SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1658

Infrastruktura javnih ključeva u prividnoj mreži računalnih sustava zasnovanih na uslugama

Davor Čuljak

Zagreb, srpanj 2007.

Zahvaljujem se mentoru prof. dr. sc. Siniši Srbljiću što mi je omogućio sudjelovanje na zanimljivom projektu.

Zahvaljujem se mr. sc. Dejanu Škvorcu na pruženoj pomoći i stručnom nadzoru pri pisanju diplomskog rada.

Posebno se zahvaljujem svojoj djevojci Ivani i svojoj obitelji što su tijekom cijelog mog studija imali razumijevanja i pružali mi potrebnu moralnu podršku.

Sadržaj

1 Uvod

U današnje vrijeme se sve veći broj programskih sustava projektira i gradi za izvođenje na globalnoj računalnoj mreži Internet. Ako se pritom mrežom prenose podaci povjerljive prirode za koje se zahtijevaju povlaštena prava pristupa, takvi programski sustavi trebaju ispunjavati sigurnosne zahtjeve poput provjere izvornosti korisnika, neporecivost izvorišta poruke i zaštitu sadržaja poruke. Zadovoljavanje navedenih zahtjeva postiže se uspostavljanjem sigurnog komunikacijskog kanala između sudionika u komunikaciji računalnom mrežom.

Sigurnosni mehanizmi koji se upotrebljavaju za uspostavljanje sigurnog komunikacijskog kanala zasnivaju se na primjeni metoda kriptografije [1] kojima se jamči tajnost, izvornost i vjerodostojnost podataka koji se prenose računalnom mrežom.

Budući da se kriptografski postupci zasnivaju na poznavanju kriptografskih ključeva sudionika u komunikaciji, posebna pažnja posvećuje se postupcima sigurne razmjene tih ključeva. Najrašireniji način razmjene kriptografskih ključeva je primjena infrastrukture javnih ključeva (engl. *Public Key Infrastructure - PKI*) [2]. Infrastruktura javnih ključeva je računalni sustav koji osigurava sigurnu razmjenu kriptografskih ključeva putem nesigurnih računalnih mreža poput mreže Internet. Infrastruktura javnih ključeva koristi digitalne vjerodajnice (engl. *digital certificate*) kako bi ispunila sve potrebne sigurnosne zahtjeve. Svaki sudionik infrastrukture javnih ključeva ima vlastitu digitalnu vjerodajnicu koja nepobitno dokazuje njegov identitet.

Infrastruktura javnih ključeva neizostavni je element u računalnim sustavima zasnovanim na uslugama. Računarstvo zasnovano na uslugama (engl. *service-oriented computing*) [4] jedna je od najznačajnijih paradigmi razvoja suvremenih programskih sustava. U računarstvu zasnovanom na uslugama programski sustavi se grade pretraživanjem i povezivanjem usluga dostupnih u globalnoj

mreži Internet. Usluge su cjelovite i samostalne jedinice programske potpore koje se mogu opisivati, pretraživati i međusobno povezivati koristeći standardni skup protokola. Povezivanjem osnovnih usluga ostvaruju se nove, složene usluge, koje se u razvoju programskih sustava koriste na jednak način kao i osnovne usluge. Primjenom infrastrukture javnih ključeva u sustavima zasnovanim na uslugama ostvaruje se sigurnost usluga, podataka i korisnika.

Ovim diplomskim radom opisana je programska izvedba infrastrukture javnih ključeva za primjenu u sustavima zasnovanim na uslugama. U drugom poglavlju diplomskog rada dan je pregled kriptografskih postupaka kojima se uspostavlja sigurni komunikacijski kanal te se ispunjavaju osnovni sigurnosni zahtjevi. Objašnjena je uloga i arhitektura infrastrukture javnih ključeva te je dan kratki pregled standarda digitalnih vjerodajnica. U trećem poglavlju dan je uvid u računarstvo zasnovano na uslugama te je opisan osnovni i prošireni skup standarda *Web Services* [5] koji se koristi za izgradnju sustava zasnovanih na uslugama. Opisana je Programirljiva Internet okolina (engl. *Programmable Internet Environment – PIE*) [22] kao primjer sustava koji ostvaruje programski model zasnovan na uslugama. U četvrtom poglavlju opisana je arhitektura infrastrukture javnih ključeva u okolini PIE te su objašnjeni protokoli dodjele, razmjene i opoziva digitalnih vjerodajnica s primjerom izvođenja. U petom poglavlju opisano je programsko ostvarenje infrastrukture javnih ključeva u okolini PIE. Završni zaključci izneseni su u šestom poglavlju.

2 Sigurnosni sustavi zasnovani na infrastrukturi javnih ključeva

Globalna svjetska računalna mreža Internet danas je prisutna u gotovo svim aspektima života. Njezin nagli rast i raznovrsne mogućnosti primjene privukao je postojeće i stvorio nove poslovne organizacije koje za svoje poslovanje koriste Internet. Nerijetki su slučajevi u kojima su poslovne organizacije zemljopisno raspodijeljene, dok zahvaljujući mreži Internet organizacijski djeluju kao cjelina. Komunicirajući putem mreže Internet, različiti poslovni procesi premošćuju zemljopisnu raspodijeljenost, ali se istovremeno susreću s novim problemima, od kojih je najveći problem sigurnosti. Povjerljive informacije koje se razmjenjuju putem računalne mreže mogu imati dalekosežne nepovoljne posljedice za rad poslovne organizacije ako dođu u posjed neovlaštenim korisnicima. Stoga je od velike važnosti uspostaviti mehanizme koji će osigurati da povjerljive informacije ostanu povjerljive te da su dostupne samo ovlaštenim osobama. Iako se mreža Internet općenito smatra nesigurnim komunikacijskim medijem, uz primjenu odgovarajućih mehanizama moguće je ostvariti sve potrebne sigurnosne zahtjeve. U ovom poglavlju objašnjeni su postupci uspostave sigurnosnih mehanizama u mreži Internet kojima se bavi grana računarskih znanosti pod nazivom kriptografija [1].

2.1 Osnove kriptografije

Podaci koji se mogu pročitati i razumjeti nazivaju se jasnim tekstom (engl. *cleartext*). Metode za skrivanje značenja i pretvaranje sadržaja jasnog teksta u nerazumljivi oblik nazivaju se metodama kriptiranja. Kriptiranjem jasnog teksta dobiva se nerazumljiv skup podataka koji se naziva kriptiranim tekstom (engl. *ciphertext*). Postupak pretvorbe kriptiranog teksta u jasni tekst naziva se dekriptiranje.

Postupcima kriptiranja i dekriptiranja podataka bavi se grana računarstva koja se naziva kriptografijom [1]. Kriptografija je znanost koja omogućuje spremanje i prijenos osjetljivih informacija kroz nesigurne računalne mreže kao što je Internet. Primjena kriptografskih postupaka omogućuje prijenos informacije nesigurnom računalnom mrežom na način da informaciju koja se prenosi ne može pročitati nitko osim onoga kome je namijenjena. Kriptografski algoritmi koriste matematičke funkcije u postupcima kriptiranja i dekriptiranja.

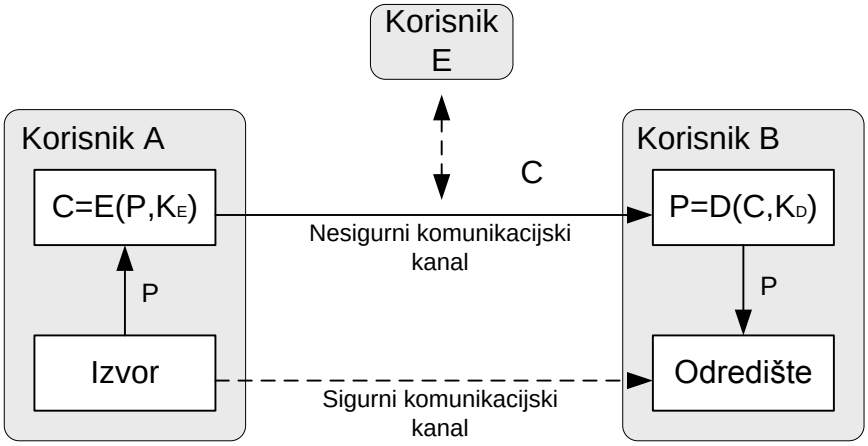
Postupak kriptiranja se formalno može izraziti u obliku

$$C = E(P,K_e)$$

gdje je P jasni tekst, K_e je ključ kriptiranja, E predstavlja metodu kriptiranja, a C je dobiveni kriptirani tekst. Postupak dekriptiranja se formalno može izraziti u obliku

$$P = D(C,K_d)$$

gdje je C kriptirani tekst, K_d je ključ dekriptiranja, D predstavlja metodu dekriptiranja, a P je dobiveni jasni tekst.



Slika 1.1 Uspostavljanje sigurnog komunikacijskog kanala

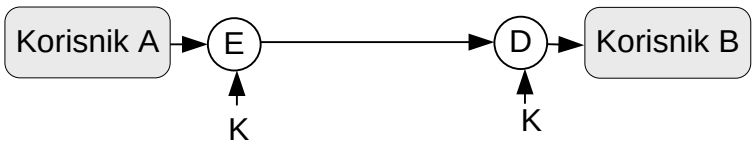
Slika 1.1 prikazuje primjenu kriptografskih postupaka pri komunikaciji dvaju korisnika. Korisnik A kriptira poruku ključem za kriptiranje te je šalje kroz nesigurni komunikacijski kanal do korisnika B. Korisnik B po primitku poruke pomoću ključa za dekriptiranje dekriptira kriptiranu poruku. Korisnik E je neovlašteni korisnik koji prisluškuje nesigurni komunikacijski kanal. Korisnik E vidi poruke koje razmjenjuju korisnici A i B, ali je njihov sadržaj kriptiran i za njega

nerazumljiv jer nema odgovarajući ključ za dekriptiranje. Sigurni komunikacijski kanal je kombinacija nesigurnog komunikacijskog kanala i određenih kriptografskih postupaka sprječava mogućnost napada na sigurnost.

Svrha kriptografskih postupaka je osiguravanje šest sigurnosnih zahtjeva koji su nužni za sigurnu komunikaciju: *povjerljivost*, *raspoloživost*, *vjerodostojnost*, *autentikacija*, *autorizacija* i *neporecivost*. *Povjerljivost* ili *tajnost* osigurava da informacije u sustavu smiju biti dostupne samo ovlaštenim korisnicima. *Raspoloživost* osigurava da su informacije uvijek dostupne ovlaštenim korisnicima usprkos mogućim neočekivanim i nepredvidljivim događajima. *Vjerodostojnost* osigurava da informacije u sustavu mogu mijenjati samo ovlašteni korisnici. Jednoznačno prepoznavanje ovlaštenog korisnika osigurano je pomoću *autentikacije*. Mehanizmom *autorizacije* se ovlaštenim korisnicima dozvoljava pristup samo do onih sadržaja do kojih imaju prava pristupa. *Neporecivost* predstavlja zaštitu od opovrgavanja prethodno počinjenog djela.

2.1.1 Simetrična kriptografija

Simetrična kriptografija, poznata i pod nazivom kriptografija tajnim ključem, najpoznatiji je i najstariji poznati oblik kriptografije [1]. Ključ kriptiranja K_E i ključ dekriptiranja K_D su jednaki i predstavljaju tajni ključ koji znaju samo sudionici sigurne komunikacije. Na slici 1.2 je prikazan primjer komunikacije između korisnika A i korisnika B koji koriste simetričnu kriptografiju.



Slika 1.2 Simetrična kriptografija

Osim tajnog ključa, za proces kriptiranja uz primjenu simetrične kriptografije potrebno je poznavati i algoritam kriptiranja. Dva simetrična algoritma koja su

danas najraširenija u Internet protokolima su DES (engl. *Data Encryption Standard*) [33] IDEA (engl. *International Data Encryption Algorithm*) [34].
DES je razvijen 1977. godine od strane tvrtke IBM. Koristi se tajnim ključem duljine 56 bitova. S obzirom na današnju snagu računala i brzinu izračunavanja, duljina tajnog ključa od samo 56 bitova postala je prekratka i predstavlja najkritičniji dio ovog algoritma. Stoga je DES 2001. godine zamijenjen novim simetričnim algoritmom AES (engl. *Advanced Encryption Standard*) [35] koji podržava ključeve duljine 128, 192 i 256 bitova.

Simetrični algoritam IDEA razvijen je 1991. godine od strane ETH Zurich te koristi ključ duljine 128 bitova. Osim algoritama DES i IDEA, koriste se još i Twofish [36], Blowfish [37], CAST5 [38], RC4 [39] i TDES [40].

2.1.2 Asimetrična kriptografija

Asimetrična kriptografija, poznata i pod nazivom kriptografija javnim ključem, objavljena je 1976. godine od strane dvaju znanstvenika, Whitfielda Diffiea i Martina Hellmana. Za razliku od simetrične, asimetrična kriptografija koristi dva različita ključa od kojih se jedan koristi za kriptiranje i drugi za dekriptiranje [1]. Asimetrični algoritmi zasnivaju se na matematičkom problemu faktorizacije velikih prirodnih brojeva na proste faktore.

Svaki sudionik u sigurnoj komunikaciji posjeduje jedinstveni par ključeva koji se nazivaju javni ključ *p* i privatni ključ *d*. Javni ključ je javno dostupan svima i služi za kriptiranje. Privatni ključ je poznat samo njegovom vlasniku i služi za dekriptiranje poruka. Određivanje javnog ključa na osnovi poznatog privatnog ključa je jednostavno, dok je određivanje privatnog ključa na osnovi poznatog javnog ključa matematički izrazito zahtjevna operacija koja je praktički neizvediva, čak i uz primjenu najsuvremenije računalne opreme. Upravo to svojstvo predstavlja osnovu sigurnosti asimetričnih algoritama.

Na slici 1.3 prikazan je princip rada asimetričnog kriptografskog algoritma. Korisnik A koristeći javni ključ K_{EB} korisnika B kriptira poruku te je pošalje korisniku B. Kako samo korisnik B zna svoj privatni ključ K_{DB}, jedino on može dekriptirati primljenu poruku. Koristeći svoj privatni ključ, korisnik B dekriptira poruku i pročita njen sadržaj.

Slika 1.3 Asimetrična kriptografija

Asimetrični algoritmi su zbog svoje matematičke složenosti sporiji u odnosnu na simetrične algoritme pa se zbog toga ne koriste za kriptiranje većih količina podataka. Uglavnom se koriste za razmjenu tajnog ključa simetrične kriptografije te za stvaranje digitalnih potpisa. Najpoznatiji asimetrični algoritam je RSA. Objavljen je 1977. godine, a naziv je dobio po svojim tvorcima Ronald L. Rivestu, Adiu Shamiru i Leonardu Adlemanu.

2.1.3 Sažimanje poruke

Sažetak poruke (engl. *Message digest*) osigurava sigurnosni zahtjev vjerodostojnosti poruke (engl. *integrity*) [1]. Kriptografski sažetak izrađuje se jednosmjernom funkcijom koja iz poruke proizvoljne duljine izračunava sažetak stalne duljine. Funkcija sažimanja je jednosmjerna, što znači da je izračunavanje sažetka poruke vrlo jednostavno i brzo, dok je dobivanje izvorne poruke na osnovi poznatog sažetka zbog velike računalne složenosti praktički neprovedivo. Formalni zapis algoritma za računanje kriptografskog sažetka poruke je

$$S = H(P)$$

gdje je P jasni tekst iz kojeg se izračunava sažetak, H predstavlja primijenjeni algoritam sažimanja, a S je dobiveni sažetak poruke.

Najjednostavniji oblik jednosmjerne funkcije sažimanja je uzastopna primjena funkcije ISKLUČIVO-ILI (engl. *XOR*) na nizu bitova koji se dobivaju dijeljenjem izvorne poruke na dijelove jednake duljine. Najpoznatiji algoritmi koji izračunavaju sažetak poruke su MD5 [1], SHA-*1* [1] i Tiger [1]. Standardne duljine sažetka kreću se od 64 bita pa do 256 bitova, a ovise o algoritmu sažimanja.

2.1.4 Digitalni potpis

Digitalni potpisi se, poput vlastoručnih potpisa, koriste za autentikaciju [16]. U kombinaciji s sažetkom poruke osiguravaju sigurnosne zahtjeve autentikacije, vjerodostojnosti i neporecivosti.

Digitalni potpis stvara se primjenom asimetrične kriptografije. Za stvaranje digitalnog potpisa ključevi se koriste obrnuto od postupka kriptiranja i dekriptiranja. Poruka se potpisuje tako da se kriptira privatnim ključem potpisnika, a ispravnost potpisa provjerava se tako da se kriptirani tekst dekriptira javnim ključem potpisnika. Kako se za potrebe digitalnog potpisa ne bi kriptirale poruke koje se razmjenjuju, jer je to vremensko zahtjevno za dulje poruke, digitalno se potpisuje samo sažetak poruke koji se dobije primjenom neke od funkcija za izračun kriptografskog sažetka.

Digitalni potpis se može formalno izraziti formulom:

$$\textit{Digitalni potpis} = E(H(m), K_D)$$

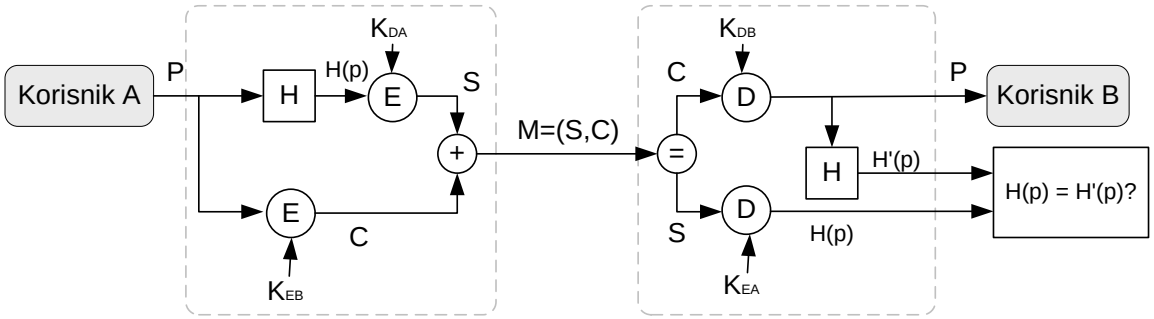
a digitalno potpisani dokument formulom:

$$\textit{Digitalno potpisani dokument} = \{m, E(H(m), K_D)\}$$

gdje je *m* digitalni dokument koji se potpisuje, *H(m)* je sažetak digitalnog dokumenta, *H* funkcija sažetka, K_D privatni ključ potpisnika, a *E* predstavlja asimetrični algoritam kriptiranja.

Primjena digitalnog potpisa prikazana je na slici 1.4 gdje korisnik B koristeći digitalni potpis korisnika A utvrđuje autentičnost poruke. Korisnik A koristeći javni

ključ K_{EB} korisnika B kriptira jasni tekst. Funkcijom za dobivanje kriptografskog sažetka H izračunava sažetak jasnog teksta te ga kriptira svojim privatnim ključem K_{DA}. Poruka koju šalje korisniku B sastoji se od kriptiranog teksta i kriptiranog sažetka poruke. Korisnik B po primitku poruke koristeći svoj privatni ključ K_{DB} dekriptrita poruku te pomoću funkcije za dobivanje kiptografskog sažetka izračunava odgovarajući sažetak. Dobiveni kriptirani sažetak dekriptira koristeći pri tomu javni ključ K_{EA} korisnika A te rezultat uspoređuje s netom izračunatim sažetkom. Ukoliko su oba sažetka jednaka, na temelju činjenice da samo korisnik A zna svoj privatni ključ kojim je kriptirao sažetak zaključuje da je izvorište poruke uistinu korisnik A, te na temelju činjenica da jedinstvenosti sažetka poruke zaključuje da sadržaj poruke nije bio mijenjan tijekom prijenosa.



Slika 1.4 Korištenje digitalnog potpisa

2.1.5 Digitalna vjerodajnica

Digitalna vjerodajnica (engl. *Digital Certificate*) je digitalno potpisani dokument koji povezuje javni ključ s osobom kojoj pripada. Uvedena je iz razloga što sudionici u komunikaciji moraju doznati ključeve svojih sugovornika. Osim toga, moraju biti uvjereni da sugovornici nisu uljezi koji se lažno predstavljaju. Zamisao digitalne vjerodajnice predložio je L. Kohnfelder 1978. godine. Da bi se osigurala vjerodostojnost digitalne vjerodajnica, ona se digitalno potpisuje, čime potpisnik jamči njezin integritet. Potpisnik digitalne vjerodajnice naziva se certifikacijski centar (engl. *Certification Authority - CA*). Certifikacijski centar je ustanova ili tijelo kojoj svi korisnici vjerodajnica vjeruju i čiji javni ključ, koji se koristi za provjeru potpisa na vjerodajnici, mora biti pouzdano ispravan.

2.2 Uspostavljane sigurnog komunikacijskog kanala pomoću infrastrukture javnih ključeva

Komunikacija kojom se razmjenjuju poruke povjerljivog sadržaja kao što su bankovne transakcije ili razmjena osobnih podataka mora zadovoljavati nekoliko osnovnih zahtjeva. Prvi zahtjev je autentikacija, odnosno mogućnost da primatelj poruke pouzdano utvrdi identitet pošiljatelja poruke. Drugi zahtjev je vjerodostojnost, odnosno mogućnost da primatelj poruke utvrdi da li se na putu od odredišta do cilja poruka mijenjala i da li je stigla cjelovita. Treći zahtjev je neporecivost, odnosno sprječavanje pošiljatelja poruke u opovrgavanju činjenice da je on poslao poruku. Četvrti zahtjev je povjerljivost kojim se osigurava tajnost sadržaja poruke za svakoga kome poruka nije namijenjena.

Infrastruktura javnih ključeva zadovoljava sve gore navedene zahtjeve te time omogućuje korisnicima da kroz nesigurnu javnu mrežu, kao što je mreža Internet, sigurno razmjenjuju podatke, novac i druge osjetljive informacije. koristeći pritom asimetričnu kriptografiju i digitalne vjerodajnice. Infrastruktura javnih ključeva predstavlja osnovu na kojoj se grade aplikacije i mrežne sigurnosne komponente. Sustavi koji često zahtijevaju sigurnosne mehanizme infrastrukture javnih ključeva su elektronička pošta (engl. *e-mail*), elektroničko poslovanje (engl. *e-commerce*), Internet bankarstvo i sl. Osnovne sigurnosne usluge kao što su SSL (engl. *Secure Socket Layer)* [23], IPsec *(engl. Internet Protocol security)* [24] i HTTPS (engl. *Hypertext Transfer Protocol over Secure Socket Layer*) [27] za sigurnosnu komunikaciju, S/MIME *(engl. Secure Multi-Purpose Internet Mail Extensions)* [25] i PGP *(engl. Pretty Good Privacy)* [26] za sigurnost elektronske zasnivaju se na infrastrukturi javnih ključeva.

2.2.1 Elementi infrastrukture javnih ključeva

Infrastruktura javnih ključeva se sastoji od elemenata koji su povezani u stablastu hijerarhijsku strukturu. Na vrhu hijerarhije nalazi se Policy Approval Authority (*PAA*), razinu ispod njega nalazi se Policy Certification Authority (*PCA*) , slijede Certification Authority (*CA*) i Registration Authority (*RA*), dok se korisnici nalaze na dnu hijerarhije.

Policy Certification Authority (*PAA)* predstavlja korijen infrastrukture javnih ključeva. Svaki PAA element izdaje svoj javni ključ, definira pravila i procedure kojih se elemetni PCA, CA, RA i korisnici koji su dio te infrastrukture moraju pridržavati.

Policy Certification Authority (PCA) se smatraju svi elementi koji se nalaze na drugoj razini hijerarhije infrastrukture javnih ključeva. Svaki PCA element opisuje korisnike kojima služi. Svi su zaduženi za upravljanje odredbama i digitalnim vjerodajnicama te moraju objaviti svoja sigurnosna pravila, procedure. Za PCA elemente korisnici mogu biti osobe, organizacije ili uređaji.

Certification Authority (CA) se nalazi razinu ispod PCA elementa. Infrastruktura javnih ključeva sadrži mnogo CA elemenata koji nemaju obvezu da donose ikakva sigurnosna pravila za razliku od PAA i PCA elemenata koji su ih morali donositi. Osnovna funkcija jednog CA elementa jest da generira, objavljuje, opoziva i arhivira vjerodajnice svojih korisnika. Slijedeći pravila koja su donijeli PAA element i PCA element, CA element ovjerava javni ključ korisnika, tj. ovjerava da javni ključ koji je predmet digitalne vjerodajnice, a kojeg je CA element generirao, pripada vlasniku te digitalne vjerodajnice.

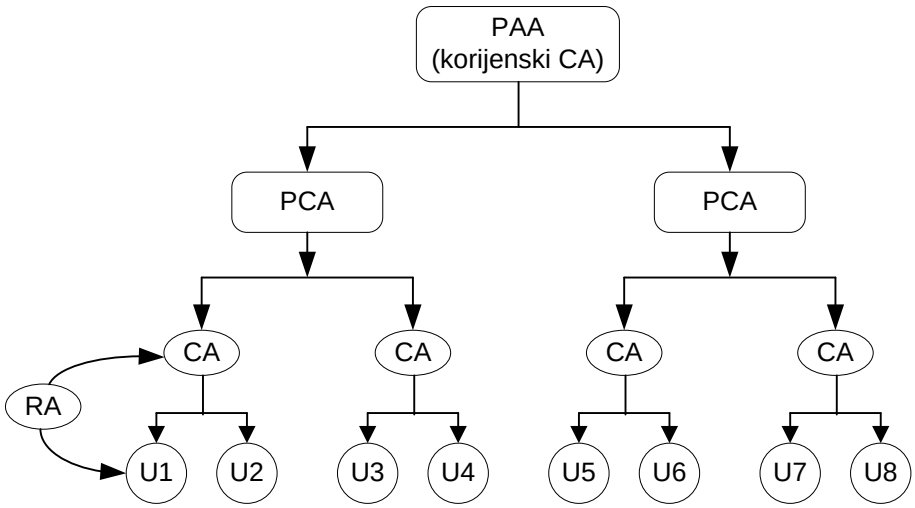
Registration Authority (RA) predstavlja sučelje između korisnika i CA elementa. Glavna funkcija RA elementa jest identifikacija i autentikacija korisnika u ime CA elementa te dostavljanje vjerodajnice generirane od strane CA elementa korisniku. Nakon autentikacije RA element šalje digitalno potpisani zahtjev za

vjerodajnicom odgovarajućem CA elementu. Kao odgovor, CA element vraća odgovarajuću vjerodajnicu. Primljenu vjerodajnicu RA element prosljeđuje korisniku. Kako sudjeluje kod generiranja vjerodajnica, RA element isto tako sudjeluje i kod njihovog opoziva. Ukratko bi se dalo reći da RA element obnaša funkcije infrastrukture javnih ključeva u ime korisnika kojem služi.

Korisnici se nalaze na dnu. Oni koriste infrastrukturu javnih ključeva kako bi uspostavili sigurni komunikacijski kanal jedni s drugima i tako razmijenili povjerljive podatke.

2.2.2 Struktura i organizacija infrastrukture javnih ključeva

Koncept infrastrukture javnih ključeva sastoji se od sigurnosnih pravila, sigurnosnih usluga te protokola za upravljanje digitalnim vjerodajnicama. Stvaranje, podjela i upravljanje vjerodajnicama s javnim ključevima zadaci su CA i RA elementa koji tvore lanac povjerenja (engl. *Chain of trust*). Lanac povjerenja slijedi strogu stablastu hijerarhiju prikazanu na slici 1.5.

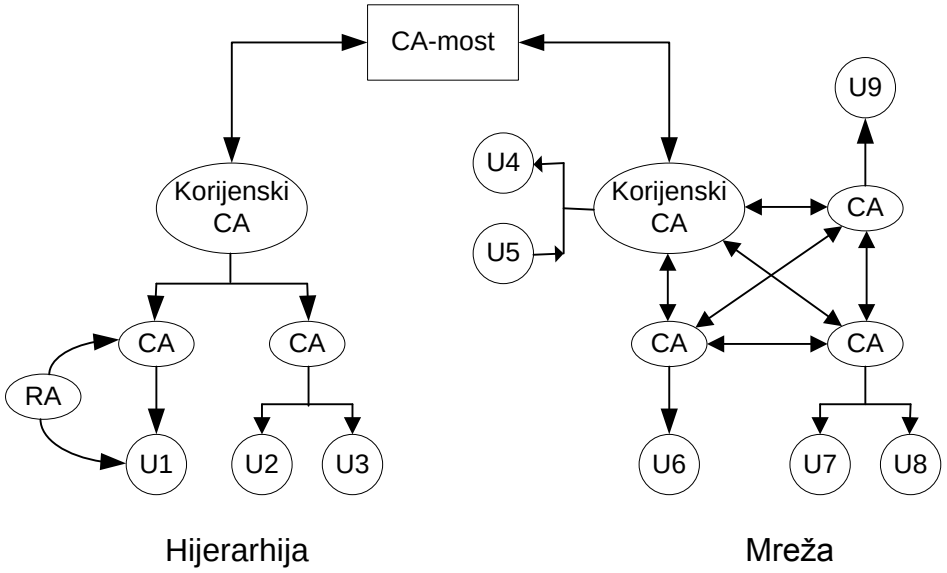


Slika 1.5 Stablasta hijerarhijska struktura infrastrukture javnih ključeva

U prikazanoj hijerarhiji korijen stabla predstavlja glavni CA element odnosno PAA na kojem se zasniva lanac povjerenja. Svaki entitet mreže infrastrukture javnih

ključeva povezan je s glavnim CA elementom te na taj način tvori lanac povjerenja. Primjer takvog lanca povjerenja je lanac *U1-CA-PCA-PAA* u kojem za vjerodostojnost identiteta U1 jamči CA element za čiju pak vjerodostojnost jamči PCA element, a njegovu vjerodostojnost jamči PAA element. PAA element je vršni element u stablu na kojem se zasniva lanac povjerenja.

Odvojene hijerarhijske strukture povjerenja mogu se međusobno povezivati te time tvoriti složene lance povjerenja. Na slici 1.6 prikazana je veza dviju hijerarhija pomoću CA-mosta. CA-most je CA element preko kojeg dvije različite hijerarhije čine lanac povjerenja.



Slika 1.6 Veza dviju hijerarhija pomoću CA-mosta

Upravo postojanje lanaca povjerenja omogućuje da korisnici koji nisu poznati jedni drugima uspostave povjerljivu vezu s CA elementom. CA element obavlja autentikaciju korisnika, objavljuje njihove vjerodajnice koje je potpisao svojim privatnim ključem te time jamči vjerodostojnost identiteta vlasnika tih vjerodajnica. Nakon što su pribavili vjerodajnice izdane od strane CA elementa, korisnici mogu uspostavili povjerljivu vezu bez potrebe za međusobnom autentikacijom. Jedna od glavnih prednosti infrastrukture javnih ključeva je mogućnost uspostave hijerarhije povjerenja u raznorodnim okolinama.

2.2.3 Funkcije infrastrukture javnih ključeva

Osnovni procesi koje obavlja infrastruktura javnih ključeva su *asimetrično kriptiranje*, *generiranje*, *provjera i opoziv digitalnih vjerodajnica*. U nastavku je dan kratak uvid u navedene procese.

Primjena asimetrične kriptografije u infrastrukturi javnih ključeva

Za asimetrično kriptiranje u infrastrukturi javnih ključeva najčešće se koriste asimetrični kriptografski postupci zasnovani na RSA algoritmu. Osim za kriptiranje RSA ključevi koriste se i za stvaranje digitalnog potpisa.

Iako RSA algoritam dozvoljava korištenje istog para ključeva za kriptiranje i digitalno potpisivanje, u praksi to nije preporučljivo. Uvodi se više parova ključeva tako da se različiti parovi koriste za različite funkcije. Primjerice, jedan par ključeva koristi se za kriptiranje dok se drugi koristi za stvaranje digitalnog potpisa. Na taj način se smanjuje rizik da će ključevi biti otkriveni.

Stvaranje digitalnih vjerodajnica

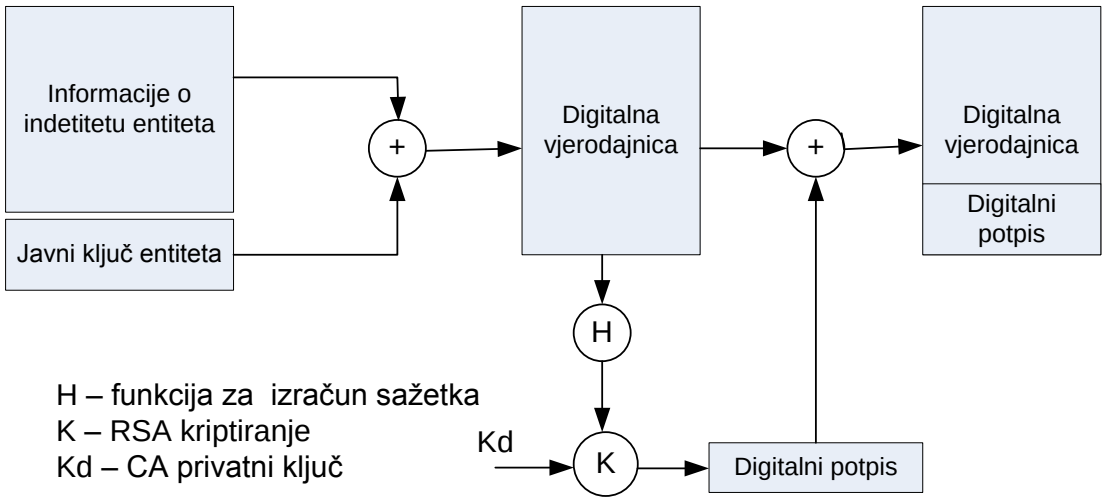
Da bi korisnik postao dio infrastrukture javnih ključeva treba se registrirati kod odgovarajućeg CA elementa. Kao rezultat registracije korisnik dobiva svoju digitalnu vjerodajnicu. Glavna zadaća CA elementa jest povezivanje korisničkog identiteta s njegovim javnim ključem. To povezivanje ostvaruje se digitalnim potpisivanjem korisnikovog javnog ključa privatnim ključem CA elementa. U nastavku je detaljnije objašnjen proces stvaranja digitalne vjerodajnice, odnosno uključivanja novog korisnika u infrastrukturu javnih ključeva.

Da bi stvorio digitalnu vjerodajnicu, CA element mora izvesti sljedeće korake: dobaviti informacije o korisniku, provjeriti identitet korisnika, odrediti značajke potrebne za digitalnu vjerodajnicu ukoliko postoje, stvoriti digitalnu vjerodajnicu te ju digitalno potpisati. Ovisno o području primjene vjerodajnice, CA element mora *dobaviti određene informacije te javni ključ korisnika*.

Također, ovisno o području namjene vjerodajnice te stupnju povjerenja u nju, CA element poduzima određene mjere *provjere identiteta korisnika* (autentikacije). Npr. CA element može jednostavno bez ikakve provjere vjerovati na riječ krajnjem korisniku da on zaista jest onaj za koga se izdaje ili pomoću dogovorenih procesa identifikacije i autentikacije utvrditi vjerodostojnost korisnikova identiteta.

Stvaranje digitalne vjerodajnice podrazumijeva stavljanje svih prikupljenih podataka na jedno mjesto, u jedan dokument. Informacije i format javnog ključa mogu varirati ovisno o potrebama infrastrukture javnih ključeva.

Na kraju CA element *digitalno potpisuje vjerodajnicu* koristeći pri tome svoj privatni ključ. Jednom potpisana vjerodajnica može biti distribuirana i objavljivana na različite načine.

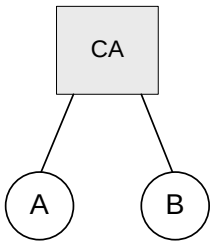


Slika 1.7 Generiranje digitalne vjerodajnice

Upotreba digitalnih vjerodajnica

Svi korisnici koji međusobno žele uspostaviti sigurni komunikacijski kanal moraju neposredno prije stvoriti ako već nisu vjerodajnice, razmijeniti ih te provjeriti jesu li valjane. Uspostavljanje sigurnog komunikacijskog kanala ilustrirano je sljedećim primjerom u kojem korisnici A i B žele komunicirati. Prije uspostave A i B trebaju postati dio infrastrukture javnih ključeva. To postaju kroz nekoliko koraka. Prvo korisnici A i B generiraju svoje javne i privatne ključeve, potom proslijede svoje javne ključeve, ime i opisne informacije do RA elementa koji provjerava njihov identitet te proslijedi zahtjeve za vjerodajnicom do CA elementa. CA element zatim generira digitalne vjerodajnice za korisnike A i B. Na kraju korisnici A i B generiraju svaki svoj tajni simetrični ključ. Sada korisnici A i B imaju javne i privatne ključeve, vjerodajnice i tajni simetrični ključ.

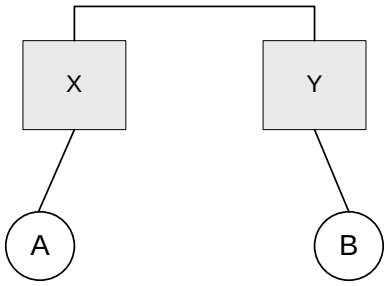
Sljedeći korak u uspostavljanju sigurnog komunikacijskog kanala je razmjena vjerodajnica. Moguća su dva slučaja: prvi kada korisnici A i B pripadaju pod isti CA element i drugi u kojem korisnici A i B pripadaju pod različite CA elemente. U prvom slučaju i korisnik A i korisnik B imaju vjerodajnice koje je generirao isti CA element kao što je to prikazano na slici 1.8. U ovom slučaju korisnik A ima pristup vjerodajnici od korisnika B i korisnik B ima pristup vjerodajnici od korisnika A jer se te vjerodajnice nalaze u repozitoriju njihovog zajedničkog CA elementa te im ih taj CA element prosljeđuje.



Slika 1.8 A i B pripadaju istom CA-u

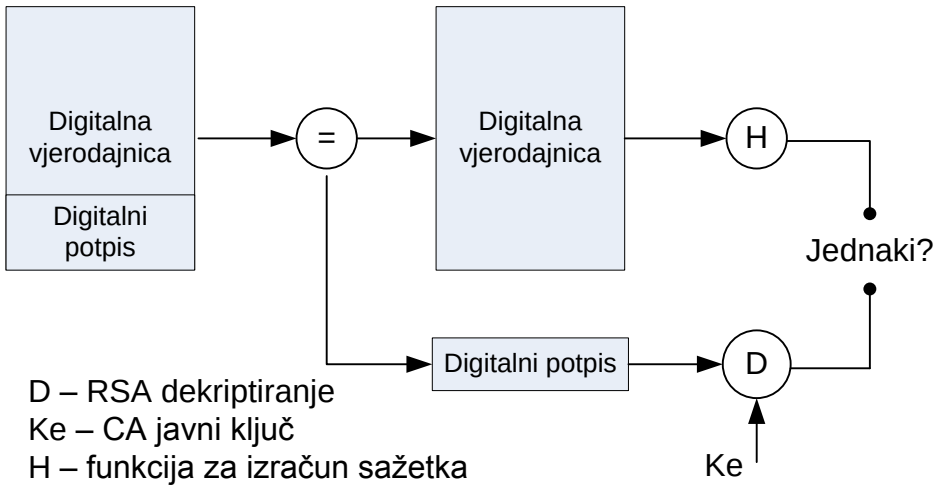
U drugom slučaju korisnici ne pripadaju istom CA elementu, pa ako korisnik A pripada pod CA element X, a korisnik B pripada pod CA element Y te korisnik A želi uspostaviti sigurni komunikacijski kanal s korisnikom B, tada mora moći

dohvatiti vjerodajnicu od korisnika B. To će učiniti tako da će za njega njegov CA element X od CA elementa Y dohvatiti digitalnu vjerodajnicu od korisnika B. Na slici 1.9 prikazan je jedan takav slučaj.



Slika 1.9 A i B ne pripadaju pod isti CA

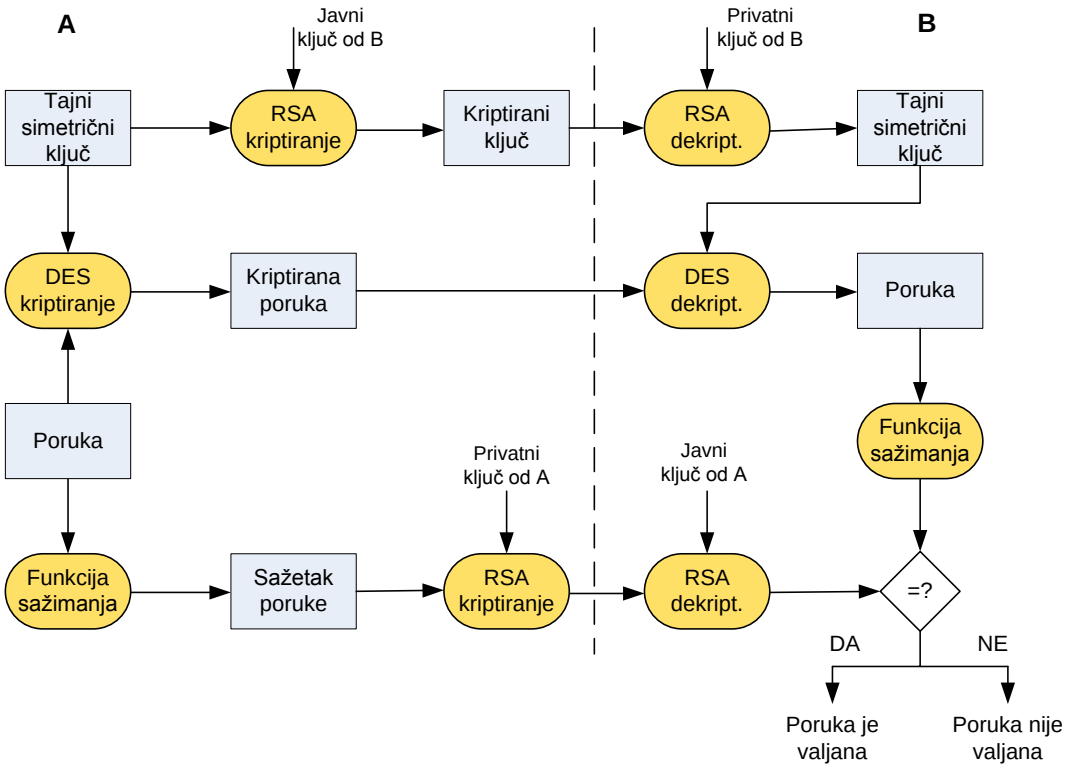
Sljedeći korak uspostave sigurnog komunikacijskog kanala jest da se korisnici jedan drugom međusobno predstave tj. moraju obaviti međusobnu autentikaciju, a to će učiniti upravo netom razmijenjenim vjerodajnicama. Ukoliko su vjerodajnice valjane tada su i informacije koje sadrže vjerodostojne. Valjanost vjerodajnica provjerava se tako da se provjeri digitalni potpis koji je stvorio CA element, a kojem korisnici A i B vjeruju tj. s njim čine lanac povjerenja.



Slika 1.10 Provjera digitalnog potpisa digitalne vjerodajnice

Na slici 1.10 prikazan je postupak provjere digitalnog potpisa digitalne vjerodajnice. Iz slike se jasno vidi da se provjera digitalnog potpisa obavlja tako da se javnim ključem od CA elementa dekriptira digitalni potpis te se dobiveni tekst usporedi s izračunatim sažetkom ostatka vjerodajnice. Ukoliko se dobivene vrijednosti podudaraju tada je vjerodajnica valjana, nitko nije mijenjao njen sadržaj. Kako nitko osim CA elementa ne zna njegov privatni ključ, samo je CA element mogao napraviti taj digitalni potpis te su stoga informacije koje sadrži vjerodajnica vjerodostojne.

Sada je sve spremno za sigurnu komunikaciju između korisnika A i B. Na slici 1.11 je prikazana takva komunikacija. Prikazane su tri poruke koje razmjenjuju korisnici A i B, a koje sadrže kriptirani simetrični tajni ključ koji se još naziva i sjednički ključ jer ga je potrebno razmijeniti samo jednom tokom jedne sjednice, nadalje prenosi se poruka koja je kriptirana simetričnim tajnim ključem i naposljetku digitalni potpis.



Slika 1.111 Sigurna komunikacija između korisnika A i B

Korisnik A simetričnim tajnim ključem kriptira poruku te je pošalje korisniku B. Zatim simetrični ključ treba razmijeniti s korisnikom B, a to će učiniti tako da javnim ključem korisnika B koristeći RSA kriptografiju kriptira simetrični ključ. Tako korisnik A može biti siguran da će tajni simetrični ključ samo korisnik B moći dekriptirati jer samo on zna svoj privatni ključ. Na kraju korisnik A stvara sažetak poruke te ga kriptira svojim privatnim ključem tj. digitalno potpisuje poruku.

Kada korisnik B primi kriptirani tajni simetrični ključ on ga dekriptira svojim privatnim ključem. Potom tim ključem dekriptira poruku i izračuna sažetak. Javnim ključem korisnika A dekriptira sažetak poruke te ga usporedi sa sažetkom kojeg je izračunao. Ukoliko su sažetci jednaki tada je poruka valjana tj. nije bila mijenjana i sigurno je stigla od korisnika A pošto jedino korisnik A zna svoj privatni ključ kojim je napravio digitalni potpis.

Kad jednom korisnici razmijene simetrični ključ tokom te sjednice oni ne moraju više razmjenjivati simetrične ključeve, nego razmjenjuju samo digitalno potpisane poruke.

Opoziv digitalnih vjerodajnica

Iako digitalne vjerodajnice imaju unaprijed definirani vremenski interval valjanosti, postoje situacije kada se više u njihovu valjanost ne može vjerovati te se te vjerodajnice moraju proglasiti nevaljanim. Taj proces poznat je kao opoziv digitalnih vjerodajnica. Postoji mnogo razloga za opoziv digitalnih vjerodajnica. Privatni ključevi mogu biti otkriveni, korisnik više nije dio infrastrukture, došlo je do promjene informacija u vjerodajnici koje se koriste za određivanje valjanosti vjerodajnice.

Opoziv vjerodajnica mora biti iniciran od strane CA elementa ili nekog drugog delegata poput RA elementa. Mehanizam koji se koristi za opozivanje vjerodajnica naziva se lista opozvanih digitalnih vjerodajnica (engl. *Certification Revocation List -CRL*). Lista opozvanih digitalnih vjerodajnica je lista koju je

generirao CA element, a sadrži jedinstvene informacije o opozvanim vjerodajnicama što omogućuje entitetima da provjere da li je vjerodajnica valjana ili ne. Element liste opozvanih digitalnih vjerodajnica mora biti serializiran, imati vremensku značku i biti digitalno potpisan od strane CA elementa. Lista opozvanih digitalnih vjerodajnica mora biti objavljena u repozitoriju kako bi bila dostupan svima. Kao repozitorij u koji se objavljuju liste objavljenih digitalnih vjerodajnica u industriji je prihvaćen *LDAP* (engl. *Lightweight Directory Access Protocol*) koji je razvijen kao komplement X.500 protokolu za Internet.

Uz korištenje liste opozvanih digitalnih vjerodajnica vezani su određeni problemi. Kad god neki korisnik želi koristiti vjerodajnicu mora se uvjeriti da ta vjerodajnica nije opozvana. Ako iz nekog razloga ne uspije izvršiti tu provjeru vjerodajnica može biti protumačena kao valjana. To znači da za efektivno korištenje infrastrukture javnih ključeva mora se imati konstantna veza s listom opozvanih digitalnih vjerodajnica. Ovaj zahtjev kontradiktoran je sa svojstvom vjerodajnica da su "samo autenticirajuće".

Drugi problem se javlja ako netko slučajno opozove krivu vjerodajnicu čime valjana vjerodajnica postaje neuporabljiva. Kako se konstantno mora provjeravati lista opozvanih digitalnih vjerodajnica javlja se potencijalna opasnost od uskraćivanja usluge ako netko onesposobi listu opozvanih digitalnih vjerodajnica. Rješenja za navedene probleme ne postoje, no industrija je prihvatila neka djelomična rješenja koja su se u praksi pokazala djelotvornima.

2.3 Standardi

Standardi koji se koriste za zapis digitalnim vjerodajnica su X.509 v1, X. 509 v2 i X509 v3. Prvi se je pojavio standard X.509 v1, koji je nedugo nakon pojavljivanja zamijenjen novim standardom X.509 v2. Standard koji je danas u upotrebi je X.509 v3. U nastavku su navedena svojstva pojedinih standarda te razlozi njihove zamjene novim standardima [18].

2.3.1 Standard X.500

X.500 standard definira konvenciju dodjele imena subjektima i svojstvima tih subjekata objedinjenim u jednom dokumentu, digitalnoj vjerodajnici. Informacija pohranjena u X.500 mapi sastoji se od nekoliko unosa. Svaki unos odnosi se na neku osobu, organizaciju ili uređaj koji imaju različita imena (engl. *Distinguished Name – DN*). Unos za neki objekt sadrži vrijednosti niza atributa koje taj objekt opisuju. Primjerice, za neku osobu unos bi se sastojao od atributa poput imena, telefonskog broja, e-mail-a i adrese. Svaki X.500 unos ima jedinstvenu strukturu imena koja se naziva *DIT* (engl. *Direct Information Tree*). DIT se sastoji od jednog korijena te neograničenog broja listova različitih imena. DN jednog unosa konstruira se spajanjem *DN*-a njegovog neposrednog prethodnika s RDN-om (engl. *Relative Distinguished Name*).

X.509 standard vjerodajnica definiran je od strane ITU-T još 1988 g. kao dio X.500 standarda za korištenje u autentikirajuće svrhe. Taj prvi X.509 standard vjerodajnica poznat je pod nazivom X.509 v1. Nakon tog uslijedile se još dvije revizije standarda, prva 1993 g. poznata pod nazivom X.509 v2 i druga 1996 g. pod nazivom X.509 v3. U tim revizijama dodana su neka nova polja koja su proširila mogućnost korištenja X.509 digitalnih vjerodajnica i o njima će biti nešto više riječi u nastavku.

2.3.2 Standard X.509 v1

X.509 v1 format nastao je 1988 g. Sadrži polja s informacijama o subjektu kome digitalna vjerodajnica pripada i o CA-*u* koji je izdao tu digitalnu vjerodajnicu. Ta polja su *verzija vjerodajnice*, *serijski broj*, *algoritam* koji je CA koristio za digitalni potpis*, imena subjekta i izdavatelja*, *interval valjanosti vjerodajnice*, *javni ključ* koji tu digitalnu vjerodajnicu povezuje sa subjektom, te *digitalni potpis izdavatelja*.

2.3.3 Standard X.509 v2

X.509 v2 proširuje X.509 v1 format vjerodajnice s dva dodatna polja. Ta dva dodatna polja su *jedinstveni identifikator subjekta* i *izdavatelja vjerodajnice*.

Jedinstveni identifikator izdavatelja je polje dodano kako bi se ime od CA moglo nanovo koristiti s vremenom. Naime, ukoliko neki CA prestane s djelovanjem te nakon nekog vremena se pojavi CA koji iz nekih razloga treba imati isto ime kao i prethodni ovim poljem se to omogućava, a da se ne naruši X.500 standard za dodjelu imena. Ovo polje sadrži opcionalni niz bitova kako bi se ime izdavatelja vjerodajnice učinilo jedinstvenim.

Jedinstveni identifikator subjekta postoji iz jednakih razloga kao i jedinstveni identifikator izdavatelja. Sadrži opcionalan niz bitova kako bi ime subjekta bilo jedinstveno.

2.3.4 Standard X.509 v3

PEM (engl. *Internet Privacy Enhanced Mail*) [32] objavljen je 1993 g. i uključuje specifikacije za infrastrukturu javnih ključeva osnovane na X.509 v1 vjerodajnici. Iskustvo je pokazalo da X.509 v1 i v2 formati vjerodajnice nisu dovoljno adekvatni iz nekoliko razloga. Pokazalo se da treba još dodatnih polja koja bi sadržavala informacije koje su potrebne za ostvarenje PEM-a. Kao odgovor na dane zahtjeve 1996 g. nastao je X.509 v3 format vjerodajnica. X.509 v3 format proširuje X.509 v2 format s poljem koje sadrži ekstenzije. Te ekstenzije uključuju podatke poput dodatnih informacija o identitetu subjekta, dodatne informacije o ključu, informacije o pravilniku, ograničenja kretanja vjerodajnice, informacije vezane uz hijerarhiju infrastrukture javnih ključeva te čak i informacije koje su svojstvene za neke zatvorene okoline koje koriste infrastrukturu javnih ključeva.

3 Računarstvo zasnovano na uslugama

Računarstvo zasnovano na uslugama (engl. *Service Oriented Computing - SOC*) je nova paradigma razvoja raspodijeljenih računalnih sustava [15]. Računarstvo zasnovano na uslugama koristi postojeće paradigme objektno-orijentiranog računarstva i računarstva zasnovanog na komponentama te omogućava izgradnju agilnih mreža poslovnih programskih sustava koji surađuju unutar i izvan granica organizacijskog okruženja. Usluge pomoću kojih se grade programski sustavi su neovisne o računalnoj platformi te su izgrađene tako da ih je moguće opisati, objaviti, pronaći te ih podešavati i njima upravljati na standardizirani način.

3.1 Mrežne usluge

Metodologija razvoja programskih sustava se s vremenom mijenjala kako bi pratila sve brži rast programskih sustava. Kako su programski sustavi postajali sve složeniji, tako je i metodologija njihova razvoja uvodila gradivne elemente sve krupnije zrnatosti. Programski gradivni elementi razvijali su se postupno, počevši od procedura, preko objekata i komponenata sve do usluga. Pojam programskog gradivnog elementa uspoređuje se s pojmom crne kutije jer poput nje skriva detalje programskog ostvarenja te upravlja pristupom ponašanju i podacima putem sučelja.

Usluga je skup ponašanja koji pruža neka komponenta na korištenje drugoj komponenti na osnovu ugovora o sučelju. Usluge pružaju mogućnost jednostavnog povezivanja i zajedničkog rada. Mrežna usluga ima sučelje kojem se pristupa putem mrežne adrese, što omogućuje njihovo dinamičko pronalaženje i korištenje.

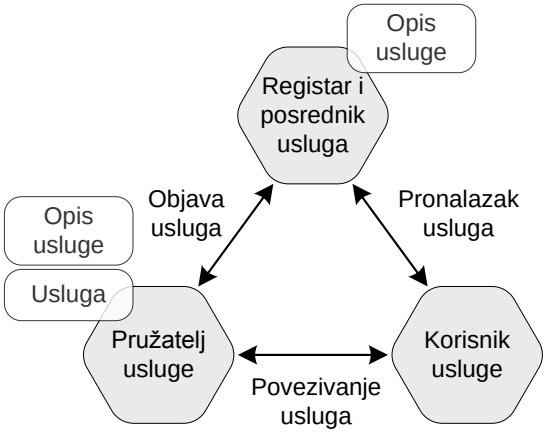
Osnovna razlika između prijašnjih i suvremenih mrežnih usluga je način njihovog korištenja. Prijašnje usluge zahtijevale su određeni zahvat korisnika putem preglednika Web stranica (engl. *web browser*). Suvremene mrežne usluge zasnivaju se na otvorenosti mrežnih sustava, odnosno izlažu svoje funkcionalnosti putem programskog sučelja koje se javno objavljuje. Na taj način je omogućeno automatizirano korištenje mrežnih usluga.

3.2 Arhitektura zasnovana na uslugama

U računarstvu zasnovanom na uslugama naglasak je na cjelokupnoj arhitekturi sustava. Tehnologije za razvoj zasebnih komponenti poput baza podataka i raznih drugih programskih komponenti razvijene su i vrlo dobro su poznate. Stoga stvarna uspješnost raspodijeljenih raznorodnih sustava ovisi o mogućnostima povezivanja postojećih tehnologija u zajednički radni okvir, odnosno cjelokupnu arhitekturu sustava. Zbog toga se nastoji postojeće tehnologije formulirati u obliku metodologija oblikovanja složenih programskih sustava. Formulirane metodologije moguće je objediniti u općenitu infrastrukturu koja olakšava i ubrzava razvoj raspodijeljenih raznorodnih sustava.

U posljednje vrijeme postignut je zamjetan napredak u razvoju standarda i alata vezanih uz mrežne usluge. Suvremene mrežne usluge opisuju se i pozivaju na standardan način. Zbog toga je također omogućeno njihovo objavljivanje i pronalazak na standardan način.

Osnovni model arhitekture zasnovane na uslugama osim samih usluga definira tri vrste sudionika i njihove međuodnose [4]. Model je prikazan na slici 3.1 Sudionici mogu biti *pružatelji mrežnih usluga, registri usluga* i *korisnici usluga*.



Slika 3.1 Vrste sudionika i njihovi međuodnosi

Pružatelji mrežnih usluga ostvaruju mrežne usluge i objavljuju ih tako da ih prijave u registar usluga. *Registri usluga* održavaju popis objavljenih usluga i omogućavaju korisnicima usluga pronalazak pružatelja usluga. *Korisnici usluga* pretražuju registar usluga za odgovarajućim pružateljima usluga, odabiru pružatelje i koriste pronađene usluge.

Web services tehnologija predstavlja ostvarenje modela arhitekture zasnovane na uslugama. Moguće je istovremeno korištenje više arhitektura zasnovanih na uslugama ostvarenih pomoću različitih tehnologija. Uvjet je da sva ostvarenja zadovoljavaju ključne zahtjeve potrebne da bi neka arhitektura bila zasnovana na uslugama. Stoga se pod pojmom arhitekture zasnovane na uslugama prvenstveno misli na određena svojstva, organizaciju, metode oblikovanja i razvoj složenih računalnih sustava [6].

Ostvarenje mrežnih usluga tehnologijom *Web services* koristi model prema kojem u određenom međudjelovanju korisnik upotrebljava jednu uslugu od jednog pružatelja. Razlog tome je da je većina programskih sustava oblikovana prema korisnik/poslužitelj arhitekturi u kojoj korisnik šalje zahtjev jednom poslužitelju i od istog dobiva odgovor.

Ključni zahtjevi nad arhitekturom zasnovanom na uslugama su *slaba povezanost*, *neutralnost platforme*, *prilagodljiva konfigurabilnost*, *postojanost*, *krupna zrnatost* i *složaj usluga* [3].

Slaba povezanost zahtijeva cjelovitost i autonomnost usluga, te oblik usluge koji ne zahtijeva informacije o internoj strukturi, mehanizmima, pravilima i podatkovnim tipovima drugih usluga. Također, ne zahtijeva se jamčenje konzistentnosti određenog podataka ili stanja tijekom međudjelovanja više usluga. Međutim, moguće je uspostavljanje ugovornih odnosa visoke razine među uslugama za postizanje konzistentnosti na razini sustava.

Neutralnost platforme zahtijeva mogućnost zajedničkog rada i povezivanja usluga samo na osnovi njihovih sučelja. Odnosno, usluge koje međudjeluju s nekom uslugom ne smiju ovisiti o detaljima njezinog programskog ostvarenja.

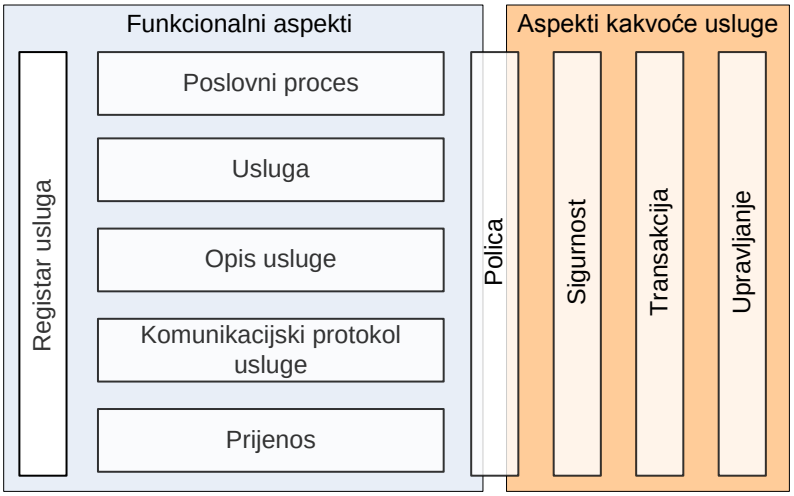
Prilagodljiva konfigurabilnost zahtijeva mogućnost konfiguriranja programskog sustava na jednostavan i prilagodljiv način. Nadalje, zahtijeva se mogućnost povezivanja usluga za vrijeme rada programskog sustava, odnosno zahtijeva se mogućnost mijenjanja konfiguracije sustava dinamički i po potrebi, bez gubitka ispravnosti.

Postojanost zahtijeva dovoljno dug vremenski period postojanja usluga koji je potreban da se otkriju važne iznimke, poduzmu mjere za ispravljanje tih iznimki, te odgovore na mjere ispravljanja iznimki koje su poduzele druge usluge. Usluge bi također trebale postojati dovoljno dugo da ih je moguće pronaći te da je moguće pouzdati se u njih i njihovo ponašanje.

Usluge se oblikuju na razini *krupne zrnatosti*. Umjesto oblikovanja akcija i međudjelovanja na detaljnoj razini, oblikuju se bitna svojstva usluga na visokoj razini koja se koriste u svrhe definiranja poslovnih ugovora među sudionicima.

Umjesto oblikovanja izračunavanja na centraliziran način, izračunavanje se oblikuje tako da ga provodi više autonomnih ravnopravnih sudionika. Za razliku od centraliziranog izračunavanja gdje jedna komponenta upravlja drugim komponentama, izračunavanje u otvorenim sustavima provodi više partnerskih usluga koje rade kao jedan tim. Stoga je umjesto individualne komponente kao u centraliziranom načinu osnovna jedinka za oblikovanje sustava *složaj usluga* (engl. *teams*). Oblikovanje zasnovano na timu usluga posljedica je razvoja arhitekture ravnopravnih sudionika (engl. *peer to peer architecture*).

Arhitektura zasnovana na uslugama sastoji se od dijelova koji se mogu podijeliti u dvije kategorije: *funkcionalnu* i *kvalitetu usluge* [21]. Na slici 3.2 prikazani su elementi arhitekture zasnovane na uslugama.



Slika 3. 2 Elementi arhitekture zasnovane na uslugama

Arhitekturni stog je podijeljen u dvije cjeline, lijevu koja pokazuje funkcionalne aspekte arhitekture i desnu koja pokazuje aspekte kakvoće usluge arhitekture.
Pod funkcionalne aspekte ubrajaju se *prijenos*, *komunikacijski protokol*, *opis usluge* i *registar usluga*. *Prijenos* (engl. *transport*) je mehanizam korišten za prenošenje zahtjeva usluge od korisnika usluge do pružatelja usluge i za prenošenje odgovora usluge od pružatelja usluge do korisnika usluge. Nadalje, *Komunikacijski protokol usluge* (engl. *service communication protocol*) je

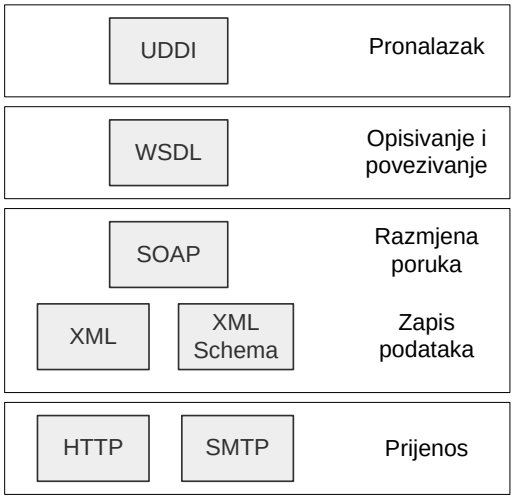
dogovorni mehanizam koji pružatelj usluge i korisnik usluge koriste za međusobnu komunikaciju. *Opis usluge* (engl. *service description*) je dogovorna shema za opisivanje što je usluga, kako ju treba pozvati, i koji podaci su potrebni da bi se usluga uspješno pozvala. Skup usluga pozvanih u određenim razmacima s određenim skupom pravila kako bi ispunile poslovne potrebe naziva se *Poslovni proces* (engl. *business process*). Poslovni proces se može i sam gledati kao usluga, pa se može koristiti u kompoziciji s drugim uslugama različite zrnatosti. *Registar usluga* (engl. *service registry*) je repozitorij usluga i opisa podataka kojeg koriste pružatelji usluga za objavljivanje usluga i korisnici usluga za pronalazak dostupnih usluga.

U aspekte kvalitete usluge ubrajaju se *polica*, *sigurnost*, *transakcija* te *upravljanje*. *Polica* (engl. *policy*) je skup pravila i uvjeta pod kojima pružatelj usluge pruža uslugu dostupnom korisnicima usluge. Postoje dijelovi police koji spadaju u funkcionalnu kategoriju kao i dijelovi police koji spadaju u kategoriju kvalitete usluge, pa se polica nalazi u oba područja – funkcionalnom i području kvalitete usluge. Nadalje, *sigurnost* (engl. *security*) je skup pravila koji se primjenjuju za identifikaciju, autorizaciju i kontrolu pristupa korisnika usluge pri pozivanju usluga. *Transakcija* (engl. *transaction*) je skup svojstava koja se mogu primijeniti na skup usluga da bi se dobio konzistentan odgovor. Na primjer, ako se koristi grupa od tri usluge pri završetku poslovne funkcije, sve moraju završiti ili niti jedna ne smije završiti. Skup svojstava koja se mogu primijeniti na pružene ili konzumirane usluge naziva se *Upravljanje* (engl. *management*).

3.2.1 *Web Services* skupina standarda

Računalna arhitektura zasnovana na uslugama koristi načela prema kojima se spajanje, komunikacija, opis i pronalazak usluga odvijaju na standardan način. Standard *Web services* ostvaruje navedena načela pomoću stoga mehanizama i protokola prikazanih na slici 3.3 [35,36]. Na najnižoj razini stoga nalaze se prijenosni protokoli koji služe za prijenos podataka među sudionicima. Mrežne

usluge razvijene su za rad na globalnoj mreži Internet, pa su za prijenosne protokole odabrani protokoli *HTTP* (engl. *HyperText Transfer Protocol*) i *SMTP* (engl. *Simple Mail Transfer Protocol*). Sljedeća razina stoga pruža zajednički način zapisivanja i oblikovanja podataka. Zadaću općenitog načina zapisivanja podataka ispunjava jezik *XML* (engl. *eXtensible Markup Language*) [7] koji je danas općeprihvaćen standard. Kao zajednički protokol za razmjenu poruka među uslugama i pozivanje metoda udaljenih usluga odabran je *Protokol za jednostavni pristup objektima* [8] (engl. *Simple Object Access Protocol*, *SOAP*) koji je zasnovan na jeziku *XML*. Sljedeća razina stoga opisuje sučelje u obliku čitljivom ostalim uslugama. Opis sučelja sastoji se od naziva metoda, njihovih parametra i rezultata izvođenja metoda. Za tu svrhu razvijen je *jezik za opisivanje mrežnih usluga* [9] (engl. *Web service description language*, *WSDL*).



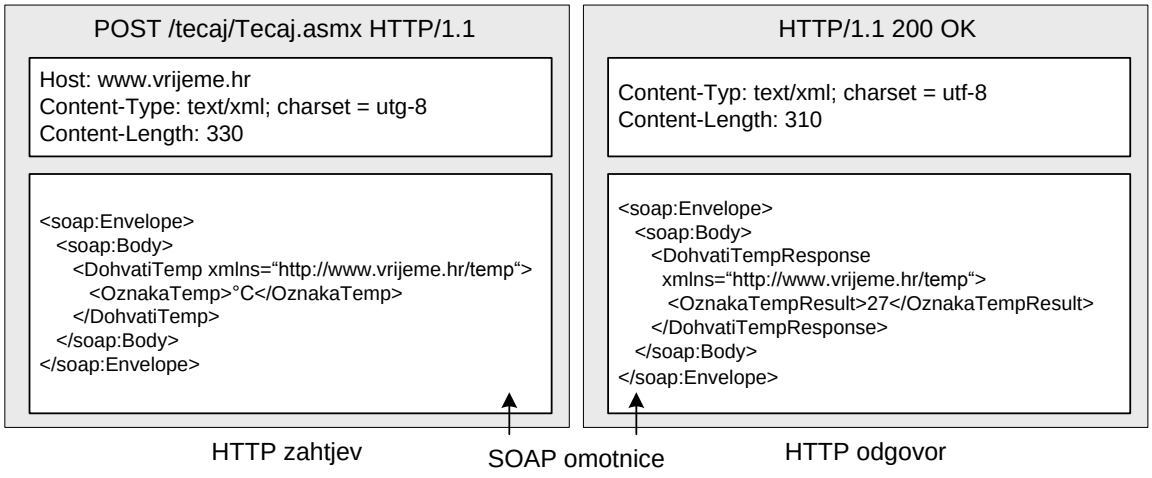
Slika 3.3 Standard Web Services

Na najvišoj razini stoga nalazi se mehanizam za pronalaženje usluga. UDDI [5] (engl. *Universal Description, Discovery, and Integration*) je standard za izgradnju registra koji sprema opise usluga i omogućuje njihovo pretraživanje.

Jezik *XML* (engl. *eXtensible Mark-up Language*) [7] je proširivi jezik zasnovan na oznakama (engl. *tags*) i svojstvima (engl. *attributes*) koji pruža tekstualni format zapisivanja podataka neovisan o računalnoj platformi. Jezik XML je podskup

jezika *SGML* (engl. *Standard Generalized Markup Language*) koji je nastao poopćenjem jezika *HTML* (engl. *HyperText Markup Language*) [28]. Dok HTML oznake određuju način prikazivanja podataka, XML oznake nose informaciju o značenju podataka. XML uvodi i dodatna pravila u odnosu na sintaksu HTML-a zbog lakšeg parsiranja i obrade. Za razliku od jezika HTML kod kojega je skup oznaka unaprijed definiran, skup oznaka XML-a ima svojstvo proširivosti, odnosno skup oznaka je moguće definirati i proširiti. XML se koristi za razne primjene jer pruža mogućnost definiranja struktura i sintakse ovisno o potrebi.

SOAP [4] je jednostavan protokol zasnovan na jeziku XML i namijenjen razmjeni poruka među ravnopravnim sudionicima u raspodijeljenoj okolini. U početku je SOAP korišten kao mehanizam udaljenog poziva procedura (engl. *Remote Procedure Call – RPC*) primjenom XML formata među umreženim računalima, no u posljednje vrijeme se razvio kao standard za razmjenu XML poruka u globalnoj mreži Internet koristeći HTTP, SMTP i SIP kao prijenosne protokole. Kao prijenosni protokol se u praksi najčešće koristi HTTP.



Slika 3.4 Primjer SOAP zahtjeva i odgovora preko HTTP protokola

Poziv udaljene mrežne usluge primjenom SOAP protokola se ostvaruje s dva tipa SOAP poruka: *SOAP zahtjevom* (engl. *request*) i *SOAP odgovorom* (engl. *response*). Primjer na slici 3.4 prikazuje poziv mrežne usluge pomoću *SOAP*

protokola. SOAP zahtjev prenosi se kao sadržaj HTTP POST zahtjeva. Slično kao i zahtjev, SOAP odgovor umetnut je u HTTP odgovor. Svaka SOAP poruka oblikovana je kao XML dokument i sastoji se od SOAP omotnice (engl. *envelope*) koja je podijeljena na dva podelementa: *SOAP zaglavlje* (engl. *header*) i *SOAP tijelo* (engl. *body*). Zaglavlja poruke najčešće sadrže podatkovne strukture definirane standardima koji koriste SOAP. Format tijela poruke ovisi o tome da li je poruka SOAP zahtjev ili SOAP odgovor, ali u oba slučaja tijelo poruke je definirano WSDL opisom i prenosi osnovne informacije o udaljenom pozivu.

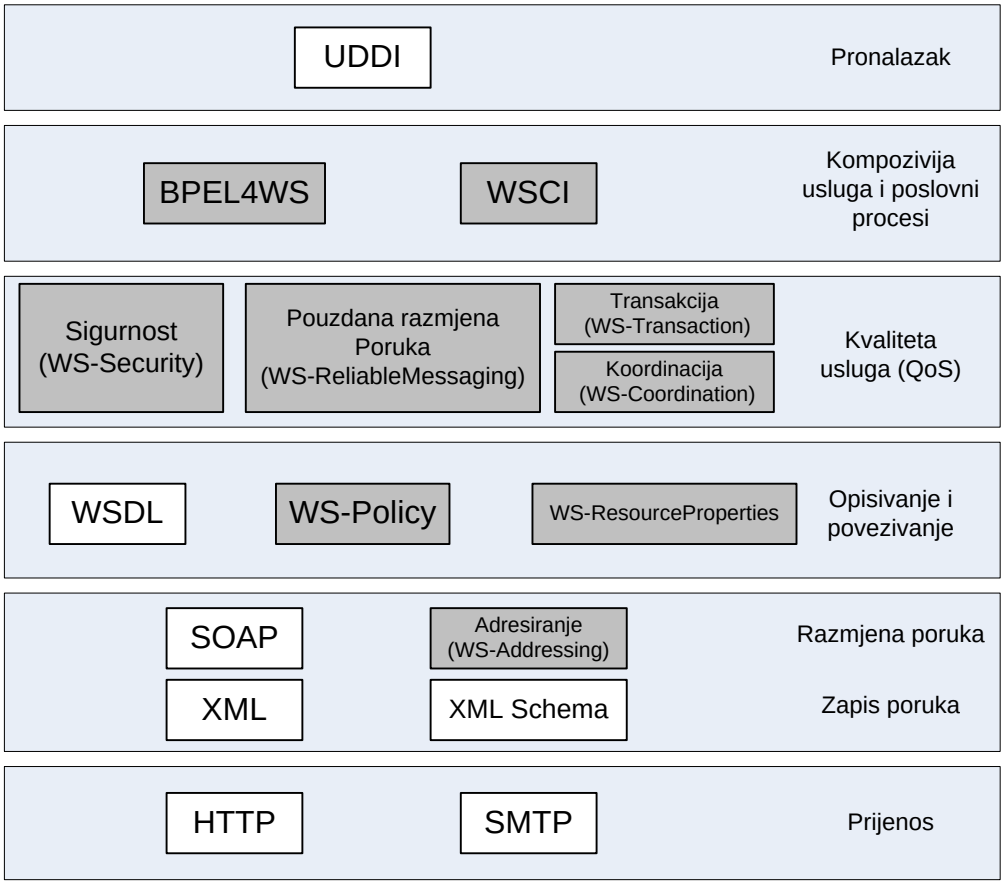
Jezik WSDL [6] se koristi za opisivanje programskog sučelja mrežne usluge. Kao i SOAP, jezik WSDL je zasnovan na XML jeziku. Opis sučelja podrazumijeva definiciju podatkovnih tipova, oblik ulaznih i izlaznih poruka, popis metoda koje usluga pruža, mrežne adrese te protokole kojima je ostvarena usluga. Struktura WSDL dokumenta dijeli se na dva dijela: sučelje usluge i ostvarenje usluge.

Standard *UDDI* (engl. *Universal Description, Discovery, and Integration*) [29] definira postupke izgradnje i korištenja registra mrežnih usluga. UDDI omogućava korisnicima objedinjen i sustavan način pronalaska pružatelja usluga putem centraliziranog registra usluga. UDDI registar pruža funkcionalnost automatiziranog *on-line imenika* mrežnih usluga.

3.2.2 Prošireni skup standarda WS-*

Osnovni mehanizmi mrežnih usluga omogućuju opisivanje, objavu i međudjelovanje usluga na standardan način. Međutim, mnogi raspodijeljeni programski sustavi zahtijevaju razna složenija ponašanja usluga poput *sigurnosti* (engl. *security*), *pouzdanosti* (engl. *reliability*) i *kompozicije usluga* (engl. *service composition*). Zato se u posljednje vrijeme puno ulaže u razvoj specifikacija više razine kojima je cilj omogućiti ostvarivanje složenog ponašanja usluga na definiran i standardiziran način. Teži se razvoju specifikacija koje nisu usko vezane uz neko posebno područje primjene kako bi ih bilo moguće koristiti za

široki skup primjena. Dosad su razvijene brojne specifikacije, a najznačajnije su specifikacija jezika za opisivanje izvođenja poslovnih procesa (engl. *Business process execution Language for Web services*, BPEL4WS) koji se koristi za kompoziciju usluga, specifikacija za koordinaciju usluga (engl. *WS-Coordination*), specifikacija koja definira transakcije, odnosno čvrsto međudjelovanje više usluga (engl. *WS-Transacation*), specifikacija za sigurni zajednički rad mrežnih usluga (engl. *WS-Security*) i specifikacija za pouzdanu razmjenu poruka (engl. *WS-ReliableMessaging*). Prošireni stog WS-* standarda prikazan je na slici 3.5.



Slika 3.5 Prošireni stog WS-* standarda

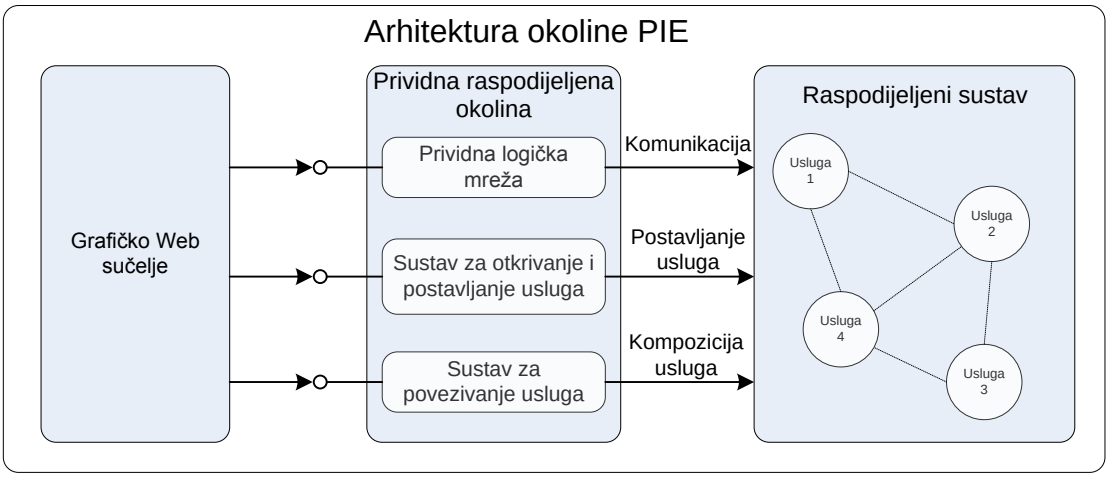
Svi navedeni aspekti ključni su elementi za značajnije poslovno međudjelovanje. Razvijen je i radni okvir WS-Policy koji definira skup pravila za opisivanje informacija o kvaliteti, mogućnostima, zahtjevima i svojstvima mrežne usluge.

3.2.3 WS-Security

Standard *WS-Security* [17] dio je proširenog skupa standarda WS-* čija je namjena omogućavanje uspostave sigurne razmjene SOAP poruka. Standard *WS-Security* je prilagodljiv i zamišljen je kao osnova za daljnju konstrukciju raznih sigurnosnih modela uključujući infrastrukturu javnih ključeva, Kerberos i SSL. Nadalje, WS-Security standard pruža podršku za višestruke sigurnosne značke (engl. *Security tokens*), višestruke formate digitalnih potpisa, i višestruke metode kriptiranja. Specifikacija osigurava tri glavna mehanizma: razmjenu sigurnosnih znački, integritet poruke te povjerljivost poruke. Ovi mehanizmi kao takvi ne osiguravaju kompletno sigurnosno rješenje, WS-Security je građevni blok koji tek u kombinaciji s ostalim standardima iz WS-* proširenog skupa i aplikacijskim protokolima ostvaruje razne sigurnosne modele i metode kriptiranja.

3.3 Programirljiva internetska okolina

Programirljiva internetska okolina (engl. *Programmable Internet Environment-PIE*) [6] ostvaruje programski model zasnovan na uslugama te je u potpunosti ostvarena primjenom *Web Services* tehnologije. Okolina PIE korisnicima omogućuje izgradnju i izvođenje raspodijeljenih primjenskih sustava zasnovanih na programskom modelu zasnovanom na uslugama. Oblikovanje raspodijeljenih primjenskih sustava za okolinu PIE zasniva se na procesima *prividnosti* (engl. *virtualization*) i *kompozicije* (engl. *composition*). Postupkom prividnosti računalna sredstva se omataju sučeljima usluge koje se zatim postupkom kompozicije organiziraju u raspodijeljeni primjenski sustav. Uloga okoline PIE je pružiti potporu osnovnim procesima izgradnje i izvođenja primjenskih sustava.



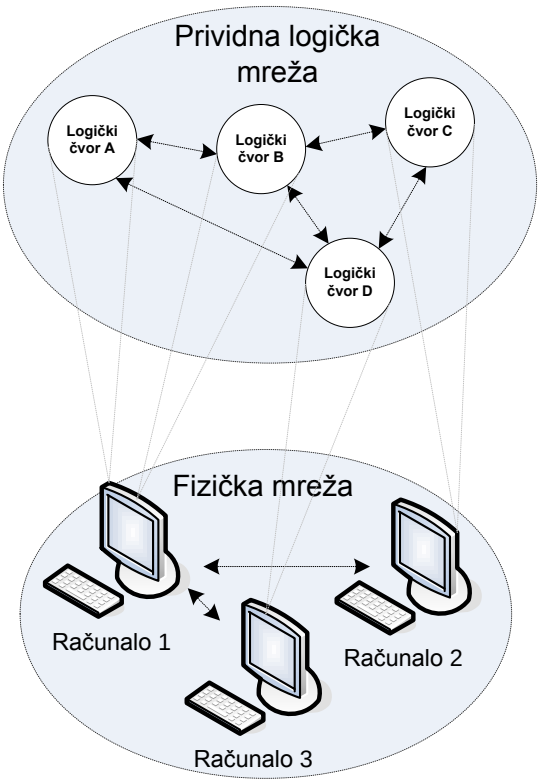
Slika 3.6 Arhitektura okoline PIE

Na slici 3.6 prikazana je arhitektura okoline PIE. Arhitektura okoline PIE sastoji se od dva osnovna dijela: *grafičkog Web sučelja* i *prividne raspodijeljene okoline*. *Prividna raspodijeljena okolina* sastoji se od tri slojevito postavljena podsustava: *Prividne logičke mreže*, *Sustava za otkrivanje i postavljanje usluga* i *Sustava za povezivanje usluga*. Podsustav koji se nalazi na višoj razini koristi funkcionalnosti podsustava na nižim razinama. *Prividna logička mreža* [10] omogućuje logičko adresiranja usluga. Sustav *za otkrivanje i postavljanje usluga* [11] ostvaruje funkcionalnost registra usluga te dodatno omogućuje postavljanje i uklanjanje usluga s čvorova *Prividne mreže*. *Sustav za povezivanje usluga* [11] ostvaruje osnovna načela programskog modela zasnovanog na uslugama, kao što su sinkronizacijske i komunikacijske usluge, te prevođenje i izvođenje raspodijeljenih programa. *Grafičko Web sučelje* [12] omogućuje potpomagano (engl. *wizard-based*) pisanje raspodijeljenih programa te pokretanje postupka njihovog prevođenja i izvođenja.

3.3.1 Prividna logička mreža

Osnova arhitekture okoline PIE je *Prividna logička mreža* (engl. *Overlay network*) [10] koja se nastavlja na fizičku računalnu mrežu. *Prividna logička mreža* ostvaruje nezavisnost raspodijeljenog primjenskog sustava o raznorodnoj fizičkoj infrastrukturi te omogućava adresiranje i komunikaciju usluga u logičkom

komunikacijskom prostoru. Navedena nezavisnost ostvarena je uvođenjem načela logičkog čvora čiji komunikacijski mehanizmi načelom prividnosti zamjenjuju komunikacijske mehanizme fizičkog računala. Slika 3.7 prikazuje primjer računalne mreže s tri računala koji se primjenom *Prividne logičke* mreže preslikavaju u logičku mrežu s četiri logička čvora.



Slika 3.7 Prividna logička mreža okoline PIE

Logički čvorovi mogu nesmetano ulaziti i izlaziti iz *Prividne logičke mreže*. Mogućnost postavljanja topologije *Prividne logičke* mreže omogućuje prenosivost raspodijeljene aplikacije s jednog skupa računala na drugi. Uvođenjem logičkih čvorova ostvaruje se nezavisnost komunikacije među uslugama o fizičkoj računalnoj mreži. Osim toga, uvođenje logičkih čvorova omogućuje logičko adresiranje usluga smještenih na logičkim čvorovima. Tako je svaka usluga raspodijeljenog sustava jedinstveno određena imenom logičkog čvora na kojem se nalazi, te imenom usluge na tom čvoru. Logičko adresiranje usluga i čvorova ostvaruje se primjenom *WS-Addressing* standarda [30]. Koristeći opisani način

adresiranja, *Prividna logička mreža* obavlja zadaću usmjeravanja zahtjeva između usluga, a osim toga zadužena je i za sigurnost i tajnost komunikacije koja prolazi sustavom.

3.3.2 Sustav za otkrivanje i postavljanje usluga

Na *Prividnu logičku mrežu* nadovezuje se *Sustav za otkrivanje i postavljanje usluga* [11]. Otkrivanje sredstava nužna je funkcionalnost za izgradnju i izvođenje raspodijeljenih primjenskih sustava u dinamičkim okolinama. *Sustav za otkrivanje i postavljanje usluga* pruža funkcionalnost registra usluga u koji se spremaju lokacije aktivnih usluga u *Prividnoj logičnoj mreži*. *Sustav za otkrivanje i postavljanje usluga* automatizira postupak postavljanja usluga na logičke čvorove prividne logičke mreže. Automatizacija postupka postavljanja usluga smanjuje složenost postupka postavljanja primjenskih sustava te pruža potporu upravljanju opterećenjem raspodijeljenog primjenskog sustava.

3.3.3 Sustav za povezivanje usluga

Sustav za povezivanje usluga [8] pruža potporu oblikovanju i izvođenju složenih raspodijeljenih primjenskih sustava primjenom programskog modela zasnovanog na uslugama. *Sustav za povezivanje usluga* ostvaruje sinkronizacijske i komunikacijske usluge te usluge prevođenja i izvođenja raspodijeljenih programa. Prevođenje i izvođenje raspodijeljenih programa podržano je skupom mrežnih usluga koje ostvaruju red programa, SSCL prevoditelje, red poslova te CL interpretatore. Raspodijeljeni programi pisani u jeziku *SSCL* (engl. *Simple Service Composition Language*) [13] predaju se *Sustavu za povezivanje usluga* na izvođenje putem *grafičkog Web sučelja.* Dostupne usluge sinkronizacije i komunikacije su *binarni* i *opći semafor, poštanski pretinac te usmjernik događaja* [14].

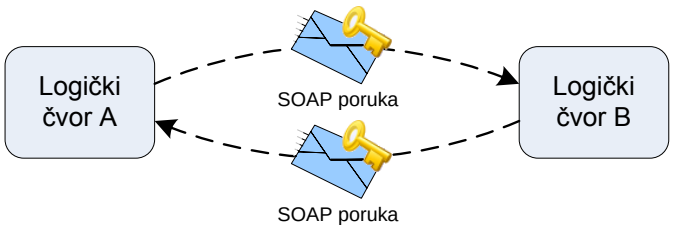
3.3.4 Grafičko Web sučelje

Grafičko Web sučelje okoline PIE [12] zasnovano je na skupu Web stranica kojima se pristupa putem Web preglednika. Grafičko Web sučelje okoline PIE omogućuje jednostavno upravljanje funkcionalnostima ostalih podsustava okoline. Web sučelje omogućuje pregledavanje, dodavanje i uklanjanje pojedinih fizičkih čvorova i logičkih adresa u *Prividnu logičku mrežu*. Putem Grafičkog Web sučelja omogućeno je postavljanje elemenata prividne raspodijeljene okoline na korisničko računalo te pokretanje i podešavanje pojedinih podsustava. Veza *grafičkog Web sučelja* sa *Sustavom za otkrivanje i postavljanje usluga* omogućuje pregledavanje, dodavanje i uklanjanje primjenskih usluga te pregledavanje, postavljanje, stvaranje i uklanjanje usluga sinkronizacije i komunikacije. *Grafičko Web sučelje* omogućuje pisanje raspodijeljenih programa primjenom jezika SSCL te pokretanje njihovog prevođenja i izvođenja primjenom usluga *Sustava za povezivanje usluga*.

4 Arhitektura PKI sustava u okolini PIE

Za osiguranje sigurnog komunikacijskog kanala u okolini PIE koristi se standard WS-Security. Logički čvorovi međusobno razmjenjuju SOAP poruke proširene sigurnosnim dodatkom u skladu s WS-Security standardom. Dodavanjem sigurnosnog dodatka SOAP porukama jamči se povjerljivost i vjerodostojnost poruke. Sadržaj sigurnosnog dodatka SOAP poruke razlikuje se ovisno o primijenjenom sigurnosnom protokolu. U okolini PIE dostupna su dva sigurnosna protokola: *MOSS* (eng . *Message Oriented Security Service*) i *COSS* (engl. *Converstation Oriented Security Service*).

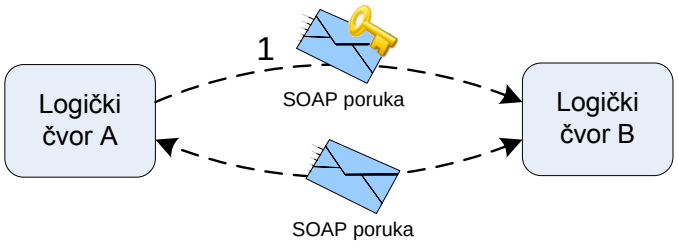
Na slici 4.1 prikazan je *MOSS* protokol. Svaka poruka koja se razmjenjuje u sebi sadrži tajni ključ kojim je obavljeno kriptiranje određenih elemenata SOAP poruke. Kako bi se osigurao sigurni prijenos tajnog ključa, on je kriptiran privatnim ključem pošiljatelja. Kada primatelj primi poruku, on javnim ključem pošiljatelja dekriptira tajni ključ te ga iskoristi za dekriptiranje kriptiranih dijelova SOAP poruke. Javni ključ pošiljatelja dostupan je primatelju iz digitalne vjerodajnice pošiljatelja koju je dobavio prije uspostavljanja sigurnog komunikacijskog kanala. Za vrijeme komunikacije MOSS protokolom pošiljatelj i primatelj pretpostavljaju da posjeduju potrebne digitalne vjerodajnice te da su iste vjerodostojne.



Slika 4.1 MOSS sigurnosni protokol

Na slici 4.2 prikazan je COSS protokol. Za razliku od MOSS protokola u kojem je svaka poruka sa sobom nosi i kriptirani tajni ključ kojim su kriptirani njeni

podatkovni elementi, u COSS protokolu tajni ključ se razmijeni prije početka komunikacije i vrijedi cijelo vrijeme trajanja komunikacije. Iz tog razloga takav tajni ključ naziva se još i sjednički ključ. Kao i kod MOSS protokola, tajni ključ se mora razmijeniti tako da ostane poznat samo sudionicima komunikacije pa stoga pošiljatelj svojim privatnim ključem kriptira sjednički ključ i šalje ga primatelju. Primatelj dekriptira i sjednički ključ koristeći se javnim ključem pošiljatelja i sprema ga za kasniju upotrebu tijekom komunikacije. Slično kao i kod MOSS protokola u komunikaciji COSS protokolom pošiljatelj i primatelj pretpostavljaju da posjeduju potrebne digitalne vjerodajnice te da su one valjane.



Slika 4.2 COSS sigurnosni protokol

MOSS i COSS protokoli pretpostavljaju da su logički čvorovi prije uspostavljanje komunikacije postali dio infrastrukture javnih ključeva okoline PIE. TO znači da su čvorovi pribavili svoje digitalne vjerodajnice u kojima su zapisani njihovi javni i privatni ključevi, da su dobavili digitalnu vjerodajnicu drugog sudionika komunikacije, da je ta vjerodajnica vjerodostojna i da jamči identitet drugog sudionika.

Mehanizam uspostave infrastrukture javnih ključeva u okolini PIE ostvaruju dva elementa: *Upravitelj digitalnim vjerodajnicama* (engl. *Certificate Manager*) i *Poslužitelja digitalnih vjerodajnica* (engl. *Certificate Service*). U okviru ovog diplomskog rada programski su ostvareni *Upravitelj digitalnim vjerodajnicama* i *Poslužitelj digitalnih vjerodajnica*.

4.1 Elementi PKI sustava u okolini PIE

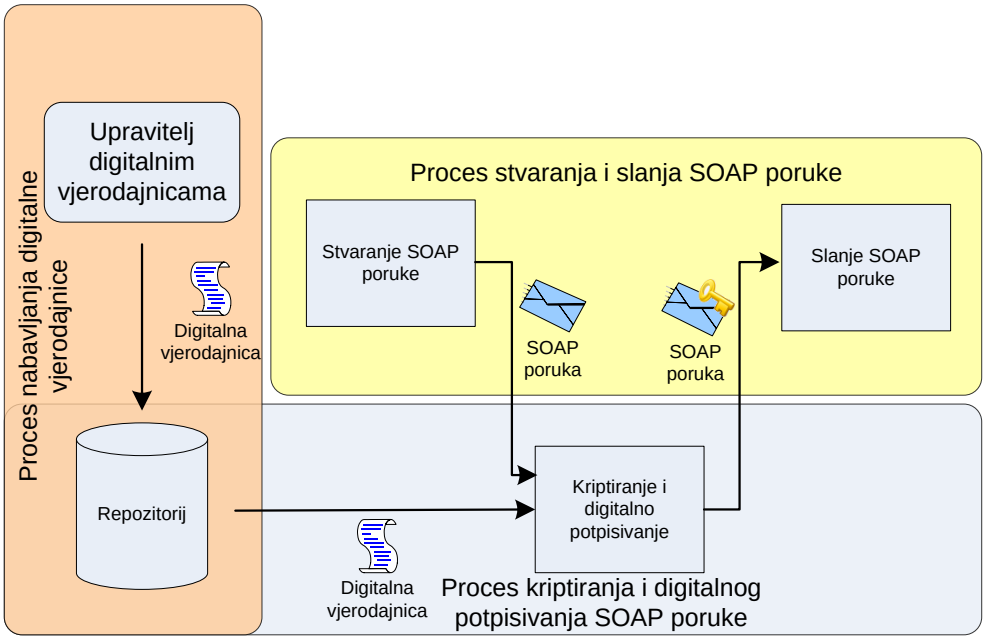
Međudjelovanjem Upravitelja digitalnim vjerodajnicama i Poslužitelja digitalnim vjerodajnicama ostvareni su procesi stvaranja, razmjene i opoziva digitalnih vjerodajnica logičkih čvorova. U nastavku poglavlja objašnjenje su njihove funkcije i način objedinjavanja s okolinom PIE.

4.1.1 Upravitelj digitalnim vjerodajnicama

Upravitelj digitalnim vjerodajnicama (engl. *Certificate Manager*) obnaša funkciju RA (engl. *Registration Authority).* On predstavlja sučelje između logičkog čvora i CA-a. *Upravitelj digitalnim vjerodajnicama* sudjeluje u procesu uključivanja logičkog čvora u infrastrukturu javnih ključeva te sudjeluje u procesima stvaranja, razmjene i opoziva digitalnih vjerodajnica logičkih čvorova.

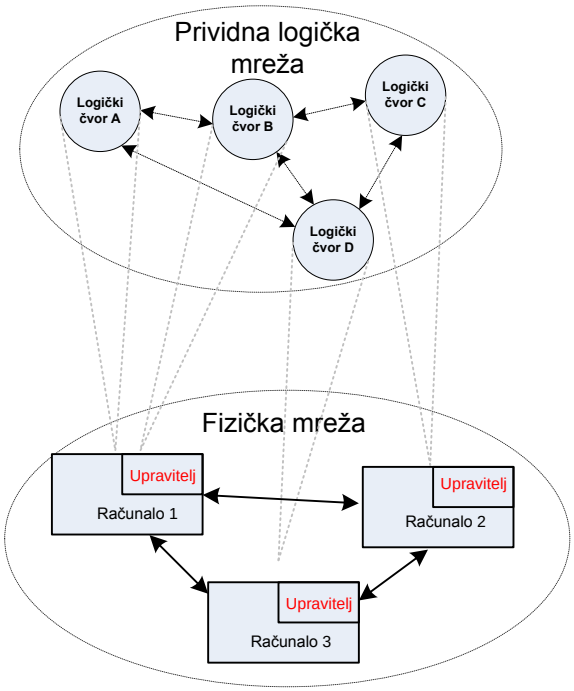
Zadaća *Upravitelja digitalnim vjerodajnicama* jest da u ime logičkog čvora stvara zahtjev za digitalnom vjerodajnicom, generira pripadajući javni i privatni ključ, šalje zahtjev za digitalnom vjerodajnicom do CA-u koji ga digitalno potpisuje i vraća natrag. *Upravitelj digitalnim vjerodajnicama* dohvaća digitalnu vjerodajnicu od CA te ju zajedno s pripadajućim privatnim ključem sprema u lokalni repozitorij. Prilikom razmjene vjerodajnica s drugim logičkim čvorom, *Upravitelj digitalnim vjerodajnicama* provjerava vjerodostojnost zaprimljene digitalne vjerodajnice koristeći pritom digitalnu vjerodajnicu CA-a. Ako primljena vjerodajnica nije vjerodostojna, *Upravitelj digitalnim vjerodajnicama* sprječava uspostavljanje komunikacijskog kanala između logičkih čvorova. Nadalje, *Upravitelj digitalnim vjerodajnicama* sudjeluje u procesu opoziva digitalne vjerodajnice logičkog čvora pri čemu u unaprijed definiranim vremenskim razmacima kontaktira CA kako bi provjerio nalazi li se vjerodajnica u *Listi opozvanih digitalnih vjerodajnica* (engl. *Certificate Revocation List, CRL*). Ako se ona nalazi u toj listi sprječava uspostavljanje komunikacijskog kanala.

Slika 4.3 prikazuje suradnju između *Upravitelja digitalnim vjerodajnicama* i sigurnosnih protokola MOSS i COSS. *Upravitelj digitalnim vjerodajnicama* dio je procesa nabavljanja digitalne vjerodajnice, koji je zadužen da se u repozitoriju nalazi vjerodostojna digitalna vjerodajnica logičkog čvora. Sigurnosni protokoli dio su procesa kriptiranja i digitalnog potpisivanja SOAP poruke te za tu svrhu koriste digitalnu vjerodajnicu iz repozitorija.



Slika 4.3 Veza Upravitelja digitalnim vjerodajnicama i sigurnosnih protokola

Upravitelj digitalnim vjerodajnicama nalazi se na svakom fizičkom čvoru okoline PIE i poslužuje logičke čvorove koji pripadaju dotičnom fizičkom čvoru. Na slici 4.4. prikazana je veza između fizičkih čvorova, logičkih čvorova i *Upravitelja digitalnim vjerodajnicama*. Sve logičke čvorove smještene na istom fizičkom čvoru poslužuje isti *Upravitelj digitalnim vjerodajnicama*. Logičke čvorove *A* i *B* poslužuje *Upravitelj digitalnim vjerodajnicama* koji se nalazi na fizičkom čvoru *Računalo 1*. Logički čvor *D* poslužuje *Upravitelj digitalnim vjerodajnicama* koji se nalazi na fizičkom čvoru *Računalo 3*, dok logički čvor C poslužuje *Upravitelj digitalnim vjerodajnicama* koji se nalazi na fizičkom čvoru *Računalo 2*.



Slika 4.4 Veza logičkih čvorova i Upravitelja digitalnim vjerodajnicama

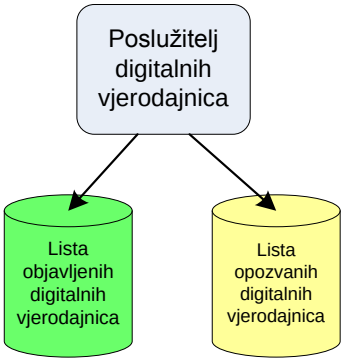
Jedno od glavnih svojstava prividne logičke mreže jest nezavisnost o fizičkoj mreži. Logička mreža može promijeniti dodijeljeni skup fizičkih čvorova bez da to utječe na radna svojstva. Slijedeći tu politiku *Upravitelj digitalnim vjerodajnicama* prilikom promijene skupa fizičkih čvorova iz lokalnog repozitorija uklanja zaostale digitalne vjerodajnice logičkih čvora te ih seli na novi skup fizičkih čvorova.

4.1.2 Poslužitelj digitalnih vjerodajnica

Poslužitelj digitalnih vjerodajnica obnaša funkciju CA-a u infrastrukturi javnih ključeva okoline PIE. Sudjeluje u procesu pristupanja logičkog čvora infrastrukturi javnih ključeva na način da poslužuje zahtjeve *Upravitelja digitalnim vjerodajnicama* za izdavanjem u procesu opoziva digitalnih vjerodajnica.

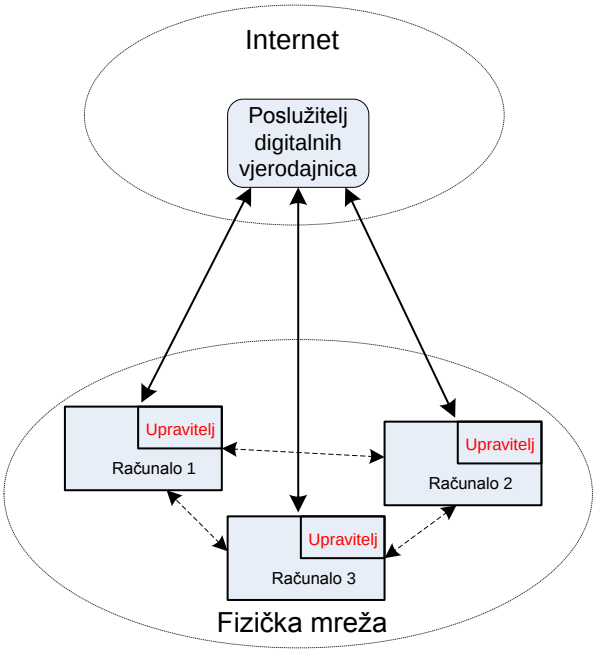
Poslužitelj digitalnih vjerodajnica na početku generira vlastitu digitalnu vjerodajnicu s pripadajućim javnim i privatnim ključem. Vlastitu digitalnu vjerodajnicu kasnije koristi za digitalno potpisivanje zaprimljenih zahtjeva za izdavanjem digitalnih vjerodajnica logičkim čvorovima. Digitalnim potpisima *Poslužitelj digitalnih vjerodajnica* jamči identitet logičkih čvorova čije je digitalne

vjerodajnice potpisao. Nadalje, sve digitalne vjerodajnice koje je koje digitalno potpisao pohranjuje u listu objavljenih digitalnih vjerodajnica kako bi ih mogao ponovno izdati ako logički se logički čvor premjesti na neki drugi fizički čvor. *Poslužitelj digitalnih vjerodajnica* stvara i održava listu opozvanih vjerodajnica. Arhitektura *Poslužitelja digitalnih vjerodajnica* prikazana je na slici 4.5.



Slika 4.5 Poslužitelj digitalnih vjerodajnica

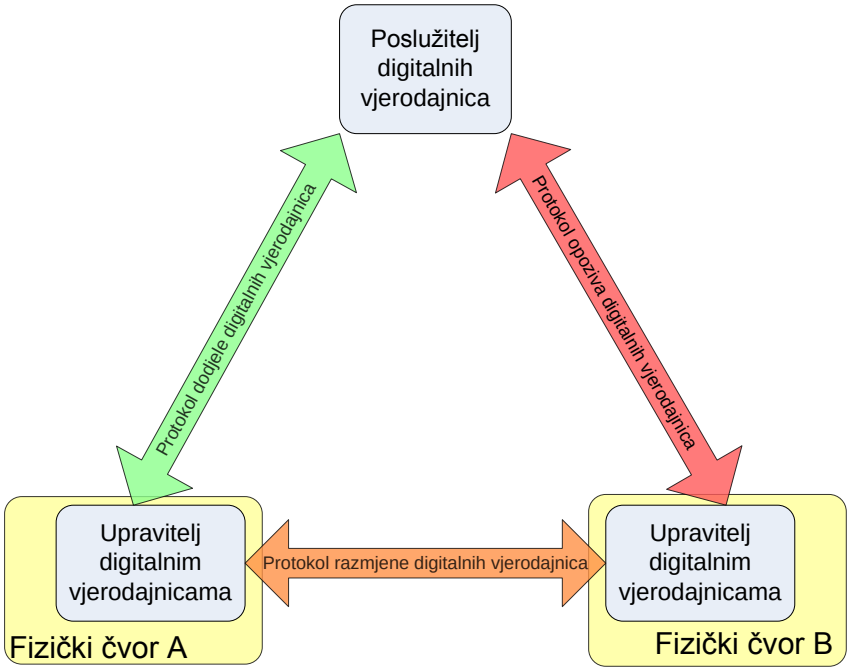
Na slici 4.6 je prikazano objedinjavanje *Poslužitelja digitalnih vjerodajnica* s okolinom PIE. *Poslužitelj digitalnih vjerodajnica* ne pripada infrastrukturi okoline PIE, nego se izvodi kao nezavisna mrežna usluga dostupna u mreži.



Slika 4.6 Poslužitelj digitalnih vjerodajnica

4.2 Protokoli uspostave i održavanja PKI sustava u okolini PIE

Uspostava i održavanje infrastrukture javnih ključeva u okolini PIE može se podijeliti na tri pod-protokola. *Protokol dodjele digitalnih vjerodajnica* je prvi protokol koji se izvršava. Tim protokolom svaki logički čvor dobiva svoju digitalnu vjerodajnicu čime postaje dio infrastrukture javnih ključeva okoline PIE. *Protokol razmjene digitalnih vjerodajnica* je protokol koji se odvija neposredno prije uspostavljanja sigurnog komunikacijskog kanala. Tim protokolom logički čvorovi međusobno razmjenjuju digitalne vjerodajnice. Treći protokol je *protokol opoziva digitalnih vjerodajnica* u kojem se vrši opoziv digitalnih vjerodajnica.



Slika 4.7 Protokoli Infrastrukture javnih ključeva

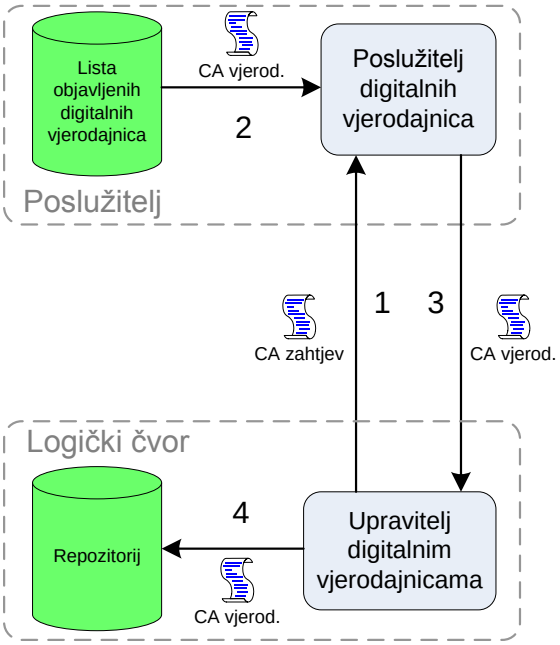
Na slici 4.7 prikazano je koji protokoli se odvijaju između kojih entiteta. *Protokol dodjele digitalnih vjerodajnica* odvija se između *Upravitelja digitalnim vjerodajnicama* i *Poslužitelja digitalnih vjerodajnica*. Nadalje, *protokol razmjene digitalnih vjerodajnica* odvija se između dvaju *Upravitelja digitalnim vjerodajnica* koji se nalaze na različitim fizičkim čvorovima. *Protokol opoziva digitalnih*

vjerodajnica odvija se između *Upravitelja digitalnim vjerodajnicama* i *Poslužitelja digitalnih vjerodajnica*. U nastavku poglavlja su podrobnije objašnjena sva tri protokola.

4.2.1 Protokol dodjele digitalnih vjerodajnica

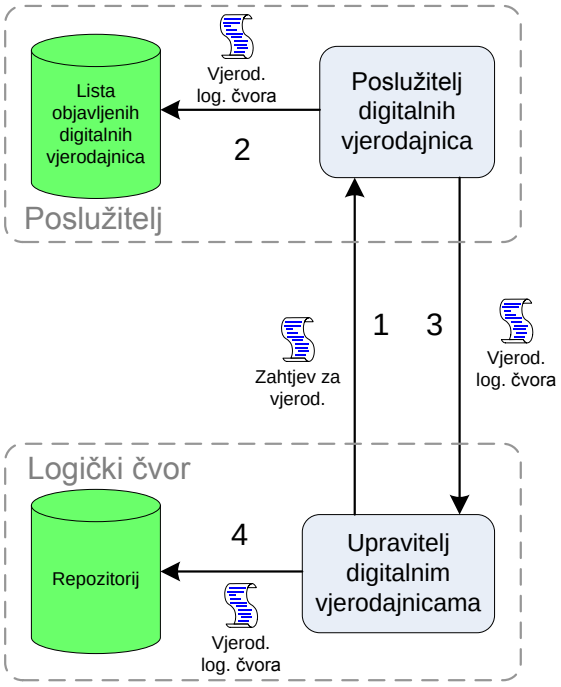
Protokol dodjele digitalnih vjerodajnica odvija se prilikom pristupanja logičkog čvora prividnoj logičkoj mreži pri čemu logički čvor automatski postaje i dio infrastrukture javnih ključeva okoline PIE. *Protokol dodjele digitalnih vjerodajnica* može se podijeliti na dva koraka: prvi korak u kojem *Upravitelj digitalnim vjerodajnicama* dohvaća vjerodajnicu *Poslužitelja digitalnih vjerodajnica* dok u drugom koraku *Upravitelj digitalnim vjerodajnicama* stvara digitalnu vjerodajnicu logičkog čvora.

Na slici 4.8 prikazan je prvi korak protokola u kojem *Upravitelj digitalnim vjerodajnicama* dohvaća digitalnu vjerodajnicu *Poslužitelja digitalnih vjerodajnica*. *Upravitelj digitalnim vjerodajnicama* prvo šalje zahtjev za digitalnom vjerodajnicom *Poslužitelja digitalnih vjerodajnica* (1). *Poslužitelj digitalnih vjerodajnica* potom dohvaća vlastitu vjerodajnicu iz liste objavljenih digitalnih vjerodajnica (2) i šalje ju *Upravitelju digitalnim vjerodajnicama* koji je poslao zahtjev (3). *Upravitelj digitalnim vjerodajnicama* dobivenu vjerodajnicu sprema u vlastiti repozitorij (4). Digitalna vjerodajnica *Poslužitelja digitalnih vjerodajnica* koristi se za provjeru vjerodostojnosti digitalnih vjerodajnica ostalih logičkih čvorova koji pripadaju istoj infrastrukturi javnih ključeva. Ona predstavlja vrh lanca povjerenja infrastrukture javnih ključeva okoline PIE.



Slika 4.8 Dohvaćanje digitalne vjerodajnice Poslužitelja digitalnih vjerodajnica

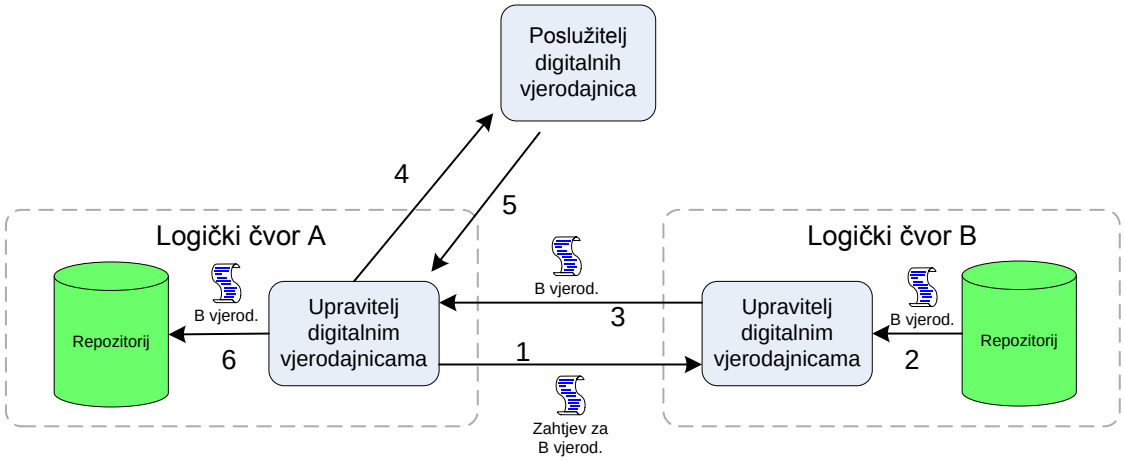
Na slici 4.9 prikazan je drugi korak protokola dodjele digitalnih vjerodajnica logičkim čvorovima. *Upravitelj digitalnim vjerodajnicama* generira zahtjev za digitalnom vjerodajnicom. Nadalje, uz zahtjev generira privatni i javni ključ koje će logički čvor koristiti prilikom postupka uspostavljanja sigurnog komunikacijskog kanala s udaljenim logičkim čvorovima. Generirani zahtjev za digitalnom vjerodajnicom *Upravitelj digitalnim vjerodajnicama* šalje do *Poslužitelja digitalnih vjerodajnica* (1). *Poslužitelja digitalnih vjerodajnica* potom digitalno potpisuje primljeni zahtjev svojim privatnim ključem, stvarajući tako vjerodostojnu digitalnu vjerodajnicu logičkog čvora. Nakon potpisivanja pristiglog zahtjeva, *Poslužitelj digitalnih vjerodajnica* sprema vjerodajnicu u listu objavljenih digitalnih vjerodajnica (2) kako bi je kasnije mogao ponovno izdati ukoliko to bude potrebno. Nadalje, *Poslužitelj digitalnih vjerodajnica* vraća potpisanu vjerodajnicu natrag do *Upravitelja digitalnim vjerodajnicama* koji je poslao zahtjev (3). Po primitku digitalno popisane vjerodajnice, *Upravitelj digitalnim vjerodajnicama* pohranjuje vjerodajnicu u odgovarajući repozitorij čineći je tako dostupnom sigurnosnim protokolima (4).



Slika 4.9 Stvaranje digitalne vjerodajnice logičkog čvora

4.2.2 Protokol razmjene digitalnih vjerodajnica

Protokol razmjene digitalnih vjerodajnica odvija se neposredno prije prve komunikacije između dvaju logičkih čvorova. Ovim protokolom razmjenjuju se digitalne vjerodajnice te samim time i informacije koje povezuju logičke čvorove s njihovim javnim ključevima i dokazuju njihov identitet.



Slika 4.10 Protokol razmijene digitalnih vjerodajnica

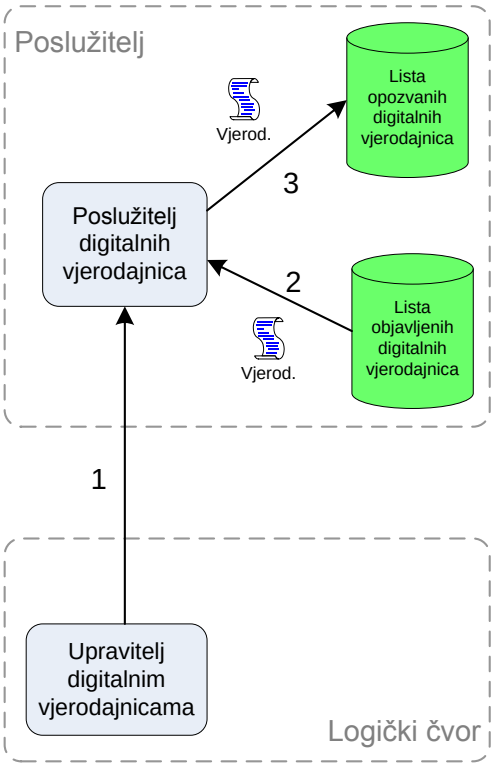
Slika 4.10 prikazuje protokol razmjene digitalnih vjerodajnica. Logički čvor A i logički čvor B žele međusobno uspostaviti sigurni komunikacijski kanal. Kako bi bili uvjereni u istinitost identiteta drugog sudionika komunikacije i imali pristup njegovom javnom ključu trebaju razmijeniti digitalne vjerodajnice. Javni ključ drugog sudionika je potreban kako bi se moglo obaviti provjera digitalno potpisanih elemenata pristigle SOAP poruke te kriptiranje elemenata odlazne SOAP poruke. *Upravitelj digitalnim vjerodajnicama* koji poslužuje logički čvor A šalje zahtjev za digitalnom vjerodajnicom *Upravitelju digitalnim vjerodajnicama* koji poslužuje logički čvor B (1). *Upravitelj digitalnim vjerodajnicama* koji poslužuje logički čvor B iz svog repozitorija dohvaća (2) te šalje natrag digitalnu vjerodajnicu logičkog čvora B (3), ukoliko se ona nalazi u zadanom repozitoriju. Po primitku digitalne vjerodajnice logičkog čvora B, *Upravitelj digitalnim vjerodajnicama* koji poslužuje logički čvor A mora utvrditi njenu vjerodostojnost. To će učiniti tako da koristeći javni ključ iz digitalne vjerodajnice *Poslužitelja digitalnih vjerodajnica* koju je primio prilikom stvaranja provjeri digitalni potpis upravo pristigle vjerodajnice. Ako je digitalni potpis valjan *Upravitelj digitalnim vjerodajnicama* prijelazi na idući korak provjere u kojem provjerava da li je digitalna vjerodajnica na listi opozvanih vjerodajnica. *Upravitelj digitalnim vjerodajnica* šalje poruku s imenom logičkog čvora B do *Poslužitelja digitalnih vjerodajnica* (4). *Poslužitelj digitalnih vjerodajnica* potom provjerava nalazi li se digitalna vjerodajnica logičkog čvora B na listi opozvanih digitalnih vjerodajnica te rezultat pretraživanja šalje natrag do *Upravitelja digitalnim vjerodajnicama* (5). Ukoliko primljena digitalna vjerodajnica nije ni na listi opozvanih digitalnih vjerodajnica, *Upravitelj digitalnim vjerodajnicama* zaključuje da je vjerodajnica vjerodostojna te je sprema u odgovarajući repozitorij (6).

Opisanim protokolom osigurana je obostrana razmjena digitalnih vjerodajnica jer logički čvor A u poruci (1) ujedno šalje vlastitu digitalnu vjerodajnicu logičkom čvoru B. *Upravitelj digitalnim vjerodajnicama* koji poslužuje logički čvor *B* obavlja isti postupak kao *Upravitelj digitalnim vjerodajnicama* koji poslužuje logički čvor *A* On u vlastiti repozitorij sprema vjerodostojnu digitalnu vjerodajnicu logičkog čvora

A. Nakon obostrane razmjene digitalnih vjerodajnica, logički čvorovi mogu uspostaviti sigurni komunikacijski kanal.

4.2.3 Protokol opoziva digitalnih vjerodajnica

Digitalne vjerodajnice mogu iz prestati biti važeće te ih stoga treba opozvati. Razlozi za opoziv digitalne vjerodajnice su da privatni ključ više nije tajan, podaci o logičkom čvoru u digitalnoj vjerodajnici su se promijenili, logički čvor je napustio infrastrukturu javnih ključeva ili logički čvor se želi isključiti iz infrastrukture javnih ključeva. *Poslužitelj digitalnih vjerodajnica* održava listu opozvanih digitalnih vjerodajnica u kojoj su navedene sve vjerodajnice koje je izdao, a više nisu važeće*. Upravitelj digitalnim vjerodajnicama* za vrijeme protokola razmjene digitalnih vjerodajnica te periodički za vrijeme trajanja komunikacije provjerava da li je digitalna vjerodajnica čvora sudionika još uvijek važeća.

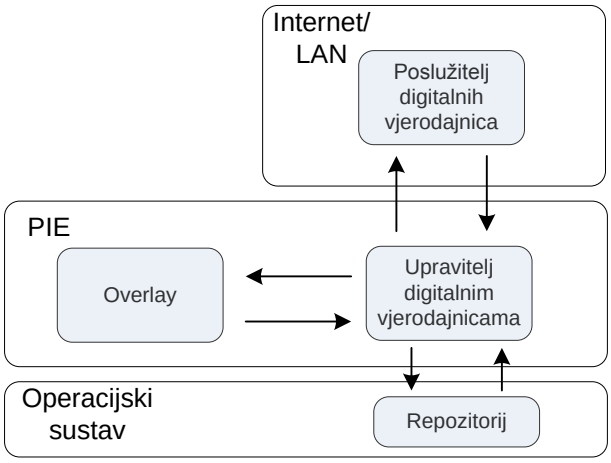


Slika 4.11 Protokol opoziva digitalne vjerodajnice

Na slici 4.11 prikazan je *Protokol opoziva digitalnih vjerodajnica*. *Upravitelj digitalnim vjerodajnicama* šalje ime logičkog čvora i razlog opoziva *Poslužitelju digitalnih vjerodajnica* kako bi opozvao digitalnu vjerodajnicu tog logičkog čvora (1). *Poslužitelj digitalnih vjerodajnica* uklanja digitalnu vjerodajnicu s liste objavljenih digitalnih vjerodajnica (2) te je stavlja na listu opozvanih digitalnih vjerodajnica (3), čime digitalna vjerodajnica postaje nevažeća. Uz digitalnu vjerodajnicu i informacije vezane uz nju, *Poslužitelj digitalnih vjerodajnica* stavlja na listu opozvanih digitalnih vjerodajnica i razlog opoziva digitalne vjerodajnice.

5 Programsko ostvarenje PKI sustava u okolini PIE

Infrastruktura javnih ključeva u okolini PIE ostvarena je pomoću dva programska elementa: *Upravitelja digitalnim vjerodajnicama* koji ostvaruje funkcionalnosti RA elementa i *Poslužitelja digitalnih vjerodajnica* koji ostvaruje funkcionalnosti CA elementa.



Slika 5.1 Međudjelovanje elemenata PKI sustava s okolinom PIE i operacijskim sustavom

Na slici 5.1 prikazano je međudjelovanje *Upravitelj digitalnim vjerodajnicama* i *Poslužitelja digitalnim vjerodajnicama* s okolinom PIE i operacijskim sustavom. *Upravitelj digitalnim vjerodajnicama* surađuje s *Overlay*-om. *Upravitelj digitalnim vjerodajnicama* sastavni je dio prividne mreže okoline PIE. *Overlay* koristi *Upravitelja digitalnim vjerodajnicama* kako bi logičke čvorove koje poslužuje osigurao mogućnost primjene sigurnosnih protokola MOSS i COSS tako da u repozitoriju fizičkog čvora budu sve potrebne digitalne vjerodajnice logičkih čvorova.

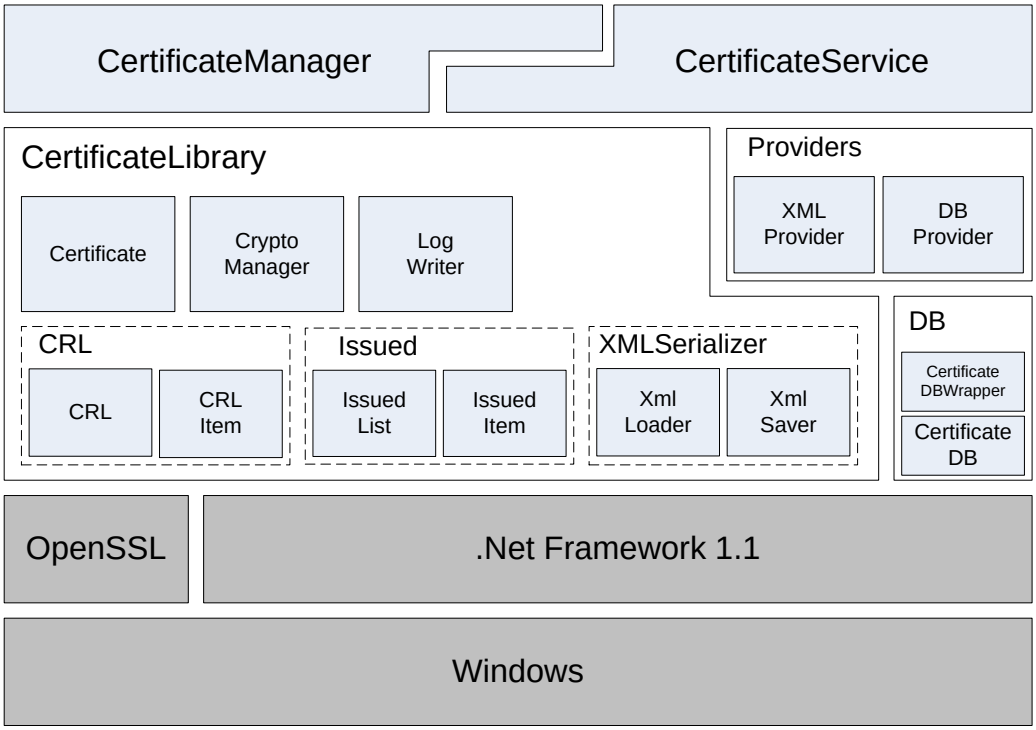
Upravitelj digitalnim vjerodajnicama sprema digitalne vjerodajnice u repozitorij digitalnih vjerodajnica koji pruža operacijski sustav. Ovisno o vrsti digitalne

vjerodajnice, ona se sprema u jedan od dva repozitorija: *Personal* repozitorij ili *Trusted root* repozitorij. Ukoliko je riječ o digitalnoj vjerodajnici nekog logičkog čvora tada se ona sprema u *Personal* repozitorij. U *Trusted root* repozitorij sprema se digitalna vjerodajnica *Poslužitelja digitalnih vjerodajnica*. Ovakav način spremanja digitalnih vjerodajnica omogućuje stvaranja lanaca povjerenja između vjerodajnica logičkih čvorova i vjerodajnice *Poslužitelja digitalnih vjerodajnica*. *Personal* i *Trusted root* repozitorij nalaze se u *Local Machine* kako bi bila dostupna neovisno o aktualnom korisničkom računu.

Za programsko ostvarenje infrastrukture javnih ključeva okoline PIE korištena je Microsoft .Net 1.1 tehnologija te programski jezik C# [41]. Korištena je razvojna okolina Microsoft Visual Studio 2003 [42]. Nadalje, za ostvarenje liste objavljenih i opozvanih digitalnih vjerodajnice korištena je SQL (engl. *Structured Query Language*) baza podataka upravljana sustavom Microsoft SQL Server 2000. Moguće je koristiti i jednostavniji način spremanja lista objavljenih i opozvanih digitalnih vjerodajnica u XML datoteke. Sve razvijene komponente ispitane su ispitnim primjercima (engl. *unit test*) pri čemu je korišten alat NUnit 2.2.9 [43] za Microsoft .Net Framework 1.1.

5.1 Programski model PKI sustava u okolini PIE

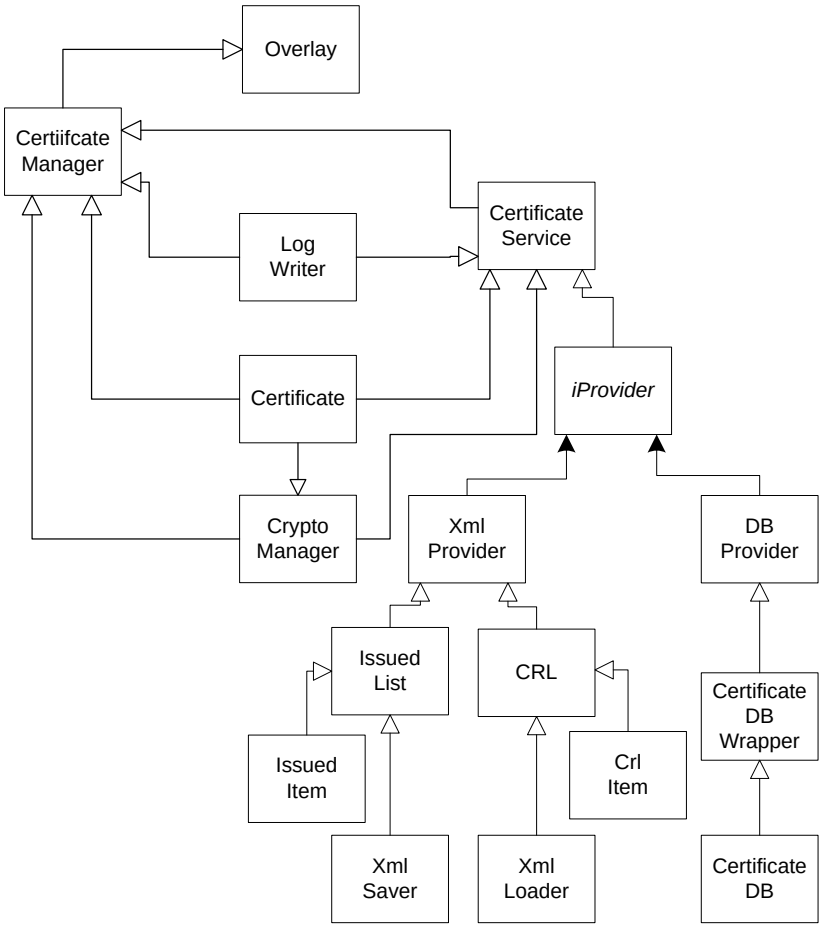
Za potrebe ostvarenja infrastrukture javnih ključeva u okolini PIE u sklopu ovog diplomskog rada razvijeno je opširno programsko rješenje koje se sastoji od nekoliko razreda i slijedi objektno-orijentiranu paradigmu. Razvijeni razredi ostvaruju određene skupove funkcionalnosti te, djelujući zajedno, čine kompletnu funkcionalnost infrastrukture javnih ključeva. Takav programski model omogućuje da *Upravitelj digitalnim vjerodajnicama* i *Poslužitelj digitalnih vjerodajnica* koriste isti skup funkcija i razmjenjuju unaprijed definirane formate poruka.



Slika 5.2 Programski model infrastrukture javnih ključeva okoline PIE

Na slici 5.2 prikazan je razvijen programski model infrastrukture javnih ključeva okoline PIE. Programski model sastoji se od pet programskih modula: *CertificateManager*, *CertificateService*, *CertificateLibrary*, *Providers* i *Database*. Moduli *OpenSSL*, *.Net Framework 1.1* i *Windows* nisu dio razvijenog programskog rješenja, već su uzeti kao gotova rješenja koja su potrebna za rad programskih modula koji nalaze iznad njih. Programski moduli koji se nalaze

iznad drugih programskih modula za svoj rad koriste funkcionalnosti koje pružaju moduli ispod njih. Programski moduli *CertificateManager* i *CertifcateService* su ujedno i razredi. Programski modul *CertificateLibrary* je biblioteka koja sadrži razrede koje koriste programski moduli *CertificateManager* i *CertificateService*. Programski modul *Providers* je također biblioteka koja sadrži razrede koje koristi programski modul *CertificateService*. *Database* programski modul sadrži dva razreda koja služe za pristup bazi podataka. Na slici 5.3 prikazan je pojednostavljen dijagram razreda programskog rješenja infrastrukture javnih ključeva okoline PIE.



Slika 5.3 Dijagram razreda infrastrukture javnih ključeva okoline PIE

5.1.1 Modul *CertificateService*

Poslužitelj digitalnih vjerodajnica ostvaren je u razredu *CertificateService* koji je izložen kako web usluga (engl. *Web Service*). Na taj način omogućen je udaljen pristup *CertificateManager*-a do metoda *CertificateService*-a te razmjena poruka čime su ostvarena tri protokola infrastrukture javnih ključeva. Iz programskog modela na slici 5.2 se vidi da razred *CertificateService* za svoj rad koristi funkcionalnosti programskog modula *CertificateLibrary* za stvaranje digitalne vjerodajnice *Poslužitelja digitalnih vjerodajnica* i digitalno potpisivanje zahtijeva za digitalnom vjerodajnicom te funkcionalnosti programskog modula *Provider* za upravljanje listama objavljenih i opozvanih digitalnih vjerodajnica. U tablici 5.3 navedene su metode razreda *CertificateService*.

Metoda	Povratni tip	Ulazni parametri	Opis
CertificateClass	void	void	Konstruktor u kojem se učitavaju konfiguracijske postavke Poslužitelja digitalnih vjerodajnica
GenerateCACertificate	bool	void	Stvara samo-potpisanu digitalnu vjerodajnicu Poslužitelja digitalnih vjerodajnica
GetCACertificate	Certificate	void	Dohvaća digitalnu vjerodajnicu Poslužitelja digitalnih vjerodajnica
SignCSR	Certificate	csrReq:string	Digitalno potpisuje zahtjev za digitalnom vjerodajnicom logičkog čvora privatnim ključem Poslužitelja digitalnih vjerodajnica te objavljuje potpisanu digitalnu vjerodajnicu
IsRevoked	bool	cert:Certificate	Provjerava da li se primljena digitalna vjerodajnica nalazi na listi opozvanih digitalnih vjerodajnica
RevokeCertificate	bool	nodeName:string	Stavlja digitalnu vjerodajnicu logičkog čvora na listu opozvanih digitalnih vjerodajnica

Tablica 5.1 Metode razreda *CertificateService*

U konstruktoru razreda *CertificateService* dohvaćaju se konfiguracijske postavke koje su bitne za rad *Poslužitelja digitalnih vjerodajnica*. Konfiguracijske postavke spremljene su u XML datoteku.

```
<ConfigCertificateService>
  <KeyLength>1024</KeyLength>
  <Duration>365</Duration>
  <CaName>PIE_CA</CaName>
  <Provider>DB</Provider>
  <CrlPath>Certifikati\CRL.xml</CrlPath>
  <PublishedPath>Certifikati\Published.xml</PublishedPath>
  <ConnectionString>data source=…;…;initial catalog=…</ConnectionString>
</ConfigCertificateService>
```

Slika 5.4 Primjer datoteke s konfiguracijskih postavki za razred CertificateService

Na slici 5.4 prikazan je primjer datoteke s konfiguracijskim postavkama za razred *CertificateService*. XML element *ConfigCertificateService* sadrži elemente *KeyLength* ,*Duration*, *CaName*, *Provider CrlPath*, *PublishedPath* te *ConnectionString*. Element *KeyLength* sadrži duljinu privatnog ključa *Poslužitelja digitalnih vjerodajnica* izraženu u bitovima, element *Duration* sadrži vremenski interval valjanosti digitalne vjerodajnice *Poslužitelja digitalnih vjerodajnica* izražen u danima. Element *CaName* sadrži ime *Poslužitelja digitalnih vjerodajnica* koje je navedeno u njegovoj digitalnoj vjerodajnici. Element *Provider* sadrži informaciju koji način zapisa liste objavljenih i opozvanih digitalnih vjerodajnica koristi *Poslužitelj digitalnih vjerodajnica*. Oznaka *DB* označava da je riječ o pružatelju koji koristi *SQL* bazu podataka, dok oznaka *XML* označava da je riječ o pružatelju koji koristi XML datoteke. Element *CrlPath* se koristi samo ako se koristi i XML pružatelj te sadrži putanju do XML datoteke sa listom opozvanih digitalnih vjerodajnica. Element *PublishedPath* se također koristi kada uz XML pružatelja te sadrži putanju do XML datoteke listom objavljenih digitalnih vjerodajnica. Element *ConnectionString* koristi se u kombinaciji s *DB* pružateljem i sadrži informaciju kako se spojiti na SQL bazu podataka u kojoj su tablice koje sadrže liste objavljenih i opozvanih digitalnih vjerodajnica.

5.1.2 Modul *CertificateManager*

Upravitelj digitalnim vjerodajnicama ostvaren je u razredu *CertificateManager*. Koristi funkcionalnosti koje mu pružaju programski moduli *CertificateLibrary* i *CertificateService*. U tablici 5.4 navedene su metode razreda CertificateManager.

Metoda	Povratni tip	Ulazni parametri	Opis
CertificateManager	void	void	Konstruktor u kojem se učitavaju konfiguracijske postavke Upravitelja digitalnim vjerodajnicama
SetPKI	bool	nodeName:string	Postavlja infrastruktura javnih ključeva za logički čvor čije se ime predaje kao ulazni parametar
ImportCACertificate	void	caCert:Certificate	Pohranjuje digitalnu vjerodajnicu Poslužitelja digitalnih vjerodajnica u Trusted root repozitorij
GenerateCSR	Certificate	nodeName:string	Generira zahtjev za digitalnom vjerodajnicom te par ključeva, privatni i javni
GetCACert	Certificate	void	Dohvaća digitalnu vjerodajnicu Poslužitelja digitalnih vjerodajnica
SendCSRToCA	Certificate	csr:Certificate	Šalje zahtjev za digitalnom vjerodajnicom Poslužitelju digitalnih vjerodajnica te prima digitalno potpisani zahtjev.
ImportCertToStore	void	cert:Certificate	Pohranjuje digitalnu vjerodajnicu logičkog čvora u Personal repozitorij
SetCommunicationReq	bool	nodeName:string, nodeURI:string	Dohvaća i postavlja u repozitorij digitalnu vjerodajnicu logičkog čvora s kojim želi uspostaviti sigurni komunikacijski kanal
ValidateCertificate	bool	cert:Certificate	Verificira digitalnu vjerodajnicu
IsCertificateRevoked	bool	cert:Certificate	Provjerava da li je digitalna vjerodajnica na listi opozvanih digitalnih vjerodajnica
GetNodeCertificate	Certificate	nodeName:string	Dohvaća digitalnu vjerodajnicu logičkog čvora čije je ime dano kao parametar

Tablica 5.2 Metode razreda *CertificateManager*

Modul *CertificateLibrary* mu pruža funkcionalnosti za stvaranje zahtijeva za digitalnom vjerodajnicom pohranjivanje i dohvaćanje digitalnih vjerodajnica iz repozitorija te verifikaciju istih. Koristi i funkcionalnosti programskog modula *CertificateService* za dohvaćanje digitalne vjerodajnice *Poslužitelja digitalnih*

vjerodajnica, digitalno potpisivanje zahtijeva za digitalnom vjerodajnicom te provjera nalazi li se digitalna vjerodajnica na listi opozvanih digitalnih vjerodajnica.

U konstruktoru razreda *CertificateManager* dohvaćaju se konfiguracijske postavke koje su bitne za rad *Upravitelja digitalnim vjerodajnicama*. Konfiguracijske postavke spremljene su u XML datoteku. Na slici 5.5 prikazan je primjer datoteke s konfiguracijskim postavkama za razred *CertificateManager*.

```
<ConfigCertificateManager>
      <KeyLength>1024</KeyLength>
      <Duration>365</Duration>
      <CaUrl>http://localhost/CertificateService/CertificateService.asmx</CaUrl>
</ConfigCertificateManager>
```

Slika 5.5 Primjer datoteke s konfiguracijskim postavkama za razred CertificateManager

XML element *ConfigCertificateManager* prikazan na priloženom izlistku sadrži elemente *KeyLength*, *Duration* i *CaUrl*. U elementu *KeyLength* navedena je duljina privatnog ključa u bitovima. Nadalje, u elementu *Duration* naveden je vremenski interval valjanosti digitalne vjerodajnice koju stvara taj *Upravitelj digitalnim vjerodajnicama*. Vremenski interval valjanosti navodi se u danima. Element *CaUrl* sadrži adresu *Poslužitelja digitalnih vjerodajnica*. Koristeći tu adresu, *CertificateManager* zna pristupiti metodama *CertificateService razreda*.

5.1.3 Modul *CertificateLibrary*

Programski modul *CertificateLibrary* pruža skup funkcionalnosti za stvaranje i upravljanje digitalnim vjerodajnicama. *CertificateLibrary* se sastoji od devet razreda: *Certificate*, *CryptoManager*, *XmlLoader*, *XmlSaver*, *CRL*, *CrlItem*, *IssuedList* , *IssuedItem* i *LogWriter*. Neki od navedenih razreda su na slici 5.2 programskog modela grupirani i to ovisno o funkcionalnosti koju ostvaruju. Tako postoje tri grupe: *CRL*, *Issued* i *XmlSerializer*. Svaka grupa predstavlja jedan prostor imena (engl. *namespace*). Prostor imena *CRL* sadrži razrede CRL i CrlItem, prostor imena *Issued* sadrži razrede *IssuedList* i *IssuedItem* dok prostor imena *XmlSerializer* sadrži razrede *XmlLoader* i *XmlSaver*.

Razred *Certificate*

Iako je korišteni format digitalnih vjerodajnica X.509 v3, za potrebe ovog rada uvedeno je jedno proširenje tog formata (engl. *wrapper*). To proširenje ostvareno je razredom *Certificate*. Neovisno o tom proširenju sigurnosni protokoli *MOSS* i *COSS* i dalje koriste X.509 v3 digitalne vjerodajnice, a iste se pohranjuju i u repozitorij digitalnih vjerodajnica. U tablici 5.3 prikazana su svojstva razreda *Certificate*.

Svojstvo	Opis
IssuedTo	Ime čvora vlasnika digitalne vjerodajnice
CSR	Zahtjev za digitalnom vjerodajnicom u PEM formatu
PrivateKey	Privatni ključ čvora zapisan u PEM formatu
SignedCertificate	Digitalno potpisana digitalna vjerodajnica u PEM formatu

Tablica 5.3 Svojstva razreda *Certificate*

Razlozi uvođenja proširenog formata su jednostavnija proširivost dodatnim informacijama i jednostavniji prijenos digitalnih vjerodajnica putem računalne mreže. Digitalne vjerodajnice se mogu proširiti informacijama kojim se definira namjena digitalne vjerodajnice, označava se s kojim sigurnosnim protokolom koristi digitalna vjerodajnica te informacijama koje dodatno opisuju logički čvor

vlasnik digitalne vjerodajnice. Jednostavniji prijenos digitalnih vjerodajnica kroz računalnu mrežu omogućen je zbog seriabilnosti proširenog formata što čini prijenos digitalnih vjerodajnica putem SOAP poruka pogodnim.

Razred *CryptoManager*

Razred *CryptoManager* pruža niz funkcionalnosti za upravljanje digitalnim vjerodajnicama. Omogućuje stvaranje zahtjeva za digitalnom vjerodajnicom, digitalno potpisivanje zahtjeva za digitalnom vjerodajnicom, stvaranje samo-potpisane digitalne vjerodajnice, provjere vjerodostojnosti digitalnih vjerodajnica, spremanje, dohvaćanje i brisanje digitalnih vjerodajnica iz repozitorija. Nadalje kako se u repozitorij pohranjuju digitalne vjerodajnice X.509 v3 formata, a u razvijenoj infrastrukturi javnih ključeva okoline PIE koriste se vjerodajnice *Certificate* formata, razred *CryptoManager* pruža mogućnost konverzije iz formata *Certificate* u format X.509 v3 i obrnuto. U tablici 5.4 navedene su metode razreda *CryptoManager*.

Metoda	Povratni tip	Ulazni parametri	Opis
GenerateCSR	Certificate	nodeName:string, keyLength:int, duration:int	Stvara zahtjev za digitalnom vjerodajnicom
GenerateCertificate	Certificate	nodeName:string, keyLength:int, duration:int	Stvara samo-potpisanu digitalnu vjerodajnicu
SignCSR	Certificate	caCert:Certificate csr:string	Potpisuje zahtjev za digitalnim vjerodajnicom
ImportCertificate	void	cert:Certificate, store:string	Sprema digitalnu vjerodajnicu u repozitorij
GetCertificateFromStore	Certificate	nodeName:string, store:string	Dohvaća digitalnu vjerodajnicu iz repozitorija
GetCertificateFromX509Cert	Certificate	xCert:X509Certificate	Pretvara digitalnu vjerodajnicu iz formata X.509 u Certificate format
GetX509CertFromCertificate	X509 Certificate	cert:Certificate	Pretvara digitalnu vjerodajnicu iz formata Certificate u X.509 format
DeleteCertificateFromStore	void	cert:Certificate, store:string	Uklanja digitalnu vjerodajnicu iz repozitorija
VerifyCertificate	bool	cert:Certificate	Provjerava vjerodostojnost digitalne vjerodajnice

Tablica 5.4 Metode razreda *CryptoManager*

Razred *CRL*

Razred *CRL* je namijenjen za upravljanje listom opozvanih digitalnih vjerodajnica koja je zapisana u XML datoteci. Omogućuje dodavanje digitalnih vjerodajnica u listu opozvanih digitalnih vjerodajnica, brisanje iz liste opozvanih digitalnih vjerodajnica te provjeru nalazi li se digitalna vjerodajnica u listi opozvanih digitalnih vjerodajnica. Razred *CRL* zapis liste opozvanih digitalnih vjerodajnica u XML datoteci prilikom rada sprema u svoju internu strukturu pretvarajući pritom niz zapisa liste opozvanih digitalnih vjerodajnica u objekte razreda *CrlItem*. Sve promjene nad listom opozvanih vjerodajnica vrši nad svojom internom strukturom koju potom pohranjuje natrag u XML datoteku. Za procese učitavanja i spremanja liste opozvanih digitalnih vjerodajnica u XML datoteku, razred *CRL* koristi razrede *XmlLoader* i *XmlSaver*. Ti razredi objašnjeni su u nastavku ovog poglavlja. U tablici 5.5 navedene su metode razreda *CRL*.

Metoda	Povratni tip	Ulazni parametri	Opis
Add	void	cert:CrlItem	Dodaje objekt razreda *CrlItem* u listu opozvanih digitalnih vjerodajnica
Add	void	cert:Certificate, comment:string	Dodaje digitalnu vjerodajnicu u listu opozvanih digitalnih vjerodajnica.
Delete	void	issuedTo:string	Uklanja digitalnu vjerodajnicu sa liste opozvanih digitalnih vjerodajnica
Contains	bool	issuedTo:string	Provjerava sadrži li lista opozvanih digitalnih vjerodajnica digitalnu vjerodajnicu određenog logičkog čvora
GetCRLFromFile	CRL	fileName:string	Puni listu opozvanih digitalnih vjerodajnica iz XML datoteke
SaveToFile	void	fileName:string	Sprema listu opozvanih digitalnih vjerodajnica u XML datoteku

Tablica 5.5 Metode razreda *CRL*

```
...
<CrlItem>
   ...
</CrlItem>
<CrlItem>
    <IssuedTo>ZaOpoziv_8.5.07_5.54.9_1024</IssuedTo>
    <ExpirationDate>7.5.2008 16:54:09</ExpirationDate>
    <CreationDate>8.5.2007 16:54:09</CreationDate>
    <CertBody>-----BEGIN CERTIFICATE-----
MIIBrjCCARcCCQDPGaKIy+qwdjANBgkqhkiG9w0BAQUFADARMQ8wDQYDVQQDEwZB
...
rYbzhgopRRTKb7ne6pEHtdXdI/FvlESWLxVxih9mx/pfZYhST1XdiM8skxf+8jPq
wKo=
-----END CERTIFICATE-----
    </CertBody>
    <Comment>Opozvano u testne svrhe</Comment>
</CrlItem>
<CrlItem>
   ...
</CrlItem>
...
```

Slika 5.6 Primjer zapisa elemenata liste opozvanih digitalnih vjerodajnica u XML formatu

Na slici 5.6 prikazan je primjer zapisa elemenata u listi opozvanih digitalnih vjerodajnica u XML formatu. Svaki zapis označen je elementom *CrlItem*. Element *CrlItem* sadrži elemente *IssuedTo*, *ExpirationDate*, *CreationDate*, *CertBody* i *Comment*. U elementu *IssuedTo* nalazi se jedinstveno ime logičkog čvora kojem pripada digitalna vjerodajnica, u elementu *ExpirationDate* je naveden datum isteka valjanosti digitalne vjerodajnice, u elementu *CreationDate* datum stvaranja digitalne vjerodajnice, a u elementu *CertBody* naveden je Base64 kodiran sadržaj digitalne vjerodajnice, dok element *Comment* sadrži obrazloženja zašto je digitalna vjerodajnica opozvana.

Razred *IssuedList*

Razred *IssuedList* je namijenjen za upravljanje listom objavljenih digitalnih vjerodajnica koja je zapisana u XML datoteci. Princip rada mu je jednak kao i razredu *CRL*. Omogućuje dodavanje digitalnih vjerodajnica u listu objavljenih digitalnih vjerodajnica, brisanje sa liste objavljenih digitalnih vjerodajnica te provjeru nalazi li se digitalna vjerodajnica na listi objavljenih digitalnih vjerodajnica te njeno dohvaćanje. Razred *IssuedList* zapis liste objavljenih digitalnih vjerodajnica u XML datoteci prilikom rada sprema u svoju internu

strukturu pretvarajući pritom niz zapisa liste objavljenih digitalnih vjerodajnica u objekte razreda *IssuedItem*. Sve promjene nad listom opozvanih vjerodajnica vrši nad svojom internom strukturom koju potom pohranjuje natrag u XML datoteku. Za procese učitavanja i pohranjivanja liste objavljenih digitalnih vjerodajnica u XML datoteku, razred *IssuedList*, jednako kako i razred *CRL*, koristi razrede *XmlLoader* i *XmlSaver*. Ti razredi objašnjeni su u nastavku ovog poglavlja. U tablici 5.6 navedene su metode razreda *IssuedList.*

Metoda	Povratni tip	Ulazni parametri	Opis
Add	void	cert:IssuedItem	Dodaje objekt razreda *IssuedItem* u listu objavljenih digitalnih vjerodajnica
Add	void	cert:Certificate	Dodaje digitalnu vjerodajnicu u listu objavljenih digitalnih vjerodajnica
Delete	void	issuedTo:string	Uklanja digitalnu vjerodajnicu sa liste objavljenih digitalnih vjerodajnica
Get	IssuedItem	issuedTo:string	Dohvaća objekt razreda *IssuedItem* sa liste objavljenih digitalnih vjerodajnica
Contains	bool	issuedTo:string	Provjerava sadrži li lista objavljenih digitalnih vjerodajnica digitalnu vjerodajnicu određenog logičkog čvora
GetPublishedFromFile	CRL	fileName:string	Puni listu objavljenih digitalnih vjerodajnica iz XML datoteke
SaveToFile	void	fileName:string	Sprema listu objavljenih digitalnih vjerodajnicu u XML datoteku

Tablica 5.6 Metode razreda *IssuedList*

```
...
<IssuedItem>
    ...
</IssuedItem>
<IssuedItem>
    <IssuedTo>NodeA</IssuedTo>
    <ExpirationDate>7.5.2008 17:14:48</ExpirationDate>
    <CreationDate>8.5.2007 17:14:48</CreationDate>
    <CertBody>-----BEGIN CERTIFICATE-----
MIIBszCCARwCCQCrG1sEcHOnjzANBgkqhkiG9w0BAQUFADARMQ8wDQYDVQQDEwZB
...
OiUnEkQ+Y/WOjG0fY9f/vwyNK6FEs7f7M0hoFWjuO+SSdTXZrpd+gPxxIAkXmnRy
Byc4zsyMeA==
-----END CERTIFICATE-----
    </CertBody>
</IssuedItem>
<IssuedItem>
    ...
</IssuedItem>
...
```

Slika 5.7 Primjer zapisa elemenata liste
objavljenih digitalnih vjerodajnica u XML formatu

Na slici 5.7 prikazan je primjer zapisa elemenata u listi objavljenih digitalnih vjerodajnica u XML formatu. Svaki zapis označen je elementom *IssuedItem*. Element *IssuedItem* sadrži elemente *IssuedTo*, *ExpirationDate*, *CreationDate* i *CertBody*. Sadržaji elemenata *IssuedTo*, *ExpirationDate*, *CreationDate* i *CertBody* jednaki su kao i kod elemenata liste opozvanih digitalnih vjerodajnica

Razred *XmlSaver*

Razred *XMLSaver* služi za pretvaranje objekta u odgovarajuću mu XML prezentaciju na način da svojstva objekta prikaže kao istoimene XML elemente, a vrijednosti svojstava objekta kao vrijednosti odgovarajućih XML elemenata. U tablici 5.7 navedene su metode razreda *XMLSaver*.

Metoda	Povratni tip	Ulazni parametri	Opis
SerializeObject	string	item:object	Serializira primljeni objekt u odgovarajuću XML reprezentaciju
SaveObjectToXml	void	items:object[], filePath:string	Serializira polje objekata u odgovarajuću XML reprezentaciju i pohranjuje ju u datoteku

Tablica 5.7 Metode razreda *XMLSaver*

Za svoj rad razred *XMLSaver* koristi tehniku refleksije što mu omogućuje da postavlja upite objektu o njegovim meta-podacima i samim time pretvorbu objekta bilo kojeg razreda u odgovarajuću XML prezentaciju.

Razred *XmlLoader*

Razred *XmlLoader* je komplementaran razred razredu *XmlSaver*. Razred *XmlLoader* odgovarajuću XML prezentaciju objekta pretvara natrag u objekt. U tablici 5.8 navedene su metode razreda *XmlLoader*.

Metoda	Povratni tip	Ulazni parametri	Opis
DeserializeObject	object	xmlPath:string, tip:Type	Deserializira XML element u objekt odgovarajućeg razreda
LoadObjectsFromXml	object[]	xml:string, tip:Type	Deserializira niz XML elemetana u polje objekata odgovarajućih razreda

Tablica 5.8 Metode razreda *XmlLoader*

Za svoj rad razred *XmlLoader* jednako kao i razred *XmlSaver* koristi tehniku refleksije što mu omogućuje da postavlja upite objektu o njegovim meta-podacima i samim time pretvorbu bilo koje XML prezentacije objekta u odgovarajući objekt.

Razred *LogWriter*

Za potrebe ovoga rada razvijen je razred *LogWriter* kojim je ostvareno bilježenje. Razredi *CertificateManager* i *CertificateService* koriste *LogWriter* kako bi u pripadajuće im dnevnike bilježaka bilježili koje akcije se trenutno izvršavaju. Na taj način ostvareno je da osoba koja nadgleda rad okoline PIE ima uvid u procese koji se izvršavaju i u njihove ishode. U tablici 5.9 naveden je popis metoda razreda *LogWriter*.

Metoda	Povratni tip	Ulazni parametri	Opis
LogWriter	void	source:string, logName:string	Konstruktor kojemu se predaju putanja i ime dnevnika
Write	void	message:string, exc:Excpetion	Zapisuje poruku u dnevnik

Tablica 5.9 Metode razreda *LogWriter*

Ukoliko je prilikom nekog od procesa došlo do pogreške, ta pogreška biti će zabilježena u dnevniku bilježaka te će se time olakšati i samo otklanjanje uzroka pogreške. U nastavku je dan regularni izraz zapisa u dnevniku bilježaka.

{vrijeme bilješke}-{START|END|ERROR}:{Naziv metode} [: Poruka o pogrešci]

Zapis u dnevniku bilježaka se sastoji o vremena nastanka zapisa, informaciji da li je riječ o početku ili završetku metode koja se izvodi ili je došlo do pogreške prilikom izvođenja metode. Potom slijedi naziv metode iza kojeg slijedi informacija o pogrešci ukoliko je došlo do pogreške.

5.1.4 Modul *Providers*

U ovom diplomskom radu ostvarena su dva načina zapisa lista objavljenih i opozvanih digitalnih vjerodajnica. Prvi zapis je u *XML* datoteku, a drugi u *SQL* bazu podataka. Za upravljanje listama objavljenih i opozvanih digitalnih vjerodajnica razred *CertificateService* koristi funkcionalnosti koje mu pruža razred koji se nalazi ispod njega, razred *iProvider*. Razred *iProvider* je sučelje (engl. *interface*) koje definira skup metoda koje moraju ostvariti razredi koji ga nasljeđuju. U tablici 5.10 navedene su metode sučelja *iProvider*.

Metoda	Povratni tip	Ulazni parametri	Opis
PublishCertificate	bool	cert:Certificate	Sprema digitalnu vjerodajnicu u listu objavljenih digitalnih vjerodajnica
GetCertificate	Certificate	issuedTo:string	Dohvaća digitalnu vjerodajnicu s liste objavljenih digitalnih vjerodajnica
RevokeCertifcate	bool	issuedTo:string, commnet:string	Opoziva digitalnu vjerodajnicu
IsRevoked	bool	issuedTo:string	Provjerava nalazi li se digitalna vjerodajnica u listi opozvanih digitalnih vjerodajnica
RePublishCertificate	bool	cert:Certificate	Ponovno objavljuje digitalnu vjerodajnicu koja je bila opozvana
DeleteCertificate	bool	issuedTo:string	Uklanja digitalnu vjerodajnicu s lista objavljenih i opozvanih digitalnih vjerodajnica

Tablica 5.10 Metode sučelja *iProvider*

Postoje dva razreda koja ostvaruju sučelje *iProvider*. To su razred *XMLProvider* za upravljanje listama objavljenih i opozvanih digitalnih vjerodajnica zapisanih u XML datoteke i razred *DBProvider* za upravljanje listama objavljenih i opozvanih digitalnih vjerodajnica zapisanih u SQL bazu podataka. U konfiguracijskim postavkama *Poslužitelja digitalnih vjerodajnica* određuje se koji će način zapisa lista objavljenih i opozvanih digitalnih vjerodajnica *Poslužitelj* koristiti, te samim time i kojeg će pružatelja koristiti. Razred *XMLProvider* koristi već spomenute razrede *CRL* i *IssuedList* za upravljanje listama objavljenih i opozvanih digitalnih vjerodajnica zapisanih u XML datotekama. Razred *DBProvider* koristi za pristup bazi podataka razred *CertificateDBWrapper* modula *Database*.

Zapis listi objavljenih i opozvanih digitalnih vjerodajnica u XML datotekama preporučljivo je koristiti ako *Poslužitelj digitalnih vjerodajnica* poslužuje relativno malen broj logičkih čvorova, kada obrađuje relativno malen broj zahtijeva koji dolaze slijednim redoslijedom i prilikom nedostataka resursa na računalu koje udomaćuje *Poslužitelja digitalnih vjerodajnica*. SQL baza podataka koristi se kada *Poslužitelj digitalnih vjerodajnica* poslužuje velik broj logičkih čvorova te kada treba istovremeno obrađivati više paralelnih zahtijeva.

5.1.5 Modul *Database*

Modul *Database* sadrži dva razreda: *CertificateDB* i *CertificateDBWrapper*. Ti razredi služe za pristup listama objavljenih i opozvanih digitalnih vjerodajnica koje se nalaze zapisane u bazi podataka. Podržana baza podataka je SQL baza podataka na Microsoft Server 2000 poslužitelju. Lista objavljenih digitalnih vjerodajnica pohranjuje se u tablicu *ISSUED_CERTIFICATES*, dok se lista opozvanih digitalnih vjerodajnica pohranjuje u tablicu *REVOKED_CERTIFICATES*. U nastavku poglavlja objašnjeni su razredi *CertificateDB* i *CertificateDBWrapper*.

Razred *CetificateDB*

Razred *CertificateDB* sadrži metode za upravljanje listama objavljenih i opozvanih digitalnih vjerodajnica zapisanih u bazi podataka. U tablici 5.x navedene su metode razreda *CertificateDB*.

Metoda	Povratni tip	Ulazni parametri	Opis
ListCertificates	Certificate[]	cmd:SqlCommand	Izlistava sve digitalne vjerodajnice koje se nalaze u tablici ISSUED_CERTIFICATES
PublishCertificate	int	cmd:SqlCommand, cert:Certificate	Sprema digitalnu vjerodajnicu u tablicu ISSUED_CERTIFICATES
GetCertificate	Certificate	cmd:SqlCommand, issuedTo:string	Dohvaća digitalnu vjerodajnicu iz tablice ISSUED_CERTIFICATES ili iz tablice REVOKED_CERTIFICATES
DeleteCertificate	int	cmd:SqlCommand, issuedTo:string	Uklanja digitalnu vjerodajnicu iz tablice ISSUED_CERTIFICATES ili iz tablice REVOKED_CERTIFICATES
RevokeCertificate	int	cmd:SqlCommand, cert:Certificate, comment:string	Sprema digitalnu vjerodajnicu u tablicu REVOKED_CERTIFICATES
IsRevoked	bool	cmd:SqlCommand, issuedTo:string	Provjerava nalazi li se digitalna vjerodajnica u tablici REVOKED_CERTIFICATES

Tablica 5.11 Metode razreda *CertificateDB*

Svaka metoda prima objekt razreda SqlCommand. To je izvedeno tako da se mogu stvarati složeniji procesi koje se sastoje od poziva nekoliko metoda unutar jedne transakcije. Primjer takvog procesa je opoziv digitalne vjerodajnice gdje se digitalna vjerodajnica briše s liste objavljenih digitalnih vjerodajnica i sprema u listu opozvanih digitalnih vjerodajnica. Takav proces će se izvršiti u cijelosti ili se uopće neće izvršiti.

Razred *CertificateDBWrapper*

Razred *CertificateDBWrapper* predstavlja „omotač“ oko razreda *CertificateDB*. Ima jednake metode kao i razred *CertificateDB* stoga ovdje te metode neće biti posebno navedene. U svakoj metodi razreda *CertificateDBWrapper* uspostavlja se veza s bazom podataka, stvara se objekt razreda SqlCommand, započinje se transakcija i poziva se odgovarajuća metoda razreda *CertificateDB* kojoj predaje

objekt razreda SqlCommand i odgovarajući ulazne parametre. Ako se pozvana metoda uspješno izvrši tada se transakcija zaključuje, no ako prilikom izvršavanja pozvane metode dođe do pogreške tada se transakcija poništava čime se poništavaju i sve promjene napravljene u bazi podataka. Na kraju svake metode razreda *CertificateDBWrapper* prekida se veza s bazom podataka.

5.1.6 Modul *OpenSSL*

Razred *CryptoManager* za operacije generiranja i digitalnog potpisivanja digitalnih vjerodajnica koristi komercijalni alat OpenSSL [31]. Alat OpenSSL razvijen je kao alat otvorene arhitekture (engl. *Open Source*). OpenSSL ostvaruje SSL v2/v3 (engl. *Secure Sockets Layer*) i TLS v1 (engl. *Transport Layer Security*) protokole te cijelu kriptografsku biblioteku opće namjene. Na OpenSSL projektu rade dobrovoljci diljem svijeta koji koriste Internet za komunikaciju, planiranje i razvoj OpenSSL alata i odgovarajuće dokumentacije. OpenSSL se temelji na biblioteci koju su razvili Eric A. Young i Tim J. Hudson te se temelji na otvorenoj licenci što podrazumijeva da je besplatan za ne-komercijalno korištenje. Open*S*SL alat još nije u potpunosti razvijen, a inačica koja je korištena za potrebe ovog rada jest OpenSSL 0.9.8e.

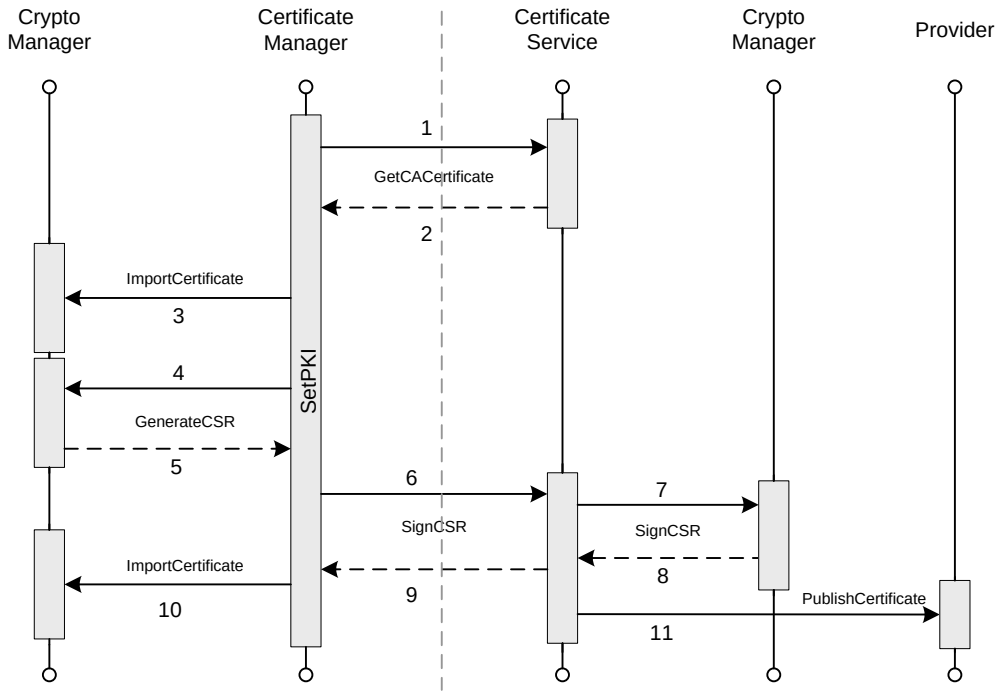
Alat OpenSSL je komandno linijski alat koji se koristi za različite kriptografske funkcije OpenSSL kriptografske biblioteke koji se poziva iz ljuske operacijskog sustava. Može se koristiti za generiranje i upravljanje privatnim, javnim ključevima i njihovim parametrima. Nadalje, koristi se za operacije asimetrične kriptografije ili kriptografije javnim ključem, generiranje X.509 digitalnih vjerodajnica, zahtijeva za digitalnim vjerodajnica i listi opoziva digitalnih vjerodajnica. Omogućuje izračunavanje digitalnog sažetka poruka te kriptiranje i dekriptiranje podataka.

5.2 Ostvarenje protokola PKI sustava u okolini PIE

Protokoli infrastrukture javnih ključeva u okolini PIE programski su ostvareni kao pozivi metoda između pojedinih razreda programskog modela infrastrukture javnih ključeva okoline PIE. U nastavku ovog poglavlja objašnjena su ostvarenja triju protokola infrastrukture javnih ključeva u okolini PIE.

5.2.1 Protokol dodjele digitalnih vjerodajnica

Prilikom stvaranja novog logičkog čvora, *Overlay* poziva metodu *SetPKI* razreda *CertificateManager*, predavajući mu pri tomu jedinstveno ime logičkog čvora. Metodom *SetPKI* ostvaren je protokol dodjele digitalne vjerodajnice za novi logički čvor.



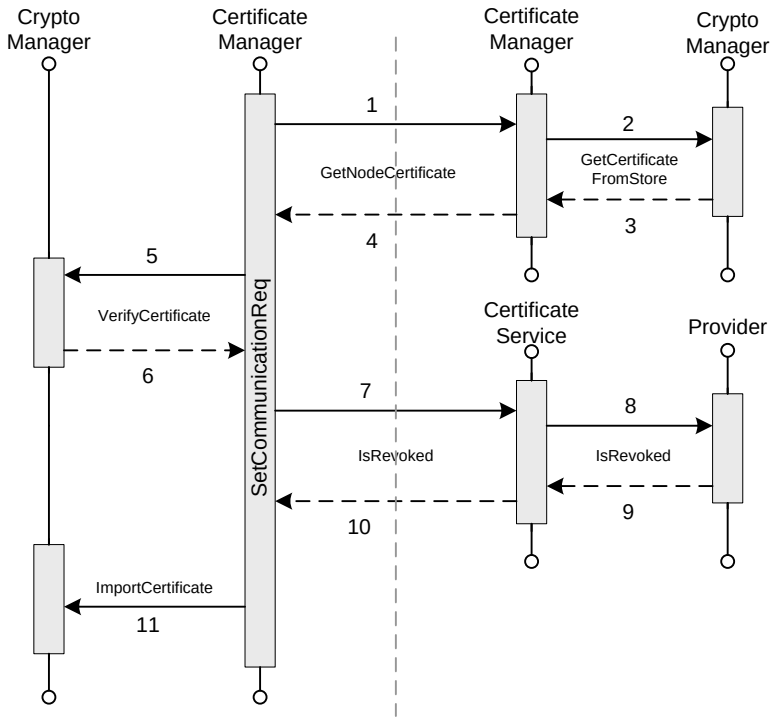
Slika 5.8 Protokol dodjele digitalnih vjerodajnica

Na slici 5.8 prikazan je vremenski dijagram poziva metoda tj. komunikacije između razreda. Metoda *SetPKI* započinje pozivom udaljene metode *GetCACertificate* web usluge *CertificateService* (1). Nakon što je digitalna

vjerodajnica *Poslužitelja digitalnih vjerodajnica* dohvaćena (2), *CertificateManager* poziva metodu *ImportCertificate* razreda *CryptoManager* kojom pohranjuje digitalnu vjerodajnicu u *TrustedRoot* repozitorij (3). Nadalje, *CertificateManager* poziva metodu *GenerateCSR* razreda *CryptoManager* predavajući pritom jedinstveno ime logičkog čvora, duljinu ključa i vremenski interval valjanosti digitalne vjerodajnice kao parametre (4). Metoda *GenerateCSR* vraća generirani zahtjev za digitalnom vjerodajnicom natrag do *CertificateManager*-a (5). *CertificateManager* šalje zahtjev za digitalnom vjerodajnicom do *CertificateService*-a pozivom njegove udaljene metode *SignCSR* (6). Za potpisivanje primljenog zahtjeva za digitalnom vjerodajnicom, *CertificateService* poziva metodu *SignCSR* razreda *CryptoManager* (7) koja kao rezultat vraća digitalnu vjerodajnicu logičkog čvora (8). *CertificateService* vraća potpisanu digitalnu vjerodajnicu natrag do *CertificateManager*-a koji je poslao zahtjev (9), te istu objavljuje pozivajući pri tomu metodu *PublishCertificate* razreda *Provider* koji digitalnu vjerodajnicu pohranjuje u *Listu objavljenih digitalnih vjerodajnica* (11). Primljenu potpisanu digitalnu vjerodajnicu *CertificateManager* pohranjuje u *Personal* repozitorij pozivajući pri tomu metodu *ImportCertificate* razreda *CryptoManager* (10). Time završava metoda *SetPKI* te ujedno i protokol dodjele digitalnih vjerodajnica.

5.2.2 Protokol razmjene digitalnih vjerodajnica

Neposredno prije početka komunikacije između dvaju logičkih čvorova, *Overlay* poziva metodu *SetCommunicationReq* razreda *CertificateManager* kao bi osigurao da sigurnosni protokoli koje će se koristiti tijekom komunikacije u repozitoriju digitalnih vjerodajnica imaju sve potrebne digitalne vjerodajnice. Na slici 5.9 prikazano je ostvarenje protokola razmijene digitalnih vjerodajnica.



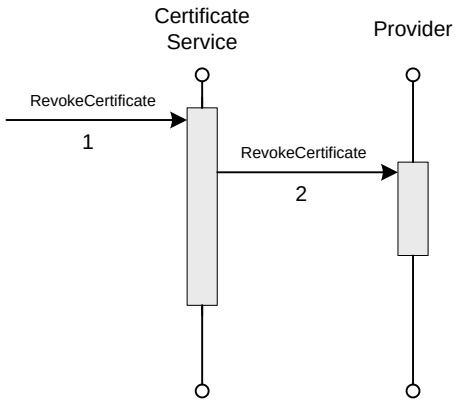
Slika 5.9 Protokol razmjene digitalnih vjerodajnica

Metoda *SetCommunicationReq* započinje pozivom udaljene metode *GetNodeCertificate* drugog razreda *CertificateManager* koji se nalazi na drugom fizičkom čvoru (1). Adresu druge instance razreda *CertificateManager* daje *Overlay*. *CertificateManager* na drugom fizičkom čvoru poziva metodu *GetCertificateFromStore* razreda *CryptoManager* koja vraća digitalnu vjerodajnicu traženog logičkog čvora (2,3). Dobavljenu digitalnu vjerodajnicu *CertificateManager* vraća do *CertificateManager*-a koji je poslao zahtjev za digitalnom vjerodajnicom (4). Nadalje, *CertificateManager* provjerava da li je primljena digitalna vjerodajnica vjerodostojna predajući je kao parametar metodi *VerifyCertificate* razreda *CryptoManager* (5,6). Ako je digitalna vjerodajnica vjerodostojna pozivom metode *IsRevoked* razreda *CertificateService* i predavajući ime logičkog čvora vlasnika digitalne vjerodajnice, *CertificateManager* provjerava da li je opozvana (7). *CertificateService* poziva metodu *IsRevoked* razreda *Provider* (8) koja pretražuje listu opozvanih digitalnih vjerodajnica i vraća rezultat pretraživanja (9). Rezultat pretraživanja *CertificateService* vraća natrag do *CertificateManager-a* (10). Ako je primljena

digitalna vjerodostojna i nije opozvana *CertificateManager* poziva metodu *ImportCertificate* razreda *CryptoManager* koja pohranjuje digitalnu vjerodajnicu u *Personal* repozitorij (11).

5.2.3 Protokol opoziva digitalne vjerodajnice

Kako je *CertificateService* izložen kao web usluga, onda digitalnu vjerodajnicu nekog logičkog čvora može opozvati svatko tko ima pristup toj web usluzi. Prilikom opoziv digitalne vjerodajnice mora se osim naziva logičkog čvora vlasnika digitalne vjerodajnice navesti i razlog opoziva. Trenutno ne postoji mehanizam koji će odlučivati tko ima pravo opozivati digitalne vjerodajnice i da li su razlozi opoziva valjani.

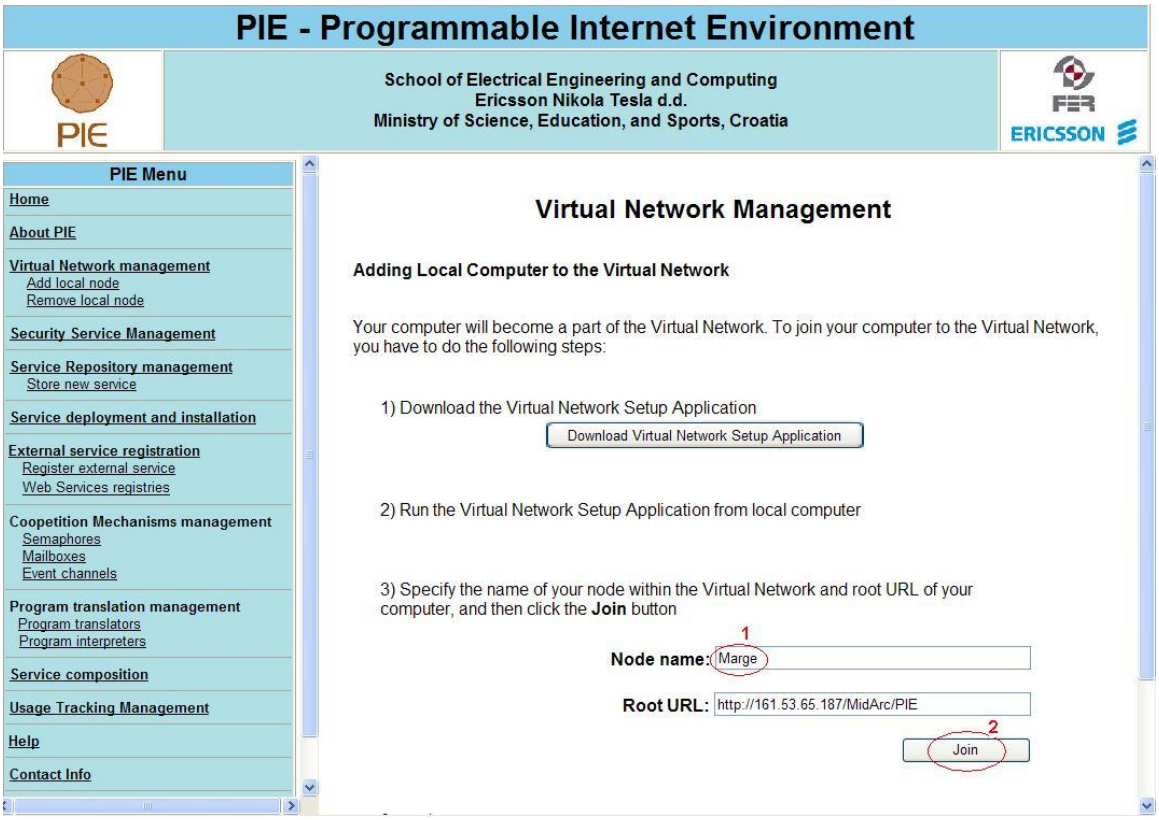


Slika 5.10 Protokol opoziva digitalne vjerodajnice.

Na slici 5.10 prikazan je vremenski dijagram protokola opoziva digitalne vjerodajnice. Protokol opoziva digitalne vjerodajnice započinje pozivom udaljene metode *RevokeCertificate* web usluge *CertificateService* kojoj se predaje ime logičkog čvora čija se digitalna vjerodajnica opoziva i razlog opoziva (1). *CertificateService* potom zahtjev za opozivom digitalne vjerodajnice prosljeđuje razredu *Provider* pozivom njegove metode *RevokeCertificate* (2). Ukoliko tražena digitalna vjerodajnica postoji na listi objavljenih digitalnih vjerodajnica, *Provider* je briše s te liste i stavlja ju na listu opozvanih digitalnih vjerodajnica.

5.3 Primjer izvođenja

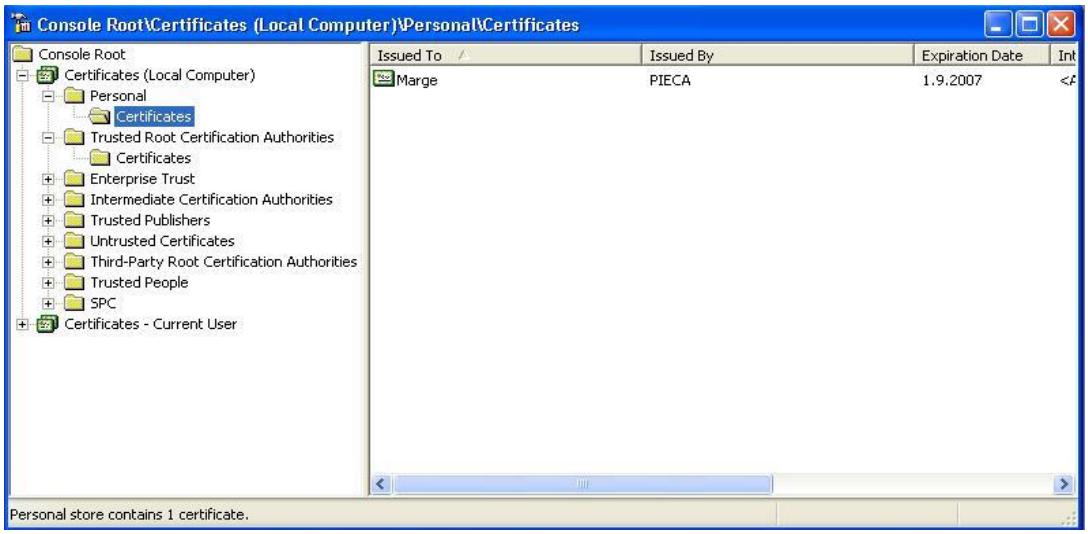
Slika 5.11 prikazuje primjer stvaranja logičkog čvora pod imenom *Marge*. Kroz *Web korisničko sučelje* unose se potrebne informacije o logičkom čvoru (1). Ime logičkog čvora mora biti jedinstveno jer se ono u okolini PIE koristi ujedno kao dio adrese logičkog čvora, pa ako bi dva logička čvora koja pripadaju pod isti fizički čvor imala jednaka imena protokoli usmjeravanja poruka ne bi znali kojem od čvorova je poruka namijenjena. Nakon unosa svih traženih informacija, pritiskom na gumb (2) započinje proces stvaranja logičkog čvora. Sa procesom stvaranja logičkog čvora započinje i proces uključivanja logičkog čvora u infrastrukturu javnih ključeva okoline PIE, izvršava se protokol dodjele digitalne vjerodajnice.



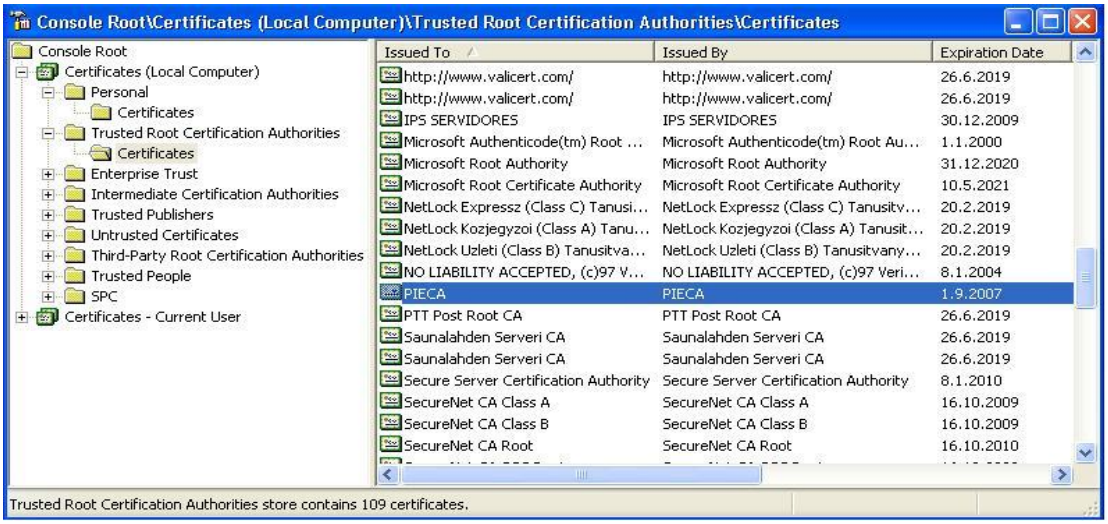
Slika 5.11 Primjer stvaranja logičkog čvora

Na slikama 5.12 i 5.13 se vidi da se kao rezultat izvođenja protokola dodjele digitalne vjerodajnice u repozitoriju digitalnih vjerodajnica na fizičkom čvoru

stvorile digitalna vjerodajnica logičkog čvora pod nazivom *Marge* i digitalna vjerodajnica *Poslužitelja digitalnih vjerodajnica PIECA* koje zajedno čine lanac povjerenja s vjerodajnicom *Poslužitelja digitalnih vjerodajnica* kao vrhovnom vjerodajnicom.



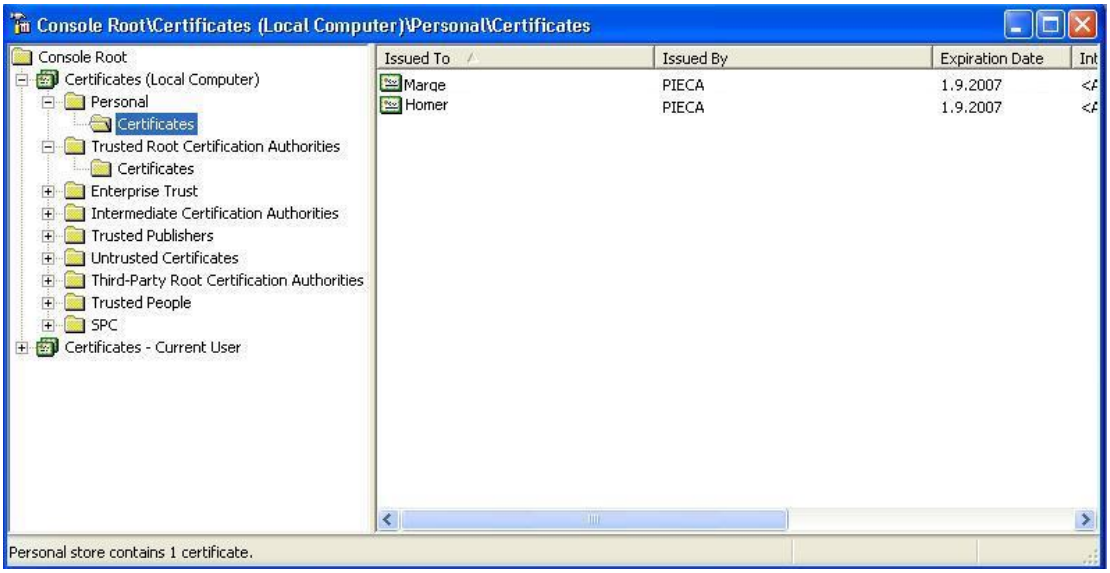
Slika 5.12 Personal repozitorij nakon izvršavanja protokola dodjele digitalne vjerodajnice



Slika 5.13 Trusted Root repozitorij nakon izvršavanja protokola dodjele digitalne vjerodajnice

U nastavku, logički čvor *Marge* želi uspostaviti sigurni komunikacijski kanal s logičkim čvorom *Homer*. Logički čvor *Marge* i logički čvor *Homer* dio su iste

infrastrukture javnih ključeva što znači da njihove digitalne vjerodajnice s digitalnom vjerodajnicom *Poslužitelja digitalnih vjerodajnica PIECA* tvore lanac povjerenja. Da bi uspostavili sigurni komunikacijski kanal *Upravitelj digitalnim vjerodajnicama* koji poslužuje logički čvor *Marge* dohvaća digitalnu vjerodajnicu logičkog čvora *Homer* preko *Upravitelja digitalnim vjerodajnicama* koji poslužuje logički čvor *Homer*.



Slika 5.14 Personal repozitorij nakon izvršavanja protokola razmjene digitalnih vjerodajnica

Na slici 5.14 prikazan je repozitorij u kojem se nalaze digitalne vjerodajnice logičkih čvorova *Marge* i *Homer* nakon izvršenog protokola razmjene digitalnih vjerodajnica. Za razliku od digitalne vjerodajnice logičkog čvora *Marge*, digitalna vjerodajnica logičkog čvora *Homer* ne sadrži informacije o privatnom ključu.

6 Zaključak

U ovom diplomskom radu ostvaren je sustav za automatsko upravljanje javnim ključevima u računalnom sustavu zasnovanom na uslugama koji se izvodi u komunikacijskom prostoru prividne mreže. Ostvareni su elementi *Upravitelj digitalnim vjerodajnicama* i *Poslužitelj digitalnih vjerodajnica* koji čine infrastrukturu javnih ključeva okoline PIE.

Upravitelj digitalnim vjerodajnicama upravlja digitalnim vjerodajnicama logičkih čvorova prividne mreže okoline PIE. *Poslužitelj digitalnih vjerodajnica* izdaje digitalne vjerodajnice logičkim čvorovima te održava liste objavljenih i opozvanih digitalnih vjerodajnica. Komunikacijom tih dvaju elemenata ostvarena su tri protokola infrastrukture javnih ključeva okoline PIE: protokol dodjele digitalnih vjerodajnica, protokol razmjene digitalnih vjerodajnica te protokol opoziva digitalnih vjerodajnica.

Programska arhitektura infrastrukture javnih ključeva okoline PIE sastoji se od pet modula od kojih se svaki sastoji od jednog ili više razreda. Razred *CertificateManager* predstavlja programsko ostvarenje *Upravitelja digitalnim vjerodajnicama*, dok je razred *CertificateService* izložen je kao web usluga i predstavlja programsko ostvarenje *Poslužitelja digitalnih vjerodajnica*. Oba razreda koriste zajednički modul *CertificateLibrary* koji stvara i upravlja digitalnim vjerodajnicama. Dodatno, razred *CertificateService* koristi modul *Providers* čime je omogućeno ostvarenje liste objavljenih i liste opozvanih digitalnih vjerodajnica na dva načina: u obliku XML dokumenata i u obliku tablica u SQL bazi podataka.

Jedno od otvorenih pitanja nakon ovog rada je mogućnost neovlaštenog opoziva digitalnih vjerodajnica. Opozivom digitalne vjerodajnice logičkog čvora taj čvor gubi mogućnost uspostave sigurnog komunikacijskog kanala s ostalim čvorovima. Taj problem je moguće riješiti uvođenjem uloga koje bi definirale skup sudionika s pravom opoziva digitalnih vjerodajnica.

7 Literatura

[1] Alan G. Konheim: **„Computer Security and Cryptography“,** Wiley siječanj 2007

[2] S. Lloyd, C. Adams **„Understanding the Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations“**, Sams, studeni 1999

[3] Singh, Munindar P., Michael N. Huhns: „**Service-Oriented Computing: Semantics, Processes, Agents“**, John Wiley & Sons, London, 2005.

[4] M. P. Papazoglou, W. J. van den Heuvel: "**Service-Oriented Computing: State-of-the-Art and Open Reserach Issues"**, *TI/RS/2003/123 (SOBI-1/D2.3),* Telematica Instituut, Enschede, Nizozemska, 2003.

[5] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, S. Weerawarana: **"Unraveling the web services web: An introduction to SOAP, WSDL, UDDI",** *IEEE Internet Computing*, ožujaktravanj 2002.

[6] **"Web Service Conceptual Architecture (WSCA 1.0)"**, *IBM Technical White Paper*, http://www- 3.ibm.com/software/solutions/webservices/pdf/WSCA.pdf, svibanj 2001.

[7] T. Bray, J. Paoli, C. M. Sperberg-McQueen: **"Extensible markup language (XML) 1.0. Recommendation (Third Edition)"**, *W3C*, http://www.w3.org/TR/REC-xml, veljača 2004.

[8] D. Box, D. Ehnebuske, G. Kakivaya, A, Layman, et al.: **"Simple Object Access Protocol (SOAP) 1.1"**, *W3C Note*, http://www.w3.org/TR/SOAP, svibanj 2000.

[9] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana: "**Web Services Description Language (WSDL) 1.1"**, *W3C Note*, http://www.w3.org/TR/wsdl, ožujak 2001.

[10] D. Škvorc: **"Prividna mreža računalnih sustava zasnovanih na uslugama",** magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2006.

[11] M. Podravec: **"Otkrivanje i postavljanje usluga u sustavima zasnovanim na uslugama",** magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2006

[12] Andro Milanović: **"Programski model zasnovan na uslugama"**, doktorska disertacija, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2005.
[13] Ivan Gavran: **"Korisnički jezik programskog modela zasnovanog na uslugama"**, magistarski rad, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu, Zagreb, 2006.

[14] Davor Čapalija: „**Objava/pretplata mehanizmi za ostvarivanje mreža zasnovanih na sadržaju**“, Diplomski rad, Fakultet elektrotehnike i računarstva, 2005.

[15] Andro Milanović, Siniša Srbljić, Daniel Skrobo, Davor Čapalija, Saša Rešković: “**Coopetition Mechanisms for Service-Oriented Distributed System”**, članak prihvaćen za objavljivanje na *The 3rd International Conference on Computing, Communications and Control Technologies (CCCT '05),* Austin, SAD, 2005.

[16] A. Menezes, P. Orschot, S. Vanstone: „**Handbook of Applied Cryptography“,** CRC press, 1996

[17] **„Web Services Security: SOAP Message Security 1.0 (WS-Security 2004**)“, OASIS Open 2003-2004, veljača 2004. http://docs.oasis-open.org/wss/200402-wss-soap-message-security-1.0.pdf

[18] A. Loyd: „**Certification Authorities and X.509“**, listopad 2000. http://lgi2p.ema.fr/~frausto/pki/X509/X509.htm

[19] M.Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon: **„XML Signature Syntax and Processing“**, veljača 2002. http://www.w3.org/TR/xmldsig-core/

[20] T. Imamura, B. Dillaway, E. Simon: „**XML Encryption Syntax and Processing“,** prosinac 2002. http://www.w3.org/TR/xmlenc-core/

[21] Singh, Munindar P., Michael N. Huhns: **„Service-Oriented Computing: Key Concepts and Principles”**, *IEEE Internet Computing,* siječanj/veljača 2005.

[22] **Programirljiva internetska okolina (PIE)**: http://www.pie.fer.hr

[23] A. O. Freier, P. Karlton, P. C. Kocher: **„Secure Socket Layer (SSL)“,** http://wp.netscape.com/eng/ssl3, studeni 1996

[24] S. Kent, R. Atkinson: **„Internet Protocol security (Ipsec)“,** http://www.ietf.org/rfc/rfc2401.txt, studeni 1998

[25] B. Ramsdell: **„Secure Multi-Purpose Internet Mail Extensions (S/MIME)“,** http://www.faqs.org/rfcs/rfc2633.html, lipanj 1999

[26] D. Atkins, W. Stallings, P. Zimmermann: **„Pretty Good Privacy (PGP**)“, http://www.faqs.org/rfcs/rfc1991.html, kolovoz 1996

[27] M. Hayoz: **„Introducing SSL The Secure Sockets Layer Protocol“**, lipanj 2003

[28] D. Raggett, A. Le Hors, I. Jacobs: **„HTML 4.01 Specification“** http://www.w3.org/TR/html401/ , prosinac 1999

[29] **„Introduction to UDDI: Important Features and Functional Concepts“**, http://uddi.org/pubs/uddi-tech-wp.pdf, listopad 2003

[30] E. Christensen, D. Box, F. Curbera, **„Web Services Addressing (WS-Addressing)“**, http://www.w3.org/Submission/ws-addressing/, W3C, rujan 2004

[31] OpenSSL http://www.openssl.org/docs/

[32] J. Linn **„Privacy Enhancement for Internet Electronic Mail“**, http://www.ietf.org/rfc/rfc1421.txt, IETF, veljača 1993

[33] U.S. DEPARTMENT OF COMMERCE / National Institute of Standards and Technology: **“Data Encryption Standard (DES**)”, http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf , listopad 1999

[34] S. Teiwes, P. Hartmann, D. Kuenzi: **„RFC 3058 - Use of the IDEA Encryption Algorithm in CMS“** http://www.faqs.org/rfcs/rfc3058.html, veljača 2001

[35] Federal Information: **„Advanced Encryption Standard (AES)“**, http://www.csrc.nist.gov/publications/fips/fips197/fips-197.pdf, studeni 2001

[36] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson: **"The Twofish Encryption Algorithm"**, http://www.schneier.com/paper-twofish-paper.pdf, lipanj 1998

[37] B. Schneier: **„The Blowfish Encryption Algorithm -- One Year Later“**, http://www.schneier.com/paper-blowfish-oneyear.html, rujan 1995

[38] C. Adams: **“The CAST-128 Encryption Algorithm”,** Entrust Technologies http://tools.ietf.org/html/rfc2144, svibanj 1997

[39] K.Kaukonen, R.Thayer: **“A Stream Cipher Encryption Algorithm”,** http://www.mozilla.org/projects/security/pki/nss/draft-kaukonen-cipher-arcfour-03.txt, srpanj 1999

[40] William C. Barker: **“Recommendation for the Triple Data Encryption Algorithm“**, http://csrc.nist.gov/publications/nistpubs/800-67/SP800-67.pdf, NIST, svibanj 2004

[41] **„Microsoft .Net Development“**, http://msdn2.microsoft.com/en-us/library/aa139615.aspx, Microsoft

[42] **„Development Tools and Languages“**, http://msdn2.microsoft.com/en-us/library/aa187916.aspx, Microsoft

[43] M. C. Two, C. Poole, J. Cansdale, G. Feldman: **„Nunit“** , http://www.nunit.org/