UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO

Matematika - uporabna smer

Mojca Mikac

EVIDENCA POŠTNIH PLAČIL V DIGITALNI DOBI

Diplomsko delo

Ljubljana, 2001

Povzetek

V tem delu bomo predstavili nov način evidentiranja plačila poštnine. V današnjem času se kot označba plačila uporablja predvsem poštna znamka. Odkar so se na trgu pojavili visoko zmogljivi tiskalniki, dostopni vsakomur, se je odpornost poštne znamke proti ponarejanju precej zmanjšala. Zato je nastala potreba po novem načinu evidentiranja plačila poštnine. Ta nov način predstavlja digitalna poštna znamka, ki jo bomo v tem delu podrobno predstavili. Spoznali bomo, iz katerih podatkov je sestavljena in raziskali način izdelave takih znamk. Ker bodo uporabljeni podatki precej zaupne narave, jih bomo zašifrirali. Da bi zagotovili pristnost uporabljenih podatkov, jih bomo podpisali z digitalnim podpisom. Tako se to delo ukvarja tudi s primernimi kriptografskimi protokoli. Uporabili bomo kriptografijo, ki temelji na eliptičnih krivuljah.

Ključne besede: digitalna poštna znamka, plačilo poštnine, digitalni podpis, eliptične krivulje.

Abstract

In this thesis a new way of evidencing postal payment is introduced. An example of such evidence of paid postage, which we use nowadays, is a post stamp. Ever since the inexpensive, high print quality computer printers appeared on the market, the resistance of post stamps against fraud was seriously damaged. So a new way of evidencing postal payment has to be introduced. The digital postal mark represents this new way. We will give a detailed description of such digital postal mark, learn all about the data which are taking part in such postal mark, and describe a way how to produce them. Because of the confidential nature of data used, we will encrypt them. Since we want to assure their authenticity, we will use a digital signature to sign them. Therefore we will also describe a corresponding cryptographic protocols. We will use cryptography based on elliptic curves.

Keywords: digital postal mark, postal revenue collection, digital signature, elliptic curves.

Math. subj. class(2001): 11T71, 11Y16, 11H52

Zahvaljujem se mojemu mentorju
doc. dr. Aleksandru Jurišiću za
vso pomoč pri pisanju tega dela.
Zahvaljujem se tudi mojim staršem,
ki mi stojijo ob strani.
Prav posebna zahvala pa gre mojemu
Bastianu za vso podporo in spodbudo.

PROGRAM DIPLOMSKEGA DELA

Digitalni podpisi in pošta

Delo naj predstavi matematične osnove, potrebne za študij ElGamalovih javnih kriptosistemov, s poudarkom na digitalnih podpisih in eliptičnih krivuljah. Glavni cilj je predstaviti osnovne kriptografske protokole, ki se uporabljajo za evidenco poštnih plačil v digitalni dobi. Izdela naj se tudi testna implementacija ustreznega protokola/kriptosistema v programskem jeziku C.

Literatura:

Douglas R. Stinson, Cryptography – Theory and Practice, CRC Press, 1995.

A. Jurišić, A. Menezes, "Elliptic Curves and Cryptography", *Dr. Dobb's Journal*, April 1997, 26-37.

L.A. Pintsov, S.A. Vanstone, Postal Revenue Collection in the Digital Age, Research Report CORR 2000-43, University of Waterloo (2000).

Information Based Indicia Program (IBIP) Performance Criteria for Information-Based Indicia And Security Architecture for Closed IBI Postage Metering Systems (PCIBI-C), draft, The United States Postal Service (USPS), January 12, 1999 (`http://56.0.78.92/html/programdoc.html`).

Information Based Indicia Program (IBIP) Performance Criteria for Information-Based Indicia And Security Architecture for Open IBI Postage Evidencing Systems (PCIBI-O), draft, The United States Postal Service (USPS), February 23, 2000. (`http://56.0.78.92/html/programdoc.html`).

Kazalo

Poglavje 1

UVOD

V tem delu bomo spregovorili predvsem o novem načinu evidentiranja plačila poštnine. Večina poštnih organizacij zahteva predplačilo za svoje storitve. To pomeni, da morajo biti vsi poštni kosi opremljeni s preverljivo označbo tega predplačila. Zgodovinsko prvi primer take označbe predstavlja poštna znamka. Običajna poštna znamka ima, kljub vsem dobrim lastnostim, tudi precej pomanjkljivosti: je predmet kraje, izdelava in distribucija sta dragi, ne nosi nobenih informacij o času in kraju pošiljanja, njena glavna pomanjkljivost pa je možnost ponarejanja. Da bi ponarejanje preprečili, so poštne znamke z leti spreminjale svoje fizične lastnosti: oblika in barva sta postajali kar se da zapleteni, izboljšala se je ločljivost tiska itd. Vendar pa vse to v današnjih časih, ko so se pojavili visoko zmogljivi tiskalniki, dostopni vsakomur, preprosto ne zadošča več. Zato je nastala potreba po drugačnem načinu evidentiranja poštnine. Ta drugačen način predstavlja digitalna poštna znamka. Taka znamka je v bistvu dvodimenzionalna črtna koda. To pomeni, da je njena fizična reprezentacija precej enostavnejša od današnje poštne znamke in bi morda zato lahko bila še bolj privlačna za vsakršno kopiranje. Da temu ne bi bilo tako, smo poskrbeli s tem, da smo vanjo vključili številne podatke. Taki podatki so recimo čas, kraj in celo oseba pošiljanja, nekatere varnostne identifikacijske številke, plačilo poštnine itd. Da ti podatki ne bi bili dostopni vsakomur, smo jih zašifrirali in tako dosegli potrebno stopnjo zaupnosti. V ta namen v diplomskem delu uporabimo kriptografske protokole. Kriptografija je veda o šifriranju podatkov z namenom zagotavljanja zaupnosti in pristnosti podatkov ali oseb. Primer kriptografskega protokola, ki ga bomo uporabili, je digitalni podpis. Tega bomo uporabili, da bi zagotovili resnično pristnost pošiljatelja oz. plačnika poštnine.

To diplomsko delo, četudi ima čisto praktične cilje, vključuje precejšen del kriptografije, ki je zgrajena na povsem matematičnih osnovah. Tako bomo v naslednjem poglavju najprej spoznali potrebne matematične prijeme in se seznanili z nekim računsko zelo težkim problemom, to je problemom diskretnega logaritma. V današnjih časih pomembno vejo kriptografije predstavlja kriptografija, ki temelji na eliptičnih krivuljah, ki se ji tudi v tem delu ne bomo mogli izogniti. Osnovne pojme eliptičnih krivulj bomo spoznali v drugem razdelku drugega poglavja. V tretjem poglavju pa bomo vso pozornost namenili čisto konkretnemu kriptografskemu protokolu, to je algoritmu digitalnega podpisa.

Najprej bomo spoznali, kaj digitalni podpis sploh je, potem bomo spregovorili o njegovih prednostih in slabostih v primerjavi z ročnim podpisom in končno podali tri različne algoritme za njegov izračun. Nato bomo primerjali te tri algoritme z vidika njihove varnosti in uporabnosti. V četrtem poglavju se bomo torej lahko lotili ideje digitalne poštne znamke. Ogledali si bomo njeno zgradbo, podali bomo način izdelave in čisto na koncu poglavja spregovorili še nekaj besed o varnosti take znamke oz. o možnosti ponarejanja. V zadnjem poglavju pa si bomo pogledali resničen primer implementacije digitalne poštne znamke. Ta implementacija bo predstavljala zgolj podlago za nadaljno izgradnjo sistema za izdelavo takih znamk in še zdaleč ne bo primerna za širšo uporabo. Vsebovala bo zgolj osnovne kriptografske algoritme.

Poglavje 2

MATEMATIČNE OSNOVE

V tem poglavju bomo spoznali vse matematične pojme, ki jih bomo potrebovali v nadaljevanju. Tako bomo najprej spoznali končni obseg $\mathbb{Z}_p$ in v njem definirali operaciji seštevanja in množenja po modulu praštevila p. Potem si bomo podrobno pogledali lastnosti grupe $\mathbb{Z}_p^*$, to je grupe vseh obrnljivih elementov obsega $\mathbb{Z}_p$. Nato bomo v tej grupi definirali nek zelo težek problem, ki ga bomo kasneje s pridom uporabili pri algoritmih digitalnega podpisa. To je problem diskretnega logaritma v grupi $\mathbb{Z}_p^*$. Izkazalo se bo, da lahko problem diskretnega logaritma definiramo v katerikoli končni grupi. V ta namen bomo najprej spoznali končni obseg $\mathbb{F}_{2^m}$, potem pa definirali grupo eliptične krivulje nad tem obsegom. Nato bomo problem diskretnega logaritma definirali v tej grupi. Izkazalo se bo, da je tako definirani problem še težji, hkrati pa zanj velja, da ga je precej lažje implementirati v računalniško okolje. Na koncu poglavja, v razdelku 2.3, bomo še formalno definirali, kdaj lahko za nek problem rečemo, da je računsko zahteven.

2.1 MULTIPLIKATIVNA GRUPA $\mathbb{Z}_p^*$

V tem razdelku bomo spregovorili o multiplikativni grupi števil $\mathbb{Z}_p^*$, kjer je p praštevilo. Pogledali si bomo osnovne lastnosti takih grup. Potem pa si bomo ogledali še problem diskretnega logaritma v tej grupi.

Množica $\mathbb{Z}_p$, kjer je p praštevilo, je množica števil $\{0, 1, 2, \ldots, p-1\}$. V to množico vpeljemo naslednji operaciji:

- **seštevanje po modulu p:** če sta $a, b \in \mathbb{Z}_p$, potem definiramo $a + b \,(\mathrm{mod} p)$, kot ostanek, ki ga dobimo pri deljenju vsote $a + b$ praštevilom p in
- **množenje po modulu p:** če sta $a, b \in \mathbb{Z}_p$, potem definiramo $a \cdot b \,(\mathrm{mod} p)$, kot ostanek, ki ga dobimo pri deljenju zmnožka $a \cdot b$ s praštevilom p.

Za tako definirani operaciji je množica $\mathbb{Z}_p$ končni obseg. Torej je množica

$$\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\},$$

ki jo sestavljajo vsi obrnljivi elementi obsega $\mathbb{Z}_p$, grupa za operacijo množenja po modulu p. Množica $\mathbb{Z}_p$ pa je grupa za operacijo seštevanja po modulu p.

Število d imenujemo **največji skupni delitelj** števil a in $b \in \mathbb{Z}$, če velja, da $d|a$ in $d|b$; za vsako število $c \in \mathbb{Z}$, ki deli a in b, pa velja, da $c|d$. Označimo ga $\gcd(a, b)$ in definiramo $\gcd(a, 0) = a$. Pravimo, da sta si števili a in $b \in \mathbb{Z}$ **tuji**, če velja $\gcd(a, b) = 1$.

Za $n \in \mathbb{N}$, $n \geq 1$ definiramo $\varphi(n) =$ število vseh celih števil z intervala $[1, n]$, ki so tuja številu n. Funkcijo $\varphi(n)$ imenujemo **Eulerjeva številska funkcija**.

Takoj lahko opazimo, da velja $\varphi(p) = p - 1$, če je p praštevilo.

Red elementa a končne grupe G je najmanjše pozitivno celo število r, za katero velja: $a^r = e$, kjer je e enota v grupi.

Poglejmo si nekaj lastnosti, ki veljajo za splošne končne grupe.

Izrek 2.1 (Lagrange) *Red elementa končne grupe vselej deli moč grupe.* ∎

Dokaz tega izreka si lahko pogledate v katerikoli knjigi o teoriji grup, recimo v knjigi [9], v kateri najdete tudi vse tu navedene trditve.

Pravimo, da je grupa G končno **generirana** z množico $\mathcal{M} = \{a_1, a_2, \ldots, a_n\}$, če je G oblike: $G = \{a_{i_1}^{k_{i_1}} \cdot a_{i_2}^{k_{i_2}} \cdot a_{i_3}^{k_{i_3}} \cdot \ldots \cdot a_{i_N}^{k_{i_N}} ; a_{i_j} \in \{a_1, \ldots a_n\}, k_{i_j} \in \mathbb{Z}\}$. Grupo, generirano z enim samim elementom, imenujemo **ciklična grupa.**

Izrek 2.2 *Vsaka končna ciklična grupa moči n je izomorfna grupi števil $\mathbb{Z}_n$.*

Dokaz: Označimo z G končno ciklično grupo moči n, generirano z elementom $a \in G$. Definirajmo preslikavo $f : \mathbb{Z} \longrightarrow G$, s predpisom $f(n) = a^n$. Ta preslikava je homomorfizem. Slika te preslikave pa je ravno ciklična grupa generirana z a, torej grupa G.
Ker je G končna grupa, potem velja $\ker f = f^{-1}(0) \neq \{0\}$. Če bi namreč veljalo, da je $\ker f = \{0\}$, potem bi, po izreku o kanonični dekompoziciji homomorfizma f, veljalo, da je slika preslikave f izomorfna množici $\mathbb{Z}$, torej $G \approx \mathbb{Z}$, kar pa ne more biti res, če je grupa G končna. Zato velja $\ker f \neq \{0\}$.
Potem pa obstaja nek $n > 0$, $n \in \ker f$. Izberimo n tako, da bo najmanjši izmed vseh takih števil w, za katera velja $w > 0$, $w \in \ker f$.
Naj bo $m \in \ker f$. Potem m lahko zapišemo $m = dn + s$, kjer je s ostanek $0 \leq s < n$. Ker velja $m \in \ker f$ in $n \in \ker f$ in ker je $\ker f$ podgrupa grupe $\mathbb{Z}$, potem velja, da je $s = m - dn$ tudi element $\ker f$. Ker pa velja $s < n$ in smo n izbrali tako, da je najmanjši izmed vseh števil w za katere velja $w > 0$ in $w \in \ker f$, potem mora za s veljati, da $s = 0$. Torej, če je $m \in \ker f$, potem zanj velja $m = dn$, zato je $\ker f$ oblike $\ker f = n\mathbb{Z}$. Slika preslikave f je po izreku o dekompoziciji homomorfizma izomorfna $\mathbb{Z}/n\mathbb{Z} = \mathbb{Z}_n$. Torej je grupa G izomorfna $\mathbb{Z}_n$. ∎

Posledica 2.3 *Vse končne grupe enake moči so med seboj izomorfne.* ∎

Vrnimo se sedaj h grupi $\mathbb{Z}_p^*$, kjer je p praštevilo. Pokazati je mogoče, da je multiplikativna grupa $\mathbb{Z}_p^*$ ciklična. Poglejmo si naslednje trditve.

Trditev 2.4 (Fermat) *Če je $a \in \mathbb{Z}_p^*$, kjer je p praštevilo, potem velja:* $a^{p-1} \equiv 1 \bmod p$.

Bralce, ki so seznanjeni z osnovami teorije grup, naj opomnim, da je ta trditev samo poseben primer Eulerjevega izreka za p je praštevilo. Pa jo dokažimo.

Dokaz: Naj bo t red elementa $a \in \mathbb{Z}_p^*$. Po Lagrangevem izreku vemo, da $t|(p-1)$. Torej lahko zapišemo $k \cdot t = p - 1$, za $k \in \mathbb{Z}$. Potem velja $a^{p-1} = (a^t)^k \equiv 1 \bmod p$. ∎

Trditev 2.5 *Naj ima $a \in \mathbb{Z}_p^*$ red t. Če za število $s \in \mathbb{Z}$ velja $a^s \equiv 1 \bmod p$, potem $t|s$.*

Dokaz: Ker je t red $a \in \mathbb{Z}_p^*$, velja $t < s$. Recimo, da t ne deli s. Potem lahko zapišemo $s = qt + r$, kjer je $0 < r < t$, $r \in \mathbb{Z}$. Velja $a^s = a^{qt+r} = (a^t)^q a^r \equiv a^r \equiv 1 \bmod p$. Torej je r število, ki je manjše od t in zanj velja da $a^r \equiv 1 \bmod p$. To pa je protislovje s predpostavko, da je t red $a \in \mathbb{Z}_p^*$. ∎

Trditev 2.6 *Naj ima $a \in \mathbb{Z}_p^*$ red t in naj ima $b \in \mathbb{Z}_p^*$ red s. Če velja $gcd(s,t) = 1$, potem je red števila $ab \in \mathbb{Z}_p^*$ enak ts.*

Dokaz: Naj ima $a \in \mathbb{Z}_p^*$ red t, $b \in \mathbb{Z}_p^*$ pa red s. Potem velja $(ab)^{ts} \equiv 1 \bmod p$. Pokažimo, da je število ts najmanjše tako število, za katerega velja $(ab)^{ts} \equiv 1 \bmod p$. Recimo, da za neko pozitivno celo število r velja $(ab)^r \equiv 1 \bmod p$. Potem je $(ab)^{rt} = a^{rt}b^{rt} \equiv b^{rt} \equiv 1 \bmod p$. Po trditvi 2.5 s/rt. Ker sta si števili s in t tuji, s deli r. Podobno lahko vidimo, da t deli r. Potem st deli r. Torej je $st \leq r$. To pomeni, da je st najmanjše tako število, da velja $(ab)^{ts} \equiv 1 \bmod p$. Torej je red ab res st. ∎

Trditev 2.7 *Naj bo m maksimalen red števil grupe $\mathbb{Z}_p^*$. Potem red poljubnega elementa iz $\mathbb{Z}_p^*$ deli m.*

Dokaz: Naj ima $a \in \mathbb{Z}_p^*$ maksimalen red m. Naj bo b poljuben element iz $\mathbb{Z}_p^*$ in naj bo njegov red t. Po Lagrangevem izreku števili m in t delita moč grupe $\mathbb{Z}_p^*$, to je $p-1$. Recimo, da t ne deli m. Potem obstaja praštevilski deljitelj q števil t in m, ki ima pri faktorizaciji števila t večji eksponent kot pri faktorizaciji števila m. Zato lahko zapišemo $m = q^e r_1$ in $t = q^f r_2$, kjer je $f > e \geq 0$ in q ne deli r_1 in ne deli r_2. Očitno je red elementa a^{q^e} enak r_1, red elementa b^{r_2} pa q^f. Ker je $\gcd(q^f, r_1) = 1$, je po trditvi 2.6 red elementa $a^{q^e}b^{r_2}$ enak $q^f r_1$. Toda $q^f r_1 > m$, kar je v nasprotju z maksimalnostjo reda m. ∎

Zdaj končno lahko dokažemo tudi naslednjo trditev.

Trditev 2.8 *Multiplikativna grupa $\mathbb{Z}_p^*$, kjer je p praštevilo, je ciklična.*

Dokaz: Naj bo $a \in \mathbb{Z}_p^*$ element z maksimalnim redom m. Pokazati želimo, da je $m = p-1$. Najprej pokažimo, da je $m \leq p-1$. Ker po Lagrangevem izreku m deli $p-1$, potem velja $m \leq p-1$. Zdaj pa pokažimo še, da $m \geq p-1$. Naj bo $b \in \mathbb{Z}_p^*$ poljuben element in naj bo njegov red enak t. Po trditvi 2.7 velja, da t/m. Potem lahko zapišemo $m = k \cdot t$ za nek $k \in \mathbb{Z}$ in velja $b^m = (b^t)^k \equiv 1 \bmod p$. Ker je bil b poljuben element iz $\mathbb{Z}_p^*$, velja ta enačba za vse elemente iz $\mathbb{Z}_p^*$. Teh pa je $p-1$. Iz teorije obsegov vemo, da ima enačba $x^m - 1$ v $\mathbb{Z}_p$ največ m različnih rešitev. Zato velja $m \leq p-1$. ∎

Naj samo povemo, da velja celo več. Tega rezultata v tem delu sicer ne bomo potrebovali in ga zato tudi ne bomo dokazovali, a ga vseeno omenimo.

Izrek 2.9 (Gauss) *Multiplikativna grupa $\mathbb{Z}_n^*$ je ciklična natanko tedaj, ko je $n = 2, 4, p^k, 2p^k$, kjer je p liho praštevilo in $k \geq 1$, $k \in \mathbb{Z}$.* ∎

Torej je multiplikativna grupa $\mathbb{Z}_p^*$, kjer je p praštevilo, ciklična. Ker je njena moč $p-1$, je izomorfna grupi $\mathbb{Z}_{p-1}$. Ker je grupa $\mathbb{Z}_p^*$ ciklična, obstaja element $a \in \mathbb{Z}_p^*$ reda $p-1$, ki to grupo generira. Torej lahko zapišemo $\mathbb{Z}_p^* = \{a^i\,;\, 0 \leq i \leq p-2\}$.

Trditev 2.10 *Vseh različnih generatorjev grupe $\mathbb{Z}_p^*$ je $\varphi(p-1)$, kjer je φ Eulerjeva številska funkcija.*

Dokaz: Naj bo $a \in \mathbb{Z}_p^*$ generator grupe $\mathbb{Z}_p^*$, torej je red a enak $p-1$ in $\mathbb{Z}_p^* = \{a^i\,;\, 0 \leq i \leq p-2\}$. Naj bo b nek od a različen element grupe $\mathbb{Z}_p^*$, potem lahko b enolično zapišemo kot $b = a^i$ za nek $0 \leq i \leq p-2$, $i \neq 1$. Tedaj je red elementa b enak $p-1/\gcd(p-1, i)$ (ker red elementa b deli moč grupe). Torej je b generator grupe $\mathbb{Z}_p^*$ natanko tedaj, ko velja $\gcd(p-1, i) = 1$. Takih pa je natanko $\varphi(p-1)$. ∎

PROBLEM DISKRETNEGA LOGARITMA V $\mathbb{Z}_p^*$

V poglavju o kriptosistemih bomo potrebovali neko operacijo, ki bo relativno lahko izračunljiva, medtem ko bo obrat te operacije zelo težak. Taka operacija oz. bolj natančno inverz take operacije je *problem diskretnega logaritma*, pa si ga kar oglejmo. Za podrobnejše informacije poglejte v [7, str. 103].

PROBLEM DISKRETNEGA LOGARITMA V $\mathbb{Z}_p^*$
Naj bo p praštevilo in naj bo a generator ciklične grupe $\mathbb{Z}_p^*$ in naj bo $b \in \mathbb{Z}_p^*$.
Poišči tako celo število x , $0 \leq x \leq p-2$, da velja $a^x \equiv b \bmod p$.

Število x imenujemo *diskretni logaritem* števila b z osnovo a in označimo $x = \log_a b$. Nekateri avtorji pa ga raje označijo $x = \operatorname{Ind}_a b$, da bi s tem poudarili, da je izračun diskretnega logaritma računsko veliko zahtevnejši problem od izračuna logaritma v realnih številih. Problem diskretnega logaritma bomo krajše pisali DLP (Discrete Logarithm Problem).

Pri danih številih a, x in p je precej lahko izračunati število b, da velja $b \equiv a^x \bmod p$. To storimo z algoritmom *kvadriraj in množi*, več o tem algoritmu v knjigi [7]. Inverzna operacija od operacije potenciranja, to je pri danih a, p in b izračunati x, da velja $a^x \equiv b \bmod p$, pa je precej težka. Za lažje razumevanje si poglejmo primer.

Primer 2.1 : Če je p enak 13 in vzamem $a = 2$ in $b = 11$, potem je težko poiskati x, da velja $2^x \equiv 11 \bmod 13$. Če pa imamo dane $p = 13$, $a = 2$ in $x = 7$, potem precej lahko izračunamo $b = 2^7 \bmod 13 = 11$.

Ker so v tem primeru uporabljena števila majhna, bi problem DLP lahko enostavno rešili tako, da bi izračunali vse možne potence $2^i \bmod 13$ za $0 \leq i \leq 11$, dokler ne bi dobili enakosti $2^i \bmod 13 = 11$. Dobljeni i bi bil torej iskani diskretni logaritem. Če pa si predstavljamo, da je p lahko tudi zelo velik, recimo $p \approx 2^{1024}$, potem tak pristop seveda ne bi bil več zelo uspešen.

Problem DLP je dejansko zelo težak problem, če le vzamemo dovolj velik p. Trenutno še ne obstaja noben algoritem, ki bi znal rešiti ta problem s polinomsko časovno zahtevnostjo. Algoritem, ki je trenutno najboljši, se imenuje *index calculus* metoda in potrebuje za izračun posameznega problema diskretnega logaritma v povprečju $\mathcal{O}(\mathrm{e}^{(\frac{1}{2}+o(1))\sqrt{\ln p \ln\ln p}})$ operacij. O pomenu notacije $\mathcal{O}$ in o bomo pisali v zadnjem razdelku tega poglavja. Več o tej metodi pa si lahko preberete v [7, str. 109-111].

V poglavju o kriptosistemih se bomo ukvarjali predvsem s problemom DLP v podgrupah grupe $\mathbb{Z}_p^*$. Zato ga definirajmo.

PROBLEM DISKRETNEGA LOGARITMA V PODGRUPI H GRUPE $\mathbb{Z}_p^*$
Naj bo p praštevilo in naj bo q tako praštevilo, da $q|(p-1)$. Naj bo $a \in \mathbb{Z}_p^*$ reda q. Potem je $H = \{a^i \,;\, 0 \leq i \leq q-1\}$ ciklična podgrupa moči q grupe $\mathbb{Z}_p^*$. Naj bo $b \in \mathbb{Z}_p^*$.
Poišči tako celo število x , $0 \leq x \leq q-1$, da velja $a^x \equiv b \bmod p$.

Tako definiran problem je še vedno zelo težek, če le vzamemo dovolj veliko praštevilo p. Praštevilo p mora biti tako, da ima število $p-1$ v svoji faktorizaciji vsaj eno veliko praštevilo q. Čeprav računske operacije potekajo znotraj grupe H in je b njen element, moramo, da bi rešili problem DLP z index calculus metodo, le-to še vedno uporabiti v grupi $\mathbb{Z}_p^*$ in ne v podgrupi H. Grupa $\mathbb{Z}_p^*$ pa je neprimerno večja od podgrupe H. Tako število potrebnih operacij za rešitev posameznega problema DLP še vedno ostaja enako, inverzna operacija, to je potenciranje $b \equiv a^x \bmod p$ pa je (zaradi krajših števil v podgrupi H) hitrejša.

2.2 GRUPA ELIPTIČNE KRIVULJE

Problem diskretnega logaritma smo definirali v grupi števil $\mathbb{Z}_p^*$. Videli bomo, da ga lahko definiramo v poljubni grupi. V ta namen bomo definirali grupo eliptične krivulje nad končnim obsegom. Posebej si bomo ogledali grupi nad obsegoma $\mathbb{Z}_p$ in $\mathbb{F}_{2^m}$. Potem bomo definirali problem diskretnega logaritma v grupi eliptične krivulje. Kot bomo videli, je ta problem še težji kot problem diskretnega logaritma v grupi števil $\mathbb{Z}_p^*$.

2.2.1 GRUPA ELIPTIČNE KRIVULJE NAD OBSEGOM $\mathbb{Z}_p$

Poglejmo si najprej množico eliptične krivulje.

Definicija 2.11 *Eliptično krivuljo definiramo kot množico točk*

$$E = \{(x, y) \in \mathbb{R} \times \mathbb{R};\ y^2 = x^3 + ax + b\,;\ a, b \in \mathbb{R}\}.$$

Kot smo že omenili, bodo za nas zanimive predvsem eliptične krivulje nad končnimi obsegi. Pa si poglejmo eliptično krivuljo nad obsegom $\mathbb{Z}_p$.

Definicija 2.12 *Eliptična krivulja nad obsegom $\mathbb{Z}_p$, kjer je p praštevilo $p > 3$, je množica točk $E = \{(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p\,;\ y^2 \equiv x^3 + ax + b \bmod p\,;\ a, b \in \mathbb{Z}_p\}$.*

Če za parametra a in b velja, da $4a^3 + 27b^2 \not\equiv 0 \bmod p$ (kar pomeni, da enačba $y^2 \equiv x^3 + ax + b \bmod p$ nima večkratnih ničel), potem lahko v množico E vpeljemo strukturo grupe.

Definicija 2.13 *Grupa eliptične krivulje nad obsegom števil $\mathbb{Z}_p$, kjer je p praštevilo $p > 3$, je sestavljena iz množice E in posebne točke $\mathcal{O}$, ki jo imenujemo točka neskončno, skupaj z operacijo seštevanja. Označimo jo z $E(\mathbb{Z}_p)$. Za operacijo seštevanja velja naslednje.*

1. *Točka $\mathcal{O}$ je enota za seštevanje: $P + \mathcal{O} = \mathcal{O} + P = P$ za vsako točko $P \in E(\mathbb{Z}_p)$.*
2. *Če je $P = (x, y) \in E(\mathbb{Z}_p)$, potem je $(x, y) + (x, -y) = (x, -y) + (x, y) = \mathcal{O}$. Točko $(x, -y)$ označimo kot $-P := (x, -y)$ in je inverzni element točke P.*
3. *Naj bosta $P = (x_1, y_1) \in E(\mathbb{Z}_p)$ in $Q = (x_2, y_2) \in E(\mathbb{Z}_p)$ in naj velja $P \neq -Q$. Potem $R = P + Q = (x_3, y_3)$ izračunamo na naslednji način:*

$$x_3 = \lambda^2 - x_1 - x_2,$$

$$y_3 = \lambda(x_1 - x_3) - y_1,$$

kjer je

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & če \quad P \neq Q, \\ \\ \dfrac{3x_1^2 + a}{2y_1}, & če \quad P = Q. \end{cases}$$

Vse naštete operacije računamo po modulu p.

Zelo podrobna predstavitev grupe eliptične krivulje je na spletni strani `http:www.certicom.ca.`

Tako definirana grupa $E(\mathbb{Z}_p^*)$ ima končno mnogo točk. Še več, velja naslednja ocena (Hasse) [6]:

$$p+1-2\sqrt{p} \leq |E(\mathbb{Z}_p)| \leq p+1+2\sqrt{p}.$$

Za natančno računanje števila točk pa uporabljamo Schoofov algoritem, njegov podroben opis si lahko ogledate v diplomski nalogi Jerneja Barbiča, diplomanta teoretične matematike iz leta 2000.

2.2.2 GRUPA ELIPTIČNE KRIVULJE NAD OBSEGOM $\mathbb{F}_{2^m}$

Ker operacijo seštevanja v grupi eliptične krivulje nad obsegom $\mathbb{Z}_p$ sestavljata operaciji množenja in seštevanja po modulu p, je še vedno precej počasna. Če pa uporabimo grupo eliptične krivulje nad obsegom $\mathbb{F}_{2^m}$, se hitrost operacije bistveno poveča.

Poglejmo si najprej nekaj lastnosti obsega $\mathbb{F}_{2^m}$.

Obseg $\mathbb{F}_{2^m}$ je končni obseg moči 2^m. Vsi končni obsegi enake moči so si med seboj izomorfni. Torej za vsak $m \in \mathbb{N}$ obstaja do izomorfizma natančno natanko en obseg moči 2^m. Označili ga bomo z $\mathbb{F}_{2^m}$, nekateri avtorji pa ga označujejo tudi s GF(2^m) (Galois Field).

Elemente obsega $\mathbb{F}_{2^m}$ lahko predstavimo kot polinome stopnje manjše od m s koeficienti v $\mathbb{Z}_2$ ali pa kot m-bitna zaporedja. To je:

$$\mathbb{F}_{2^m} = \{a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_2x^2 + a_1x + a_0\,;\, a_i \in \{0,1\} \text{ za } i = 0, \ldots, m-1\}$$

ali

$$\mathbb{F}_{2^m} = \{(a_{m-1}, a_{m-2}, \ldots, a_2, a_1, a_0)\,;\, a_i \in \{0,1\} \text{ za } i = 0, \ldots, m-1\}.$$

Glavni operaciji v $\mathbb{F}_{2^m}$ sta seštevanje in množenje. Operacijo množenja računamo s pomočjo polinoma

$$f(x) = x^m + f_{m-1}x^{m-1} + \ldots + f_2x^2 + f_1x + f_0\,,\; f_i \in \{0,1\} \text{ za } i = 0, \ldots, m-1,$$

ki je nerazcepen znotraj $\mathbb{F}_{2^m}$. Za polinom $f(x)$ običajno izberemo trinom

$$f(x) = x^m + x^k + 1,$$

ker se operacija redukcije tako precej poenostavi.

Pa si poglejmo operacije v obsegu $\mathbb{F}_{2^m}$.

Seštevanje
Naj bosta $a = (a_{m-1}, \ldots, a_1, a_0) \in \mathbb{F}_{2^m}$ in $b = (b_{m-1}, \ldots, b_1, b_0) \in \mathbb{F}_{2^m}$, potem definiramo: $a + b = c = (c_{m-1}, \ldots, c_1, c_0)$, kjer za vsak c_i velja $c_i = a_i + b_i \bmod 2$ za $i = 0, \ldots, m-1$. Hitro se prepričamo, da je operacija seštevanja v bistvu ekskluzivni ali (XOR) bitnega zaporedja a z bitnim zaporedjem b, torej $a + b = a \oplus b$.

Enota za seštevanje je element $(0,0,\ldots,0,0) \in \mathbb{F}_{2^m}$.

Inverzni element za seštevanje
Ker velja $(a_{m-1},\ldots,a_1,a_0)+(a_{m-1},\ldots,a_1,a_0) = (0,\ldots,0,0)$ za vsak $(a_{m-1},\ldots,a_1,a_0) \in \mathbb{F}_{2^m}$, je $(a_{m-1},\ldots,a_1,a_0)$ sam svoj inverzni element za seštevanje. Hkrati pa lahko ugotovimo, da sta tako operaciji seštevanja in odštevanja ena in ista operacija.

Množenje
Naj bosta $a = (a_{m-1},\ldots,a_1,a_0) \in \mathbb{F}_{2^m}$ in $b = (b_{m-1},\ldots,b_1,b_0) \in \mathbb{F}_{2^m}$, potem definiramo: $a \cdot b = r = (r_{m-1},\ldots,r_1,r_0)$, kjer polinom $r_{m-1}x^{m-1} + r_{m-2}x^{m-2} + \ldots + r_1x + r_0$ dobimo tako, da izračunamo ostanek pri deljenju zmnožka polinomov $(a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_1x + a_0) \cdot (b_{m-1}x^{m-1} + b_{m-2}x^{m-2} + \ldots + b_1x + b_0)$ s polinomom f. Označimo $r = a \cdot b \bmod f$. Ta operacija je analogna operaciji množenja po modulu praštevila p v obsegu $\mathbb{Z}_p$. Vse operacije seveda spet računamo po modulu 2.

Enota za množenje je element $(0,0,\ldots,0,0,1) \in \mathbb{F}_{2^m}$.

Inverzni element za množenje
$\mathbb{F}_{2^m}$ je končni obseg, torej je $\mathbb{F}^*_{2^m}$ multiplikativna grupa. Dokazali bomo celo, da je ta grupa ciklična. Zdaj torej to dejstvo samo privzemimo. Ker je $\mathbb{F}^*_{2^m}$ ciklična grupa, obstaja vsaj en element $\mathbb{F}^*_{2^m}$, ki jo generira. Torej lahko vsak element $\mathbb{F}^*_{2^m}$ enolično zapišemo kot $a = g^i$ za $i = 0,1,\ldots,2^m-1$. Zlahka se prepričamo, da je potem inverz za množenje elementa a enak $a^{-1} = g^{(-i)\bmod 2^m-1}$. Velja namreč $a \cdot a^{-1} = g^i \cdot g^{-i \bmod 2^m-1} = g^{(i-i)\bmod 2^m-1} = g^0 = 1$.

Za boljše razumevanje si poglejmo majhen primer.

Primer 2.2 : Obseg $\mathbb{F}_{2^4}$ sestavlja 16 bitnih zaporedij dolžine 4:

(0000)	(0001)	(0010)	(0100)
(1000)	(0011)	(0101)	(1001)
(0110)	(1010)	(1100)	(1111)
(0111)	(1011)	(1101)	(1110)

Recimo, da za nerazcepen polinom uporabimo $f(x) = x^4 + x + 1$. Za elementa obsega pa si izberemo $a = (1101)$ in $b = (1001)$. Potem je $(1101) + (1001) = (0100)$. In $(1101) \cdot (1001) =$

$$= (x^3 + x^2 + 1)(x^3 + 1) \bmod f(x) \;=\; (x^6 + x^5 + 2x^3 + x^2 + 1) \bmod f(x) \;=$$

$$=\; (x^6 + x^5 + x^2 + 1) \bmod f(x) \;=\; (x^6 + x^5 + 2x^3 + x^2 + 1) \bmod (x^4 + x + 1) \;=$$

$$=\; (x^4 + x + 1)(x^2 + x) + (x^3 + x^2 + x + 1) \bmod (x^4 + x + 1) \;=$$

$$=\; x^3 + x^2 + x + 1 \;=\; (1111).$$

Če si izberemo $a = (1011)$, potem a^{-1} izračunamo takole. Ker je $g = (0010)$ generator grupe $\mathbb{F}^*_{2^4}$ in velja $a = g^7$, je $a^{-1} = g^{-7 \bmod 15} = g^{8 \bmod 15} = g^8 = (0101)$. Zlahka preverimo, da je dobljeni a^{-1} res inverz za množenje elementa a.

Ker se operacije v obsegu $\mathbb{F}_{2^m}$ vršijo na m-bitnih zaporedjih, jih precej lažje implementiramo v računalnike kot operacije obsega $\mathbb{Z}_p$. To velja predvsem za seštevanje, ki je kar operacija ekskluzivni ali dveh m-bitnih zaporedij. Velja pa tudi za množenje oz. potenciranje, ki je še vedno precej lažje od potenciranja po modulu praštevila p.

Obljubili smo še dokaz dejstva, da je grupa $\mathbb{F}_{2^m}^*$ ciklična. V ta namen bomo definirali nova pojma.

Če je operacija v grupi $(G, \circ)$ komutativna, tj. $x \circ y = y \circ x$ za vsaka $x, y \in G$, pravimo, da je grupa G **Abelova**. Grupa $\mathbb{F}_{2^m}^*$ je Abelova, saj velja $a \cdot b\,(\mathrm{mod} f) = b \cdot a\,(\mathrm{mod} f)$.

Naj bo G grupa moči m. Naj bo p praštevilo, ki deli m in k najvišja potenca tega praštevila, tako da p^k še deli m. Če obstaja podgrupa H grupe G z močjo p^k, potem ji pravimo **podgrupa Sylowa**.

Izrek 2.14 (Sylow) *Ob zgornjih predpostavkah podgrupa Sylowa vselej obstaja.* ∎

Dokaz tega izreka bomo izpustili, lahko pa si ga pogledate v [9].

Trditev 2.15 *Vsaka končna Abelova grupa A se da zapisati na en sam način kot direktna vsota svojih podgrup: $A = H_1 \oplus H_2 \oplus \ldots \oplus H_n$, kjer so H_i moči $p_1^{k_i}$ in so p_i sama različna praštevila za $i = 1, \ldots, n$.* ∎

Te trditve ne bomo dokazali. Naj samo povemo, kako poiščemo te podgrupe. Namreč če je moč grupe A enaka $m = p_1^{k_1} \cdot p_1^{k_1} \cdot \ldots \cdot p_n^{k_n}$, potem so podgrupe H_i natanko podgrupe Sylowa. Zdaj lahko dokažemo, da je grupa $\mathbb{F}_{2^m}^*$ ciklična.

Trditev 2.16 *Multiplikativna grupa $\mathbb{F}_{2^m}^*$ je ciklična.*

Dokaz: Grupa $\mathbb{F}_{2^m}^*$ je Abelova po definiciji in ima moč $2^m - 1$. Faktorizirajmo to število $2^m - 1 = p_1^{k_1} \cdot p_2^{k_2} \cdot \ldots \cdot p_n^{k_n}$, kjer so p_i paroma različna praštevila in $k_i \in \mathbb{Z}$, $k_i \geq 1$ za vsak $i = 1, \ldots n$. Potem se da $\mathbb{F}_{2^m}^*$ na en sam način zapisati kot direktna vsota podgrup Sylowa: $\mathbb{F}_{2^m}^* = H_1 \oplus H_2 \oplus \ldots \oplus H_n$, kjer je moč H_i enaka $p_i^{k_i}$ za $i = 1, \ldots n$. Pokažimo, da je red poljubnega elementa iz H_i enak neki potenci praštevila p_i. Izberimo poljuben $a_i \in H_i$. Po Lagrangevem izreku red elementa a_i deli $p_i^{k_i}$. To pa je mogoče le, če je njegov red enak neki potenci praštevila p_i. Ker je bil element a_i poljubno izbran, lahko trdimo, da je red vsakega elementa iz H_i enak neki potenci praštevila p_i. Zdaj pa izberimo $a_i \in H_i$ tak, da ima maksimalni red. Naj bo red elementa a_i enak $r_i = p_i^{l_i}$, kjer je $l_i \leq k_i$. Ker velja $\gcd(p_i, p_j) = 1$ za vsak $i \neq j$, ima element $a = a_1 \cdot a_2 \cdot \ldots \cdot a_n \in \mathbb{F}_{2^m}^*$ po trditvi 2.6 red enak $r = r_1 \cdot r_2 \cdot \ldots \cdot r_n = p_1^{l_1} \cdot p_2^{l_2} \cdot \ldots \cdot p_n^{l_n}$. Velja tudi, da je red elementa a maksimalen red elementov grupe $\mathbb{F}_{2^m}^*$. Ker po Lagrangevem izreku velja, da r deli $2^m - 1$, zato velja, da je $r \leq 2^m - 1$. Naj bo b_i poljuben element iz H_i. Kot smo že videli, je njegov red enak $p_i^{f_i}$, kjer je $f_i \leq l_i$. Spet sklepamo podobno kot prej. Ker je $\gcd(p_i, p_j) = 1$ za vsak $i \neq j$, ima element $b = b_1 \cdot b_2 \cdot \ldots \cdot b_n \in \mathbb{F}_{2^m}^*$ po trditvi 2.6 red enak $t = p_1^{f_1} \cdot p_2^{f_2} \cdot \ldots \cdot p_n^{f_n}$. Torej red b deli red a. Zato velja $r = k \cdot t$ za $k \in \mathbb{Z}$ in velja $b^r = (b^t)^k = 1$. Ker je bil element b poljuben, velja $x^r = (x^t)^k = 1$ za vsak element $x \in \mathbb{F}_{2^m}^*$, takih pa je natanko $2^m - 1$. Iz teorije obsegov pa vemo, da ima enačba $x^r - 1$ največ r rešitev v $\mathbb{F}_{2^m}^*$, torej je $r \geq 2^m - 1$. Pokazali smo, da je red elementa $a \in \mathbb{F}_{2^m}^*$ enak $r = 2^m - 1$, torej je a generator grupe $\mathbb{F}_{2^m}^*$, ki je tako ciklična. ∎

Zdaj pa si poglejmo še grupo eliptične krivulje nad obsegom $\mathbb{F}_{2^m}$. Enačba, ki ji v tem primeru zadoščajo točke eliptične krivulje, ima obliko $y^2 + xy = x^3 + ax + b$, kjer sta $a, b \in \mathbb{F}_{2^m}$ in velja, da $b \neq 0$.

Definicija 2.17 *Grupa eliptične krivulje nad obsegom* $\mathbb{F}_{2^m}$ *je množica točk* $E(\mathbb{F}_{2^m}) = \{(x, y) \in \mathbb{F}_{2^m} \times \mathbb{F}_{2^m}\, ;\, y^2+xy = x^3+ax+b\, ;\, a, b \in \mathbb{F}_{2^m}\ b \neq 0\} \cup \{\textit{točka neskončno } \mathcal{O}\}$ *z operacijo seštevanja, za katero velja naslednje.*

1. *Točka* $\mathcal{O}$ *je enota za seštevanje:* $P + \mathcal{O} = \mathcal{O} + P = P$ *za vsako točko* $P \in E(\mathbb{F}_{2^m})$.
2. *Če je* $P = (x, y) \in E(\mathbb{F}_{2^m})$, *potem je* $(x, y) + (x, x + y) = (x, x + y) + (x, y) = \mathcal{O}$. *Točko* $(x, x + y)$ *označimo kot* $-P := (x, x + y)$ *in je inverzni element točke* P.
3. *Naj bosta* $P = (x_1, y_1) \in E(\mathbb{F}_{2^m})$ *in* $Q = (x_2, y_2) \in E(\mathbb{F}_{2^m})$ *in naj velja* $P \neq -Q$. *Potem* $R = P + Q = (x_3, y_3)$ *izračunamo na naslednji način:*

$$x_3 = \lambda^2 + \lambda + a + x_1 + x_2,$$

$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1,$$

kjer je

$$\lambda = \begin{cases} \dfrac{y_2 + y_1}{x_2 + x_1}, & \textit{če} \quad P \neq Q, \\[2ex] \dfrac{x_1 + y_1}{x_1}, & \textit{če} \quad P = Q. \end{cases}$$

Za podrobnejše informacije spet priporočam učbenik na spletni strani `http:www.certicom.ca.`, mi pa se posvetimo problemu diskretnega logaritma v grupi $E(\mathbb{F}_{2^m})$.

PROBLEM DISKRETNEGA LOGARITMA V $E(\mathbb{F}_{2^m})$

Kot smo že omenili, so operacije v grupi eliptične krivulje $E(\mathbb{F}_{2^m})$ precej hitrejše in jih je lažje implementirati v računalniško okolje od onih v grupi $E(\mathbb{Z}_p)$, zato bomo problem diskretnega logaritma definirali in uporabljali v grupi $E(\mathbb{F}_{2^m})$.

PROBLEM DISKRETNEGA LOGARITMA V GRUPI $E(\mathbb{F}_{2^m})$
Naj bo $E(\mathbb{F}_{2^m})$ grupa eliptične krivulje nad končnim obsegom $\mathbb{F}_{2^m}$ moči N, kjer je N praštevilo in naj bo n tako praštevilo, da $n|N$. Naj bo $P \in \mathbb{F}_{2^m}$ točka reda n. Potem je $H = \{iP\, ;\, 0 \leq i \leq n - 1\}$ ciklična podgrupa moči n grupe $\mathbb{F}_{2^m}$. Naj bo $Q \in \mathbb{F}_{2^m}$.
Poišči tako celo število d , $0 \leq d \leq n - 1$, da velja $dP = Q$.

Problem diskretnega logaritma v grupi eliptične krivulje bomo krajše pisali ECDLP (Elliptic Curve Discrete Logarithm Problem).

Kot lahko opazimo, sta problema DLP in ECDLP povsem analogna, le da se operacija v DLP imenuje množenje, v ECDLP pa seštevanje (ker smo tako definirali operacije v grupi

$\mathbb{Z}_p^*$ oz. $E(\mathbb{F}_{2^m})$). Splošno prepričanje je, da je problem ECDLP mnogo težje rešljiv od problema DSA, če le uporabimo ustrezni vrednosti p in m. Bolj podrobno bomo problema primerjali v naslednjem poglavju, ko ju bomo dejansko uporabili v algoritmu digitalnega podpisa. Naj samo povemo, da je trenutno najboljši algoritem za reševanje problema ECDLP *Pollardova ρ-metoda.* Ta metoda za rešitev posameznega problema potrebuje približno $\sqrt{\pi n/2}$ operacij, kjer za operacijo štejemo seštevanje v grupi eliptične krivulje. Leta 1993 sta Paul von Oorschot in Michael Wiener pokazala, kako lahko to metodo uporabimo kot vzporedno metodo na r procesorjih. Potem je pričakovano število operacij za rešitev problema ECDLP približno $\sqrt{\pi n/2}/r$. Označimo besedno zvezo *milijon ukazov na sekundo* s kratico MIPS (million instructions per second), stroj, ki je zmožen izvesti milijon ukazov na sekundo, pa MIPS stroj. Potem velja, da MIPS stroj lahko izvede $4 \cdot 10^4$ seštevanj v grupi eliptične krivulje na sekundo. Zato je število seštevanj v grupi eliptične krivulje, ki jih MIPS stroj lahko izvede v enem letu, enako $(4 \cdot 10^4) \cdot (60 \cdot 60 \cdot 24 \cdot 365) \approx 2^{40}$. Tabela 2.3 nam za različne vrednosti n prikazuje potrebni čas za rešitev enega samega problema ECDLP (s Pollardovo ρ-metodo). Z oznako MIPS let smo označili število let, ki jih potrebuje MIPS stroj za rešitev problema ECDLP.

Velikost obsega $\mathbb{F}_{2^m}$ (v bitih)	Velikost parametra n (v bitih)	$\sqrt{\pi n/2}$	MIPS let
155	150	2^{75}	$3.8 \cdot 10^{10}$
210	205	2^{108}	$7.1 \cdot 10^{18}$
239	234	2^{117}	$1.6 \cdot 10^{28}$

Tabela 2.3: Čas, potreben za rešitev problema ECDLP, v odvisnosti od dolžine parametra n

Kot vidimo, so številke v tabeli 2.3 precej velike in nekako nedosegljive. Recimo, če imamo na razpolago 10 000 računalnikov, od katerih vsak zmore 1000 MIPS, potem bomo za rešitev enega samega problema ECDLP potrebovali kar 3800 let. Z uporabo vzporednih algoritmov bi za $n \approx 10^{36} \approx 2^{120}$ in s strojem z 325 000 procesorji (mimogrede, cena takega stroja bi znašala približno 10 milijonov USD) za rešitev posameznega problema ECDLP potrebovali 35 dni. Torej so številke nedosegljive in je problem ECDLP resnično težak. Več o tem si poglejte v [3].

KOMPRESIJA IN DEKOMPRESIJA TOČK GRUPE $E(\mathbb{F}_{2^m})$

Naj točka P pripada grupi eliptične krivulje $E(\mathbb{F}_{2^m})$ podane z enačbo $y^2+xy = x^3+ax^2+b$, kjer sta $a, b \in \mathbb{F}_{2^m}$ in $b \neq 0$. Potem točko P podamo z $P = (x, y)$, kjer sta x in y elementa obsega $\mathbb{F}_{2^m}$, ki zadoščata zgornji enačbi. Vsak element obsega $\mathbb{F}_{2^m}$ je neko m-bitno število, kar pomeni, da za predstavitev točke P potrebujemo $2m$ bitov. Ker pa je m običajno zelo veliko število, je tudi predstavitev točke P zelo dolga. Na srečo pa se izkaže, da za predstavitev točke P ne potrebujemo v celoti obeh njenih koordinat. Zadostovalo bo zgolj poznati koordinato x točke P in nek določen bit koordinate y. Tako bomo točko P podali že z $m + 1$ biti.

KOMPRESIJA TOČKE $P = (x, y) \in E(\mathbb{F}_{2^m})$

Kompresija točke $P \in E(\mathbb{F}_{2^m})$ je postopek, s katerim točko P podamo zgolj z $m + 1$ biti.

Rezultat tega postopka je neko m-bitno zaporedje A in nek bit B. Ta postopek je zelo enostaven.

Kompresija točke $P = (x, y)$

1. Postavi $A := x$
2. Poišči mesto prvega neničelnega bita v koordinati x.
3. Če na istem mestu v koordinati y stoji 1, potem postavi $B := 1$.
   Sicer postavi $B := 0$.

Tak par (A, B) imenujemo **kompresirana** oz. **stisnjena** točka. Zdaj pa si oglejmo še obraten proces.

DEKOMPRESIJA KOMPRESIRANE TOČKE (A, B)

Dekompresija kompresirane točke (A, B) je postopek, s katerim poiščemo točko $P = (x, y)$, to je poiščemo njeno koordinato y. Ta postopek pa je nekoliko težji.
Koordinato x točke P poznamo, saj je le-ta enaka $x := A$. Koordinato y pa bomo izračunali s pomočjo enačbe $y^2 + xy = x^3 + ax^2 + b$, ki ji morata zadoščati x in y. To pomeni, da moramo rešiti kvadratno enačbo za $y \in \mathbb{F}_{2^m}$. Reševanje kvadratnih enačb v obsegu $\mathbb{F}_{2^m}$ pa se nekoliko razlikuje od reševanja enačb v obsegu $\mathbb{R}$. Zato si bomo morali pogledati nekaj novih pojmov. Najprej bomo definirali sled in pogledali nekaj lastnosti, ki pa jih ne bomo dokazovali. Dokaze si lahko ogledate v diplomskem delu [10].
Vsak obseg je vektorski prostor nad poljubnim svojim podobsegom. Tako je $\mathbb{F}_{2^m}$ vektorski prostor nad $\mathbb{Z}_2$.
Sled je preslikava $\mathrm{sl} : \mathbb{F}_{2^m} \longrightarrow \mathbb{Z}_2$ definirana s predpisom $\mathrm{sl}(x) = x^{2^0} + x^{2^1} + \ldots + x^{2^{m-1}}$.
Sled sl je linearen funkcional na vektorskem prostoru $\mathbb{F}_{2^m}$ nad $\mathbb{Z}_2$.
Lastnosti funkcionala sl:

1. $\mathrm{sl}(x)^2 = \mathrm{sl}(x)$ za vsak $x \in \mathbb{F}_{2^m}$.
2. $\mathrm{sl}(x^{2^i}) = \mathrm{sl}(x)$ za vsak $x \in \mathbb{F}_{2^m}$ in vsak $i = 0, \ldots, m-1$.
3. Polinom $\mathrm{sl}(x) = \sum_{i=0}^{m-1} x^{2^i}$ ima v $\mathbb{F}_{2^m}$ natanko 2^{m-1} različnih ničel.

Zdaj pa si poglejmo še reševanje kvadratnih enačb v obsegu $\mathbb{F}_{2^m}$. Videli bomo, da ni vsaka kvadratna enačba v obsegu $\mathbb{F}_{2^m}$ rešljiva. Oglejmo si splošno enačbo oblike

$$ax^2 + bx + c = 0 \quad a, b, c \in \mathbb{F}_{2^m} \quad \text{in} \quad a \neq 0.$$

To enačbo pomnožimo z a^{-1} in dobimo enačbo

$$x^2 + ux + v = 0 \quad u, v \in \mathbb{F}_{2^m}.$$

Najprej si oglejmo primer, ko je $u = 0$.

Trditev 2.18 *Naj bo v poljuben element obsega $\mathbb{F}_{2^m}$. Enačba $x^2 = v$ je rešljiva za vsak $v \in \mathbb{F}_{2^m}$ in ima pri vsakem $v \in \mathbb{F}_{2^m}$ natanko eno dvojno rešitev.* ∎

Iz te trditve neposredno sledi, da velja razcep

$$x^2 - v = (x - \sqrt{v})(x + \sqrt{v}) = (x + \sqrt{v})(x + \sqrt{v}).$$

Element $\sqrt{v}$ pa lahko enostavno izračunamo. Velja namreč

$$(v^{2^{m-1}})^2 = v^{2^m} = v, \text{ torej } \sqrt{v} = v^{2^{m-1}}.$$

Težji primer nas čaka, ko $u \neq 0$. Potem enačbo $x^2 + ux + v = 0$, kjer sta $u, v \in \mathbb{F}_{2^m}$ pomnožimo še z u^{-2} in uvedemo novo spremenljivko $y = u^{-1}x$. Enačba se poenostavi v obliko

$$y^2 + y = w \quad w \in \mathbb{F}_{2^m}.$$

Trditev 2.19 *Enačba $y^2 + y = w$, kjer $w \in \mathbb{F}_{2^m}$ ima rešitev v $\mathbb{F}_{2^m}$ natanko tedaj, ko je sled elementa w enaka 0.* ∎

Zdaj pa se končno lahko lotimo dekompresije kompresirane točke. Torej imamo m-bitno zaporedje A in nek bit B. Radi pa bi poiskali točko $P = (x, y)$ tako, da bi njene koordinate zadoščale enačbi $y^2 + xy = x^3 + ax^2 + b$, kjer sta $a, b \in \mathbb{F}_{2^m}$ in $b \neq 0$.

Dekompresija kompresirane točke (A, B)

1. Postavi $x := A$.
2. Izračunaj $\alpha = x + a + bx^{-2} \quad (\alpha = (y/x)^2 + (y/x))$.
3. Če je $\mathrm{sl}(\alpha) = 0 \quad$ (enačba $z^2 + z = \alpha$ ima rešitev v $\mathbb{F}_{2^m}$)
    poišči tak z, da velja $z^2 + z = \alpha$,
    izračunaj $y = xz$,
    poišči mesto prvega neničelnega bita v koordinati x,
    če $B = 1$
        če na istem mestu v y stoji 0, rešitev je $-P$ (pomeni $y = x + y$).
    če $B = 0$
        če na istem mestu v y stoji 1, rešitev je $-P$ (pomeni $y = x + y$).

Za boljše razumevanje si sedaj poglejmo še primer.

Primer 2.4
Vzemimo $m = 25$, torej obseg $\mathbb{F}_{2^{25}}$ in naj bosta parametra enačbe, ki ji zadoščajo točke eliptične krivulje enaka

$$\begin{array}{rclcll} a & = & 4322 & = & (0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,1,1,0,0,0,1,0) & \text{in} \\ b & = & 9 & = & (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1). & \end{array}$$

Potem točka P s koordinatama

$$\begin{array}{rcll} x & = & (0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,1) & \text{in} \\ y & = & (0,0,0,1,1,1,0,1,0,0,0,1,1,1,1,0,0,1,1,0,1,1,1,0,1,0) & \end{array}$$

leži na tej krivulji. Poskusimo ugotoviti, kakšna je kompresirana točka.
Zaporedje m bitov A je enako koordinati x. Prvi neničelni bit koordinate x je kar prvi bit. Prvi bit koordinate y pa je enak 0, zato $B = 0$. To je naša kompresirana točka. Izračunajmo sedaj še njeno dekompresijo.
Najprej postavimo x koordinato enako zaporedju A. Potem izračunamo

$$\alpha = x + a + bx^{-2} = (0,0,1,1,1,0,0,0,0,1,0,0,1,1,1,0,1,0,0,0,1,0,1,1,0,0).$$

Potem izračunamo sled elementa α, da se prepričamo ali je enačba $z^2 + z = \alpha$ sploh rešljiva v $\mathbb{F}_{2^{25}}$ in dobimo

$$\mathrm{sl}(\alpha) = 0.$$

Poiščemo tak z, da $z^2 + z = \alpha$ in dobimo

$$z = (0,1,0,0,1,0,0,0,1,0,0,0,1,1,1,1,0,1,0,1,1,1,1,1,0,0).$$

Postopek za iskanje takega z je naslednji.

Postopek za reševanje enačbe $z^2 + z = \alpha$

1. Če je m liho število, izračunaj
$$z = \alpha + \alpha^{2^2} + \alpha^{2^4} + \ldots + \alpha^{2^{m-1}}.$$
2. Sicer naredi naslednje korake:
3. izberi naključni $r \in \mathbb{F}_{2^m}$,
4. postavi $z = 0$ in $w = r$,
5. za $i = 1$ do $m - 1$ naredi:
$$z = z^2 + w^2\alpha \quad \text{in} \quad w = w^2 + r.$$
6. Če je w enak 0, pojdi na 2. korak, sicer je dobljeni z rešitev.

Končno izračunamo še koordinato y

$$y = z \cdot x = (0,0,0,1,1,1,0,1,0,0,0,1,1,1,1,0,0,1,1,0,1,1,1,0,1,0).$$

Poiščemo prvi neničelni bit v koordinati x, to je kar prvi bit. Ker je bit B enak 0 in v y na prvem mestu stoji 0, je dobljeni y že naša rešitev.

2.3 ALGORITMI

Ko bomo računali digitalni podpis, bomo to storili z algoritmom za izračun digitalnega podpisa. Zato bomo v tem razdelku definirali pojem algoritma in pogledali nekatere njegove lastnosti.

Algoritem je računski postopek, ki pri danih podatkih v končnem času reši problem in vrne rezultat. Ponavadi nas najbolj zanima, kako dolg je ta čas, zato definiramo naslednji pojem.

Definicija 2.20 **Časovna zahtevnost** *algoritma pri določenih začetnih podatkih je enaka številu izvedenih osnovnih operacij. Osnovna operacija je ponavadi množenje ali seštevanje števil.*

V večini primerov je težko določiti natančno časovno zahtevnost algoritma, zato si pomagamo z asimptotično oceno. To pomeni, da gledamo, kako se časovna zahtevnost povečuje s povečanjem velikosti podatkov. V ta namen bomo definirali $\mathcal{O}$-notacijo in o-notacijo.

Definicija 2.21 *Naj bosta $f(n)$ in $g(n)$ neki funkciji. Če obstaja konstanta $c > 0$ in tako število $n_0 > 0$, da velja $0 \leq f(n) < c \cdot g(n)$ za vse $n \leq n_0$, potem označimo $f(n) = \mathcal{O}(g(n))$. Z drugimi besedami, $f(n)$ ne raste asimptotično hitreje kot $g(n)$ do multiplikativne konstante natančno. Recimo:* $384x^2 + 12x + 232 = \mathcal{O}(x^2)$ *ali* $\sin x = \mathcal{O}(x)$.

Definicija 2.22 *Naj bosta $f(n)$ in $g(n)$ neki funkciji. Če za vsako konstanto $c > 0$ obstaja tako število $n_0 > 0$, da velja $0 \leq f(n) < c \cdot g(n)$ za vse $n \leq n_0$, potem označimo $f(n) = o(g(n))$. Z drugimi besedami, $g(n)$ je zgornja meja za $f(n)$ ali $\lim_{n\to\infty} f(n)/g(n) = 0$. Recimo:* $19992x = o(x^2)$ *ali* $\log x = o(x)$ *ali* $x^n = o(\mathrm{e}^x)$.

Definicija 2.23 **Polinomski algoritem** *je algoritem, katerega časovna zahtevnost je enaka $\mathcal{O}(n^k)$, kjer je n velikost podatka in k neka konstanta $k > 0$. Algoritem, katerega časovne zahtevnosti ne moremo tako omejiti, imenujemo* **eksponentni algoritem**.

Velikost podatka se ponavadi meri v številu bitov, ki jih potrebujemo za njegovo reprezentacijo. Npr. velikost pozitivnega števila n je približno $\ln n$. V tem primeru je algoritem polinomski, če je njegova časovna zahtevnost $\mathcal{O}((\ln n)^k)$ za neko konstanto k in je eksponentni, če je njegova časovna zahtevnost enaka $\mathcal{O}(\mathrm{e}^{(f(\ln n))})$, kjer je f neka funkcija. V praksi veljajo polinomski algoritmi za učinkovite algoritme, medtem ko eksponentni veljajo za neučinkovite in zato neuporabne v smislu prevelike časovne zahtevnosti. V prejšnjem razdelku smo spoznali problem diskretnega logaritma v grupi $\mathbb{Z}_p^*$. Spoznali smo, da je to zelo težek problem v tem smislu, da ne obstaja učinkovit, to je polinomski, algoritem za rešitev tega problema. Najhitrejši možni algoritem je index calculus metoda, katere časovna zahtevnost je $\mathcal{O}(\mathrm{e}^{(\frac{1}{2}+o(1))\sqrt{\ln p \ln \ln p}})$. Podatek je v tem primeru praštevilo p, torej je njegova dolžina enaka $\ln p$. Kot vidimo, je ta algoritem eksponentni in torej neučinkovit.

Poglavje 3

KRIPTOGRAFSKI PROTOKOLI

V tem poglavju se bomo ukvarjali predvsem z digitalnim podpisom. Digitalni podpis je metoda podpisovanja dokumentov, ki so shranjeni v elektronski obliki. Ko se ročno podpišemo pod nek dokument, to običajno pomeni, da smo seznanjeni z njegovo vsebino in da se z njo strinjamo, ali pa potrjujemo svojo prisotnost. Poleg tega podpis služi tudi za identifikacijo podpisnika, saj vemo, da so podpisi različnih ljudi različni. Seveda pričakujemo, da ima digitalni podpis prav take lastnosti. Vendar se izkaže, da ima lahko digitalni podpis poleg naštetih lastnosti še kakšno novo lastnost. Najbolj pomembna med njimi je ta, da dokumenta, ko bo enkrat podpisan, ni več možno neopazno spreminjati. Zamislimo si recimo, da želimo dvigniti denar v banki in v ta namen podpišemo dokument, na katerega smo zapisali željeni znesek dviga. Bančni uslužbenec nam denar izplača, a ko odidemo, vpiše na koncu zneska dodatno ničlo in razliko spravi v svoj žep. Z algoritmom digitalnega podpisa poskrbimo, da take stvari niso več mogoče. Vsaka, še tako majhna sprememba podpisanega dokumenta povzroči, da postane njegov digitalni podpis neveljaven in s tem tudi celoten dokument. Da bi lažje razumeli sam algoritem digitalnega podpisa, pa se bomo, v razdelku 3.1, morali najprej seznaniti z osnovami kriptografije. Bolj podrobno si bomo ogledali le ElGamalove kriptosisteme, ki temeljijo na problemu diskretnega logaritma. Potem se bomo, v razdelku 3.2, lotili različnih algoritmov digitalnega podpisa. Pogledali si bomo njihove prednosti in slabosti. Na koncu poglavja, v razdelku 3.3, pa bomo omenili še zgoščevalne funkcije.

3.1 UVOD V KRIPTOGRAFIJO

V tem razdelku si bomo pogledali osnove kriptografije. Najprej si bomo pogledali njeno definicijo. Potem se bomo seznanili z dvema glavnima razredoma kriptosistemov. To bosta razred simetričnih kriptosistemov in razred kriptosistemov z javnimi ključi. Med slednje spada tudi ElGamalov kriptosistem, ki si ga bomo ogledali bolj podrobno.
Beseda kriptografija izvira iz grških besed *Kryptos*, ki pomeni skriven in *Graphen*, ki pomeni pisati. Torej lahko besedo kriptografija razumemo kot vedo o skrivnih pisavah. Njen osnovni cilj je omogočiti varen način komunikacije med dvema osebkoma, imenujmo

ju Anže in Boštjan. Varen način komunikacije pomeni, če si recimo Anže in Boštjan dopisujeta prek javnega kanala, to je lahko računalniška mreža ali pa so to nezaščitene telefonske linije, nihče razen njiju ni sposoben ugotoviti, o čem teče beseda. Govorili bomo o *čistopisu*, ki je sporočilo, ki ga Anže želi poslati Boštjanu. Govorili bomo tudi o *tajnopisu*, to pa je sporočilo, ki ga Anže dejansko pošlje Boštjanu in za katerega velja, da ga nihče razen Boštjana ne razume. Anže torej, s pomočjo prej izbranega *ključa*, zašifrira svoj čistopis v odgovarjajoči tajnopis in ga pošlje Boštjanu. Ker je Boštjan edini, ki ga razume, (to je, je edini, ki zna ta tajnopis z uporabo ključa dešifrirati v čistopis), nihče razen njiju ne bo mogel ugotoviti vsebine poslanega sporočila.

Poglejmo si še formalno definicijo kriptosistema.

Definicija 3.1 *Kriptosistem je peterica* $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, *ki zadošča naslednjim zahtevam:*

1. $\mathcal{P}$ *je končna množica možnih čistopisov.*
2. $\mathcal{C}$ *je končna množica možnih tajnopisov.*
3. $\mathcal{K}$ *je končna množica možnih ključev.*
4. *Za vsak* $K \in \mathcal{K}$ *obstaja šifrirna funkcija* $e_K \in \mathcal{E}$ *in ustrezna dešifrirna funkcija* $d_K \in \mathcal{D}$. *Za funkciji* $e_K : \mathcal{P} \longrightarrow \mathcal{C}$ *in* $d_K : \mathcal{C} \longrightarrow \mathcal{P}$ *velja:*
$$d_K(e_K(x)) = x \quad \text{za vsak} \quad x \in \mathcal{P}.$$

Najprej se Anže in Boštjan domenita za nek skrivni ključ $K \in \mathcal{K}$. To storita, ne da bi kdorkoli izvedel za to (bodisi sta sama v istem prostoru ali pa komunicirata preko nekega varnega kanala). Kasneje, ko želi Anže poslati Boštjanu nek čistopis $x \in \mathcal{P}$, ga najprej zašifrira s šifrirno funkcijo $e_K \in \mathcal{E}$, da dobi tajnopis $y = e_K(x) \in \mathcal{C}$. Tajnopis pošlje Boštjanu. Ker ta pozna ključ $K \in \mathcal{K}$, lahko tajnopis dešifrira z dešifrirno funkcijo d_K in tako dobi čistopis $x = d_K(y) \in \mathcal{P}$.
Šifrirna funkcija e_K mora biti injektivna. Če bi namreč obstajala $x_1 \in \mathcal{P}$ in $x_2 \in \mathcal{P}$ taka, da $x_1 \neq x_2$ in $e_K(x_1) = e_K(x_2) = y$, potem Boštjan ne bi mogel vedeti ali naj tajnopis y dešifrira v čistopis x_1 ali v x_2.
V tem razdelku bomo govorili o različnih kriptosistemih, bolj podrobno pa se bomo posvetili digitalnemu podpisu. Ko bomo poskušali oceniti uporabnost nekega kriptosistema, bomo to storili po naslednjih kriterijih [7].

1. *Stopnja varnosti.* Ponavadi jo podamo kot število operacij , ki jih potrebuje najboljši možni algoritem, da razkrije skrite informacije.
2. *Težavnost implementacije.* Zaželeno je, da se ti algoritmi čim lažje implementirajo. To pomeni, da imajo čim nižjo tako časovno kot tudi prostorsko zahtevnost.
3. *Splošni vidiki uporabnosti.* To je kriterij, ki je specifičen glede na vrsto in namembnost kriptosistema. V splošnem od nekega kriptosistema pričakujemo, da bo kar najmanj posegal v vsakodnevno življenje (recimo, če želim podpisati nek dokument v elektronski obliki, pričakujem, da to ni nič težje oziroma še lažje storiti kot podpisati ročno nek dokument).

Kriptosisteme delimo na dva glavna razreda.

1. **Klasični oz. simetrični kriptosistemi** Njihova glavna značilnost je ta, da se morata Anže in Boštjan vnaprej dogovoriti za neki skrivni ključ, s katerim bo Anže zašifriral, Boštjan pa odšifriral sporočilo. Najbolj značilen predstavnik simetričnih kriptosistemov je DES (Data Encryption Standard). Več o kriptosistemu DES si lahko ogledate v [6] ali [7].

   Ker pa lahko Anže in Boštjan živita na različnih koncih sveta in ker nimata vedno na voljo varnega kanala, da bi se domenila za skrivni ključ, je nastala še druga vrsta kriptosistemov.

2. **Kriptosistemi z javnimi ključi** Pri teh kriptosistemih si samo Boštjan izbere svoj skrivni ključ in izračuna odgovarjajoči javni ključ, ki ga pošlje Anžetu. Anže potem zašifrira sporočilo, to je izračuna tajnopis $e_K(x)$ glede na javni ključ K in ga pošlje Boštjanu. Ta je edini, ki pozna svoj skrivni ključ, je edini, ki pozna d_K, torej je edini, ki lahko prebere sporočilo, to je izračuna čistopis $x = d_K(y)$. Najbolj znan primer kriptosostema z javnimi ključi je kriptosistem RSA (imenovan po avtorjih Rivest-Shamir-Adleman). Več o kriptosistemu RSA si preberite v [6] ali [7].

Mi se bomo bolj posvetili kriptosistemom z javnimi ključi, ki temeljijo na problemu diskretnega logaritma, o katerem smo govorili v prejšnjem poglavju. Najprej si oglejmo ElGamalov kriptosistem.

ELGAMALOV KRIPTOSISTEM
Naj bo p tako praštevilo, da je problem diskretnega logaritma v $\mathbb{Z}_p^*$ težko rešljiv in naj bo $\alpha \in \mathbb{Z}_p^*$ primitiven element. Naj bo $\mathcal{P} = \mathbb{Z}_p^*$, $\mathcal{C} = \mathbb{Z}_p^* \times \mathbb{Z}_p^*$ in definiramo:
$\mathcal{K} = \{(p, \alpha, a, \beta)\, ;\, \beta = \alpha^a \bmod p\}$.
Vrednosti (p, α, β) so javne in so Boštjanov javni ključ, ki ga Boštjan pošlje Anžetu.
Vrednost (a) je skrivna in je Boštjanov tajni ključ.
Da bi Anže zašifriral sporočilo x, naredi naslednje korake.

1. Izbere naključno celo število $k \in [1, p-2]$.
2. Zašifrira čistopis x glede na Boštjanov javni ključ (p, α, β)
   s šifrirno funkcijo $e_{(p,\alpha,\beta)}$, to pomeni, izračuna $e_{(p,\alpha,\beta)}(x) = (y_1, y_2)$,
   kjer $y_1 = \alpha^k \bmod p$ in $y_2 = x\, \beta^k \bmod p$.
3. Boštjanu pošlje tajnopis (y_1, y_2).

Da bi Boštjan dešifriral prejeti tajnopis, naredi naslednje.

1. S pomočjo svojega tajnega ključa (a) izračuna $d_{(a)}(y_1, y_2) = y_2\, (y_1^a)^{-1} \bmod p$.
2. Dobljena vrednost je Anžetovo sporočilo.

Ker je Boštjan edini, ki pozna svoj tajni ključ (a), je edini, ki lahko dešifrira Anžetovo sporočilo.

Prepričajmo se, da je rezultat dešifrirne funkcije resnično začetni čistopis, torej, da velja: $d_K(e_K(x)) = x$.

$$d_K(e_K(x)) = d_{(a)}(e_{(p,\alpha,\beta)}(x)) = y_2\,(y_1^a)^{-1} \bmod p = x\,\beta^k\,(\alpha^{ka})^{-1} \bmod p =$$
$$= x\,\beta^k\,(\beta^k)^{-1} \bmod p = x \bmod p = x$$

Varnost tega kriptosistema temelji na težavnosti problema diskretnega logaritma. V prejšnjem poglavju smo videli, da je problem težko rešljiv, če le vzamemo pravi p, to je p mora biti tako praštevilo, da ima $p-1$ v svoji faktorizaciji neko veliko praštevilo q, recimo $q \approx 2^{160}$. Za podrobnejše informacije o ElGamalovem kriptosistemu priporočam knjigi [6] ali [7]. Zdaj pa si poglejmo njegovo posplošitev. Kot smo posplošili problem diskretnega logaritma v grupi $\mathbb{Z}_p^*$ na poljubno končno grupo G, tako lahko posplošimo tudi ElGamalov kriptosistem na poljubno končno grupo G. Hkrati pa lahko, namesto, da računamo v celi grupi G, računamo zgolj v neko podgrupi H grupe G.

POSPLOŠENI ELGAMALOV KRIPTOSISTEM
Naj bo G končna grupa in naj bo $\alpha \in G$ točka reda n, α naj bo generator podgrupe H grupe G. Podgrupa $H = \{\alpha^i\,;\, i = 0,\dots,n-1\}$ naj bo taka, da je problem diskretnega logaritma v njej težko rešljiv. Naj bo $\mathcal{P} = G$, $\mathcal{C} = G \times G$ in definiramo:
$\mathcal{K} = \{(G, \alpha, a, \beta) \quad ; \quad \beta = \alpha^a\}$.
Vrednosti (G, α, β) so javne, vrednost (a) pa je tajna.
Za $K = (G, \alpha, a, \beta) \in \mathcal{K}$ in naključno celo število $k \in [1, n-1]$ in za čistopis $x \in \mathcal{P}$ definiramo:
$e_K(x, k) = (y_1, y_2)$, kjer $y_1 = \alpha^k$ in $y_2 = x \circ \beta^k$.
Za tajnopis (y_1, y_2) pa definiramo: $d_K(y_1, y_2) = y_2 \circ (y_1^a)^{-1}$.

Kot bomo videli v naslednjem razdelku, v razdelku o digitalnih podpisih, je kriptosistem, kjer računske operacije potekajo v grupi eliptične krivulje $E(\mathbb{F}_{2^m})$, boljši vsaj zaradi dveh razlogov. Prvi je ta, da je računalniška implementacija mnogo hitrejša in bolj enostavna. Drugi razlog pa je, da je problem diskretnega logaritma v grupi eliptične krivulje v splošnem težji od problema diskretnega logaritma v grupi števil. Več o tem si bomo pogledali v naslednjem razdelku. Torej zdaj namesto splošne grupe G uporabimo kar grupo eliptične krivulje $E(\mathbb{F}_{2^m})$.

ELGAMALOV KRIPTOSISTEM V GRUPI ELIPTIČNE KRIVULJE
Naj bo $E(\mathbb{F}_{2^m})$ končna grupa eliptične krivulje in naj bo $P \in E(\mathbb{F}_{2^m})$ točka reda n, P naj bo generator podgrupe H grupe G. Podgrupa $H = \{i \cdot P; i = 0,\dots,n-1\}$ naj bo taka, da je problem diskretnega logaritma v njej težko rešljiv. Naj bo
$\mathcal{P} = E(\mathbb{F}_{2^m})$, $\mathcal{C} = E(\mathbb{F}_{2^m}) \times E(\mathbb{F}_{2^m})$ in definiramo :
$\mathcal{K} = \{(E(\mathbb{F}_{2^m}), P, d, Q) \quad ; \quad Q = d \cdot P\}$.
Vrednosti $(E(\mathbb{F}_{2^m}), P, Q)$ so javne, vrednost (d) pa je tajna.
Za $K = (E(\mathbb{F}_{2^m}), P, d, Q) \in \mathcal{K}$ in naključno celo število $k \in [1, n-1]$ in za čistopis $x \in \mathcal{P}$ definiramo:
$e_K(x, k) = (y_1, y_2)$, kjer $y_1 = k \cdot P$ in $y_2 = x + k \cdot Q$.
Za tajnopis (y_1, y_2) pa definiramo: $d_K(y_1, y_2) = y_2 - (d \cdot y_1)$.

Varnost tega kriptosistema temelji na težavnosti problema diskretnega logaritma v grupi eliptične krivulje $E(\mathbb{F}_{2^m})$. V naslednjem razdelku, ko bomo govorili o algoritmu digitalnega podpisa v grupi števil $\mathbb{Z}_p^*$ in v grupi eliptične krivulje $E(\mathbb{F}_{2^m})$, si bomo podrobno pogledali primerjavo velikosti obeh grup, ob predpostavki, da sta tako problem DLP kot problem ECDLP enako težka (glej tabelo 3.1).

3.2 DIGITALNI PODPIS

V tem razdelku bom predstavila digitalni podpis. Kot smo že omenili v uvodnem delu tega poglavja, je digitalni podpis metoda podpisovanja dokumentov, ki so v elektronski obliki. Povedali smo tudi, da mora digitalni podpis imeti vse lastnosti, ki veljajo za ročni podpis (možnost identifikacije podpisnika, digitalni podpis se nanaša na nek dokument oz. je njegov del itd.). Potem pa smo poudarili, da , poleg naštetih lastnosti, za digitalni podpis velja še neka nova zelo pomembna lastnost, tj. da vsebine podpisanega dokumenta ni več možno spreminjati. Četudi skuša digitalni podpis posnemati ročnega, je med obema oblikama podpisovanja še vedno nekaj bistvenih razlik.

- Prva je vprašanje podpisovanja dokumenta. V primeru ročnega podpisa je ta podpis fizično del dokumenta, ki ga podpisujemo. V primeru digitalnega podpisa pa temu ni tako, torej mora algoritem, s katerim bomo izračunali digitalni podpis, le-tega nekako povezati z dokumentom. To bomo storili tako, da bo digitalni podpis neposredno odvisen od vsebine dokumenta. Tako bomo preprečili ne samo spreminjanje vsebine dokumenta, ampak tudi možnost, da bi se dalo podpis uporabiti tudi na nekem drugem dokumentu.

- Druga razlika je vprašanje preverjanja. Običajni podpis preverimo s primerjanjem z nekim drugim podpisom, za katerega vemo, da je pristen. Tako preverjanje je precej nezanesljivo in tudi zamudno, saj vsi vemo, da večina blagajničarjev ne primerja podpisa na kartici s podpisom na računu, pa bi to seveda morali. Digitalne podpise bomo preverjali z nekim javno znanim algoritmom preverjanja. Torej bo vsakdo lahko preveril pristnost mojega digitalnega podpisa.

- Tretja pomembna razlika pa je ta, da je kopija podpisanega elektronskega sporočila popolnoma enaka originalu. Pri običajnem podpisu kopijo sporočila v večini primerov lahko ločimo od izvirnika. Torej moramo tudi v primeru digitalnega podpisa znati preprečiti ponovno uporabo podpisanega sporočila. Recimo, če Boštjan pooblasti Anžeta, da lahko dvigne 10 000 SIT z njegovega računa, pričakuje, da bo Anže to lahko storil le enkrat. Zato mora sporočilo samo vsebovati dovolj podatkov, recimo datum in uro, ki bodo preprečili ponovno uporabo.

Pogledali si bomo tri različne algoritme za izračun digitalnega podpisa, njihove prednosti in pomanjkljivosti. Prvi algoritem je splošno sprejeti algoritem, ki ga definiramo v grupi števil $\mathbb{Z}_p^*$. Drugi algoritem je različica prvega, le da vse operacije potekajo v grupi eliptične krivulje. Ker so operacije v grupi eliptične krivulje v praksi hitrejše od onih v grupi števil, je prednost drugega algoritma predvsem v večji hitrosti. Tretji algoritem pa je nekakšna modifikacija drugega. Je ravno tako hiter, vendar pa je dobljeni digitalni podpis precej krajši od digitalnega podpisa, izračunanega s prvima dvema algoritmoma. Slaba stran zadnjega algoritma pa je, da lahko z njim podpisujemo le tista sporočila, katerih vsebina je vsaj okvirno poznana prejemniku. Recimo, če Anže pričakuje, da mu bo Boštjan poslal neko programsko kodo, bo tako sporočilo znal prebrati, ker bo vedel, da je sestavljeno iz končno možnih stavkov nekega programskega jezika. Za naključna sporočila pa ta algoritem žal ni primeren.

3.2.1 DEFINICIJE

Algoritem digitalnega podpisa sestavljajo trije deli:

- algoritem generiranja ključa,
- algoritem generiranja digitalnega podpisa in
- algoritem preverjanja digitalnega podpisa.

Najprej si Boštjan izbere svoj tajni ključ ter sporoči Anžetu ustrezni javni ključ. Da bi podpisal sporočilo x, Boštjan uporabi skrivni algoritem generiranja digitalnega podpisa in izračuna digitalni podpis $sig_B(x)$. Nato pošlje sporočilo x skupaj z digitalnim podpisom Anžetu. Ker ta pozna Boštjanov javni ključ, lahko z uporabo javnega algoritma preverjanja digitalnega podpisa preveri pristnost podpisa.
Naj na tem mestu opomnim bralca, da ko govorim o skrivnem algoritmu, to nikakor ne pomeni, da je sam algoritem skriven, temveč zgolj to, da ga nihče razen Boštjana ne more izračunati, ker ne pozna njegovega tajnega ključa.

Zdaj pa še formalno definirajmo algoritem digitalnega podpisa [6]:

Definicija 3.2 *Algoritem digitalnega podpisa je peterica* $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ *, ki zadošča naslednjim zahtevam.*

1. $\mathcal{P}$ *je končna množica možnih sporočil.*
2. $\mathcal{A}$ *je končna množica možnih podpisov.*
3. $\mathcal{K}$ *je končna množica možnih ključev.*
4. *Za vsak* $K \in \mathcal{K}$ *obstaja algoritem generiranja digitalnega podpisa* $sig_K \in \mathcal{S}$ *in ustrezni algoritem preverjanja digitalnega podpisa* $ver_K \in \mathcal{V}$*. Funkciji* $sig_K : \mathcal{P} \longrightarrow \mathcal{A}$ *in* $ver_K : \mathcal{P} \times \mathcal{A} \longrightarrow \{veljaven,\ neveljaven\}$ *zadoščata pogoju, da za vsako sporočilo* $x \in \mathcal{P}$ *in za vsak podpis* $y \in \mathcal{A}$ *velja:*

$$ver(x,y) = \begin{cases} veljaven, & \text{če} \quad y = sig(x) \\ neveljaven, & \text{če} \quad y \neq sig(x). \end{cases}$$

Za vsak $K \in \mathcal{K}$ morata biti funkciji sig_K in ver_K izračunljivi v polinomskem času. Funkcija ver_K je javna funkcija, sig_K pa skrivna.

Poglejmo si dve zahtevi, ki morata biti izpolnjeni, da lahko govorimo o varnem algoritmu.

1. Boštjan pričakuje, da je on edini, ki lahko podpiše sporočilo s svojim podpisom, torej je edini, ki zna pri danem x izračunati y tako, da velja $ver_{K_B}(x, y)$ =veljaven.

2. Anže pa pričakuje, da se bo sposoben popolnoma prepričati, ali gre za Boštjanov podpis ali pa gre za nek ponaredek.

Kadar bosta ti dve zahtevi izpolnjeni, bomo lahko rekli, da je algoritem digitalnega podpisa *varen.* Seveda algoritem nikoli ne more biti brezpogojno varen. Naš nasprotnik namreč lahko testira vse možne podpise y za sporočilo x z uporabo javnega algoritma preverjanja ver_{K_B}, dokler ne najde pravega. Torej lahko, če ima dovolj časa, vedno ponaredi Boštjanov podpis. Naš cilj je poiskati algoritem digitalnega podpisa, ki bo računsko čim bolj varen.

Algoritme digitalnega podpisa delimo po [7] na:

Algoritme digitalnega podpisa z dodatkom: to so algoritmi, pri katerih v procesu preverjanja potrebujemo originalno sporočilo. Taka algoritma sta denimo algoritma DSA in ECDSA, ki si ju bomo ogledali v naslednjih dveh razdelkih.

Algoritme digitalnega podpisa s povračilom sporočila: to pa so algoritmi, pri katerih v procesu preverjanja ne potrebujemo originalnega sporočila. Primer takega algoritma je algoritem PVSSR, ki si ga bomo ogledali v razdelku 3.2.4.

V vseh treh algoritmih bomo sporočilo, ki ga bomo podpisovali, najprej preslikali z *zgoščevalno funkcijo.* Glavna lastnost te funkcije je, da nam sporočilo poljubne dolžine preslika v sporočilo točno določene dolžine (pri nas bo to 160 bitov). Bolj podrobno pa si bomo zgoščevalne funkcije in njihov pomen za varnost algoritma pogledali v razdelku 3.3.

3.2.2 ALGORITEM DSA

Algoritem digitalnega podpisa DSA (The Digital Signature Algorithm) je bil predlagan avgusta 1991 na inštitutu NIST (U.S. National Institute of Standards and Technology) in je bil leta 1993 sprejet kot standard digitalnega podpisovanja v Združenih državah Amerike (FIPS 186), več podrobnosti je na voljo v [7]. Ta algoritem je različica ElGamalove metode digitalnega podpisa. Bistvena razlika med ElGamalovim algoritmom in DSA je, da v DSA uporabljamo manjše podgrupe $\mathbb{Z}_p^*$ in s tem zmanjšamo dolžino podpisa. Pa si kar poglejmo sam algoritem.

GENERIRANJE KLJUČA Z DSA:

Boštjan si izbere svoj privatni in javni ključ:

1. Izbere praštevilo q tako, da $2^{159} < q < 2^{160}$.
2. Izbere praštevilo p tako, da $p \approx 2^{1024}$ in velja $q|(p-1)$.
3. V $\mathbb{Z}_p^*$ izbere ciklično podgrupo H reda q:
   ponavljaj

   izberi naključni $h \in \mathbb{Z}_p^*$,

   izračunaj $g = h^{\frac{p-1}{q}}$,

   dokler $g \neq 1$.

   g je generator enolične ciklične podgrupe H reda q v $\mathbb{Z}_p^*$.
4. Izbere naključno celo število $x \in [1, q-1]$.
5. Izračuna $y = g^x \bmod p$.
6. (p, q, g, y) je Boštjanov javni ključ, (x) pa Boštjanov tajni ključ.

GENERIRANJE PODPISA Z DSA:

Da bi podpisal sporočilo m, Boštjan naredi:

1. Izbere naključno celo število $k \in [1, q-1]$.
2. Izračuna $r = (g^k \bmod p) \bmod q$.
3. Izračuna $k^{-1} \bmod q$.
4. Izračuna $s = k^{-1}(H(m) + xr) \bmod q$, kjer je H zgoščevalna funkcija recimo SHA-1.
5. Če $s = 0$, pojdi na 1. korak. (Če $s = 0$, potem $s^{-1} \bmod q$ ne obstaja, s^{-1} pa potrebujem v algoritmu preverjanja.)
6. Podpis sporočila m je par (r, s).

PREVERJANJE PODPISA Z DSA:

Če želi Anže preveriti Boštjanov podpis (r, s) na sporočilu m, naredi:

1. Dobi pristno kopijo Boštjanovega javnega ključa (p, q, g, y).
2. Izračuna $w = s^{-1} \bmod q$ in $H(m)$, kjer je H zgoščevalna funkcija recimo SHA-1.
3. Izračuna $u_1 = H(m)w \bmod q$ in $u_2 = rw \bmod q$.
4. Izračuna $v = (g^{u_1} y^{u_2} \bmod p) \bmod q$.
5. Podpis je veljaven natanko tedaj, ko velja $v = r$.

Ker sta števili r in s obe manjši od q, katerega velikost je največ 160 bitov, je podpis (r, s) dolg največ 320 bitov.

Varnost DSA temelji na dveh različnih, a povezanih problemih diskretnega logaritma. Prvi se nanaša na grupo $\mathbb{Z}_p^*$ in preprečuje odkritje Boštjanovega tajnega ključa, drugi pa deluje v podgrupi H reda q v $\mathbb{Z}_p^*$ in preprečuje, da bi kdorkoli ponaredil podpis na sporočilu m. Ker sta števili p in q dovolj veliki, veljata za dovolj varno izbiro. Do sedaj znani algoritmi, ki rešujejo problem diskretnega logaritma, so namreč pri tako veliki izbiri števil p in q prepočasni. Najboljša do sedaj znana metoda za reševanje problema diskretnega logaritma v $\mathbb{Z}_p^*$ je index calculus metoda, ki za rešitev posameznega problema diskretnega logaritma potrebuje približno $\mathcal{O}(\mathrm{e}^{(\frac{1}{2}+o(1))\sqrt{\ln p \ln\ln p}})$ korakov. Ker je p 1024-bitno število, algoritem DSA ni občutljiv na te vrste napad. Za rešitev problema diskretnega logaritma v podgrupi H pa bi bila primerna Pollardova ρ-metoda, ki bi potrebovala v povprečju $\mathcal{O}(\sqrt{q})$ operacij. Ker je $q \approx 2^{160}$, algoritem DSA ni ranljiv na te vrste napad, za podrobnejše informacije glejte [7]. Zato algoritem DSA ob taki izbiri p in q velja za varen algoritem.

3.2.3 ALGORITEM ECDSA

Algoritem digitalnega podpisa v grupi eliptične krivulje ECDSA (The Elliptic Curve Digital Signature Algorithm) je analogija DSA, le da vse operacije izvajamo namesto v grupi števil $\mathbb{Z}_p^*$ v grupi eliptične krivulje. V razdelku 2.2, kjer smo govorili o računalniški implementaciji, smo omenili nekatere prednosti, ki jih ima grupa eliptične krivulje $E(\mathbb{F}_{2^m})$ v primerjavi z grupo eliptične krivulje $E(\mathbb{Z}_p)$, zato bomo v algoritmu ECDSA seveda uporabili grupo $E(\mathbb{F}_{2^m})$. Poglejmo si torej algoritem.

GENERIRANJE KLJUČA Z ECDSA:

Boštjan si izbere svoj privatni in javni ključ:

1. Izbere praštevilo n tako, da $n \approx 2^{160}$.
2. Izbere grupo eliptične krivulje $E(\mathbb{F}_{2^m})$ tako, da $|E(\mathbb{F}_{2^m})| = N$ in velja $n|N$.
3. V $E(\mathbb{F}_{2^m})$ izbere ciklično podgrupo H reda n:
   ponavljaj
       izberi naključni $S \in E(\mathbb{F}_{2^m})$,
       izračunaj $P = NS/n$
   dokler $P \neq \mathcal{O}$.
   P je generator enolične cilkične podgrupe H reda n v $E(\mathbb{F}_{2^m})$.
4. Izbere naključno celo število $d \in [2, n-2]$.
5. Izračuna $Q = dP$.
6. $(E(\mathbb{F}_{2^m}), n, P, Q)$ je Boštjanov javni ključ, (d) pa Boštjanov tajni ključ.

GENERIRANJE PODPISA Z ECDSA:

Da bi podpisal sporočilo m, Boštjan naredi:

1. Izbere naključno celo število $k \in [2, n-2]$.
2. Izračuna $kP = (x_1, y_1)$ in $r = x_1 \bmod n$.
   Če $r = 0$, pojdi na 1. korak. (Če je namreč $r = 0$, potem enačba
   $s = k^{-1}(H(m) + dr) \bmod n$ ni odvisna od Boštjanovega privatnega ključa d.)
3. Izračuna $k^{-1} \bmod n$.
4. Izračuna $s = k^{-1}(H(m) + dr) \bmod n$, kjer je H zgoščevalna funkcija recimo SHA-1.
5. Če $s = 0$, pojdi na 1.korak.(Če $s = 0$, potem $s^{-1} \bmod n$ ne obstaja, s^{-1} pa potrebujem v algoritmu preverjanja.)
6. Podpis sporočila m je par (r, s).

PREVERJANJE PODPISA Z ECDSA:

Če želi Anže preveriti Boštjanov podpis (r, s) na sporočilu m, naredi:

1. Dobi pristno kopijo Boštjanovega javnega ključa $(E(\mathbb{F}_{2^m}), n, P, Q)$.
2. Izračuna $w = s^{-1} \bmod n$ in $H(m)$, kjer je H zgoščevalna funkcija recimo SHA-1.
3. Izračuna $u_1 = H(m)w \bmod n$ in $u_2 = rw \bmod n$.
4. Izračuna $u_1P + u_2Q = (x_0, y_0)$ in $v = x_0 \bmod n$.
5. Podpis je veljaven natanko tedaj, ko velja $v = r$.

Kot lahko vidimo, sta algoritma DSA in ECDSA skoraj popolnoma enaka, le da prvi deluje znotraj grupe $\mathbb{Z}_p^*$, drugi pa znotraj grupe $E(\mathbb{F}_{2^m})$.
Varnost ECDSA seveda spet temelji na težavnosti problema diskretnega logaritma, vendar tokrat v grupi eliptične krivulje. Splošno prepričanje je, da je ECDLP veliko težje rešljiv kot problem diskretnega logaritma v $\mathbb{Z}_p^*$, če seveda uporabim primerno veliki grupi. Čas reševanja, ki ga bo porabil trenutno najboljši možni algoritem za reševanje problema diskretnega logaritma v grupi $\mathbb{Z}_p^*$, kjer je p 512-bitno število, je približno enak času, ki ga bo porabil trenutno najboljši možni algoritem za reševanje problema diskretnega logaritma v grupi $E(\mathbb{F}_{2^m})$, kjer je m 110-bitno število. Ko p povečujemo, se razlika v velikosti m in p še povečuje, pri recimo 2048-bitnem p je ustrezni m dolg samo 210 bitov, to pa je že skoraj desetkrat manj. Tabela 3.1 prikazuje ustrezne vrednosti parametrov p in m. [2]

m	p	$\lfloor p/m \rfloor$
110	512	5
130	768	6
160	1024	7
210	2048	10

Tabela 3.1: Primerjava velikosti števil m in p, če je problem DLP in enako zahteven kot problem ECDLP

Da bi torej obdržali isto stopnjo varnosti kot v DSA (z 1024 bitnim p in 160 bitnim q), mora m imeti vsaj 160 bitov in prav tako n. V tem primeru sta velikosti samega podpisa z DSA in ECDSA enaki (320 bitov). Vendar pa je ECDSA bolj uporaben algoritem vsaj zaradi naslednjih razlogov:

- Manjša dolžina ključev [3]:
  V praksi si ponavadi že prej izberemo neko grupo eliptične krivulje $E(\mathbb{F}_{2^m})$ in neko točko P reda n. V tem primeru je Boštjanov javni ključ sestavljen samo iz točke Q na krivulji. To točko lahko (v našem primeru je m 160 bitno število) predstavimo z 161 bitnim zaporedjem, če uporabimo tehniko kompresije točk, ki je opisana v 2.poglavju. Dolžina javnih ključev je pomembna zato, ker so le-ti običajno del certifikatov, za katere je pomembno, da so čim krajši.
- Hitrejša implementacija [3]:
  Računske operacije v $E(\mathbb{F}_{2^m})$ so običajno hitrejše od tistih v $\mathbb{Z}_p^*$. Recimo množenje v $E(\mathbb{F}_{2^m})$ (recimo izračun $k \cdot P$, kjer je $P \in E(\mathbb{F}_{2^m})$, $E(\mathbb{F}_{2^m})$ je grupa eliptične krivulje nad $\mathbb{F}_{2^m}$, m in k pa sta 160 bitni števili) je približno 8-krat hitrejše od potenciranja v $\mathbb{Z}_p^*$ (recimo izračun $g^k \bmod p$, kjer je p 1024 bitno število in k 160 bitno število).

3.2.4 ALGORITEM S POVRAČILOM SPOROČILA

V tem razdelku bom opisala algoritem digitalnega podpisa, ki ima to lastnost, da lahko sporočilo v procesu preverjanja razberemo iz digitalnega podpisa samega. Pri tem bomo govorili o *algoritmu digitalnega podpisa s povračilom sporočila*, kjer lahko razberemo iz digitalnega podpisa celotno sporočilo ali pa o *algoritmu digitalnega podpisa z* **delnim** *povračilom sporočila,* kjer bomo lahko iz digitalnega podpisa razbrali samo nek določen del sporočila.
Če si želimo bolj podrobno pogledati algoritem digitalnega podpisa z delnim ali celotnim povračilom sporočila, moramo znani petorki $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ iz definicije digitalnega podpisa (razdelek 3.2.1) dodati še naslednji novosti, glej [7].

- $\mathcal{P}_{\mathcal{S}}$ je prostor podpisovanja na katerem deluje funkcija $sig_K : \mathcal{P}_{\mathcal{S}} \longrightarrow \mathcal{S}$.
- Tr *funkcijo dodatka*, ki bo slikala iz prostora sporočil v prostor podpisovanja, $Tr : \mathcal{P} \longrightarrow \mathcal{P}_{\mathcal{S}}$, sliko Tr pa bomo označili $\mathcal{P}_{\mathcal{R}}$.

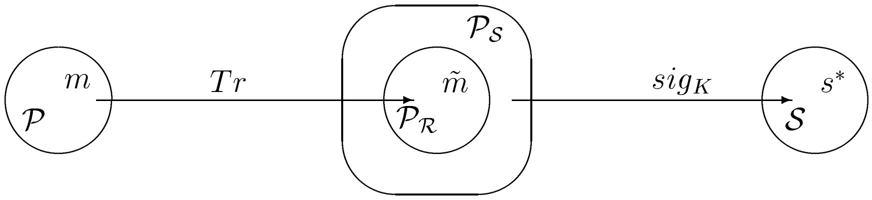
Algoritem pa bo izgledal takole [7].

Boštjan izbere ključ $K \in \mathcal{K}$, s katerim podpiše sporočilo m na naslednji način:

1. Izračuna $\tilde{m} = Tr(m)$.
2. Izračuna $s^* = sig_K(\tilde{m})$.
3. Digitalni podpis sporočila m je s^*. Boštjan pošlje Anžetu vrednost s^*.

Anže preveri pristnost digitalnega podpisa in prebere sporočilo na naslednji način:

1. Dobi javno znano funkcijo preverjanja glede na Boštjanov javni ključ ver_K in pozna funkcijo Tr, torej tudi $\mathcal{P}_{\mathcal{R}}$.
2. Izračuna $\tilde{m} = ver_K(s^*)$.
3. Če velja $\tilde{m} \notin \mathcal{P}_{\mathcal{R}}$, potem je podpis neveljaven. Če pa velja $\tilde{m} \in \mathcal{P}_{\mathcal{R}}$, potem je podpis veljaven in $m = Tr^{-1}(\tilde{m})$ je Boštjanovo sporočilo.



Slika 3.2: Generiranje podpisa z algoritmom digitalnega podpisa s povračilom sporočila

Funkcija dodatka je javno znana funkcija, vendar pa je za varnost tega algoritma zelo pomembno, katero funkcijo si bomo izbrali. Poglejmo si primer, ki je predstavljen v [7].

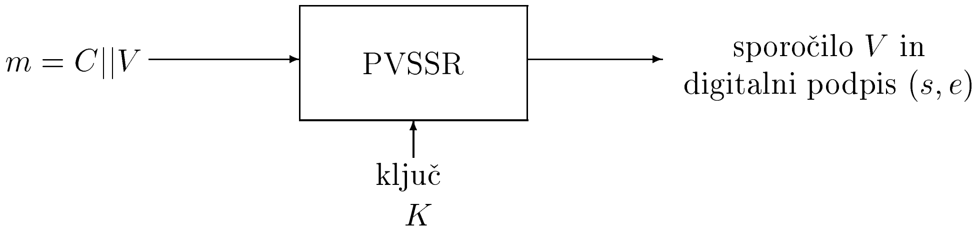
Primer 3.3:
Recimo, da je $\mathcal{P}_{\mathcal{R}} = \mathcal{P}_{\mathcal{S}}$ in sta Tr in sig_K bijektivni funkciji, $Tr : \mathcal{P} \longrightarrow \mathcal{P}_{\mathcal{R}} = \mathcal{P}_{\mathcal{S}}$ in $sig_K : \mathcal{P}_{\mathcal{R}} = \mathcal{P}_{\mathcal{S}} \longrightarrow \mathcal{S}$. To pomeni, da imata $\mathcal{P}$ in $\mathcal{S}$ enako število elementov. Potem je za vsak $s^* \in \mathcal{S}$ neki $ver_K(s^*) = \tilde{m} \in \mathcal{P}_{\mathcal{R}}$ in trivialno je poiskati tako sporočilo m, ki ustreza digitalnemu podpisu s^*. Najdemo ga po naslednjem postopku.

1. Izberi naključni $s^* \in \mathcal{S}$ in $ver_K \in \mathcal{V}$.
2. Izračunaj $\tilde{m} = ver_K(s^*)$.
3. Izračunaj $m = Tr^{-1}(\tilde{m})$.

Podpis s^* je veljaven podpis na sporočilu m, ki smo ga izračunali brez poznavanja skrivnega algoritma sig_K. Torej smo ugotovili, da v takem primeru algoritem seveda nima primerne stopnje varnosti, zato bomo morali paziti, kakšno funkcijo bomo izbrali za funkcijo dodatka. V naslednjem razdelku pa bomo pogledali še konkreten primer algoritma z delnim povračilom sporočila.

ALGORITEM DIGITALNEGA PODPISA Z DELNIM POVRAČILOM SPOROČILA - PVSSR

Algoritem digitalnega podpisa z delnim povračilom sporočila PVSSR (The Pintsov-Vanstone Signature Scheme with Partial Message Recovery) je različica Schnorr-Nyberg-Rueppel-ovega algoritma digitalnega podpisa [1]. Algoritem PVSSR je algoritem z *delnim* povračilom sporočila, kar pomeni, da bomo v procesu preverjanja iz digitalnega podpisa mogli razbrati samo *del* sporočila. Sporočilo m, ki ga Boštjan želi poslati Anžetu, bo razdelil na dva dela $m = C||V$, kjer $||$ pomeni operacijo pripojitve zaporedja bitov C na zaporedje bitov V. Pri procesu generiranja podpisa z algoritmom PVSSR bo za izračun digitalnega podpisa uporabil samo del C. V procesu preverjanja bo Anže iz digitalnega podpisa mogel razbrati samo del C. Del V bo Boštjan seveda moral poslati Anžetu skupaj z digitalnim podpisom.

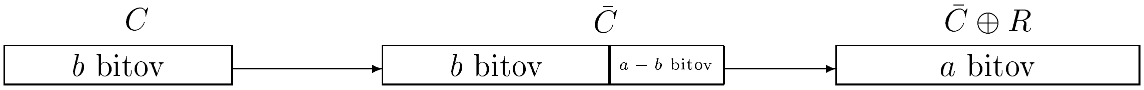


Slika 3.4: Prikaz generiranja podpisa z algoritmom PVSSR

Za funkcijo dodatka bomo vzeli naslednjo funkcijo Tr. Recimo, da je del C dolg a bitov in recimo, da je Boštjanov javni ključ dolg b bitov. Predpostavimo še, da si Boštjan izbere neko skrivno zaporedje bitov, ki je prav tako dolžine b. To skrivno zaporedje imenujmo zaporedje R. Funkcija Tr je sestavljena iz naslednjih dveh operacij.

1. Iz bitnega zaporedja C tvorimo novo bitno zaporedje $\bar{C}$ dolžine b tako, da bitnemu zaporedju C dodamo naključnih $a - b$ bitov.

2. Izračunamo $\bar{C} \oplus R$, kjer $\oplus$ pomeni operacijo ekskluzivni ali (XOR).



Slika 3.5: Prikaz delovanja funkcije Tr

Pri algoritmu PVSSR bomo v postopku preverjanja sprejeli za veljavne samo tiste digitalne podpise, pri katerih bodo iz njih razbrana sporočila pripadala neki vnaprej določeni podmnožici množice vseh sporočil, ki jih je mogoče razbrati iz našega digitalnega podpisa. Recimo, da za to podmnožico vzamemo množico zaporedij, ki "kaj pomenijo". V splošnem je težko preveriti, ali sporočilo "kaj pomeni" ali ne. Lažje je, če vemo, kaj pričakovati, recimo, če vemo, da je sporočilo nek naslov ali da je ukazna vrstica v nekem programskem jeziku itd. Zato se algoritem PVSSR uporablja predvsem pri tisti vrsti sporočil, kjer nam je že vnaprej znana vsaj okvirna vsebina. Za potrebe digitalne poštne znamke, kjer vnaprej vemo, katere podatke bomo podpisovali (datum, poštnina, itd.), je ta algoritem kot na kožo pisan. V našem primeru digitalne poštne znamke bomo zahtevali, naj del C pripada podmnožici $\mathcal{Z}$, moči 2^a, množice vseh bitnih zaporedij dolžine b. Množica $\mathcal{Z}$ naj bo množica vseh takih zaporedij, ki imajo na prvih a_1 bitih zapisan recimo naslov, na naslednjih a_2 bitih recimo vrednost poštnine itd. Ker mora del C imeti na prvih a bitih neke točno določene vrednosti, naslednjih $a - b$ bitov, ki mu jih dodamo, pa je popolnoma

naključnih, rečemo, da ima del C *velikost dodatka* enako $a-b$ bitov.
Zdaj pa si poglejmo algoritem digitalnega podpisa s PVSSR .

GENERIRANJE KLJUČA S PVSSR :

Boštjan si izbere svoj privatni in javni ključ

1. Izbere praštevilo n tako, da $n \approx 2^{160}$.
2. Izbere grupo eliptične krivulje $E(\mathbb{F}_{2^m})$ tako, da $|E(\mathbb{F}_{2^m})| = N$ in velja $n|N$.
3. V $E(\mathbb{F}_{2^m})$ izbere ciklično podgrupo H reda n:
   ponavljaj
   izberi naključni $S \in E(\mathbb{F}_{2^m})$,
   izračunaj $P = NS/n$,
   dokler $P \neq \mathcal{O}$.
   P je generator enolične cilkične podgrupe H reda n v $E(\mathbb{F}_{2^m})$.
4. Izbere naključno celo število $d \in [2, n-2]$.
5. Izračuna $Q = dP$.
6. $(E(\mathbb{F}_{2^m}), n, P, Q)$ je Boštjanov javni ključ, (d) pa Boštjanov tajni ključ.

GENERIRANJE PODPISA S PVSSR :

Da bi podpisal sporočilo m, Boštjan naredi naslednje korake.

1. Razdeli sporočilo m na dva dela $m = C||V$.
2. Izbere naključno celo število $k \in [2, n-2]$.
3. Izračuna $R = kP$.
4. Izračuna $e = Tr_R(C)$, kjer je Tr_R prej opisana funkcija dodatka.
5. Izračuna $h = H(e||V)$, kjer je H zgoščevalna funkcija recimo SHA-1.
6. Izračuna $s = dh + k \bmod n$.
7. Podpis sporočila $m = C||V$ je par (s, e).

Boštjan bo Anžetu torej poslal sporočilo V in digitalni podpis (s, e).

PREVERJANJE PODPISA S PVSSR :
Če želi Anže preveriti Boštjanov podpis (s, e) na sporočilu V in seveda, če želi sploh prebrati celotno sporočilo (to je razbrati del C), naredi naslednje.

1. Dobi pristno kopijo Boštjanovega javnega ključa $(E(\mathbb{F}_{2^m}), n, P, Q)$, pozna množico $\mathcal{Z}$ in funkcijo dodatka Tr.
2. Izračuna $h = H(e||V)$, kjer je H zgoščevalna funkcija recimo SHA-1.
3. Izračuna $U = sP - hQ$.
4. Izračuna $X = Tr_R^{-1}(e)$.
5. Če $X \notin \mathcal{Z}$, potem je podpis neveljaven.
   Če $X \in \mathcal{Z}$, potem postavi $C := X$, podpis je veljaven, sporočilo pa je $m = C||V$.

Zdaj pa si poglejmo, kako velikost dodatka dela C vpliva na varnost tega algoritma. Če hočemo zaščititi Boštjana pred odkritjem njegovega tajnega ključa z isto stopnjo varnosti kot pri ECDSA ali DSA, moramo spet vzeti $m = 160$ in $n = 160$. Tako dobimo 160 bitni ključ, torej $b = 160$. Število $a - b$ je število dodatnih bitov v delu C. Vsak algoritem digitalnega podpisa s povračilom sporočila je občutljiv na ponarejanje te vrste, da lahko ponarejevalec, če si izbere nek podpis (s, e), podpiše neko sporočilo m brez vedenja Boštjanovega tajnega ključa, vendar pa nima nikakršne kontrole nad sporočilom samim (ne more vnaprej določiti vsebine sporočila). To smo opisali v primeru 3.3 .
Da bi takšno vrsto napadov preprečili, moramo v sporočilu (v delu C) uporabiti primerno funkcijo dodatka oz. primerno velikost dodatka $a - b$ bitov [2]. Če recimo pri dolžini ključa 160 bitov uporabimo $a - b = 10$, je tak ponaredek uspešen z verjetnostjo $1/2^{10}$, to je eden v tisoč poskusih. Velikost parametra $a - b$ moramo torej prilagoditi dejanskim potrebam po varnosti. Seveda nam večji parameter zagotavlja večjo stopnjo varnosti, a tudi večjo vrednost b, torej večjo dolžino podpisa. V primeru digitalne poštne znamke bo del C dolg 120 bitov, torej bo velikost dodatka znašala 40 bitov, kar nam zagotavlja stopnjo varnosti 2^{40} [4].

Ker se vse računske operacije vršijo v grupi eliptične krivulje, lahko spet omenimo krajšo dolžino ključev (160 bitov) in hitrejšo implementacijo.

Glavna prednost algoritma PVSSR je krajša dolžina podpisa, le-ta je namreč kar za pol krajša od dolžine podpisa izračunanega z algoritmoma ECDSA ali DSA, saj znaša 160 bitov in še nekaj bitov dodatka. Tukaj moram poudariti, da za dolžino podpisa štejem tiste bite, ki jih podpis prispeva k skupni dolžini sporočila in digitalnega podpisa. Recimo, da želi Boštjan poslati sporočilo m dolžine x. Sporočilo m razdeli na dva dela $m = C||V$, kjer je del C dolg a bitov, del V pa $a - x$ bitov. Z algoritmom PVSSR izračuna digitalni podpis (s, e), kjer je parameter s dolg 160 bitov, parameter e pa b bitov (v našem primeru je to tudi 160 bitov). Boštjan pošlje Anžetu del V in digitalni podpis (s, e), torej $a - x + 320$ bitov, kar je seveda manj kot pri algoritmu ECDSA ali DSA, ko mora Boštjan poslati Anžetu celotno sporočilo m in digitalni podpis (r, s), torej $x + 320$ bitov. Druga dobra stran tega algoritma pa je, da na ta način Boštjan del C dejansko skrije pred vsemi, ki ne poznajo njegovega javnega ključa.

3.3 ZGOŠČEVALNA FUNKCIJA

V algoritmih digitalnega podpisa, v prejšnjem razdelku, smo, da bi zmanjšali velikost sporočila, uporabili posebno funkcijo - zgoščevalno funkcijo (Hash Function). Zato si jo bomo v tem razdelku na kratko ogledali. Velik del tega razdelka je povzet po knjigi [6].
Problem velikosti sporočila bi lahko rešili tudi drugače. Recimo, da bi sporočilo razbili na k delov zahtevane dolžine (pri nas je to 160 bitov) in bi potem podpisali vsakega posebej. Vendar pa tak način podpisovanja daljših sporočil ni brez problemov. Najbolj očitni problem je, da bi na ta način za zelo dolgo sporočilo dobili resnično ogromen digitalni podpis (v primeru DSA ali ECDSA bi imel podpis dvakratno dolžino sporočila). Drugi, še važnejši problem pa je, da bi bilo tako možno te dele sporočila med seboj premešati in nekatere celo odstraniti, pa bi podpis še vedno ostal veljaven. Tega seveda ne želimo, zato bomo raje uporabili zgoščevalne funkcije. Oglejmo si osnovno idejo njihovega delovanja.

Naj bo $\mathcal{P}$ končna množica možnih sporočil. Sporočilo $m \in \mathcal{P}$ naj bo neko bitno zaporedje poljubne dolžine. Zgoščevalna funkcija H slika iz množice $\mathcal{P}$ v množico bitnih zaporedij neke določene končne dolžine, recimo dolžine n (v naših primerih je bil n vedno enak 160 bitov). Tem slikam sporočil bomo rekli zgoščena sporočila. Boštjan, ki želi poslati Anžetu podpisano sporočilo m, bo najprej izračunal zgoščeno sporočilo $z = H(m)$, dolžine n. Potem bo, glede na svoj ključ $K_B \in \mathcal{K}$, izračunal $y = sig_{K_B}(z)$. Anžetu bo poslal sporočilo m in digitalni podpis y.
Zgoščevalna funkcija je javno znana funkcija. Ker pa je del algoritma digitalnega podpisa, jo moramo izbrati tako, da ne bo ogrozila njegove varnosti. Zato zahtevamo, da ima vsaj naslednje lastnosti.

Lastnost enosmernosti

Definicija 3.3 *Zgoščevalna funkcija H je enosmerna, če je za znano zgoščeno sporočilo z računsko nemogoče (ne poznamo učinkovitega algoritma) poiskati tako sporočilo m, da velja $H(m) = z$.*

Predpostavimo, da zna Vasja ponarediti podpis na nekem naključnem zgoščenem sporočilu z, torej zna izračunati y (pri nekaterih algoritmih digitalnega podpisa je to možno, za podrobnejše informacije glej [6, razdelek 3.5]). Če zna poiskati sporočilo m tako, da velja $z = H(m)$, potem je par (m, y) veljaven ponaredek. Če ima zgoščevalna funkcija lastnost enosmernosti, tak način ponarejanja seveda ni mogoč.

Lastnost WCF

Definicija 3.4 *Zgoščevalna funkcija H ima lastnost WCF (Weakly Collision free), če je za znano sporočilo m računsko neizvedljivo poiskati tako sporočilo $\tilde{m}$, da velja $m \neq \tilde{m}$ in $H(m) = H(\tilde{m})$.*

Recimo, da želi Vasja ponarediti Boštjanov podpis in dobi v roke neko sporočilo m z veljavnim Boštjanovim podpisom y. Potem Vasja izračuna $z = H(m)$. Če mu uspe poiskati tako sporočilo $\tilde{m}$, da velja $m \neq \tilde{m}$ in $H(m) = H(\tilde{m})$, potem je par $(\tilde{m}, y)$ sporočilo podpisano z veljavnim Boštjanovim podpisom. Spet velja, da, če ima zgoščevalna funkcija lastnost WCF, tak način ponarejanja ni mogoč.

Lastnost SCF

Definicija 3.5 *Zgoščevalna funkcija H ima lastnost SCF (Strongly Collision free), če je računsko neizvedljivo poiskati dve sporočili m in $\tilde{m}$, da bi veljalo $m \neq \tilde{m}$ in $H(m) = H(\tilde{m})$.*

Recimo, da ima Vasja sporočili m in $\tilde{m}$, za kateri velja $m \neq \tilde{m}$ in $H(m) = H(\tilde{m})$. Če uspe prepričati Boštjana, da podpiše sporočilo m z veljavnim podpisom y, je par $(\tilde{m}, y)$ spet veljaven ponaredek. Spet sledi, da, če ima funkcija lastnost SCF, tak način ponarejanja ni mogoč.

Očitno je, da, če ima neka zgoščevalna funkcija H lastnost SCF, potem ima tudi lastnost WCF. Pokazali pa bomo, da je lastnost SCF dejansko najmočnejša in implicira obstoj ostalih dveh. To je, pokazali bomo, da, če ima neka zgoščevalna funkcija lastnost SCF, ima tudi lastnost enosmernosti.

Trditev 3.6 *Če ima zgoščevalna funkcija H lastnost SCF, ima tudi lastnost enosmernosti.*

Dokaz: Dokazali bomo, da, če zgoščevalna funkcija nima lastnosti enosmernosti, potem tudi nima lastnosti SCF. Dokaz bomo zožili na primer, ko funkcija H slika $H : \mathcal{P} \longrightarrow \mathcal{Z}$ in sta $\mathcal{P}$ in $\mathcal{Z}$ končni množici, za kateri velja, da $|\mathcal{P}| \geq 2|\mathcal{Z}|$. Če funkcija H nima lastnosti enosmernosti, bomo pokazali, da obstaja algoritem, ki zna najti sporočili m in $\tilde{m}$, za kateri velja $m \neq \tilde{m}$ in $H(m) = H(\tilde{m})$ z verjetnostjo vsaj $\frac{1}{2}$. To pa pomeni, da funkcija H nima lastnosti SCF.
Naj torej velja, da zgoščevalna funkcija H nima lastnosti enosmernosti. Torej obstaja taka preslikava $A : \mathcal{Z} \longrightarrow \mathcal{P}$, ki je inverzna preslikava funkcije H, da za vsak $z \in \mathcal{Z}$ velja $H(A(z)) = z$. Naj bo B algoritem, ki naredi naslednje.

1. Izbere poljubno sporočilo $m \in \mathcal{P}$.
2. Izračuna $z = H(m)$.
3. Izračuna $\tilde{m} = A(z)$.
4. Če velja $\tilde{m} \neq m$, potem vrne *resnično.*
   Drugače vrne *neresnično.*

Pokazali bomo, da algoritem B vrne rezultat *resnično* z verjetnostjo vsaj $\frac{1}{2}$, kar pomeni, da zgoščevalna funkcija H nima lastnosti SCF. Na množici $\mathcal{P}$ definiramo ekvivalenčno relacijo z naslednjim predpisom:

$$m \sim \tilde{m} \quad \text{natanko tedaj, ko} \quad H(m) = H(\tilde{m}).$$

Ni težko preveriti, da je to res ekvivalenčna relacija.
Ekvivalenčni razred $[m] = \{m \in \mathcal{P}; m \sim \tilde{m}\}$ je natanko inverzna slika nekega elementa

$z \in \mathcal{Z}$, torej je število ekvivalenčnih razredov največ $|\mathcal{Z}|$. Označimo množico teh ekvivalenčnih razredov s $\mathcal{C}$.

V 1. koraku algoritma B si izberemo poljubno sporočilo $m \in \mathcal{P}$. Potem obstaja $|[m]|$ takih sporočil $\tilde{m} \in \mathcal{P}$, ki jih lahko dobimo kot rezultat v 3. koraku. Izmed teh $|[m]|$ sporočil je $|[m]| - 1$ različnih od začetnega $m \in \mathcal{P}$. Torej je verjetnost, da algoritem vrne vrednost resnično za nek $m \in \mathcal{P}$ enaka

$$\frac{|[m]| - 1}{|[m]|}.$$

V povprečju je torej verjetnost, da nam algoritem B vrne vrednost *resnično* navzdol omejena:

$$p = \frac{1}{|\mathcal{P}|} \sum_{m \in \mathcal{P}} \frac{|[m]| - 1}{|[m]|} = \frac{1}{|\mathcal{P}|} \sum_{c \in \mathcal{C}} \sum_{m \in c} \frac{|c| - 1}{|c|} = \frac{1}{|\mathcal{P}|} (\sum_{c \in \mathcal{C}} |c| - \sum_{c \in \mathcal{C}} 1) \geq$$

$$\geq \frac{1}{|\mathcal{P}|} (|\mathcal{P}| - |\mathcal{Z}|) \geq \frac{|\mathcal{P}| - |\mathcal{P}|/2}{|\mathcal{P}|} = \frac{1}{2}.$$

Torej bomo v povprečju znali poiskati $m \in \mathcal{P}$ in $\tilde{m} \in \mathcal{P}$ taka, da velja $m \neq \tilde{m}$ in $H(m) = H(\tilde{m})$. To pa pomeni, da zgoščevalna funkcija H nima lastnosti SCF. ∎

Dokazali smo, če ima zgoščevalna funkcija lastnost SCF, ima tudi lastnost enosmernosti. Torej je lastnost SCF najmočnejša, saj sta ostali dve lastnosti njeni posledici. Zato bomo v nadaljevanju za zgoščevalne funkcije zahtevali zgolj, da imajo lastnost SCF.

NAPAD S PARADOKSOM ROJSTNEGA DNEVA

V tem razdelku bomo za zgoščevalne funkcije določili varnostni pogoj, ki bo odvisen samo od števila elementov množice $\mathcal{Z}$ oz. ekvivalentno od dolžine zgoščenih sporočil. Določili bomo spodnjo mejo dolžine zgoščenih sporočil tako, da *napad s paradoksom rojstnega dneva* ne bo uspešen.

Paradoks rojstnega dneva pravi: če imamo množico vsaj 23 ljudi, potem imata vsaj dva izmed njih rojstni dan na isti dan v letu z verjetnostjo vsaj $\frac{1}{2}$. To sicer ni resnični paradoks, vendar pa se ta izjava nekako upira preprosti logiki. Kako je ta paradoks povezan z našim napadom, bomo videli v nadaljevanju.

Recimo, da imamo zgoščevalno funkcijo $H : \mathcal{P} \longrightarrow \mathcal{Z}$, kjer sta $\mathcal{P}$ in $\mathcal{Z}$ spet končni množici z lastnostjo $|\mathcal{P}| \geq 2|\mathcal{Z}|$. Označimo $|\mathcal{P}| = m$ in $|\mathcal{Z}| = n$. Recimo, da iz množice $\mathcal{P}$ naključno izberemo k elementov $m_1, m_2, \ldots, m_k$ in izračunamo $z_i = H(m_i)$ za $1 \leq i \leq k$. Potem pogledamo, če so kateri z_i enaki. Če so, potem lahko trdimo, da funkcija H nima lastnosti SCF.

Izračunali bomo spodnjo mejo verjetnosti, da funkcija H nima lastnosti SCF. Ta spodnja meja bo odvisna le od števil k in n, ne pa tudi od števila m.

Predpostavimo, da je $|H^{-1}(z)| \approx m/n$ za vsak $z \in \mathcal{Z}$. To lahko predpostavimo, kajti če inverzne slike nimajo približno enake moči, se verjetnost, da funkcija H nima lastnosti SCF, kvečjemu poveča. Ker so inverzne slike približno enake velikosti in ker so sporočila m_i izbrana naključno, lahko tudi na zgoščena sporočila z_i gledamo kot na naključne (ne nujno različne) elemente množice $\mathcal{Z}$. Izračunajmo verjetnost, da so z_i med seboj različni.

Imamo z_1, ki je naključno izbran. Verjetnost, da je z_2 različen od z_1 je $(n-1)/n$, verjetnost, da je z_3 različen od z_1 in z_2 je $(n-2)/2$ itd. Verjetnost, da so vsi z_i med seboj različni za $1 \leq i \leq k$, je

$$\Big(1-\frac{1}{n}\Big)\cdot\Big(1-\frac{2}{n}\Big)\cdot\ldots\cdot\Big(1-\frac{k-1}{n}\Big) = \prod_{i=1}^{k-1}\Big(1-\frac{i}{n}\Big).$$

Za majhne realne x velja ocena $1-x \approx \mathrm{e}^{-x}$. To oceno dobimo iz razvoja v Taylorjevo vrsto eksponentne funkcije e^{-x}. Velja namreč

$$\mathrm{e}^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \ldots$$

Torej lahko našo verjetnost ocenimo takole

$$\prod_{i=1}^{k-1}(1-\frac{i}{n}) = \prod_{i=1}^{k-1}\mathrm{e}^{i/n} = \mathrm{e}^{-k(k-1)/2n}.$$

Če označimo z ε verjetnost, da zgoščevalna funkcija H nima lastnosti SCF, potem velja

$$\varepsilon \approx 1 - \mathrm{e}^{\frac{-k(k-1)}{2n}}.$$

Zdaj pa iz ocene izrazimo število k:

$$\frac{-k(k-1)}{2n} \approx \ln(1-\varepsilon),$$

$$k^2 - k \approx 2n\ln\Big(\frac{1}{1-\varepsilon}\Big).$$

Če zanemarimo vrednost $-k$, dobimo

$$k \approx \sqrt{2n\ln\Big(\frac{1}{1-\varepsilon}\Big)}.$$

Če vzamemo $\varepsilon = 1/2$, je naša ocena

$$k \approx 1.17\sqrt{n}.$$

Ugotovili smo, če vzamemo več kot $\sqrt{n}$ elementov množice $\mathcal{P}$ in izračunamo njihova zgoščena sporočila, potem je verjetnost, da sta vsaj dve zgoščeni sporočili izmed njih enaki, približno $1/2$.
Če vzamemo za množico $\mathcal{P}$ množico vseh ljudi, za množico $\mathcal{Z}$ pa 365 različnih dni v neprestopnem letu , $H(m)$ pa naj pomeni datum rojstnega dneva človeka m, potem dobimo natanko paradoks rojstnega dneva. Če namreč v naši oceni vzamemo $n = 365$, dobimo $k \approx 22.3$. Torej, kot smo omenili na začetku, če izmed vseh ljudi izberem množico 23 ljudi, bosta imela vsaj dva izmed njih rojstni dan na isti dan z verjetnostjo vsaj $1/2$. Ta napad nam določa spodnjo mejo dolžine zgoščenih sporočil. Zgoščevalna funkcija, ki

kot rezultat vrne 40-bitna zgoščena sporočila, ni preveč varna, ker lahko že pri 2^{20} (približno milijon) naključno izbranih sporočilih najdemo dve z enakim zgoščenim sporočilom z verjetnostjo približno $1/2$. Priporočena spodnja meja za dolžino zgoščenih sporočil je vsaj 128 bitov (potem bomo potrebovali vsaj 2^{64} naključno izbranih sporočil, da bi dobili dve enaki zgoščeni sporočili z verjetnostjo približno $1/2$). Pri algoritmih digitalnega podpisa pa uporabljamo 160 bitna zgoščena sporočila.

V naših algoritmih digitalnega podpisa smo uporabili zgoščevalno funkcijo SHA-1 (Secure Hash Algorithm). Poglejmo si njeno predhodnico, to je zgoščevalno funkcijo MD4. Funkcija SHA-1 je nekoliko bolj zapletena kot funkcija MD4, osnovna ideja pa je pri obeh funkcijah enaka.

ZGOŠČEVALNA FUNKCIJA MD4

Recimo, da imamo sporočilo m. Najprej razdelimo to sporočilo na N delov:

$$m = M[0]\, M[1]\, \ldots\, M[N-1].$$

Vsak del $M[i]$ imenujemo *beseda*. Vsaka beseda je neko bitno zaporedje dolžine 32. Za število N pa velja $N \equiv 0 \bmod 16$. Zdaj bomo iz sporočila m konstruirali 128-bitno zgoščeno sporočilo po naslednjem postopku.

Zgoščevalna funkcija MD4:

1. $A = 67452301\,(\text{hex})$,
   $B = efcdab89\,(\text{hex})$,
   $C = 98badcfe\,(\text{hex})$,
   $D = 10325476\,(\text{hex})$.
2. <u>for</u> $i = 0$ <u>to</u> $N/16 - 1$ <u>do</u>
3.   <u>for</u> $j = 0$ <u>to</u> 15 <u>do</u>
           $X[j] = M[16i + j]$.
4. $AA = A$,
       $BB = B$,
       $CC = C$,
       $DD = D$.
5.   Naredi 1.krog zgoščevanja,
6.   naredi 2.krog zgoščevanja,
7.   naredi 3.krog zgoščevanja.
8. $A = A + AA$,
       $B = B + BB$,
       $C = C + CC$,
       $D = D + DD$.

V prvem koraku izberemo začetne vrednosti štirih besed A, B, C in D, ki jih imenujemo *registri*. Nato shranimo prvih 16 besed sporočila m v tabelo X. Potem shranimo vrednosti registrov v začasne spremenljivke AA, BB, CC in DD (4.korak). Nato se izvedejo trije krogi zgoščevanja. V vsakem krogu se izvede kombinacija naslednjih operacij na vsaki besedi tabele X:

$X \wedge Y$	po bitih
$X \vee Y$	po bitih
$X \oplus Y$	XOR po bitih
$\neg X$	negacija
$X + Y$	seštevanje po modulu 2^{32}
$X \lll s$	ciklični pomik v levo za s mest

V 1., 2. in 3. krogu funkcije **MD4** uporabimo zaporedoma funkcije f, g, in h, definirane spodaj.

$$\begin{aligned} f(X,Y,Z) &= (X \wedge Y) \vee ((\neg X) \wedge Z) \\ g(X,Y,Z) &= (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z) \\ h(X,Y,Z) &= X \oplus Y \oplus Z \end{aligned}$$

1. krog

1.	$A = (A + f(B,C,D) + X[0]) \lll 3$
2.	$D = (D + f(A,B,C) + X[1]) \lll 7$
3.	$C = (C + f(D,A,B) + X[2]) \lll 11$
4.	$B = (B + f(C,D,A) + X[3]) \lll 19$
5.	$A = (A + f(B,C,D) + X[4]) \lll 3$
6.	$D = (D + f(A,B,C) + X[5]) \lll 7$
7.	$C = (C + f(D,A,B) + X[6]) \lll 11$
8.	$B = (B + f(C,D,A) + X[7]) \lll 19$
9.	$A = (A + f(B,C,D) + X[8]) \lll 3$
10.	$D = (D + f(A,B,C) + X[9]) \lll 7$
11.	$C = (C + f(D,A,B) + X[10]) \lll 11$
12.	$B = (B + f(C,D,A) + X[11]) \lll 19$
13.	$A = (A + f(B,C,D) + X[12]) \lll 3$
14.	$D = (D + f(A,B,C) + X[13]) \lll 7$
15.	$C = (C + f(D,A,B) + X[14]) \lll 11$
16.	$B = (B + f(C,D,A) + X[15]) \lll 19$

2. krog

$$
\begin{array}{rl}
1. & A = (A + g(B,C,D) + X[0] + 5A827999) \lll 3 \\
2. & D = (D + g(A,B,C) + X[4] + 5A827999) \lll 5 \\
3. & C = (C + g(D,A,B) + X[8] + 5A827999) \lll 9 \\
4. & B = (B + g(C,D,A) + X[12] + 5A827999) \lll 13 \\
\hline
5. & A = (A + g(B,C,D) + X[1] + 5A827999) \lll 3 \\
6. & D = (D + g(A,B,C) + X[5] + 5A827999) \lll 5 \\
7. & C = (C + g(D,A,B) + X[9] + 5A827999) \lll 9 \\
8. & B = (B + g(C,D,A) + X[13] + 5A827999) \lll 13 \\
\hline
9. & A = (A + g(B,C,D) + X[2] + 5A827999) \lll 3 \\
10. & D = (D + g(A,B,C) + X[6] + 5A827999) \lll 5 \\
11. & C = (C + g(D,A,B) + X[10] + 5A827999) \lll 9 \\
12. & B = (B + g(C,D,A) + X[14] + 5A827999) \lll 13 \\
\hline
13. & A = (A + g(B,C,D) + X[3] + 5A827999) \lll 3 \\
14. & D = (D + g(A,B,C) + X[7] + 5A827999) \lll 5 \\
15. & C = (C + g(D,A,B) + X[11] + 5A827999) \lll 9 \\
16. & B = (B + g(C,D,A) + X[15] + 5A827999) \lll 13
\end{array}
$$

3. krog

$$
\begin{array}{rl}
1. & A = (A + h(B,C,D) + X[0] + 6ED9EBA1) \lll 3 \\
2. & D = (D + h(A,B,C) + X[8] + 6ED9EBA1) \lll 9 \\
3. & C = (C + h(D,A,B) + X[4] + 6ED9EBA1) \lll 11 \\
4. & B = (B + h(C,D,A) + X[12] + 6ED9EBA1) \lll 15 \\
\hline
5. & A = (A + h(B,C,D) + X[2] + 6ED9EBA1) \lll 3 \\
6. & D = (D + h(A,B,C) + X[10] + 6ED9EBA1) \lll 9 \\
7. & C = (C + h(D,A,B) + X[6] + 6ED9EBA1) \lll 11 \\
8. & B = (B + h(C,D,A) + X[14] + 6ED9EBA1) \lll 15 \\
\hline
9. & A = (A + h(B,C,D) + X[1] + 6ED9EBA1) \lll 3 \\
10. & D = (D + h(A,B,C) + X[9] + 6ED9EBA1) \lll 9 \\
11. & C = (C + h(D,A,B) + X[5] + 6ED9EBA1) \lll 11 \\
12. & B = (B + h(C,D,A) + X[13] + 6ED9EBA1) \lll 15 \\
\hline
13. & A = (A + h(B,C,D) + X[3] + 6ED9EBA1) \lll 3 \\
14. & D = (D + h(A,B,C) + X[11] + 6ED9EBA1) \lll 9 \\
15. & C = (C + h(D,A,B) + X[7] + 6ED9EBA1) \lll 11 \\
16. & B = (B + h(C,D,A) + X[15] + 6ED9EBA1) \lll 15
\end{array}
$$

Rezultat teh operacij so nove vrednosti registrov A, B, C in D. V zadnjem koraku shranimo vsoto novih in starih vrednosti registrov kot vrednosti registrov. Ta vsota je definirana kot vsota dveh pozitivnih celih števil, ki jih seštejemo po modulu 2^{32}. Potem ponovimo isti postopek na naslednjih 16 besedah sporočila m, to je vrnemo se na 2.korak algoritma. Ko pridemo do zadnjih šestnajstih besed sporočila m, pomeni, ko izračunamo zadnje vrednosti registrov, končamo. Te zadnje vrednosti registrov A, B, C in D zložimo v 128 bitno zaporedje, ki je naše zgoščeno sporočilo. Povejmo samo še to, da je zgoščevalna funkcija MD4 zelo hiter algoritem.

Zgoščevalna funkcija SHA-1 je bolj zapletena in počasnejša od funkcije MD4. Poglejmo si samo nekatere razlike med funkcijama MD4 in SHA-1.

1. Funkcija SHA-1 kot rezultat vrne 160-bitno zgoščeno sporočilo (v algoritmu se uporablja 5 registrov).

2. Vrednosti $X[0] \dots X[15]$ naredimo na enak način, le da se v krogih zgoščevanja uporabljajo tudi vrednosti $X[16] \dots X[80]$, ki jih izračunamo iz $X[0] \dots X[15]$.

Poglavje 4

EVIDENCA POŠTNIH PLAČIL

V tem poglavju bom poskušala predstaviti nov način evidentiranja plačila poštnine. Ta način bo izkoriščal možnosti digitalne in informacijske tehnologije. Dopuščal bo manjšo možnost goljufije in zmanjševal stroške, povezane z evidentiranjem plačila poštnine. V ta namen bom predstavila digitalno poštno znamko, ki bo temeljni kamen te evidence.

Najprej si bomo ogledali razloge, zaradi katerih je sploh nastala potreba po digitalni poštni znamki. Sledile bodo nekatere zahteve, ki jim bo taka znamka morala zadoščati. Potem bomo spregovorili o njeni zgradbi in o napravi, ki bo take znamke znala izdelovati. Ta naprava se bo imenovala poštna varnostna naprava in bo skrbela za vse potrebne kriptografske algoritme ter hkrati tudi za vse plačilne akcije. Ogledali si bomo tudi potrebno komunikacijo med poštno varnostno napravo in poštno organizacijo. Na koncu pa bomo poskušali oceniti varnost tega novega načina evidentiranja.

Večina predlogov v tem poglavju se sklada s predlogi, ki jih USPS (The United States Postel Service) uporablja v svojem programu IBIP (Information-Based Indicia Program) [5]. V tem programu je specificirana uporaba digitalne poštne znamke in je trenutno v fazi testiranja v Združenih državah Amerike.

4.1 UVOD

Za začetek si poglejmo zgodovino evidentiranja plačevanja poštnine, kako to počnemo danes in kako naj bi to počeli v bodoče. Da bomo dobili boljšo predstavo o tem, kakšen naj bi ta bodoči sistem pravzaprav bil, si bomo pogledali tudi nekaj zahtev, ki jim bo moral zadoščati.

4.1.1 POGLED V ZGODOVINO

Večina poštnih organizacij po vsem svetu zahteva predplačilo za svoje storitve. Ta način je sicer zelo dober v ekonomskem smislu (manjši stroški), vendar pa zahteva, da je vsak poštni kos opremljen s preverljivo označbo predplačila. Najbolj poznana in zgodovinsko prva taka označba je poštna znamka. Četudi so se znamke izkazale za dobre na veliko načinov, imajo tudi precej pomanjkljivosti. Izdelava in distribucija sta dragi, poleg tega so predmet kraje. Znamke same ne nosijo nobenih informacij o recimo času in kraju pošiljanja. Kakor bomo videli, ponujajo le omejeno varnost plačevanja poštnine. Arthur Pitney [2] je leta 1902 izumil prvi poštni števec, ki naj bi nekoliko omilil pomanjkljivosti poštnih znamk. Poštni števec je mehanska naprava z združeno tiskalno in plačilno akcijo. To napravo, sicer z mnogimi izboljšavami, uporabljamo še dandanes in njena osnovna funkcija še vedno ostaja enaka. Njena varnost temelji na predpostavki, da ni mogoče izdelati učinkovitega ponaredka poštne znamke. To pa v današnjih časih, ko so se pojavili tiskalniki visoke kvalitete, dostopni vsakomur, preprosto ne drži več. Zato se je pojavila potreba po drugačnem načinu evidentiranja plačila poštnine. Ta drugačen način predstavlja digitalna poštna znamka.

4.1.2 POGLED V DANAŠNJI ČAS

Čeprav poštni sistem v današnji dobi velja za manj učinkovitega od ostalih naprednejših načinov komunikacije, kot so elektronska pošta, telefon, fax, ostaja še vedno edini univerzalni sistem raznašanja sporočil. Razlogov za to je več: ponuja razvejan sistem raznašanja sporočil po razumni ceni, ima prednosti varnega prenosa in ima prednost prenosa papirja, ki zakonsko še vedno ostaja podlaga vsakršnega poslovanja.

Razvoj digitalne tehnologije je doprinesel k dramatičnim spremembam v procesu pisanja poštnih pošiljk, njihovi obdelavi in celo raznašanju. Ocenjuje se, da je približno 80% pošte, ki nastane v industriji, napisane s pomočjo računalnika in tiskalnika ter da je vsaj pol teh računalnikov priključenih na neko omrežje [4].
Poštni sistem za pobiranje plačila ima posebno lastnost, za svoje storitve namreč zahteva predplačilo. In v ta namen zahteva tudi fizično evidenco tega predplačila, natisnjeno na pošiljko ali kako drugače pripeto zraven.

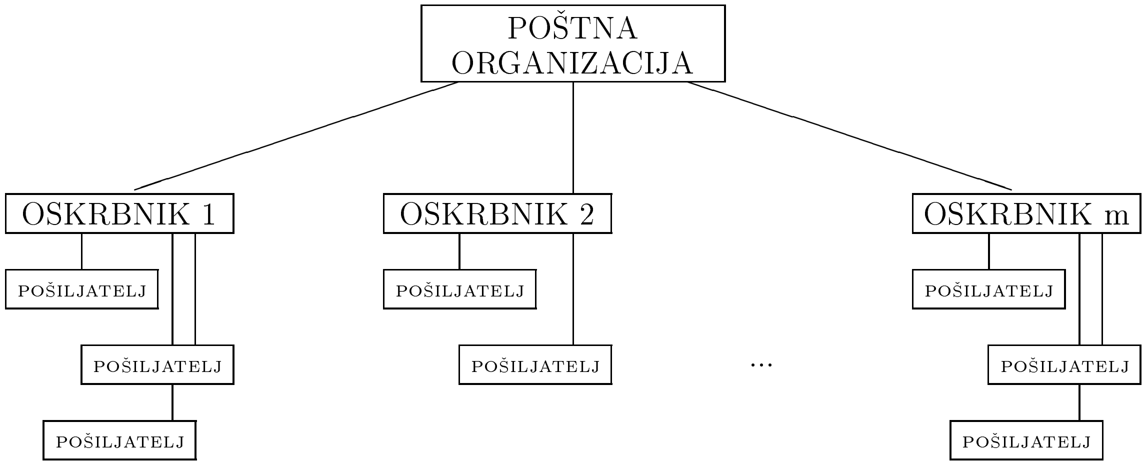
OSNOVNA IDEJA PROCESA POŠILJANJA POŠTE
Recimo, da je Boštjan zaposlen v časopisni družbi Delo d.d. Tam skbi za pravilno izstavljanje računov naročnikom časopisa Delo ali Slovenskih novic na področju Ljubljane. Vsakega desetega v mesecu mora Boštjan poskrbeti, da vsak naročnik dobi svoj mesečni račun. Ker Boštjan pozna natančno število naročnikov, ki jim bo poslan račun in če zraven predpostavi, da bo približno polovici naročnikov potrebno poslati tudi opomin, lahko precej dobro oceni, koliko znamk bo za to potreboval vsak mesec. Ker je lepiti znamke na toliko pisem zamudno opravilo, so si v časopisni družbi Delo d.d omislili posebno napravo, s katero na vsako pismo odtisnejo poseben žig, ki označuje plačano poštnino. Te žige pa je precej lahko ponarediti. Zato so se pri Delu d.d. odločili, da bodo v bodoče na račune tiskali posebne digitalne poštne znamke. Zdaj pa si oglejmo, kakšen naj bi bil ta proces, ki bi tako tiskanje omogočil. V tem procesu bodo sodelovali trije glavni udeleženci, več o tem si lahko pogledate v [5]. To so *pošiljatelj* oz. *uporabnik*, *oskrbnik* in *centralna poštna organizacija*.

Pošiljatelj oz. uporabnik: je nekdo, ki želi poslati neko poštno pošiljko. V našem primeru je to časopisna družba Delo d.d. oz. Boštjan.

Oskrbnik: je neko računalniško podjetje, ki nadzoruje izdelavo digitalnih poštnih znamk. Hkrati je tudi nekakšen posrednik med uporabnikom in poštno organizacijo. Je namreč edini, ki neposredno komunicira s poštno organizacijo, uporabniki dostopajo do njenih podatkov posredno-preko oskrbnika. Recimo, da si Delo d.d. izbere za svojega oskrbnika podjetje Računalnik d.o.o. To podjetje bo skrbelo, da bodo vsi potrebni procesi potekali pravilno, razreševalo bo morebitne nejasnosti in podobno.

Centralna poštna organizacija: ima popoln pregled nad količino izdelanih poštnih znamk. Da bi se zavarovala pred različnimi goljufijami (ponarejene ali kopirane znamke), mora poštna organizacija digitalne poštne znamke tudi pregledovati. Ko bomo govorili o centralni poštni organizaciji, jo bomo raje imenovali kar poštna organizacija.

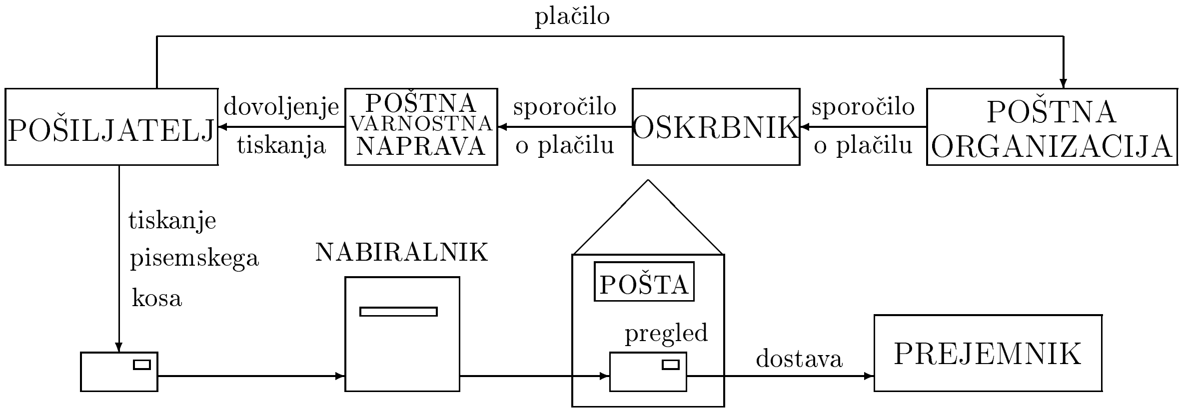
Ureditev udeležencev lahko predstavimo z drevesno strukturo, kot kaže slika 4.1.



Slika 4.1 : Drevesna struktura udeležencev

Veliko vlogo v procesu pošiljanja pošte bo igrala *poštna varnostna naprava.* To je naprava, ki zna zaščititi podatke na digitalni poštni znamki z digitalnim podpisom in hkrati skrbi, da so izvedena vsa potrebna predplačila. Tako napravo lahko poseduje oskrbnik (v tem primeru se podatki do pošiljatelja prenašajo elektronsko) ali pa je, zlasti kadar gre za količinsko večje uporabnike poštnih storitev, last pošiljatelja (v tem primeru se podatki do oskrbnika prenašajo elektronsko). Družba Delo d.d mora vsak mesec poslati ogromno pisem, zato bo njihova poštna varnostna naprava stala kar v Boštjanovi pisarni, potrebni podatki pa se bodo do in od podjetja Računalnik d.o.o prenašali po telefonskih linijah. Ker se bliža konec leta, mora Boštjan tudi v svojem privatnem življenju poslati veliko čestitk. Ker noče izgubljati časa s kupovanjem znamk in si želi vse opraviti z domačega naslonjača ob enih ponoči, se je tudi Boštjan odločil pošiljati svojo pošto s pomočjo digitalnih znamk. Seveda si v ta namen ne bo omislil tudi svoje poštne varnostne naprave, ki bi stala sredi dnevne sobe. Raje se bo povezal s podjetjem Računalnik d.o.o., ki na svoji spletni strani za take uporabnike, kot je Boštjan, ponuja možnost izdelave digitalne poštne znamke. Boštjan bo tako samo poslal potrebne podatke (količino in vrsto znamk) podjetju in seveda izvršil plačilo (elektronsko). V tem primeru bo poštna varnostna naprava znotraj podjetja Računalnik d.o.o. Ta naprava bo izračunala zahtevane parametre (katere, bomo izvedeli kasneje), ki bodo poslani Boštjanu. Zdaj bo Boštjanov tiskalnik lahko natisnil digitalno poštno znamko neposredno na pisemsko ovojnico in postopek bo končan.

Kot je v navadi pri poštnih sistemih, se postopek pošiljanja pošte prične tako, da pošiljatelj nakaže plačilo poštni organizaciji. Poštna organizacija izda sporočilo o prejemu plačila oskrbniku, ki to sporoči poštni varnostni napravi. Šele ko ta dobi sporočilo o plačilu, dovoli pošiljateljevemu tiskalniku dejansko tiskanje digitalne poštne znamke. Zdaj torej pošiljateljev tiskalnik končno lahko na izbrani poštni kos natisne znamko. Pošiljatelj nato poštni kos odda v nabiralnik in s pošiljateljevega stališča je proces pošiljanja pošte zaključen.



Slika 4.2 : Postopek pošiljanja pošte

Za poštno organizacijo pa proces še ni končan. Ko se vsi poštni kosi iz poštnih nabiralnikov zberejo na eni izmed lokalnih pošt, jih poštni uslužbenci najprej pregledajo. To pomeni, da z optičnim čitalcem odčitajo podatke na digitalni poštni znamki, nato računalnik preveri njihovo pristnost, več o tem v nadaljevanju. Tako poštna organizacija preveri veljavnost neke poštne znamke in skuša izločiti vse goljufive poštne znamke. Če pisemski kos pregled uspešno prestane, ga pošljejo v proces razvrščanja in končno dostavijo prejemniku. Slika 4.2 prikazuje pravkar opisani postopek pošiljanja pošte.

4.1.3 ZAHTEVE

Naša želja je, da bi bila digitalna poštna znamka čim bolj uporabna in čim bolj varna. Želimo si, da bi zadostila vsem kriterijem poštne organizacije, kot tudi, da bi bila čim bolj všečna pošiljateljem in enostavna za uporabo. Zato mora biti njena zgradba podvržena nekaterim zahtevam, naštetim spodaj [4].
Digitalna poštna znamka bo vsebovala nekatere podatke (vrednost poštnine, datum pošiljanja, identiteta poštne varnostne naprave). Te podatke bomo zaščitili z digitalnim podpisom.

1. **VISOKA STOPNJA VARNOSTI** Recimo, da bi najboljši algoritem za odkrivanje tajnih ključev za digitalni podpis potreboval vsaj 2^{80} operacij [4].

2. **ODPORNOST PROTI PONAREDKOM** To je odpornost proti ponarejanju digitalnega podpisa s poznavanjem javnih ključev (algoritem za izračun podpisa mora biti dovolj dober, da bi algoritem, ki bi znal ponarediti posamezni podpis, potreboval vsaj 2^{40} operacij) [4].

3. **MINIMALNA VELIKOST DIGITALNE POŠTNE ZNAMKE** Fizična reprezentacija znamke je omejena glede na mesto, ki ga sme zasesti na recimo dopisnici ali kartici, zato digitalna znamka ne sme biti veliko večja od današnje poštne znamke. Tudi optični čitalci so tem bolj natančni in hitrejši, čim manjša je.

4. **VELIKOST PODPISA NAJ BO ODPORNA PROTI VEČANJU** Četudi se kriptografski algoritmi in moč računalnikov nenehno izboljšujejo, mora velikost digitalnega podpisa vseeno ostati kar se da majhna.

5. **RAČUNSKA UČINKOVITOST** Od algoritma za izračun digitalnega podpisa in od algoritma za njegovo preverjanje pričakujemo, da bosta najhitrejša možna.

6. **SAMOZADOSTNOST** Digitalna znamka mora sama vsebovati vse podatke, ki jih potrebujemo, da preverimo digitalni podpis. Tisti, ki preverja, mora biti torej zmožen preveriti konsistentnost podatkov digitalne znamke iz podatkov samih.

7. **TESTI DOSLEDNOSTI** Da bi preprečili različne zlorabe takih znamk, je zaželeno, da se v procesu preverjanja izvedejo različni testi na podatkih znamke, ne samo preveri pristnost digitalnega podpisa.

8. **ZAUPNOST** Zaželeno je, da ohranimo zaupnost nekaterih podatkov v digitalni znamki. To pomeni, da ti podatki ne smejo biti del odprtega dela podatkov, temveč morajo biti shranjeni znotraj digitalnega podpisa in jih torej more in sme brati samo poštna organizacija oz. tisti, ki preverja.

9. **EKONOMIČNA UČINKOVITOST** Cena celotnega generiranja digitalne znamke in njenega preverjanja mora biti kar se da majhna.

Poiskati rešitev, ki bi zadostila vsem kriterijem, je izredno težko. Opazimo lahko, da so zahteve 1,2 in 6 v direktnem nasprotju z zahtevo 3. Zahtevi 1 in 2 sta zahtevi po varnem algoritmu, torej po večanju števil, uporabljenih v algoritmu, kar je seveda v nasprotju z zahtevo po čim manjši reprezentaciji. Zahteva 6 je zahteva po čim večjem številu podatkov, ki naj jih vključimo v digitalno znamko, kar je seveda spet v nasprotju s potrebo po minimalni velikosti. Da bi ustregli zahtevi 6, bomo uporabili kriptografijo javnih ključev. Zahtevi 1 in 2 sta zahtevi po visoki stopnji varnosti, zahteva 5 pa je zahteva po računski učinkovitosti. Da bi jim ustregli, bomo uporabili kriptosistem, ki bo temeljil na eliptičnih krivuljah. Da bi zadostili zahtevi 3, to je zahtevi po minimalni velikosti, bomo za algoritem digitalnega podpisa uporabili algoritem PVSSR, opisan v prejšnjem poglavju. Poglejmo si torej natančno zgradbo digitalne poštne znamke.

4.2 DIGITALNA POŠTNA ZNAMKA

V tem razdelku si bomo ogledali natančno zgradbo digitalne poštne znamke. Najprej bomo ugotovili, katere podatke je smiselno vključiti vanjo. Potem si bomo ogledali algoritem za izračun certifikata tj. potrdila, ki ga bo poštna organizacija podelila poštni varnostni napravi in jo tako pooblastila za izdelavo digitalnih poštnih znamk. Ogledali si bomo tudi algoritem digitalnega podpisa (njegovo generiranje in preverjanje), za konec pa še končno podobo digitalne znamke. Potem se bomo seveda morali vprašati še, v kolikšni meri smo zadostili zahtevam iz prejšnjega razdelka.

4.2.1 PODATKI NA DIGITALNI POŠTNI ZNAMKI

Digitalno poštno znamko, imenujmo jo znamka DPM (Digital Post Mark) sestavljata dvodimenzionalna črtna koda in nekatere informacije, napisane v človeku berljivi obliki. Priporočeni format črtne kode je PDF 417. Ta vrsta črtne kode je dvodimenzionalna črtna koda, ki je še posebej primerna za večjo količino podatkov.
V tabeli 4.3 so predstavljeni podatki, potrebni za generiranje znamke DPM [5].

PODATEK	PREDSTAVLJENOST S ČRTNO KODO	PREDSTAVLJENOST V ČLOVEKU BERLJIVI OBLIKI	DOLŽINA (bit)
ŠTEVILKA VERZIJE DPM	DA	NE	8
ŠTEVILKA ALGORITMA	DA	NE	8
ID PSD	DA	DA	16
ŠTEVILKA PROGRAMA	DA	NE	48
NARAŠČAJOČI REGISTER	DA	NE	40
PADAJOČI REGISTER	DA	NE	32
POŠTNINA	DA	DA	24
DATUM POŠILJANJA	DA	DA	16
POŠTNA ŠT. POŠILJATELJA	DA	DA	32
POŠTNA ŠT. PREJEMNIKA	DA	NE	40
POŠTNI RAZRED	DA	DA	32
ŠTEVILKA CERTIFIKATA	DA	NE	160
DIGITALNI PODPIS	DA	NE	160 + 40
REZERVIRANO POLJE			prilagodljiva

Tabela 4.3: Podatki, potrebni za generiranje znamke DPM in njihova dolžina

ŠTEVILKA VERZIJE DPM je številka, ki jo podeli poštna organizacija tej vrsti poštne znamke (glede na podatke, ki jo sestavljajo).

ŠTEVILKA ALGORITMA je številka, ki označuje, kateri algoritem smo uporabili za generiranje digitalnega podpisa.

ID PSD je edinstvena identifikacijska številka poštne varnostne naprave (različni napravi imata različni številki ID PSD), ki jo podeli oskrbnik v dogovoru s poštno organizacijo.

ŠTEVILKA PROGRAMA je identifikacijska številka programa, s katerim dostopamo do poštne varnostne naprave.

NARAŠČAJOČI REGISTER je skupna vrednost poštnine, ki jo je poštna varnostna naprava porabila za generiranje poštnih znamk v vsem svojem življenjskem obdobju.

PADAJOČI REGISTER je vrednost, ki je še na voljo za plačevanje poštnine.

POŠTNINA je vrednost poštnine za ta specifični kos pošte.

DATUM POŠILJANJA je datum pošiljanja tega specifičnega kosa pošte.

POŠTNA ŠTEVILKA POŠILJATELJA je poštna območna številka pošiljatelja. V človeku berljivi obliki sta to številka in ime območne poštne enote.

POŠTNA ŠTEVILKA PREJEMNIKA je poštna območna številka prejemnika.

POŠTNI RAZRED je vrsta poštnega kosa ali način pošiljanja.

ŠTEVILKA CERTIFIKATA je edinstvena številka, ki ga poštna varnostna naprava prejme od poštne organizacije. V pričujočem delu bom uporabila certifikat OMC (glej razdelek 4.2.2) , tako je njegova dolžina 160 bitov.

DIGITALNI PODPIS za izračun digitalnega podpisa bom uporabila algoritem PVSSR. Kot smo videli v prejšnjem poglavju, tako izračunani digitalni podpis prispeva k dolžini podatkov 160 bitov in še nekaj bitov dodatka. Videli bomo, da bo ta dodatek znašal 40 bitov.

4.2.2 OPTIMALNI POŠTNI CERTIFIKAT

V tem poglavju bom podala opis algoritma za izračun certifikata [4], to je enolične številke, ki jo poštni varnostni napravi podeli CA (Certificate Authority). To je ustanova, ki je edina odgovorna za izbiro in podelitev certifikatnih številk. Taka ustanova je lahko znotraj krovne poštne organizacije ali pa je njen zunanji del. V nadaljevanju bomo privzeli, da je CA del poštne organizacije. V nasprotnem primeru se mora med CA in poštno

organizacijo vzpostaviti nek način varne komunikacije.
Videli bomo, da ta algoritem ustreza vsem devetim zahtevam (glej razdelek 4.1.3), ki naj jim zadošča znamka DPM na kar najbolj optimalen način.
Naj opomnim, da, ko govorim o pošiljanju sporočil od poštne organizacije k poštni varnostni napravi in obratno, v resnici mislim na pošiljanje sporočil od poštne organizacije k oskrbniku in od tod k poštni varnostni napravi in obratno.
Za krajše pisanje imenujmo poštno varnostno napravo kar naprava PSD (Postal Security Device) in jo označimo kot naprava A.
V postopku inicializacije naprave PSD (razdelek 4.3.1) oskrbnik dodeli napravi PSD edinstveno identifikacijsko številko, to je ID PSD. V procesu poštne pooblastitve naprave PSD (razdelek 4.3.2) naprava PSD od poštne organizacije prejme vrednost I_A. Ta vrednost je sestavljena iz več številk: iz številke verzije DPM, številke algoritma in iz številke ID PSD.
Javni podatki, ki so znani vnaprej in enaki za vse naprave PSD, so :
Naj bo $E(\mathbb{F}_{2^m})$ grupa eliptične krivulje nad končnim obsegom $\mathbb{F}_{2^m}$ moči N. Naj bo $P \in E(\mathbb{F}_{2^m})$ točka na tej krivulji reda n. Točka P naj bo izbrana tako, da je n veliko praštevilo, dolgo najmanj 160 bitov in da zanj velja $n|N$.

- Poštna organizacija ima svoj tajni in svoj javni ključ. Naj bo c naključno pozitivno celo število in $c < n$, potem je c *tajni ključ poštne organizacije*, $B = c \cdot P$ pa *javni ključ poštne organizacije*.

- Naprava PSD, ki smo jo označili z A, izbere naključno pozitivno celo število $k_A < n$, potem je k_A *tajni parameter naprave A,* nato naprava A izračuna $k_A \cdot P$, ki pa je *javni parameter naprave A*. Vrednost $k_A \cdot P$ naprava A pošlje poštni organizaciji.

- Poštna organizacija izbere naključno pozitivno celo število $c_A < n$. Nato izračuna $\gamma_A = k_A \cdot P + c_A \cdot P$. Vrednost γ_A imenujemo *optimalni poštni certifikat* oziroma krajše *OMC* (Optimal Mail Certificate). γ_A je točka na krivulji $E(\mathbb{F}_{2^m})$. Potem poštna organizacija izračuna še: $f = H(\gamma_A || I_A)$, kjer je H zgoščevalna funkcija SHA-1, $||$ pa operacija pripojitve zaporedja bitov I_A na zaporedje bitov γ_A in vrednost $m_A = c \cdot f + c_A \bmod n$. Poštna organizacija pošlje vrednost m_A in certifikat OMC γ_A napravi A .

- Naprava A sedaj lahko izračuna svoj tajni in javni ključ. Število $a = m_A + k_A \bmod n$ je *tajni ključ naprave A,* število $Q_A = a \cdot P$ pa je *javni ključ naprave A*.

Opazimo, da lahko tajni ključ a naprave PSD izračunamo tudi na naslednji način:

$$a = m_A + k_A \bmod n = c \cdot f + k_A + c_A \bmod n.$$

Torej je a funkcija tajnega ključa poštne organizacije c , tajnega poštnega parametra c_A, ki je drugačen za vsako napravo PSD in tajnega parametra k_A naprave PSD.
Javni ključ Q_A naprave PSD pa je funkcija samih javnih parametrov:

$$Q_A = a \cdot P = cf \cdot P + \gamma_A = f \cdot B + \gamma_A.$$

Torej lahko Q_A poštna organizacija izračuna samo s poznavanjem javnega parametra k_A naprave A , s pomočjo katerega dobi vrednost OMC γ_A. Vrednost B je javni ključ poštne

organizacije, vrednost f pa je javna vrednost, ki jo lahko izračuna kdorkoli.
Na tak način izračunani certifikat se imenuje *implicitni certifikat* [1]. Implicitni certifikat je primeren zaradi dveh razlogov. Prvi je ta, da znatno poenostavi proces upravljanja s ključi. Drugi pa je ta, da je na ta način naš certifikat OMC γ_A enostavno neka točka na krivulji $E(\mathbb{F}_{2^m})$, torej je njegova velikost v našem primeru (če uporabimo tehniko kompresije bitov, ki smo jo omenili v prejšnjem poglavju) približno 160 bitov. To je precej manj kot pri običajnih certifikatih, ki so sestavljeni iz certifikatne številke, javnega ključa in digitalnega podpisa tako, da je njihova dolžina najmanj 480 bitov (če uporabimo ECDSA) [2].

4.2.3 ZAHTEVE ZA DIGITALNI PODPIS

Digitalni podpis izračuna naprava PSD za vsak kos pošte posebej. Nato ga pošlje pošiljateljevemu tiskalniku, ki ga natisne kot del znamke DPM.
Za algoritem digitalnega podpisa bom uporabila algoritem PVSSR [4]. Ta algoritem se mi zdi primeren zaradi dveh razlogov.

- Prvi je, da njegova dolžina poveča skupno dolžino podatkov za samo 160 bitov in za velikost dodatka (glej prejšnje poglavje), kar je manj od dolžine digitalnega podpisa izračunanega z algoritmom ECDSA ali DSA.
- Drugi pa je ta, da na ta način lahko skrijemo nekatere podatke, ki jih želimo ohraniti zaupne, v sam digitalni podpis.

V prejšnjem poglavju, ko smo govorili o algoritmu PVSSR, smo vse podatke, ki smo jih želeli ohraniti skrivne, združili v del C, ostale podatke pa v del V. V del C vzamemo naslednje podatke: vrednosti naraščajočega in padajočega registra in številko programa. Dolžina dela C tako znaša $48 + 40 + 32 = 120$ bitov, torej je velikost dodatka 40 bitov (dolžina ključa je 160 bitov). Stopnja varnosti je tako 2^{40} [1]. To pomeni, da bo najboljši možni algoritem, ki bo znal ponarediti digitalni podpis s poznavanjem javnih informacij potreboval v povprečju 2^{40} operacij in ga bo potrebno ponoviti za vsak podpis posebej. Meja -40- je funkcija denarne vrednosti, pridobljene s ponaredkom in vrednosti uporabe vseh računalniških virov, ki so na voljo ponarejevalcu. Ker je v našem primeru denarna vrednost zelo majhna in če predpostavimo, da ima ponarejevalec v najboljšem primeru na razpolago zgolj zelo močen PC, bo minimalni čas, ki ga bomo porabili za ponaredek, še vedno nekaj ur [4]. Zato je določena meja smiselna in primerna.
Kot del V pa vzamemo vse še preostale podatke : poštnina, datum pošiljanja, poštna številka pošiljatelja, poštna številka prejemnika in poštni razred.
Torej smo podatke za generiranje digitalnega podpisa razdelili na način $C||V||I_A$, kjer bo del C skriven, del $V||I_A$ pa ne.

GENERIRANJE DIGITALNEGA PODPISA
Naprava PSD A , ki generira digitalni podpis za podatke $C||V||I_A$, mora seveda poznati $E(\mathbb{F}_{2^m})$, $P \in E(\mathbb{F}_{2^m})$, svoj tajni ključ a , svoj javni ključ Q_A in certifikatno številko γ_A. Potem naprava naredi naslednje korake.

1. Izbere naključno pozitivno celo število $k < n$.
2. Izračuna $R = k \cdot P$ je neka točka na krivulji $E(\mathbb{F}_{2^m})$.
3. Izračuna $e = Tr_R(C)$, kjer je Tr_R funkcija dodatka, ki smo jo opisali v prejšnjem poglavju.
4. Izračuna $d = H(e||I_A||V)$, kjer je H zgoščevalna funkcija SHA-1.
5. Izračuna $s = ad + k \bmod n$.
6. Par (s, e) je digitalni podpis podatkov $C||V||I_A$.

PREVERJANJE DIGITALNEGA PODPISA
Da začne proces preverjanja, mora poštna organizacija poznati $E(\mathbb{F}_{2^m})$, $P \in E(\mathbb{F}_{2^m})$ in svoj javni ključ B. Podatke na znamki DPM mora razdeliti na vrednost I_A, del V , certifikatno številko γ_A in digitalni podpis (s, e), potem naredi naslednje korake.

1. Izračuna $Q_A = f \cdot B + \gamma_A$ javni ključ naprave A.
2. Izračuna $d = H(e||I_A||V)$, kjer je H zgoščevalna funkcija SHA-1.
3. Izračuna $U = s \cdot P - d \cdot Q_A$.
4. Izračuna $X = Tr_U^{-1}(e)$.
5. Preveri, če $X \in \mathcal{Z}$ (glej prejšnje poglavje) in, če to drži, potem postavi $C := X$ in sprejme podpis kot veljaven. V nasprotnem primeru pa javi napako.

Zaupnost dela C lahko ohranimo le, če javni ključ poštne organizacije B ni znan zunaj sistema poštnega preverjanja. Na srečo se v našem primeru B uporablja zgolj v procesu preverjanja in torej ni nobenega dobrega razloga, zakaj bi ga postavili za javnega. Zanimivo je poudariti, da na ta način naš kriptografski algoritem javnih ključev v bistvu uporabljamo, kot bi imeli privatne ključe. Prednost je seveda v tem, da tudi, če se ključ B odkrije, zaupnost sicer izgubimo, a stopnja varnosti algoritma še vedno ostane enaka.

4.2.4 KONČNA PODOBA DIGITALNE POŠTNE ZNAMKE

Podatke za generiranje digitalne poštne znamke razdelimo na tri skupine :

1. Identifikacijska številka I_A (32 bitov) :
    - številka verzije DPM,
    - številka algoritma,
    - številka ID PSD.
2. Zaupni del C (120 bitov) :
    - številka programa,
    - vrednost naraščajočega registra,
    - vrednost padajočega registra.
3. Del V (144 bitov) :
    - poštnina,
    - datum pošiljanja,
    - poštna številka pošiljatelja,
    - poštna številka prejemnika,
    - poštni razred.

Iz teh podatkov in iz certifikatne številke γ_A naprava PSD generira digitalni podpis (s, e). Končno je tako znamka DPM sestavljena iz vrednosti I_A , V , (s, e) in γ_A . Te vrednosti predstavimo z dvodimenzionalno črtno kodo, najbolje s formatom PDF 417. Nekateri podatki, kot so: številka ID PSD, vrednost poštnine, datum pošiljanja, poštna številka pošiljatelja in poštni razred, pa so predstavljeni tudi v človeku berljivi obliki, torej so zapisani nekje ob tej kodi. Skupna velikost znamke DPM je 616 bitov, ki ji moramo prišteti še velikost dodatka, ki v našem primeru znaša 40 bitov, torej skupno 656 bitov.

4.2.5 RAZPRAVA

Tak način računanja certifikatne številke in digitalnega podpisa je način, ki najbolj optimalno zadosti vsem nasprotujočim si zahtevam iz razdelka 4.1.3 . Ta način je očitno dovolj varen, saj ima stopnjo varnosti približno enako algoritmu digitalnega podpisa z DSA, kjer uporabim 1024 bitni modul. Dolžini digitalnega podpisa in certifikatne številke sta znatno manjši, kot če uporabimo standardne algoritme (DSA ali ECDSA). Tabela 4.4 predstavlja dolžine znamke DPM pri uporabi različnih protokolov z isto stopnjo varnosti za računanje digitalnega podpisa in certifikatne številke [4].

	DSA	ECDSA	PVSSR	PVSSR in OMC
Podatki	296	296	296	296
Podpis	320	320	160 + 40	160 + 40
Certifikat št.	1344	480	480	160
Skupaj DPM	1960	1096	976	656

Tabela 4.4: Velikosti (v bitih) znamke DPM izračunane z različnimi algoritmi

1. V algoritmu DSA predpostavljamo uporabo 1024 bitnega modula. Certifikat pa je sestavljen samo iz javnega ključa (1024 bitov) in podpisa CA (320 bitov).
2. V algoritmu ECDSA je red eliptične krivulje približno 160 bitov, torej je dolžina podpisa 320 bitov. Certifikat pa je sestavljen iz javnega ključa (160 bitov) in podpisa CA (320 bitov).
3. V algoritmu PVSSR upoštevamo, da digitalni podpis prispeva k skupni dolžini podatkov 160 bitov in velikosti dodatka, ki je v našem primeru 40 bitov. Certifikat pa je sestavljen iz javnega ključa (160 bitov) in podpisa CA (320 bitov).
4. V algoritmu PVSSR in OMC smo za izračun digitalnega podpisa uporabili algoritem PVSSR, za izračun certifikatne številke pa algoritem optimalnega poštnega certifikata. Torej je podpis dolg 160 + 40 bitov, certifikatna številka pa je v tem primeru zgolj točka na krivulji in je njena dolžina 160 bitov.

Vidimo, da lahko z uporabo predlaganih algoritmov (algoritem digitalnega podpisa PVSSR in algoritem optimalnega poštnega certifikata) dramatično zmanjšamo velikost znamke DPM, a vendar se pri tem stopnja varnosti ne spremeni. Torej smo uspešno zadostili prvim dvem zahtevam po visoki stopnji varnosti in hkrati tudi zahtevi po minimalni velikosti znamke DPM. Če gledamo na daljši rok, je zadostiti četrti zahtevi verjetno še bolj pomembno. Pričakuje se, da se bo dolžina ključa oz. digitalnega podpisa, izračunanega z nekaterimi vrstami algoritmov, v naslednjih petih letih zaradi izboljšanja računskih zmožnosti in moči računalnikov povečala med 20% do 30% [4]. V našem primeru pa uporabljamo tehniko računanja z eliptičnimi krivuljami, ki smo jo sicer malo prilagodili za potrebe znamke DPM. Ta tehnika doslej velja za veliko bolj odporno proti takim povečanjem od drugih tehnik [4]. Z uporabo kriptografije javnih ključev smo zadostili zahtevi šest, to je zahtevi po samozadostnosti. Ker smo v znamko DPM vgradili dovolj podatkov, smo omogočili izvajanje različnih testov doslednosti (vrednost naraščajočega registra, datum pošiljanja, poštna številka prejemnika,itd.). Z uporabo digitalnega podpisa z algoritmoma PVSSR smo uspešno zagotovili zaupnost nekaterih podatkov. In končno še nekaj besed o ekonomični učinkovitosti take vrste evidentiranja plačil poštnine, ki je naša deveta zahteva. To je zelo pomembna zahteva in je v bistvu predpogoj za prvih osem. Evidentiranje plačila poštnine na običajen način, ki je trenutno v uporabi v večini držav, je verjetno enako učinkovit kot digitalen. Prednosti digitalnega načina se kažejo v zmanjševanju stroškov zaradi goljufije in v zmanjševanju stroškov pri generiranju znamke DPM kot tudi pri preverjanju. Tako lahko zahtevo po minimalni velikosti znamke DPM razumemo tudi kot zahtevo po zmanjševanju stroškov. Pri manjši znamki DPM so stroški tiskanja manjši, manjši pa so predvsem stroški branja, saj so optični čitalci dokazano bolj učinkoviti na krajših črtnih kodah [4]. Podobno lahko zahtevo po računski učinkovitosti razumemo kot zahtevo po zmanjšanju stroškov generiranja in preverjanja znamke DPM.

4.3 POŠTNA VARNOSTNA NAPRAVA

Poštna varnostna naprava oz. naprava PSD je naprava, ki skrbi za varno in pravilno generiranje znamke DPM [5]. Torej zna izvajati vse potrebne kriptografske algoritme in vrši vso kontrolo nad plačilnimi akcijami. Taka naprava mora biti dobro zaščitena in odporna proti vsakršnjim vdorom. Naprava PSD je lahko v oskrbnikovi infrastrukturi in uporabniki do nje dostopajo elektronsko. V tem primeru mora oskrbnik poskrbeti za varen način prenosa podatkov. Lahko pa je del uporabnikove infrastrukture. V tem primeru mora uporabnik preprečiti vsak nepooblaščen dostop. Njene glavne funkcije so:

- inicializacija,
- funkcija digitalnega podpisa in
- upravljanje z registroma.

Te funkcije morajo delovati usklajeno s funkcijami, ki jih vrši poštna organizacija v procesu generiranja znamke DPM. To so:

- pooblastitev naprave,
- finančna funkcija,
- kreiranje znamke DPM in
- funkcija kontrole registrov.

Tabela 4.5 prikazuje sodelovanje med funkcijami poštne organizacije in funkcijami naprave PSD:

Funkcije poštne organizacije $\longrightarrow$	Pooblastitev naprave	Finančna funkcija	Kreiranje znamke DPM	Funkcija kontrole registrov
Funkcije naprave PSD $\downarrow$				
Inicializacija	DA			
Funkcija dig. podpisa	DA	DA	DA	DA
Upravljanje z registroma		DA	DA	

Tabela 4.5: Sodelovanje med funkcijami naprave PSD in funkcijami poštne organizacije

V naslednjih dveh razdelkih si bomo podrobneje ogledali funkcije naprave PSD in funkcije poštne organizacije.

4.3.1 FUNKCIJE NAPRAVE PSD

V tem razdelku si bomo pogledali tri glavne funkcije naprave PSD. To so: inicializacija, funkcija digitalnega podpisa in funkcija upravljanja z registroma. Z inicializacijsko funkcijo naprava PSD od oskrbnika dobi začetne podatke (edinstveno številko ID PSD, glavne parametre digitalnega podpisa itd). Funkcijo digitalnega podpisa naprava PSD potrebuje za generiranje podpisa na znamki DPM. Ker pa naprava PSD skrbi tudi za vse plačilne akcije, mora znati spreminjati vrednosti registrov. Za to bo poskrbela funkcija upravljanja z registroma. Pa si jih poglejmo bolj podrobno.

FUNKCIJA INICIALIZACIJE NAPRAVE PSD

Inicializacijska funkcija se izvrši samo enkrat in sicer na začetku življenjskega obdobja naprave PSD in zajema naslednje akcije.

- Oskrbnik določi edinstveno številko za napravo PSD, ki jo imenujemo številka ID PSD (oskrbnik jo določi s privoljenjem poštne organizacije).
- Vrednosti naraščajočega in padajočega registra se postavijo na nič.
- Oskrbnik oskrbi napravo PSD z glavnimi parametri algoritma digitalnega podpisa (m, f, a, b, P in n).

S to funkcijo napravo PSD nekako pripravimo na delovanje.

FUNKCIJA DIGITALNEGA PODPISA

Funkcija digitalnega podpisa zajema generiranje digitalnega podpisa in njegovo preverjanje. Kot sem že omenila, bomo v procesu generiranja znamke DPM uporabili algoritem digitalnega podpisa PVSSR. Uporabili ga bomo, ker s tem dosežemo manjšo dolžino znamke DPM in ker lahko ohranimo nekatere podatke zaupne. Zaupnost podatkov pa temelji na nepoznavanju javnega ključa poštne organizacije. Ker je v primeru pošiljanja pošte komunikacija enosmerna, napravi PSD ni potrebno poznati javnega ključa poštne organizacije. Ko govorim o enosmerni komunikaciji, mislim na dejstvo, da naprava PSD samo podpisuje, poštna organizacija pa samo preverja.

V procesu generiranja znamke DPM pa bo potrebna tudi dvosmerna komunikacija in sicer med oskrbnikom in napravo PSD. Ta dvosmerna komunikacija bo sestavljena iz poročil o stanju na registrih, zahtev po transakciji poštnine in podobno. Nekatera od teh sporočil bo pošiljala naprava PSD oskrbniku, nekatera pa bo pošiljal oskrbnik napravi PSD. Zato bomo pri tej vrsti sporočil uporabili algoritem digitalnega podpisa ECDSA. Naprava PSD bo torej vršila dva različna algoritma za digitalni podpis, proces preverjanja podpisa pa bo vršila le z algoritmom ECDSA in bo morala zato poznati javni ključ oskrbnika.

V tabeli 4.6 so našteti parametri, potrebni za generiranje digitalnega podpisa, bodisi z algoritmom ECDSA ali pa z algoritmom PVSSR.

parameter	izvor	komentar
m	naprava PSD ga dobi v procesu inicializacije	standarden parameter za PVSSR, enak za vse PSD; definira končni obseg $\mathbb{F}_{2^m}$
f	naprava PSD ga dobi v procesu inicializacije	standarden parameter za PVSSR, enak za vse PSD; definira bazo za reprezentacijo končnega obsega
a, b	naprava PSD ga dobi v procesu inicializacije	standardna parametra za PVSSR, enaka za vse PSD; definirata eliptično krivuljo
P	naprava PSD ga dobi v procesu inicializacije	standarden parameter za PVSSR, enak za vse PSD; definira točko na krivulji
n	naprava PSD ga dobi v procesu inicializacije	standarden parameter za PVSSR, enak za vse PSD; definira red točke, je praštevilo velikosti $\approx 2^{160}$
a	naprava PSD ga izbere v procesu pooblastitve naprave	tajni ključ naprave PSD
M	sporočilo, ki ga generira PSD na podlagi zahteve pošiljatelja in svojih informacij	naprava PSD ga sporoči pošiljatelju
k	izbere ga naprava PSD	naključna vrednost; drugačna za vsak podpis
(s, e)	izračunani vrednosti v procesu podpisovanja	digitalni podpis

Tabela 4.6:Parametri, potrebni za generiranje digitalnega podpisa

V obeh algoritmih digitalnega podpisa naprava PSD uporabi zgoščevalno funkcijo, opisano v razdelku 3.3.

V tabeli 4.7 so našteti parametri, potrebni za preverjanje digitalnega podpisa z algoritmom ECDSA.

parameter	izvor	komentar
m	naprava PSD ga dobi v procesu inicializacije	standarden parameter za ECDSA, enak za vse PSD; definira končni obseg $\mathbb{F}_{2^m}$
f	naprava PSD ga dobi v procesu inicializacije	standarden parameter za ECDSA, enak za vse PSD; definira bazo za reprezentacijo končnega obsega
a, b	naprava PSD ga dobi v procesu inicializacije	standardna parametra za ECDSA, enaka za vse PSD; definirata eliptično krivuljo
P	naprava PSD ga dobi v procesu inicializacije	standarden parameter za ECDSA, enak za vse PSD; definira točko na krivulji
n	naprava PSD ga dobi v procesu inicializacije	standarden parameter za ECDSA, enak za vse PSD; definira red točke, je praštevilo velikosti $\approx 2^{160}$
C	naprava PSD ga dobi v procesu pooblastitve naprave	javni ključ oskrbnika
M'	sporočilo, ki ga naprava PSD prejme od oskrbnika	
(r', s')	digitalni podpis, ki ga prejme naprava PSD od oskrbnika	
w, u_1, u_2, v	izračunane vrednosti v procesu preverjanja	podpis je veljaven natanko tedaj, ko $v = r'$

Tabela 4.7:Parametri, potrebni za preverjanje digitalnega podpisa

UPRAVLJANJE Z REGISTROMA

Funkcija upravljanja z registroma skrbi za varno spreminjanje vrednosti obeh registrov. Kot smo že omenili, je vrednost naraščajočega registra skupna vrednost poštnine, ki jo je naprava PSD že porabila za plačevanje poštnine na doslej generiranih znamkah DPM v vsem svojem življenjskem obdobju, vrednost padajočega registra pa je vrednost, ki je še na voljo za plačevanje poštnine. Ti dve vrednosti se lahko spremenita samo v dveh primerih :

1. ko naprava PSD prejme poročilo o transakciji poštnine od finančne funkcije poštne organizacije, ali

2. ko naprava PSD prejme zahtevo po kreiranju nove znamke DPM.

Spomnimo se Boštjana, ki skrbi za pošiljanje računov v časopisni družbi Delo d.d. Vsak mesec mora poslati račune vsem naročnikom, nekaterim pa pošlje tudi opomin. Boštjan pozna natančno število naročnikov, recimo da jih je n. Ocenjuje, da bo opomin poslal približno polovici, torej bo mesečno poslal okoli $3n/2$ pisem. Če je x vrednost poštnine za en račun, bi moral Boštjan izvršiti $3n/2$ plačil po x tolarjev. Ker je n lahko precej velik, bi bilo to zelo zamudno. Boštjan raje izvrši enkratno plačilo v znesku $3nx/2$ poštni organizaciji. Ta to sporoči oskrbniku, ki sporočilo prenese napravi PSD. To sporočilo bomo imenovali *poročilo o transakciji poštnine.* Zdaj naprava PSD poveča vrednost padajočega registra za $3nx/2$ tolarjev. Padajoči register si lahko predstavljamo kot nekakšen dobropis - toliko tolarjev ima Delo d.d v dobrem in jih lahko porabi za plačevanje poštnine na kasnejših računih. Zdaj pa želi Boštjan začeti s tiskanjem znamk. Za vsak račun bo potreboval novo poštno znamko, torej bo vsakič zahteval od naprave PSD generiranje nove znamke. Temu bomo rekli *zahteva po kreiranju nove znamke DPM.* Ko bo naprava PSD prejela to zahtevo, bo izdala potrebne podatke za tiskanje znamke, hkrati pa bo zmanjšala vrednost padajočega registra za vrednost ravnokar porabljene poštnine (to je za x) in za natanko toliko tudi povečala vrednost naraščajočega registra. Pa si poglejmo še bolj natančno ta proces.
V prvem primeru, kadar želi uporabnik povečati vrednost padajočega registra, izvrši plačilo poštni organizaciji. Ko poštna organizacija prejme to plačilo, prejme naprava PSD poročilo o transakciji poštnine. To sporočilo vsebuje naslednje podatke: številko ID PSD, številko transakcije, trenutno stanje registrov, dodana vrednost poštnine, digitalni podpis oskrbnika. Ko naprava PSD prejme to sporočilo, mora najprej preveriti veljavnost digitalnega podpisa. Nato preveri, če se številka transakcije ujema s številko na zahtevi, kot je opisano v finančni funkciji v razdelku 4.3.2. Potem pa preveri še ujemanje trenutnega stanja na registrih. Če se vse vrednosti ujemajo, se izvršijo naslednje operacije:

- vrednost padajočega registra se poveča za dodano vrednost poštnine na tem poročilu,
- vrednost naraščajočega registra ostaja nespremenjena.

V primeru neujemanja podatkov naprava PSD javi napako.

V drugem primeru, kadar uporabnik želi natisniti novo znamko DPM, pošlje napravi PSD zahtevo po kreiranju nove znamke DPM. Ta zahteva je sestavljena iz naslednjih podatkov: številka kreiranja (je zaporedna številka zahteve po kreiranju nove znamke DPM), vrednost poštnine, poštna številka prejemnika, itd. Naprava PSD najprej preveri, ali je zahtevana vrednost poštnine v dopustnih mejah, ki smo jih določili s funkcijo pooblastitve naprave v razdelku 4.3.2. Potem preveri, če je vrednost padajočega registra dovolj velika (večja ali enaka zahtevani vrednosti poštnine). Če se številke ujemajo, se izvršijo naslednje operacije:

- vrednost padajočega registra se zmanjša za zahtevano vrednost poštnine,
- vrednost naraščajočega registra se zveča za zahtevano vrednost poštnine.

4.3.2 FUNKCIJE POŠTNE ORGANIZACIJE

Ogledali smo si že funkcije naprave PSD. Ker mora ta naprava delovati usklajeno s poštno organizacijo (recimo, poštna organizacija pošlje napravi PSD sporočilo o prejetem plačilu, ki smo ga imenovali poročilo o transakciji poštnine, naprava PSD pa spremeni vrednost padajočega registra), si sedaj oglejmo še štiri funkcije, ki jih vrši poštna organizacija. Najprej si bomo ogledali funkcijo pooblastitve naprave PSD. S to funkcijo poštna organizacija nekako registrira napravo PSD. To pomeni, da se naprava PSD in poštna organizacija "dogovorita" za bistvene parametre (naprava PSD prejme številko I_A, izvede se algoritem optimalnega poštnega certifikata, torej naprava PSD dobi številko certifikata γ_A in podobno). Naslednja bo finančna funkcija poštne organizacije, ta bo skrbela za izdajanje poročil o transakciji poštnine po vsakem prejetem plačilu. Potem si bomo ogledali funkcijo kreiranja nove znamke DPM. Ta funkcija je v bistvu postopek računanja potrebnih podatkov, ki jih naprava PSD posreduje pošiljateljevemu tiskalniku. Zadnjo pa si bomo ogledali še funkcijo kontrole registrov. Ta funkcija bo skrbela za usklajevanje podatkov med napravo PSD in poštno organizacijo. Recimo, ko bo naprava PSD izdala podatke za tiskanje nove znamke DPM, bo s funkcijo upravljanja z registroma, ki smo jo opisali v razdelku 4.3.1, spremenila vrednosti obeh registrov. Ti novi vrednosti bo sporočila poštni organizaciji.

POOBLASTITEV NAPRAVE PSD

Pooblastitev naprave PSD je proces, v katerem dobi naprava PSD še vse ostale podatke, potrebne za njeno normalno delovanje. To so denimo specifični podatki o uporabniku oz. stranki, določi se številka certifikata OMC γ_A, naprava PSD dobi vse potrebne kriptografske ključe itd. Ta proces se izvede, ko se je funkcija inicializacije naprave PSD že končala. Taka pooblastitev se lahko izvede tudi večkrat v življenjskem obdobju naprave PSD, vendar vsakič s privoljenjem poštne organizacije. Izvršijo se naslednji koraki.

1. Naprava PSD prejme številko I_A.

2. Naprava PSD prejme poštno številko pošiljatelja oz. stranke in številko programa, s katerim bo pošiljatelj dostopal do podatkov naprave PSD.

3. Izvede se algoritem za izračun optimalnega poštnega certifikata in tajnega ključa naprave PSD (glej razdelek 4.2.2).

    - Naprava PSD izbere naključno celo pozitivno število $k_A < n$, izračuna vrednost $k_A \cdot P$ in jo pošlje poštni organizaciji.
    - Poštna organizacija izračuna številko certifikata γ_A in vrednost m_A ter ju pošlje napravi PSD.
    - Naprava PSD izračuna svoj tajni ključ a in svoj javni ključ Q_A.

4. Naprava PSD prejme javni ključ oskrbnika C.

5. Oskrbnik določi najmanjšo in največjo dopustno vrednost poštnine, za katero sme naprava PSD generirati znamke DPM.

Če se kasneje kateri od podatkov spremeni, npr. spremeni se poštna številka uporabnika ali pa se zamenja certifikatna številka in s tem tudi tajni ključ naprave PSD, moramo proces pooblastitve naprave ponoviti.

Spet naj opomnim, da, ko govorim o pošiljanju sporočil od poštne organizacije k napravi PSD in obratno, v resnici mislim na pošiljanje sporočil od poštne organizacije k oskrbniku in od tod k napravi PSD in obratno.

FINANČNA FUNKCIJA

Finančna funkcija se izvede, ko stranka želi povečati vrednost padajočega registra, torej plačati poštnino za nekaj naslednjih pošiljk. Stranka mora seveda najprej izvesti plačilo poštni organizaciji. To naredi tako, da pošlje naprava PSD poštni organizaciji *zahtevo po transakciji poštnine*. Ta zahteva je sestavljena iz:

- številke ID PSD,
- številke transakcije (zaporedna številka zahteve po transakciji poštnine),
- zahtevane dodane vrednosti poštnine (vrednost izvedenega plačila),
- vrednosti naraščajočega in padajočega registra,
- datuma in časa pošiljanja zahteve,
- prejšnje transakcije poštnine (vrednost dodatne poštnine na prejšnji transakciji),
- datuma in časa prejšnje transakcije poštnine,
- številke certifikata in
- digitalnega podpisa (ECDSA standard).

Ko oskrbnikova infrastruktura uspešno preveri pristnost podpisa in ujemanje podatkov na zahtevi s podatki v podatkovni bazi pošte in ko preveri dejansko plačilo, pošlje *poročilo o transakciji poštnine*. To poročilo je sestavljeno iz naslednjih podatkov:

- številke ID PSD,
- številke transakcije (je številka z zahteve po transakciji poštnine),
- trenutnega stanja na registrih (ta podatek preprečuje večkratno spremembo registrov, ki bi jo povzročilo isto poročilo),
- dodane vrednosti poštnine in
- digitalnega podpisa oskrbnika (ECDSA standard).

Ko naprava PSD prejme poročilo o transakciji poštnine, preveri pristnost digitalnega podpisa, ujemanje številke transakcije z zahteve in s poročila ter ujemanje trenutnega stanja na registrih. Če je preverjanje uspešno, naprava PSD spremeni vrednosti registrov s funkcijo upravljanja z registroma in sporoči novi vrednosti registrov oskrbniku. Če v kateremkoli delu procesa preverjanje ni uspešno, ustrezna stranka, naprava PSD ali oskrbnik, generira sporočilo o napaki in prekine z akcijo.

KREIRANJE ZNAMKE DPM

Ko stranka želi natisniti novo znamko DPM, se izvedejo naslednji koraki.

1. Naprava PSD od pošiljatelja prejme zahtevo po kreiranju nove znamke DPM. Ta zahteva vsebuje najmanj naslednje podatke: zaporedno številko kreiranja, vrednost poštnine, poštno številko prejemnika, itd.
2. Naprava PSD preveri, ali je vrednost poštnine v dopustnih mejah, ki smo jih določili s funkcijo pooblastitve naprave. Potem preveri, če je vrednost padajočega registra dovolj velika (večja ali enaka vrednosti poštnine). Če se številke ujemajo, potem se spremenijo vrednosti registrov s funkcijo upravljanja z registroma.
3. Naprava PSD generira digitalni podpis, kot je opisano v razdelku 4.2.3.
4. Kot rezultat naprava PSD vrne nove vrednosti obeh registrov in digitalni podpis pošiljateljevemu tiskalniku.
5. Pošiljateljev tiskalnik lahko sedaj natisne znamko DPM.
6. Naprava PSD sporoči novi vrednosti registrov oskrbniku.

Če v koraku 1 preverjanje ni uspešno, naprava PSD ustavi proces kreiranja nove znamke DPM in javi napako.

FUNKCIJA KONTROLE REGISTROV

Funkcija kontrole registrov omogoča poštni organizaciji usklajevanje vrednosti registrov in hkrati njuno kontrolo. Naprava PSD generira poročilo o stanju na registrih in ga pošlje oskrbniku. To poročilo se generira vsakič po prejemu poročila o transakciji poštnine, vsakič po kreiranju nove znamke DPM, lahko pa tudi po naročilu uporabnika, v tem primeru se poročilo izda uporabniku. To poročilo je sestavljeno iz naslednjih podatkov:

- številke ID PSD,
- številke transakcije, če je poročilo posledica transakcije poštnine ali
- številke kreiranja, če je poročilo posledica kreiranje nove znamke DPM ali
- ne vsebuje take številke, če je poročilo posledica zahteve uporabnika,
- vrednosti naraščajočega in padajočega registra in
- digitalnega podpisa naprave PSD (ECDSA standard).

Ko oskrbnik prejme poročilo, seveda najprej preveri pristnost digitalnega podpisa. Potem preveri ujemanje podatkov (številka transakcije). Če se podatki ujemajo, uskladi vrednosti registrov z vrednostmi v bazi podatkov poštne organizacije. Nato pošlje poročilo o prejemu nazaj k napravi PSD. To poročilo mora vsebovati podobne podatke kot poročilo o stanju registrov. Če v kateremkoli delu procesa preverjanje ni uspešno, ustrezna stranka, naprava PSD ali oskrbnik, javi napako in prekine proces.

Še enkrat se spomnimo Boštjana, ki mora vsak mesec poslati $3n/2$ računov. Kot smo že omenili, bo časopisna družba Delo d.d v ta namen najprej plačala poštni organizaciji $3nx/2$ tolarjev, kjer je x vrednost poštnine za posamezen račun. To bo storila tako, da bo naprava PSD najprej poslala poštni organizaciji zahtevo po transakciji poštnine, potem bo časopisna družba Delo d.d izvedla plačilo. Ko bo poštna organizacija prejela plačilo, bo napravi PSD poslala poročilo o transakciji poštnine. Ta bo povečala vrednost padajočega registra za $3nx/2$ tolarjev in to sporočila poštni organizaciji. Zatem bo Boštjan od naprave PSD zahteval kreiranje nove znamke DPM tako, da bo sporočil poštno številko prejemnika računa in vrsto poštne pošiljke (v njegovem primeru tako ali tako govorimo le o ljubljanski območni številki). Naprava PSD bo izračunala potrebne podatke (digitalni podpis) in jih posredovala Boštjanovemu tiskalniku. Nato bo spremenila vrednosti obeh registrov in to sporočila poštni organizaciji. Boštjan lahko sedaj natisne digitalno znamko za posamezen račun. Potem ponovi proces za naslednji račun (ponovi zahtevo po kreiranju nove znamke DPM itd.). To lahko dela toliko časa, dokler ne izčrpa vrednosti padajočega registra. Ko se to zgodi, mora časopisna družba Delo d.d ponoviti postopek transakcije poštnine. V našem primeru je morda najbolje, če se plačilo izvrši enkrat mesečno, lahko pa tudi v drugačnih časovnih intervalih.

4.4 VARNOSTNA STRATEGIJA IN GOLJUFIVE POŠTNE ZNAMKE

V tem razdelku se bomo ukvarjali z vprašanjem goljufivih poštnih znamk. To so poštne znamke, ki niso bile izdelane na prej opisan legalen način, njihova glavna značilnost pa je, da poštnina zanje ni bila plačana. Ukvarjali se bomo predvsem z načini odkrivanja takih poštnih znamk [2]. Seveda bomo poskušali poiskati tudi načine kako njihovo pojavnost čim bolj zajeziti, torej kako goljufijo precej otežiti oziroma narediti neprivlačno. Kot na vseh področjih človekovega življenja, tudi tukaj igrajo stroški veliko vlogo. Paziti moramo, da stroški odkrivanja in preganjanja goljufije ne presežejo stroškov izpada dohodka zaradi goljufije. Že na začetku si lahko priznamo, da kot nobena druga stvar, tudi varnost znamke DPM nikoli ne bo 100% zanesljiva in bo torej vedno obstajala neka stopnja izgube. Goljufive poštne znamke delimo na dva razreda:

- na kopirane znamke DPM: to so znamke DPM, ki so neplačane, a vendar izgledajo kot neke legalne plačane znamke DPM in
- na ponarejene znamke DPM: to so ponaredki plačanih legalnih znamk DPM.

Najprej si bomo ogledali znamke iz obeh razredov in poskušali ugotoviti, kaj naj ukrene poštna organizacija, da bi bilo takih znamk čim manj. Potem si bomo ogledali še, katero varnostno strategijo naj izbere poštni sistem.

PONAREJENE ZNAMKE DPM

Podatke na znamki DPM, ki so edinstveni za vsakega uporabnika, smo zaščitili z digitalnim podpisom. Ponarejene znamke odkrivamo s preverjanjem digitalnega podpisa. Če bi ponarejevalec spremenil kateregakoli izmed podatkov na znamki, bi moral spremeniti tudi digitalni podpis.
Algoritem PVSSR, ki smo ga uporabili za izračun digitalnega podpisa, ima v našem primeru stopnjo varnosti 2^{40}. To pomeni, da bo najboljši možni algoritem, ki bo znal ponarediti podpis samo s poznavanjem javnih informacij, potreboval v povprečju 2^{40} operacij in ga bo potrebno ponoviti za vsak podpis posebej. Ker je denarna vrednost, pridobljena s ponaredkom (to je vrednost poštnine), relativno majhna glede na veliko število operacij, ki jih je potrebno izvesti v ta namen, je to vsekakor sprejemljiva in smiselna stopnja varnosti. Več o tem si lahko pogledate v [4].
Druga vrsta ponaredkov predpostavlja poznavanje tajnega ključa naprave PSD. Če ponarejevalec pozna tajni ključ naprave PSD, lahko jasno priredi podatke in ponaredi podpis. V generiranju znamke DPM smo uporabili algoritem PVSSR, ki je v bistvu rahlo modificiran ECDSA. Varnost ECDSA temelji na težavnosti problema diskretnega logaritma v grupi eliptične krivulje, več o tem si lahko ogledate v [3]. Če je red grupe dovolj velik, to je n je najmanj 160 bitov, je varnost tega algoritma enaka varnosti algoritma DSA s 1024 bitnim modulom. Slednji pa velja za dovolj varen sistem. Torej lahko predpostavimo, da je možnost odkritja tajnega ključa zanemarljivo majhna. Seveda moramo prej poskrbeti,

da je naprava PSD ustrezno zaščitena pred morebitnom nepooblaščenim vdorom in odkritjem tajnega ključa.

KOPIRANE ZNAMKE DPM

Težje, kot odkrivati ponarejene znamke DPM, je gotovo odkrivati kopirane znamke DPM. Ker so podatki na vseh kopijah identični, predstavlja seveda velik problem dejstvo, da nikoli ne moremo zagotovo vedeti, katera izmed kopij je izvirnik, torej legitimna plačana znamka DPM in katere so samo njene kopije, torej neplačane znamke DPM. Za odkrivanje kopiranih znamk DPM bo tisti, ki preverja, potreboval dostop do neke baze podatkov, v kateri bodo evidentirane vse doslej pregledane znamke DPM (zabeležile se bodo vrednosti padajočega in naraščajočega registra, datum pošiljanja, poštnina in identifikacijska številka naprave PSD); za več informacij glej [2]. Ta baza podatkov bo hranila svoje informacije kar nekaj časa, odvisno od vrste poštnih pošiljk. Če tisti, ki preverja, ne odkrije dvojnika v tej bazi, potem po vsej verjetnosti ne gre za kopirano znamko DPM. Če pa dvojnika odkrije, seveda ne more vedeti, kateri je legalna znamka DPM in kateri goljufiva.
Da bi preprečili kopiranje v večjem obsegu, smo v podatke znamke DPM vključili tako imenovane edinstvene podatke. To so podatki, ki so specifični samo za tisti kos pošte, na katerem so natisnjeni. Recimo: številki ID PSD in vrednost naraščajočega registra ne moreta nikoli biti enaki na dveh različnih kosih pošte. Vrednost naraščajočega registra neke naprave PSD z identifikacijsko številko ID PSD kar naprej raste, z vsakim kosom pošte je večja za vrednost poštnine. Torej ob pojavitvi dveh ali več kosov pošte z istimi številkami ID PSD in vrednostjo naraščajočega registra lahko z gotovostjo trdimo, da gre pri vseh, razen enem, za kopirane znamke DPM. Med edinstvene podatke gotovo sodi tudi datum izdajanja znamke DPM.
Vendar je to premalo, da bi popolnoma preprečili kopiranje. Zato smo v podatke znamke DPM vključili še poštno številko prejemnika. S tem smo kopiranje naredili še manj privlačno, ker bo vsak poštni kos z isto znamko DPM lahko odposlan samo v neko določeno geografsko območje. Seveda bi bilo najbolje, če bi v podatke znamke DPM vključili kar cel naslov prejemnika. Tako bi vsaka kopija nujno pomenila pošiljanje na isti naslov. Vendar pa vključevanje celega naslova v podatke znamke DPM ni brez težav. Prva je ta, da sta plačevanje poštnine in tiskanje naslova pogosto dva ločena procesa, narazen v času in prostoru. Druga pa je ta, da je celoten naslov gotovo prevelik, da bi ga vključili med podatke znamke DPM, kajti velikost znamke bi se s tem drastično povečala in s tem tudi stroški preverjanja in branja informacij znamke. Ti stroški so lahko celo večji od stroškov izgube, ki jih ima poštna organizacija zaradi goljufivih poštnih znamk in se torej taka poteza gotovo ne izplača. Zato se bomo zadovoljili zgolj z vključitvijo poštne številke prejemnika med podatke znamke DPM, kar bo prav tako nekoliko omejilo možnosti kopiranja.
Med podatke znamke DPM bi gotovo lahko vključili še druge podatke, ki so specifični za nek kos pošte, recimo težo poštnega kosa. To je sicer zelo uporaben podatek, vendar pa ga lahko vključimo samo, kadar pošto oddajamo skozi poštna okenca na Pošti in ne v poštne nabiralnike. Ker pa se ocenjuje, da je kar 75% vse pošte oddane skozi poštna okenca, lahko v teh primerih to zagotovo storimo in tako povečamo možnost odkrivanja nelegalnih kopij. Moja verzija digitalne poštne znamke je namenjena predvsem tistim

uporabnikom, ki neradi zavijejo na Pošto.

VARNOSTNA STRATEGIJA POŠTNEGA EVIDENTIRANJA

Kakšna naj bi bila varnostna strategija evidentiranja plačila poštnine? Glavni kriterij je seveda obseg goljufije, ki mora biti odkrita. Svetovni poštni sistem povezuje več milijard uporabnikov, več sto milijard kosov pošte in več milijonov poštnih delavcev. Za sistem take velikosti je najbolj praktičen način statistični način [2]. Velika večina pošiljateljev plačuje poštnino. Samo majhen del uporabnikov bo poskušal z goljufijo tudi vpričo poostrenih varnostnih ukrepov. Zato naj varnostna politika temelji na pregledovanju prilagodljivih statističnih vzorcev, podobno kot se to počne pri kontroli kakovosti in neoporečnosti različnih prodajnih izdelkov. V ta namen moramo določiti neko sprejemljivo stopnjo goljufije in prilagoditi velikost vzorca pregledanih znamk DPM tako, da ohranimo sistem v nekih sprejemljivih mejah goljufije. Na ta način imajo uporabniki še vedno občutek, da so nadzorovani, hkrati pa uravnotežimo stroške izgube zaradi goljufije s stroški odkrivanja in preganjanja. Velikost vzorca pregledanih znamk DPM naj bo spremenljiva in naj se prilagaja trenutnim potrebam v okolju - recimo, če je neko okolje znano po veliki razliki med številčnostjo poslane pošte in dejanskim plačilom poštnine, moramo seveda tu velikost vzorca ustrezno povečati.

Poglavje 5

IMPLEMENTACIJA

V zadnjem poglavju moje diplomske naloge bom predstavila implementacijo digitalnega podpisa z algoritmom PVSSR. Pri tem bomo potrebovali tri vrste aritmetičnih operacij: seštevanje in množenje po modulu praštevila p, seštevanje in množenje elementov obsega $\mathbb{F}_{2^m}$ in seštevanje točk grupe eliptične krivulje $E(\mathbb{F}_{2^m})$. Nekatere od teh operacij si bomo pogledali podrobneje medtem, ko bomo pri drugih le približno nakazali njihovo delovanje. Ogledali si bomo tudi podatkovne strukture, uporabljene za predstavitev elementov obsega $\mathbb{F}_{2^m}$ in za predstavitev točk grupe eliptične krivulje nad tem obsegom.

PODATKOVNE STRUKTURE

Že v drugem poglavju, ko smo govorili o končnem obsegu $\mathbb{F}_{2^m}$, smo povedali, da si lahko njegove elemente predstavljamo kot polinome stopnje manjše od m s koeficienti v $\mathbb{Z}_2$ ali pa kot m-bitna zaporedja. Torej:

$$\mathbb{F}_{2^m} = \{a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_2x^2 + a_1x + a_0\,;\, a_i \in \{0,1\} \text{ za } i = 0, \ldots, m-1\}$$

ali

$$\mathbb{F}_{2^m} = \{(a_{m-1}, a_{m-2}, \ldots, a_2, a_1, a_0)\,;\, a_i \in \{0,1\} \text{ za } i = 0, \ldots, m-1\}.$$

V implementaciji digitalnega podpisa z algoritmom PVSSR bomo elemente obsega $\mathbb{F}_{2^m}$ imenovali *tvektorji*. Vsak tvektor je sestavljen iz nekega končnega zaporedja *tbesed*. Vsaka tbeseda je neko 32-bitno število.

tvektor:	$\boxed{010\ldots110}$	$\boxed{110\ldots00}$	$\boxed{111\ldots111}$	$\ldots$	$\boxed{00\ldots110}$
	tbeseda 1	tbeseda 2	tbeseda 3	$\ldots$	tbeseda k

Slika 5.1: grafični prikaz tvektorja

Tako je vsak tvektor zaporedje bitov. Število tbesed pa je seveda odvisno od velikosti obsega $\mathbb{F}_{2^m}$, to je od velikosti števila m. Na primer: če je m dolg manj kot 160 bitov, potem je število potrebnih tbesed v tvektorju 5, tvektor pa bo dolg natančno 160 bitov.

Tvektor si še vedno lahko predstavljamo tudi kot polinom stopnje manjše od m s koeficienti v $\mathbb{Z}_2$. Pri tem je zadnji bit na levi strani koeficient, ki pripada x^0.

$$\begin{array}{ccccccc} & x^{32k+31}\dots x^{32k} & \dots & x^{63}\dots x^{32} & x^{31}\dots x^{0} \\ \text{tvektor:} & \boxed{a_{32k+31}\dots a_{32k}} & \dots & \boxed{a_{63}\dots a_{32}} & \boxed{a_{31}\dots a_{0}} \end{array}$$

Slika 5.2: tvektor kot polinom

Točke grupe eliptične krivulje $E(\mathbb{F}_{2^m})$ pa predstavimo tako, da je vsaka točka sestavljena iz dveh tvektorjev, to je: x-koordinata točke je nek tvektor in prav tako tudi njena y-koordinata.

ARITMETIČNE OPERACIJE

V algoritmu PVSSR potrebujemo tri vrste aritmetičnih operacij:

- seštevanje in množenje po modulu praštevila p,
- seštevanje, množenje in deljenje elementov obsega $\mathbb{F}_{2^m}$ in
- seštevanje točk grupe eliptične krivulje $E(\mathbb{F}_{2^m})$.

Za operacijo seštevanja dveh naravnih števil po modulu praštevila p lahko uporabimo kar navadno seštevanje naravnih števil in potem vsoto reduciramo po modulu p.
Za operacijo množenja dveh naravnih števil po modulu praštevila p lahko prav tako uporabimo navadno množenje naravnih števil in potem zmnožek reduciramo po modulu p.

Kot smo že povedali v drugem poglavju, je seštevanje elementov obsega $\mathbb{F}_{2^m}$ enako XOR operaciji dveh bitnih zaporedij. Dva tvektorja, imenujmo ju A in B, seštejemo tako, da naredimo XOR operacijo na njunih zaporednih tbesedah. Torej, če naj bo tvektor C vsota tvektorjev A in B, potem je prva tbeseda tvektorja C enaka prva tbeseda tvektorja A $\oplus$ prva tbeseda tvektorja B. Podobno velja tudi za drugo tbesedo in nato tretjo itd.
Množenje elementov obsega $\mathbb{F}_{2^m}$ pa je nekoliko bolj zapleteno. Recimo, da imamo dva tvektorja A in B.

$$A\!: \ \boxed{010\dots\ 100}\ \ \boxed{111\dots\ 100}\ \ \dots\ \ \boxed{111\dots\ 000}$$

$$B\!: \ \boxed{0001\dots\ 11}\ \ \boxed{000\dots\ 101}\ \ \dots\ \ \boxed{110\dots\ 001}$$

Recimo, da je naš redukcijski polinom enak: $f(x) = x^m + x^k + 1$.

$$f\!: \ \boxed{100\dots\ 000}\ \ \boxed{001\dots\ 000}\ \ \dots\ \ \boxed{000\dots\ 001}$$

Potem tvektor C, ki je zmnožek tvektorjev A in B, izračunamo na naslednji način. Najprej poiščemo prvi neničelni bit v tvektorju B. Recimo, da je ta bit pri potenci x^{r_1}. Potem premaknemo bite tvektorja A za r_1 mest v levo. Naj bo to nek novi tvektor A_1. Tvektor A_1 je dolg največ $2m-2$ bitov. Potem tvektor A_1 reduciramo po modulu f, to je

izračunamo ostanek pri deljenju polinoma A_1 s polinomom f. Rezultat shranimo v tvektor A_1. Nadaljujemo tako, da poiščemo naslednji neničelni bit v tvektorju B. Recimo, da je ta pri potenci x^{r_2}. Zdaj premaknemo bite tvektorja A za $r_1 - r_2$ v levo in rezultat shranimo v tvektor A_2 dolžine 2m-2. Tvektor A_2 reduciramo po modulu f. Potem seštejemo tvektorja A_1 in A_2 ter rezultat shranimo v tvektor A_1. Tako nadaljujemo, dokler ne izčrpamo vseh neničelnih bitov v tvektorju B. Končni tvektor A_1 je iskani tvektor C.
Deljenje elementov obsega $\mathbb{F}_{2^m}$ lahko izvedemo s pomočjo množenja oz. potenciranja elementov tega obsega. Velja namreč: $B^{2^m-2} = B^{-1}$ za vsak tvektor $B \neq 0$. Potem lahko kvocient A/B, kjer $B \neq 0$ izračunamo kot zmnožek tvektorjev A in B^{-1}. Tvektor B^{-1} pa izračunamo s pomočjo potenciranja. Še bolj učinkovit način računanja tvektorja B^{-1} pa je postopek s *razširjenim Evklidovim algoritmom.* Osnovno verzijo tega algoritma si lahko ogledate v [6].

Zdaj, ko poznamo princip seštevanja, množenja in deljenja tvektorjev, znamo tudi seštevati točke grupe eliptične krivulje $E(\mathbb{F}_{2^m})$. To naredimo preprosto z algoritmom opisanem v drugem poglavju, seveda moramo razlikovati med seštevanjem dveh različnih točk, seštevanjem dveh istih točk in seštevanjem inverznih si točk.
V algoritmu PVSSR bomo potrebovali množenje točk z nekim velikim praštevilom. Če je P točka na krivulji in n neko veliko število, potem $n \cdot P$ pomeni: $n \cdot P = P + P + \ldots + P$, to je $n-1$ operacij seštevanja. Ker je n velik, bi tako množenje pomenilo ogromno operacij seštevanja in temu primerno potrebovalo precej časa. Zato bomo za množenje točk grupe eliptične krivulje z velikimi števili uporabili različico algoritma *kvadriraj in množi*, prirejenega za točke grupe eliptične krivulje. V našem primeru je ta algoritem bolje poimenovati *podvajaj in seštej.* Število n zapišemo kot bitno zaporedje $n = n_1 \ldots n_\ell$, kjer so $n_i \in \{0, 1\}$ za vsak $i = 1, \ldots, \ell$. Potem izračunamo $Q = n \cdot P$ takole.

Različica algoritma kvadriraj in množi:

1. $Q = 0$
2. <u>for</u> $i = 1$ <u>to</u> ℓ <u>do</u>
3. $\quad Q = Q + Q$
4. $\quad$ <u>if</u> $n_i = 1$ <u>then</u> $Q = Q + P$
5. <u>end</u>

Naj opomnim, da se operacija seštevanja iste točke oz. operacija podvajanja $Q + Q$ razlikuje od operacije seštevanja dveh različnih točk $Q + P$. Obe operaciji sta opisani v 2.poglavju.

ALGORITEM PVSSR

Za ta algoritem potrebujemo naslednje funkcije:

1. funkcije pretvarjanja (s temi funkcijami pretvorimo število iz heksadecimalnega oz. decimalnega zapisa v tvektor in obratno),

2. funkciji seštevanja in množenja po modulu praštevila p,

3. funkciji seštevanja in množenja tvektorjev in

4. funkcijo množenja točke grupe eliptične krivulje z velikim številom n.

Ko te funkcije imamo, nam preostane le še to, da zapišemo operacije algoritma PVSSR. To pomeni, da najprej izberemo velikost obsega $\mathbb{F}_{2^m}$, to je velikost števila m. Nato izberemo grupo eliptične krivulje nad tem obsegom, to pomeni izberemo parametra a in b enačbe $y^2 + xy = x^3 + ax + b$. Nato na tej krivulji izberemo še točko P, ki je praštevilskega reda n. Sam algoritem PVSSR sestavljajo tri funkcije:

1. funkcija generiranja ključa,

2. funkcija podpisovanja sporočila in

3. funkcija preverjanja podpisa.

Funkcija generiranja ključa za izbrani tajni ključ d izračuna ustrezni javni ključ $Q = d \cdot P$.
Funkcija podpisovanja sporočila za izbrano sporočilo $m = C||V$ in tajni ključ d izračuna podpis (s, e).
Funkcija preverjanja podpisa pa za del sporočila V, javni ključ Q in podpis (s, e) izračuna pripadajoči del sporočila C in razglasi podpis za veljaven, če je dobljeni del sporočila C ustrezen (če pripada neki vnaprej določeni množici sporočil) oz. za neveljaven, če je dobljeni del sporočila C neustrezen.

Poglejmo si primer uporabe. Potrebni podatki za generiranje digitalne poštne znamke so naslednji: velikost obsega $\mathbb{F}_{2^m}$ oz. parameter $m = 97$, parametra enačbe eliptične krivulje

$$a = 1EA5CE2B7F0A58E01B4389418 \quad \text{in} \quad b = 09687742B6329E70680231988.$$

Točka grupe eliptične krivulje ima koordinati:

$$x_P = 094D3BA5743058882623639 29 \qquad \text{in} \quad y_P = 13CE0562CC51EF416B2C0CA80,$$

njen red pa je enak

$$n = 10000000000007383E2DE1E81.$$

Za tajni ključ d smo izbrali:

$$d = A0331.$$

Identifikacijska številka ID PSD je enaka:

$$I_A = 57E15F08,$$

vrednost optimalnega poštnega certifikata γ_A pa je:

$$\gamma_A = 2153F110257EEA5351BB609DE.$$

Vsi podatki, razen števila m so v heksadecimalnem zapisu. Ostali podatki, potrebni za izračun digitalnega podpisa, pa so sestavljeni iz delov

$$C = 1E4560000EF305689A21CCA02 \quad \text{in} \quad V = 000002F1023D00.$$

V tem primeru je podpis par (s, e):

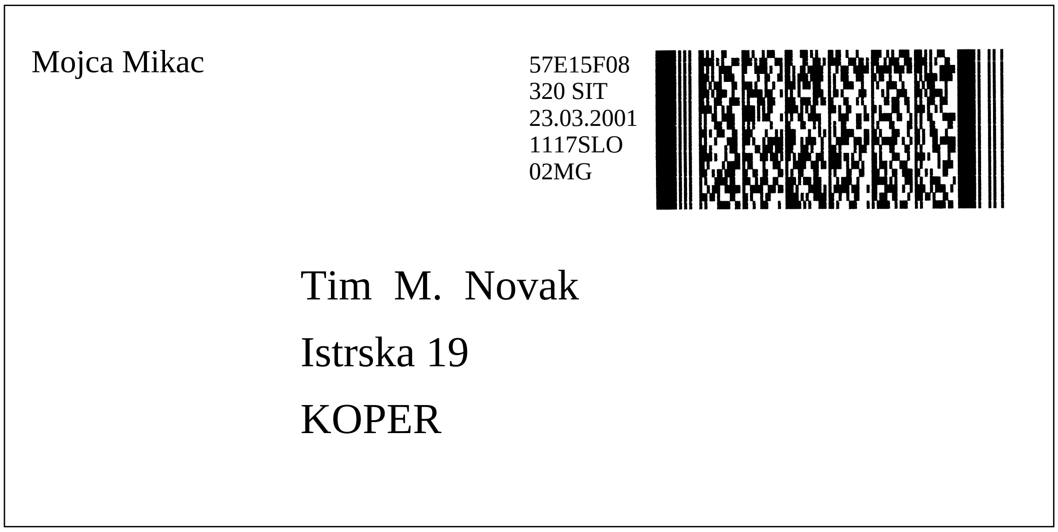
$$s = 1EB1BD5C57E7E4692DB8F99EE \quad \text{in} \quad e = 1D737A8907EEB1.$$

V digitalno poštno znamko bomo tako shranili del V, digitalni podpis (s, e), optimalni poštni certifikat γ_A in identifikacijsko številko I_A.

Kot smo že omenili, bomo podatke pretvorili v dvodimenzionalno črtno kodo formata PDF417. Primer programa, ki dane podatke pretvori v črtno kodo PDF417 lahko najdete na spletni strani

`http://www.symbol.com/produtcs/barcode_scanners/2d_tools_and_downloads`,

program pa se imenuje `pwsetup.exe`.



Slika 5.3: Digitalna poštna znamka predstavljena s črtno kodo formata PDF417 na pisemski ovojnici.

Literatura

[1] D. R. L. Brown in D. B. Johnson, Formal Security Proofs for a Signature Scheme with Partial Message Recovery, Research Report CORR 2000-39, University of Waterloo (2000).

[2] R. Cordery, L. Pintsov, S. Vanstone, Cryptographic Postage Evidencing, preprint, 1996.

[3] A. Jurišić in A. J. Menezes, Elliptic Curves and Cryptography, *Dr. Dobbs Journal,* april 1997, 26-37.

[4] L. A. Pintsov in S. A. Vanstone, Postal Revenue Collection in the Digital Age, Research Report CORR 2000-43, University of Waterloo (2000).

[5] The United States Postal Service (USPS), Information-Based Indicia Program (IBIP), Performance Criteria for Information-Based Indicia and Security Architecture for Open IBI Postage Evidencing Systems (PCIBI-O), 23. 02. 2000. (`http://56.0.78.92/html/programdoc.html`).

[6] D. R. Stinson, Cryptography: Theory and Practice, CRC Press, 1995

[7] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone: Handbook of Applied Cryptography, CRC Press 1997

[8] Elliptic Curve Online Tutorial, (`http://www.certicom.ca`).

[9] I. Vidav, Algebra, Mladinska knjiga, 1989.

[10] J. Barbič, Diplomska naloga: Schoofov algoritem, Fakulteta za matematiko in fiziko v Ljubljani, 2000.