotnom vlasništvu Republike Hrvatske, ali posluje isključivo na tržišnom principu. FINA dugi niz godina uspješno surađuje s bankama, te mnogim državnim institucijama, ministarstvima i ostalim tijelima. Na području javnih financija, FINA je za državu provela nekoliko velikih ključnih projekata. Općenito, u Republici Hrvatskoj FINA ima ključnu ulogu u digitalizaciji države.

Uz sve ostale djelatnosti, FINA je u RH najveći pružatelj usluge PKI, odnosno izdavanja kvalificiranih certifikata. Da bi mogla obnašati tu dužnost, FINA mora poslovati u skladu s rigoroznim sigurnosnim pravilima. Na novoj produkcijskoj okolini FINA-e, digitalni certifikati i vremenski žigovi u skladu su sa svim važećim međunarodnim normama i normama Europske unije. Radi se o normama iz područja elektroničkog potpisa, izdavanja digitalnih certifikata i vremenskih žigova, te najboljim praksama. Novi sustav za izdavanje certifikata temelji se na

dvorazinskoj arhitekturi certifikacijskih tijela. Taj sustav sastoji se od korijenskog CA koji izdaje certifikate za posrednička certifikacijska tijela koja pak izdaju certifikate krajnjim korisnicima. Na produkcijskoj razini, FINA ima spomenuto korijensko certifikacijsko tijelo koje se naziva „Fina Root CA“, te dva posrednička, „Fina RDC 2015“ i „Fina RDC-TDU 2015“. „Fina RDC 2015“ izdaje certifikate za fizičke osobe (osobni certifikati), fizičke osobe koje su povezane sa poslovnih subjektom (poslovni certifikati), te IT opremu koja je povezana sa poslovnim subjektom (poslovni certifikati za IT opremu). „Fina RDC-TDU 2015“ bavi se izdavanjem certifikata za državne dužnosnike i zaposlenike u tijelima državne uprave. Za opisana dva posrednička tijela certifikat je izdalo tijelo „Fina Root CA“, te je to praktični primjer hijerarhijskog modela povjerenja koji je opisan u jednom od prethodnih poglavlja. U tom primjeru, „Fina Root CA“ predstavlja spomenuto sidro povjerenja, te izdaje i potpisuje certifikate za posredne CA-ove.

Kod potpisivanja certifikata, za izračun sažetka korisni se algoritam SHA-256. Duljina CA ključeva je 4096 bitova, dok je duljina korisničkih ključeva 2048 bitova. U oba slučaja za kriptiranje se koristi RSA algoritam. FINA također nudi i uslugu za online provjeru valjanosti certifikata. FINA OCSP servis se temelji na klijent-server arhitekturi. OCSP klijent stranke šalje FINA-inom OCSP serveru zahtjev za provjeru statusa certifikata, a OCSP servis mu vraća odgovor. OCSP servis daje informacije o certifikatima koji su izdani od strane Fina Root CA, Fina RDC 2015 ili Fina RDC-TDU 2015. Adresa FINA-inog OCSP servera nalazi se u polju *Authority Information Access* svakog FINA-inog produkcijskog certifikata. Kako bi se mogla provjeriti autentičnost odgovora, FINA-in OCSP server potpisuje odgovore na zahtjeve RSA privatnim ključem duljine 2048 bitova, pri čemu koristi kriptografske algoritme SHA-256 i RSA. Uz to, provjera statusa certifikata može se obaviti i dohvatom CRL, ali se preporučuje korištenje OCSP servera, osim ako isti nije dostupan zbog tehničkih poteškoća.

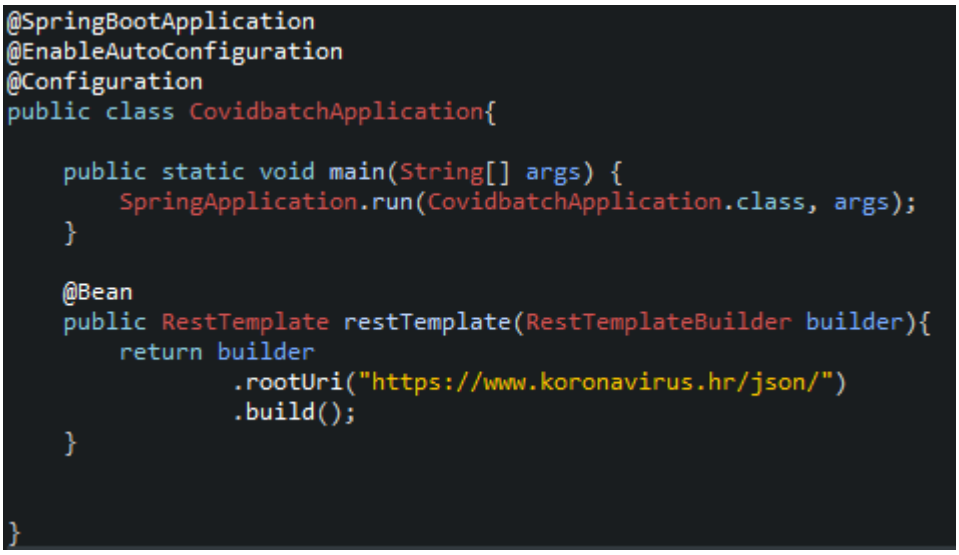
6. Praktični primjer

U ovom poglavlju objasnit ću kroz praktični primjer primjenu certifikata u identifikaciji web mjesta. S obzirom na trenutno aktualnu situaciju s koronavirusom, za tematiku sam odabrao izradu aplikacije za pregled podataka o kretanju broja zaraženih osoba koronavirusom u Hrvatskoj. Tehnologije koje sam koristio u izradi aplikacije dijelom sam upoznao već tijekom studiranja, dok sam neke tehnologije upoznao tijekom studentskog rada.

6.1. Korištene tehnologije

U izradi aplikacije za pregled broja zaraženih osoba koronavirusom koristio sam različite tehnologije, a mogu ih podijeliti u četiri osnovne kategorije: baza podataka, back-end aplikacija, front-end aplikacija i hosting poslužitelj. Svaku od navedenih tehnologija ću u ovom poglavlju objasniti, te ću objasniti na koji način sam u iskoristio za potrebe razvoja aplikacije.

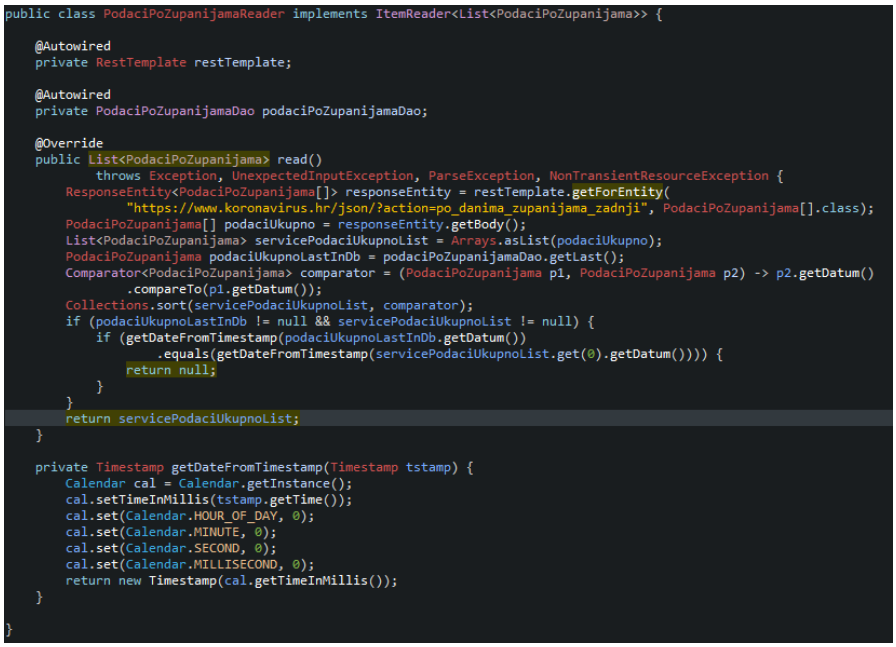
Najprije bih želio objasniti u kratkim crtama kako sustav funkcionira, te iz kojih izvora vuče podatke. Cijeli sustav je zamišljen na sljedeći način. Postoje 3 aplikacije. Prva od njih je batch aplikacija koja svakodnevno opskrbljuje sustav svježim podacima. Batch aplikacije imaju ulogu serijske obrade podataka. Osnovni zadatak im je zamijeniti serijsku obradu podataka koju bi inače trebali raditi ljudi. Zadaci koje obavljaju batch aplikacije obično se odnose na čitanje, obradu, odnosno pretvorbu podataka, te zapisivanje obrađenih podataka u novo spremište podataka koje može biti datoteka, baza podataka i sl. Koraci obrade najčešće su povezani određenim pravilima na način da se pojedini koraci mogu obavljati i paralelno. U izradi batch aplikacije koristio sam Spring Framework. Spring Framework predstavlja aplikacijski okvir čije osnovne funkcionalnosti mogu biti korištene u izgradnji bilo kakve Java aplikacije. Spring Framework je programski okvir otvorenog koda, te uvelike programerima olakšava i ubrzava izradu Java aplikacija. Za kreiranje projekta batch aplikacije iskoristio sam Spring Initializr alat koji služi za generiranje „kostura“, odnosno strukture aplikacije. U tom procesu ne generira se kod aplikacije, već samo osnovna struktura sa dodacima koji se mogu dinamički odabrati. Prilikom kreiranja projekta pomoću tog alata kreiraju se dvije osnovne datoteke: glavna klasa aplikacije koja služi za pokretanje same aplikacije i pom.xml datoteka koja služi za uključivanje različitih modula koji će se koristiti u razvoju aplikacije. Slika 8. prikazuje izgled glavne klase batch aplikacije.



```
@SpringBootApplication
@EnableAutoConfiguration
@Configuration
public class CovidbatchApplication{

    public static void main(String[] args) {
        SpringApplication.run(CovidbatchApplication.class, args);
    }

    @Bean
    public RestTemplate restTemplate(RestTemplateBuilder builder){
        return builder
                .rootUri("https://www.koronavirus.hr/json/")
                .build();
    }


}
```

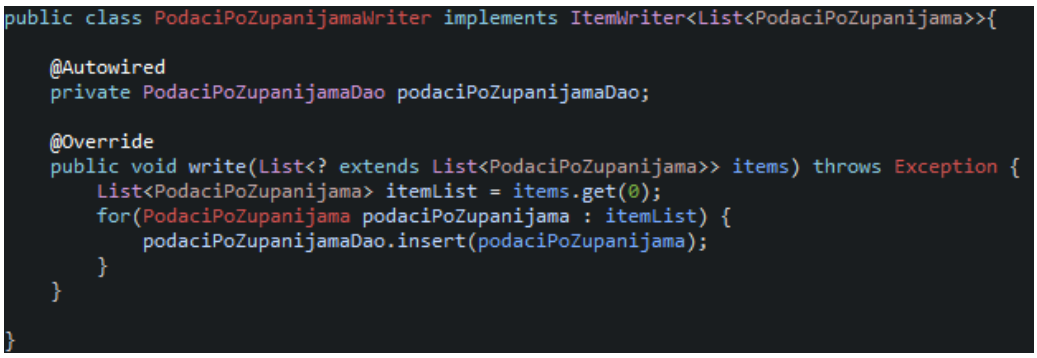
Slika 8. Prikaz glavne klase aplikacije

„restTemplate“ metoda nije dodana u klasu prilikom generiranja aplikacije, već sam tu metodu dodao sam. Svrha te metode je kreiranje komponente aplikacije koja će služiti za dohvat javno dostupnih strojno čitljivih podataka. Na putanji https://www.koronavirus.hr/json/ zapravo se nalazi REST (eng. *Representational state transfer*) servis koji je napravila Vlada RH, a koji služi za pružanje pravodobnih i točnih informacija o koronavirusu. U ostatku aplikacije, najvažnije komponente su komponenta koja obavlja čitanje podataka (eng. *Reader*), te komponenta koja obavlja zapisivanje podataka u bazu sustava (eng. *Writer*). Treća komponenta koja inače u batch aplikacijama također igra važnu ulogu je komponenta za obradu podataka (eng. *Processor*). Međutim, u mojem slučaju obrada podataka nije ništa drugo osim pretvorbe u novi model pogodan za spremanje u bazu podataka. Slika 9. prikazuje implementaciju čitača podataka po županijama.



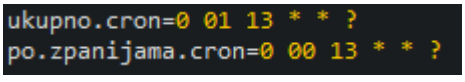
Slika 9. Prikaz implementacije čitača podataka

Slika 10. prikazuje implementaciju pisača koji služi za zapisivanje dohvaćenih podataka za županije u bazu.



Slika 10. Prikaz pisača podataka za županije

Batch aplikacija podatke mora prikupljati i obrađivati u točno određenim vremenskim intervalima koji se mogu konfigurirati. U mom slučaju, aplikacija podatke dohvaća svaki dan točno u 15 sati. Za implementaciju te mogućnosti koristio sam cron izraz. S obzirom na to da je aplikacija postavljena na serveru koji nije u našoj vremenskoj zoni, cron izraz sam postavio na 13 sati, kako bi se obrada podataka izvršavala u 15 sati po našem lokalnom vremenu. Slika 11. prikazuje izgled konfiguracije cron izraza.



Slika 11. Cron izrazi u batch aplikaciji

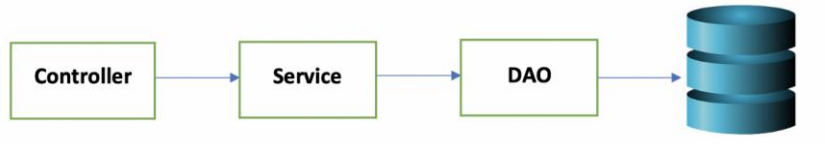
Druga od spomenute tri aplikacije je REST servis koji služi za dohvaćanje podataka iz baze i opskrbu front-end aplikacije. REST predstavlja arhitekturalni stil izveden iz poznatog klijent-poslužitelj stila, koji pruža visoke performanse i skalabilnost. Osnovna značajka REST stila koja ga razlikuje od ostalih arhitekturalnih stilova je definiranje jedinstvenog sučelja koje se u pravilu dijeli među svim komponentama arhitekture. REST zahtijeva da se web resursi identificiraju putem jedinstvenih globalnih adresa ili identifikatora kao što su URI-ovi (eng. *Uniform Resource Identifier*). [8, str. 2]

Kod implementacije REST servisa također sam koristio Spring Boot tehnologiju, te sam i ovaj put, baš kao i kod izrade batch aplikacije, koristio Spring Initializr alat za kreiranje osnovne strukture projekta. Razlika je ovaj put u pom.xml datoteci u kojoj sam uključio neke druge module koji su mi bili potrebni u implementaciji. Također, kod REST servisa sam koristio neke drugačije postavke, pa i application.properties datoteka izgleda drugačije. Slika 12. prikazuje application.properties datoteku REST aplikacije. Pojedini dijelovi slike su cenzurirani iz sigurnosnih razloga.

```
spring.datasource.url=jdbc:mysql://164.90.222.161:25060/defaultdb?useSSL=true
spring.datasource.username=
spring.datasource.password=

server.port=43210

server.ssl.key-store-type=PKCS12
server.ssl.key-store=classpath:164.90.223.110.p12
server.ssl.key-store-password=

server.ssl.enabled=true
```

Slika 12. Prikaz postavki aplikacije

Kao što je vidljivo iz postavki, aplikacija je pokrenuta na portu 43210, kod povezivanja s bazom podataka koristi se SSL, a i sami REST servis se predstavlja certifikatom. REST servis sastoji se od više klasa u različitim paketima. Osnova REST servisa je klasa kontrolera koja sadrži metode koje „čekaju“ zahtjeve na različitim putanjama, te ih obrađuju na način da dohvaćaju tražene podatke i poslužuju ih u JSON formatu. Kod dohvata podataka, kontroler koristi servisne klase u kojima je implementirana logika te poziv DAO (eng. *Dana Access Object*) objekata za pristup bazi podataka koji kao jedinu ulogu imaju dohvat podataka iz baze. Slika 13. prikazuje prethodno opisanu strukturu kontrolera, servisnog sloja i sloja pristupa bazi podataka.



Slika 13. Prikaz slojeva REST servisa

Konačno, treća aplikacija je front-end aplikacija koja služi za vizualizaciju podataka o kretanju broja zaraženih koronavirusom u Hrvatskoj. U izradi te aplikacije koristio sam znanja stečena na internom tečaju tvrtke u kojoj radim. Front-end aplikacija implementirana je u React-u. React predstavlja Javascript biblioteku koja služi za izradu korisničkih sučelja. React je danas jedan od najpopularnijih biblioteka koje se koriste u razvoju korisničkih sučelja, te je njegova krivulja korištenja u rastu. Pojavio se 2011. godine, a razvijen je od strane Facebook-a.

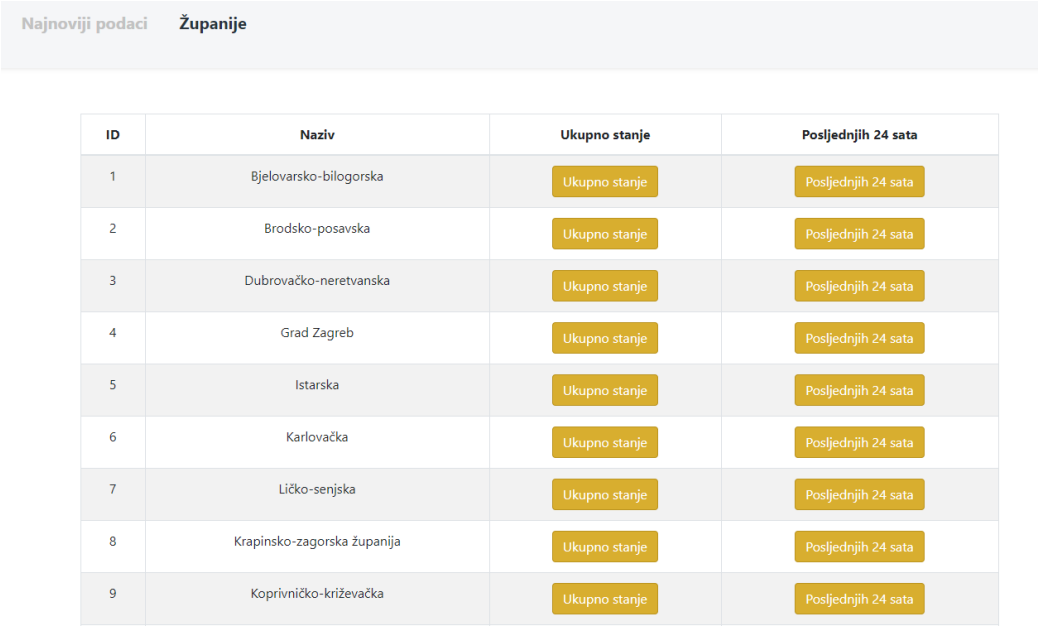
U ovome radu neću previše zalaziti u dubinu što se tiče React biblioteke jer se ne osjećam još dovoljno iskusnim u njezinom korištenju s obzirom da mi je ovaj diplomski rad bio prva aplikacija koju sam samostalno razvio u React-u, već ću samo ukratko objasniti kako funkcioniraju stvari. React kod sastoji se od različitih komponenata. Te komponente su ponovno iskoristive, te se renderiraju u DOM (eng. *Document Object Model*) uz pomoć React DOM biblioteke. Postoje funkcionalne komponente i klasne komponente. Razlika između njih je ta da funkcionalne komponente predstavljaju običnu Javascript funkciju i ne mogu imati stanje, dok klasne komponente mogu imati stanje koje se tijekom vremena može mijenjati. U spomenutom stanju najčešće se pohranjuju podaci dohvaćeni iz nekog servisa koji se prikazuju korisniku na ekranu. Kako kod učitavanja stranice treba proći određeno vrijeme da se podaci dohvate, a i tijekom rada na aplikaciji korisnik može mijenjati neke podatke (ažurirati/dodavati/brisati), postoji render() metoda koja je ujedno i jedina metoda koju svaka komponenta React aplikacije mora imati. Uloga te metode je da vrati vrijednost koja React-u daje do znanja koji dio podataka na ekranu je potrebno renderirati. Tako u slučaju da korisnik na primjer izbriše jedan redak iz tablice, kod osvježavanja podataka neće se osvježavati cijeli ekran, već samo komponenta tablice.

Kod aplikacija dostupan je na GitHub sustavu za verzioniranje na sljedećim poveznicama:

- Batch aplikacija (Spring Boot): https://github.com/vcerovecki/covid-batch
- REST servis (Spring Boot): https://github.com/vcerovecki/covid-rest
- Front-end aplikacija (React): https://github.com/vcerovecki/covid-app

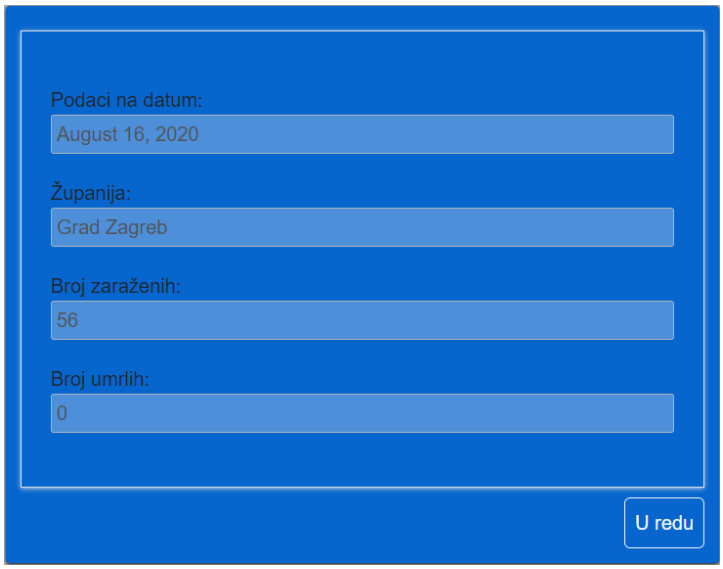
6.2. Prikaz korištenja aplikacije

Kroz sljedećih nekoliko slika prikazat ću mogućnosti aplikacije koju sam napravio. Slika 14. prikazuje tablični pregled županija kod kojeg je za svaku županiju moguće odabrati pregled ukupnog broja dosad zaraženih koronavirusom ili pak broj novo-zaraženih u posljednja 24 sata.



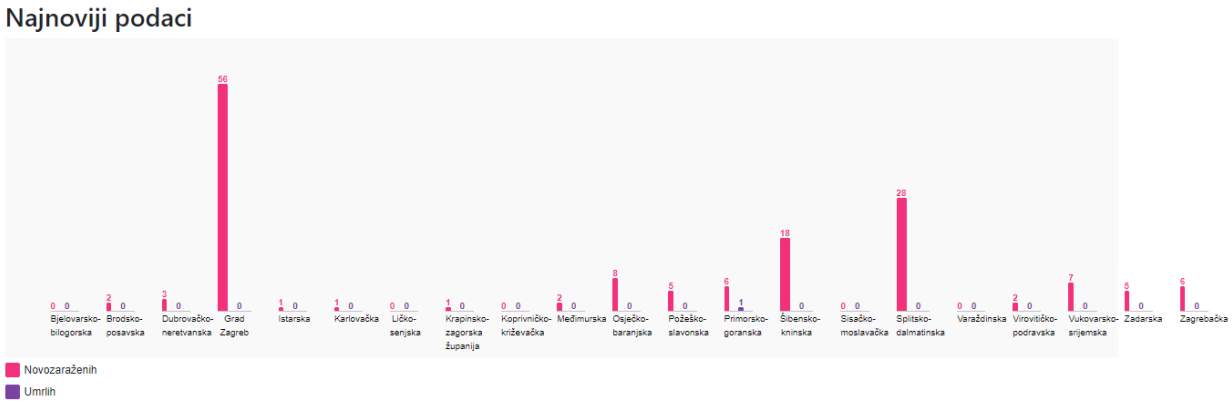
Slika 14. Pregled županija

Slika 15. prikazuje podatke za posljednjih 24 sata za županiju Grad Zagreb, a slični prikaz je i za ukupno stanje.



Slika 15. Prikaz podataka za Grad Zagreb

Klikom na karticu „Najnoviji podaci“ prikazuju se dva stupčasta grafa od kojih jedan prikazuje ukupno stanje u Hrvatskoj od početka bilježenja podataka do posljednjeg ažuriranja. Drugi graf prikazuje najnovije podatke za posljednja 24 sata. Slika 16. prikazuje graf koji se odnosi na posljednja 24 sata.



Slika 16. Prikaz podataka o broju novo-zaraženih u posljednja 24 sata

Aplikaciju sam odlučio staviti na javni poslužitelj. Nakon istraživanja različitih opcija, zaključio sam da je za mene najbolja opcija Digitalocean platforma. Na Digitalocean-u sam kreirao korisnički račun, instalirao Ubuntu operacijski sustav, Nginx server, MySQL, Javu 11, yarn (za React aplikaciju). Slika 17. prikazuje izgled konfiguracije Nginx poslužitelja.

```
server {
        listen 443 default_server;
        ssl on;
        ssl_certificate /etc/letsencrypt/live/ecorona.tk/fullchain.pem;
        ssl_certificate_key /etc/letsencrypt/live/ecorona.tk/privkey.pem;
        ssl_password_file /etc/ssl/ssl.pass;

        root /var/www/covid-app/build;
        index index.html index.htm;

        server_name ecorona.tk www.ecorona.tk;

        location / {
                try_files $uri /index.html;
        }

        location /eCoronaRestApi/ {
                proxy_pass http://localhost:43210/;
        }


}

server {
        listen 80;
        server_name _;
        return 301 https://$host$request_uri;
}
```

Slika 17. Konfiguracija Nginx poslužitelja

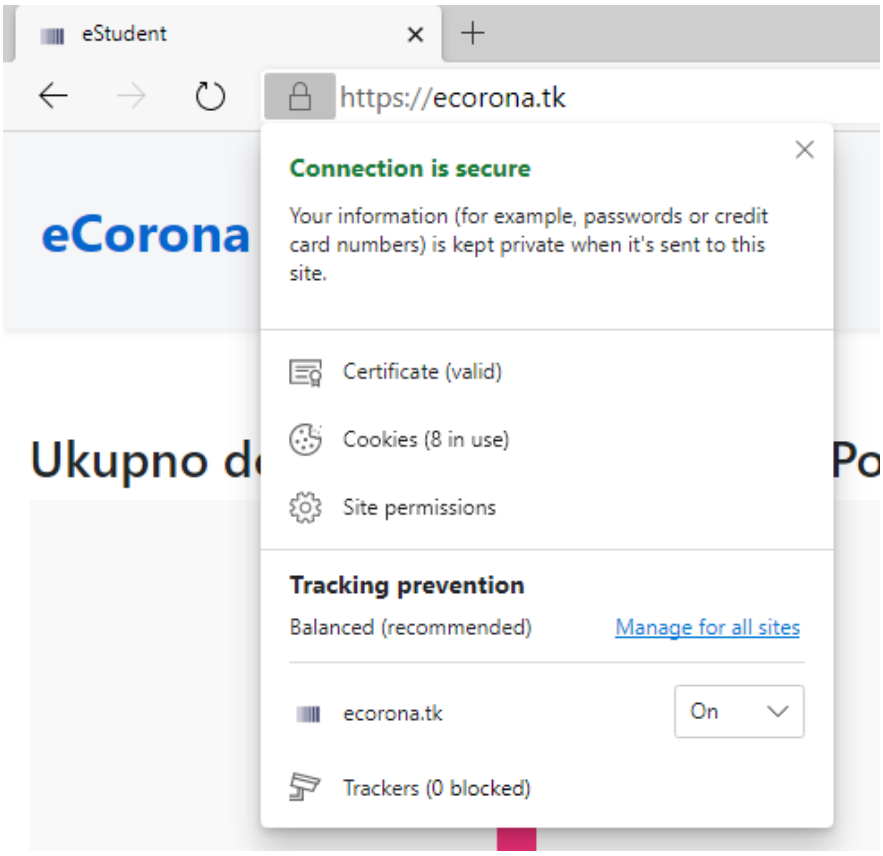
Prvi dio konfiguracije definira postavke potrebne da bi se omogućio SSL. Zadani port za SSL je 443. Potrebno je također specificirati i putanju do certifikata i privatnog ključa, te do

datoteke u kojoj je lozinka za privatni ključ. *location* atributi definiraju na koje putanje će se preusmjeravati zahtjevi u slučaju pojedine kontekst-putanje. U mom slučaju, bazna putanja vodi do React aplikacije, dok /eCoronaRestApi vodi do Rest servisa. Element *server_name* sadrži naziv domene aplikacije. U mom slučaju to je jako važan element zbog postavki koje ću još objasniti. Registrirao sam besplatnu domenu *ecorona.tk* te sam kod registratora podesio Nameserver parametre na NS1.DIGITALOCEAN.COM, NS2.DIGITALOCEAN.COM i NS3.DIGITALOCEAN.COM. Slika 18. prikazuje konfiguraciju domene na strani Digitalocean platforme.

Type	Hostname	Value	TTL (seconds)	
A	www.ecorona.tk	directs to 164.90.223.110	3600	More
A	ecorona.tk	directs to 164.90.223.110	3600	More
NS	ecorona.tk	directs to ns3.digitalocean.com.	1800	More
NS	ecorona.tk	directs to ns1.digitalocean.com.	1800	More
NS	ecorona.tk	directs to ns2.digitalocean.com.	1800	More

Slika 18. Konfiguracija domene na strani Digitalocean platforme

Prva dva retka tablice predstavljaju mapiranje domene na IP adresu „droplet-a“ na kojem se nalaze aplikacije. Na Slici 19. vidljivo je da je u web pregledniku uz domenu prikazan simbol lokota. Taj simbol u pregledniku označava da je web mjesto sigurno, te da je certifikat web mjesta važeći.



Slika 19. Prikaz uspješne identifikacije certifikatom (Microsoft Edge preglednik)

Aplikaciji se može pristupiti putem sljedećeg linka: https://ecorona.tk . Također, server je konfiguriran na način da i HTTP zahtjeve preusmjerava na HTTPS, tako da u pregledniku nije potrebno navoditi protokol.

7. Zaključak

U ovom diplomskom radu opisani su najvažniji kriptografski algoritmi koji se danas koriste u omogućavanju sigurne komunikacije putem nesigurne Internet mreže. Opisan je i SSL/TLS protokol koji je također ključan faktor u osiguravanju sigurne komunikacije, te su opisani simetrični i asimetrični kriptografski algoritmi i način korištenja kombinacije istih u svrhu postizanja veće efikasnosti. Nadalje, opisani su pojmovi vezani uz infrastrukturu javnog ključa, uloga Financijske agencije u Republici Hrvatskoj kao pružatelja usluga digitalnog certificiranja, te hijerarhijski model povjerenja. U praktičnom dijelu implementiran je sustav koji služi za vizualizaciju aktualnih podataka o kretanju broja zaraženih osoba koronavirusom, te je pritom implementirana identifikacija poslužitelja digitalnim certifikatom. U implementaciji je korišteno Eclipse razvojno okruženje, Spring Boot razvojni okvir, MySQL baza podataka, te Nginx poslužitelj na platformi Digitalocean. Kroz izradu ovog diplomskog rada, kao što mi je i bila želja, naučio sam mnogo toga o kriptografiji, SSL/TLS protokolu i općenito infrastrukturi čija je uloga omogućavanje sigurne komunikacije, te sada imam širu sliku važnosti implementacije sigurnosnih mehanizama kod web aplikacija.

Na temelju povratnih informacija koje sam dobio od kolega, a koje se odnose na sustav koji sam implementirao, predvidio sam već neka poboljšanja koja u skorijoj budućnosti svakako planiram implementirati.

Popis literature

[1] I. Ristić, *Bulletproof SSL and TLS : Understanding and Deploying SSL/TLS and PKI to Secure Servers and Web Applications*, United Kingdom: Feisty Duck Limited. 2014.

[2] „Uvod u Internet“ (bez dat.). Nacionalni portal za učenje na daljinu „Nikola Tesla“, dostupno: https://tesla.carnet.hr/mod/book/tool/print/index.php?id=5428#ch879 [pristupano 02.04.2020.]

[3] „TLS protokol“, CARNet CERT, dostupno : https://www.cert.hr/wp-content/uploads/2009/03/CCERT-PUBDOC-2009-03-257.pdf [pristupano 02.04.2020.]

[4] „Scytale“, Wikipedia, dostupno : https://en.wikipedia.org/wiki/Scytale [pristupano 03.04.2020.]

[5] „Kriptografija u službi napadača“, CARNet CERT, dostupno : https://www.cert.hr/wp-content/uploads/2019/04/CCERT-PUBDOC-2008-04-226.pdf [pristupano 05.04.2020.]

[6] R.Oppliger, *SSL and TLS : Theory and Practice*, Switzerland: eSECURITY Technologies. 2009.

[7] „Nedostaci PKI infrastrukture“, CARNet CERT, dostupno : https://www.cert.hr/wp-content/uploads/2009/02/CCERT-PUBDOC-2009-02-255.pdf [pristupano 05.07.2020.]

[8] C.Pautasso, E.Wilde, R.Alacron, *REST: Advanced Research Topics and Practical Applications*, New York: Springer. 2014.

Popis slika

Slika 1. Osi model (vlastita izrada prema opisu iznad) 5
Slika 2. Komunikacija klijenta i poslužitelja kod SSL/TLS protokola [3, str. 11] 6
Slika 3. Primjer ClientHello poruke s ključnim podacima [1, str. 28] 8
Slika 4. Primjer „ServerHello“ poruke [1, str. 30] 9
Slika 5. Simetrična kriptografija 14
Slika 6. Asimetrična kriptografija 16
Slika 7. Struktura PKI infrastrukture 22
Slika 8. Prikaz glavne klase aplikacije 32
Slika 9. Prikaz implementacije čitača podataka 33
Slika 10. Prikaz pisača podataka za županije 33
Slika 11. Cron izrazi u batch aplikaciji 33
Slika 12. Prikaz postavki aplikacije 34
Slika 14. Pregled županija 36
Slika 15. Prikaz podataka za Grad Zagreb 36
Slika 16. Prikaz podataka o broju novo-zaraženih u posljednja 24 sata 37
Slika 18. Konfiguracija domene na strani Digitalocean platforme 38
Slika 19. Prikaz uspješne identifikacije certifikatom (Microsoft Edge preglednik) 39

Popis tablica

Tablica 1. XOR operator 12
Tablica 2. Prikaz XOR operacije 12
Tablica 3. Neki od simetričnih algoritmi 13
Tablica 4. Asimetrični algoritmi 15
Tablica 5. Prednosti i nedostaci simetričnog sustava kriptiranja 18
Tablica 6. Prednosti i nedostaci asimetričnog sustava kriptiranja 19