RSA javni ključ

Fundak, Vlatka

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku

Vlatka Fundak

RSA javni ključ

Završni rad

Osijek, 2021

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku

Vlatka Fundak

RSA javni ključ

Završni rad

Mentor: prof.dr.sc. Ivan Matić

Osijek, 2021

Sadržaj: Primarni cilj kriptografije je omogućiti dvjema osobama komunikaciju preko nesigurnog kanala na način da ju protivnik ne može razumjeti. Tu dolazimo do potrebe za sustavom javnog ključa, te ćemo posebno proći RSA javni ključ. RSA kriptosustav omogućava sigurnu komunikaciju preko nesigurnog komunikacijskog kanala pomoću nasumičnog teksta bitova. Sigurnost RSA sustava većinski je bazirana u težini faktoriziranja velikih prirodnih brojeva, te je potrebno imati što više znanja o načinima faktorizacije brojeva, kako bi sustav bio što sigurniji. U radu ćemo detaljnije o RSA kriptosustavu te samoj sigurnosti istog.

Ključne riječi: kriptosustav, javni ključ, RSA, faktorizacija, prost broj, sigurnost, napadi

Abstract: Prime goal of cryptography is to allow two people to communicate through an unsafe channel in a way that the opponent cannot understand it. Therefore, there has become a need for the public key system, where we will concentrate on RSA public key. RSA cryptosystem allows a safe communication through an unsafe channel with a use of random bits of plaintext. Safety of the RSA system is mostly based on difficulty of factorizing large integers, so it is necessary to have as far as possible knowledge about methods of factorization, so that the system would be safer. In this paper, we will go in more detail about RSA cryptosystem and it's safety.

Keywords: cryptosystem, public key, RSA, factorization, prime number, security, attacks

Sadržaj

1 Uvod **1**

2 Nastanak javnog ključa **2**

3 RSA javni ključ **4**
3.1 Euklidov algoritam 4
3.2 Kineski teorem 4
3.3 Korijen modulo n 5
3.4 Definicija RSA kriptosustava 7
3.5 Implementacija RSA javnog ključa 8

4 Sigurnost i napadi na RSA kriptosustav **10**
4.1 Faktorizacija 10
4.1.1 Pollardov $p-1$ algoritam 10
4.1.2 Faktorizacija preko razlike kvadrata 11
4.1.3 Algoritmi za faktorizaciju u praksi 12
4.2 Računanje $\varphi(n)$ 12
4.3 Dekripcijski eksponent 13

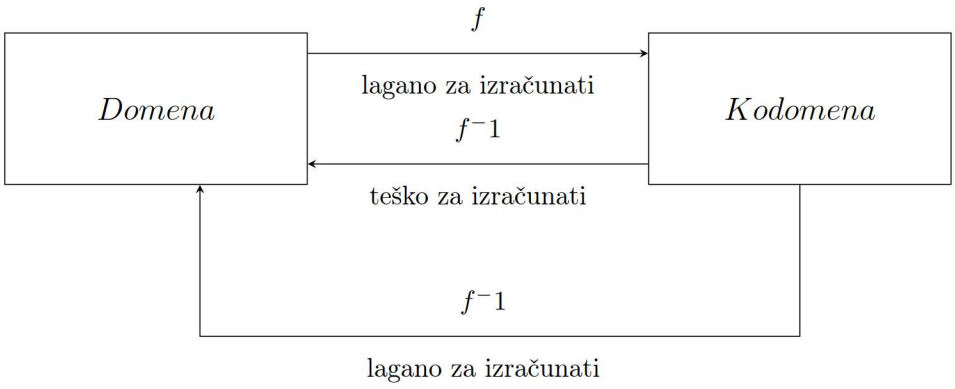
1 Uvod

U klasičnom modelu kriptografije, kada primjerice dvije osobe žele razmjeniti povjerljivu informaciju, tajno odabiru ključ K nakon čega K izaziva pravilo enkripcije e_K te pravilo dekripcije d_K. Takav način kriptosustava je poznat kao *kriptosustav simetričnog ključa*, pošto izlaganjem bilo e_K ili d_K pravila, dobivamo nesiguran sustav. Nedostatak tog sustava je što zahtjeva dijeljenje ključa K, između osoba koje su u komunikaciji, preko sigurnog kanala koji je teško postići.

Ideja kriptosustava javnog ključa je zapravo mogućnost pronalaska kriptosustava u kojem je računski neizvedivo odrediti d_K s javnim e_K pravilom. U tom slučaju je pravilo enkripcije e_K zapravo *javni ključ* koji može biti javno dostupan, primjerice u direktoriju (tu dolazimo do samog naziva sustava javnog ključa). Prednost sustava javnog ključa je slanje enkriptirane poruke drugoj strani koristeći javno pravilo enkripcije e_K bez prethodne komunikacije i tajnog odabira ključa. U tom slučaju je druga strana jedina u mogućnosti dešifrirati dobiveni šifrirani tekst koristeći pravilo dekripcije d_K zvano *privatni ključ*.

Jako bitno je naglasiti kako kriptosustav javnog ključa ne može pružiti potpunu sigurnost. Taj dio slijedi od strane protivnika koji, promatrajući šifrirani tekst y, može šifrirati svaki mogući izvorni tekst koristeći javno pravilo enkripcije e_K spomenuto ranije, dok ne nađe jedinstveni x takav da je $y = e_K(x)$. Taj x je zapravo dekripcija samog y-a.

Sustav javnog ključa možemo u okviru apstrakcije nazvati *jednosmjerna funkcija sa stupicom*. Jednosmjerna funkcija je inverzna funkcija koju je lagano za izračunati, no njen inverz je teško pronaći. Intuitivno, funkciju je generalno teško za izračunati ukoliko bi bilo koji algoritam koji služi izračunavanju inverza u razumnom vremenu, primjerice manjem od starosti svemira, gotovo sigurno podbacio. Stupica je dio pomoćne informacije koji nam dopušta lagano izračunavanje inverza funkcije. Ilustracija jednosmjerne funkcije sa stupicom se može pogledati na Slici 1.



Slika 1: Ilustracija jednosmjerne funkcije sa stupicom

2 Nastanak javnog ključa

Koncept javnog ključa kao sustava šifriranja formulirali su Whitfield Diffie i Martin Hellman 1976. godine u radu "Novi smjerovi u kriptografiji". Nisu konstruirali primjer kriptosustava javnog ključa, primarno zbog manjka informacija o odgovarajućoj stupici za funkciju, no opisali su metodu javnog ključa s kojom se određeni materijal može sigurno dijeliti preko nesigurnog kanala. Kratko vrijeme prije toga, Ralph Merkle je izolirao jedan od fundamentalnih problema te osmislio konstrukciju javnog ključa u svom radu koji nije bio u potpunosti shvaćen. Smrću James Ellisa 1997. godine, dolazi se do saznanja da je on otkrio koncept kriptografije javnog ključa još 1969. godine radeći za Komunikacijski centar Britanske vlade (GCHQ). Njegovi materijali i rad su bili tajni te su objavljeni javnosti tek nakon njegove smrti. Nakon objave rada "Novi smjerovi u kriptografiji" krenula je utrka za izumom praktičnog kriptosustava javnog ključa, te su unutar dvije godine 1977. godine Rivest, Shamir i Adleman izumili dobro poznati *RSA Kriptosustav* o kojem ćemo nešto detaljnije reći u Poglavlju 3.

Još uvijek nije dokazano da postoji prava jednosmjerna funkcija, no može se navesti primjer funkcije za koju se vjeruje da je jednosmjerna:

Uzmimo da je n produkt dvju velikih prostih brojeva p i q, te neka je b pozitivan prirodni broj. Tada definiramo funkciju $f\colon\ \mathbb{Z}_n \longrightarrow \mathbb{Z}_n$

$$f(x) = x^b \pmod{n}$$

Ukoliko je D(b,$\varphi(n)$) = 1, tada je to u pravilu funkcija enkripcije RSA sustava (pogledati Algoritam 3.1. za računanje D).

Jedni od najbitnijih kriptosustava javnog ključa su RSA i ElGammal. Iako je RSA sustav povijesno prvi kriptosustav javnog ključa, prirodniji razvoj takvog sustava baziranog na Diffie-Hellman radu je sustav opisan od strane Taher ElGammala 1985.godine. Prva objavljena konstrukcija javnog ključa je bazirana na problemu diskretnog algoritma u konačnom polju $\mathbb{F}_p$, gdje je $\mathbb{F}_p$ polje s prostim brojem elemenata. Iako Diffie-Hellman algoritam izmjene ključa omogućava javno dijeljenje nasumičnog tajnog ključa, ne postiže cilj da bude kriptosustav javnog ključa, pošto kriptosustav omogućava izmjenu specifične informacije, ne samo nasumičnog teksta bitova. ElGammal algoritam enkripcije javnog ključa je baziran na problemu diskretnog logaritma (pogledati Definiciju 2.1).

Definicija 2.1 Neka je g primitivni korijen polja, te neka je h nenul element polja $\mathbb{F}_p$. *Problem diskretnog algoritma* (DLP) je problem pronalaska eksponente x takve da je

$$g^x \equiv h \pmod{p}.$$

Broj x nazivamo *diskretni algoritam od h s bazom g* te se označava s $\log_g(h)$.

Sigurnost ElGammal kriptosustava bazirana je na problemu diskretnog algoritma, odnosno težini rješavanja jednadžbe oblika $a^x \equiv b \pmod{p}$ gdje su a, b i p poznate vrijednosti, p prost broj i x nepoznanica.

Sigurnost *RSA Kriptosustava* bazirana je u težini faktoriziranja velikih prirodnih brojeva, odnosno težini rješavanja jednadžbe oblika $x^e \equiv c \pmod{n}$ gdje su e, c i n poznate, a x nepoznata vrijednost. Drugim riječima, sigurnost je bazirana na pretpostavci da je teško

izvaditi e-te korijene modulo n. Detaljnije o sigurnosti te samom *RSA Kriptosustavu* opisujemo u nastavku.

Svakako je dobro napomenuti, koliko god je kriptosustav baziran na teškim matematičkim problemima, teško je i samo kreiranje sigurnog kanala komunikacije.

3 RSA javni ključ

Prije objašnjavanja samog rada *RSA Kriptosustava*, proći ćemo neke činjenice iz modularne aritmetike i teorije brojeva. Dva fundamentalna alata koja će nam biti potrebna su Euklidov algoritam i Kineski teorem o ostacima, te korijen modulo n koji je usko vezan za sigurnost sustava.

3.1 Euklidov algoritam

Pošto Euklidov algoritam računa najveći zajednički djelitelj, može se iskoristiti za provjeravanje ima li pozitivan broj $b <$ n multiplikativan inverz modulo n, pozivajući Euklidov algoritam (n, b) (pogledati Algoritam 3.1) i provjeravajući je li $r_m = 1$. Euklidov algoritam ne računa vrijednost $b^{-1} \pmod{n}$ (ukoliko postoji), no prošireni Euklidov algoritam računa.

Algoritam 3.1 Euklidov algoritam $\mathrm{D}(a, b)$

$r_0 \leftarrow a$
$r_1 \leftarrow b$
$m \leftarrow 1$
while $r_m \neq 0$
do $\begin{cases} q_m \leftarrow \left\lfloor \frac{r_{m-1}}{r_m} \right\rfloor \\ r_{m+1} \leftarrow r_{m-1} - q_m r_m \\ m \leftarrow m + 1 \end{cases}$
$m \leftarrow m - 1$
return $(q_1, ..., q_m; r_m)$
comment : $r_m = \mathrm{D}\ (a, b)$

3.2 Kineski teorem

Kineski teorem o ostacima je zapravo metoda rješavanja određenih sustava kongruencije, primjerice

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\vdots \\ x &\equiv a_r \pmod{m_r} \end{aligned}$$

gdje su $m_1, \ldots, m_r$ u paru relativno prosti pozitivni brojevi, tj. $\mathrm{D}(m_i, m_j) = 1$ ukoliko je $i \neq j$, te $a_1, \ldots, a_r$ prirodni brojevi.
Kineski teorem o ostacima tvrdi da gore navedeni sustav ima jedinstveno rješenje modulo $M = m_1 \times m_2 \times \cdots \times m_r$ (pogledati Teorem 3.1).

Teorem 3.1(Kineski teorem o ostacima) Uzmimo da su $m_1, \ldots, m_r$ u paru relativno prosti pozitivni prirodni brojevi, te uzmimo da su $a_1, \ldots, a_r$ prirodni brojevi. Tada je sustav od r kongruencija $x \equiv a_i \pmod{m_i} (1 \leq i \leq r)$ ima jedinstveno rješenje modulo $M = m_1 \times \cdots \times m_r$ dano izrazom

$$x = \sum_{i=1}^{r} a_i M_i y_i \pmod{M}$$

gdje su $M_i = M/m_i$ te $y_i = M^{-1} \pmod{m_i}$, za $1 \geq i \geq r$.

Također postoji efikasni algoritam za rješavanje sustava kongruencija gore navedenog tipa (pogledati Algoritam 3.2).

Algoritam 3.2 MULTIPLIKATIVNI INVERZ(a, b)

$a_0 \leftarrow a$
$b_0 \leftarrow b$
$t_0 \leftarrow 0$
$t \leftarrow 1$
$q \leftarrow \left\lfloor \frac{a_0}{b_0} \right\rfloor$
while $r > qb_0$
do $\begin{cases} temp \leftarrow (t_0 - qt) \pmod{a} \\ t_0 \leftarrow t \\ t \leftarrow temp \\ a_0 \leftarrow b_0 \\ b_0 \leftarrow r \\ q \leftarrow \left\lfloor \frac{a_0}{b_0} \right\rfloor \\ r \leftarrow a_0 - qb_0 \end{cases}$
if $b_0 \neq 1$
then b nema inverz modulo a
else return (t)

3.3 Korijen modulo n

U ovom poglavlju ćemo proći nekoliko korisnih rezultata vezanih za postojanje korijena modulo n. Uzet ćemo da je n neparan broj te $\mathrm{D}(n, a) = 1$. Prvo ćemo obratiti pozornost na broj rješenja $y \in \mathbb{Z}_n$ u kongruenciji $y^2 \equiv a \pmod{n}$. Ukoliko je n prost broj, kongruencija ili nema rješenja ili ima dva rješenja, u ovisnosti o tome je li $\left(\frac{a}{p}\right) = -1$ ili $\left(\frac{a}{p}\right) = 1$ (pogledati Teorem 3.2).

Definicija 3.1 Uzmimo da je p neparan prost broj i a prirodni broj. a definiramo kao *kvadratni ostatak* modulo p ukoliko je $a \not\equiv 0 \pmod{p}$ te kongruencija $y^2 \equiv a \pmod{p}$ ima cjelobrojno rješenje. a je *kvadratni neostatak* modulo p ukoliko je $a \not\equiv 0 \pmod{p}$ i a nije kvadratni ostatak modulo p.

Primjerom 3.1 objašnjavamo gornju definiciju.

Primjer 3.1 *U* $\mathbb{Z}_n$ *imamo* $1^2 = 1,\ 2^2 = 4,\ 3^2 = 9,\ 4^2 = 5,\ 5^2 = 3,\ 6^2 = 3,\ 7^2 = 5,\ 8^2 = 9,\ 9^2 = 4$ *te* $10^2 = 1$. *Što znači da su kvadratni ostaci modulo* 11 1, 3, 4, 5 *i* 9 *te kvadratni neostaci modulo* 11 2, 6, 7, 8 *i* 10.

Definicija 3.2 Pretpostavimo da je p neparan prost broj. Za bilo koji prirodni broj a, definiramo *Legendreov simbol* $\left(\frac{a}{p}\right)$:

$$\left(\frac{a}{p}\right) = \begin{cases} 0 \text{ za } a \equiv 0 \pmod{p}, \\ 1 \text{ ukoliko je } a \text{ kvadratni ostatak modulo } p, \\ -1 \text{ ukoliko je } a \text{ kvadratni neostatak modulo } p \end{cases}$$

Ukoliko je a višekratnik broja p, tada je $a^{(p-1)/2} \equiv 0 \pmod{p}$. Također, ukoliko je a kvadratni neostatak modulo p, tada je $a^{(p-1)/2} \equiv -1 \pmod{p}$, zato što je

$$(a^{(p-1)/2})^2 \equiv a^{p-1} \equiv 1 \pmod{p}$$

i $a^{(p-1)/2} \not\equiv 1 \pmod{p}$

Teorem 3.2 Uzmimo da je p neparan prost broj, e pozitivan prirodni broj te $\mathrm{D}(a, p) = 1$. Tada kongruencija $y^2 \equiv a \pmod{p^e}$ ima dva rješenja (*modulo* p^e) ukoliko je $\left(\frac{a}{p}\right) = 1$ ili nema rješenja ukoliko je $\left(\frac{a}{p}\right) = -1$.

Možemo primjetiti kako nam Teorem 3.2 govori da postojanje korijena modulo a može biti utvrđeno evaluacijom Legendreovog sumbola $\left(\frac{a}{p}\right)$ (pogledati Definiciju 3.2), koji je također proširenje Definicije 3.1. Ukoliko uzmemo proizvoljan neparan pozitivan broj n, uz primjenu Kineskog teorema o ostacima (pogledati Teorem 3.1) dobivamo sljedeći teorem:

Teorem 3.3 Uzmimo da je $n > 1$ neparan broj dan izrazom

$$n = \prod_{i=1}^{l} p_i^{e_i}$$

gdje su p_i različiti prosti brojevi te e_i pozitivni prirodni brojevi. Nadalje, uzmimo da je $\mathrm{D}(a, n) = 1$. Tada kongruencija $y^2 \equiv a \pmod{n}$ ima 2^l rješenja modulo n ukoliko je $\left(\frac{a}{p_i}\right) = 1$, za svaki $i \in 1, \ldots, l$, te nema rješenja u suprotnom.

3.4 Definicija RSA kriptosustava

Ovaj kriptosustav koristi operacije u $\mathbb{Z}_n$, gdje je n produkt dvaju različitih neparnih prostih brojeva p i q. Za takav prirodni broj n vrijedi $\varphi(n) = (p-1)(q-1)$.
Formalni opis *RSA Kriptosustava* je sljedeći:

Neka je $n = pq$, gdje su p i q prosti brojevi. Neka je $\mathcal{P} = \mathcal{C} = \mathbb{Z}_n$, te definiramo

$$\mathcal{K} = (n, p, q, a, b) : \ ab \equiv 1 \ (mod \ \varphi(n)).$$

Za $K = (n, p, q, a, b)$, definiramo

$$e_K(x) = x^b \pmod{n}$$

i

$$d_K(y) = y^a \pmod{n}$$

$(x, y \in \mathbb{Z}_n)$. Vrijednosti n i b obuhvaćaju javni ključ, a vrijednosti p, q i a čine privatni ključ.

Vrijednost n nazivamo modul, b *enkripcijski eksponent* te a *dekripcijski eksponent.*

Primjer 3.2 *Pretpostavimo da Bob izabere* $p = 101$ *i* $q = 113$. *Tada su* $n = 11413$ *i* $\varphi(n) = 100 \cdot 112 = 11200$. *Kako je* $11200 = 2^6 5^2 7$, *prirodni broj* b *se može koristiti kao enkripcijski eksponent ako i samo ako* b *nije djeljiv s* $2, 5$ *ili* 7. *U praksi, Bob neće faktorizirati* $\varphi(n)$. *Provjerit će da je* $D\ (\varphi(n), b) = 1$ *koristeći Algoritam 3.3. Ukoliko je to slučaj, tada će također izračunati* b^{-1}. *Pretpostavimo da Bob odabere* $b = 3533$. *Tada je*

$$b^{-1} \pmod{11200} = 6597.$$

Rezultat dekripcijskog eksponenta je $a = 6597$.
Bob objavljuje $n = 11413$ *i* $b = 3533$ *u direktorij. Sada, pretpostavimo da Alice želi enkriptirati obični tekst 9726 kako ga može poslati Bobu. Ona će izračunati*

$$9726^{3533} \pmod{11413} = 5761$$

i poslati šifrirani tekst 5761 preko kanala komunikacije. Nakon što Bob primi šifrirani tekst 5761, koristi svoj tajni dekripcijski eksponent kako bi izračunao

$$5761^{6597} \pmod{11413} = 9726.$$

3.5 Implementacija RSA javnog ključa

Postoje više aspekata *RSA Kriptosustava*, od detalja postavljanja kriptosustava, efikasnosti enkripcije i dekripcije do sigurnosnih problema.
Kako bi se *RSA Kriptosustava* mogao postaviti potrebno je sljedeće:

1. Odabir parametara (pogledati Algoritam 3.3)
2. Funkcija enkripcije
3. Funkcija dekripcije

Algoritam 3.3 RSA GENERIRANJE PARAMETARA
1. Generirati dva velika prosta broja p i q, takvi da vrijedi da $p \neq q$
2. $n \leftarrow pq$ i $\varphi(n) \leftarrow (p - 1)(q - 1)$
3. Odabrati nasumičan b ($1 < b < \varphi(n)$) takav da je D(b, $\varphi(n)$) = 1
4. $a \leftarrow b_{-1} \pmod{n}$
5. Javni ključ je (n,b) i privatni ključ je (p, q, a).

Objasnimo kratko Algoritam 3.3
U postavljanju *RSA Kriptosustava* nužno je generirati velike 'nasumične proste brojeve'. Način na koji se isto radi je nasumično generirati velike brojeve te ih testirati za prostost. U praksi, testiranje se radi koristeći Solovay-Strassen ili Miller-Rrabin algoritme. Također je potrebno voditi računa o važnom pitanju kolko je nasumičnih prirodnih brojeva dovoljno testirati dok ne nađemo prost.
Treći i četvrti korak koriste Algoritam 3.2.

Kako bi *RSA Kriptosustav* bio siguran, svakako je potrebno da $n = pq$ bude dovoljno velik da faktoriziranje istog bude računski nemoguće izračunati. Trenutno algoritmi za faktoriziranje brojeva mogu faktorizirati brojeve do 512 bita u njihovom binarnom zapisu. Općenito je preporučljivo da se odaberu p i q 512-bitni prosti brojevi, tada će n biti 1021-bitni modulo. Faktoriziranje broja ove veličine je svakako iznad kapaciteta trenutno najboljih algoritama za faktoriziranje.

Nadalje, enkripcija (ili dekripcija) uključuju potenciranje modulo n, tj. računanje funkcija oblika $x^c \pmod{n}$. Računanje $x_c \pmod{n}$ se može riješiti koristeći $c-1$ modularnih množenja, no to je dosta neefikasno ukoliko je c veliki. c može biti veličine kao $\varphi(n)$ - 1, što je skoro veliko kao n i eksponencijalno veliko u usporedbi s k.

Algoritam 3.4 Kvadriraj - množi (x, c, n)

$z \leftarrow 1$
for $i \leftarrow l - 1$ **downto** 0
do
- $z \leftarrow z^2 \pmod{n}$
- **if** $c_i = 1$
- **then** $z \leftarrow (z \cdot x) \pmod{n}$

return (z)

Algoritam Kvadriraj i množi (pogledati Algoritam 3.4) smanjuje broj modularnih multiplikacija potrebnih za računanje $x^c \pmod n$ do najviše $2l$, gdje je l broj bitova u binarnom zapisu. Pošto je n jako veliki, moramo koristiti multipreciznu aritmetiku kako bismo obavili računanja u $\mathbb{Z}_n$, te će potrebno vrijeme ovisiti o broju bitova u binarnoj reprezentaciji n-a.

Kako je prethodno spomenuto, u postavljanju *RSA Kriptosustava* potrebno je generirati velike nasumične proste brojeve. Pošto su vrijednosti nasumičnih brojeva jako velike, dolazi se do problema provjere je li odabrani broj prost. Iako je 2002. godine dokazano, od strane Agrawala, Kayala i Saxena, kako postoji deterministički algoritam za provjeru prostosti broja, zbog samog trajanja izračuna, u praksi se koriste gore spomenuti brži Solovay-Strassen ili Miller-Rrabin algoritmi. Navedeni algoritmi jesu brzi, no i dalje postoji mogućnost da algoritam tvrdi da je n prost broj iako nije. No, ako se algoritam vrti dovoljan broj puta, mogućnost za pogreškom se može smanjiti ispod bilo kojeg željenog praga.

Primjer 3.3 *Uzet ćemo da je* $n = 11413$*, te javni eksponent* $b = 3533$*. Alice enkriptira obični tekst* 9726 *računajući izraz* $9726^{3533} \pmod{11413}$*, koristeći 'Kvadriraj i množi' algoritam:*

i	b_i	z
11	1	$1^2 \times 9726 = 9726$
10	1	$9726^2 \times 9726 = 2659$
9	0	$2659^2 = 5634$
8	1	$5634^2 \times 9726 = 9167$
7	1	$9167^2 \times 9726 = 4958$
6	1	$4958^2 \times 9726 = 7783$
5	0	$7783^2 = 6298$
4	0	$6298^2 = 4629$
3	1	$4629^2 \times 9726 = 10185$
2	1	$10185^2 \times 9726 = 105$
1	0	$105^2 = 11025$
0	1	$11025^2 \times 9726 = 5761$

gdje smo dobili tekst 9726.

4 Sigurnost i napadi na RSA kriptosustav

Kako smo ranije spomenuli, *RSA Kriptosustav* je baziran na teškim matematičkim problemima. Ukoliko se jedan od ključnih problema može izračunati, dolazimo do proboja u sigurnost sustava, gdje dolazimo do potencijalnog izlijeva informacija neželjenim stranama. Kako bi se minimalizirala mogućnost takvih situacija, provode se razna istraživanja te matematički izračuni na osnovu kojeg se pokušava doći do samih rješenja gore navedenih problema, tj. ciljano se napada kriptosustav radi kvalitetnije i bolje zaštite i sigurnosti. U nastavku ćemo se dotaknuti nekoliko načina s kojima se može napasti sustav.

4.1 Faktorizacija

S obzirom da uvijek postoji mogućnost napada kriptosustava općenito, jedan od očitih napada *RSA Kriptosustava* je faktoziriziranje javnog modula. Nauka o faktorizaciji brojeva datira od Antičke Grčke, no tek dolaskom računala su ljudi počeli razvijati algoritme sposobne faktorizirati jako velike brojeve. Kako bi RSA bio više učinkovit, želimo odabrati modul $n = pq$ da bude što manji. S druge strane, ukoliko protivnik može faktorizirati n, tada naša enkriptirana poruka nije sigurna, što je sam paradoks u RSA sustavu. Stoga je od vitalne važnosti razumjeti koliko je teško faktorizirati velike brojeve, i posebno, razumjeti kapacitete različitih algoritama koji se trenutno koriste za faktoriziranje brojeva. U ovom poglavlju će se spomenuti nekoliko takvih algoritama.

Kroz ovo poglavlje ćemo reći da je n neparan prirodni broj. Ukoliko je n složen, tada je lagano vidjeti kako n ima proste faktore $p \leq \lfloor \sqrt{n} \rfloor$. Tu se može koristiti jednostavna metoda podjele, koja se sastoji od podjele broja n svakim prostim brojem sve do $\lfloor \sqrt{n} \rfloor$, gdje je dovoljno zaključiti je li n prost ili složen. Navedena metoda se u pravilu koristi za brojeve $n \leq 10^{12}$, no za veće n je potrebno koristiti malo sofisticiranije tehnike koje opisujemo u nastavku.

4.1.1 Pollardov $p-1$ algoritam

Algoritam koji koji nije koristan za sve brojeve, no poprilično je efikasan za određeni tip brojeva koji na prvi pogled djeluju sigurno.

Imamo broj $n = pq$ gdje je potrebno izračunati proste faktore p i q. Pretpostavimo da smo uspjeli pronaći prirodni broj l (pukom srećom možda ili teškim radom), sa svojstvom:

$$p-1 \text{ dijeli } l \text{ te } q-1 \text{ ne dijeli } l.$$

To znači da postoje prirodni brojevi i, j i k, gdje je $k \neq 0$ te vrijedi:

$$l = i(p-1) \text{ i } l = j(q-1) + k.$$

Uzimajući u obzir što će se dogoditi ukoliko se nasumično odabere prirodni broj a i izračuna a^l. Fermatov teorem nam kaže sljedeće:

$$a^l = a^{i(p-1)} = (a^{p-1})^i \equiv 1^i \equiv 1 \pmod{p},$$

$$a^l = a^{j(q-1)+k} = a^k (a^{q-1})^j \equiv a^k \cdot 1^j \equiv a^k \pmod{q}.$$

Eksponent k je različit od 0, pa je malo vjerojatno da će a^k biti kongruentan 1 modulo q, pa dolazimo do toga da za većinu odabranih vrijednosti a, vrijedi:

$$p \text{ dijeli } a^l - 1 \text{ te } q \text{ ne dijeli } a^l - 1.$$

To je odlična informacija pošto to znači da možemo dobiti p preko računanja najvećeg zajedničkog djelitelja:

$$p = \mathrm{D}\ (a^l - 1, n).$$

Preostaje pronaći odgovarajući l koji je djeljiv s $p-1$, no ne i s $q-1$. Pollardova observacija kaže, ukoliko je $p-1$ produkt puno manjih prostih brojeva, tada će dijeliti $n!$ za neke ne prevelike vrijednosti n. Ideja je da za svaki broj $n = 2, 3, 4, ...$ odaberemo vrijednost od a i izračunamo $\mathrm{D} = (a^{n!} - 1, n)$. Ukoliko je D jednak 1, tada prelazimo na sljedeću vrijednost od n. Ukoliko D bude vrijednosti n, tada odabiremo drugačiju vrijednost a s kojom možemo doći do gore navedenog rezultata, te ukoliko dobijemo broj između 1 i n, tada imamo netrivijalne faktore od n, gdje se završava s računanjem.

Kako bi se izbjegle opasnosti Pollardove $(p-1)$ metode, dovoljno je provjeriti da odabrani prosti brojevi p i q imaju svojstvo da se $p-1$ i $q-1$ ne mogu zapisati kao produkt malih prostih brojeva.

4.1.2 Faktorizacija preko razlike kvadrata

Jedna od najučinkovitijih današnjih metoda faktorizacije se bazira na jednom od jednostavnijih identiteta u cijeloj matematici:

$$X^2 - Y^2 = (X + Y)(X - Y)$$

Ova formula kaže da je razlika kvadrata jednaka produktu, gdje je primjenjivost faktorizaciji instantna. Kako bismo faktorizirali broj n, gledamo prirodni broj b takav da je vrijednost od $n + b^2$ potpuni kvadrat, primjerice, jednak a^2. Tada je $n + b^2 = a^2$, te se dobiva

$$n = a^2 - b^2 = (a + b)(a - b)$$

gdje je faktorizacija postignuta.
U ovoj metodi je potrebno pronaći pametan način kako odabrati b, nakon čega je dovoljno izračunati $D(n, a + b)$ i $D(n, a - b)$, pa tako umjesto da se traži razlika kvadrata $a^2 - b^2$ - višekratnici od n, tražimo različite brojeve a i b koji zadovoljavaju kongruenciju:

$$a^2 \equiv b^2 \pmod{n}.$$

U praksi nije izvedivo direktno pronaći brojeve a i b koji zadovoljavaju gore navedenu kongruenciju, te se koristi proces od tri koraka opisan u Tablici 1. Ovaj proces čini podlogu većini modernih metoda faktorizacije.

Tablica 1
1. **Izgradnja relacija**: Pronaći puno prirodnih brojeva $a_1, a_2, a_3, \ldots, a_r$ sa svojstvom da se količina $c_i \equiv a_i^2 \pmod{n}$ faktorizira kao produkt malih prostih brojeva
2. **Eliminacija**: Uzmimo produkt $c_{i_1} c_{i_2} \cdots c_{i_s}$ nekih c_i tako da svaki prost broj koji se pojavljuje u produktu, pojavljuje se u parnoj potenciji. Tada je $c_{i_1} c_{i_2} \cdots c_{i_s} = b^2$ potpuni kvadrat.
3. **Računanje najvećeg zajedničkog djelitelja D**: Neka su $a = a_{i_1} a_{i_2} \cdots a_{i_s}$ i $d = \mathrm{D}(n, a - b)$. Pošto je

$$a^2 = (a_{i_1} a_{i_2} \cdots a_{i_s})^2 \equiv a_{i_1}^2 a_{i_2}^2 \cdots a_{i_s}^2 \equiv c_{i_1} c_{i_2} \cdots c_{i_s} \equiv b^2 \pmod{n}$$

postoji razumna šansa da je d netrivijalan faktor od n.

4.1.3 Algoritmi za faktorizaciju u praksi

Posebno, algoritam široko korišten u praksi je *Kvadratno sito*. Sam naziv dolazi od procedure sijanja koja je korištena za određivanje vrijednosti $z^2 \pmod n$ koje faktoriziraju $\mathcal{B}$. *Sito s brojnim poljima* je nešto noviji algoritam za faktorizaciju od 1980-ih. On također faktorizira n konstruirajući kongruenciju $x^2 \equiv y^2 \pmod n$ koristeći svojstva algebarskih prstena prirodnih brojeva. Također možemo spomenuti algoritam *Eliptična krivulja* koji je koristan ukoliko su prosti faktori od n različitih veličina.

Za faktoriziranje RSA modula ($n = pq$, gdje su p i q prosti brojevi približno istih vrijednosti), *Kvadratno sito* je bio najčešće korišten algoritam do sredine 90-ih. Od tri navedena algoritma, *Sito s brojnim poljima* je najkasnije razvijen, s prednošću što je asimptotsko vrijeme izvršavanja algoritma brže od prethodna dva. Također, pokazalo se da je navedeni algoritam brži za brojeve koji imaju više od 125-130 znamenki.

RSA je 2001. godine iznio nove faktorizacijske izazove, gdje su veličine uključenih brojeva mjerene u bitovima, umjesto decimalnim brojevima. Postoje osam brojeva u novom RSA izazovu zapisanih na sljedeći način: RSA-576, RSA-640, RSA-704, RSA-768, RSA-896, RSA-1021, RSA-1536 i RSA-2048. Postoje nagrade za faktoriziranje tih brojeva, u vrijednosti od \$10,000 do \$200,000 dolara. Primjerice, faktorizacija RSA-576 je realizirana 2003. godine koristeći *Generalno sito s brojnim poljima.*

4.2 Računanje $\varphi(n)$

Računanje $\varphi(n)$ nije nešto lakše od samog faktoriziranja n-a. Ukoliko su n i $\varphi(n)$ poznati, te je n produkt dvaju prostih brojeva p i q, tada se n može lagano faktorizirati rješavajući sustav jednadžbi

$$n = pq$$

$$\varphi(n) = (p-1)(q-1)$$

za nepoznate p i q. Ukoliko uvedemo supstituciju $q = n/p$ u drugu jednadžbu, dobivamo kvadratnu jednadžbu s nepoznanicom p:

$$p^2 - (n - \varphi(n) + 1)p + n = 0.$$

Dva korijena gore nadevenog sustava jednadžbi će biti p i q, faktori broja n. U prijevodu, ukoliko kriptoanalitičar može saznati vrijednost $\varphi(n)$, tada može faktorizirati n i probiti sustav.

Primjer 4.1 *Uzmimo da je* $n = 84773093$*, te da je protivnik saznao da je* $\varphi(n) = 84754668$*. S tom informacijom dolazimo do sljedeće kvadratne jednadžbe:*

$$p^2 - 18426p + 84773093 = 0.$$

Jednadžbu možemo riješiti pomoću kvadratne formule, gdje dolazimo do dva rješenja, 9539 *i* 8887*. To su dva faktora broja* n*.*

4.3 Dekripcijski eksponent

Ukoliko je dekripcijski eksponent a poznat, tada n može biti faktoriziran u razumnom vremenu. Time možemo reći da računanje a u pravilu nije lakše nego samo faktoriziranje broja n. Ukoliko je a otkriven (slučajno ili drugačije), nije dovoljno da se odabere novi enkripcijski eksponent, mora se odabrati i novi modul n.

Tip ovog alogiritma se naziva *Las Vegas* tip, što u ovom kontekstu znači imati vjerojatnost uspjeha najmanje $1 - e$ u najgorem slučaju. Stoga, za bilo koji primjer problema, algoritam može ne dati odgovor s vjerojatnošću ne višom od e. Ako se susretnemo s takvom vrstom algoritma, dovoljno je primjeniti algoritam sve dok se ne dođe do odgovora. Vjerojatnost da algoritam vrati *nemaodgovora* m puta, uspješnost je e^m. Prosječan broj primjenjivanja algoritma kako bi se dobio odgovor je $1/(1 - e)$.

Algoritam je baziran na određenim činjenicama koje se dotiču korijena iz 1 modulo n, gdje je $n = pq$ produkt dvaju različitih neparnih prostih brojeva. $x^2 \equiv 1 \pmod{p}$ i Teorem 3.3 tvrde kako postoje četiri korijena iz 1 modulo n, gdje su od toga dva $\pm 1 \pmod{n}$; nazivaju se *trivijalni* korijeni iz 1 modulo n. Preostala dva korijena su *netrivijalna* te su ujedno i negativne vrijednosti svakog drugog modulo n.

Primjer 4.2 *Uzmimo da je* $n = 403 = 13 \cdot 31$. *Četiri korijena iz* 1 *modulo* 403 *su* $1, 92, 311$ *i* 402. *Korijen* 92 *je dobiven rješavanjem sustava jednadžbi:*

$$x \equiv 1 \pmod{13}$$
$$x \equiv -1 \pmod{31}$$

koristeći Kineski teorem o ostacima (pogledati Teorem 3.1). Netrivijalni korijen je $403 - 92 = 311$, *koji je ujedno i rješenje sustava:*

$$x \equiv 1 \pmod{13}$$
$$x \equiv -1 \pmod{31}.$$

Algoritam 4.1 pokušava faktorizirati n pronalaskom netrivijalnih korijena iz 1 modulo n. U Primjeru 4.3 ilustriramo primjenu istog.

Algoritam 4.1 RSA - Faktor (n, a, b)

comment : pretpostavljamo da je $ab \equiv 1 \pmod{\varphi(n)}$
write $ab - 1 = 2^s r, r$ neparan
odabrati w nasumično takav da je $1 \leq w \leq n - 1$
$x \leftarrow \mathrm{D}(w, n)$
if $1 < x < n$
  then return (x)
comment: x je faktor od n
$v \leftarrow w^r \pmod{n}$
if $v \equiv 1 \pmod{n}$
  then return ("failure")
while $v \not\equiv 1 \pmod{n}$
do $\begin{cases} v_0 \leftarrow v \\ v \leftarrow v^2 \pmod{n} \end{cases}$
if $v_0 \equiv -1 \pmod{n}$
  then return ("failure")
else $\begin{cases} x \leftarrow \mathrm{D}(v_0 + 1, n) \\ \textbf{return } (x) \end{cases}$
comment x je faktor od n

Primjer 4.3 *Uzmimo da je* $n = 89855713$, $b = 34986517$, $a = 82330933$ *te nasumična vrijednost* $w = 5$. *Imamo*

$$ab - 1 = 2^3 \cdot 360059073378795.$$

Tada

$$w^r \pmod{n} = 85877701.$$

Dobivamo

$$85877701^2 \equiv 1 \pmod{n}.$$

Tada će algoritam vratiti vrijednost

$$x = \mathrm{D}(85877702, n) = 9103.$$

Ovo je jedan faktor od n; *drugi je* $n/9103 = 9871$.

Literatura

[1] Jeffrey Hofstein, Jill Pipher, Joseph H. Silverman, An instroduction to Mathematical Cryptography, Springer Science+Business Media, LLC, New York, 2008.

[2] Douglas R. Stinson, Criptography, Theory and Practise, Third Edition, Taylor & Francis Group, LLC, Northwest, 2006.