UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Aleksander Pejčić

Agencija za časovno žigosanje

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: prof. dr. Denis Trček

Ljubljana, 2010

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01684/2010
Datum: 01.09.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **ALEKSANDER PEJČIĆ**

Naslov: **AGENCIJA ZA ČASOVNO ŽIGOSANJE**
TIME STAMPING AUTHORITY

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

V nalogi obdelajte problematiko časovnega žigosanja v povezavi z digitalnim podpisovanjem. Delo razdelite na dva sklopa, kjer naj prvi sklop obsega pregled standardov in tehnologij, ki se uporabljajo pri časovnem žigosanju, drugi sklop pa naj obsega podrobnosti za vzpostavitev konkretne agencije, ki ponuja storitve zunanjemu okolju. Implementirajte tudi prototip tovrstne agencije.

Mentor:

prof. dr. Denis Trček

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Franc Solina

Namesto te strani **vstavite** original izdane teme diplomskega dela s podpisom mentorja in dekana ter žigom fakultete, ki ga diplomant dvigne v študentskem referatu, preden odda izdelek v vezavo!

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani Aleksander Pejčić,

z vpisno številko 63040122,

sem avtor diplomskega dela z naslovom:

Agencija za časovno žigosanje

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Denisa Trčka;
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela;
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 14.9.2010 Podpis avtorja:

Zahvala

Vsekakor so na prvem mestu starši in sestra, ki so me tekom študija bodrili in podpirali ter mi bili vzgled delavnosti.

Posebna zahvala gre prof. dr. Denisu Trčku, ki mi je svetoval pri izbiri teme za diplomsko delo in mi vseskozi pomagal z nasveti, popravki in komentarji.

Hvala tudi Emi za spodbudo in Petri za pomoč pri lektoriranju diplomske naloge ter prijateljem, ker je z vami življenje lepše.

Iskrena hvala!

Kazalo

Seznam uporabljenih kratic in simbolov

IJK	Infrastruktura javnih ključev (Public key infrastructure)
EZF	Enosmerne zgoščevalne funkcije
IETF	The Internet Engineering Task Force
HTTP	Hypertext Transfer Protocol
NTP	Network Time Protocol
HSM	Hardware Security Module
CA	Certificate Authority (Overitelj digitalnih certifikatov)
TSA	Timestamping Authority (Agencija za časovno žigosanje)
TTP	Trusted Third Party (Zaupanja vredna tretja oseba)
PGP	Pretty Good Privacy (Razmeroma dobra zasebnost)
CRL	Certificate Revocation List
DER	Distinguished Encoding Rules
XER	XML Encoding Rules
XML	Extensible Markup Language
XSD	XML Schema Definition
OID	Object Identifiers
ISO	International Organization for Standardization
RSA	Rivest, Shamir and Adleman
TCP/IP	Transmission Control Protocol and the Internet Protocol
DoS	Denial-of-Service attack

Povzetek

Diplomsko delo obravnava področje časovnega žigosanja, digitalnega podpisovanja in kriptografijo na splošno. Vsebina je razdeljena na dva večja sklopa, pri katerem je prvi sklop pregled standardov in tehnologij, ki se uporablja pri časovnem žigosanju. Drugi sklop pa obravnava agencijo za časovno žigosanje, kot entiteto, ki ponuja storitve časovnega žigosanja zunanjemu okolju. Predstavimo tudi smisel, potrebo in namen takih agencij. Opredelimo infrastrukturo javnih ključev (IJK) in različne kriptografske tehnike in standarde. Osrednji del naloge je predstavitev standarda RFC 3161 ki definira način, tehnike in podatkovne strukture, ki so potrebni za časovno žigosanje poljubnega digitalnega dokumenta. Izpostavimo tudi omejitve in pomanjkljivosti, ki jih standard prinaša. Podamo tudi več načinov ponujanja storitve časovnega žigosanja preko interneta. V zaključku je predstavljena programska oprema za implementacijo agencije za časovno žigosanje, ki je bila narejena v okviru diplomskega dela.

Ključne besede:

časovni žig, agencija za časovno žigosanje, infrastruktura javnih ključev, kriptografija, digitalni podpis, RFC 3161, zaupanje, spletna storitev

Abstract

This thesis deals with the field of timestamping, digital signing and cryptography in general. Content is divided into two main parts, first being the overview of standards and technology behind timestamping. The second part deals with trusted timestamping authority as an entity, which offers timestamping services to outside enviroment. The need for this kind of authorities and their purpose is also explained. We present public key infrastructure (PKI) and various cryptographic standards and technologies. The central piece of this thesis is a presentation of RFC 3161 standard that defines the way, technology and data structures that are needed for timestamping any kind of digital document. The shortcomings and limitations of RFC 3161 standard are also presented. We also present multiple ways of offering timestamping services over internet. At the end we present the software for implementing trusted timestamping authority, that was made as a part of this thesis.

Key words:

timestamp, trusted timestamp authority, public key infrastructure, cryptography, digital signature, RFC 3161, trust, web service

Poglavje 1

Uvod

1.1 Motivacija in cilji

Ideja o časovnem žigosanju je pravzaprav stara nekaj stoletij, ko je angleški znanstvenik Robert Hooke objavil anagram[1] odkritja svojega zakona ki ga še ni želel objaviti, vendar je želel imeti dokaz o odkritju. Motivacija za časovno žigosanje se je obdržala vse do današnjih dni. Namenjen je predvsem dokazovanju avtorstva nad določenimi podatki ob specifičnem času[9, s. 71]. Spremenila se je le izvedba časovnega žigosanja, saj se dandanes za to uporabljajo zaupanja vredne tretje osebe, imenovane agencije za časovno žigosanje. Razširil pa se je seveda tudi nabor uporabnosti časovnega žiga, saj ni več namenjen le dokazovanju avtorstva znanstvenih odkritij, ampak tudi avtorstva različnih dokumentov kot so digitalne pogodbe, elektronske pošte, plačilni nalogi itd.

Največkrat se končni uporabniki z agencijo za časovno žigosanje ne srečajo neposredno ampak le posredno, saj je temeljni element v sistemih kot so elektronski arhivi ali pa digitalne notarske storitve.

1.2 Pregled vsebine

V okviru naloge je predstavljeno predvsem časovno žigosanje kot je določeno s standardom RFC 3161, ki se opira na širšo množico ostalih standardov. Ti se uporabljajo za predstavitev podatkov s standardom ASN.1, predstavitev javnih certifikatov X.509 s standardom RFC 5280 in predstavitev časovnega žiga s standardom RFC 2630.

[1] "ceiiinosssttuv" predstavlja latinsko "ut tensio sic vis" (angl. "as is the extension, so is the force")

Ker je zaupanja vredno časovno žigosanje v veliki meri odvisno tudi od verodostojnosti trenutnega časa, je v nalogi predstavljeno tudi varno vzdrževanje točnega časa, ki je implementirano s protokolom NTP.

Poleg tega je tu tudi vprašanje prenosa podatkov preko interneta. V nalogi sta predstavljeni dve rešitvi in sicer s protokolom HTTP in pa s spletno storitvijo.

Predstavimo pa tudi način digitalnega podpisovanja, ki je ena izmed temeljih tehnologij pri časovnem žigosanju. Ker pa se časovno žigosanje ne izvaja nad celotno vsebino podatkov, je v nalogi predstavljena tudi tehnika podatkovnih izvlečkov, ki unikatno določijo podatek, ki jih kasneje lahko uporabimo za časovni žig.

Razložimo tudi kaj je to agencija za časovno žigosanje in kaj so nekateri izzivi, kot je skladnost z zakonodajo, ki jih je potrebno rešiti ob vzpostavitvi takšne agencije.

Poglavje 2

Časovno žigosanje

2.1 Splošno

Časovni žigi tvorijo skupaj z digitalnimi podpisi eno izmed temeljnih tehnologij v kriptografiji. Uporabljajo se predvsem v elektronskem poslovanju. Spodnji primer prikazuje primer problema, ki ga lahko rešimo z uporabo časovnih žigov.

Ana pošlje Mihu digitalno podpisan ček za 100€. Miha vzame ček in odide na banko, kjer ta ček unovči. Banka preveri digitalni podpis in ker je ta pravilen, Mihu nakaže denar. Vendar, ker je Miha zvit, si je digitalno podpisan ček shranil na računalniku in naslednji teden spet odide na banko, kjer mu ta ček ponovno uspe unovčiti. Problem lahko preprosto rešimo, če digitalnemu podpisu dodamo časovni žig, ko dokument nastane. Tako Mihu čeka ne uspe ponovno unovčiti, saj si banka zapomni časovni žig ob prvi unovčitvi čeka [8].

2.2 Infrastruktura javnih ključev

Infrastruktura javnih ključev (IJK, angl. public key infrastructure) je nabor tehnologij, metod, naprav, predpisov in entitet, ki so potrebne za varno in zaupno razdelitev, upravljanje in preklic digitalnih certifikatov[9, s. 69]. V kriptografiji je IJK dogovor za povezavo digitalnih certifikatov z identitetami oseb, ki jih overi vrhovna certifikatna agencija (CA, angl. certificate authority). Delo CA je opredeljeno z zakonodajo, saj v IJK nastopajo zaupanja vredne tretje osebe (TTP, angl. trusted third party). IJK se uporablja za enkripcijo, podpisovanje, vzpostavitev kriptiranega kanala itd.

V IJK imamo opravka s t.i. asimetričnimi ključi, ki jih lahko ustvarimo npr. z algoritmom RSA, s katerim lahko kasneje izvajamo tudi kriptografske

operacije. Pri tem pristopu ima vsaka oseba znotraj IJK dva ključa. Prvi je javni in je dostopen vsem ostalim udeležencem v pogovoru, drugi pa je privatni in je dostopen le njegovemu lastniku. Oba ključa skupaj tvorita par ki se lahko uporabi za različne kriptografske operacije. Vsak udeleženec v IJK ima lahko tudi overjen certifikat, ki vsebuje udeležencev javni ključ in informacije o lastniku certifikata. Ta certifikat služi predvsem za predstavitev lastnika javnega ključa ostalim udeležencem v IJK.

Varnost pri uporabi asimetričnih ključev izvira iz dejstva, da lahko kriptografske operacije s privatnim ključem opravlja le njegov lastnik. Prednost asimetričnih ključev je tudi v lažji distribuciji oz. izmenjavi ključev, saj javnega ključa ni potrebno varovati in skrivati, zato ga lahko lastnik prostovoljno dostavi vsakemu ki bi želel z njim komunicirati. Obratno pa je pri simetričnih ključih nujno da vsi udeleženci pri komuniciranju skrivajo ključ, saj je odkrit ključ neuporaben.

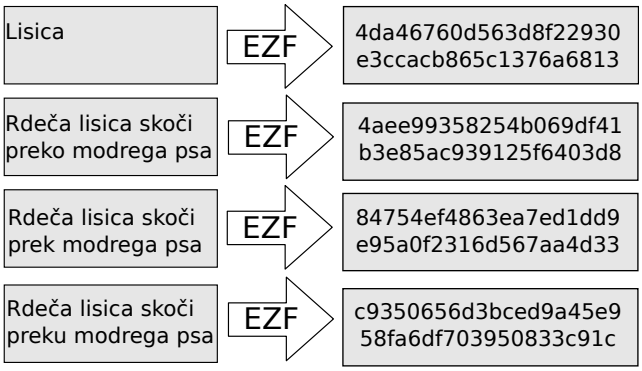
V praksi se privatni ključi hranijo v *Hardware Security Module* (HSM), ki je fizična naprava namenjena varovanju privatnih asimetričnih ključev. Vsaka nadaljnja uporaba privatnega ključa je možna le na napravi HSM. Privatni ključ se zato ne uporablja več neposredno, temveč tako, da uporabnik napravi HSM pošlje sporočilo in podatke o zahtevani operaciji, nato pa HSM uporabniku vrne rezultat operacije nad sporočilom s privatnim ključem. Naprave HSM so torej nekakšni fizični digitalni sefi, ki so namenjeni ne le varovanju svoje vsebine, temveč tudi razumejo svojo vsebino in jo znajo uporabljati.

2.2.1 Enosmerne zgoščevalne funkcije

Enosmerne zgoščevalne funkcije (EZF) omogočajo izračun podatkovnih izvlečkov. EZF kot vhod sprejme podatek poljubne dolžine, rezultat pa je niz fiksne dolžine. Vhod imenujemo sporočilo, izhod pa podatkovni izvleček. Ponavadi že zelo majhna sprememba v sporočilu povzroči veliko spremembo v podatkovnem izvlečku, kar prikazuje slika 2.1.

EZF za izračun podatkovnega izvlečka mora zadoščati naslednjim štirim zahtevam[9]:

- Izračun izvlečka za poljubno sporočilo mora biti računsko učinkovito.
- Računsko neobvladljivo je iskanje sporočila, ki ustreza danemu izvlečku. To nam zagotavlja da je funkcija enosmerna saj iz izvlečka ni mogoče izračunati sporočila. To pravilo imenujemo tudi *primarna rezistenca (angl. Preimage resistance).*



Slika 2.1: Primer izvlečka za funkcijo SHA-1

- Računsko neobvladljivo je iskanje spremembe sporočila, tako da se izvleček ne spremeni. Pravilo imenujemo tudi *sekundarna rezistenca (angl. second preimage resistance)*.

- Računsko neobvladljivo je iskanje dveh sporočil z istim izvlečkom. Pravilo imenujemo tudi *kolizijska rezistenca (angl. collision resistance)*.

Računsko neobvladljivo pomeni, da zahtevnost iskanja rešitve ni mogoče poiskati v polinomskem času, ampak zahtevnost narašča eksponentno, kar hitro pripelje do pojava kombinatorne eksplozije.

Zgornje lastnosti nam omogočijo, da namesto celotnega sporočila za časovno žigosanje uporabimo kar njegov izvleček. S tem lahko končni uporabnik agenciji za časovno žigosanje pošlje v žigosanje le izvleček in ne celotnega sporočila. V tem primeru celotno sporočilo nikoli ni vidno agenciji za časovno žigosanje, kar zagotovi sporočilu anonimnost saj zaradi primarne rezistence iz podatkovnega izvlečka ne moremo izračunati originalnega sporočila.

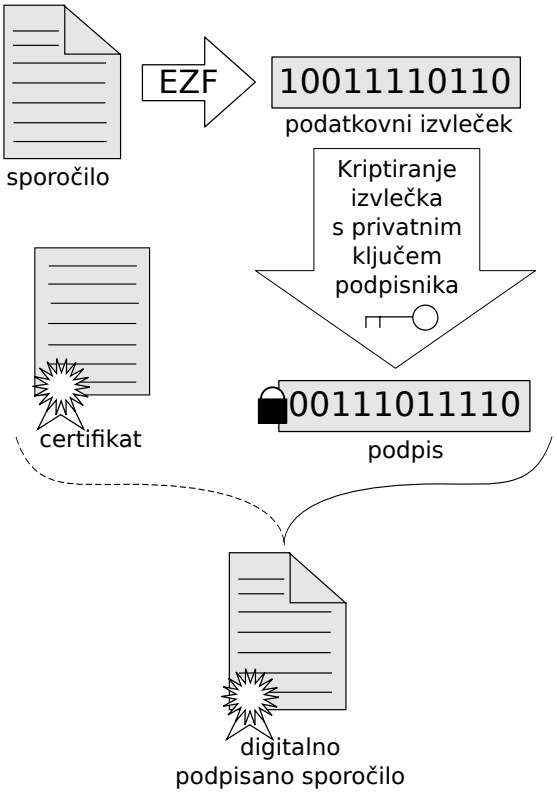
Velika prednost EZF je tudi, da algoritma za izračun podatkovnega izvlečka ni potrebno skrivati, zato ga lahko svobodno uporabijo vsi udeleženci v IJK.

Najbolj znane družine EZF so SHA, MD in RIPEMD[18].

2.2.2 Digitalni podpis

Digitalni podpis je način uporabe IJK, tako da lahko overimo in dokažemo celovitost danega sporočila. Digitalni podpis zadosti naslednjim zahtevam[8]:

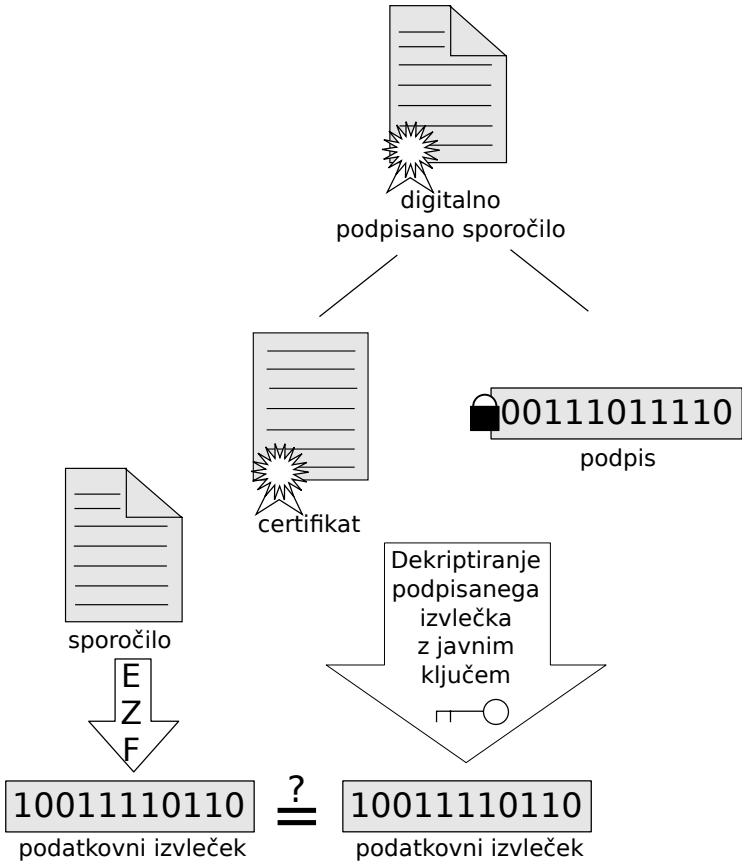
- Podpis je *verodostojen*. Prejemnik lahko verjame, da je podpisnik namenoma podpisal sporočilo.



Slika 2.2: Ilustracija poteka podpisovanja

- Podpisa se *ne da ponarediti*. Podpis je dokaz, da je sporočilo podpisal podpisnik in nihče drug.
- Podpis *ni ponovno uporabljiv*. Podpis je del sporočila in ga ni mogoče prenesti na drugo sporočilo.
- Podpisano sporočilo je *nespremenljivo*. Ko je sporočilo podpisano, se ga ne da spreminjati.
- Podpisa se ne da *zavrniti*. Ko je sporočilo enkrat podpisano, podpisnik ne more trditi, da tega ni podpisal.

Kot prikazuje slika 2.2, se digitalno podpisovanje izvaja nad vhodnim sporočilom, kateremu se z EZF izračuna podatkovni izvleček. Podpisnik nato izračunani podatkovni izvleček podpiše s svojim privatnim ključem. Rezultat podpisovanja je digitalni podpis podatkovnega izvlečka, ki lahko služi kot dokaz da je bilo osnovno sporočilo podpisano s privatnim ključem, do katerega pa ima



Slika 2.3: Ilustracija poteka preverjanja podpisa

dostop le njegov lastnik. Podpisu je priložen certifikat, ki ga lahko uporabniki digitalnega podpisa uporabijo za preverjanje podpisa, saj ta nosi informacije o podpisniku in njegov javni ključ.

Preverjanje digitalnega podpisa se izvaja na sledeč način. Preverjevalec razčleni digitalno podpisano sporočilo na podpisan izvleček in podpisnikov certifikat. Nato s pomočjo javnega ključa v certifikatu dekriptira podpisan izvleček in s tem pridobi originalni podatkovni izvleček, ki ga je podpisani podpisal. Podpis je resničen, če se originalni podatkovni izvleček ujema z izvlečkom, ki ga preverjevalec ponovno izračuna iz originalnega sporočila z EZF. Seveda je potrebno paziti da preverjevalec uporabi enako EZF kot jo je uporabil podpisnik. Postopek prikazuje slika 2.3.

Takšna shema uporabe javnih in privatnih ključev nam zagotavlja, da lahko dokažemo lastnika digitalnega sporočila, saj lahko le njegov javni ključ pravilno dekriptira podpisani podatkovni izvleček.

Če pa poznamo lastnika podpisa, nam digitalni podpis omogoča preverjanje integritete sporočila v komunikacijah. Pošiljatelj skupaj s sporočilom pošlje tudi digitalni podpis sporočila, s katerim lahko prejemnik preveri da poslano sporočilo na poti ni bilo spremenjeno. To nam seveda zagotavlja isti podatkovni izvleček iz sporočila in podpisa.

2.2.3 Obvladovanje zaupanja

Pri obvladovanju zaupanja želimo doseči predvsem tri stvari:

- zaupanje v lastnika ključa,
- zaupanje v podpis,
- zaupanje v veljavnost ključa.

X.509

X.509 je standard za IJK, ki omogoča avtorizacijo in predstavitev uporabnikov v globalnem distribuiranem direktoriju[9, s. 70]. Njegova primarna naloga je povezati javni del asimetričnega ključa z uporabnikovo identiteto. Vsak certifikat je podpisan s strani izdajatelja, kar nam omogoča da preverimo kdo je certifikat overil in se nato odločimo ali temu certifikatu zaupamo. Iz prej napisanega je razvidno, da lahko z X.509 gradimo hierarhični model izdanih certifikatov. Vrhovni overitelj je CA, katerega certifikat je vedno dosegljiv če sledimo t.i. verigi overjanja od trenutnega certifikata do vrha. Standard določa tudi sezname preklicanih certifikatov (angl. Certificate Revocation List, CRL)[9, s. 69]. V tem modelu nastopa vrhovni CA kot TTP.

PGP

PGP (angl. Pretty Good Privacy) se od standarda X.509 razlikuje v tem da nima hierarhične strukture[7, s. 278]. V tem modelu lahko vsakdo podpiše certifikat in s tem nastopa kot vrhovni CA. Z PGP lahko ustvarimo tudi t.i. samo-podpisane certifikate, ki pa ni primeren za elektronsko poslovanje saj podpisanega certifikata ni preverila TTP, zato izgubimo zaupanje v lastnika certifikata. Vendar pa model predvideva t.i. mrežo zaupnja (angl. web of trust), kjer je vsak certifikat overjen s strani večih uporabnikov. S tem zvišamo stopnjo zaupanja v lastnika. Model se uporablja predvsem pri preprostejših

storitvah, kot je npr. podpisovanje elektronske pošte. Odprtokodna različica PGP je GnuPG[1].

2.3 Postopek časovnega žigosanja

Časovno žigosanje je v temelju zelo podobno digitalnemu podpisovanju, z razliko da je potrebno podpisu dodati tudi čas kdaj se je sporočilo podpisalo. S tem časovni žig razširja digitalni podpis, tako da je poleg dokazovanja lastništva možno dokazovati tudi čas nastanka digitalnega podpisa. Zelo pomembno je tudi dejstvo, da uporabniku ni potrebno časovno žigosati celotnega dokumenta ampak le podatkovni izvleček, saj je ta unikaten za vsako sporočilo.

2.3.1 Izdaja časovnega žiga

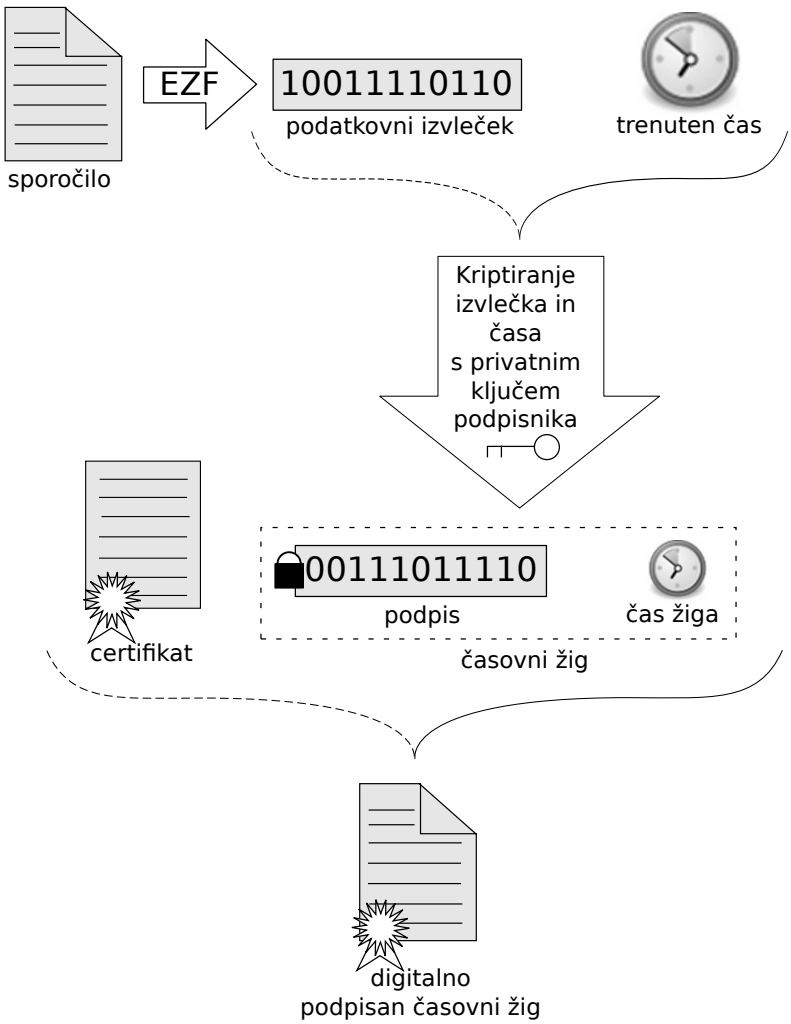
Čeprav obstajajo tudi druge sheme oz. načini časovnega žigosanja, se v praksi največkrat uporablja način ki temelji na infrastrukturi javnih ključev (IJK). Z IJK lahko preprosto distribuiramo časovni žig, ker ima ta vgrajen certifikat in s tem javni ključ podpisnika. To omogoča uporabniku časovnega žiga tudi enostavno preverjanje in pregledovanje časovnega žiga, saj mora le dekriptirati časovni žig z javnim ključem podpisnika. Zaradi narave asimetričnih ključev, ki se uporabljajo v IJK, pa je kakršnokoli spreminjaje časovnega žiga s strani uporabnika nemogoče, ker bi zato potreboval privatni ključ podpisnika [10], ki pa ga seveda pozna le podpisnik. Takšna shema nam nudi visoko zaupanje v varnost samega žiga.

Ilustracija 2.4 nam prikazuje potek izdaje časovnega žiga. V temelju uporablja že znano tehnologijo digitalnega podpisovanja, le da jo razširja s časovnim žigom.

2.3.2 Preverjanje časovnega žiga

Preverjanje časovnega žiga poteka na podoben način kot pri preverjanju digitalnega podpisa. Za dekripcijo digitalno podpisanega časovnega žiga se uporabi javni ključ, ki je del priloženega certifikata. Nato lahko uporabnik preveri čas ob katerem se je izdal časovni žig. Prav tako pa mora preveriti veljavnost podpisa, tako da izračuna podatkovni izvleček sporočila za katerega je izdal

[1]GNU Privacy Guard - http://www.gnupg.org



Slika 2.4: Ilustracija poteka časovnega žigosanja

časovni žig in ga primerja z izvlečkom, ki je del časovnega žiga. Časovni žig je veljaven, če se izvlečka ujemata.

2.4 Vzdrževanje točnega časa

Ker je časovni žig odvisen od časa ob katerem nastane zahteva za časovno žigosanje, je izredno pomembno natančno obravnavanje časa. Problema sta predvsem dveh izvorov.

Prvi izvira iz dejstva, da računalniški sistemi skrbijo za merjenje časa z mehansko oscilacijo kristala ki oscilira z določeno frekvenco. Tu pa se pojavi prvi problem, saj je oscilacija kristala natančna le do določene mere. Poleg tega pa na kristal vplivajo okoljski dejavniki, kot tudi samo staranje kristala.

Drugi problem, ki vpliva na točnost časa s katerim izvajamo časovno žigosanje, pa je zaupanje uporabnikov v administratorja sistema ki izvaja časovno žigosanje, saj ima ta možnost spreminjanja sistemske ure.

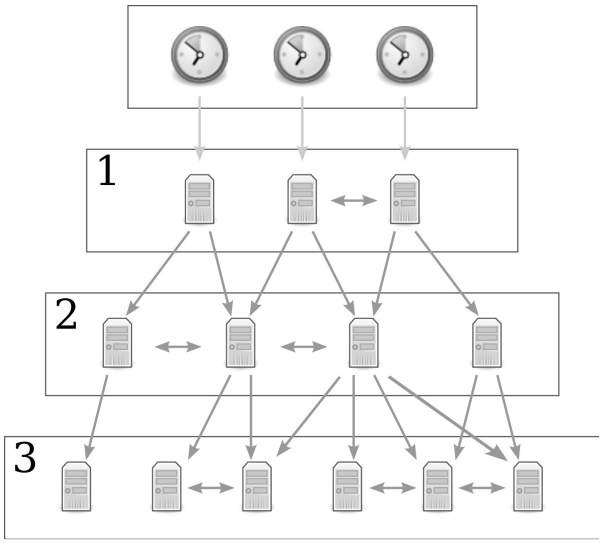
Torej sta problema predvsem kako mehansko natančno meriti čas in kako doseči zaupanje v izmerjen čas.

Čas se da izredno natančno meriti z t.i. atomskimi urami, ki za merjenje časa uporabljajo mikrovalovni signal, le tega pa oddajajo elektroni ob spremembi energijskega stanja. Takšne ure lahko dosežejo izredno natančnost, večina pa jih zagotavlja natančnost 10^{-9} sekunde na dan. Seveda pa večina operaterjev agencij za časovno žigosanje nima denarnih sredstev za nakup atomskih ur. V ta namen obstajajo javne in privatne agencije, ki širši javnosti nudijo uporabo časa atomskih ur kar preko različnih komunikacijskih kanalov. Tak kanal lahko ustvarimo z protokolom *Network Time Protocol* (NTP)[6], ki ga lahko uporabimo preko interneta.

Če za sinhronizacijo časa uporabimo protokol NTP ki za izvor uporablja izredno natančen čas, lahko rešimo tudi problem zaupanja v izmerjen čas, saj lahko program za časovno žigosanje namesto sistemske ure uporablja kar zunanji čas, ki ga prejme preko protokola NTP. S tem se izognemo administratorju sistema in povečamo zaupanje uporabnikov v časovni žig.

Protokol NTP je zasnovan tako, da skuša čim bolj izničiti slabe lastnosti paketno preklopnega komunikacijskega kanala kot je internet. Predvsem skuša izničiti efekt različnih zakasnitev ki nastanejo zaradi izbire drugačnih poti po internetu, različnih paketov[6]. Preprostejša verzija protokola NTP ki ne potrebuje začasnih pomnilnikov za stanja, je definirana s standardom RFC 5905.

NTP določa tudi shemo izvorov točnega časa, kot hierarhično strukturiran sistem.



Slika 2.5: Shema hierarhične strukture izvora časa

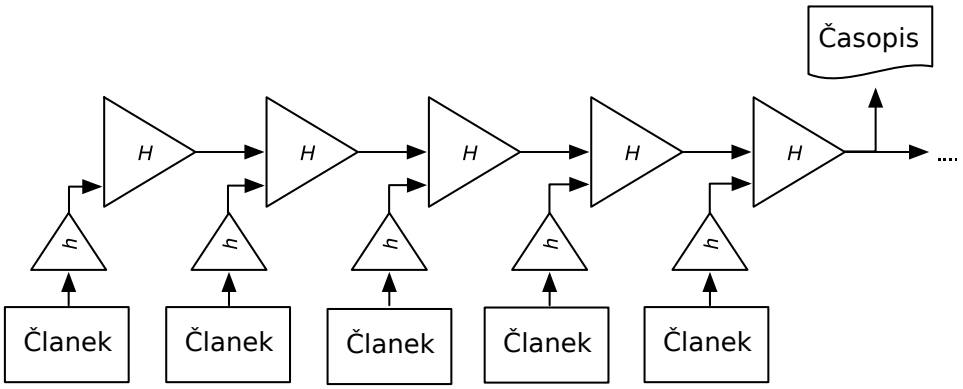
Slika 2.5 prikazuje strukturo izvora časa v NTP protokolu. Najpomembnejši je nivo ena, ki svoj čas meri neposredno iz meritvene naprave, kot je npr. atomska ura. Nato se izmerjen čas v nivoju ena seli na nižja nivoja preko protokola NTP. Možna je tudi sinhronizacija med računalniškimi sistemi na istem nivoju, kjer prihaja do izraza predvsem detekcija različnih zakasnitev ki nastajajo v kanalu. Primarna sinhronizacija poteka vedno od višjega nivoja k nižjemu. Vsakemu nivoju v shemi je predpisana tudi potrebna natančnost, kjer so seveda sistemi na višjem nivoju bolj natančni. Končni uporabniki se ponavadi povezujejo na strežnike NTP, ki so na nivoju tri.

NTP za zapis časa uporablja skupaj 64 bitov, kjer je 32 bitov namenjenih zapisu sekund od leta 1900 dalje, ostalih 32 bitov pa delu sekunde, kar omogoča natančnost 2^{-32} oz. 233 piko sekund ($1/2^{32} * 1e^{12}$). Zaradi tako kratkega zapisa sekund bo leta 2036 prišlo do preliva, vendar ker protokol NTP deluje tako da zapiše le razliko v času med strežnikom in uporabnikom in ne absolutno vrednost časa od leta 1900 dalje, to ne bo povzročilo hujših posledic.

Primer časa zapisanega v NTP 64 bitnem zapisu:

- celi del: 0000 0110 0010 0011 0100 0010 1011 0000 =
  $= 102974128s = 1191dni, 19ur, 55min, 28s$

- del sekunde: 0111 1001 0111 1010 1110 0001 1000 1101 =
  $= 2038096269/2^{32} = 474ms, 531\mu s, 266ns, 139pc$



Slika 2.6: Struktura povezanih časovnih žigov

2.5 Modeli časovnega žigosanja

Čeprav se to delo posveča predvsem načinu ki temelji na tehnologiji IJK, obstajajo tudi drugi pristopi za izvajanje časovnega žiga [10]. Zanimivi so predvsem tisti, ki rešujejo probleme ki nastanejo pri uporabi IJK, kot je npr. razkritje privatnega ključa.

Povezano časovno žigosanje je shema, kjer so časovni žigi medsebojno odvisni in skupaj tvorijo celoto. Uporaben je predvsem v primerih, ko imamo opravka z množico vhodnih sporočil ki so del logične končne celote.

Slika 2.6 prikazuje uporabo povezanega časovnega žigosanja na primeru, kjer časopis dobi časovni žig ki ga tvorijo posamezni članki v časopisu. Nemogoče je tudi spreminjati posamezne dele časovnega žiga, saj bi to razveljavilo celotno verigo [2, 5]. Takšno žigosanje se tudi ne zanaša na privatne ključe, kar izniči probleme ki nastanejo ob njihovi uporabi in s tem poveča zaupanje v storitev [2]. Oteži preverjanje časovnega žiga, saj je potrebno slediti celotni verigi od trenutnega časovnega žiga nazaj vse do prvega žiga.

Standarda, ki opisujeta povezano časovno žigosanje sta *ISO 18014* in *ANSI ASC X9.95 Standard*, vendar sta oba plačljiva.

2.6 Zapis časovnega žiga po standardu ASN.1

2.6.1 Splošno

ASN.1 je standard, ki se v kriptografiji uporablja za predstavitev različnih kriptografskih informacij[4] kot je npr. tip algoritma, parametri algoritma, dolžina ključa itd. Definira torej notacijo za opis podatkovnih struktur za predstavitev,

kodiranje, prenos in dekodiranje podatkov. Predstavitev podatkov je formalna notacija ki je natančna, nedvoumna in neodvisna od vrste računalniškega sistema. Ker je ASN.1 tako obsežen standard in določa tako pomembno stvar kot je zapis podatkov, ga uporabljajo praktično vsi ostali kriptografski standardi.

Pri branju ASN.1 zapisa je potrebno prepoznati naslednje štiri elemente:

- identifikator objektov (angl. Object Identifiers, OIDs),
- struktura modulov,
- tipe vrednosti,
- način kodiranja podatkov.

2.6.2 Identifikator objektov

OIDs služi za zapis unikatne predstavitve objekta ali modula ASN.1 v kriptografskem svetu. OIDs v standardu ASN.1 omogočajo gradnjo unikatnega prostora imen, v katerem lahko nastopajo tudi organizacije, ki lahko prostor imen dopolnjujejo v svojem domenskem prostoru. V takšnem kontekstu si je OIDs najlažje predstavljati kot pot skozi drevo[8].

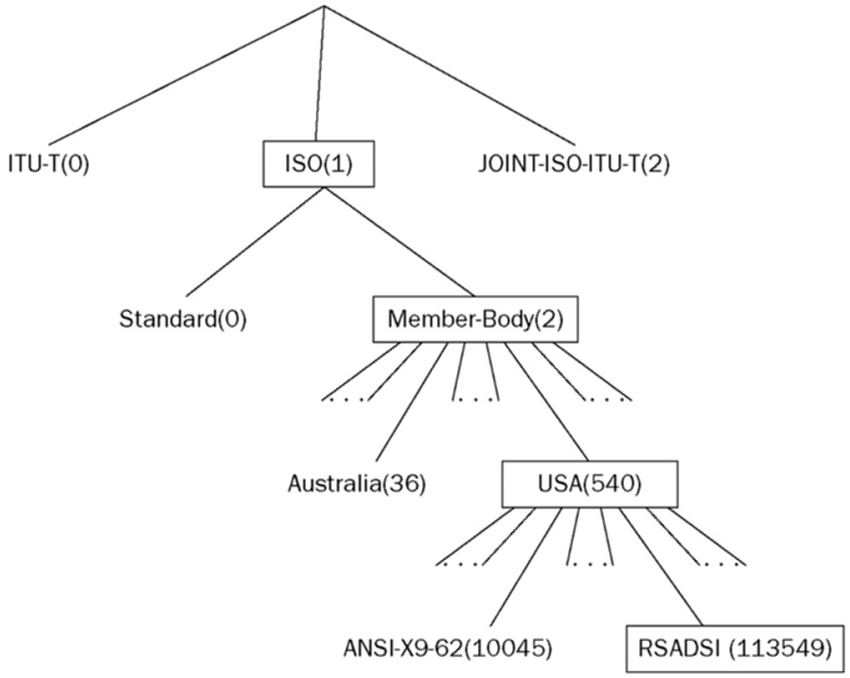
Primer OIDs: *iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 11 oz. 1.2.840.113549.1.1.11*, ki predstavlja funkcijo SHA dolgo 256 bajtov z RSA enkripcijo oz *SHA256withRSA*. *1.2.840.113549* predstavlja vejo *RSA Security*, ki jo je določila ISO v ZDA z naslovom *1.2.840*. Nadaljnje se *RSA Security* deli na *1.2.840.113549.1*, ki predstavlja PKCS standarde, ta se deli naprej na PKCS#1 *1.2.840.113549.1.1* in končno ta definira različne algoritme, kjer je *SHA256withRSA* en od njih. Razvidno je da OIDs naslov tvori le eno vejo skozi drevo, kot to prikazuje slika 2.7.

Znak pika v OIDs zapisu loči različne domene imen. Sam zapis je podoben kot ga uporabljamo že v drugih standardih, kot npr. za predstavitev naslovov IP, le da OIDs nimajo omejene dolžine znakov za eno domeno in niso omejeni po število domen za posamezen zapis.

Vrhovni domeni pripadata organizaciji ITU-T(0.x) in ISO(1.x) in skupna domena ISO/ITU-T(2.x). Ti dve organizaciji sta skupaj tudi standardizirali ASN.1.

Za iskanje različnih OIDs je na voljo več imenikov, ki so dostopni na internetu[2].

[2]OID Repository - http://www.oid-info.com/



Slika 2.7: Prikaz drevesne hierarhije OIDs

2.6.3 Struktura modulov

Modul v ASN.1 pomeni objekt ki nosi informacijo za uporabnika, podobno kot to stori element v XML formatu. Struktura modula je tipično:

```
ModuleName { ObjectIdentifier }
DEFINITIONS Tagging TAGS ::=
BEGIN
EXPORTS export_list;
IMPORTS import_list;
body
END
}
```

ModuleName in *ObjectIdentifier* predstavljata in določata ime modula. Tagging določa način binarnega zapisa modula, kjer je nabor vrednosti "IMPLICIT", "EXPLICIT" ali "AUTOMATIC"[3]. Simbol "::=" se prebere kot "je definiran kot". *Export_list* določa seznam vseh tipov, ki jih ta modul definira

[3]Za nadaljnjo razlago glej [8]

in jih lahko ostali ASN.1 moduli uvozijo in uporabijo. *Import_list* določa katere vse tipe uvozi modul in iz katerega modula. Body deklarira vse tipe in vsebino v modulu ter se konča z besedo END.

2.6.4 Tipi vrednosti

ASN.1 pozna več tipov vrednosti ki okvirno sodijo v tri grobe kategorije: preprosti tipi, niz znakov in strukturirani tipi. Preprosti tipi so tisti, ki jih poznamo že iz klasičnih programskih jezikov, kot so: števila, čas, ničnа vrednost (angl. null) in seznam vrednosti (angl. enumerated).

Nizi znakov se delijo na bitne nize in na nize črk. Nizi črk lahko predstavljajo množico podatkov vse od numeričnega niza pa do slikovnega niza, ki predstavlja sliko.

Sekvenca (angl. sequence) in množica (angl. set) sta tipa, ki opisujeta strukturo in sta verjetno najpomembnejša tipa v standardu ASN.1. Z njima lahko gradimo hierarhično strukturo vrednosti, ki predstavljajo nek tip, podobno kot nam to omogoča format XML.

Primer zapisa ASN.1[4]

2.6.5 Kodiranje podatkov

ASN.1 definira abstraktno sintakso informacij, ki je neodvisna od načina kodiranja podatkov. Vseeno pa standard predvideva več vrst kodiranja podatkov, ki so primerni za prenos podatkov. Najbolj uporabni dve vrsti kodiranja podatkov sta predvsem:

- DER (Distinguished Encoding Rules)
- XER (XML Encoding Rules)

[4]Primer vzet iz http://en.wikipedia.org/wiki/Abstract$_{S}yntax_{N}otation_{O}ne$:

```
FooProtocol DEFINITIONS ::=
    FooQuestion ::= SEQUENCE {
        trackingNumber INTEGER,
        question       IA5String
    }
    FooAnswer ::= SEQUENCE {
        questionNumber INTEGER,
        answer         BOOLEAN
    }
```

Kodiranje DER je najpreprostejši in najpogostejši način kodiranja. Zapisan je v binarni človeku neberljivi obliki. Sestavljen je kot kombinacija "tip-dolžina-vrednost". Primeri zapisa v DER kodiranju:

```
30 -- tip SEQUENCE
13 -- dolžina v bajtih

02 -- tip INTEGER
01 -- dolžina v bajtih
05 -- vrednost

16 -- tip IA5String
06 -- dolžina v bajtih
48 65 6C 6C 6F 21 -- vrednost ("Hello!" v ASCII)
```

Kodiranje XER nima kakšnih posebnih pravil, saj temelji na že uveljavljeni tehnologiji XML. Zaradi načina zapisa XML pomeni da je to kodiranje bolj prijazno saj omogoča človeku berljiv zapis.

2.7 Standard RFC 3161

RFC 3161 iz leta 2001 je po devetih letih še vedno v stanju "Proposed standard", kar v organizaciji IETF pomeni prvi od treh nivojev, skozi katere mora preiti standard, da postane uradno sprejet. V praksi pa to ne pomeni veliko, saj se je RFC 3161 že dodobra prijel. Polni naslov standarda je "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", iz katerega je vidno, da se standard ukvarja predvsem s protokolom med uporabnikom in agencijo za časovno žigosanje.

Standard se ukvarja predvsem s shemo časovnega žigosanja ki jo omogoča tehnologija IJK, torej izkorišča predvsem lastnosti asimetričnih ključev z algoritmom RSA, kot je to predstavljeno v prejšnjih sekcijah diplomske naloge.

Zahteve za TSA[11]:

- Uporablja zaupanja vredni izvor časa.
- Vsakemu časovnemu žigu pripne trenuten zaupanja vredni čas.
- Vsakemu časovnemu žigu pripne unikatno število.
- Ko prejme zahtevo s pravilnimi parametri, vedno odgovori s časovnim žigom.

- Časovnemu žigu priloži OID, ki predstavlja politiko za žigosanje pod katero je časovni žig nastal.
- Da časovno žigosa le podatkovni izvleček datuma, katerega dobi z uporabo EZF, ki je v odgovoru predstavljena z OID.
- Da preveri katera funkcija je uporabljena kot EZF v zahtevi in če se ta še vedno smatra za zanesljivo ter ali se ujema dolžina podatkovnega izvlečka z uporabljeno EZF.
- Da ne preverja podatkovnega izvlečka drugače kot v prejšnji zahtevi.
- Da izdaja časovne žige le s ključem, ki je temu ekskluzivno namenjen in da se to odraža v certifikatu ključa.
- V odgovoru ne vrača identitete osebe, ki je časovni žig zahtevala.
- Da v odgovor doda vse dodatne informacije, ki jih je uporabnik zahteval.

2.7.1 Transakcija

Kot prvo sporočilo v tem mehanizmu, uporabnik pošlje v TSA zahtevo za časovno žigosanje. Po obdelavi zahteve TSA pošlje odgovor z časovnim žigom uporabniku. Po odgovoru uporabnik preveri status transakcije, ki je zapisan v odgovoru. Uporabnik preveri tudi podpis časovnega žiga, identifikacijo TSA s certifikatom, pravilnost uporabljene EZF za časovno žigosanje. Potrebno je preveriti pravočasnost odgovora, s tem da preveri čas v časovnem žigu z lokalnim časom ter ali se ujema naključno število ki ga je poslal uporabnik v zahtevi, s tistim številom v odgovoru. Če katerokoli preverjanje ne uspe, se časovni žig zavrne.

Ker se lahko zgodi tudi da je certifikat TSA preklican, je potrebno preveriti tudi veljavnost certifikata, preko ustreznega *certificate revocation list (CRL).*

2.7.2 Identifikacija TSA

TSA vsakemu odgovoru priloži certifikat, ki se uporablja ekskluzivno le za časovno žigosanje. TSA ima lahko več različnih certifikatov ki se lahko uporabljajo za različne politike, različne algoritme, velikost ključev ali pa le za optimizacijo. Certifikat ki se uporablja za časovno žigosanje mora imeti v svojih

atributih lastnost, ki določa da se certifikat uporablja izključno za časovno žigosanje [5].

2.7.3 Format zahteve

RFC 3161 predvideva zahtevo v standardu ASN.1, ki ima naslednjo obliko[11]:

```
TimeStampReq ::= SEQUENCE  {
    version              INTEGER  { v1(1) },
    messageImprint       MessageImprint,
    reqPolicy            TSAPolicyId              OPTIONAL,
    nonce                INTEGER                  OPTIONAL,
    certReq              BOOLEAN                  DEFAULT FALSE,
    extensions           [0] IMPLICIT Extensions  OPTIONAL  }
```

Polje *version* določa verzijo zahteve po RFC 3161, ki ima ob času pisanja diplomskega dela vrednost 1.

Polje *messageImprint* nosi podatkovni izvleček, ki ga želimo časovno žigosati in OID funkcije EZF, s katero je podatkovni izvleček nastal. Če TSA ne prepozna uporabljene EZF ali če jo smatra za zastarelo, potem bo TSA v odgovoru javil napako in napisal obrazložitev. Opis polja v predstavitvi ASN.1:

```
MessageImprint ::= SEQUENCE  {
    hashAlgorithm       AlgorithmIdentifier,
    hashedMessage       OCTET STRING  }
```

Polje *reqPolicy* označuje pod katero politiko naj se izvede časovno žigosanje. Te politike so znane uporabniku vnaprej in se razlikujejo za vsako TSA.

Polje *nonce* je poljubna vrednost, ki jo določi uporabnik in jo TSA prepiše v odgovor. S tem lahko uporabnik identificira kateri odgovor pripada kateri zahtevi.

Polje *certReq* sporoči TSA, da je potrebno odgovoru priložiti certifikat s katerim je bil časovni žig ustvarjen.

Polje *extension* vsebuje vsa dodatna polja, določena v RFC 2459, ki jih TSA lahko ali pa ne podpira. Če TSA dodatno polje ne podpira potem bo v odgovoru vrnjen status z napako in obrazložitvijo.

Format zahteve je torej zelo enostaven, še posebej če upoštevamo, da so vsa polja z izjemo *version* in *messageImprint* poljubna in imajo v standardu privzete vrednosti, ki jih večini uporabnikov ni potrebno vedeti.

[5] *ExtKeyUsageSyntax*[12], ki ima vrednost OID: *iso(1) identified-organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) kp (3) timestamping (8)*[11]

2.7.4 Format odgovora

Odgovor TSA je, kot bomo videli, malo bolj zapleten kot zahteva, saj mora TSA poleg samega časovnega žiga javiti tudi različne napake in statuse, ki se lahko pri časovnem žigosanju pojavijo. Te napake in statusi so predvideni v standardu RFC. Predstavljena bosta torej odgovora ob pravilnem delovanju, ki vsebuje časovni žig in odgovor, ki javi napako ter opis te napake. Splošno lahko odgovor predstavimo z:

```
TimeStampResp ::= SEQUENCE  {
    status          PKIStatusInfo,
    timeStampToken  TimeStampToken  OPTIONAL  }

PKIStatusInfo ::= SEQUENCE {
    status          PKIStatus,
    statusString    PKIFreeText     OPTIONAL,
    failInfo        PKIFailureInfo  OPTIONAL  }
```

Polje status, ki je tipa PKIStatusInfo in je definiran v standardu RFC 2510[13], vsebuje polja, ki uporabniku sporočijo ali je prišlo med izvajanjem do napake in kakšna napaka je to bila.

Odgovor ob pravilnem delovanju

Ob pravilnem delovanju je TSA zavezana da v odgovoru prilepi časovni žig ki ga imenujemo *timeStampToken*. Odgovor predstavimo kot:

```
TimeStampResp ::= SEQUENCE  {
    status          PKIStatusInfo,
    timeStampToken  ContentInfo  }
```

Polje status v *PKIStatusInfo* vsebuje ob pravilnem delovanju vrednost 0 ali 1, kjer 1 pomeni, da je časovno žigosanje uspelo, vendar z manjšimi spremembami v zahtevi.

Polje *timeStampToken* je definiran kot *ContentInfo* oz. *Cryptographic Message Syntax (CMS)*[14], ki je definiran v RFC 2630 in je standardni način zapisa digitalnih podpisanih podatkov. RFC 3161 ga določa kot:

```
ContentInfo ::= SEQUENCE {
    contentType ContentType { id-signedData },
    content SignedData }
```

```
SignedData ::= SEQUENCE {
    version CMSVersion,
    digestAlgorithms DigestAlgorithmIdentifiers,
    encapContentInfo EncapsulatedContentInfo,
    certificates [0] IMPLICIT CertificateSet OPTIONAL,
    crls [1] IMPLICIT CertificateRevocationLists OPTIONAL,
    signerInfos SignerInfos }

EncapsulatedContentInfo ::= SEQUENCE {
    eContentType ContentType { id-ct-TSTInfo },
    eContent TSTInfo }

TSTInfo ::= SEQUENCE  {
   version                      INTEGER  { v1(1) },
   policy                       TSAPolicyId,
   messageImprint               MessageImprint,
   serialNumber                 INTEGER,
   genTime                      GeneralizedTime,
   accuracy                     Accuracy                 OPTIONAL,
   ordering                     BOOLEAN              DEFAULT FALSE,
   nonce                        INTEGER                  OPTIONAL,
   tsa                          [0] GeneralName          OPTIONAL,
   extensions                   [1] IMPLICIT Extensions   OPTIONAL  }
```

Tipi *ContentInfo*, *SignedData* in *EncapsulatedContentInfo* so vsi določeni v standardu RFC 2630[14] in se uporabljajo za generičen zapis podpisanega podatka. Tip *TSTInfo*, ki ga določa standard RFC 3161[11], pa je specifičen za TSA in si zasluži nadaljnjo razlago.

Polje *version* ima vedno vrednost 1. Polje *policy* nosi OID politike pod katero je TSA izvedla časovno žigosanje, če je bila v zahtevi podana politika potem ima to polje seveda isto vrednost kot v zahtevi. Polje *messageImprint* je kopiran iz istoimenskega polja v zahtevi. Polje *serialNumber* ima v standardu pomembno vlogo, saj to polje unikatno predstavlja vsak časovni žig ki ga je TSA kadarkoli izdala. To mora veljati tudi v primeru prekinitev izvajanja TSA ali sistemskih napak. Polje *genTime* nosi čas ko je bil časovni žig izdan. Polje *accuracy* je predvidena napaka polja *genTime*, časovna deviacija zapisana kot:

```
Accuracy ::= SEQUENCE {
    seconds        INTEGER              OPTIONAL,
    millis     [0] INTEGER  (1..999)    OPTIONAL,
```

```
    micros     [1] INTEGER  (1..999)    OPTIONAL  }
```

Kot vidimo, so vsa polja poljubna zato nekritičnim TSA ni potrebno računati predvidene deviacije napake. Z deviacijo lahko uporabnik izračuna tudi zgornjo in spodnjo limito *genTime*. Če je polje *ordering* enak *true*, to pomeni, da se lahko uporabnik zanese na natančnost polja *genTime* do te mere, da sme urejati časovne žige glede na polje *genTime*, saj je natančnost izvora časa v TSA dovolj velika. Polje *nonce* je kopija istoimenskega polja v zahtevi. Polje *tsa* nima pomembnega pomena, služi lahko kot namig o identiteti TSA ki je odgovorila z časovnim žigom. Na to polje se uporabnik absolutno ne sme zanesti pri identifikaciji TSA. Polje *extensions* nosi dodatna polja, ki so definirana v RFC 2459.

Odgovor ob napaki

Ob nepravilnem delovanju TSA ne vrne časovnega žiga ampak le status, ki uporabniku nudi informacije zakaj časovno žigosanje ni uspelo. V ASN.1 predstavimo z:

```
TimeStampResp ::= SEQUENCE  {
    status          PKIStatusInfo }
```

Kjer ima status v *PKIStatusInfo* vrednost od 2 do 5. Vrednosti *failInfo* zavzemajo številčne vrednosti, ki so opisane v standardu. Polje *statusString* pa nosi besedni opis napake in je lahko poljuben.

Iz polja *failInfo* je uporabniku tudi razvidno zakaj je prišlo do napake. Standard jih definira kot:

```
PKIFailureInfo ::= BIT STRING {
   badAlg               (0),
     -- neprepoznan ali nepodprt OID funkcije EZF
   badRequest           (2),
     -- transakcija ni dovoljena ali podprta
   badDataFormat        (5),
     -- zahteva ni pravilno formatirana
   timeNotAvailable    (14),
     -- TSA trenutno nima na voljo izvor časa
   unacceptedPolicy    (15),
     -- zahtevana politika ni na voljo
   unacceptedExtension (16),
     -- zahtevana dodatna polja niso podprta
```

```
addInfoNotAvailable (17)
 -- dodatne informacije v zahtevi niso podprte
 -- ali niso na voljo
systemFailure       (25)
  -- žigosanje ni uspelo zaradi sistemske napake  }
```

Način prenosa

Standard RFC 3161 ne določa specifičnega načina prenosa zahtev in odgovorov, vendar pa vseeno namigne nekaj možnosti in podrobneje razloži njihovo delovanje. Omenimo le, da lahko uporabnik komunicira s TSA preko elektronske pošte, preko protokola TCP/IP, preko protokola HTTP ali pa TSA nudi svoje storitve kot spletna storitev. V praksi se največkrat uporabljata zadnji dve možnosti, implementirani sta tudi v sistemu TSA, ki je bil razvit v okviru te diplomske naloge.

Pri uporabi protokola HTTP standard določa, da uporabimo meta podatek *Content-Type: application/timestamp-query* pri zahtevi in pri odgovoru *Content-Type: application/timestamp-reply*.

Poglavje 3

Agencija za časovno žigosanje

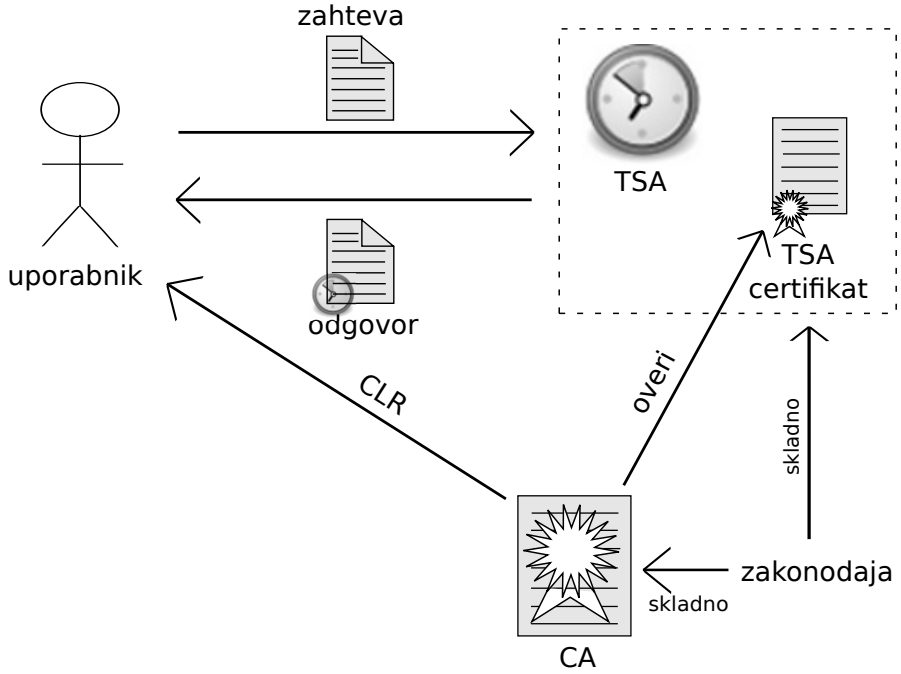
3.1 Entitete pri časovnem žigosanju

Pri časovnem žigosanju nastopa več entitet, kjer je TSA le ena izmed njih. V ospredju je vedno uporabnik ki ima željo svoje sporočilo časovno žigosati.

Slika 3.1 prikazuje shemo povezanosti in odvisnosti nastopajočih entitet med seboj. Glavna akterja sta uporabnik in TSA, ki si med seboj izmenjujeta zahteve in odgovore oz. sporočila in časovne žige. Zaupanje ustvarja certifikat TSA, ki ga je overila CA. Dokler je ta certifikat zaupanja vreden ga uporabnik ne bo našel v seznamu CRL. Ko pa certifikatu poteče veljavnost ali pa če sumimo da je kompromitiran, ga CA prekliče in doda v svoj CRL. Tu nastane problem v primeru ko napadalec ukrade certifikat in ta preklic ni takojšen, saj to omogoči napadalcu da ustvari škodo. Entiteti CA in TSA morata biti tudi skladni z zakonodajo, saj ta določa stopnjo zaupanja v storitev in v overjen certifikat.

TSA je sistem, ki ga sestavlja več modulov:

- modul za sprejem sporočil in za oddajo časovnih žigov na transportnem nivoju;
- modul za obdelavo formata podatkov (npr. DER ali XML kodiranje);
- modul za časovno žigosanje;
- modul za digitalno podpisovanje;
- modul za upravljanje s politikami časovnega žigosanja in ostalimi nastavitvami;



Slika 3.1: Prikaz nastopajočih entitet pri časovnem žigosanju

- modul za sledenje izvajanju (angl. logging);
- modul za usklajevanje trenutnega časa.

Predvsem pa je na TSA potrebno gledati širše kot le na programsko opremo.

3.2 Varnost in zaupanje

Standard RFC 3161 opisuje tudi nekaj varnostnih izzivov, ki se tičejo zaupanja v TSA in v njeno storitev časovnega žigosanja.

1. Ko TSA preneha z delovanjem in če njen privatni ključ ni bil ogrožen, potem ga CA ki je overil TSA, prekliče. CA bo certifikat dodala v CRL in v polje *reasonCode* ustrezno vrednost[1]. V tem primeru bo vsak časovni žig v prihodnosti neveljaven, že izdani časovni žigi pa ostanejo veljavni. V primeru da polje *reasonCode* nima vrednosti, potem se vsi časovni žigi ki jih je TSA kadarkoli izdala, smatrajo za neveljavne.

[1] *unspecified, affiliationChanged, superseded, cessationOfOperation*

2. V primeru ko je privatni ključ TSA ogrožen, se TSA certifikat prekliče in doda v CRL. Polje *reasonCode* v CRL se ne nastavi, oz. če se nastavi dobi vrednost *keyCompromise*. V tem primeru uporabniki storitve TSA ne morejo več zaupati izdanim časovnim žigom, zato vsi časovni žigi ki so bili kadarkoli izdani v preteklosti postanejo neveljavni. Prav zaradi tega je izrednega pomena, da TSA skrbno čuva svoj privatni ključ, kar se največkrat rešuje z uporabo HSM enot.

3. Ključ ki ga TSA uporablja za podpisovanje, mora biti dovolj dolg tako da omogoči zadostno časovno uporabo. Tudi če to upoštevamo, bo vsak ključ imel končno dolgo življenje, zaradi napredka v tehnologiji kriptografije in hitrosti računalniški sistemov[9, s. 69]. Zato je zaželjeno da vsak izdan časovni žig žigosamo ponovno v prihodnosti in s tem obnovimo zaupanje v časovni žig, ki nam ga nudi TSA.

4. Uporabnik, ki nima lokalnega izvora časa in uporablja le polje *nonce*, mora skrbeti za časovni interval v katerem je pripravljen čakati na odgovor TSA. Napad *man-in-the-middle* lahko povzroči zakasnitve v odgovoru, zato je sumljiv vsak odgovor ki zakasni več kot je dovoljeno. Kako dolga je lahko zakasnitev odgovora je odvisno od načina prenosa, ki ga uporablja TSA in od okolja v katerem se ta prenos vrši.

5. Če več uporabnikov izvede časovno žigosanje nad istim sporočilom, ali če en uporabnik večkrat časovno žigosa isto sporočilo, potem bo imel časovni žig isto polje za podatkovni izvleček[2]. Posledica tega je, da lahko možni prisluškovalec ugotovi, da se časovni žigi nanašajo na isto sporočilo.

6. Vedno se lahko namenoma ali nenamenoma zgodi, da uporabnik prejme več odgovorov na isto zahtevo. Nenamenoma se to lahko zgodi, ko pride do napak na transportnem nivoju in TSA prejme več zahtev, ki so duplikati. Namenoma pa se to zgodi, kadar napadalec odgovarja na zahtevo istočasno, ko na zahtevo odgovori veljavni TSA. Da uporabnik ugotovi, da je do takšne situacije prišlo, lahko vedno uporabi polje *nonce*, s katerim lahko enolično določi vsako poslano sporočilo in prejet odgovor, zato je uporaba tega polja skrajno priporočena. Druga možnost je, da si uporabnik v določenem časovnem intervalu zapomni vsa poslana sporočila in vse prejete odgovore, s katerim si lahko zago-

[2]Polje *messageImprint* v odgovoru.

tovi, da je odgovor prišel z določeno zakasnitvijo in da je odgovor znotraj zakasnitve prišel le enkrat.

3.3 Zakonodaja

TSA ki deluje na določenem ozemlju, mora slediti zakonodaji ki velja na tistem ozemlju, če želi, da ima njena storitev časovnega žigosanja in njeni izdani časovni žigi veljavo v morebitnih sporih na sodišču. V nasprotnem primeru se bo po vsej verjetnosti v morebitnem sporu izkazalo, da časovni žig nima zadostne teže kot dokaz da so bili določeni podatki časovno žigosani, kot to skuša dokazati časovni žig ki ni v skladu z zakonodajo.

3.3.1 Slovenija

V Sloveniji določata delovanje TSA predvsem dva zakona:

- Zakon o elektronskem poslovanju in elektronskem podpisu (ZEPEP-UPB1)[16].
- Uredba o pogojih za elektronsko poslovanje in elektronsko podpisovanje[17].

Zakona obravnavata elektronsko poslovanje na splošno, s tem pa določata tudi določbe ki se tičejo vsake TSA.

Uredba o pogojih za elektronsko poslovanje in elektronsko podpisovanje

Uredba predpisuje naslednje pomembnejše določbe:

- Strojna in programska oprema ter postopki mora izpolnjevati merila, standarde in pogoje, ki so splošno priznani v EU (uporaba HSM, ki zadoščajo standardu FIPS 140-1[3]).
- Overitelj mora opravljati redne varnostne preglede svoje infrastrukture ter preverjati, da ni prišlo do kakšni poskusov vdora nepooblaščenih oseb.
- Overitelj mora zagotavljati varno shranjevanje najmanj dveh varnostnih kopij podatkov.
- Overitelj mora voditi dnevnik v pisni obliki o vseh posegih v infrastrukturo in hraniti dnevnik najmanj 5 let.

[3]Dostopen na http://csrc.nist.gov/publications/fips/fips1401.htm

- Overiteljeva informacijsko komunikacijska infrastruktura mora biti primerno varovana (požarna pregrada, sistem za odkrivanje in preprečevanje vdorov in podobno).
- Overitelj mora zaposlovati najmanj tri osebe z univerzitetno izobrazbo (dva tehnične ali naravoslovne smeri), poleg tega pa mora zaposlovati ali pa imeti sklenjeno svetovalno pogodbo z univerzitetnim diplomiranim pravnikom.
- Overitelj mora imeti notranja pravila, ki imajo javni in zaupni del. Javni del mora biti dostopen na internetu ali v drugih medijih.
- Časovna veljavnost kvalificiranega potrdila je največ pet let.
- Kdor hrani podpisane podatke mora pred iztekom veljavnosti podatkov ali pa pred iztekom veljavnosti kvalificiranega potrdila ponovno obnoviti podpis s strani oseb, ki so podatke podpisale prvič ali pa s strani notarja.
- Varni časovni žig mora vsebovati nedvoumne in pravilne podatke o datumu in točnem času najmanj na sekundno natančno.
- Varen časovni žig je lahko dokumentu dodan ali priložen in z njim povezan, vendar morajo biti pri tem vedno izpolnjene enake zahteve kot za varen elektronski podpis s kvalificiranim potrdilom.
- Overitelj, ki izdaja varne časovne žige, mora uporabljati informacijski sistem, ki je sinhroniziran z izvorom točnega časa.

Za TSA so najpomembnejše zadnje tri določbe, ki se nanašajo nanašajo na časovno žigosanje.

ZEPEP

Zakon obravnava nekatere določbe, ki se pojavijo že v zgornji uredbi, določa pa tudi naslednje pomembnejše določbe:

- Varen elektronski podpis je podpis, ki je povezan izključno z uporabnikom; da je iz njega mogoče zanesljivo ugotoviti podpisnika; da je ustvarjen s sredstvi za varno elektronsko podpisovanje, ki so izključno pod podpisnikovim nadzorom; da je povezan z podatki, na katere se nanaša, tako da je opazna vsaka kasnejša sprememba.

- Podatkom se ne sme odreči veljavnosti ali dokazne vrednosti samo zato, ker so v elektronski obliki.
- Elektronska oblika podatkov ni enakovredna pisni, če se podatki nanašajo na pravni posel, s katerim se prenaša lastninska pravica na nepremičnini.
- Overitelj mora začetek opravljanja dejavnosti prijaviti ministrstvu, pristojnemu za informacijsko družbo in mu poslati svoja notranja pravila glede elektronskega podpisovanja. Ministrstvo mora nemudoma preklicati potrdila overitelja, če ta preneha z delovanjem.
- Kvalificirana potrdila overiteljev s sedežem v tujih državah so enakovredna domačim kvalificiranim potrdilom.
- Časovni žig je elektronsko podpisano potrdilo overitelja, ki potrjuje vsebino podatkov, na katere se nanaša, v navedenem času; varni časovni žig pa elektronsko podpisano potrdilo overitelja, ki izpolnjuje pogoje kot varni elektronski podpis.

3.3.2 Tujina

V tujini najdemo zakone, ki so podobni slovenskim. V večjih državah oz. unijah z močnejšo zakonodajo so zakoni celo podrobnejši in predpisujejo tudi potrebno in dovoljeno tehnologijo za izvajanje kriptografskih procesov. V EU je predpisana direktiva *Electronic Signature Directive (1999/93/EC)*, ki opisuje način uporabe elektronskih podpisov. V ZDA je zakonodaja še podrobnejša in je določana z zakoni *Electronic Signatures in Global and National Commerce Act*, *Uniform Electronic Transactions Act*, *Digital Signature And Electronic Authentication Law*. Poleg tega pa je v ZDA predpisano, da morajo federalne agencije uporabljati elektronsko poslovanje, kjer je to le mogoče[4].

3.4 Politika časovnega žigosanja

Vsaka TSA lahko izbere politiko (angl. policy, definicija: ravnanje posameznika z ljudmi, ustrezno okoliščinam[15]) pod katero želi nuditi svoje storitve. Politiko lahko tipično razumemo tudi kot principe in pravila, ki vodijo odločitve tako, da dosežemo racionalne oz. želene rezultate.

Tipično lahko politika pod katero deluje TSA določa naslednje lastnosti:

[4]Government Paperwork Elimination Act

- dolžina privatnega ključa,
- dolžina življenjskega cikla ključa,
- način hranjenja privatnega ključa,
- sledljivost z revizijskimi sledmi,
- anonimnost,
- shranjevanje časovnih žigov v skladišče ali podatkovno bazo,
- plačilo,
- osveževaje časovnega žiga,
- izvor časa.

TSA mora politiki dodeliti tudi unikaten OID, ki ga uporabnik navede v zahtevi za časovno žigosanje in s tem izbere politiko pod katero želi storitev TSA. To naredi tako, da zaprosi za lasten OID organizacijo IANA (angl. Internet Assigned Numbers Authority)[5], kateri pripada OID *1.3.6.1.4.1.* Organizacija, ki je zaprosila IANA za lasten OID, nato dobi OID v obliki *1.3.6.1.4.1.XXXXX*, ki ga lahko veji naprej po lastni presoji.

Politika časovnega žigosanja je navadno formalen dokument, ki opisuje vse lastnosti izdanega časovnega ključa, certifikata, ki se uporabi za časovno žigosanje in notranja pravila TSA, ki opisujejo na kakšen način TSA zagotavlja varnost časovnega žiga.

RFC 3628 je dokument, ki ureja področje politike časovnega žigosanja. Dokument opisuje tudi najboljše prakse notranjih procesov TSA in lahko služi kot iztočnica za pisanje politike časovnega žigosanja. Opisuje namreč tudi procese kot je varno kreiranje ključev za podpisovanje, zato so razložene najboljše prakse uporabne tudi za druge storitve elektronskega poslovanja in ne samo za pisanje politike TSA.

[5]Obrazec za prošnjo je na voljo na http://pen.iana.org/pen/PenApplication.page

Poglavje 4

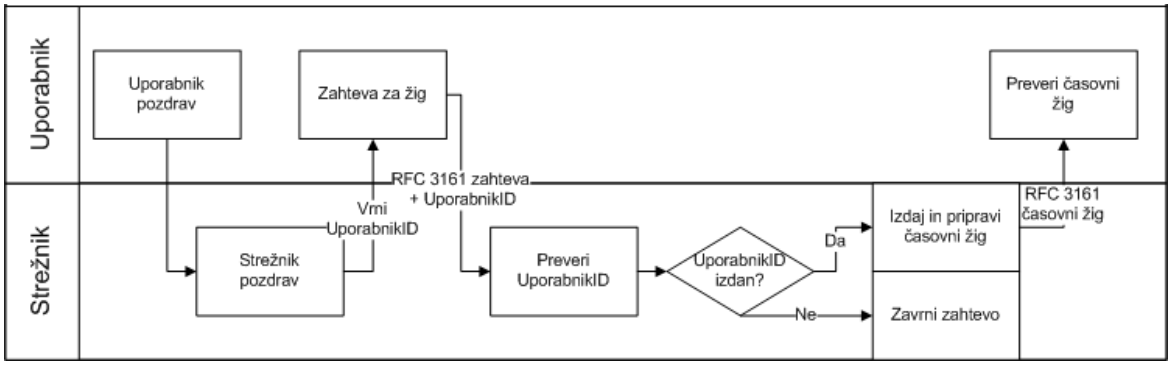
Razširitve standardov

4.1 Obramba pred napadom *denial of service*

Denial of service (DoS) je napad, ki ga izvedejo napadalci z namenom da ostalim uporabnikom onemogočijo dostop do napadene storitve. Napad je postal popularen s pojavom interneta, saj napadalcem omogoča, da izkoristijo tuje računalnike do katerih imajo dostop preko škodljivih programov ali zaradi varnostnih lukenj v operacijskih sistemih. Poleg tega pa ponudniki različnih storitev za komunikacijski kanal uporabljajo internet, kar napadalcem omogoča direktni napad, brez velikega denarnega vložka, saj za napad uporabijo tuje računalnike. To dejstvo otežuje tudi iskanje napadalcev, saj je napadalec ponavadi skrit in njegov računalnik ne udejstvuje v samem DoS napadu.

4.1.1 Vpliv na TSA

Standard RFC 3161 ne obravnava zaščite pred napadom DoS. Problem nastane, ker je izdajanje časovnega žiga izjemno potratna in računsko zahtevna operacija. Če se podpisovanje izvaja na osrednjem CPU in ne na enoti HSM, lahko na modernem PC računalniku podpisovanje z RSA 1024 bit dolgim ključem traja tudi dve sekundi. Nasprotno lahko zmogljivejši HSM-ji podpišejo tudi do 10.000 zahtev na sekundo, kar je za kar nekaj faktorjev hitreje. Vendar tudi to ni dovolj, saj lahko napadalci ustvarijo več milijonov zahtevkov za časovno žigosanje v sekundi. V nasprotju z nekaterimi drugimi storitvami na internetu, pri TSA ni problem sama zmogljivost povezave na internet, saj bo napadalec dosegel svoj namen že z nekaj tisoč zahtevami za časovno žigosanje in s tem povzročil preobremenitev strežnika TSA.



Slika 4.1: Preprost prikaz *handshaking* protokola za razširitev RFC 3161.

4.1.2 Razširitev RFC 3161

Kot že ugotovljeno, lahko nivo obrambe pred DoS napadi dvignemo tako, da ustvarimo prepreko, ki bo napadalcem preprečila, da pošiljajo zahteve na katere bo TSA takoj začela izdajati časovne žige. To lahko storimo tako, da protokol v RFC 3161 razširimo kot prikazuje slika 4.1.

Možna razširitev ni popolna obramba pred napadom DoS. Obramba izkorišča dejstvo, da je v sistemu TSA najpočasnejša operacija izdaja časovnega žiga in njegovo podpisovanje, zato napadalcu oteži dostop do te operacije. Sedaj do časovnega žigosanje pride le, če uporabnik pozna številko, ki mu jo dodeli strežnik TSA (polje *UporabnikID*). Šele ko uporabnik zahtevi priloži dodeljeno številko, mu strežnik izda časovni žig.

Obramba je podobna shemi seje pri spletnih aplikacijah, kjer se vsaka zahteva identificira s številko seje, ki jo je strežnik dodelil uporabniku.

Ko ima uporabnik dodeljen *UporabnikID* lahko znotraj ene seje zahteva več časovnih žigov. Tako lahko tudi TSA nadzoruje maksimalno število izdanih časovnih žigov znotraj ene seje. V primeru, ko uporabnik v določenem času ne zahteva novega časovnega žiga, mu seja poteče in *UporabnikID* se sprosti za nadaljnjo uporabo.

Primer obrambe:

- Če nek uporabnik zahteva v eni seji preveliko število časovnih žigov, mu lahko TSA ukine sejo.

- Če se v kratkem časovnem obdobju pojavi veliko zahtev za vzpostavitev seje, lahko TSA implicira, da je bil proti njej sprožen DoS napad. Zato lahko TSA selektivno izbira katere seje želi vzpostaviti po različnih parametrih (npr. pozna IP naslov legitimnega uporabnika; glede na izbrano poli-

tiko, kjer imajo nekatere višjo prioriteto in niso javno objavljene). S takim mehanizmom se lahko TSA obrani DoS napada, vendar bo verjetno med zavrnjenimi sejami tudi nekaj takšnih, ki so jih zahtevali legitimni uporabniki.

4.2 Časovni žig v formatu XML

V poglavju 2.6 smo obravnavali način zapisa časovnega žiga v ASN.1 standardu ki določa različna kodiranja podatkov. Standard predvideva tudi *XER* kodiranje ki temelji na tehnologiji XML, vendar zaradi narave standarda ASN.1 ki je precej abstrakten standard, ta ne določa točne strukture zapisa. Zato tudi ni mogoče sam časovni žig pripeti v že obstoječo XML datoteko, kot to lahko to naredimo z XML digitalnim podpisom, ki je določen s standardom *xmldsig*[1] konzorcija W3C. Ko se digitalni podpis pripne v XML datoteko, skupaj tvorita nerazdružljivo celoto. Prav tako pa RFC 3161 striktno določa kodiranje nekaterih vrednosti v DER načinu.

4.2.1 XML struktura časovnega žiga

Kot omenjeno, je najpreprosteje zgraditi časovni žig XML tako, da uporabimo kodiranje XER standarda ASN.1 tam kjer je to mogoče. Drugje, kot npr. zapis certifikata pa pustimo v formatu DER in ga le ovijemo v element XML. Primer časovnega žiga zapisanega v XML formatu[2]:

```
<ns1:TimeStampResponseXER xmlns:ns1="protocol.tsa.fri">
    <StatusInfo>
        <status>0</status>
        <StatusText>Operation Okay</StatusText>
    </StatusInfo>
    <ns1:TimeStampToken>
        <ContentType>1.2.840.113549.1.2</ContentType>
        <Content>
            <Version>3</Version>
            <DigestAlgorithms>1.3.14.3.2.26</DigestAlgorithms>
            <EncapContentInfo>
                <EContentType>1.2.840.113549.1.9.16.1.4</EContentType>
```

[1] Na voljo na http://www.w3.org/TR/xmldsig-core/

[2] Shema, ki določa XML strukturo časovnega žiga je priložena diplomski nalogi v digitalni obliki.

```
                <ns1:TSTInfo>
                    <ns1:Version>1</ns1:Version>
                    <ns1:MessageImprint>
                        <hashAlgorithm>1.3.14.3.2.26</hashAlgorithm>
                        <hashedMessage>aaf4c61ddcc5e8a2dabede0f3b482cd9aea9434d</has
                    </ns1:MessageImprint>
                    <SerialNumber>32</SerialNumber>
                    <GenTime>20100822224740Z</GenTime>
                </ns1:TSTInfo>
            </EncapContentInfo>
        </Content>
    </ns1:TimeStampToken>
</ns1:TimeStampResponseXER>
```

Poglavje 5

Zaključek

Časovno žigosanje je že ustaljena tehnologija, ki se uporablja predvsem v poslovnem svetu. Skupaj z elektronskim podpisom tvorita tehnološko podlago za varno elektronsko poslovanje z visoko stopnjo zaupanja. Poleg same tehnologije časovnega žigosanja pa je za varno poslovanje potrebna tudi zakonodaja, ki ureja različne aspekte elektronskega poslovanja.

V diplomski nalogi je predstavljen standard RFC 3161, ki je najbolj uporabljan standard na področju časovnega žigosanja. RFC 3161 skupaj z RFC 3628, ki opisuje najboljše prakse s področja TSA, tvorita celovito podlago, po kateri lahko vsakdo implementira zaupanja vredno TSA, ki bo tudi ustrezala zakonodaji v večini razvitega sveta. V nalogi je predstavljena tudi slovenska zakonodaja, ki ureja področje elektronskega poslovanja in s tem časovnega žigosanja.

V drugem poglavju smo analizirali tehnične rešitve, ki so trenutno najbolj uporabljene in nam omogočajo časovno žigosanje s tehnologijo IJK. Dotaknili smo se tudi protokola NTP, ki nam služi za sinhronizacijo lokalnega časa z virom točnega časa preko interneta. Predstavljen je tudi standard ASN.1, ki je obširen standard in zajema format zapisa različni kriptografskih podatkov, uporablja pa ga tudi standard RFC 3161.

V poglavju tri je predstavljena agencija za časovno žigosanje, ki je v osnovi TTP, ki ponuja storitev časovnega žigosanja. Kot ponudnik zaupanja vredne storitve je potrebno na storitev gledati širše kot le na skupek tehnologij, ki storitev omogočajo. Potrebno je misliti na zaupanje, varnost, zakonodajo, procese itd. Pomembna je tudi ugotovitev, da TSA za časovno žigosanje ne potrebuje celotnega dokumenta, ampak le njegov podatkovni izvleček. Tako lahko sama vsebina podatka, ki se časovno žigosa, ostane skrita.

V poglavju štiri sta predstavljene dve pomanjkljivosti in sicer, da je TSA

posebno občutljiva na napad DoS, ter da v trenutnih standardih nikjer ni definirana struktura zapisa časovnega žiga v XML formatu. Predlagani rešitvi sta preprosti in praktični.

Nazadnje je predstavljena realizacija TSA po standardu RFC 3161 v javanskem okolju. Za implementacijo je uporabljenih obilo odprto kodnih rešitev na področju kriptografije. Napisana je tudi primerna politika, ki sledi najboljšim praksam iz standarda 3628.

Dodatek A

FriTSA

A.1 Opis in funkcije

V sklopu diplomske naloge je bila razvita agencija za časovno žigosanje, ki je v skladu s standardom RFC 3161. Sistem je popolnoma avtomatiziran in od sistemskega administratorja ne potrebuje, z izjemo prvotnega zagona, nikakršnega posega ob izdaji časovnih žigov. Uporabnik lahko dostopa do storitve časovnega žigosanja preko spletne storitve ali preko protokola HTTP. V sklopu TSA deluje tudi spletna stran, kjer lahko uporabnik najde navodila za časovno žigosanje, politiko žigosanja in pa enostavni spletni vmesnik za časovno žigosanje, kjer lahko uporabnik preko grafičnega vmesnika izbere želeno datoteko, ki jo želi časovno žigosati. Za potrebe uporabnika je bil razvit tudi odjemalec za storitev časovnega žigosanja, ki se zažene na uporabnikovem računalniku, ter kot vhod sprejme datoteko, ki jo uporabnik želi časovno žigosati.

A.2 Uporabljene tehnologije

Za sam razvoj agencije za časovno žigosanje je bil uporabljen programski jezik Java. Za hitrejši razvoj je bilo uporabljenih nekaj odprtokodnih knjižnic, med katerimi izstopa knjižnica *BouncyCastle*[1], ki nudi ogromno kriptografskih operacij z IJK.

TSA je uporabniku dostopen preko protokola HTTP. Tu glavno vlogo igra programski strežnik HTTP, ki preko FastCGI vmesnika komunicira s programom, ki izvaja časovno žigosanje. FastCGI je protokol, ki je namenjen

[1]Dostopna na http://www.bouncycastle.org/

komuniciranju med programskim strežnikom in poljubnim programom, ki od strežnika prejme uporabnikovo zahtevo in nato nazaj posreduje odgovor, ki ga strežnik posreduje končnemu uporabniku[2]. Prednost protokola FastCGI je v tem, da se delovni program zažene le enkrat in se nato uporabi večkrat za različne zahteve. S tem dosežemo občutno hitrejše delovanje strežbe.

TSA je dostopen tudi kot spletna storitev preko protokola SOAP. Tu glavno vlogo igra aplikacijski strežnik, ki podpira Javo[3]. Za obdelavo sporočil SOAP je bila uporabljena javanska knjižnica *Axis2*, ki olajša razvoj spletnih storitev v javi.

Ker je ena od zahtev standarda RFC 3161, da mora TSA v časovnem žigu pripeti serijsko številko trenutnega odgovora, vsi vmesniki za TSA uporabljajo skupno kodo za časovno žigosanje, ki si tudi zapomni zadnjo izdano serijsko številko, ne glede na to preko katerega kanala je uporabnik sprožil zahtevo za časovno žigosanje. Prav tako je poenoteno sledenje izvajanju, tako da lahko sistemski administrator naenkrat sledi izvajanju tako programu FastCGI, kot spletni storitvi.

Uporabljeni certifikati v TSA so samo-podpisani in niso shranjeni v HSM napravi.

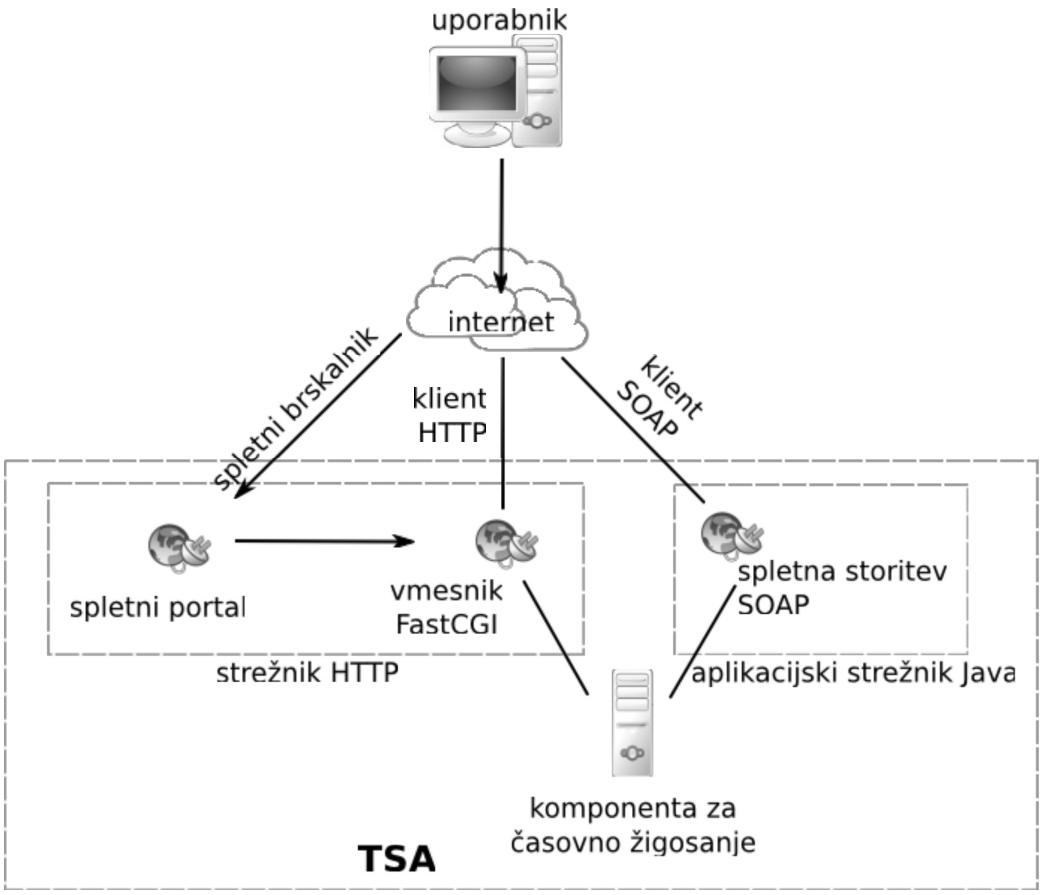
A.3 Shema sistema

Shema A.1 prikazuje poljudno konceptualno zasnovo sistema TSA. Uporabnik lahko do storitve dostopa preko treh različnih vmesnikov. Najpreprostejši je spletni portal, ki uporabniku ponuja zaslonsko masko s katero lahko izbere različne parametre in izbere datoteko, ki jo želi časovno žigosati. Portal ne izvede zahteve sam, temveč jo prekodira in pošlje zahtevo vmesniku FastCGI.

Vmesnika FastCGI in SOAP sta oba dosegljiva uporabniku direktno, za kar potrebuje klienta ki zna ustvariti ustrezno zahtevo, za vmesnik FastCGI zahtevo v DER formatu in za vmesnik SOAP zahtevo v XER formatu. S stališča TSA sta vmesnika t.i. *frontend*, ki ga naslovi uporabnik. *Backend* nudi komponento za časovno žigosanje, ki skrbi za preverjanje ustreznosti zahteve in izdajo časovnega žiga. Komponenta tudi skrbi za ustrezno serijsko številko odgovora in za enoten zapis sledi izvajanja.

[2]Lighttpd je strežnik HTTP, ki podpira FastCGI.

[3]Tomcat je aplikacijski strežnik s katerim lahko izvajamo spletno storitev



Slika A.1: Shema sistema TSA.

Slike

Literatura

[1] C. Adams and D. Pinkas, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)," avg. 2001.

[2] A. Buldas, H. Lipmaa, "Digital Signatures, Timestamping and the Corresponding Infrastructure," stran 8-9, 1998.

[3] S. Farrell, S. Santesson, D. Cooper, S. Boeyen, T. Polk, and R. Housley, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," maj. 2008.

[4] D. Hook, Beginning Cryptography with Java, Wrox, 2005.

[5] H. Massias, J.J. Quisquater, "Time and Cryptography," 1997.

[6] D.L. Mills, "Network Time Protocol (NTP)," 1985.

[7] K. Schmeh Cryptography and Public Key Infrastructure on the Internet, Wiley, 2003.

[8] B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, Second Edition, Wiley, 1996.

[9] D. Trcek, Managing Information Systems Security and Privacy, Springer, 2005.

[10] M. Une, The Security Evaluation of Time Stamping Schemes: The Present Situation and Studies, 2001.

[11] RFC 3161: X.509 Internet Public Key Infrastructure Time-Stamp Protocol (TSP), 2001. Na voljo: http://tools.ietf.org/html/rfc3161

[12] RFC 2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile, 1999. Na voljo: http://tools.ietf.org/html/rfc2459

[13] RFC 2510: Internet X.509 Public Key Infrastructe Certificate Management Protocols, 1999. Na voljo: http://tools.ietf.org/html/rfc2510

[14] RFC 2630: Cryptographic Message Syntax, 1999. Na voljo: http://tools.ietf.org/html/rfc2630

[15] SSKJ na internetu. Na voljo: http://bos.zrc-sazu.si/sskj.html

[16] Zakon o elektronskem poslovanju in elektronskem podpisu (ZEPEP-UPB1), Uradni list RS, št. 98/2004. Na voljo: http://www.uradni-list.si/1/content?id=51150

[17] Uredbo o pogojih za elektronsko poslovanje in elektronsko podpisovanje, Uradni list RS, št. 77/2000. Na voljo: http://www.uradni-list.si/1/content?id=27295

[18] Wikipedia: Cryptographic hash function. Na voljo: http://en.wikipedia.org/wiki/Cryptographic$_hash_function$