SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Vesna Magjarević

DIGITALNI POTPIS

Diplomski rad

Voditelj rada:
Izv. prof. dr. sc.
Filip Najman

Zagreb, srpanj 2017.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Za voljene roditelje i svu njihovu podršku u svim oblicima.

Za mojih 3V koji su oduvijek i zauvijek moj najveći oslonac.

[0, 1]

Sadržaj

Uvod

U današnjem vremenu velik dio komunikacije odvija se putem interneta: zahvaljujući njemu, ljudi iz različitih dijelova svijeta razmjenjuju informacije unutar nekoliko sekundi.

U komunikaciji nam je često vrlo važna autentičnost, odnosno važno je znati da je informacija koju smo primili zaista stigla od pošiljatelja kojeg smo očekivali. U povijesti se to rješavalo na različite načine: voštanim pečatom na rubu kuverte u srednjem vijeku ili vlastoručnim potpisom na kraju pisma u skorijem vremenu.

Postavlja se pitanje kako riješiti problem autentičnosti u elektroničkoj komunikaciji. Nije više dovoljno digitalizirati vlastoručni potpis i dodati ga dokumentu. Naime, potencijalni neprijatelj može presresti dokument, iz njega izrezati potpis i njime kasnije potpisati dokumente koje vlasnik potpisa nikad nije vidio. Upravo zato je nužno da dokaz autentičnosti ne bude odvojiv od dokumenta koji je poslan - dokaz autentičnosti ovisit će o pošiljatelju, ali i o poruci koju osoba šalje. Takvu vrstu dokaza zovemo digitalni potpis.

U ovom radu bit će iznesen općeniti pregled digitalnih potpisa, a zatim će se promatrati neki od najpoznatijih algoritama za digitalni potpis: RSA algoritam, ElGamalova shema, DSA algoritam te njegova verzija koja koristi eliptičke krivulje ECDSA. Za svaki od algoritama bit će dan njihov formalni opis, primjer te uvid u njihovu sigurnost i mogućnosti napada.

Poglavlje 1

Osnovni matematički pojmovi i rezultati

U ovom poglavlju bit će izneseni pojmovi i rezultati koji će biti korišteni kasnije u radu. Za početak navodimo pojmove iz teorije brojeva, zatim par algebarskih pojmova te nekoliko pojmova vezanih uz eliptičke krivulje. Na samom kraju poglavlja dajemo par definicija vezanih uz složenost algoritama koje ćemo u ostatku rada proučavati.

1.1 Pojmovi iz teorije brojeva

Definicija 1.1. ***Pojam djeljivosti***
Neka su a i b cijeli brojevi takvi da je $a \neq 0$. Kažemo da je b djeljiv s a ako postoji cijeli broj x takav da je $b = ax$. To zapisujemo sa $a|b$. Ako b nije djeljiv s a, zapisujemo $a \nmid b$.

Na primjer, $17|323$ jer postoji x takav da vrijedi $17x = 323$. U ovom slučaju, $x = 19$. Obratno, vrijedi $17 \nmid 325$, jer ne postoji $x \in \mathbb{Z}$ takav da je $17x = 325$.

Teorem 1.2. ***Teorem o dijeljenju s ostatkom***
Za proizvoljan prirodni broj a i cijeli broj b postoje jedinstveni cijeli brojevi q i r, $0 \leq r < a$, takvi da je $b = qa + r$.

Definicija 1.3. ***Najveći zajednički djelitelj***
Za cijele brojeve b i c, cijeli broj a zovemo njihovim zajedničkim djeliteljem ako vrijedi $a|b$ i $a|c$. Ako je $b \neq 0$ ili $c \neq 0$, tada postoji konačno mnogo zajedničkih djelitelja od b i c. Najveći od njih zove se najveći zajednički djelitelj i označava sa $nzd(b, c)$.

Promotrimo brojeve $b = 18$ i $c = 48$. Zajednički djelitelji brojeva b i c su $D = \{1, 2, 3, 6\}$, jer za svaki $d \in D$ vrijedi $d|18$ i $d|48$. Najveći element skupa D je 6 pa označavamo $nzd(18, 48) = 6$. Navedeni brojevi su mali i nije bilo teško naći njihove

zajedničke djelitelje. Postavlja se pitanje kako naći najveći zajednički djelitelj velikih brojeva. Odgovor na to pitanje dat će Euklidov algoritam u teoremu 1.6.

Definicija 1.4. ***Relativno prosti brojevi***
Za cijele brojeve a i b kažemo da su relativno prosti ako je $nzd(a, b) = 1$.

Definicija 1.5. *Ako cijeli broj $m \neq 0$ dijeli razliku $a - b$, onda kažemo da je a kongruentan b modulo m i pišemo $a \equiv b \pmod{m}$.*

Teorem 1.6. ***Euklidov algoritam***
Neka su b i c cijeli brojevi, $b > 0, c > 0$. Uzastopnom uporabom teorema 1.2., dobivamo niz jednakosti:

$$\begin{aligned}
b &= cq_1 + r_1, \quad 0 < r_1 < c \\
c &= r_1q_2 + r_2, \quad 0 < r_2 < r_1 \\
r_1 &= r_2q_3 + r_3, \quad 0 < r_3 < r_2 \\
&\ldots \\
r_{j-2} &= r_{j-1}q_j + r_j, \quad 0 < r_j < r_{j-1} \\
r_{j-1} &= r_jq_{j+1}
\end{aligned}$$

Tada je $nzd(b, c)$ jednak r_j, odnosno posljednjem ostatku različitom od nule.

Kao primjer pogledajmo traženje najvećeg zajedničkog djelitelja brojeva $b = 973$ i $c = 301$.

$$\begin{aligned}
973 &= 3 \cdot 301 + 70 \\
301 &= 4 \cdot 70 + 21 \\
70 &= 3 \cdot 21 + 7 \\
21 &= 3 \cdot 7 + 0
\end{aligned}$$

Dobili smo $nzd(973, 301) = 7$

Prošireni Euklidov algoritam

Često će biti potrebno naći modularni inverz nekog broja, odnosno za dani $a, b \in \mathbb{Z}$, naći x takav da vrijedi:

$$ax \equiv 1 \pmod{b}.$$

Prošireni Euklidov algoritam za brojeve $a, b \in \mathbb{Z}$ naći će brojeve $s, t \in \mathbb{Z}$ takve da vrijedi:

$$as + bt = nzd(a, b). \tag{1.1}$$

Primijetimo, za a i b za koje vrijedi $nzd(a, b) = 1$ će vrijediti

$$as \equiv 1 \pmod{b},$$

odnosno proširenim Euklidovim algoritmom dobit ćemo traženi modularni inverz.

Ideja algoritma je izvesti klasični Euklidov algoritam uz dodatak da u svakom koraku izrazimo ostatak r_i kao linearnu kombinaciju brojeva a i b:

$$r_i = s_i \cdot a + t_i \cdot b.$$

Tim postupkom ćemo na kraju dobiti

$$r_j = nzd(a, b) = s_j \cdot a + t_j \cdot b,$$

odnosno dobit ćemo tražene s i t iz jednadžbe 1.1.

Na istom primjeru kao za Euklidov algoritam pogledajmo kako doći do brojeva s, t za koje vrijedi

$$973s + 301t = nzd(973, 301) = 7.$$

Krećemo se po prethodnom primjeru "odozdo prema gore" izražavajući ostatke r_i u svakoj jednadžbi:

$$\begin{aligned} 7 &= 70 - 3 \cdot 21 = \\ &= 70 - 3 \cdot (301 - 4 \cdot 70) = -3 \cdot 301 + 13 \cdot 70 = \\ &= -3 \cdot 301 + 13 \cdot (973 - 3 \cdot 301) = 13 \cdot 973 - 42 \cdot 301. \end{aligned}$$

Vrijedi dakle $s = 13, t = -42$, odnosno

$$973 \cdot 13 + 301 \cdot (-42) = nzd(973, 301) = 7.$$

Definicija 1.7. ***Eulerova funkcija***

Eulerovu funkciju broja n označavamo s $\varphi(n)$*, a definiramo kao broj elemenata u nizu* $1, 2, ..., n - 1$ *koji su relativno prosti s n.*

Npr. pogledajmo broj 12: on je relativno prost s brojevima 1, 5, 7, 11, dakle vrijedi $\varphi(12) = 4$. Primijetimo, za prost broj p vrijedi $\varphi(p) = p - 1$.

Teorem 1.8. ***Eulerov teorem***

Ako je $nzd(a, m) = 1$*, onda je* $a^{\varphi(m)} \equiv 1 \pmod{m}$.

1.2 Algebarski pojmovi

Definicija 1.9. ***Grupa***

Neprazan skup $(G, \circ)$, gdje je $\circ : G \times G \to G$ binarna operacija, zove se grupa ako vrijede sljedeća svojstva:

1. ***asocijativnost***

$$(\forall x, y, z \in G) \quad (x \circ y) \circ z = x \circ (y \circ z)$$

2. ***neutralni element***

$$(\exists e \in G)(\forall x \in G) \quad e \circ x = x \circ e = x$$

3. ***inverzni element***

$$(\forall x \in G)(\exists! x^{-1} \in G) \quad x \circ x^{-1} = x^{-1} \circ x = e$$

Ako još vrijedi i svojstvo

$$(\forall x, y \in G) \quad x \circ y = y \circ x,$$

tada kažemo da je grupa komutativna (ili abelova).

Kao primjer grupe možemo razmotriti $(\mathbb{Z}, +)$: u njoj je $e = 0$ neutralni element te $-z$ inverz za element $z \in G$. Štoviše, ova grupa je komutativna. Primjer strukture koja nije grupa može biti $(\mathbb{Z}\setminus\{0\}, \cdot)$: naime, u njoj ni za jedan element (osim 1 i -1) ne postoji inverzni element.

Definicija 1.10. ***Red grupe***

Za grupu $(G, \circ)$ definiramo red grupe $|G|$ kao broj elemenata u G. Ako je $|G| < \infty$, grupu zovemo konačnom; inače je G beskonačna grupa.

Definicija 1.11. ***Red elementa grupe***

Red elementa α grupe G definiramo kao najmanji $k \in \mathbb{N}$ za koji vrijedi

$$\alpha^k = \underbrace{\alpha \circ \alpha \circ \ldots \circ \alpha}_{k \text{ puta}} = e,$$

gdje je e neutralni element grupe G. Označavamo $|\alpha| = k$.

Definicija 1.12. ***Ciklička grupa, generator grupe***

Kažemo da je grupa G ciklička ako u G postoji element α takav da vrijedi:

$$|G| = |\alpha|.$$

Ako takav α postoji, zovemo ga generatorom grupe G.

Promotrimo grupu $\mathbb{Z}_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ s operacijom množenja modulo 11 i promotrimo je li element $\alpha = 2$ generator te grupe:

$$\begin{aligned} \alpha &= 2 & \alpha^6 &\equiv 9 \pmod{11} \\ \alpha^2 &= 4 & \alpha^7 &\equiv 7 \pmod{11} \\ \alpha^3 &= 8 & \alpha^8 &\equiv 3 \pmod{11} \\ \alpha^4 &\equiv 5 \pmod{11} & \alpha^9 &\equiv 6 \pmod{11} \\ \alpha^5 &\equiv 10 \pmod{11} & \alpha^{10} &\equiv 1 \pmod{11} \end{aligned}$$

Iz navedenog slijedi $|\alpha| = 10 = |\mathbb{Z}_{11}^*|$, što znači da je α generator grupe te da je $\mathbb{Z}_{11}^*$ ciklička grupa.

Propozicija 1.13. *Neka je α generator grupe Z_p^*. Tada vrijedi:*

$$\alpha^x \equiv \alpha^y \pmod{p} \iff x \equiv y \pmod{p-1}.$$

1.3 Pojmovi vezani uz eliptičke krivulje

Definicija 1.14. ***Eliptička krivulja***

Neka su $a, b \in \mathbb{F}_p$ takvi da vrijedi $4a^3 + 27b^2 \neq 0 \pmod{p}$. Eliptička krivulja nad poljem $\mathbb{F}_p$, $p > 3$, je skup svih točaka (x, y) koje zadovoljavaju sljedeću jednadžbu:

$$y^2 \equiv x^3 + a \cdot x + b \pmod{p},$$

zajedno s elementom kojeg zovemo točka u beskonačnosti te označavamo s $\mathcal{O}$.

Primjer eliptičke krivulje prikazan je na slici 1.1.

Zbrajanje točaka na eliptičkoj krivulji

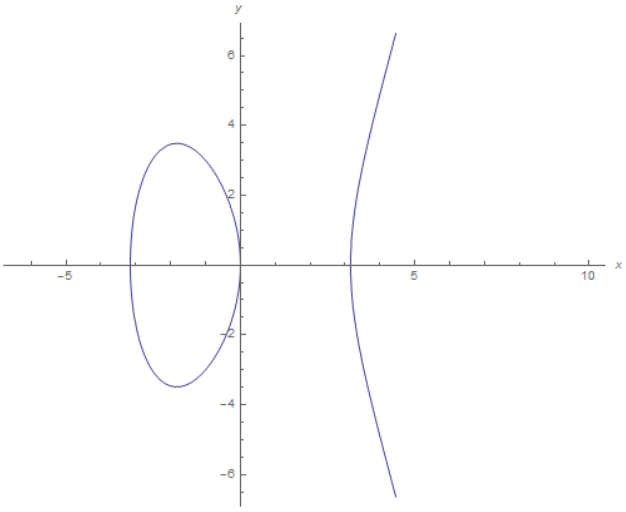
Neka su $P = (x_1, y_1), Q = (x_2, y_2) \in E$.
$P + Q = (x_0, y_0)$ definiramo razlikujući dva slučaja:

1. $P \neq Q$

$$\begin{aligned} x_0 &= \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \pmod{p} \\ y_0 &= \frac{y_2 - y_1}{x_2 - x_1}(x_1 - x_3) - y_1 \pmod{p} \end{aligned}$$

2. $P = Q$

$$x_0 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - x_1 - x_2 \pmod{p}$$
$$y_0 = \frac{3x_1^2 + a}{2y_1}(x_1 - x_3) - y_1 \pmod{p}$$



Slika 1.1: Eliptička krivulja $y^2 = x^3 - 10x$ nad poljem $\mathbb{R}$

1.4 Pojmovi vezani uz složenost algoritama

Definicija 1.15. ***Veliko O i malo o notacija***

Neka su $f, g : \mathbb{N} \to \mathbb{R}$ dvije funkcije. Tada pišemo:

1. $f(n) = O(g(n))$ *ako postoji $c > 0$ i $n_0 \in \mathbb{N}$ tako da za svaki $n \geq n_0$ vrijedi $f(n) \leq c \cdot g(n)$;*

2. $f(n) = o(g(n))$ *ako vrijedi* $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$

Definicija 1.16. ***Polinomijalan algoritam***

Algoritam zovemo polinomijalnim ako je broj operacija potrebnih za izvršavanje algoritma u najlošijem slučaju funkcija oblika $O(n^k)$, gdje je n duljina ulaznog podatka u bitovima, a k neka konstanta.

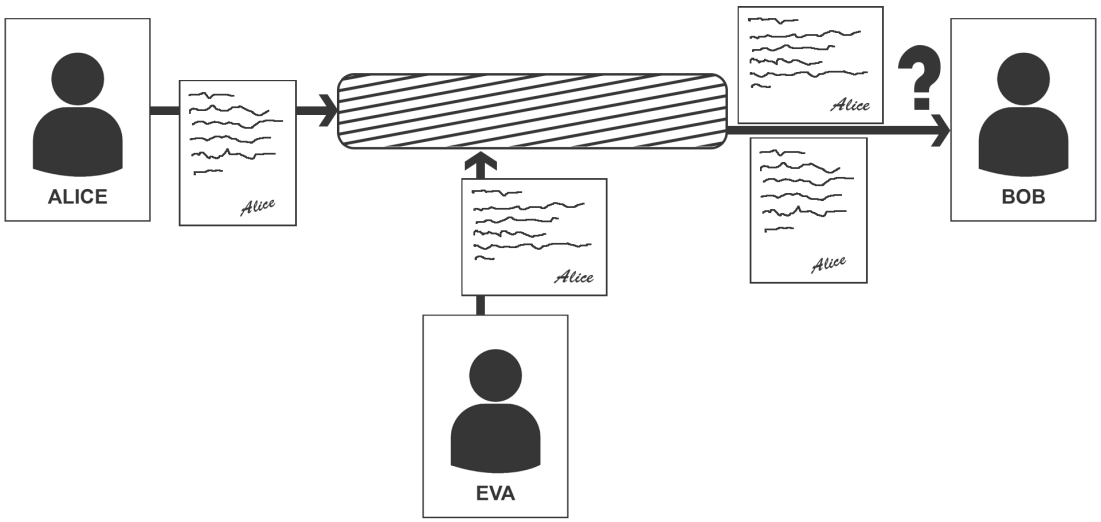
Definicija 1.17. ***Subeksponencijalni algoritam***

Subeksponencijalni algoritam je algoritam čija je složenost funkcija oblika $O(e^{o(n)})$*, gdje je n duljina ulaznog podatka.*

Poglavlje 2

Digitalni potpis općenito

Osnovni zadatak digitalnog potpisa je omogućiti autentičnost u komunikaciji između dvije osobe (zvat ćemo ih Alice i Bob) preko nekog nesigurnog kanala. Nesigurni kanal podrazumijeva mogućnost da postoji treća osoba (koju zovemo Eva), neprijatelj Alice i Boba, koja može presretati poruke unutar komunikacijskog kanala. Eva također može pokušati poslati Bobu poruku predstavljajući se kao Alice.



Slika 2.1: Ilustracija potrebe za digitalnim potpisom

Digitalni potpis će u takvim nesigurnim kanalima omogućiti utvrđivanje autentičnosti poslanog dokumenta: njime pošiljatelj može potpisati svoju poruku, tako da osoba koja prima poruku može provjeriti da je dobivena poruka zaista potekla od očekivanog pošiljatelja. Osim autentičnosti, on u komunikaciji osigurava integritet (moguće je utvrditi je li se

poruka mijenjala na putu od pošiljatelja do primatelja) te neporecivost (nakon potpisivanja pošiljatelj ne može poreći slanje poruke jer samo on posjeduje ključ kojim je poruka potpisana).

Definicija 2.1. ***Shema digitalnog potpisa***
Shema digitalnog potpisa uređena je trojka $\mathcal{DS} = (\mathcal{K}, \text{Sign}, \text{Vrfy})$ *gdje su:*

- $\mathcal{K}$ *- algoritam za generiranje ključeva (pk, sk) koje zovemo javni ključ (eng. public key) i tajni ključ (eng. secret key)*
- Sign - *algoritam potpisivanja koji prima poruku M i tajni ključ sk te vraća potpis* σ
- Vrfy - *algoritam verifikacije koji prima javni ključ pk, poruku M i potpis* σ *te vraća boolean varijablu: true ako je potpis valjan te false ako se potpis i poruka ne podudaraju*

Kažemo da je potpis valjan ako je $\text{Vrfy}_{pk}(M, \sigma) = true$.

Neka je Alice osoba koja želi potpisati neku poruku i poslati je Bobu. Ono što Alice prvo mora učiniti je generirati ključeve. Alice izvršava algoritam $\mathcal{K}$ i dobiva par ključeva (pk, sk): pk koji Alice objavljuje kao javni ključ te sk koji zadržava kao svoj tajni ključ. Sada za željenu poruku M, Alice provodi $\text{Sign}_{\text{sk}}(M)$ te dobiva potpis σ. Tada je par (M, σ) potpisana verzija poruke M koju Alice šalje Bobu.

Po primitku poruke (M', σ'), Bob koristi Alicein javni ključ pk te računa $\text{Vrfy}_{pk}(M', \sigma')$. Ako je dobivena vrijednost *true*, Bob može biti siguran da je primio Aliceinu, a ne neku drugu, možda malicioznu poruku.

Primijetimo, pretpostavljamo da Bob posjeduje Alicein javni ključ i da je siguran kako je taj ključ autentičan. Postoje mehanizmi kojima se utvrđuje autentičnost javnih ključeva i o njima će biti riječi kasnije (2.3). Zasad ćemo se držati pretpostavke da osoba koja želi provjeriti nečiji digitalni potpis posjeduje javni ključ pošiljatelja i sigurna je u njegovu autentičnost.

2.1 Primjer korištenja

Kako bismo bolje razumijeli važnost korištenja digitalnog potpisa i opasnosti koje nam prijete u komunikaciji, pogledajmo sljedeći primjer.

Neka je *ViSec* tvrtka koja razvija softver za zaštitu računala od virusa. U *ViSecu* je upravo razvijena nova verzija softvera koju je potrebno isporučiti korisnicima. Kada korisnici prime vijest o nadogradnji, nužno je da budu sigurni da je nadogradnja autentična, odnosno da nisu primili malicioznu poruku od treće osobe koja im želi podmetnuti kompromitiranu verziju softvera.

Da bi *ViSec* postigao sigurnu isporuku, treba generirati javni ključ pk i tajni ključ sk, te svojim korisnicima isporučiti pk na neki siguran način (npr. možemo pretpostaviti da su korisnici primili pk kao dio inicijalne verzije softvera). Kada je nadogradnja N spremna za isporuku, tvrtka pomoću svojeg privatnog ključa sk izračuna digitalni potpis σ te svojim korisnicima pošalje (N, σ). Svaki korisnik tada pomoću javnog ključa pk može provjeriti je li σ valjani potpis za N, odnosno je li N zaista nadogradnja koju šalje *ViSec*.

Kada bi treća osoba pokušala poslati nevaljanu verziju nadogradnje (N', σ'), gdje N' može biti i tek malo izmijenjena verzija originalne nadogradnje N, klijent bi provjerom potpisa σ uz javni ključ pk morao zaključiti kako je potpis nevaljan, odnosno odbiti potencijalno malicioznu nadogradnju N'.

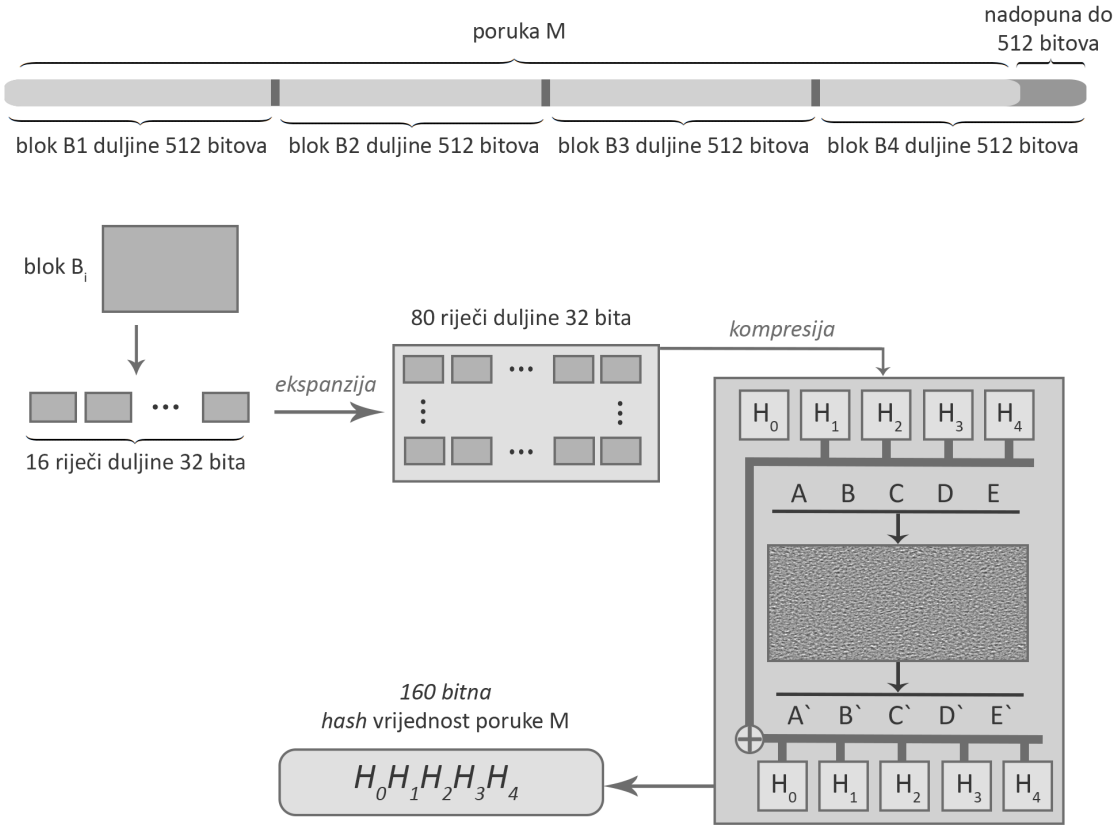
2.2 *Hash* funkcije

Važan alat u stvaranju digitalnih potpisa predstavljaju *hash* funkcije. *Hash* funkcija H je funkcija koja za poruku M proizvoljne duljine računa vrijednost $H(M)$ unaprijed određene, fiksne duljine. Za primjenu u kriptografiji na *hash* funkciju H postavit ćemo određene zahtjeve.

1. H kao ulaz prima poruku proizvoljne duljine.
2. Izlaz funkcije H je fiksne duljine.
3. Za svaki ulaz x lako je izračunati $H(x)$.
4. Za zadani y efektivno je nemoguće naći x takav da vrijedi $H(x) = y$.
5. Za zadani x_1 efektivno je nemoguće naći x_2 takav da vrijedi $H(x_1) = H(x_2)$.
6. Efektivno je nemoguće naći par (x_1, x_2) takav da je $H(x_1) = H(x_2)$.

SHA-1 algoritam

Kao primjer jedne *hash* funkcije promatrat ćemo SHA-1 (eng. *Secure Hash Algorithm*), koju je 1993. objavio NIST - američki Institut za standarde i tehnologiju (eng. *National Institute of Standards and Technology*). Potrebno je naglasiti da je SHA-1 algoritam zastario: naime, u veljači 2017. pronađena su dva PDF dokumenta s različitim sadržajem, a istom SHA-1 *hash* vrijednosti, čime je direktno narušen šesti zahtjev koji smo postavili na kriptografsku *hash* funkciju (detaljnije u [20]). U ovom ćemo radu ipak promatrati SHA-1 kao primjer hash funkcije jer su jači i sigurniji algoritmi iz SHA-2 familije temeljeni na sličnim principima.



Slika 2.2: Ilustracija postupaka u SHA-1

SHA-1 prima riječi proizvoljne duljine, a kao izlaz daje *hash* vrijednost duljine 160 bitova. Prije računanja same *hash* vrijednosti, ulaznu riječ algoritam prelama u blokove duljine 512 bitova (zadnji blok se po potrebi nadopunjuje do duljine 512). Svaki blok dijeli se na 16 riječi duljine 32 bita, a zatim se u procesu *ekspanzije* tih 16 riječi proširuje na 80 riječi iste duljine.

Sljedi proces *kompresije*: SHA-1 koristi pet registara H_0, H_1, H_2, H_3, H_4 u kojima su zapisane inicijalne vrijednosti duljine 32 bita u heksadecimalnom zapisu - suma duljina svih registara je $5 \cdot 32 = 160$ bitova. Za svaki blok B, vrijednosti registara H_0, H_1, H_2, H_3, H_4 prepišu se u varijable A, B, C, D, E te se za svaku od 80 riječi iz bloka te varijable promijene na neki zadani način. Svaki će blok dakle, ovisno o svom sadržaju, promijeniti vrijednosti u varijablama A, B, C, D, E. Po završetku kompresije svakog bloka, te se vrijednosti zbrajaju u H_0, H_1, H_2, H_3, H_4.

Kada svi blokovi poruke prođu procese ekspanzije i kompresije, konačna *hash* vrijed-

nost dobiva se konkatenacijom vrijednosti iz registara H_0, H_1, H_2, H_3, H_4 što daje, kako je ranije napomenuto, duljinu od 160 bitova. Cijeli postupak ilustriran je na slici 2.2.

2.3 Problem identiteta

Jedan od problema s kojima se sve sheme potpisivanja susreću je problem razmjene javnih ključeva. Naime, svaka od shema koje ćemo u sljedećim poglavljima obrađivati pretpostavljat će da osobe čiju komunikaciju štitimo imaju mogućnost sigurne razmjene javnih ključeva.

Međutim, treba uzeti u obzir mogućnost tzv. *Man-in-the-middle* napada u kojem Eva presretne poruku u kojoj Alice Bobu šalje javni ključ i zamijeni ga svojim javnim ključem. Eva bi tada mogla slati i potpisivati poruke u Aliceino ime: Bobov algoritam za verificiranje bi mu potvrdio kako se potpis poklapa s ključem koji posjeduje i za koji smatra da je Alicein, a koji je zapravo Evin.

Rješenje navedenog problema predstavlja korištenje *certifikata*. Ideja certifikata je da Alice poruku u kojoj šalje svoj javni ključ Bobu potpiše - međutim, Bob bez njenog javnog ključa taj potpis neće moći verificirati. Upravo zato se uvodi treća osoba, odnosno tijelo kojem vjeruju i Bob i Alice i koje će potpisati Alicein javni ključ, to jest provjeriti njegovu valjanost. To tijelo zove se *središnji autoritet* (CA, eng. *Certification Authority*), čiji ključ Bob ima i čiji će potpis lako provjeriti.

Alternativno rješenje je korištenje *mreže povjerenja* (eng. *web of trust*) - u njoj korisnici jamče jedni za druge. Koristi se pravilo tranzitivnosti: ako Alice vjeruje Bobu, tada se pretpostavlja da može vjerovati i svima u koje Bob ima povjerenja.

2.4 Sigurnost digitalnog potpisa

Kako bi digitalni potpis zaista omogućio autentičnost, integritet i neporecivost u komunikaciji nesigurnim kanalom, nužno je osigurati ga od svih poznatih napada. Treba biti svjestan da neprijatelji, odnosno osobe koje su spremne razbijati nečji kriptosustav, ne igraju po pravilima i da je napasti moguće i nelegalnim aktivnostima poput ucjenjvivanja, podmićivanja i prijevare. U ovom se radu takvim napadima ne bavimo - proučavat ćemo isključivo napade temeljene na kriptoanalizi.

U nastavku navodimo Kerckhoffov princip koji je 1883. iznio Auguste Kerckhoff, a koji bi svi bolji kriptosustavi morali poštovati. Iako u radu proučavamo isključivo dio kriptografije koji se bavi digitalnim potpisima, ovdje dajemo Kerckhoffov princip u izvornom obliku, imajući na umu da su ekvivalenti algoritama enkripcije i dekripcije u našem slučaju algoritmi za potpisivanje i verifikaciju.

Definicija 2.2. ***Kerckhoffov princip***

Kriptosustav mora biti siguran i pod pretpostavkom da neprijatelj zna sve detalje njegove implementacije, osim tajnog ključa. Kriptosustav mora dakle pretpostaviti kako su neprijatelju poznati algoritmi enkripcije i dekripcije.

Ono na čemu će se temeljiti obrane digitalnog potpisa koje ćemo susretati jest olakotna okolnost da svaka komunikacija traje neko konačno vrijeme. Naime, dovoljno je osigurati da priprema napada traje toliko dugo da bude sigurno da je, u trenutku kad je neprijatelj spreman napasti, komunikacija završena. Algoritmi digitalnih potpisa koje ćemo obrađivati u ovom radu zasnovani su na jednoj od tri velike familije algoritama javnog ključa. U nastavku ćemo opisati svaku od njih, te nakon toga razmotriti kako duljina ključeva utječe na sigurnost i koja se razina sigurnosti smatra dovoljnom.

Faktorizacija velikih složenih brojeva

Faktorizacija velikih složenih brojeva smatra se teškim problemom i na njoj se zasniva, između ostalih, RSA shema potpisivanja. Naravno, postoje mnogi algoritmi koji rješavaju ovaj problem. Na primjer, jednostavna ideja jest naivna metoda faktorizacije koja podrazumijeva dijeljenje zadanog broja n sa svim prostim brojevima manjima od $\sqrt{n}$. Ono zbog čega ovaj problem smatramo teškim jest činjenica da svi poznati algoritmi zahtijevaju subeksponencijalno vrijeme za rješavanje problema u odnosu na zadani n. U nastavku dajemo tablicu s najpoznatijim algoritmima za faktorizaciju (vidi [6]) te njihovu složenost.

Metoda	**Složenost**
Naivna metoda faktorizacije	$O(\sqrt{n}\ln n)$
Pollardova ρ metoda	$O(n^{1/4}\ln n^2)$
Faktorizacija pomoću eliptičkih krivulja	$O\left(e^{\sqrt{(\log p \log\log p)(1+O(1))}}\right)$ (gdje je p najmanji faktor od n)
Metoda verižnih razlomaka	$O\left(e^{\sqrt{2\ln n \ln\ln n}}\right)$
Metoda kvadratnog sita	$O\left(e^{\sqrt{\log n \log\log n}}\right)$
Metoda sita polja brojeva	$O\left(e^{c(\log n)^{1/3}(\log\log n)^{2/3}}\right)$

Tablica 2.1: Poznate metode faktorizacije i njihove složenosti

Diskretni logaritmi

Još jedan problem koji se smatra teškim za izračunavanje je problem diskretnog logaritma (DLP, eng. *discrete logarithm problem*).

Definicija 2.3. ***Problem diskretnog logaritma u*** $\mathbb{Z}_p^*$

Neka je dana konačna ciklička grupa $\mathbb{Z}_p^*$ *reda* $|\mathbb{Z}_p^*| = p - 1$*, generator grupe* $\alpha \in \mathbb{Z}_p^*$ *te* $\beta \in \mathbb{Z}_p^*$*. Problem diskretnog logaritma je problem određivanja broja x takvog da vrijedi:*

$$\alpha^x = \underbrace{\alpha * \alpha * ... * \alpha}_{x \text{ puta}} = \beta$$

Problem diskretnog logaritma osigurat će da za velike parametre neprijatelj neće moći u dovoljno kratkom vremenu izračunati ključeve i time oštetiti komunikaciju Alice i Boba. I ovdje navodimo tablicu s poznatijim algoritmima za rješavanje problema diskretnog logaritma u grupi G reda p (vidi [15], [11]).

Metoda	**Složenost**
Napad grubom silom	$O(\|G\|)$
Shanksova metoda mali-korak veliki-korak	$O(\sqrt{\|G\|})$
Pollardova ρ metoda	$O(\sqrt{\|G\|})$
Index calculus metoda	$2^{O(\sqrt{\log p \log\log p})}$

Tablica 2.2: Poznate metode rješavanja DLP-a i njihove složenosti

Diskretni logaritam na eliptičkim krivuljama

Algoritmi za digitalni potpis zasnovani na eliptičkim krivuljama oslanjaju se na problem diskretnog logaritma na eliptičkim krivuljama (ECDLP, eng. *Elliptic Curve Discrete Logarithm Problem*). To je zapravo analogon diskretnog logaritma u $\mathbb{Z}_p^*$, u kojem je grupa $\mathbb{Z}_p^*$ zamijenjena s grupom točaka na eliptičkoj krivulji nad konačnim poljem.

Definicija 2.4. ***Problem diskretnog logaritma na eliptičkim krivuljama***

Neka je dana eliptička krivulja E nad $\mathbb{F}_p$*, te točka* $P \in E(\mathbb{F}_p)$ *reda n. Zadana je točka Q formulom*

$$Q = lP, \qquad za\ l \in 2, 3, ..., n - 1$$

Problem diskretnog logaritma na eliptičkim krivuljama je problem određivanja broja l za koji vrijedi navedena formula.

Za razliku od problema faktorizacije složenih brojeva i problema diskretnog logaritma u $\mathbb{Z}_p^*$, za rješavanje ove verzije diskretnog logaritma nije poznat subeksponencijalni algoritam. Upravo će zbog toga za jednaku razinu sigurnosti kod ECDSA biti dovoljna mnogo manja duljina ključeva nego kod ostalih algoritama koje ćemo susretati (ilustrirano u Tablici 2.3).

Duljina ključeva i razine sigurnosti

Kako je i očekivano, što dulji ključ postavimo, to su algoritmi sigurniji. Reći ćemo da algoritam ima *n-bitnu razinu sigurnosti* ako najbolji poznati napad zahtjeva 2^n koraka. Tablica 2.3 predstavlja preporučene duljine ključeva u bitovima za tri navedene familije algoritama (preuzeto iz [15]).

Familija algoritama	**Shema potpisivanja**	**Razina sigurnosti** [*bit*]			
		80	**128**	**192**	**256**
Faktorizacija velikih prirodnih brojeva	RSA	1024	3072	7680	15360
Diskretni logaritam	ElGamal, DSA	1024	3072	7680	15360
Eliptičke krivulje	ECDSA	160	256	384	512

Tablica 2.3: Duljina ključeva za sheme potpisivanja u bitovima

Postavlja se pitanje kojom razinom sigurnosti štititi potpis od potencijalnih napada. Naravno, bilo bi dobro imati što višu razinu sigurnosti, ali s njezinim povećanjem nužno se povećava i vrijeme izvršavanja algoritama za potpisivanje i verifikaciju. Na primjer, prelazak s 1024-bitnog ključa na 3076-bitni u algoritmu za potpisivanje u RSA shemi potpisivanja uzrokovao bi 27 puta sporije vrijeme izvršavanja ([15]).

U nastavku navodimo dio teksta koji je 2015. objavio američki Institut za standarde i tehnologiju (NIST), u kojem su iznesene preporuke za duljinu ključeva koji se koriste u kriptografskim algoritmima:

> **Generiranje digitalnih potpisa**
> Duljine ključeva koje osiguravaju manje od 112 bitova sigurnosti ne bi se smjele koristiti za generiranje digitalnih potpisa.
> Duljine ključeva koje osiguravaju više od 112 bitova sigurnosti su prihvatljive za korištenje u generiranju digitalnih potpisa. [14]

U daljnjim razmatranjima pretpostavljat ćemo da je duljina ključa u shemama potpisivanja ”zbrinuta”, odnosno dovoljno velika da štiti sudionike u komunikaciji od eventualnih napada.

Poglavlje 3

RSA algoritam

RSA je asimetrični algoritam koji se koristi i za kriptiranje poruka i za njihovo potpisivanje. Ime je dobio po znanstvenicima Ronu Rivestu, Adiju Shamiru i Lenu Adlemanu koji su ga opisali u članku objavljenom 1977. godine. Kasnije se pojavila informacija kako je istu tehniku koristio i Clifford Cocks 1973., koji je tada radio za GCHD (eng. *Government Communications Headquarters*), ali ju je držao tajnom do 1997.

Sigurnost RSA sustava zasnovana je na problemu faktorizacije velikih prirodnih brojeva. Ovaj se algoritam smatra jednim od najraširenijih i najpopularnijih algoritama za digitalni potpis.

3.1 Formalna definicija

Definicija 3.1. ***RSA shema potpisivanja***

Neka su p i q prosti brojevi te neka je $n = pq$*. Tada definiramo RSA shemu potpisivanja RSA =*$(\mathcal{K}, \text{Sign}, \text{Vrfy})$*:*

$$\mathcal{K} = \{(n, p, q, d, e) : n = pq, de \equiv 1 \pmod{\varphi(n)}\}$$

Ovdje (n, e) *predstavlja javni ključ (u oznaci pk), a* (p, q, d) *tajni (u oznaci sk). Za* $K \in \mathcal{K}$ *definiramo algoritme* Sign *i* Vrfy*:*

- Sign - *Za poruku M, vraća potpis* $\sigma = \text{Sign}_{sk}(M) \equiv M^d \pmod{n}$
- Vrfy - $\text{Vrfy}_{pk}(M, \sigma)$ = *true ako i samo ako je* $\sigma^e \equiv M \pmod{n}$

Provjera korektnosti

Ako Alice potpiše poruku M sa $\sigma = \text{Sign}_{sk}(M) = M^d \pmod n$ te Bob uspješno primi (M, σ), hoće li Bobov algoritam Vrfy vratiti *true*? Da bi navedeno vrijedilo, treba vrijediti

$$\text{Vrfy}_{pk}(M, \text{Sign}_{sk}(M)) = true.$$

Po definiciji algoritama Vrfy i Sign, vrijedi

$$\begin{aligned} \text{Vrfy}_{pk}(M, \text{Sign}_{sk}(M)) = true &\iff (\text{Sign}_{sk}(M))^e \equiv M \pmod n \\ &\iff M^d \pmod n^e \equiv M \pmod n \\ &\iff M^{de} \equiv M \pmod n. \end{aligned}$$

Pokažimo da zaista vrijedi $M^{de} \equiv M \pmod n$. U definiciji RSA sheme potpisivanja zahtijevali smo $de \equiv 1 \pmod{\varphi(n)}$, iz čega slijedi da postoji $k \in \mathbb{N}$ takav da je $de = k\varphi(n) + 1$.

Promotrimo sada $nzd(M, n)$, najveći zajednički djelitelj brojeva M i n. Iz definicije broja $n = pq$, gdje su p i q prosti brojevi, slijedi $nzd(M, n) \in \{1, n, p, q\}$. Pogledajmo svaki od navedenih slučajeva:

1. $nzd(M, n) = 1$

   Iz navedenog, po Eulerovom teoremu, imamo $M^{\varphi(n)} \equiv 1 \pmod n$. Slijedi

   $$M^{de} = M^{k\varphi(n)+1} = (M^{\varphi(n)})^k \cdot M \equiv M \pmod n.$$

2. $nzd(M, n) = n$

   Tada je $M^{de} \equiv 0 \equiv M \pmod n$.

3. $nzd(M, n) = p$

   Tada vrijedi $M^{de} \equiv 0 \equiv M \pmod p$. Također, zbog $n = pq$ slijedi $(M, pq) = p$, što dalje znači da je $(M, q) = 1$ (jer bi inače vrijedilo $q|M$ pa bi moralo vrijediti $(M, pq) = pq = n$). Opet, koristeći Eulerov teorem, imamo $M^{\varphi(q)} = M^{q-1} \equiv 1 \pmod q$. Sada vrijedi (primjenom činjenice da je $\varphi(n) = \varphi(pq) = (p-1)(q-1)$)

   $$M^{de} = M^{k\varphi(n)+1} = \left(M^{q-1}\right)^{(p-1)k} \cdot M \equiv M \pmod n.$$

4. $nzd(M, n) = q$

   Analognim postupkom kao za $nzd(M, n) = p$, dolazimo do

   $$M^{de} = M^{k\varphi(n)+1} = \left(M^{p-1}\right)^{(q-1)k} \cdot M \equiv M \pmod n.$$

U svakom slučaju vrijedi $M^{de} \equiv M \pmod n$, što potvrđuje korektnost RSA sheme potpisivanja.

3.2 Primjer korištenja

Za ilustraciju navodimo primjer korištenja RSA algoritma s malim parametrima. Pretpostavimo da Alice želi Bobu poslati poruku $M = 2810$. Da bi potpisala navedenu poruku, Alicein prvi korak jest generiranje ključeva (n, p, q, d, e).

Alice prvo mora izabrati dva prosta broja p i q, neka to budu $p = 7$ i $q = 17$. Tada je $n = pq = 119$ te $\varphi(n) = (7-1)(17-1) = 96$. Sada je potrebno izabrati broj e, koji mora biti relativno prost s $\varphi(n) = 96 = 2^5 \cdot 3$. Neka je zato $e = 77 = 7 \cdot 11$. Ostalo je još samo izračunati d. Znamo da vrijedi $de \equiv 1 \pmod{\varphi(n)} \Rightarrow 77d \equiv 1 \pmod{96}$. Koristeći prošireni Euklidov algoritam, dobivamo da je $d = 5$.

Dakle, Alicein javni ključ je $(n, e) = (119, 77)$ koji je dostupan Bobu, a tajni ključ $(p, q, d) = (7, 17, 5)$. Alice sada računa potpis $\sigma = M^d \pmod{n} = 2810^5 \pmod{119} = 82$ i šalje Bobu potpisanu poruku $(M, \sigma) = (2810, 82)$.

Bob po primitku poruke provjerava vrijedi li $\sigma^e \equiv M \pmod{n}$:

$$82^{77} \equiv 73 \pmod{119} \equiv 2810 \pmod{119}.$$

Bob sada može potvrditi da je poruku zaista poslala Alice.

Primjer s ovako malim parametrima p, q i e koristimo samo za lakše razumijevanje funkcioniranja RSA sheme potpisivanja. U stvarnim primjenama koriste se parametri s više od 100 znamenaka kako bi se osigurali od potencijalnih napada. Upravo u sljedećoj točki promatramo na koje se načine može kompromitirati sigurnost RSA algoritma.

3.3 Mogućnosti napada

Neka je Eva osoba koja želi Bobu poslati poruku predstavljajući se kao Alice. Pretpostavljamo da Eva poznaje Aliceine javne ključeve (n, e).

Napad slučajnog potpisa

Prvi trivijalni napad je generiranje slučajnog potpisa σ te računanje poruke $M = \sigma^e \pmod{n}$. Očito je da će za primljenu poruku Bobov algoritam Vrfy vratiti *true*, odnosno da će Bob biti uvjeren kako je poruku M poslala Alice. No gotovo je nemoguće da je M smislena poruka - Eva nema kontrolu nad time što će M sadržavati.

Upravo zato se napad slučajnog potpisa ne smatra opasnim. Ipak, u nekim izvorima ([11], [1]) krivotvorljivost digitalnog potpisa definirana je vrlo strogo i već ovaj napad RSA shemu potpisa klasificira kao nesigurnu metodu.

Napad razlomljenom porukom

Promotrimo još jedan jednostavni napad: ovaj put Eva ima kontrolu nad porukom koju šalje predstavljajući se kao Alice. Neka je M poruka koju Eva želi potpisati Aliceinim potpisom i poslati Bobu. Eva tada odabire proizvoljnu poruku m_1 te postavlja $m_2 = M/m_1 \pmod{n}$. Tada Eva šalje poruke m_1 i m_2 Alice i traži od nje da ih potpiše potpisima σ_1 i σ_2. U slučaju da Alice zaista potpiše te poruke, Eva vrlo lako dobiva Alicein potpis σ za svoju poruku M, jer vrijedi:

$$\sigma^e = (\sigma_1 \cdot \sigma_2)^e = (m_1^d \cdot m_2^d)^e = m_1^{de} \cdot m_2^{de} \equiv m_1 m_2 \equiv M \pmod{n}$$

Ovaj napad možda omogućuje Evi kontrolu nad porukom, ali podrazumijeva da će Eva uspjeti uvjeriti Alice da potpiše njezine poruke m_1 i m_2. Iako je vrlo malo vjerojatno da će Alice to htjeti učiniti, neki od izvora ([11], [1]) smatraju da je to rizik koji sigurna shema digitalnog potpisa ne bi smjela podnositi.

Faktorizacija parametra n

Ako bi Eva na neki način mogla saznati kako faktorizirati n, odnosno saznati p i q takve da je $n = pq$, tada bi lako izračunala $\varphi(n) = (p-1)(q-1)$ i nakon toga proširenim Euklidovim algoritmom našla d za koji vrijedi $de \equiv 1 \pmod{\varphi(n)}$.

Ono što onemogućuje korištenje ovog napada je činjenica da do današnjeg dana nije poznat polinomijalni algoritam za faktorizaciju (što znači da bi za rješavanje faktorizacije brojeva s više od 200 znamenaka čak i najbržem poznatom računalu trebalo nekoliko stotina tisuća godina).

Ipak, postoje n-ovi koji su, iako veliki, jednostavni za faktorizaciju. Takav n je onaj kojemu su p i q vrlo blizu jedan drugome ili onaj kod kojega vrijedi da $p-1$ i $q-1$ imaju samo male proste faktore. Takve parametre treba izbjegavati u implementaciji algoritma za generiranje ključeva $\mathcal{K}$.

Mali eksponent d

Prethodna točka pokazala je da moramo biti pažljivi s izborom parametra n, ali to nije dovoljno: dokazano je kako je RSA shemu moguće razbiti ako se koriste mali parametri d, odnosno e.

1990. godine Michael J. Wiener otkrio je napad na RSA koji koristi razvoj u verižni razlomak, a kojim je moguće otkriti tajni parametar d, u slučaju da je d malen u odnosu na parametar n $\left(d < \frac{1}{3}n^{0.25}\right)$. Kasnije su otkrivene i dvije varijante Wienerova napada koje uspijevaju otkriti d i kad je $d > \sqrt[4]{n}$. Ipak, svi ovi algoritmi su neprimjenjivi za $d > \sqrt{n}$, pa se RSA i dalje smatra sigurnim za $d > \sqrt{n}$.

Svi navedeni napadi sugeriraju da valja biti oprezan pri potpisivanju nepoznatih poruka i pri izboru parametara u RSA shemi potpisivanja. Ako pretpostavimo da su parametri dobro postavljeni i da je potpisivač oprezan, RSA možemo smatrati sigurnom shemom potpisivanja.

Poglavlje 4

ElGamalova shema potpisivanja

Ovaj algoritam također je dobio ime po svojem stvoritelju, Taheru ElGamalu, koji ga je opisao 1985. Kao što je bio slučaj i kod RSA, postoji verzija algoritma za enkripciju i verzija za stvaranje digitalnog potpisa - međutim, kod ElGamala ta su dva algoritma prilično različita. Sigurnost ElGamalove sheme potpisivanja zasniva se na problemu diskretnog logaritma.

4.1 Formalna definicija

Definicija 4.1. ***ElGamalova shema potpisivanja***

Neka je p prost broj. ElGamalova shema potpisivanja uređena je trojka ElGamal = $(\mathcal{K}, \text{Sign}, \text{Vrfy})$*, gdje su redom:*

$$\begin{aligned}\mathcal{K} = \{(p, d, \alpha, \beta) : &\ \alpha \text{ je generator grupe } \mathbb{Z}_p^*,\\ &d \in \{2, 3, ..., p-2\},\\ &\beta = \alpha^d \pmod{p}\}\end{aligned}$$

U ElGamalovoj shemi potpisivanja javni ključ dan je s $pk = (p, \alpha, \beta)$*, a tajni ključ je* $sk = d$*. U nastavku navodimo algoritme* Sign *i* Vrfy.

Sign - *algoritam za potpisivanje poruke M*

1. *Odaberi slučajan broj* $k \in \{0, 1, 2, ..., p-2\}$ *takav da je* $nzd(k, p-1) = 1$.
2. *Izračunaj potpis* $\sigma = (r, s)$ *gdje su:*

$$\begin{aligned}r &\equiv \alpha^k \pmod{p}\\ s &\equiv (M - dr) \cdot k^{-1} \pmod{p-1}\end{aligned}$$

Vrfy - *algoritam za verifikaciju potpisa* (r, s) *poruke* M

1. *Izračunaj* $t \equiv \beta^r \cdot r^s \pmod{p}$.
2. *Ako je* $t \equiv \alpha^M \pmod{p}$*, tada je* $\mathrm{Vrfy}_{pk}(M, \sigma) = true$*, inače* $\mathrm{Vrfy}_{pk}(M, \sigma) = false$

Provjera korektnosti

Potrebno je pokazati da za poruku poslanu od očekivanog pošiljatelja algoritam za verifikaciju vraća *true*. Neka je M poruka koju Alice šalje Bobu uz potpis (r, s) nastao algoritmom za potpisivanje.

$$\begin{aligned}
\mathrm{Vrfy}_{pk}(M, \sigma) = true &\iff t \equiv \alpha^M \pmod{p} \\
&\iff \beta^r \cdot r^s \equiv \alpha^M \pmod{p} \\
&\iff (\alpha^d)^r \cdot (\alpha^k)^s \equiv \alpha^M \pmod{p} \\
&\iff \alpha^{dr+ks} \equiv \alpha^M \pmod{p}
\end{aligned}$$

Sada koristimo svojstvo generatora grupe α (1.13) koje govori da je prethodna tvrdnja istinita ako i samo ako vrijedi $dr + ks \equiv M \pmod{p-1}$. Navedeno vrijedi po definiciji parametra s : $s \equiv (M - dr) \cdot k^{-1} \pmod{p-1}$, iz čega slijedi da je algoritam verifikacije korektan.

4.2 Primjer korištenja

Navedimo primjer korištenja s malim parametrima: neka je opet Alice ta koja Bobu želi poslati poruku $M = 74$.

Alice prvo generira ključeve: odabire prosti broj $p = 29$, $\alpha = 2$, generator grupe $\mathbb{Z}_p^*$ te slučajni $d = 12$. Nakon toga, Alice računa $\beta = \alpha^d \equiv 7 \pmod{29}$. Generirani javni ključ $pk = (29, 2, 7)$ Alice šalje Bobu, dok tajni ključ $sk = 12$ zadržava za sebe. Na Slici 4.1 prikazujemo proces potpisivanja, slanja i verifikacije poruke.

Alice
Izračunaj potpis $\sigma = (r, s)$ za poruku $M = 74$:

1. Odaberi $k = 5, nzd(5, 28) = 1$.
2. $r = \alpha^k = 2^5 \equiv 3 \pmod{29}$
3. $s = (M - dr) \cdot k^{-1} \equiv 38 \cdot 17 \equiv 2 \pmod{28}$

Slanje potpisane poruke $(M, (r,s)) = (74, (3,2))$ →

Bob
Provjeri odgovara li potpis $(r, s) = (3, 2)$ poruci $M = 74$:

1. $t = \beta^r \cdot r^s \equiv 7^3 \cdot 3^2 \equiv 13 \pmod{29}$.
2. $\alpha^M \equiv 2^{74} \equiv 13 \pmod{29}$.
3. $t \equiv \alpha^M \pmod{29} \Rightarrow$ Potpis je valjan.

Slika 4.1: Primjer korištenja ElGamalove sheme potpisivanja

4.3 Mogućnosti napada

Rješavanje problema diskretnog logaritma

Kada bi napadač imao brzu metodu za rješavanje problema diskretnog logaritma, odnosno kad bi mogao saznati vrijednost tajnog ključa pošiljatelja $d = \log_\alpha \beta \pmod{p}$, imao bi sve uvjete za krivotvorenje potpisa pošiljatelja na bilo kojem dokumentu. Ipak, kako smo rekli u poglavlju 2, pretpostavljamo da je duljina korištenih ključeva dovoljna da ovaj napad bude spriječen.

Ponovna upotreba slučajnog parametra k

Kada bi pošiljatelj u algoritmu potpisivanja dva puta upotrijebio isti broj k, neprijatelj bi lako došao do privatnog ključa pošiljatelja $sk = d$. Evo kako bi taj napad funkcionirao uz Evu kao neprijatelja te Alice i Boba kao pošiljatelja i primatelja. Pretpostavimo da je

Alice u dvije poruke M_1 i M_2 upotrijebila isti k za generiranje potpisa. Eva ima pristup porukama u nesigurnom kanalu, pa tako presreće dvije poruke $(M_1, (r_1, s_1))$ i $(M_2, (r_2, s_2))$. Eva primjećuje da je $r_1 = r_2$ i iz toga lako zaključuje da je Alice dva puta upotrijebila isti k.

Eva tada promatra:

$$s_1 \equiv (M_1 - dr) \cdot k^{-1} \pmod{p-1},$$

$$s_2 \equiv (M_2 - dr) \cdot k^{-1} \pmod{p-1}.$$

Za Evu ovo je sustav dvije jednadžbe s dvije nepoznanice: parametrom k i Aliceinim tajnim ključem d. Oduzme li Eva drugu jednadžbu od prve, dobiva

$$s_1 - s_2 \equiv (M_1 - M_2) \cdot k^{-1} \pmod{p-1} \Rightarrow k \equiv \frac{x_1 - x_2}{s_1 - s_2} \pmod{p-1}.$$

Ako je $nzd(s_1 - s_2, p-1) \neq 1$, jednadžba će imati više rješenja pa će Eva morati provjeriti koje je ispravno. Nakon što se domogne parametra k, Eva lako dolazi do tajnog ključa $sk = d$ kojim može potpisivati dokumente u Aliceino ime:

$$d \equiv \frac{x_1 - s_1 k}{r} \pmod{p-1}.$$

Kako bismo spriječili ovaj napad, moramo biti sigurni da je slučajno izabrani k svjež, odnosno različit od nedavno upotrijebljenih parametara k.

Napad slučajne poruke

Kao i kod RSA sheme potpisivanja, i ovdje je moguć napad slučajne poruke. Eva naime, uz poznavanje Aliceinih javnih ključeva $pk = (p, \alpha, \beta)$, može generirati slučajne i, j takve da vrijedi $nzd(j, p-1) = 1$ te izračunati potpis:

$$r \equiv \alpha^i \beta^j \pmod{p},$$
$$s \equiv -r j^{-1} \pmod{p}.$$

Nakon toga, Eva računa poruku:

$$M \equiv si \pmod{p-1}$$

te Bobu šalje $(M, (r, s))$.

Bobov algoritam za verifikaciju za ovu poruku vratit će *true*, jer vrijedi:

$$\begin{aligned} t &\equiv \beta^r \cdot r^s \pmod{p} \\ &\equiv \alpha^{dr} \cdot r^s \pmod{p} \\ &\equiv \alpha^{dr} \cdot \alpha^{(i+dj)s} \pmod{p} \\ &\equiv \alpha^{dr} \cdot \alpha^{(i+dj)(-rj^{-1})} \pmod{p} \\ &\equiv \alpha^{dr-dr} \cdot \alpha^{-rij^{-1}} \pmod{p} \\ &\equiv \alpha^{si} \pmod{p}. \end{aligned}$$

Kako je poruka $M \equiv si \pmod{p-1}$, a α generator grupe, vrijedi:

$$\alpha^M \equiv \alpha^{si} \pmod{p},$$

što je po spomenutnom razmatranju ekvivalentno parametru t, a to je uvjet pod kojim Bob prihvaća poruku kao Aliceinu.

Kao što smo i prije primijetili, ovaj napad ne smatra se toliko opasnim budući da će Bob primiti nerazumljivu poruku i vjerojatno je proglasiti pogrešnom. Ipak, bilo bi dobro koristiti neku *hash* funkciju prije potpisivanja poruke i na taj način spriječiti ovakvu vrstu napada.

Poglavlje 5

DSA - Digital Signature Algorithm

Digital Signature Algorithm posebna je varijanta poznatog ElGamalovog algoritma koju je 1991. patentirao David W. Kravitz, bivši zaposlenik američke sigurnosne agencije (NSA, eng. *National Security Agency*). Tri godine kasnije, 1994. američki institut za standarde i tehnologiju (NIST) prihvatio je DSA kao standard digitalnog potpisa (DSS, eng. *Digital Signature Standard*).

5.1 Formalna definicija

Definicija 5.1. ***DSA shema potpisivanja***

DSA shemu potpisivanja definiramo kao uređenu trojku DSA $= (\mathcal{K}, \text{Sign}, \text{Vrfy})$*:*

$$\begin{aligned}\mathcal{K} = \{(p,q,g,x,y) : \; & p = qz + 1, \text{ za neki } z \in \mathbb{N}, \\ & g \equiv h^z \pmod{p}, \text{ za } h \text{ takav da je } 1 < h < p \text{ i } h^z \pmod{p} > 1, \\ & x < g \\ & y \equiv g^x \pmod{p}\}\end{aligned}$$

U DSA algoritmu (p, q, g, y) *je javni ključ, a* x *tajni.*
Algoritme Sign *i* Vrfy *prikazujemo navodeći korake u njihovom izvršavanju:*

Sign - *algoritam za potpisivanje poruke* M

1. *generiranje slučajnog* $k \in \mathbb{Z}, 1 < k < q$
2. *izračun* $r \equiv (g^k \pmod{p}) \pmod{q}$
3. *izračun* $s \equiv k^{-1}(\text{SHA-1}(M) + xr) \pmod{q}$
4. *ako je* $r = 0$ *ili* $s = 0$*, ponoviti postupak, inače je* (r, s) *potpis*

Vrfy - *algoritam za verifikaciju potpisa* (r, s) *poruke* M

1. *provjera vrijedi li* $0 < r < q$ *i* $0 < s < q$*: ako ne, potpis nije valjan*
2. *izračun* $w \equiv s^{-1} \pmod{q}$
3. *izračun* $u_1 \equiv \text{SHA-1}(M) \cdot w \pmod{q}$
4. *izračun* $u_2 \equiv rw \pmod{q}$
5. *izračun* $v \equiv (g^{u_1} y^{u_2} \pmod{p}) \pmod{q}$
6. *ako vrijedi* $v = r$*, potpis je valjan*

Provjera korektnosti

Ako Alice Bobu pošalje poruku M s potpisom (r, s), hoće li Bobov algoritam Vrfy potvditi da je potpis valjan, odnosno vrijedi li $r = v$ (gdje su r, v varijable iz definicije DSA sheme potpisivanja)?

Najprije primijetimo:

$$s \equiv k^{-1}(\text{SHA-1}(M) + xr) \pmod{q} \quad \Rightarrow \quad k \equiv (\text{SHA-1}(M)s^{-1} + xrs^{-1}) \pmod{q}.$$

Usporedimo sada vrijednosti varijabli r i v:

$$\begin{aligned} r \equiv (g^k \pmod{p}) \pmod{q} &\overset{k \leq q}{\equiv} (g^{k \pmod{q}} \pmod{p}) \pmod{q} \\ &\equiv (g^{\text{SHA-1}(m)s^{-1} + xrs^{-1}} \pmod{p}) \pmod{q} \\ &\equiv (g^{\text{SHA-1}(m)s^{-1}} g^{xrs^{-1}} \pmod{p}) \pmod{q} \\ &\overset{def\ w,y}{\equiv} (g^{\text{SHA-1}(m)w} y^{rw} \pmod{p}) \pmod{q} \\ &\overset{def\ u_1,u_2}{\equiv} (g^{u_1} y^{u_2} \pmod{p}) \pmod{q} \equiv v. \end{aligned}$$

Dakle, Bob dobiva iste vrijednosti za r i v te može zaključiti kako je poruka zaista potekla od Alice.

5.2 Primjer korištenja

I za DSA dat ćemo pregled korištenja sva tri algoritma $\mathcal{K}$, Sign, Vrfy na malim parametrima. Napomenimo i ovdje kako se malim parametrima narušava sigurnost potpisa - oni služe isključivo za ilustraciju i bolje razumijevanje koraka algoritama.

Neka je Alice opet ta koja Bobu šalje poruku $M = 612$. Alice prvo izabire prosti broj q, na primjer $q = 509$ te p takav da je $p = qz + 1$. Neka je $z = 2$, odnosno $p = 1019$. Nadalje, Alice bira h za koji vrijedi $1 < h < p, h^z \pmod{p} = 1$ te izračunava $g \equiv h^z \pmod{p}$.

Uzmimo $h = 192$, iz čega slijedi $g \equiv 192^2 \pmod{1019} \equiv 180 \pmod{1019}$. Alice odabire i nasumični broj $x < q$, neka on bude $x = 186$. Zadnji korak generiranja ključeva je izračun parametra y: $y \equiv g^x \pmod{p} \equiv 180^{186} \pmod{1019} \equiv 371 \pmod{1019}$. Dakle, Alicein javni ključ je $(p, q, g, y) = (1019, 509, 180, 371)$, a tajni $x = 186$.

S dobivenim ključevima, Alice provodi algoritam Sign: prvo izabire slučajni $0 < k < 509$, uzmimo $k = 45$ te računa

$$r \equiv (g^k \pmod{p}) \pmod{q} \equiv (509^{45} \pmod{1019}) \pmod{509} \equiv 78 \pmod{509}.$$

Radi lakšeg računanja, u sljedećem koraku koristimo zamjenski SHA-1 (u stvarnosti će SHA-1(M) biti 160-bitni broj s kojim bi nam bilo teško računati): pretpostavit ćemo SHA-1(612) = 121. Uz tu pretpostavku Alice dalje računa

$$s \equiv k^{-1}(\text{SHA-1}(M) + xr) \pmod{q} \equiv 181 \cdot (121 + 186 \cdot 78) \pmod{509} \equiv 31 \pmod{509}.$$

Ovim postupkom Alice je izračunala svoj potpis $(r, s) = (78, 31)$ koji šalje Bobu, zajedno s porukom $M = 612$.

Po dobitku potpisane poruke, Bob prvo provjerava vrijedi li $0 < r < q$ i $0 < s < q$ te nakon obavljene provjere prelazi na sljedeći korak algoritma Vrfy, računanje vrijednosti w, u_1, u_2 i v:

$$\begin{aligned} w &\equiv s^{-1} \pmod{q} \\ &\equiv 312 \pmod{509} \end{aligned}$$

$$\begin{aligned} u_1 &\equiv \text{SHA-1}(M) \cdot w \pmod{q} \\ &\equiv 121 \cdot 312 \pmod{509} \\ &\equiv 86 \pmod{509} \end{aligned}$$

$$\begin{aligned} u_2 &\equiv rw \pmod{q} \\ &\equiv 78 \cdot 312 \pmod{509} \\ &\equiv 413 \pmod{509} \end{aligned}$$

$$\begin{aligned} v &\equiv (g^{u_1} y^{u_2} \pmod{p}) \pmod{q} \\ &\equiv (180^{86} \cdot 371^{413} \pmod{1019}) \pmod{509} \\ &\equiv 78 \pmod{509} \end{aligned}$$

Kako za dobiveni v i za r iz Aliceinog potpisa vrijedi $v = r$, Bob zaključuje da je primljenu poruku $M = 612$ zaista potpisala i poslala Alice.

5.3 Mogućnosti napada

Kako je DSA shema potpisivanja samo jedna verzija ElGamalove sheme, i ovdje su mogući napadi rješavanjem problema diskretnog logaritma (DLP) te napad uslijed višestrukog korištenja istog parametra k.

Napad slučajnom porukom kod DSA algoritma nije moguć - naime, DSA koristi *hash* funkciju SHA-1 koja taj napad onemogućava.

Poglavlje 6

ECDSA - Elliptic Curve DSA

ECDSA je, kako mu i samo ime govori, analogon DSA algoritma koji koristi eliptičke krivulje. Algoritam je predložio Scott Vanstone 1998. i iste te godine ECDSA je prihvaćen kao ISO (eng. *International Standards Organization*) standard. Za razliku od problema faktorizacije velikih cijelih brojeva i problema diskretnog logaritma nad grupom Z_p^*, za rješavanje problema diskretnog logaritma nad eliptičkim krivuljama (ECDLP, eng. *Elliptic Curve Discrete Logarithm Problem*) ne postoji subeksponencijalni algoritam. Zbog te činjenice bit će moguće koristiti manje duljine ključeva koje će i dalje jamčiti visoku razinu sigurnosti.

6.1 Formalna definicija

Definicija 6.1. ***ECDSA shema potpisivanja***

Neka je E eliptička krivulja nad $\mathbb{F}_p$, a P točka prostog reda n na $E(\mathbb{F}_p)$. Tada definiramo ECDSA shemu potpisivanja ECDSA = $(\mathcal{K}, \text{Sign}, \text{Vrfy})$*:*

$$\mathcal{K} = \{(Q, d) : \textit{slučajno odabrani } d \in \{1, 2, ..., n-1\}, Q = [d]P\}$$

Ovdje Q predstavlja javni ključ (u oznaci pk), a d tajni (u oznaci sk). Za $K \in \mathcal{K}$ algoritme Sign *i* Vrfy *prikazujemo navodeći korake u njihovom izvršavanju:*

Sign - *algoritam za potpisivanje poruke M*

1. *izbor slučajnog broja* $k \in \{1, 2, ..., n-1\}$
2. *izračun* $[k]P = (x_1, y_1)$ *i* $r \equiv x_1 \pmod{n}$
3. *izračun* $s \equiv k^{-1}(\text{SHA-1}(M) + dr) \pmod{n}$
4. *ako je* $r = 0$ *ili* $s = 0$*, vraćanje na prvi korak; inače je* (r, s) *potpis poruke M*

Vrfy - *algoritam za verifikaciju potpisa* (r, s) *poruke* M

1. *provjera vrijedi li* $r, s \in \{1, ..., n-1\}$*: ako ne, potpis nije valjan*
2. *izračun* $w \equiv s^{-1} \pmod{n}$
3. *izračun* $u_1 \equiv \text{SHA-1}(M) \cdot w \pmod{n}, u_2 \equiv rw \pmod{n}$
4. *izračun* $[u_1]P + [u_2]Q = (x_0, y_0)$
5. *izračun* $v = x_0 \pmod{n}$
6. *ako vrijedi* $v = r$*, potpis je valjan - Vrfy vraća true*
7. *ako je* $v \neq r$*, , Vrfy vraća false*

Provjera korektnosti

Potrebno je provjeriti hoće li Alicein potpis

$$(r, s) = (x_1 \pmod{n}, k^{-1}(\text{SHA-1}(M) + dr) \pmod{n})$$

zadovoljiti Bobov uvjet za prihvaćanje potpisa kao valjanog.
Primijetimo,

$$\begin{aligned} [u_1]P + [u_2]Q &\overset{def\, u_1, u_2}{=} [\text{SHA-1}(M)\ w]P + [rwd]P \\ &= [w(\text{SHA-1}(M) + rd)]P \\ &\overset{def\, s, w}{=} [s^{-1}ks]P \\ &= [k]P \end{aligned}$$

iz čega jasno slijedi da je r, prva koordinata točke $[k]P$, jednaka prvoj koordinati točke $[u_1]P + [u_2]Q$, odnosno broju v.

6.2 Primjer korištenja

Promatramo primjer Aliceinog slanja poruke potpisane ECDSA shemom potpisivanja Bobu. Kao i dosad, Alice prvo mora generirati ključeve, što u slučaju ovog algoritma najprije znači izabrati eliptičku krivulju i točku na njoj. Pretpostavimo da je Alice izabrala krivulju

$$E : y^2 = x^3 + 2x + 2 \pmod{17}$$

i točku $P = (5, 1)$ reda $n = 19$ na njoj. Dalje Alice na slučajan način određuje $d = 7$ te računa

$$Q = [d]P = 7 \cdot (5, 1) = (0, 6).$$

Alice za sebe zadržava tajni ključ $sk = d = 7$, a Bobu šalje javni $pk = Q = (0, 6)$. Na Slici 6.1. ilustriran je proces potpisivanja, slanja i verifikacije Aliceine poruke $M = 141$. Kao i kod DSA, umjesto velike 160-bitne vrijednosti SHA-1(M) koristimo zamjenski SHA-1(M) = 26.

Alice
Izračunaj potpis $\sigma = (r, s)$ za poruku $M = 141$:

1. Izaberi $k = 10$.
2. Izračunaj $[k]P = 10 \cdot (5, 1) = (7, 11)$.
3. Postavi $r = 7$.
4. Izračunaj
   $s \equiv (26 + 7 \cdot 7) \cdot 10^{-1} \equiv 17 \pmod{19}$.

Slanje potpisane poruke $(M, (r,s)) = (141, (7,17))$ →

Bob
Provjeri odgovara li potpis $(r, s) = (7, 17)$ poruci $M = 141$:

1. Vrijedi li $r, s \in \{1, ..., n - 1\}$? Ako da, nastavi dalje. Ako ne, vrati *false*.
2. Računaj $w \equiv 17^{-1} \equiv 9 \pmod{19}$.
3. Definiraj:
   $u_1 \equiv 26 \cdot 9 \equiv 6 \pmod{19}$,
   $u_2 \equiv 7 \cdot 9 \equiv 6 \pmod{19}$.
4. Izračunaj $[6](5, 1) + [6](0, 6) = (7, 11)$.
5. Vrijedi $v = 7 = r \Rightarrow$ Potpis je valjan.

Slika 6.1: Primjer korištenja ECDSA sheme potpisivanja

6.3 Mogućnosti napada

Kao što je već spomenuto, ne postoji subeksponencijalni algoritam za rješavanje problema diskretnog logaritma nad eliptičkim krivuljama, pa ECDLP smatramo teškim problemom. Ipak, postoje tipovi eliptičkih krivulja kod kojih je taj problem znatno lakši. U nastavku navodimo neka od svojstava koja treba izbjegavati kod eliptičkih krivulja koje će biti korištene u ECDSA shemi potpisivanja (preuzeto iz [4])

Nedostatak velikog prostog faktora reda grupe $\mathbb{F}_p$

Jedan od zahtjeva kako bi sigurnost ECDSA sheme potpisivanja bila na visokoj razini jest da red grupe $E(\mathbb{F}_q)$ ima barem jedan prosti faktor veći od 2^{160}. U protivnom je moguće ECDLP riješiti na primjer Pohlig-Hellmanovim algoritmom (detaljnije u [6]).

Anomalna eliptička krivulja

Eliptičku krivulju zovemo *anomalnom* ako joj je Frobeniousov trag jednak jedan:

$$t = q + 1 - |E(\mathbb{F}_q)| = 1,$$

odnosno ako je $|E(\mathbb{F}_q)| = q$. Za takvu krivulju otkriven je polinomijalni algoritam koji rješava ECDLP, pa ju nikako ne bismo smjeli koristiti u ECDSA shemi potpisivanja. Algoritme su nezavisno otkrili Smart ([18]), Satoh-Araki ([16]) i Semaev ([17]).

Supersingularna eliptička krivulja

Eliptička krivulja E nad $\mathbb{F}_q$, gdje je $q = p^k$ je *supersingularna* ako p dijeli Frobeniousov trag krivulje t. Za takve krivulje postoji algoritam koji u polinomnom vremenu reducira ECDLP na DLP u polju $\mathbb{F}_{q^2}$ i time ga čini lakšim za rješavanje. Navedeni napad otkrili su Menezes, Okamoto i Vanstone ([12]), po kojima algoritam i nosi ime *MOV-napad*.

Bibliografija

[1] M. Bellare, *Modern Cryptography, lectures - Digital Signature*, https://cseweb.ucsd.edu/~mihir/cse207/w-ds.pdf.

[2] Nacionalni CERT, *Digitalni potpis*, 2007, http://www.cert.hr/sites/default/files/CCERT-PUBDOC-2007-02-182.pdf.

[3] A. Dujella, *Teorija brojeva u kriptografiji*, 2004, https://web.math.pmf.unizg.hr/~duje/tbkript/tbkriptlink.pdf.

[4] A. Dujella, *Eliptičke krivulje u kriptografiji*, 2013, https://web.math.pmf.unizg.hr/~duje/elkript/elkripto2.pdf.

[5] A. Dujella, *Uvod u teoriju brojeva*, https://web.math.pmf.unizg.hr/~duje/utb/utblink.pdf.

[6] A. Dujella i M. Maretić, *Kriptografija*, Element Zagreb, 2007.

[7] B. Ibrahimpašić i E. Liđan, *Digitalni potpis*, sv. 10, 2010.

[8] B. Širola, *Algebarske strukture, predavanja*.

[9] D. Johnson, A. Menezes i S. Vanstone, *The elliptic curve digital signature algorithm (ECDSA)*, International journal of information security **1** (2001), br. 1, 36–63.

[10] J. Katz, *Digital Signatures*, CRC Press, 2010.

[11] J. Katz i Y. Lindell, *Introduction to Modern Cryptography*, CRC Press, 2015.

[12] A. Menezes, S. Vanstone i T. Okamoto, *Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field*, Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '91, ACM, 1991, str. 80–89.

[13] NIST, *Digital Signature Standard (DSS)*, (2013), http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[14] NIST., *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*, (2015), http://csrc.nist.gov/publications/drafts/800-131A/sp800-131a_r1_draft.pdf.

[15] C. Paar i J. Pelzl, *Understanding Cryptography: a textbook for students and practitioners*, Springer, 2010.

[16] T. Satoh i K. Araki, *Fermat Quotients and the Polynomial Time Discrete Log Algorithm for Anomalous Elliptic Curves*, Commentarii Mathematici Universitatis Sancti Pauli **47** (1998), 81–92.

[17] I. A. Semaev, *Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p*, Mathematics of Computation (1998).

[18] N. P. Smart, *The discrete logarithm problem on elliptic curves of trace one*, Journal of Cryptology **12** (1999), br. 3, 193–196.

[19] W. Stallings, *Cryptography and Network Security*, Prentice Hall, 2005.

[20] M. Stevens, E. Bursztein, P. Karpman, A. Albertini i Y. Markov, *The first collision for full SHA-1*, (2017).

Sažetak

Digitalni potpisi u komunikaciji osiguravaju autentičnost, integritet i neporecivost - svojstva koja su u doba internetske trgovine, sklapanja ugovora na daljinu i školovanja preko interneta neizmjerno važna.

Sve češća uporaba digitalnih potpisa dovodi do povećanja broja napadača i njihovih pokušaja krivotvorenja tog potpisa. Uz to, mogućnosti računala se iz godine u godinu povećavaju i vrlo je moguće da algoritmi koji se danas smatraju sigurnima za nekoliko desetljeća to više neće biti. Upravo je zbog toga važno neprestano istraživati nove načine zaštite i preispitivati jesu li sadašnji dovoljno sigurni.

U ovom radu navedeni su osnovni pojmovi vezani uz digitalni potpis, objašnjeni neki od sistema koji se koriste u komunikaciji u kojoj je prisutan digitalni potpis kao što su *hash* funkcije i digitalni certifikati te su predstavljene najpoznatije sheme za stvaranje digitalnih potpisa. Svaka od shema objašnjena je jednostavnim primjerom te su razmotreni najpoznatiji napadi na svaku od njih.

Summary

Digital signatures assure that communication between two or more parties is provided with the following basic cryptographic services: message authentication, data integrity and non-repudiation. Those features are crucial in communication nowadays, when most of the people can't imagine day without internet shopping; when business transactions are made within seconds with people from around the world; when many students attend online courses and receive online certificates as proof of their knowledge.

As digital signature usage becomes wider and bigger, there are more and more attackers that are looking for a way to forge signatures. Also, computer possibilities are growing every year: that is why we cannot be sure that the algorithms used today will be considered safe in a few decades. We must not stop questioning the security of algorithms used nowadays and exploring new means of protection.

In this thesis we will cover basic definitions and explore some of the techniques used in digital signature algorithms. In chapters 2-6 we will see into some of the most popular digital signature schemes: RSA, ElGamal's, DSA and ECDSA. For each of them, we will show a very basic example and consider the ways the enemy might try to attack the given scheme.

Životopis

Rođena sam 15. prosinca 1992. u Zagrebu, gdje sam pohađala osnovnu školu Mate Lovraka. Već u osnovnoj školi pokazujem interes za matematiku i prirodne znanosti te mi najveće izazove predstavljaju natjecanja iz prirodoslovno-matematičkog područja. Međutim, najznačajniji rezultat postižem osvojenim petim mjestom na državnom natjecanju iz francuskog jezika 2005. godine. Dvije godine kasnije upisujem XV. gimnaziju u Zagrebu u kojoj produbljujem interes za matematiku i ljubav prema njoj.

Po završetku gimnazije, odlučujem se upisati preddiplomski studij Matematike na Prirodoslovno-matematičkom fakultetu u Zagrebu te kroz studij shvaćam da me najviše privlače predmeti računarskog smjera. 2011. godine završavam preddiplomski te upisujem diplomski studij Računarstva i matematike. Na drugoj godini diplomskog studija pruža mi se prilika za odlazak na studentsku razmjenu u sklopu Erasmus+ programa koju prihvaćam, te ljetni semestar 2016. godine provodim na Sveučilištu u Wrocławu, na Fakultetu matematike i računarskih znanosti.

Paralelno uz fakultetske obaveze trudim se skupiti praktična iskustva zbog čega odrađujem studentske prakse u tvrtkama Ericsson Nikola Tesla (2014.) te HR Cloud (2017.). Uz to, 2013.-2015. bila sam aktivna članica studentske udruge eSTUDENT u kojoj također skupljam iskustva u struci (kroz članstvo u IT timu udruge) te razvijam organizacijske i komunikacijske vještine kao voditeljica tima za organizaciju natjecanja u izradi mobilnih aplikacija. 2016. aktiviram se u studentskoj udruzi AEGEE u sklopu koje sam u ožujku 2017. bila dio organizacijskog tima konferencije European Planning Meeting na mjestu voditeljice tima za prikupljanje financijskih sredstava.