Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni diplomski studij matematike
Smjer: Matematika i računarstvo

Slaven Viljevac

Infrastruktura javnog ključa

Diplomski rad

Osijek, 2018.

Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni diplomski studij matematike
Smjer: Matematika i računarstvo

Slaven Viljevac

Infrastruktura javnog ključa

Diplomski rad
Mentor: izv. prof. dr. sc. Ivan Matić

Osijek, 2018.

Sadržaj

1 Uvod

Sigurnost na mreži danas je često spominjana tema. U ovom ćemo radu obraditi infrastrukturu javnog ključa koja se koristi u procesu manipulacije certifikatima kojima se predstavljamo na mreži. Certifikati su ti koji neku osobu predstavljaju na mreži i sadrže informacije koje ju identificiraju. Predstavit ćemo Secure Socket Layer ili SSL protokol koji se najčešće koristi u praksi i demonstrirati na primjeru kako on radi. U trećem poglavlju ovog rada obradit ćemo model povjerenja, model koji daje pravila koja određuju kako će se graditi put certifikata. Navesti ćemo primjere modela povjerenja i demonstrirati ih na primjerima. Na posljetku reći ćemo nešto o budućnosti PKI-ja i o kriptografiji baziranoj na identitetu.

2 Što je PKI

Da bismo uveli pojam PKI moramo prvo uvesti pojam certifikata[1]. Certifikat za neku osobu na mreži sadrži informacije koje tu osobu identificiraju na mreži (npr. njihovo ime, email adresa itd.), njihov javni ključ i potpis TA-a. TA (povjerljivi autoritet, od engl. thrusted authority) označava treću stranu koja potpisuje javne ključeve za sve osobe na mreži. Certifikati dakle služe korisnicima za provjeru autentičnosti na mreži. Kasnije ćemo u radu pobliže obraditi same certifikate.
Pretpostavimo npr. da Ivan želi od TA-a dobiti certifikat koji sadrži kopije Ivanovog javnog verifikacijskog ključa za potpis, tada se izvodi sljedeći protokol, no nećemo ulaziti u detalje kako se Ivan identificira TA-u, niti ćemo dati precizni format ID(Ivan), ni kako se javni i privatni ključevi dodaju Ivanu.

Protokol 1 (Izdavanje Certifikata za Ivana)**.**

1. *TA uspostavlja Ivanov identitet uobičajenim oblikom identifikacije, kao što je npr. putovnica. TA kreira niz znakova (string) u oznaci ID(Ivan), koji sadrži Ivanove identifikacijske informacije.*

2. *za Ivana se određuju privatni prijavni ključ u oznaci sig_{Ivan} i korespondirajući javni verifikacijski ključ u oznaci ver_{Ivan}.*

3. *TA generira svoj potpis kao:*

$$s = sig_{TA}(ID(Ivan) \mid\mid ver_{Ivan})$$

na Ivanovom identifikacijskom stringu i verifikacijskom ključu, pri čemu $\mid\mid$ označava konkatenaciju. Certifikat

$$\boldsymbol{Cert}(Ivan) = (ID(Ivan) \mid\mid ver_{Ivan} \mid\mid s)$$

je dodijeljen Ivanu, zajedno s njegovim privatnim ključem sig_{Ivan}.

Svako tko zna TA-ov verifikacijski ključ, ver_{TA} može verificirati certifikat ostalih. Pretpostavimo da se Marko želi uvjeriti da je Ivanov javni ključ autentičan. Ivan može svoj certitikat dati Marku, Marko može provjeriti potpis TA-a provjerivši da je

$$ver_{TA}(ID(Ivan) \mid\mid ver_{Ivan}, s) = true.$$

Sigurnost certifikata slijedi direktno iz sigurnost potpisa sheme koju koristi TA. Kako je već rečeno, svrha verifikacije certifikata je potvrditi autentičnost nečijeg javnog ključa. Certifikat sam po sebi ne pruža nikakav dokaz identiteta, jer sadrži samo javne informacije. Certifikat se može distribuirat ili redistribuirati bilo kome, no posjedovanje certifikata ne znači da smo i njegovi vlasnici.
Pogledajmo sada što je PKI (infrastruktura javnog ključa, od engl. Public-key Infrastructure).
Najveći izazov u kriptografiji javnog ključa je uvjeriti se u autentičnost samog javnog ključa. Kako smo vidjeli u prethodnom primjeru, ukoliko Ivan želi kriptirati informaciju i poslati je Marku, pri čemu se Ivan i Marko osobno ne poznaju, postavlja se pitanje kako zapravo

[1]Sam koncept certifikata uveo je Loren Kohnfelder 1978.

Ivan može biti siguran da je sadržaj javnog ključa zaista Markov ključ, a ne ključ neke treće osobe. Vidjeli smo da tu dolaze certifikati, kao alat koji pomaže u autentifikaciji javnog ključa. PKI je sigurnosni sustav koji se koristi za upravljanje i kontrolu certifikata.

Definicija 1. [2] *PKI je osnova sveobuhvatne sigurnosne infrastrukture čije se usluge implementiraju i isporučuju koristeći se konceptima i tehnikama javnog ključa.*

Postoji nekoliko gledišta na ovu definiciju.
Prvo je ideja da je PKI infrastruktura. Idejno, mora radit bez ikakve aktivne intervencije korisnika. Druga nasljeđena značajka PKI-ja je da koristi tehnike kriptografije javnog ključa od kojih je najbitnija potpisna shema[3]. Općenito se PKI sastoji od mnogo dijelova, pri čemu su najbitiniji:

- **Izdavanje certifikata**. Odnosi se na izdavanje novog certifikata korisniku s danim PKI-om. Većina PKI-a ima jedan ili više TA-ja koji se najčešće nazivaju CA (engl. certification authorities) i oni kontroliraju izdavanje certifikata. Prije nego je certifikat izdan, potrebno je potvrditi identitet i podatke korisnika nekriptografskim metodama. Tek nakon toga se korisniku izdaje certifikat. U tom trenutku potrebno je kreirati i javne i privatne ključeve za vlasnika certifikata. Nakon što je certifikat kreiran, potrebno ga je dostaviti korisniku sigurnim putem.

- **Ukidanje certifikata**. Odnosi se na ukidanje certifikata prije njegovog redovnog roka isteka koji je specifičan za svaki certifikat. Do ukidanja može doći zbog neke nepredviđene okolnosti, npr. ako je privatni ključ izgubljen ili ako dođe do zlouporabe ključa. Uzmimo npr. da je certifikat valjan do 13. siječnja 2020. no korisnikov korespondirajući privatni ključ biva ukraden prije datuma isteka certifikata. Tada se certifikat ukida i on prestaje biti valjan, te dolazi do izdavanja novog certifikata. (Ovaj primjer se može poistovjetiti s gubitkom bankovne kartice, gdje dolazi do izdavanja nove kartice u zamjenu za staru.). Dakle novi certifikat zamjenjuje ukradeni ključ s novim sigurnim ključem. Kako certifikat ne sadrži ništa što će reći da li je ukinut, potrebna je dodatna infrastruktura koja će prepoznati ukinuti certifikat.

- **Sigurnosna kopija / Oporavak / Ažuriranje ključeva**. Sigurnosna kopija ključa odnosi se na sigurnu pohranu privatnog ključa od strane administratora PKI-a u slučaju da korisnici izgube ili zaborave svoje privatne ključeve.
  Oporavak ključa je protokol kojim se omogućava da se izgubljeni ili zaboravljni ključ vrati ili ponovno aktivira. U tom slučaj korisnik mora dokazati svoj identitet prije nego mu se dopusti pristup spremljenom ključu.
  Do ažuriranja ključa može doći ukoliko se ključ mora promijeniti iz nekog razloga ili zbog sigurnosnih mjera. Npr. kada certifikat treba isteći, putem protokola se može odabrati novi zamjenski ključ i generirati novi certifikat koji će zamijeniti stari. Taj protokol za ažuriranje može iskoristiti stari ključ (prije nego istekne) da bi kriptirao novi, pa se ažurirani certifikat elektronski može dostaviti vlasniku.

[2] prema Carlisle Adams i Steve Lloyd

[3] varijanta potpisa u kojoj se svaki korisnik predstavlja sa svojim javnim ključem, pri čemu zadržava tajni ključ. Potpis poruke se može izračunati s tajnim ključem i verificirati javnim. Štoviše, ako je poruka potpisana od strane korisnika, nitko drugi ne može dati valjan potpis na tu poruku čak i ako se korisnika uvjeri da potpiše druge, moguće povezane poruke izabrane od druge strane.

- **Vremensko mapiranje**. Iz mnogih razloga vrijeme u kojem je ključ izdan, ukinut ili ažuriran može biti bitno. Npr. certifikati najčešće imaju fiksno vrijeme trajanja. Potpis na nekom podatku, bio on certifikat ili ne, koji uključuje specifično vrijeme ili period tokom kojeg je ključ valjan naziva se *vremensko mapiranje*.

Nakon što je PKI implementiran i funkcionalan, on omogućuje korisniku da na njega nadograđuje razne aplikacije. Te aplikacije ćemo nazivati *PKI-enabled services*. Navest ćemo nekoliko primjera PKI-enabled services koji sadrže sljedeće:

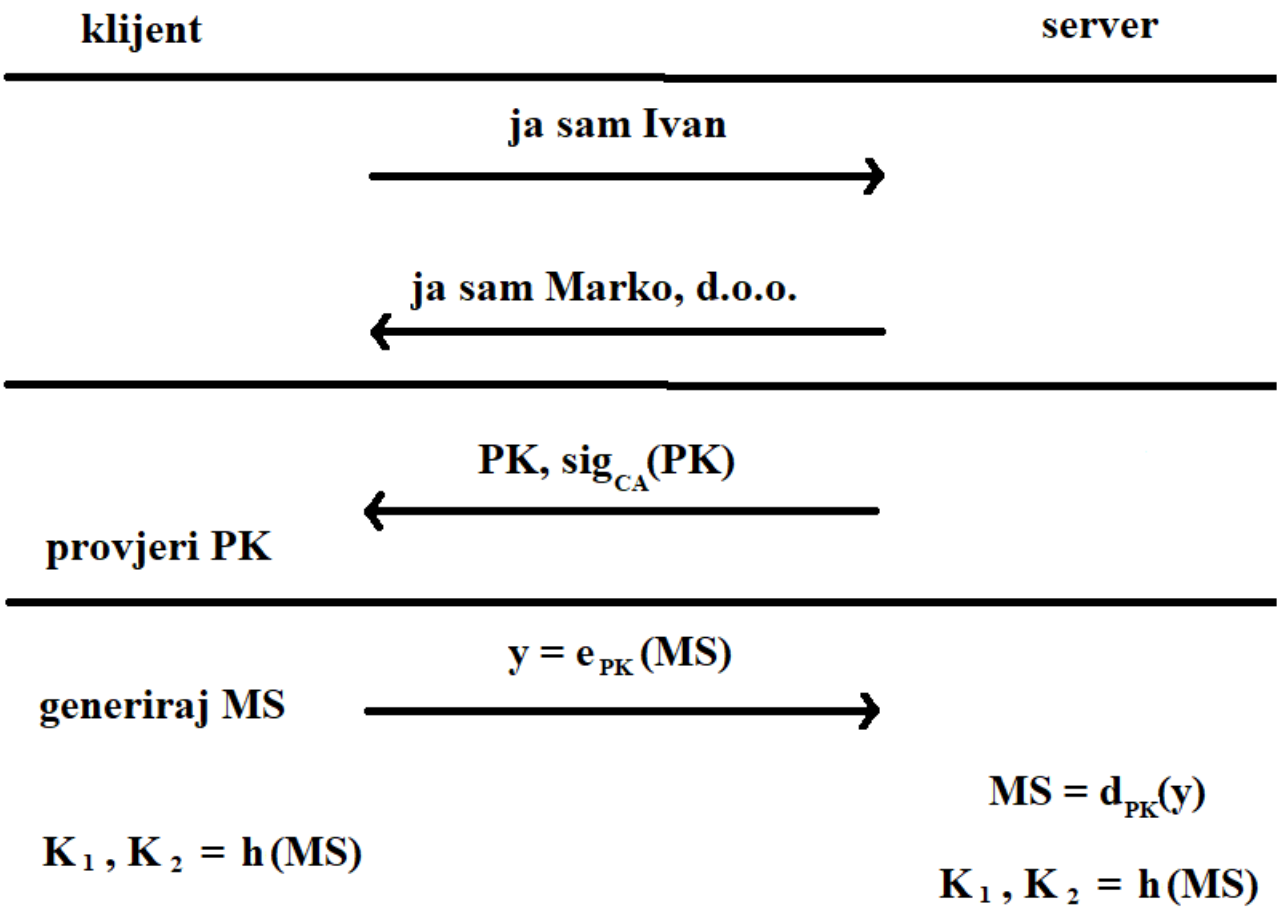
- **Sigurna komunikacija**. Neki od primjera koji se svakodnevno koriste. Secure e-mail protocols, sadrže Secure Multipurpose Internet Mail Extensions (S/MIME) i Pretty Good Privacy (PGP). Secure web service access dodjeljen putem Secure Socket Layer (SSL) ili Transport Layer Security (TLS). Secure virtual private networks (VPNs) koji koriste Internet Protocol Security (IPsec) protokole.

- **Kontrola pristupa**. Također poznata i kao *Upravljanje privilegijama*. Uključuje autentikaciju, autorizaciju i delegiranje.
  Primjer kontrole pristupa bi bio prijava na bazu podataka. Razne skupine ljudi mogu imati različite razine pristupa, bilo da imaju pristup samo dijelu informacija na bazi podataka ili njihov status u organizaciji zajedno s uvjetima pristupa podatcima određuje kojoj informaciji određena osoba ima pristup. Kontrola pristupa može uključivati neke oblike autentikacije korisnika, npr. putem lozinke ili kriptografske identifikacijske sheme.
  Delegacija se može koristiti u situacijama gdje postoji odobrenje za privremeni pristup bazi podataka od strane više rangirane osobe.

- **Arhitektura privatnosti**. Arhitektura privatnosti omogućava korištenje anonimnih/pseudoanonimnih kriptografskih alata. Takav tip certifikata može prikazivati članstvo pojedine osobe u određenoj klasi korisnika bez da specificira njihov identitet.

2.1 Praktični protokol: SSL ili Secure Socket Layer

Protokoli koji se koriste u praksi, u realnim aplikacijama mogu uključivati mnoge kriptografske alate. Mi ćemo se za ilustraciju zadržati na SSL-u. SSL sesije se npr. koriste prilikom online kupovina na stranicama kompanija putem web preglednika. Pretpostavimo da *klijent*, u našem slučaju to je Ivan, želi nešto kupiti od *servera* (Marko, d.o.o.). Osnovni korak je uspostaviti SSL sesiju, vidi Slika 2.1. Detaljnije, što je potrebno napraviti? Prvo, Ivan i "Marko, d.o.o." se predstavljaju jedan drugomu. To se naziva "pozdrav" i u tome koraku se ne koriste kriptografski alati. U tom trenutku Ivan i "Marko, d.o.o." se "dogovore" koji će se kriptografski algoritam koristiti u ostatku protokola.
Dalje "Marko, d.o.o." se dokazuje Ivanu; šalje Ivanu svoj certifikat koji sadrži kopiju njegova javnog ključa, u oznaci PK, potpisan od strane CA-ja (Certificate Authority-ja). Ivan provjerava CA-ov potpis na PK-ju koristeći CA-ov javni verifikacijski ključ (koji je pohranjen u web pregledniku koji je pokrenut na Ivanovom računalu).
Sada, Ivan i Marko, d.o.o. dolaze do određivanja dva zajednička tajna ključa. Ivan generira slučajnu tajnu, u oznaci MS (Master Secret), koristeći prikladan pseudoslučajni generator brojeva. Ivan šifriria MS koristeći javni ključ od Marko, d.o.o. i šalje rezultirajući šifrat Marku, d.o.o. Marko, d.o.o. dešifrira šifrat i pri tome dobiva MS. Sada Ivan i Marko, d.o.o. iz MS-a neovisno generiraju jednaka dva ključa, K_1 i K_2 (ovaj korak uključuje korištenje hash funkcija, u oznaci h).



Slika 2.1: Uspostava SSL Sesije

Na posljetku, Ivan i Marko, d.o.o. imaju oba jednaka izvedena tajna ključa. Te ključeve koriste za autentikaciju i kriptiranje poruka koje razmjenjuju. Ključ K_1 se koristi za autentikaciju podataka koristeći autentikacijsku poruku, dok se K_2 koristi za šifriranje i dešifriranje podataka koristeći kriptosustav tajnog ključa. Stoga *SSL* Protokol omogućava sigurnu komunikaciju između Ivana i Marko, d.o.o.
Primjetimo da je jedino server (Marko, d.o.o.) potreban kako bi se dostavio certifikat u *SSL* sesiji. Klijent, Ivan, uopće ne mora imati niti javni ključ niti certifikat. U današnje vrijeme to je zajedničko svim e-trgovinama. Kompanije postavljaju web stranice koje koriste za posao, koje zahtjevaju certifikate, ali korisnici ne trebaju certifikat kako bi obavili online-kupnju. S gledišta kompanija, nije bitno da je Ivan onaj kojim se predstavlja, bitnije je da je njegov broj bankovne kartice, koji se dostavlja kao dio osiguranja financijske transakcije ispravan. Broj bankovne kartice i osobni podatci dostavljeni od Ivana će biti šifrirani (i provjereni putem MAC-a) koristeći ključeve koji su kreirani u *SSL* sesiji.

2.2 Certifikati

Kako smo već u samom početku naveli, certifikati sadrže informacije koje neku osobu identificiraju na mreži (npr. njihovo ime, email adresa itd.), njihov javni ključ i potpis TA-a.
Certifikati su građevni dijelovi PKI-a i u konačnici oni su ti koji omogućju da se izgradi siguran i skalabilan PKI. U najjednostavnijem obliku, certifikat veže identitet u javni ključ. To

se najčešće radi tako da TA potpisuje informaciju na certifikatu kako je opisano u Protokolu 1. U nastavku ćemo za TA koristiti CA (engl. certification authoritiy). Generalno pretpostavimo da svatko ima pristup autentičnoj kopiji javnog ključa CA-a. Stoga se CA-ov potpis može provjeriti, što omogućava potvrdu informacije na certifikatu. Generalno govoreći, potvrditi CA-ov potpis, omogućava samo da se provjeri da je certifikat izdan od strane CA-a. Iako, ako potvrdimo CA-ov potpis, korisnik će vjerovati da je inforamcija koju certifikat sadrži ispravno dodjeljena i da vjeruje CA-u da je provjerio informaciju prije potpisivanja certifikata.
Nadalje, u svrhu ilustracije, opisat ćemo $X.509$ v3 format certifikata. $X.509$ certifikat sadrži sljedeće:

1. broj verzije
2. serijski broj
3. ID algoritma potpisa
4. ime izdavača
5. period ispravnosti
6. ime subjekta (npr. vlasnik certifikata)
7. javni ključ vlasnika certifikata
8. opcionalno polje
9. potpis CA-ja na svim prethodnim poljima

$X.509$ *certifikati* su originalno definirani koristeći $X.500$ *imena* za imena subjekata. $X.500$ *imena* ima hijerarhijski format kao npr.:

C	=	HR
O	=	Sveučilište u Zagrebu Sveučilišni računski centar HR34016189309
L	=	Zagreb
CN	=	Status studenta,

gdje "C" označava državu, "O" označava organizaciju, "L" označava mjesto, "CN" označava zajedničko ime. Dodatno možemo imati "S" koji označava regiju (npr. županija) i "OU" koji označava dio organizacije. Imena subjekata su šifrirana koristeći numeričke oznake, koje ćemo označiti sa OID (*engl. object identifier*). Dakle, umjesto da se koristi string "Sveučilište u Zagrebu Sveučilišni računski centar", certifikat sadrži numerički OID (34016189309) koji je zamjena za "Sveučilište u Zagrebu Sveučilišni računski centar".
Hijerarhijski format $X.500$ *imena* osigurava da globalno svatko ima jedinstveno ime. Namjera je da $X.500$ postane pravi imenik, analogon globalno poznatom telefonskom imeniku, koji bi omogućio $X.509$ certifikatima da se mogu pregledati i da im se može pristupiti na daljinu. No nažalost, dan danas široko rasprostranjen $X.500$ imenik ne postoji.
Drugi problem je u tome što $X.500$ *imena* ima drugačiji format i strukturu od *DNS*-a (engl. domain name system) i *IP* adrese (engl. internet protocol). Npr. e-mail adresa je

$$sviljeva@mathos.hr,$$

pri čemu je

mathos.hr

domena. IP adresa je (hijerarhijski) numerički kod koji omogućava internetskom softwareu da pristupi određenoj domeni. U bilo kojem lokalnom okruženju, DNS mapira imena računala u IP adrese. Na primjer, domena

mathos.hr

ima IP adresu

193.198.69.3

Većina ljudi je upoznata te koristi imena i e-mail adrese u formatu kakav je prethodno opisan i nije upoznata sa formatom $X.500$ *imena*. Prema tome, bilo kakav pokušaj za prelazak na korištenje $X.500$ *imena* povukao bi pitanje na koji način prijeći iz jednog formata (ili sustava imenovanja) na drugi.
Postoje i brojni drugi formati certifikata iako niti jedan nije prihvaćen kao univerzalan.
Neki primjeri sadrže sljedeće:

- **SPKI**. SPKI (engl. simple public key infrastructure) certifikati koriste lokalna imena (za razliku od globalnih). Stoga, oni naglašavaju autorizaciju više od identiteta. SPKI certifikati nisu kompatibilni s $X.509$ certifikatima.
- **PGP**. PGP (engl. Pretty Good Privacy) je korisnički sustav e-maila baziran na lokalnim imenima. PGP certifikati nisu kompatiblini s $X.509$ certifikatima.
- **SET**. SET (engl. Secure Electronic Transaction) specifikacije koriste napredne (modificirane) $X.509$ certifikate.

2.2.1 Upravljanje životnim vijekom certitfikata

Upravljanje životnim vijekom certifikata se sastoji od nekoliko faza, a to su:

1. registracija
2. generiranje i distribucija ključa
3. sigurnosna kopija ključa
4. izdavanje certifikata
5. potvrda certifikata
6. ažuriranje ključa
7. oporavak ključa
8. ukidanje certifikata
9. istek certifikata
10. povijest ključeva
11. arhiva ključeva.

U ovom dijelu, posvetit ćemo se metodama za potvrđivanje certifikata. U većini slučajeva, potvrđivanje certifikata uključuje operacije koje su opisane u sljedećem algoritmu.

Algoritam 1.

1. *Provjeri integritet i autentičnost certifikata, tako da provjeriš potpis CA-ja na certifikatu.(Pretpostavljamo da je verifikacijski ključ CA ili unaprijed poznat, ili mu vjerujemo ili da je već provjeren, npr. da smo ga provjerili na službenim web stranicama).*

2. *Provjeri da certifikat nije istekao (provjeri datum isteka certifikata).*

3. *Provjeri da certifikat nije ukinut.*

4. *Ako je relevantan, provjeri da je korištenje certifikata u skladu s uvjetima specificiranim u opcionalnim poljima certifikata.*

Primijetimo da samo prvi korak gore spomenutog algoritma koristi kriptografiju. Ukratko ćemo samo prokomentirati popularnije tehnike koje se koriste da bismo se osigurali da certifikat nije ukinut.
Prisjetimo se prije svega kako je ukidanje certifikata potrebno ukoliko certifikat postane beskoristan prije nego mu je istekao rok uporabe. PKI, dakle treba mehanizam kako bi provjerio da certifiakt nije ukinut. Najučestalija tehnika je *CRL* (engl. certificate revocation list), što je lista svih serijskih brojeva certifikata koji su ukinuti, ali nisu istekli. *CRL* je pripremljen i potpisan od strane *CA*-a. *CRL* se periodički treba osvježavati, te mora biti javno dostupan kako bi bio koristan.
Umjesto da se stalno izdaju novi *CRL*-ovi, učinkovitija metoda jest uvođenje *Delta CRL*-ova. *Delta CRL* sadrži promjene koje su se pojavile na prethodno izdanim *CRL*-ovima ili *Delta CRL*-ovima. Npr. *CRL* se može izdavati jednom mjesečno, s *Delta CRL*-ovima koji su izdani na dnevnoj ili tjednoj bazi.
Alternativni pristup *CRL*-u je *OCRL* (engl. online certificate status protocol), u kojem je određeni server koji se brine o ukidanju certifikata. Taj server mora biti u mogućnosti održavati, ili biti u mogućnosti prstupiti osvježenom *CRL*-u u stvarnom vremenu kako bi pravovremeno mogao procesuirati upite koji su mu zadani.

3 Modeli povjerenja

Certifikati najčešće nisu potpisani direktno od strane *CA*-a kojem vjerujemo. Umjesto toga, potrebno je pratiti *put certifikata* od *CA*-a kojem vjerujemo do izdanog certifikata. Svaki certifikat u tom putu bi trebao biti potpisan od strane posjednika prethodnog certifikata u tom putu. Kako bi korisnik bio siguran da je certifikat ispravan, mora se moći potvrditi ispravnost svih certifikata na tom putu.
Model povjerenja daje jasna pravila koja određuju na koji način će se graditi put certifikata. Sada ćemo navesti i opisati neke primjere modela povjerenja:

1. Strogo hijerarhijski model
2. Umreženi PKI
3. Model mrežnog preglednika
4. Mreža povjerenja

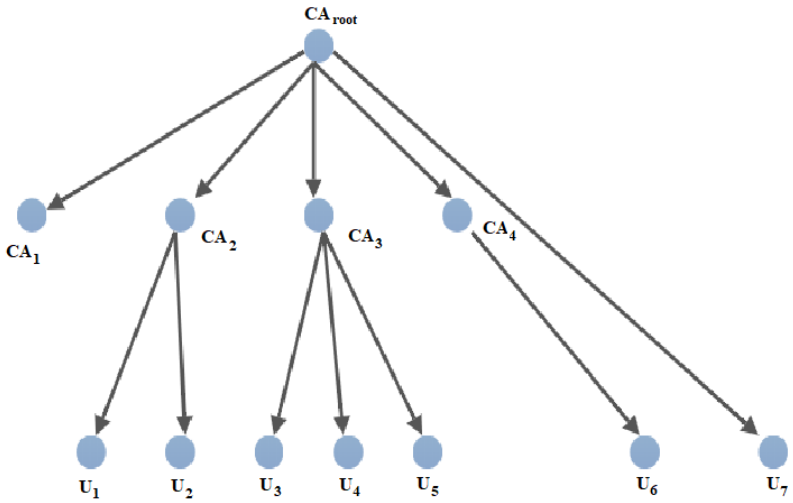
3.1 Strogo hijerarhijski model

U ovom modelu, korjenski *CA* sadrži samopotpisni i samoizdani certifikat. Korjenski *CA* se naziva još i "sidro povjerenja". Korjenski *CA* može izdati certifikate za niže rangirane *CA*-ove, i bilo koji *CA* može izdati certifikat krajnjem korisniku.
Pogledajmo sada strogo hijerarhijski model na primjeru. Model ilustriran u ovom primjeru (vidjeti Sliku 3.1) ima korijenski *CA*, četiri niže rangirana *CA*-a i sedam krajnjih korisnika. Primjetimo da imamo usmjereni graf gdje *CA*-ovi i krajnji korisnici čine čvorove. Usmjereni bridovi od x prema y označavaju da čvor x potpisuje certifikat za čvor y. Dodatno primjetimo da svaki od krajnjih korisnika ima certifikat koji je potpisan od strane nekog od pet *CA*-ova.
Ukoliko sada, recimo Ivan, želi provjeriti certifikat nekog od korisnika, recimo Marka, Ivan treba provjeriti cijeli put certifikata, tj. sve certifikate koji se nalaze na putu od korjenskog *CA*-a do Markovog. Uzmimo primjerice da CA_{root} potpisuje certifikat za CA_1, pri čemu se pretpostavlja da CA_1 potpisuje Markov certifikat, tada se to može smatrati jednim putem certifikata. U strogo hijerarhijskom modelu od Marka se može očekivati da sprema informacije na svom računalu. Tada ih Marko može proslijediti Ivanu ili bilo kome drugome tko to zatraži.
U navedenom primjeru, pretpostavlja se da Marko Ivanu šalje sve certifikate koji se nalaze na putu certifikata, dakle možemo zapisati:

$$CA_{root} \rightarrow CA_1 \rightarrow Marko.$$

Pretpostavlja se da Ivan zna verifikacijski ključ od CA_{root}, označimo ga s $ver_{CA_{root}}$. Zajedno s putem certifikata koje dobiva od Marka, Ivanu se omogućava da dobije valjan put certifikata. Dani put certifikata je valjan ako vrijedi sljedeće:

1. Ivan potvrđuje $\mathbf{Cert}(CA_{root})$ koristeći ključ $ver_{CA_{root}}$
2. Ivan potvrđuje $\mathbf{Cert}(CA_1)$ koristeći ključ $ver_{CA_{root}}$
3. Ivan dobiva ključ ver_{CA_1} iz $\mathbf{Cert}(CA_1)$
4. Ivan potvrđuje $\mathbf{Cert}(Marko)$ koristeći ključ ver_{CA_1}



Slika 3.1: Strogo hijerarhijski model

5. Ivan dobiva Markov javni ključ iz **Cert**$(Marko)$.

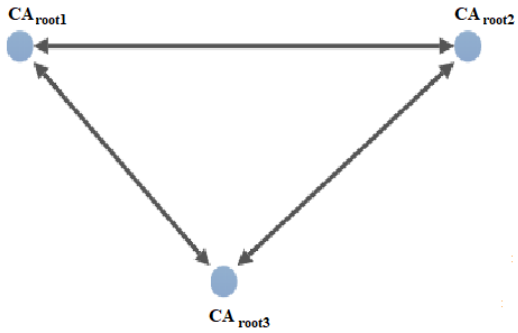
Na posljetku, Ivan se mora uvjeriti da put pristaje povjerljivom modelu, tj. da put započinje u CA_{root} i završava kod Marka, ili da eventualno sadrži neki međučvor, koji označava CA nižeg ranga. Uočimo da svaki certifikat treba sadržavati informaciju o statusu vlasnika certifikata i krajnjeg korisnika.
U ovom modelu povjerenja, krajnjem korisniku nije omogućeno da potpisuje cetifikat koji pripada nekom drugom krajnjem korisniku. Ako korisnik ipak kreira takav certifikat, tada se taj certifikat neće smatrati ispravnim jer put certifikata neće odgovarati modelu povjerenja, npr. ako Matea potpiše Markov certifikat, imamo sljedeći put:

$$CA_{root} \rightarrow Matea \rightarrow Marko.$$

Primjetimo da taj put nije dobar jer Ivan ne može potvrditi Markov certifikat koristeći taj put.

3.2 Umreženi PKI-jevi

Prethodno spomenuti model se pokazuje dobrim u jednostavnoj organizaciji, no ponekad postoji potreba da se korjenski CA poveže s dvije ili više različitih PKI domena i tada se taj postupak naziva umrežavanje PKI-jeva. Tim postupkom dobivamo tzv. "super-PKI" koji se sastoji od korisnika u različitim domenama. Same domene ne moraju nužno biti hijerarhijske, pa tako možemo imati *unakrsno cetificiranje* u slučaju da jedan CA potpisuje certifikat drugog CA. U *mrežnoj konfiguraciji* će svi korjenski certifikati certificirati jedni druge. Prema tome, n korjenskih CA-ova u mrežnoj konfiguraciji trebat će $n(n-1)$ unakrsnih-certifikata. Na Slici 3.2 možemo vidjeti primjer s tri korjenska CA-ja i usmjerenim bridovima koji prikazuju unakrsno certificiranje. Alternativni pristup je *konfiguracija središta i končanica* u kojoj se svaki od n korjenskih CA-ova neovisno unakrsno certificira s novim središnjim CA-om. Broj potrebnih međusobnih certificiranja je jednak $2n$. Primijetimo da je za $n > 3$ u



Slika 3.2: Mrežna konfiguracija

usporedbi s mrežnom konfiguracijom potrebno više međusobnog certificiranja.
Pogledajmo nadalje kako će se put certificiranja ponašati u ove dvije konfiguracije. Prvo ćemo se osvrnuti na mrežnu konfiguraciju.
Kako bi potvrdio Markov certifikat, Ivan mora moći pronaći put certifikata od svog do Markovog "sidra povjerenja" i taj postupak ćemo nazvati *pronalazak puta*. Pretpostavimo da su Ivanovo i Markovo "sidro povjerenja" redom CA_{root_i} i CA_{root_j} pri čemu je $i \neq j$.
Pretpostavimo da Marko Ivanu šalje certifikate koji se nalaze u putu od CA_{root_j} do Marka, prema tome, Ivan mora pronaći certifikat koji pripada CA_{root_j} (koji označava Markovo sidro povjerenja) i potpisan je od strane CA_{root_i} (što je Ivanovo sidro povjerenja). U idealnom slučaju, certifikat i se mogao pronaći u imeniku koji se održava od strane CA_{root_i}.
Sada, kada Ivan ima sve certifikate sa puta od CA_{root_i} do Marka, Ivan može potvrditi Markov certifikat.
Promotrimo sada konfiguraciju središta i konačnica. Ponovno ćemo pretpostaviti da Marko Ivanu šalje certifikate u putu od CA_{root_j} do Marka. Ivan sada treba pronaći put certificiranja od CA_{root_i} do CA_{hub} do CA_{root_j}. To se može napraviti na sljedeći način:

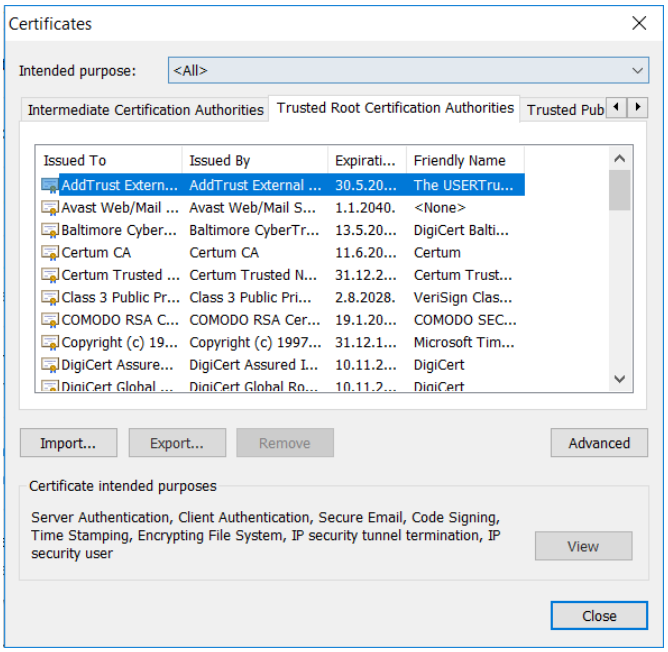
- Prvo Ivan od CA_{root_i} dobiva kopiju certifikata CA_{hub}-a koji potpisani od strane CA_{root_i}.
- Tada Ivan potraži certifikat CA_{root_j} koji je potpisan od strane CA_{hub}-a u imeniku kojeg održava CA_{hub}.

Sada Ivan ima sve certifikate koji se nalaze na putu od CA_{root_i} do Marka, što mu omogućava da potvrdi Markov certifikat.

3.3 Model mrežnog preglednika

Model mrežnog preglednika se malo razlikuje od prethodno spomenutih modela. Većina preglednika (npr. Internet Explorer ili Chrome) dolazi sa unaprijed postavljenim skupom neovisnih korijenskih *CA*-ova, koje korisnik preglednika koristi kao sidra povjerenja. Npr. kod preglednika *Chrome* do popisa certifikata dolazimo na sljedeći način (vidjeti Sliku 3.3):

$$chrome://settings/$$



Slika 3.3: Trusted Root Certification Authorities

te u naprednom izborniku odabiremo *Manage certificates.*

Dobivamo popis korijenskih *CA*-ova kojima vjerujemo i među tim korijenskim *CA*-ovima ne postoji međusobno certificiranje. Konceptualno model povjerenja se može promatrati kao jedna stroga hijerarhija sa "virtualnim" korijenom *CA* (ovdje bi to bio preglednik sam za sebe). Na posljetku, korisnik vjeruje da je davatelj usluge za mrežni preglednik u sam preglednik uključio i ispravni korijenski *CA*.

U modelu internetskog preglednika postoji i nekoliko sigurnosnih problema. Prvi je da korisnik ne mora imati nikakve informacije o sigurnosit predkonfiguriranih korijenskih *CA*-ova. Korisnik sam može uređivati listu certifikata, no za to su mu potrebna određena znanja. Ne postoje mehanizmi koji će ukinuti korijenski *CA* za mrežni preglednik. Ne postoji veza ili ugovor između korisnika i skupa *CA*-ova koji su postavljeni na mrežnom pregledniku. Ako certifikat nije potpisan od poznatog *CA*, korisnik ima mogućnost prihvatiti ga kroz skočni prozor u SSL sesiji. Međutim treba zapamtiti da ukoliko provjerimo certifikat korijenskog *CA* to ne znači da je certifikat autentičan, jer kako je već bilo rečeno, certifikat je samopotpisan. Drugi problem se javlja kod datuma isteka. Kako ne postoji automatizirani mehanizam koji će osvježavati certifikate korijenskih *CA*-ova korisnicima se prilikom uspostave SSL sesije zna pojaviti poruka da je certifikat istekao. Pri tome se korisniku daje mogućnost da prihvati certifikat iako je on istekao. Prirodna reakcija na ovo je da prihvaćamo certifikat iako je on istekao jer je to jedina alternativa prekidu željene SSL sesije. U praksi ovo je upitno i dovodi do pitanja same svrhe certifikata.

3.4 Prilično dobra privatnost

Prilično dobra privatnost *PGP* (od engl. Pretty Good Privacy) koristi se u sustavima e-maila, gdje svaki korisnik ima vlastiti *CA*. PGP certifikati sadrže e-mail adresu (ID), javni ključ (PK) i jedan ili više potpisa na paru (ID, PK).

Pogledajmo primjer u kojem Ivan može kreirati samopotpisni certifikat

$$\mathbf{Cert}(Ivan) = (\mathbf{podatak}, \mathbf{potpisi})$$

koji sadrži sljedeće informacije:

$$\textbf{podatak} = (ID = ivan@mathos.hr, PK = 12345)$$
$$\textbf{potpisi} = sig_{Ivan}(\textbf{podatak}).$$

Algoritam 2 (Računanje legitimiteta područja ključa)**.**

1. *KLF (od engl. Key Legitimacy Field) za korisnika U poprima vrijednost "ispravan" ukoliko je podatak korisnika U potpisan barem jednim ključem za kojeg OTF (od engl. Owner Trust Field) ima vrijednost "pouzdan" (implicitno ili u potpunosti), ili barem dva ključa za koje OTF ima vrijednost "djelomično pouzdan".*

2. *KLF poprima vrijednost "krajnje ispravan" ako su korisnikovi podatci potpisani barem jednim ključem za koje OTF ima vrijednost "djelomično pouzdan".*

3. *Inače, KLF-u je pridružena vrijednost "neispravan."*

Nadalje, drugi korisnici također mogu dodavati potpise na podatke Ivanovog certifikata, stoga npr. ako Marko da Ivanu takav potpis, Ivan će ga dodati na listu potpisa svog certifikata:

$$\textbf{potpisi} = (sig_{Ivan}(\textbf{podatak}), sig_{Marko}(\textbf{podatak})).$$

Vidjet ćemo da potpisi certifikata pomažu prilikom provjere autentičnosti certifikata kod drugi korisnika.
U skupu podataka koji se naziva *Prsten ključeva* Ivan posjeduje kolekciju certifikata koje je prikupio na raznim izvorima. Svaki certifikat u prstenu ključeva povezan je s KLF-om i OTF-om. Štoviše, KLF će dati potvrdu da li je određeni javni ključ zapravo važeći za Ivana. Kako je već spomenuto u samom algoritmu, KLF ima tri mogućnosti: "ispravan", "krajnje ispravan" i "neispravan." Treba napomenuti da vrijednost "neispravan" ne znači da je dobiveni ključ neispravan, nego da ne postoji dovoljno dokaza koji će pokazati da je ključ valjan. KLF se računa preko PGP-a koristeći određene metode koje ćemo navesti kasnije.
Vrijednost OTF-a određuje sam Ivan prema svom osobnom vjerovanju, što proizlazi iz Ivanovog povjerenja prema vlasniku ključa koji potpisuje ostale ključeve. Kao i KLF, OTF također poprima određene vrijednosti, a to su: "implicitno povjerenje", "potpuno povjerenje", "djelomično povjerenje" ili "nepovjerenje."
Primjerice Ivanov OTF ima vrijednost "implicitno povjerenje." Ako Ivan postavi vrijednost Markovog OTF-a na "potpuno povjerenje" u svom prstenu ključeva, tada Ivan vjeruje da:

1. Marko javni ključ je valjan i

2. Marko je pojedinac koji je dovoljno oprezan da ne potpisuje bilo kakav neispravan uređeni par (ID, PK).

Jednom kada su vrijednosti za OTF postavljene, možemo izračunati vrijednosti za KLF i te vrijednosti računamo pomoću PGP-jevog softwarea koristeći Algoritam 2.
Pogledajmo sad primjer računanja vrijednosti KLF-a.

Primjer 1. *Pretpostavimo da Ivanov prsten ključeva sadrži podatke koji su dani u Tablica 1. Korisnik može provjeriti da se KLF-ovi određuju kako slijedi:*

- *Kako je Ivan potisao Markove i Brankine certifikate i Ivan implicitno vjeruje vlastitim potpisima, Ivan Markove i Brankine ključeve smatra ispravnima.*

vlasnik	OTF	potpisi
Ivan	implicitno	Ivan, Marko, Martina, Antonio
Marko	potpuno	Ivan, Marko, Antonio, Milan
Matea	djelomično	Marko, Matea, Antonio
Martina	nepovjerenje	Martina, Antonio, Branka
Antonio	nepovjerenje	Martina, Antonio, Boris
Boris	djelomično	Marko, Antonio, Boris
Milan	djelomično	Matea, Antonio, Boris, Milan
Danilo	nepovjerenje	Antonio, Danilo, Damir
Damir	nepovjerenje	Antonio, Boris, Danilo, Damir
Branka	potpuno	Ivan, Marko, Antonio, Branka

Tablica 1: Ivanov prsten ključeva

- *Kako je Marko potpisao Mateine i Borisove certifikate i Ivan potpuno vjeruje Markovim potpisima, Ivan smatra Mateine i Borisove ključeve ispravnima.*
- *Kako je Branka potpisala Martinin certifikat i Ivan u potpunosti vjeruje Brankinim potpisima, Ivan smatra Martinin ključ ispravnim.*
- *Kako su i Matea i Boris potpisali Milanov certifikat i kako Ivan djelomično vjeruje i Matei i Borisu, Ivan smatra Milanov ključ ispravnim.*
- *Antonio i Damir imaju jedan potpis od nekoga komu Ivan djelomično vjeruje, te stoga Ivan te ključeve smatra krajnje ispravnim.*
- *Danilo nema potpisa od nikoga komu Ivan barem djelomično vjeruje, te stoga Ivan Danilove ključeve smatra nevažećima.* □

Postoji nekoliko problema sa PGP oblikom PKI. Prije svega PGP nije prava infrastruktura. Korisnici sami moraju postaviti razine povjerenja i opozivanje se može napraviti jedino od strane vlasnika certifikata. Nadalje, ne postoje mehanizmi koji će opozvati "prividne" certifikate, jer svatko može napraviti falsificirani certifikat i ne postoji osiguranje da će falsifikati biti otkriveni. Drugi problem PGP-ja jest što nije lako skalabilan za velike mreže ili za zajednice korisnika, najbolje radi na "lokalnim" zajednicama gdje se većina korisnika poznaje. Naposljetku, teško je provoditi jedinstvena pravila ili standard, pa PGP oblik PKI-ja nije pogodan za velike organizacije.

4 Budućnost PKI-ja

Kada pogledamo na praktičnu uporabu PKI-ja, pojavljuju se mnoge potencijalne poteškoće. Prvi i najbanalniji problem jest razvoj, održavanje i regulacija PKI-ja, nije razjašnjeno treba li za njegovu administraciju biti zadužena vlada ili industrija.
Drugi problem je koji pak standardi trebaju biti korišteni u PKI-ju. Problemi vezani za standardizaciju uključuju format certifikata, kriptografske alate, opozivanje, otkrivanje puteva i mnoge druge.
Treći problem se krije u tome da razlikovanje PKI-ja zahtjeva različita okruženja. PKI koji se koristi u centraliziranim kompanijama se razlikuje od PKI-ja koji osiguravaju komunikaciju u široko disperznim grupama koje pripadaju zajedničkoj organizaciji. Neki PKI-jevi zahtijevaju jedino certifikate kako bi potvrdili korisnikove javne ključeve, dok se drugi mogu koristiti prvenstveno za kontrolu pristupnih privilegija.
Naposljetku, manjak aplikacija koje su kompatibilne s PKI-jem usporavaju razvoj PKI-jeva. Sami developeri ne žele razvijati aplikacije kompatibilne s PKI-jem dok se PKI dovoljno ne razvije, dok s druge strane PKI se neće razviti dovoljno dok ne bude dovoljno aplikacija koje su izgrađene koristeći mogućnosti PKI-ja.
Teško je predvidjeti budućnosti PKI-ja, ali čini se razumnim fokusirati se na skromna očekivanja i očekivanja koja je moguće postići, tj. prepoznati da je prvotna svrha PKI-ja upravljanje ključevima i pokušati pronaći način da se prvo riješi taj problem.

4.1 Alternative za PKI

Kako smo u prethodnom poglavlju vidjeli da razvoj PKI-ja ide sporo, realno je za očekivati da se postavi pitanje koje alternative za PKI uopće postoje.
Ako se PKI-jevi neće koristiti za provjeru javnih ključeva, koje će se tehnike za to koristiti umjeto PKI-ja. Ako u potpunosti izbacimo kriptografiju javnog ključa, vraćamo se u 20. stoljeće kada se koristio server mrežnih ključeva koji je implementiran koristeći kriptografiju tajnog ključa, no takav sustav u današnjem svijetu interneta nebi funkcionirao.
Trenutna infrastruktura koja se koristi može se nazvati "ad hoc" infrastrukturom. U današnje vrijeme imamo široko korištenje certifikata potpisanih od različitih CA iako mnogi certifikati koji se danas koriste su ili istekli ili su potpisani od CA koji nisu poznati krajnjim korisnicima. Zapravo je teško ocijeniti stvarnu vrijednost infrastrukture koja se stalno koristi na način koji nije u skladu s onima za što je namijenjena. Povucimo paralelu sa sigurnosnim pojasom u automobilu, sam pojas je beskorisan ako putnici u automobilu nisu vezani.
Druga alternativa je kriptografija bazirana na identitetu koja podsjeća na kriptografiju javnog ključa, budući da se vrijednosti javih ključeva mogu izračunati iz identiteta njihovih vlasnika. Kriptografiji baziranoj na identitetu certifikati su nepotrebni, prema tome ne postoji ni potreba za infrastrukturom koja provjerava javne ključeve.

5 Kriptografija bazirana na identitetu

Osnovna ideja ovog koncepta jest da se javni ključ korisnika U dobiva primjenom javne hash funkcije h na korisnikov ID, tj. $ID(U)$. Korespondirajući privatni ključ će se generirati od strane središnjeg TA-a. Taj privatni ključ dostavit će se korisniku U nakon što sam korisnik svoj identitet dokaže TA-u. Izdavanje privatnog ključa od strane TA-a zamjenjuje izdavanje certifikata. Rezultirajući javni i privatni ključ se dalje koriste za enkripcijsku, potpisnu ili neku drugu kriptografsku shemu. Shema uključuje određene javne sistemske parametre (uključujući "master ključ[4]") koji svi koriste.
Treba primjetiti da kriptografija bazirana na identitietu uklanja potrebu za certifikatima. Iako nam je i dalje potrebna pouzdana i provjerena metoda koja povezuje osobu s ID-jem. Problemi imanovanja koji postoje u PKI-u, a koji su spomenuti u potpogavlju 2.2 korištenjem kriptosustava baziranom na identitetu nisu riješeni. Treba se pozabaivit i s rješavanjem drugih problema vezanih za PKI, kao npr. opozivanje, gdje problem nastaje ako se u kriptosustavu baziranom na identitetu e-mail adresa koristi kao ID, te ukoliko dođe do promjene e-mail adrese.
Dizajniranje kriptosustava baziranom na identitetu nije jednostavno, nažalost čini se da nema jasnog puta kako prebaciti proizvoljan kriptosustav javnog ključa u kriptosustav baziran na identitetu. Pokažimo to na primjeru, pretpostavimo naivno da želimo transformirati RSA kriptosustav u kriptosustav baziran na identitetu. Možemo pretpostaviti da TA odabire RSA modulo $n = pq$, pri čemu je n master ključ. Faktori p i q ne bi bili poznati nikome osim TA-a i oni bi imali ulogu master privatnog ključa.
RSA javni ključ korisnika U je enkripcijski eksponent, dok je privatni ključ dekripcijski eksponent. Jednom kada U ima javni ključ i korespondirajući privatni ključ, tada korisnik jednostavno može faktorizirati n. Kada korisnik U zna privatni master ključ, on može "utjeloviti" TA i izdavati javne ključeve za bilo koga, baš kao što može i izračunati bilo čiji privatni ključ. Prema tome ova metoda kreiranja kriptosustava baziranog na identitetu ne funkcionira.
Kako možemo vidjeti iz prethodnog primjera kriptografija bazirana na identitetu zahtjeva osmišljavanje sustava gdje korisnikov javni i privatni ključ ne mogu biti iskorišteni za određivanje TA-evog privatnog master ključa.
Pogledajmo sada detaljan opis potrebnih operacija i kako treba raditi u enkripcijska shema (*javnog ključa*) bazirana na identitetu.

- **Generiranje master ključa.** TA kreira javni i korespondirajući privatni master ključ u oznakama M^{pub} i M^{priv}. Master ključ je oblika $M = (M^{pub}, M^{priv})$. Hash funkcija je također javna, te master ključ i hash funkcija čine parametre sustava.

- **Generiranje ključa korisnika.** Kada se korisnik U identificira TA-ju, TA tada koristi funkciju **extract** da izračuna korisnikov privatni ključ K_U^{priv} kako slijedi:

  $$K_U^{priv} = \textbf{extract}(M, K_U^{priv})$$

  pri čemu je korisnikov javni ključ

  $$K_U^{pub} = h(ID(U)).$$

  Ključ korisnika U je oblika $K_U = (K_U^{pub}, K_U^{priv})$.

[4]U kriptografiji baziranoj na identitietu javni ključ može biti bilo koji proizvoljan znak koji je u vezi s korisnikovim identitetom, privatni ključ je izveden iz iz identiteta entiteta, dok je **master ključ** poznat samo KGC-u (od engl. Key Generation Center). KGC je uz CA jedan od izdavača certifikata.

- **Šifriranje.** Javni ključ K_U^{pub} korisnika U definira javno pravilo šifriranja e_{K_U} koje svatko može koristiti kako bi šifrirao poruke koje su poslane korisniku U.
- **Dešifriranje.** Privatni ključ K_U^{priv} korisnika U definira pravilo dešifriranja d_{K_U} koje će korisnik U korisiti da dešifira poruke koje dobije.

5.1 Cocksova enkripcijska shema bazirana na identitetu

Cocksov kriptosustav baziran na identitetu dan je s Kriptosustavom 1.
Prije nego se posvetimo Cocksovom kriptosustavu uvedimo sljedeće skupove koje ćemo kasnije koristiti. Sama shema algoritma se bazira na aritmetici u $\mathbb{Z}_n$, pri čemu je $n = pq$ gdje su p i q međusobno različiti prosti brojevi kongruentni 3 modulo 4. $\mathbf{QR}(n)$ označava skup kvadratnih ostataka modulo n :

$$\mathbf{QR}(n) = \left\{ x \in \mathbb{Z}_n : \left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = 1 \right\}.$$

$\widetilde{\mathbf{QR}}(n)$ označava pseudokvadrate modulo n :

$$\widetilde{\mathbf{QR}}(n) = \left\{ x \in \mathbb{Z}_n : \left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = -1 \right\}.$$

Sigurnost sheme ovisi o težini problema određivanja **složenih kvadratnih ostataka** Problem 1 u $\mathbb{Z}_n$ što ćemo promatrati kasnije.
Kriptosustav 1 ovisi o određenim svojstvima Jakobijevog simbola. Postavke koje se koriste jednake su postavkama Algoritma 3, tj. *Blum-Blum-Shub* generatoru. Više o tom generatoru može se pronaći u [1], potpoglavlje 3.8.

Problem 1 (Složeni kvadratni ostatci)**.**

- ***Instanca:*** *Pozitivan cijeli broj n koji je produkt dva nepoznata različita cijela broja p i q te cijeli broj $x \in \mathbb{Z}_n^*$ takav da je $(\frac{x}{n})$ jednak 1, pri čemu je $\mathbb{Z}_n^* = \mathbb{Z}_n \setminus \{0\}$*
- ***Pitanje:*** *Je li $x \in \boldsymbol{QR}(n)$?*

Problem složenih kvadratnih ostataka u osnovi zahtjevaju razdvajanje kvadratnih od pseudokvadratnih ostataka modulo n. Ovo nije ništa teže od faktorizacije broja n. Ako se $n = pq$ može faktorizirati, dalje je jednostavno odrediti $(\frac{x}{p})$. Ako imamo $(\frac{x}{n}) = 1$, slijedi da je x kvadratni ostatak modulo n ako i samo ako je $(\frac{x}{p})$ jednak 1.

Kriptosustav 1 (Cocksov kriptosustav baziran na identitetu)**.** *Neka su p i q dva različita prosta broja takva da je $p \equiv q \equiv 3 \pmod 4$ i definirajmo $n = pq$.*

- ***Parametri sustava:*** *Master ključ je oblika $M = (M^{pub}, M^{priv})$, pri čemu su*

$$M^{pub} = n$$

i

$$M^{priv} = (p, q).$$

Javna hash funkcija $h : \{0,1\}^* \to \mathbb{Z}_n$ ima svojstvo $h(x) \in \boldsymbol{QR}(n) \cup \widetilde{\boldsymbol{QR}}(n)$ za svaki $x \in \{0,1\}^*$.

- ***Generiranje ključa korisnika:*** *Za korisnika U ključ K_U je oblika $K_U = (K_U^{pub}, K_U^{priv})$, pri čemu su:*

$$K_U^{pub} = h(ID(U))$$

i

$$(K_U^{priv})^2 = \begin{cases} K_U^{pub} & \text{ako je } K_U^{pub} \in \boldsymbol{QR}(n) \\ -K_U^{pub} & \text{ako je } K_U^{pub} \in \widetilde{\boldsymbol{QR}}(n). \end{cases}$$

- ***Šifriranje:*** *Otvoreni tekst je element skupa $\{0,1\}$. Element $x \in \{0,1\}$, šifriramo na sljedeći način:*

  1. *Odaberi dvije proizvoljne vrijednosti $t_1, t_2 \in \mathbb{Z}_n$ takve da su Jakobijevi simboli $(\frac{t_1}{n}) = (\frac{t_2}{n})$ jednaki x*
  2. *Izračunaj:*

$$y_1 = t_1 + K_U^{pub}(t_1)^{(-1)} \pmod{n}$$

i

$$y_2 = t_2 - K_U^{pub}(t_2)^{(-1)} \pmod{n}.$$

  3. *Šifrat je $y = (y_1, y_2)$.*

- ***Dešifriranje:*** *Za dani šifrat $y = (y_1, y_2)$, dešifriranje vršimo na sljedeći način:*

  1. *Ako je $(K_U^{priv})^2 = K_U^{pub}$ tada postavi $s = y_1$, u suprotnom postavi $s = y_2$.*
  2. *Izračunaj Jakobijev simbol*

$$x = \left(\frac{s + 2K_{U^{priv}}}{n}\right).$$

  3. *x je dešifrirani otvoreni tekst.*

Algoritam 3 (Blum-Blum-Shub Generator)**.** *Neka su p, q dva $(\frac{k}{2})$-bitna prosta broja takva da je $p \equiv q \equiv 3 \pmod 4$, i definiramo $n = pq$. Neka $\boldsymbol{QR}(n)$ označava skup kvadratnih ostataka modulo n.*
Sjeme s_0 je bilo koji element iz $\boldsymbol{QR}(n)$. Za $0 \leq i \leq l-1$ definiramo

$$s_{i+1} = s_i^2 \pmod{n},$$

i

$$f(s_0) = (z_1, z_2, \ldots, z_l),$$

gdje je

$$z_i = s_i \pmod 2,$$

za $1 \leq i \leq l$. f se tada naziva Blum-Blum-Shub Generator.
Jedan način kako odabrati odgovarajuće sjeme je odabrati element $s_{-1} \in \mathbb{Z}_n^*$ i izračunati $s_0 = s_{-1}^2 \pmod n$ što osigurava da je $s_0 \in \boldsymbol{QR}(n)$.

Potrebno je objasniti nekoliko aspekata Kriptosustava 1. Za početak krenuli smo od pretpostavke da hash funkcija h daje elemente koji su uvijek u skupu $\mathbf{QR}(n) \cup \widetilde{\mathbf{QR}}(n)$ što je ekvivalentno tome da je $0 \leq h(x) \leq n$ i Jakobijev simbol $(\frac{h(x)}{n})$ je jednak 1 za svaki $x \in \{0,1\}^*$.
Moguće je odrediti $(\frac{h(x)}{n})$ u praksi. Ako je on jednak -1 tada je $h(x)$ dovoljno pomnožiti s

nekim fiksnim cijelim brojem $a \in \mathbb{Z}_n$. Cijeli broj a može biti unaprijed određen i javan.
Kreiranje korisnikovog privatnog ključa je stvar računanja korijena modulo n, kako se to već radi u RSA kriptosustavu i samo računanje odrađuje TA jer on zna kako je n faktoriziran.
Treba primjetiti da će TA za korisnika računati korijene pomoću unaprijed definirane specijalne forme koju ćemo nazvati $h(ID(U))$ ili $-h(ID(U))$, jer se unaprijed definirani korijen modulo n može koristiti za faktorizaciju od n. Ovaj napad se ne može izvesti iz konteksta Coksovog kriptosustava baziranog na identitetu jer korisnik ne može iskoristiti TA-ev unaprijed definirani korijen.
Kada korisnik V želi šifrirati otvoreni tekst $x = \pm 1$ kako bi ga poslao korisniku U, V mora nasumično izabrati dva elementa iz $\mathbb{Z}_n$ čija oba Jakobijeva simbola moraju biti jednaka x. Toće učiniti tako da odabere proizvoljne elemente iz $\mathbb{Z}_n$ i računa njihove Jakobijeve simbole, sve dok ne dobije elemetne sa željenim Jakobijevim simbolima. Ako V želi šifrirati duži tekst, tada se svaki element mora neovisno šifrirati koristeći proizvoljne različite t-ove.
Kada U želi dešifrirati šifrat y, potrebni su mu y_1 ili y_2. U odabitre odgovarajući pri čemu drugi može zanemariti. Razlog zašto su oba y_1 i y_2 prosljeđena korisniku U jer korisnik V ne zna je li privatni ključ korisnika U kvadratni korijen K_U^{pub} ili $-K_U^{pub}$.
Pokažimo sada korektnost operacije dešifriranja. Bilo koji šifrat od x lako se može dešifrirati sa dobro zadanim privatnim ključem. Pretpostavimo da korisnik U dobije šifrat (y_1, y_2) i pretpostavimo da je $(K_U^{priv})^2 = K_U^{pub}$. Pokažimo da je

$$\left(\frac{y_1 + 2K_U^{priv}}{n}\right) = x.$$

Koristeći se osnovinim svojstvima Jakobijevih simbola dobivamo sljedeće:

$$\begin{aligned}
\left(\frac{y_1 + 2K_U^{priv}}{n}\right) &= \left(\frac{t_1 + K_U^{pub}(t_1)^{-1} + 2K_U^{priv}}{n}\right) \\
&= \left(\frac{t_1 + 2K_U^{priv} + (K_U^{priv})^2(t_1)^{-1}}{n}\right) \\
&= \left(\frac{t_1(1 + 2K_U^{priv}(t_1)^{-1} + (K_U^{priv})^2(t_1)^{-2})}{n}\right) \\
&= \left(\frac{t_1}{n}\right)\left(\frac{(1 + 2K_U^{priv}(t_1)^{-1} + (K_U^{priv})^2(t_1)^{-2})}{n}\right) \\
&= \left(\frac{t_1}{n}\right)\left(\frac{(1 + K_U^{priv}(t_1)^{-1})^2}{n}\right) \\
&= \left(\frac{t_1}{n}\right)\left(\frac{1 + K_U^{priv}(t_1)^{-1}}{n}\right)^2 \\
&= \left(\frac{t_1}{n}\right).
\end{aligned}$$

U posljednjoj smo jednakosti koristili činjenicu da je

$$\left(\frac{1 + K_U^{priv}(t_1)^{-1}}{n}\right) = \pm 1,$$

što se lako pokaže.
Nadalje ćemo promatrati sigurnost sheme, pokazati ćemo da se uvijeti dani za Kriptosustav 1 mogu iskoristiti za rješavanje problema složenih kvadratnih ostataka u $\mathbb{Z}_n$. Najprije ćemo iskazati jednu tehničku lemu:

Lema 1. *Pretpostavimo da su* $x = \pm 1$ *i* $(\frac{t}{n}) = x$, *pri čemu su* x *i* t *nepoznati. Ako je* $(K_U^{priv})^2 \equiv K_U^{pub} \pmod{n}$, *tada vrijednost*

$$t - K_U^{pub} t^{-1} \pmod{n}$$

ne daje nikakvu informaciju o x. *Analogno, ako je* $(K_U^{priv})^2 \equiv -K_U^{pub} \pmod{n}$, *tada vrijednost*

$$t + K_U^{pub} t^{-1} \pmod{n}$$

ne daje nikakvu informaciju o x.

Dokaz. Pretpostavimo da je

$$(K_U^{priv})^2 \equiv K_U^{pub} \pmod{n}$$

i uzmimo da je

$$y = t - K_U^{pub} t^{-1} \pmod{n}.$$

Tada imamo

$$t^2 - ty - K_U^{pub} \equiv 0 \pmod{n},$$
$$t^2 - ty - K_U^{pub} \equiv 0 \pmod{p}$$

i

$$t^2 - ty - K_U^{pub} \equiv 0 \pmod{q}.$$

Prva kongruencija ima dva rješenja modulo p i produkt ta dva rješenja je kongruentan $-K_U^{pub}$ modulo p. Ako su r_1 i r_2 ta dva rješenja tada imamo

$$\left(\frac{r_2}{p}\right) = \left(\frac{-r_1 K_U^{pub}}{p}\right) = \left(\frac{-r_1 (K_U^{priv})^2}{p}\right) = \left(\frac{-r_1}{p}\right) = -\left(\frac{r_1}{p}\right).$$

Slično se pokaže za drugu kongruenciju, ako su dva rješenja s_1 i s_2 tada imamo

$$\left(\frac{s_2}{q}\right) = -\left(\frac{s_1}{q}\right).$$

Sada kongruencije modulo n imaju četiri rješenja za t. Lako se pokaže da dva rješenja imaju Jakobijev simbol $(\frac{t}{n}) = 1$ a druga dva -1. Stoga nije moguće dobiti ikakve informacije o Jakobijevom simbolu $(\frac{t}{n})$.
Drugi dio leme može se pokazati na sasvim analogan način.

□

Pretpostavimo da COCKS-DECRYPT direktno daje otvoreni tekst šifrata za Cocksov kriptosustav baziran na identitetu (kažemo da je Cocks-Decrypt dešifracijski *oracle*[5] za Cocksov kriptosustav baziran na identitetu). Drugim riječima COCKS-DECRYPT(K_U^{pub}, n, y) vraća točno rješenje x kad god je y ispravan šifrat od x. Pokazat ćemo kako iskoristiti COCKS-DECRYPT algoritam da odredimo da li je K_U^{pub} kvadratni ostatak modulo n ili pseudokvaratni ostatak modulo n. Pogledajmo sada Algoritam 4 koji ćemo nadalje proanalizirati.

[5]kažemo da je nešto oracle ako to nešto u istom trenutku, tj. u vremenu $O(1)$ daje rješenje zadanog problema.

Algoritam 4 (COCKS-ORACLE-RESIDUE-TESTING(n, a))**.**

komentar: $(\frac{a}{n}) = 1$

proizvoljno odaberi $x \in \{1, -1\}$

proizvoljno odaberi $t \in \mathbb{Z}_n$ *takav da je* $(\frac{t}{n}) = x$

$y_1 \leftarrow t + at^{-1} \pmod{n}$

proizvoljno odaberi $y_2 \in \mathbb{Z}_n^*$

$y \leftarrow (y_1, y_2)$

$x' \leftarrow$ *COCKS-DECRYPT*(n, a, y)

ako je $x' = x$

tada vrati ("$a \in \boldsymbol{QR}(n)$")

inače vrati ("$a \in \widetilde{\boldsymbol{QR}}(n)$")

Prije svega pogledajmo koje operacije algoritam izvodi. Ulaz a nam dolazi iz $\mathbf{QR}(n) \cup \widetilde{\mathbf{QR}}(n)$ i smatrati ćemo ga javnim ključem za Cocksov kriptosustav, te ćemo šifrirati proizvoljan otvoreni tekst x. Prema pravilu šifriranja računamo jedino y_1, y_2 biramo proizvoljno iz $\mathbb{Z}_n^*$. Nadalje prosljeđujemo uređeni par (y_1, y_2) dešifracijskom oracleu COCKS-DECRYPT, koji nam vraća šifrat x'. Algoritam 4 nam vraća informaciju da je a kvadratni ostatak modulo n ako i samo ako je $x = x'$.
Pretpostavimo da je $a \in \mathbf{QR}(n)$. Iz prethodne leme slijedi da, iako je y_2 izračunat prema pravilu šifriranja, on ne daje nikakvu informaciju za x. Prema tome COCKS-DECRYPT može lako izračunati x iz y, i u tom slučaju Algoritam 4 vraća da je a kvadratni ostatak.
S druge strane, ako pretpostavimo da je $a \in \widetilde{\mathbf{QR}}(n)$, tada prethodna lema kaže da y_1 ne daje nikakvu informaciju za x. Očito y_2 ne daje nikakvu informaciju za x jer je y_2 proizvoljno odabran. Stoga će vrijednost x' koju COCKS-DECRYPT vraća biti jednaka x u točno polovici slučajeva jer je x proizvoljan i y neovisan o x. Prema tome Algoritam 4 će biti korektan sa vjerojatnošću $1/2$.
Ako je $x \neq x'$, tada možemo biti sigurni da je $a \in \widetilde{\mathbf{QR}}$. S druge strane ako je $x = x'$ ne možemo sa sigurnošću reći da je $a \in \mathbf{QR}(n)$, jedino što se može biti jest da COCKS-DECRYPT točno pogodi vrijednost x. Dakle potrebno je nekoliko puta pokrenuti prethodni algoritam na istim ulazima, ako uvijek vrati da je $a \in \mathbf{QR}(n)$ tada možemo biti sigurni da je zaključak točan.

Sažetak

U ovom radu bavili smo se infrastrukturom javnog ključa u oznaci PKI. Pogledali smo neke bitnije dijelove od kojih se PKI sastoji i ukratko ih opisali. Nakon toga, pogledali smo jedan protokol koji se koristi u praksi, a to je SSL protokol kojega smo ilustrirali na primjeru. Objasnili smo pojam certifikata koji je građevna jedinica PKI-ja i uz to dali primjer ceritfikata. Opisali smo model koji daje jasna pravila koja određuju način na koji će se graditi put certifikata, a to je model povjerenja. Naveli smo neke primjere modela povjerenja i opisali ih na primjerima. U poglavlju nakon toga rekli smo nešto o budućnosti PKI-a i na kraju smo objasnili kriptografiju baziranu na identitetu, tj. obradili smo Cocksovu enkripcijsku shemu baziranu na identitetu, kao jedan oblik ove kriptografije.

Ključne riječi: PKI, certifikati, SSL, Cocksova enkripcijska shema, Blum-Blum-Shub Generator, Javni ključ, Privatni ključ.

Summary

This paper aims to discuss the Public-key Infrastructure in the PKI label, which includes and describes some of the more important parts of the PKI. One of the more used protocols, SSL, was presented along with an example. In this paper certificates as the founding blocks of PKI were also more thoroughly explained, as well as a model that provides clear rules which define the certificates path si built, the trust model for which more examples are provided. The chapter after that discusses the future of PKI and identity-based cryptography, such as the Cocks Identity-based Encryption Scheme.

Key words: PKI, Certificates, SSL, Cocks Identity-based Encryption Scheme, Blum-Blum-Shub Generator, Public key, Private key.

Literatura:

[1] A. Das, C. E. Veni Madhavan *Public-key Cryptography: Theory and Practice*, Pearson India, 2009.

[2] W. Diffie, M.E. Hellman, *New Directions in Cryptography*, IEEE Transactions on information theory, 6(22), 644-654, 1976.

[3] L. He, C. Yuan, H. Xiong, Z. Qin, *Certificateless Key-Insulated Encryption: Cryptographic Primitive for Achieving Key-Escrow Free and Key-Exposure Resilience*. 387-395, U: Wang Y., Yu G., Zhang Y., Han Z., Wang G. (eds) Big Data Computing and Communications. BigCom 2016. Lecture Notes in Computer Science, vol 9784. Springer, Cham

[4] D. R. Stinson *Cryptography: Theory and Practice, Third Edition (Discrete Mathematics and Its Applications)*, Chapman & Hall/CRC, Taylor & Francis Group, LCC, 2006

Životopis

Rođen sam 27. svibnja 1991. u Našicama. Osnovnu školu pohađao sam u Beničancima i nakon toga upisao Opću gimnaziju u Našicama. Nakon završene gimnazije dolazim u Osijek gdje upisujem Preddiplomksi studij matematike na Odjelu za matematiku, na Sveučilištu J. J. Strossmayera. Nakon završenog Preddiplomskog studija s temom Kongruencije višeg reda stekao sam akademski stupanj prvostupnika matematike. Zaposlio sam se kao agent teleprodaje u službi za korisnike operatera Tele2 gdje sam radio 6 mjeseci i nakon tih 6 mjeseci upisujem diplomski studij na Odjelu za matematiku na smjeru Matematika i računarstvo. Tijekom završne godine diplomksog studija odradio sam stručnu studentsku praksu u tvrtki Adacta d.o.o.