Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku

Silvija Adašević

Kriptografija javnog ključa u primjeni

Diplomski rad

Osijek, 2013.

Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku

Silvija Adašević

Kriptografija javnog ključa u primjeni

Diplomski rad

Mentor: doc.dr.sc. Ivan Matić

Osijek, 2013.

Sadržaj

1 Uvod

U ovom diplomskom radu govoriti ćemo o kriptografiji, znanstvenoj disciplini koja proučava metode za slanje poruka na siguran način. Kako se sustav koji koriste ove metode dijele na simetrične i asimetrične, u ovom radu prednost ćemo dati asimetričnim kriptosustavima.

U drugom poglavlju (koje slijedi iza uvoda) objasniti ćemo povijesni razvoj, te osnovne pojmove šifre i kriptosustava.

U trećem poglavlju objašanjavamo razmijenu informacija pomoću nesigurnog komunikacijskog kanala čiji su začetnici Whitfield Diffie i Martin Hellman.

U četvrtom poglavlju dajemo algoritam za određivanje prostih brojeva (Miller- Rabinov test) i njegovu primjenu (Pretraga slučajnim odabirom uz pomoć Miller- Rabinovog testa).

U petom poglavlju govorimo što su to jaki prosti brojevi, za što su potrebni, te algoritme za njihov pronalazak (Gordnov algoritam, Mauerov algoritam).

U šestom poglavlju ćemo pokazati koji su najpoznatiji sustavi koji koriste kriptografiju sa javnim ključem, te primjere kako oni rade. Ti sustavi su RSA, Rabinov kriptosustav, te ELGamalov kriptosustav.

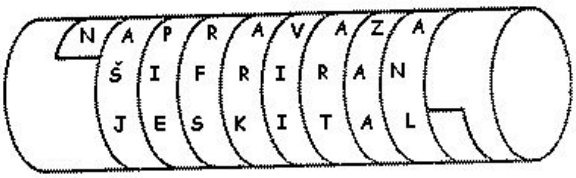
Na poslijetku, u sedmom poglavlju ćemo navesti jedan praktični primjer primjene spomenutih kriptosustava.

2 Kriptografija - pojam i upotreba

Kriptografija je znanstvena disciplina koja se bavi proučavanjem metoda za slanje poruka koje se nalaze u obliku čitljivom samo onome kome su namijenjene. Riječ kriptografija dobila je oblik iz grčkog jezika od pridjeva $\kappa\rho\upsilon\pi\tau\circ\varsigma$, (kryptos), što znači skriven i glagola $\gamma\rho\alpha\varphi\omega$ (grafo), što znači pisati. Doslovni prijevod riječi kriptografija bio bi tajnopis.

2.1 Povijesni razvoj

Kroz cijelu povijest ljudskog roda postojala je potreba za sigurnom razmijenom informacija. Elementi kriptografije bili su prisutni već kod starih Grka. Spartanci su u 5. stoljeću prije Krista koristili napravu za šifriranje skital. Skital je predstavljao drveni štap oko kojeg se namotavala vrpca od pergamenta, a na nju se okomito pisala poruka. Kada bi upisivanje poruke bilo gotovo, vrpca bi se odmotala, a na njoj bi bili izmiješani znakovi koje je mogla pročitati samo osoba koja je imala štap jednake debljine.

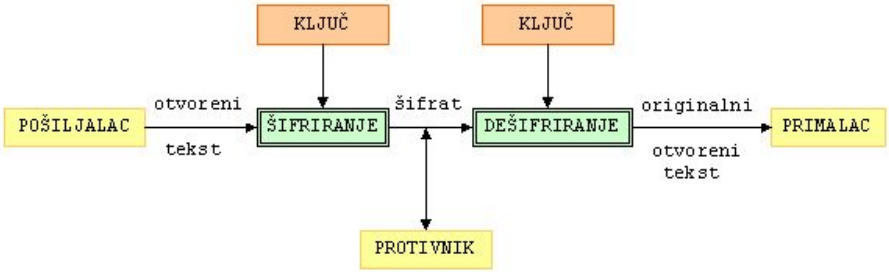


Slika 1. Skital

2.2 Osnovni zadatak

Glavni zadatak kriptografije je razmjena podataka između dviju osoba preko nesigurnog ili nezaštićenog komunikacijskog kanala (telefonska linija) takva da neka treća osoba koja može nadzirati komunikacijski kanal ne razumije njihove poruke. U literaturi su uobičajeni nazivi za osobe koje komuniciraju pošiljalac i primalac i imena Alice i Bob, dok je treća osoba njihov neprijatelj - Oskar. U nastavku ćemo prikazati pojednostavljeni način komunikacije.
Poruku koju naš pošiljalac šalje zovemo otvoreni tekst, koji može biti neki

numerički podatak, tekst na materinjem jeziku ili nešto drugo. Pošiljalac transformira otvoreni tekst koristeći dogovoreni ključ. Ovakav postupak zovemo šifriranje, a rezultat tako dobiven je šifrat ili kriptogram. Treća osoba tj. protivnik slušanjem može doznati sadržaj šifrata, ali ne i odrediti otvoreni tekst. Primalac poruke može odrediti otvoreni tekst ako zna ključ kojim je poruka šifrirana. Postupak možemo prikazati Slikom 2. Osim dešifriranja,



Slika 2.

postoji znanstvena diciplina, kriptoanaliza koja se bave proučavanjem postupaka za čitanje skrivenih poruka bez poznavanja ključa.

2.3 Šifra i kriptosustav

Šifra (kriptografski alogritam) je matematička funkcija koja se koristi za šifriranje i dešifriranje. To su dvije funkcije, jedna za šifriranje a druga za dešifriranje, koje rade tako da preslikavaju osnovne elemente otvorenog teksta u osnovne elemente šifrata i obrnuto. Ove funkcije se biraju iz familije funkcija u ovisnosti o ključu. Prostor ključeva je skup svih mogućih vrijednosti ključeva. Kriptosustav se sastoji od kriptografskog algoritma, svih mogućih tekstova, ključa i šifrata. Definirajmo sada kriptosustav.

Definicija 2.1 *Kriptosustav je uređena petorka* ***(P,C,K,E,D)*** *za koju vrijedi:*

1. ***P*** *je konačan skup svih mogućih osnovnih elemenata otvorenog teksta.*
2. ***C*** *je konačan skup svih mogućih osnovnih elemenata šifrata.*
3. ***K*** *je prostor ključeva, tj. konačan skup svih mogućih ključeva.*

4. *Za svaki* $K \in \boldsymbol{K}$ *postoji funkcija šifriranja* $e_K \in \boldsymbol{E}$ *i odgovarajuća funkcija dešifriranja* $d_K \in \boldsymbol{D}$. *Pri tome su* $e_K : \boldsymbol{P} \to \boldsymbol{C}$ *i* $d_K : \boldsymbol{C} \to \boldsymbol{P}$ *funkcije sa svojstvom da je* $d_K(e_K(x)) = x$ *za svaki otvoreni tekst* $x \in \boldsymbol{P}$.

Iz zadnjeg svojstva definicije slijedi činjenica da funckije e_K moraju biti injekcije.

2.3.1 Podjela kriptosustava

Kriptosustavi se dijele obzirom na kriterije:

1. ***Tip operacija koje se koriste pri šifriranju***
   Ovdje imamo podjelu na supstitucijske i transpozicijske šifre. Uzmimo primjer: ako riječ TAJNA šifriramo u XIWOI, imamo supstituciju, a ako ju šifriramo u JANAT imamo transpoziciju. Postoji mogućnost kombiniranja ove dvije metode.

2. ***Način na koji se obrađuje otvoreni tekst***
   Ovdje razlikujemo blok šifre, gdje se obrađuje jedan po jedan blok elemenata, te protočne šifre, gdje se obrađuje jedan po jedan element.

3. ***Tajnost i javnost ključa***
   Osnovna podjela se radi na kriptosustave s javnim ključem (asimetrične) i simetrične kriptosustave. Kod kriptosustava s javnim ključem, ključ za šifriranje je javni, a dešifritati može samo osoba koja ima odgovarajući ključ za dešifriranje. Kod simetričnih kriptosustava ključ je tajan (najčešće su ključ za šifriranje i dešifriranje identični).

3 Kriptosustavi s javnim ključem

3.1 Javni ključ

Kao što smo već naveli, kriptosustave dijelimo na simetrične i kriptosustave s javnim ključem. Za sigurnost simetričnih kriptosustava nužna je tajnost ključa, što je ujedno njih veliki minus. Kod simetričnih kriptosustava pošiljalac i primatelj trebaju razmijeniti ključ preko nekog sigurnog komunikacijskog kanala, što nije uvijek moguće. Također ključeve bi morali često

mijenjati kako se ne bi smanjila sigurnost.
Godine 1976. Whitfield Diffie[1] i Martin Hellman[2] su pronašli moguće rješenje problema razmijene ključeva putem nesigurnih komunikacijskih kanala.
Pretpostavimo da se osobe A i B žele dogovoriti o jednom tajnom slučajnom elementu u cikličkoj grupi G, kojeg bi onda poslije mogli koristiti kao ključ za šifriranje u nekom simetričnom kriptosustavu. Podsjetimo se, ciklička grupa je ona koja je generirana samo jednim elementom. Oni taj svoj dogovor moraju provesti preko nekog nesigunog komunikacijskog kanala, bez da su prethodno razmijenili bilo kakvu informaciju. Jedina informacija koju imaju jest grupa G i njezin generator g.

Diffie-Hellmanov protokol za razmjenu ključeva:

1. Osoba A generira slučajan prirodan broj $a \in \{1, 2, ...|G| - 1\}$. Ona pošalje osobi B element g^a.

2. Osoba B generira slučajan prirodan broj $b \in \{1, 2, ..., |G|-1\}$, te pošalje osobi A element g^b.

3. Osoba A izračuna $(g^b)^a = g^{ab}$.

4. Osoba B izračuna $(g^a)^b = g^{ab}$.

   Sada je njihov tajni ključ $K = g^{ab}$.

Njihov protivnik koji može prisluškivati njihovu komunikaciju preko nesigurnog komunikacijskog kanala zna sljedeće podatke: G, g, g^a, g^b, te treba iz ovih podataka izračunati g^{ab}. Ako protivnik iz poznavanja g i g^a može izračunati a, onda može pomoću a i g^b izračunati g^{ab}.
Diffie i Hellman su začetnici kriptografije javnog ključa koja ima ideju da se konstruiraju kriptosustavi kod koji ako poznajemo funkciju šifriranja e_k je gotovo nemoguće izračunati funkciju dešifriranja d_k. To znači da funkcija šifriranja može biti javna.
Definirajmo kriptosustav s javnim ključem.

[1]Bailey Whitfield Diffie, rođen 1944., američki kriptograf i jedan od pionira kriptografije javnog ključa zajedno sa Martinom Hellmanom

[2]američki kriptograf, rođen 1945.

Definicija 3.1 *Kriptosustav s javnim ključem sastoji se od dviju familija* $\{e_K\}$ *i* $\{d_K\}$ *funkcija za šifriranje i dešifriranje (gdje* K *prolazi skupom svih mogućih korisnika) sa svojstvom:*

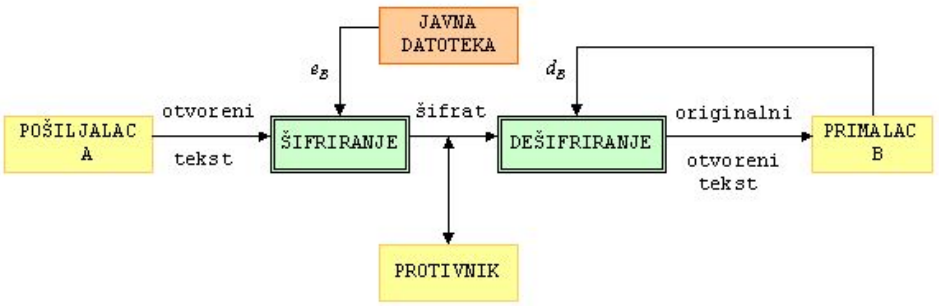
1. *Za svaki* K *je* d_K *inverz* e_K*.*

2. *Za svaki* K *je* e_K *javan, ali je* d_K *poznat samo osobi* K*.*

3. *Za svaki* K *je* e_K *osobna jednosmjerna funkcija.*

e_K *se zove javni ključ, a* d_K *tajni ili osobni ključ.*

Komunikacija pomoću javnog ključa se odvija tako da primatelj pošalje pošiljatelju svoj javni ključ, pošiljatelj šifrira svoju poruku te šalje primatelju šifrat koji primatelj dešifrira koristeći svoj tajni ključ. U ovakvoj komunikaciji može sudjelovati i grupa korisnika, gdje svi korisnici stave svoje javne ključeve u neku javnu datoteku kao što je prikazano na Slici 3.



Slika 3.

3.2 Prednosti i nedostatci

U odnosu na simetrične kriptosustave, sustavi s javnim ključem ne zahtijevaju sigurni komunikacijski kanal za razmjenu ključeva, za komunikaciju grupe od N članova potrebno je $2N$ ključeva, za razliku od $N(N-1)/2$ ključeva u simetričnom sustavu i postoji mogućnost potpisa poruke. U realnom svijetu kriptografija javnog ključa se koristi za šifriranje ključeva, a ne za

šifriranje poruka, iz razloga što su algoritmi s javnim ključem oko 1000 puta sporiji od modernih simetričnih algoritama. Još jedan nedostatak ovakvih sustava jest slabost na napad na odabrani otvoreni tekst.

4 Određivanje parametara kriptografije javnog ključa

4.1 Nasumična pretraga za vjerojatne proste brojeve

Zapišimo najprije teorem o prostim brojevima.

Teorem 4.1 *Neka je $\pi(x)$ ukupan broj prostih brojeva koji su manji ili jednaki x. Tada je $\lim_{x\to\infty} \pi(x)/(\frac{x}{\ln x}) = 1$. To znači da je za velike vrijednosti broja x, $\pi(x) \approx x/\ln x$.*

Prema prethodnom teoremu broj pozitivnih cijelih brojeva koji su manji ili jednaki broju x je približno $\pi(x)lnx$. Znamo da je jedna polovina tih brojeva parna, stoga je broj neparnih cijelih brojeva manjih ili jednakih broju x približno $\pi(x)lnx/2$. Ovo nas upućuje na logičnu metodu za odabir slučajnog k-bitnog prostog broja na način da ponavljanjem uzimamo k-bitne neparne cijele brojeve dok ne pronađemo broj koji je prost prema Miller-Rabinovom testu (algoritmu) za odgovarajuću vrijednost sigurnosnog parametra t. Miller- Rabinov test daje odgovor na pitanje '*Je li n prost?*' u obliku '*Prost*' ili '*Složen*'. Pokažimo kako izgleda taj algoritam.

Algoritam 4.1 ***MILLER- RABINOV TEST***
MILLER- RABIN (n, t)

INPUT : *neparan cijeli broj* $n \geq 3$ *i sigurnosni parametar* $t \geq 1$;

OUTPUT : *odgovor 'Složen' ili 'Prost' na pitanje 'Je li* n *prost';*

1. *napisati* $n-1$ *u obliku* $2^s r$, *gdje* r *je neparan;*

2. *za* $i = 1$ *do* t *radi:*

2.1 *izaberi slučajni cijeli broj* a, $2 \leq a \leq n-2$;

2.2 *izračunaj* $y = a^r \pmod{n}$*;*

2.3 *ako* $y \neq 1$ *i* $y \neq n-1$ *radi:*

$j \leftarrow 1$*;*

dok $j \leq s-1$ *i* $y \neq n-1$ *radi:*

izračunaj $y \leftarrow y^2 \pmod{n}$*;*

ako je $y = 1$ *vrati 'Složen';*

$j \leftarrow j+1$*;*

ako $y \neq n-1$ *vrati 'Složen';*

3. Vrati 'Prost'.

Prikažimo algoritam koji koristi Miller- Rabinov test.

Algoritam 4.2 ***PRETRAGA SLUČAJNIM ODABIROM UZ POMOĆ MILLER- RABINOVOG TESTA***

NASUMIČNA PRETRAGA (k, t)

INPUT : cijeli broj k *i sigurnosni parametar* t*;*

OUTPUT : slučajni k*-bitni prost broj;*

1. Generirajte neparan k*-bitni cijeli broj slučajnim odabirom;*

2. Koristite uzastopno dijeljenje kako bi odredili je li n *djeljiv s nekim prostim brojem manjim ili jednakim* B *(*B *je granica kod uzastopnog dijeljenja), ako je idi na korak 1.;*

3. Ako MILLER- RABINOV (n, t) *algoritam vraća vrijednost 'Prost', vrati* n*, inače idi na korak 1.*

Opišimo ukratko parametar B. Sa E označimo vrijeme koje je potrebno za k-bitno modularno potenciranje, a s D označimo vrijeme potrebno za pronalazak malog prostog djelitelja k-bitnog cijelog broja. Vrijednosti parametara E i D ovise o pojedinačnoj primjeni računa sa 'long' cijelim brojevima. Tada je parametar B, koji je granična vrijednost kod uzastopnog dijeljenja (koja

smanjuje očekivano vrijeme izvršenja Algoritma 4.2) otprilike $B = E/D$. Ako želimo eksperimentalnim putem se uvijek može odrediti bolja procjena za odabir parametra B. Neparni prosti brojevi koji su manji od B se mogu izračunati i pohraniti u tablicu. Ako imamo na raspolaganju malo memorije, za parametar B može se uzeti vrijednost koja je manja od optimalne.

Vjerojatnost da Algoritam 4.2 vraća složen broj ćemo se označiti s $p_{k,t}$. Gornje granice za parametar $p_{k,t}$ se mogu dobiti iz sljedećih uvjeta:

i) $p_{k,1} < 4^{2-\sqrt{k}}$, za $k \geq 2$.

ii) $p_{k,t} < k^{\frac{3}{2}} 2^t t^{\frac{-1}{2}} 4^{2-\sqrt{tk}}$, za ($t = 2$ i $k \geq 88$) ili ($3 \leq t \leq \frac{k}{9}$ i $k \geq 21$).

iii) $p_{k,t} < \frac{7}{20} k 2^{-5t} + \frac{1}{7} k^{\frac{15}{4}} 2^{\frac{-k}{2-2t}} + 12k2^{\frac{-k}{4-3t}}$, za $\frac{k}{9} \leq t \leq \frac{k}{4}, k \geq 21$.

iv) $p_{k,t} < \frac{1}{7} k^{\frac{15}{4}} 2^{\frac{-k}{2-2t}}$ za $t \geq \frac{k}{4}$, $k \geq 21$.

Uzmimo za primjer parametre $k = 512$ i $t = 6$, tada dobivamo rezultat $p_{512,6} \leq (\frac{1}{2})^{88}$, tj. vjerojatnost da pretraga Algoritmom 4.2 vrati 512- bitni složeni broj je manja od $(\frac{1}{2})^{88}$.

4.2 Kontroliranje greške i vjerojatnosti postizanja greške

U praksi je uobičajeno kod upotrebe Algoritma 4.2 dopustiti grešku ne veću od $(\frac{1}{2})^{80}$ pri generiranju prostih brojeva. U Tablici 1 prikazani su rezultati za parametar t koji se dobiju za proizvoljne vrijednosti parametra k, ako se koristi uvjet $p_{k,t} \leq (\frac{1}{2})^{80}$.

k	**t**
100	27
150	18
200	15
250	12
300	9
350	8
400	7
450	6

k	**t**
500	6
550	5
600	5
650	4
700	4
750	4
800	4
850	3

k	**t**
900	3
950	3
1000	3
1050	3
1100	3
1150	3
1200	3
1250	3

k	**t**
1300	2
1350	2
1400	2
1450	2
1500	2
1550	2
1600	2
1650	2

k	**t**
1700	2
1750	2
1800	2
1850	2
1900	2
1950	2
2000	2
2050	2

Tablica 1.

4.2.1 Inkrementalna pretraga

Kod izvođenja Algoritma 4.2 za generiranje prostih brojeva u prvom koraku možemo postupiti na drugi način. Najprije trebamo izabrati slučajni k-bitni neparni broj n_0 i onda testirati s takvih brojeva, $n = n_0, n_0 + 2, n_0 + 4, n_0 + 2(s-1)$. Ako su svi kandidati složeni brojevi, algoritam je podbacio. Ako je $s = c \ln 2^k$ gdje je c konstanta, vjerojatnost da inkrementalna pretraga vrati složen broj je manja od $\delta k^3 2^{-\sqrt{k}}$, za neku konstantu δ. Prednost inkrementalnog pretraživanja je ta da zahtijeva manje slučajnih bitova. Također se uzastopno dijeljenje malim prostim brojevima u drugom koraku Algoritma 4.2 može učinkovitije izvesti.

5 Jaki prosti brojevi

Prvi, a ujedno i najpopularniji i najšire korišteni kriptosustav s javnim ključem je RSA kriptosustav. Izumili su ga Ron Rivest, Adi Shamir i Len Adleman 1977. godine. Njegova sigurnost je zasnovana na teškoći faktorizacije velikih prirodnih brojeva. RSA kriptosustav koristi parametar oblika $n = pq$, gdje su p i q posebni prosti brojevi. Oni moraju biti dostatne veličine tako da je faktorizacija njihovog umnoška izvan računalnih dosega. Ako želimo još jači uvjet možemo zahtijevati da p i q budu odabrani na način da su oni funkcija nasumičnih unosa iz mora kandidata. Otkriće RSA kriptosustava dovelo je do par novih uvjeta pri odabiru brojeva p i q koji su postali neophodni kako bi se RSA kriptosustav zaštitio od kriptonapada. Ovi uvjeti su doveli do stvaranja pojma jakih prostih brojeva.

Definicija 5.1 *Za prost broj p kažemo da je jaki prosti broj, ako postoje cijeli brojevi r, s, t za koje vrijedi:*

i) $p - 1$ *ima velik prost faktor r;*

ii) $p + 1$ *ima velik prost faktor s;*

iii) $r - 1$ *ima velik prost faktor t.*

Tražena veličina prostog faktora ovisi o napadu koji će biti korišten na dani kriptosustav. Jedan od najbitnijih algoritama za pronalazak jakih prostih brojeva je Gordonov Algoritam 5.1.

Algoritam 5.1 ***GORDONOV ALGORITAM ZA PRONALAZAK JAKIH PROSTIH BROJEVA***

1. *Generirajte dva velika prosta broja* s *i* t *slučajnim odabirom, gdje se* s *i* t *sastoje od jednako bitova (jednake su duljine bitova);*

2. *Izaberite cijeli broj* i_0 *i pronađite prvi prost broj u nizu* $2it+1$*, za* $i = i_0, i_0+1, i_0+2, \ldots.$ *Označimo taj prost broj s* $r = 2it + 1$*;*

3. *Izračunajte* $p_0 = 2(s^{r-2} \ mod \ r)s - 1$*;*

4. *Izaberite cijeli broj* j_0 *i pronađite prvi prost broj u nizu* $p_0 + 2jrs$ *za* $j = j_0, j_0 + 1, j_0 + 2, \ldots$*;*
   Označimo ga sa $p = p_0 + 2jrs$*;*

5. *Vrati* (p)*.*

Kako bi smo se uvjerili da Gordonov algoritam zaista vraća jake proste brojeve, uvjerimo se najprije kako je $s^{r-1} \equiv 1 \pmod{r}$ (ova kongruencija slijedi iz Malog Fermatovog teorema koji nam govori da ako je (a, p)=1, onda je $a^{p-1} \equiv 1 \pmod{p}$, $s \neq r$. Dakle, $p_0 \equiv 1 \pmod{r}$ i $p_1 \equiv -1 \pmod{s}$. Nadalje, koristeći svojstva jakih prostih brojeva, dobivamo:

i) $p - 1 = p_0 + 2jrs - 1 \equiv 0 \pmod{r}$, stoga $p - 1$ sadrži prost faktor r;

ii) $p + 1 = p_0 + 2jrs - 1 \equiv 0 \pmod{s}$, stoga $p + 1$ sadrži prost faktor s;

iii) $r - 1 = 2it \equiv 0 \pmod{t}$, stoga $r - 1$ sadrži prost faktor t.

Prosti brojevi s i t koji su potrebni u prvom koraku mogu biti oni dobiveni upotrebom Algoritma 4.2. Nadalje, Algoritam 4.1 se može koristiti kako bi se testirali kandidati u koracima 2. i 4. nakon što izbacimo one koji su djeljivi s malim prostim brojem (koji je manji od neke granice B). Kako je Miller- Rabinov test vjerojatnosni test za proste brojeve, rezultat primjene Algoritma 5.1 je vjerojatno prost broj. Vrijeme izvršenja Gordonovog algoritma ako koristimo Miller- Rabinov test u koracima $1, 2, 4$ za pronalazak jakog prostog broja je 19% dulje od vremena izvršenja Algoritma 4.2.

5.1 Metode konstrukcija dokazivo prostih brojeva

Osim Algoritma 4.2 za pronalazak prostih brojeva, postoji još nekoliko algoritama za pronalazak istih brojeva. Jedan od takvih algoritama je i Mauerov Algoritam 5.2. On generira slučajne dokazivo proste brojeve koji su jednoliko raspoređeni u skupu prostih brojeva određene veličine. Ovaj algoritam neznatno je sporiji od Algortima 4.2 sa sigurnosnim parametrom 1. Pokažimo kako on izgleda:

Algoritam 5.2 ***MAUEROV ALGORITAM***

DOKAZIVO PROST BROJ k*;*

INPUT: pozitivan cijeli broj k*;*

OUTPUT: k*-bitni prost broj* n*;*

1. *(Ako je* k *mali broj, testirajte nasumične brojeve uzastopnim dijeljenjem.)*
   Ako je $k \geq 20$ *onda ponavljajte sljedeće korake:*

   1.1 *Odaberite nasumični* k*-bitni neparni cijeli broj* n*;*

   1.2 *Ispitati djeljivost broja* n *prostim brojevima manjim od* $\sqrt{n}$ *kako bi odredili je* n *prost;*

   1.3 *Ako je* n *prost, vrati*(n)*;*

2. *Postavite* $c \leftarrow 0.1$ *i* $m \leftarrow 20$*;*

3. *Postavite granicu* $B \leftarrow ck^2$*;*

4. *Generirajte* r*, veličinu od* q *relativnu s* n*, tj.* $r = \log q / \log n$*. Ako je* $k > 2m$ *onda ponavljajte: Izaberite nasumično broj* s *iz intervala* $[0, 1]$*, postavite* $r \leftarrow 2^{s-1}$ *sve dok je* $(k - rk) > m$*. Ako je* $(k \leq 2)$*, postavite* $r \leftarrow 0.5$*;*

5. *Izračunajte* $q \leftarrow$ *DOKAZIVO PROST BROJ* $(\lfloor rk \rfloor + 1)$*;*

6. *Postavite* $I \leftarrow \lfloor 2^{k-1}/(2q) \rfloor$*;*

7. *Uspjeh* $\leftarrow 0$*;*

8. *Sve dok je Uspjeh* $= 0$ *radite:*

8.1 *Odaberite* n*, zatim odaberite cijeli broj* R *iz intervala* $[I+1, 2I]$ *i postavite* $n \leftarrow 2Rq+1$*;*

8.2 *Pomoću uzastopnog dijeljenja odredite je li* n *djeljiv s bilo kojim prostim brojem manjim od* B*. Ako nije, ponavljajte sljedeće:*

Odaberite slučajni broj a *iz intervala* $[2, n-2]$*.*
Izračunajte $b \leftarrow a^{n-1} \pmod{n}$*;*
Ako je $b = 1$ *radite sljedeće:*
Izračunajte $b \leftarrow a^{2R} \pmod{n}$ *i* $d \leftarrow (b-1, n)$*;*
Ako je $d = 1$ *onda Uspjeh* $\leftarrow 1$*;*

9. *Vrati*(n)*.*

Misao vodilja za Mauerov algoritam bila je sljedeća lema:

Lema 5.1 *Neka je* $n \geq 3$ *neparni cijeli broj i pretpostavimo da je* $n = 1+2Rq$ *(*q *je prost). Nadalje, pretpostavimo da je* $q > R$*.*

i) Ako postoji cijeli broj koji zadovoljava $a^{n-1} \equiv 1 \pmod{n}$ *i* $(a^{2R-1}, n) = 1$*, tada je* n *prost.*

ii) Ako je n *prost, vjerojatnost da slučajno odabrani broj* a*,* $a \in [1, n-1]$ *zadovoljava* $a^{n-1} \equiv 1 \pmod{n}$ *i* $(a^{2R-1}, n) = 1$ *je* $1 - \frac{1}{q}$*.*

Algoritam 5.2 rekurzivno generira neparni broj q, zatim izabire nasumično brojeve R, $R < q$, sve dok se ne dokaže da je $n = 2Rq+1$ prost uz korištenje Leme 5.1.

5.1.1 Konstante c i m u Maureovom algoritmu

Optimalna vrijednost konstante c pri definiranju granice probnog dijeljenja $B = ck^2$ u drugom koraku Algoritma 5.2 ovisi o izvršenju računskih operacija s višeznamenkastim cijelim brojevima i najbolje se može odrediti eksperimentalno.

Konstanta $m = 20$ nam osigurava da je I duljine barem 20 bitova. Također jamči da je interval $[I + 1, 2I]$ odakle odabiremo R, dovoljno dug tako da se u njemu nalazi barem jedna vrijednost parametra R za koju je $n = 2Rq + 1$ prost broj.

5.1.2 Poboljšanja Mauerovog algoritma

Navedimo tri najvažnije činjenice koje mogu pridonijeti poboljšanju Algoritma 5.2.

- ✓ Brzina izvođenja se može poboljšati ako u koraku 8.2 zahtijevamo da je $q > \sqrt[3]{n}$.
- ✓ Ako u koraku 8.2 odabrani n zadovoljava postupak uzastopnog dijeljenja, tada na n možemo primijeniti Miller- Rabinov test s bazom $a = 2$, što služi u svrhu poboljšanja brzine izvođenja našeg početnog Algoritma 5.2.
- ✓ U četvrtom koraku Mauerovog algoritma zahtijeva se račun s realnim brojevima pri određivanju broja 2^{s-1}. Kako bi to izbjegli, možemo unaprijed izračunati i pohraniti listu takvih vrijednosti za nasumične brojeve $s \in [0, 1]$.

6 Kriptosustavi koji koriste metodu javnog ključa

U radu smo se do sada bavili problemom generiranja prostih brojeva i algoritmima kojima se postižu najbolji rezultati. U ovom poglavlju ćemo prikazati različite kriptosustave s javnim ključem, poznatije pod nazivom asimetrična enkripcija (šifriranje). Općenite činjenice o kriptosustavima i njihovoj podjeli smo prikazali u Poglavlju 3. Kako imamo ključeve koji su javni, logično se nameće činjenica da će protivnici izvesti kriptonapad.
Glavni cilj nekog kriptonapada je dobiti otvoreni tekst iz šifrata koji je namjenjen nekom subjektu A. Ako je kriptonapad uspješan, kažemo da je enkripcijska shema probijena. Još jedan cilj je saznati privatni ključ subjekta A i tada kažemo da je shema u cijelosti probijena jer protivnik može dešifrirati

cijelu poruku poslanu subjektu A. Kako je šifriranje javno znanje dostupno svima, protivnik uvijek može postaviti napad na neki tekst koji je šifriran pomoću javnog ključa. Jači napad bi bio onaj u kojem protivnik izabere određeni šifrat po vlastitom nahođenju i nekim metodama od subjekta A, iz šifrata dobije otvoreni tekst. Razlikujemo dvije vrste ovakvih jačih napada:

1. Sporedno odabrani napad, gdje protivnik posjeduje dešifriranje bilo kojeg šifrata po vlastitom izboru, ali šifrat mora biti odabrana prije nego protivnik dobije šifrat c koju želi dešifrirati.

2. Prilagođeni napad, gdje protivnik može imati pristup sustavu za dešifriranje koji koristi subjekt A (ali ne i privatni ključ), čak niti nakon što je saznao ciljani šifrat c. Protivnik može zatražiti dešifriranje šifrata, koja može biti povezana s ciljanim šifratom i rezultatima dobivenim iz prijašnjih upita, ali ne može zatražiti dešifriranje cilja c.

   Raspodjela javnih ključeva
   Šifriranje uz javni ključ uzima za pretpostavku činjenicu da postoji sredstvo kojim pošiljatelj dolazi u posjed autentične kopije primateljevog javnog ključa. Kada pak nemamo na raspolaganju prethodnu činjenicu, sustav je odmah osjetljiviji na imitirajuće napade (protivnik u protokol ubacuje svoj lažni javni ključ).

   Blokovi poruka
   Neki od sustava koji koriste javni ključ uzimaju za gotovo činjenicu da su poruke koje se prenose u komunikaciji fiksne veličine (odnosno duljine bitova). Obična tekstualna poruka koja je dulja od neke zadane maksimalme duljine se mora razdijeliti na blokove točno određene veličine. Tada se svaki blok poruke može zasebno šifrirati. Kada imamo poruku koju smo podijelili na blokove, za šifriranje i dešifriranje iste se moze primjeniti javni ključ.

Pokažimo sada kako izgledaju najpoznatiji sustavi koji koriste kriptografiju javnog ključa.

6.1 RSA kriptosustav

Kao što smo već rekli, RSA je najpoznatiji i najčešće korišteni sustav kriprografije javnog ključa. Može se koristiti kako bi se postigla tajnost nekih podataka i u upotrebi digitalnih potpisa. Prije nego damo definiciju samog sustava, pokažimo kako djeluje algoritam koji generira ključ.

Algoritam 6.1 ***GENERIRANJE KLJUČA ZA RSA ŠIFRIRANJE***

SAŽETAK: svaki subjekt kreira javni ključ i njemu odgovarajući tajni ključ. Da bi to napravio, subjekt A treba raditi:

1. *Generirati dva velika prosta broja* p *i* q *(međusobno različita), koji su otprilike jednake bitne duljine;*

2. *Izračunati parametre* n *i* $\varphi(n)$*, gdje je* $n = pq$ *i* $\varphi(n) = (p-1)(q-1)$*;*

3. *Izabrati cijeli broj* e *slučajnim odabirom,* $1 < e < \varphi(n)$*, takav da je* $(e, \varphi(n)) = 1$*;*

4. *Koristiti prošireni Euklidov algoritam i izračunati* d*,* $1 < d < \varphi(n)$ *takav da vrijedi* $ed \equiv 1 \pmod{\varphi(n)}$*;*

5. *Javni ključ subjekta A je* (n, e)*, a privatni ključ je* d*.*

Euklidov algoritam se koristi pri pronalasku najvećeg zajedničkog dijelitelja d, brojeva a i b,dok prošireni algoritam zadovoljava i jednakost $ax + by = d$ gdje su x, y cijeli brojevi. Parametri e i d se nazivaju vršitelj šifriranja i vršitelj dešifriranja (enkripcijski i dekripcijski eksponent), dok se n naziva modul. Budući smo sada generirali javni i tajni ključ, možemo krenuti u postupak šifriranja.

Algoritam 6.2 ***RSA ŠIFRIRANJE SA JAVNIM KLJUČEM***

SAŽETAK: subjekt B *šifrira poruku* m *za subjekt* A*, koju* A *dešifrira.*

1. *Šifriranje. B neka radi:*

   a) *Dohvati autentični javni ključ* (n, e) *subjekta* A*;*

b) Prikaži poruku kao cijeli broj m *iz intervala* $[0, n-1]$*;*

c) Izračunaj vrijednost $c = m^e \pmod{n}$*;*

d) Pošalji šifrat c *subjektu* A*;*

2. Dešifriranje. Kako bi iz šifrata c *dobio poruku* m*, A neka radi:*

a) Pomoću privatnog ključa d *odrediti* $m = c^d \pmod{n}$*.*

Dokaz. Pokažimo da dešifriranje valjano radi.
Kako je $ed \equiv 1 \pmod{\varphi(n)}$, postoji cijeli broj k, takav da je $ed = 1 + k\varphi(n)$.
Nadalje, ako je $(m, p) = 1$ tada (po Malom Fermatovom teoremu) je

$$m^{p-1} \equiv 1 \pmod{p}.$$

Ako potenciramo obje strane na potenciju $k(q-1)$ i zatim pomnožimo s m dobivamo

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{p}.$$

S druge strane, ako je $(m, p) = p$ tada je prethodna kongruencija i u ovom slučaju točna budući je svaka strana konguentna s 0 modulo p.
Dakle u svim slučajevima je

$$m^{ed} \equiv m \pmod{p}.$$

Po istom agrumentu je

$$m^{ed} \equiv m \pmod{q}.$$

Konačno, kako su p i q različiti slijedi da je

$$m^{ed} \equiv m \pmod{n}$$

i stoga

$$c^d \equiv (m^e)^d \equiv m \pmod{n}.$$

□

6.1.1 Primjena

Pogledajmo na primjeru kako radi RSA šifriranje.

Primjer 6.1 *Uzmimo za primjer prividno male parametre.*
Generiranje ključa.
Subjekt A odabire proste brojeve $p = 2357$, $q = 2551$ *i računa* $n = pq = 6012707$ *i* $\varphi(n) = (p-1)(q-1) = 6007800$. *A odabire e i pomoću proširenog Euklidovog algoritma, izračunava* $d = 422191$ *tako da vrijedi* $ed \equiv 1 \pmod{\varphi(n)}$. *Javni ključ subjekta A je par (*$n = 6012707$, $e = 3674911$*), a privatni ključ je* $d = 422191$.
Šifriranje.
Kako bi šifrirali poruku $m = 5234673$, *B koristi algoritam za modularno potenciranje kako bi izračunao*

$$c = m^e \pmod{n} = 5234673^{3674911} \pmod{6012707} = 3650502$$

i šalje c prema A.
Dešifriranje.
Kako bi dešifrirao c A računa

$$c^d \pmod{n} = 3650502^{422191} \pmod{6012707} = 5234673.$$

Napomenimo kako se nekada umjesto parametra φ može koristiti parametar $\lambda = (p-1, q-1)$ koji se često naziva univerzalni eksponent od n. Možemo uočiti kako je λ pravi dijelitelj od φ. Prednost korištenja parametra λ je ta što dobijemo manju vrijednost za vršitelja dešifriranja d i na taj način može se ubrzati sam postupak. Ako se p i q biraju slučajnim odabirom, $(p-1, q-1)$ je mali pa su tada φ i λ otprilike jednake veličine.
Jedan od algoritam koji se može koristi kod modularnog potenciranja je i slijedeći:

Algoritam 6.3 ***ALGORITAM ZA POTENCIRANJE U*** $\mathbb{Z}_n$

INPUT: $a \in \mathbb{Z}_n$ *i cijeli broj* $0 \leq k \leq n$, *čiji je binarni zapis* $\Sigma_{i=0}^{t} k_i 2^i$.

OUTPUT: $ak \pmod{n}$.

1. *Postavi* $b \leftarrow 1$. *Ako je* $k = 0$ *onda vrati* (b).

2. *Postavi* $A \leftarrow a$.

3. *Ako je* $k_0 = 1$ *onda postavi* $b \leftarrow a$.

4. *Za* i *od 1 raditi slijedeće:*

4.1 *Postavi* $A \leftarrow A^2 \pmod{n}$.

4.2 *Ako je* $k_i = 1$ *onda postavi* $b \leftarrow Ab \pmod{n}$.

5. *Vrati* (b).

6.1.2 Sigurnost RSA kriptosustava

Postoje mnoga sigurnosna pitanja kada se radi o procesu šifriranja. Nabrojimo neke osnovne vrste napada i obrane od istih s kojima se može pojedinac susresti pri RSA šifriranju.

Veza sa faktorizacijom

Problem s kojim se susreće protivnik je dobivanje poruke m iz odgovarajućeg šifrata c kada je poznat javni ključ (n, e) nekog subjekta A. Jedno od mogućih rješenja za ovaj problem bi bilo najprije faktorizirati parametar n a potom izračunati φ i d kao u Algoritmu 6.1. Kada izračuna d, protivnik može dešifrirati bilo koju šifriranu poruku najmijenjenu subjektu A.

Mali vršitelj šifriranja e

Napomenimo najprije kako je zbog bolje učinkovitosti šifriranja poželjno odabrati za vršitelja šifriranja mali broj (npr. $e = 3$). No, ako odaberemo ovako mali parametar e dolazimo do velike nesigurnosti ako npr. želimo prema više subjekata poslati poruku istog sadržaja.

Forward napadi

Ako je prostor za poruku malog kapaciteta, protivnik može dešifrirati šifriranu poruku c na jednostavan način tako da šifrira sve poruke dok ne dobije šifrat c.

Mali vršitelj dešifriranja d

Kao što smo naveli za vršitelja šifriranja e, možemo pomisliti kako bi za d bilo bolje izabrati malu vrijednost (u svrhu poboljšanja sustava). Postoji učinkovit algoritam s kojim se iz javnog ključa (n, e) može izračunati d. Kako bi onemogućili ovakav napad, vršitelj šifriranja d bi trebao biti otprilike jednake veličine kao modul n.

Skrivene poruke

Za otvoreni tekst kažemo da nije skriven ako se šifrira sam u sebe. Uvijek prilikom šifriranja neke poruke ostaju neskrivene. Ipak, u praski je broj poruka koje nisu skrivene je zanemarivo mali tako da ne predstavlja prijetnju za sigurnost RSA šifriranja.

6.2 Rabinov kripotsustav

Jedna od poželjnih osobina svake sheme šifriranja je dokaz kako je razbijanje samog kriptosustava teško kao neki poznati računski problem. Za razbijanje RSA kriptosustava se vjeruje da ima težinu kao faktoriziranje modula n, što nije dokazano. Rabinov kriptosustav je bila prva dokazivo sigurna shema koja koristi kriptografiju javnog ključa. Problem koji se stavlja pred protivnike koji žele saznati otvoreni tekst iz nekog šifrata računski je ekvivalentan problemu faktorizaciji.

Algoritam 6.4 ***GENERIRANJE KLJUČA ZA RABINOV KRIPTOSUSTAV***

SAŽETAK: svaki subjekt kreira svoj javni i privatni ključ. Subjekt A neka radi:

1. *Generirati dva velika prosta broja p i q (međusobno različita) otprilike jednake veličine;*
2. *Izračunati $n = pq$;*
3. *Javni ključ od A je n a privatni ključ je (p, q).*

Algoritam 6.5 ***RABINOV KRIPTOSUSTAV***

B šifrira poruku m koju šalje subjektu A, koju potom A dešifrira.

1. *Šifriranje*
   B neka radi:
   (a) *Dohvati autentični javni ključ n subjekta A.*
   (b) *Prikaži poruku kao cijeli broj m iz intervala $\{0, 1, ..., n-1\}$.*

(c) *Izračunaj* $c = m^2 \pmod{n}$.

(d) *Pošalji šifrat* c *subjektu* A.

2. *Dešifriranje*
Kako bi došao do otvorenog teksta m *iz šifrata* c, A *neka radi:*

(a) *Pronađi kvadratne korijene* m_1, m_2, m_3, m_4 *od* c *modulo* n *(korijene pronalazimo pomoću algoritma za pronalaženje kvadratnih korijena modulo* n *kada su poznati njegovi prosti faktori* p *i* q, *čiju varijaciju ćemo opisati u nastavku).*

(b) *Poruka koja je poslana je* m_1, m_2, m_3 *ili* m_4. A *odlučuje koja je od ovih poruka poslana.*

Napomena (Upotreba zalihosti)

a) Nedostatak Rabinovog kriptosustava uz upotrebu javnog ključa je to što je primatelj primoran na odabir ispravnog otvorenog teksta između četiri ponuđena. Ova prepreka u dešifriranju lako se može svladati u praksi dodavanjem unaprijed određene zalihosti originalnom otvorenom tekstu prije šifriranja (npr. mogu se kopirati zadnja 64 bita poruke). Tada je velika vjerojatnost kako samo jedan od četiri kvadratna korijena zadanog šifrata c posjeduje ovu zalihost, te primatelj odabire njega kao otvoreni tekst. Ako pak niti jedan od ovih korjena ne posjeduje ovu zalihost primatelj bi trebao odbiti šifrat c jer je onda riječ o prevari.

b) Ako se koristi zalihost kao u dijelu a), Rabinov algoritam nije više podložan odabranim napadima. Ako protivnik odabere poruku m uz posjedovanje potrebne zalihosti i izračuna $c = m^2 \pmod{n}$ te ga pošalje primatelju na dešifriranje, primatelj će najvjerojatnije protivniku vratiti otvoreni tekst m (poruku m) budući da preostala tri kvadratna korjena od c vjerojatno ne sadrže potrebnu zalihost. Tako protivnik ostaje zakinut za nove informacije. Ako pak protivnik odabere poruku m koja nema traženu zalihost, tada vjerojatno niti korijeni od c ju ne sadrže. Tada dešifriranje neće biti moguće obaviti i protivnik

neće dobiti odgovor. Napomenimo kako dokaz ekvivalencije za razbijanje sheme uz pasivni napad ovdje više ne vrijedi. Suprotno tome, ako uzmemo pretpostavku da se dešifriranje sastoji od dva procesa, gdje je prvi pronalazak kvadratnih korijena od $c \pmod n$, drugi od odabira određenog kvadratnog korijena za otvoreni tekst, tada je dokaz ekvivalencije valjan. Možemo zaključiti kako je Rabinovo šifriranje s javnim ključem, uz upotrebu zalihosti od velike koristi u praktičnoj primjeni.

Navedimo algoritam koji smo koristili pri određivanju kvadratnih korijena:

Algoritam 6.6 ***PRONALAZAK KVADRATNIH KORIJENA*** $c \pmod n$ ***KADA SU POZNATI FAKTORI*** p ***i*** q
Preduvijeti za rad algoritma su $n = pq$ *i* $p \equiv q \equiv 3 \pmod 4$.

1. *Uz pomoć proširenog Euklidovog algoritma za pronalazak najvećeg zajedničkog djelitelja izračunati brojeve* a *i* b *koji zadovoljavaju* $ap + bq = 1$. *Dovoljno je samo jednom izračunati* a *i* b *u postupku generiranja ključa.*

2. *Izračunati* $r = c^{(p+1)/4} \pmod p$.

3. *Izračunati* $s = c^{(q+1)/4} \pmod q$.

4. *Izračunati* $x = (aps + bqr) \pmod n$.

5. *Izračunati* $y = (aps - bqr) \pmod n$.

6. *Kvadratni korijeni od* $c \pmod n$ *su* x, $-x$, y *i* $-y$.

6.2.1 Sigurnost šifriranja Rabinovim kriptosustavom

Sigurnost uz koju se odvija šifriranje kada se koristi Rabinov algoritam možemo promatrati iz tri aspekta:

i) Pasivni protivnik dobiva zadatak odrediti otvoreni tekst m iz odgovarajućeg šifrata c. Zadatak se zapravo svodi na problem prolaska kvadratnih korijena. Pronalazak kvadratnih korijena modulo n i faktorizacija od n su računski ekvivalentne. Dakle, ako pretpostavimo kako je faktorizacija od n računski teško izvediva, Rabinov algoritam za šifriranje je dokazivo siguran protiv pasivnog protivnika.

ii) Kao što smo naveli u i), Rabinovo šifriranje je dokazivo sigurno protiv pasivnog protivnika, ali podliježe odabranom napadu. Takav napada se odvija na sljedeći način:
Protivnik odabire nasumični cijeli broj $m \in \mathbb{Z}_n^\times$ (gdje je $\mathbb{Z}_n^\times$ multiplikativna grupa brojeva modulo n) i izračunava $c = m^2 \pmod{n}$. Nakon toga, on prezentira c stroju za dešifriranje koji pripada subjektu A, koji tada dešifrira c i vraća neki otvoreni tekst y. Kako subjekt A ne zna koja je poruka m, ona se odabire slučajnim odabirom, te otvoreni tekst y nije nužno isti kao m. S vjerojatnošću od $1/2$, $y \not\equiv \pm m \pmod{n}$, gdje je $(m - y, n)$ jedan od prostih faktora od n. Ako je $y \equiv \pm m \pmod{n}$, tada se napad ponavlja s novim m.

iii) Šifriranje Rabinovim sustavom je osjetljivo na napade slične onima koje smo spomenuli u dijelu o sigurnosti RSA sustava (mali vršitelj šifriranja e, forward napadi). Ovdje postupamo kao i u slučaju šifriranja s RSA sustavom, ako su u pitanju napadi s malim vršiteljem šifriranja e ili forward napadi, oni se mogu zaobići upotrebom otovrenog teksta.

6.2.2 Primjena

Primjer 6.2 *Šifriranje uz pomoć Rabinovog sustava sa prividno malim parametrima.*

Generiranje ključa.
Subjekt A odabire proste brojeve $p = 127$, $q = 131$ *i izračunava* $n = pq = 16637$. *Javni ključ od A je* $n = 16637$, *a privatni ključ je* $(p = 127, q = 131)$.

Šifriranje.
Pretpostavimo kako se zadnjih 6 bitova originalne poruke trebaju kopirati prije šifriranja. Kako bi šifrirao 10*-bitnu poruku* $\overline{m} = 11111010$, *B kopira zadnji* 6 *bitova poruke* $\overline{m}$ *kako bi dobili* 16*-bitnu poruku* $m = 11111010111010$, *što je u decimalnom zapisu* $m = 16058$. *Subjekt B tada izračunava*

$$c = m^2 \pmod{n} = 16058^2 \pmod{16637} = 2501$$

i šalje ju subjektu A.

Dešifriranje.
Kako bi dešifrirao c, A koristi algoritam za pronalazak kvadratnih korjena modulo n kada su zadani prosti faktori p i q, izračunava četiri kvadratna korjena od c (mod n)*:*

$$m_1 = 579,\ m_2 = 16058,\ m_3 = 4247,\ m_4 = 12390,$$

što u binarnom zapisu daje:

$$m_1 = 1001000011,\ m_2 = 11111010111010,$$

$$m_3 = 1000010010111,\ m_4 = 11000001100110.$$

Budući m_2 posjeduje zalihost koja nam je potrebna, A dešifrira c kao m_2 i dobiva originalnu poruku $\overline{m} = 11111010$.

6.3 ElGamalov kriptosustav

ElGamalov kriptosustav se može promatrati i kroz Diffie-Hellmanov protokol za razmjenu ključeva. Sigurnost ovog sustava se temelji na težini problema diskretnih logaritama i Diffie-Hellmanovom problemu.
Diffie-Hellmannov problem usko je vezan uz problem diskretnog logaritma. Definirajmo diskretni logaritam:
Neka je G konačna ciklička grupa reda n. Neka je α generator grupe G i β proizvoljni element iz G. Diskretni logaritam od β po bazi α ($\log_\alpha \beta$) je jedinstveni cijeli broj x, $0 \leq x \leq n-1$, takav da je $\beta = \alpha^x$.
Diffie-Hellmanov problem se sastoji u tome da za dani prost broj p i generator α cikličke grupe $\mathbb{Z}_p^\times$, te elemente $\alpha^a \pmod p$ i $\alpha^b \pmod p$ treba odrediti $\alpha^{ab} \pmod p$. $\mathbb{Z}_p$ je kao skup jednaka $\{0, 1, \ldots, p-1\}$, dok je grupovna operacija množenje modulo p. Grupa je ciklička ako je generirana jednim elementom, a generator ove grupe je svaki n iz $\{1, 2, \ldots, p-1\}$, jer je p prost pa je takav n relativno prost s p. Pokažimo kako izgleda ElGamalov i generalizirani ElGamalov kriptosustav.

6.3.1 Osnovni ElGamalov kriptosustav

Algoritam 6.7 ***GENERIRANJE KLJUČA***
SAŽETAK: svaki subjekt kreira javni i njemu odgovarajući tajni ključ. Svaki subjekt treba:

1. *Generirati velik prost broj* p *i generator* α *multiplikativne grupe* $\mathbb{Z}_p^\times$ *cijelih brojeva modulo* p *uz korištenje Algoritma 6.8.*

2. *Odabrati cijeli broj* a, $1 \leq a \leq p-2$ *i izračunati* $\alpha^a \pmod p$.

3. *Javni ključ od* A *je* (p, α, α^a), *a privatni ključ je* a.

Algoritam 6.8 ***ODABIR*** k***-bitnog PROSTOG BROJA*** p ***I GENERATORA*** α

INPUT: Duljina bita k *i sigurnosni parametar* t.
OUTPUT: k*-bitni prost broj* p *takav da* $p-1$ *ima prost faktor koji je* $\geq t$ *i generator* α *od* $\mathbb{Z}_p^\times$.

Ponavljati:

1.1 *Odabrati* k*-bitni prost broj* p *(korištenjem Algoritma 4.2.)*

1.2 *Faktorizirati* $p-1$.

Sve dok $p-1$ *sadrži prost faktor* $\geq t$

2. *Pronaći generator* α *uz korištenje Algoritma 6.9 za određivanje generatora cikličke grupe* G $(\mathbb{Z}_p^\times, n = p-1)$.

3. *Vrati* (p, α).

Algoritam 6.9 ***ODREĐIVANJE GENERATORA CIKLIČKE GRUPE***

INPUT: Ciklička grupa G *reda* n *i rastav na proste faktore od* n, $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$.
OUTPUT: Generator α *grupe* G.

1. *Izabrati nasumično element* α *iz grupe* G.

2. *Za* i *od* 1 *do* k *raditi:*

2.1 *Izračunati* $b \leftarrow \alpha^{n/p_i}$.

2.2 *Ako je* $b = 1$, *prijeći na korak* 1.

3. Vrati (α).

Nakon generiranja ključa možemo započeti s kriptostustavom.

Algoritam 6.10 ***ELGAMALOV KRIPTOSUSTAV***
SAŽETAK: B šifrira poruku m *za* A*, te* A *dešifrira tu poruku.*

1. Šifriranje. B neka radi:

a) Dohvati javni ključ od A (p, α, α^a).

b) Prikaži poruku kao cijeli broj m *iz intervala* $\{0, 1, ..., p-1\}$.

c) Izaberi nasumični cijeli broj k, $1 \leq k \leq p-2$.

d) Izračunaj $\gamma = \alpha^k \pmod{p}$, *te* $\delta = m \cdot (\alpha^a)^k \pmod{p}$.

e) Pošalji A *šifrat* $c = (\gamma, \delta)$.

2. Dešifriranje. Kako bi iz šifrata c *dobio otvoreni tekst* m, A *neka radi slijedeće:*

a) Izračunaj $\gamma^{p-1-a} \pmod{p}$ *(primjetimo da je* $\gamma^{p-1-a} \equiv \gamma^{-a} \equiv \alpha^{-ak} \pmod{p}$*) koristeći privatni ključ od* A.

b) Izračunaj $(\gamma^{-a}) \cdot \delta \pmod{p}$ *kako bi dobio poruku* m.

Navedimo jedan primjer iz prakse.

6.3.2 Primjena

Primjer 6.3 *ElGamalov kriptosustav s prividno malim parametrima.*

Generiranje ključa.
Subjekt A *izabire prost broj* $p = 1777$ *i generator* $\alpha = 6$ *od* $\mathbb{Z}_{1777}^{\times}$. A *izabire privatni ključ* $a = 1009$ *i izračunava:*

$$\alpha^a \pmod{p} = 6^{1009} \pmod{1777} = 1729.$$

Javni ključ od A *je* $(p = 1777, \alpha = 6, \alpha^a = 1729)$.

Šifriranje.
Kako bi šifrirao poruku $m = 1483$*, B izabire nasumični cijeli broj* $k = 701$ *i izračunava:*

$$\gamma = 6^{701} \pmod{1777} = 1664$$

i

$$\delta = 1483(1729)^{701} \pmod{1777} = 1241.$$

B šalje $\gamma = 1664$ *i* $\delta = 1241$ *subjektu A.*

Dešifriranje.
Kako bi dešifrirao šifrat, A izračunava

$$\gamma^{p-1-a} = 1664^{767} \pmod{1777} = 1572,$$

i dobiva m

$$m = 15721241 \pmod{1777} = 1483.$$

Kod ElGamalovog kriptosustava svaki subjekt može odabrati iste parametre p i generator α, gdje onda ti parametri postaju dio javnog ključa. Rezultat ovoga mogu biti ključevi malih veličina. Prednost je svakako kada imamo fiksne veličine (α) što može ubrzati potenciranje uz neke unaprijed izračunate (poznate) rezultate. Nedostatak ovakvog sustava može biti pojava velikog parametra p.

6.3.3 Učinkovitost ElGamalovog kriptosusutava

1. Proces šifriranja zahtijeva dva modularna potenciranja, to su $\alpha^k \pmod{p}$ i $(\alpha^a)^k \pmod{p}$. Potenciranje se može ubrzati odabirom nasumičnog eksponenta k koji imaju neki dodatak, npr. malu Hammingovu težinu, tj. broj jedinica koje se nalaze u binarnom prikazu tog broja. Treba povesti računa da mogući broj eksponenata bude dovoljno velik kako bi izbjegli račun s diskretnim logaritmima.

2. Nedostatak ElGamalovog kriptosustava je povećanje poruke, tj. uvećavanje za faktor 2 (šifrat je dvostruko dulji od otvorenog teksta).

6.3.4 Nasumično šifriranje

ElGamalov kriptosustav je jedan od mnogih u kojima se koristi nasumični odabir pri šifriranju. Sustavi poput RSA kriptosustava mogu također koristiti nasumičost kako bi se izbjegli određeni napadi. Temeljna ideja nasumičnog šifriranja je povećavanje sigurnosti šifriranja kroz neku od slijedećih metoda:

1. povećanje efektivne veličine prostora u kojem se nalazi otvoreni tekst;
2. isključenjem ili smanjenjem učinkovitosti odabranih napada korištenjem preslikavanja jednog elementa u više različitih elemenata prilikom šifriranja;
3. isključenjem ili smanjenjem učinkovitosti statističkih napada ostavljajući a priori vjerojatnosnu distribuciju ulaznih podataka.

6.4 Sigurnost ElGamalov kriptosustava

i) Problem prilikom razbijanja ElGamalovog kriptosustava, gdje je potrebno saznati poruku m ako su zadani parametri p, α, α^a, γ i δ je istovjetan rješavanju Diffie-Hellmanovog problema. Naime, ElGamalov sustav možemo promatrati kao Diffie-Hellmanovu jednostavnu razmjenu ključeva kako bi se odredio ključ α^{ak}, a zatim se šifrira množenjem s tim istim ključem. Kaže se kako se sigurnost ElGamalovog kriptosustava temelji na problemu diskretnog logaritma u $\mathbb{Z}_p$, iako takva ekvivalencija nije dokazana.

ii) Ključno je da se prilikom šifriranja različitih poruka, koriste različiti parametri k. Ako bismo pak isti k koristili za šifriranje dvije različite poruke m_1 i m_2, dobivamo šifrate (γ_1, δ_1) i (γ_2, δ_2). Tada je $\frac{\delta_1}{\delta_2} = \frac{m_1}{m_2}$, te se m_2 može lako izračunati ako poznajemo m_1.

6.5 Generalizirani ElGamalov kriptosustav

ElGamalov kriptosustav je najčešće povezan sa multiplikativnom grupom $\mathbb{Z}_p$, ali se lako može generalizirati tako da radi sa bilo kojom konačnom cikličkom grupom G. Sigurnost ovog sustava se također temelji na problemu diskretnog

logaritma u grupi G. Grupa G se odabire tako da su zadovoljena sljedeća dva uvjeta:

1. operacije u grupi se trebaju lako primjenjivati kako bi se povećala učinkovitost;

2. problem diskretnog logaritma u grupi bi trebao biti računski neizvediv kako bi se povećala sigurnost.

Neke od grupa koje zadovoljavaju gornja dva uvijeta su:

1. multiplikativna grupa $\mathbb{Z}_p$ cijelih brojeva modulo p (p je prost broj);

2. multiplikativna grupa $\mathbb{F}_{2^m}^{\times}$ (koju čine svi elementi različiti od nule) konačnog polja $\mathbb{F}_{2^m}$;

3. multiplikativna grupa $\mathbb{F}_q^{times}$, konačnog polja $\mathbb{F}_q$, gdje je $q = p^m$, p je prost;

4. grupa invertibilnih elemenata multiplikativne grupe $\mathbb{Z}_n$, gdje je n složen cijeli broj (to su zapravo elementi koji su relativno prosti sa n);

Algoritam 6.11 ***GENERIRANJE KLJUČA ZA GENERALIZIRANI ELGAMALOV KRIPTOSUSTAV***

SAŽETAK: Svaki subjekt kreira javni i odgovarajući privatni ključ. Subjekt A neka radi:

1. *Izabrati odgovarajuću cikličku grupu G reda n, sa generatorom α (pretpostavimo da u grupi G koristimo multiplikativnu notaciju).*

2. *Izabrati nasumični cijeli broj a, $1 \leq a \leq n-1$, te izračunati α^a.*

3. *Javni ključ od A je (α, α^a) zajedno s opisom kako se množe elementi grupe G, a privatni ključ je a.*

Algoritam 6.12 ***GENERALIZIRANI ELGAMALOV KRIPTOSUSTAV***

B šifrira poruku m koju je dobio od A, te A dešifrira poruku od B.

1. *Šifriranje.* B *neka radi:*

   a) *Dohvati javni ključ od* A, (α, α^a).

   b) *Zapiši poruku kao element* m *grupe* G.

   c) *Izaberi nasumični cijeli broj* k, $1 \leq k \leq n-1$.

   d) *Izračunaj* $\gamma = \alpha^k$ *i* $\delta = m \cdot (\alpha^a)^k$.

   e) *Pošalji šifrat* $c = (\gamma, \delta)$ *subjektu* A.

2. *Dešifriranje. Kako bi iz šifrata* c *dobio otvoreni tekst* m, A *neka radi:*

   a) *Uz pomoć privatnog ključa* a *izračunaj* γ^a *a zatim* γ^{-a}.

   b) *Izračunaj* m *kao* $(\gamma^{-a}) \cdot \delta$.

U ovom kriptosustavu svi subjekti mogu odabrati istu cikličku grupu G i njezin generator α, te se tada α i opis same multiplikativne grupe G ne objavljuju kao dio javnog ključa.
Prije nego prikažemo primjenu ovog generaliziranog kriptosustava, navedimo rezultat koji uvelike utječe na sami proces.

Konačno polje $\mathbb{F}_{2^4}$ reda 16
Može se provjeriti kako je polinom $f(x) = x^4 + x + 1$ ireducibilan nad poljem $\mathbb{F}_2$ (polinom $f(x) \in \mathbb{Z}_p$, stupnja $m \geq 1$ je ireducibilan nad $\mathbb{Z}_p$ ako se ne može prikazati kao produkt dva polinoma iz $\mathbb{Z}_p$ koji su stupnja manjeg od m). Konačno polje $\mathbb{F}_{2^4}$ se može prikazati kao skup polinoma nad $\mathbb{F}_2$ stupnja manjeg od 4, tj.

$$\mathbb{F}_{2^4} = \{a_3x^3 + a_2x^2 + a_1x + a_0 : a_i \in \{0, 1\}\}.$$

Iz praktičnih razloga polinom $a_3x^3 + a_2x^2 + a_1x + a_0$ je zapisan kao vektor duljine 4 $(a_3a_2a_1a_0)$ i

$$\mathbb{F}_{2^4} = \{(a_3a_2a_1a_0) : a_i \in \{0, 1\}\}.$$

6.5.1 Primjena

Primjer 6.4 *Generalizirani ElGamalov kriptosustav sa multiplikativnom grupom F_{2^m} sa prividno malim parametrima.*

Generiranje ključa.
Subjekt A izabire grupu G koja je multiplikativna grupa konačnog polja $\mathbb{F}_{2^4}$. Grupa G je reda $n = 15$ i generator $\alpha = (0010)$.
A odabire privatni ključ $a = 7$ i izračunava $\alpha^a = \alpha^7 = (1011)$. Javni ključ subjekta A je $\alpha^a = (1011)$.

Šifriranje.
Kako bi šifrirao poruku $m = (1100)$, B odabire nasumični cijeli broj $k = 11$ i izračunava $\gamma = \alpha^{11} = (1110)$, $(\alpha^a)^{11} = (0100)$ i $\delta = m \cdot (\alpha^a)^{11} = (0101)$. B šalje $\gamma = (1110)$ i $\delta = (0101)$ subjektu A.

Dešifriranje.
Kako bi dešifrirao poruku, subjekt A izračunava $\gamma^a = (0100)$, $(\gamma^a)^{-1}$ i dobiva poruku m iz računa $m = (\gamma^{-a}) \cdot \delta = (1100)$.

7 Primjena kriptosustava s javnim ključem

Praktična primjena šifriranja danas se koristi u mnogim djelatnostima gdje je bitno očuvati tajnost i zaštitu podataka (u bazama podataka, internet bankarstvu, razmjeni elektroničke pošte). Razmjena elektoriničke pošte (emaila) je danas jedan od najpopularnijih oblika komunikacije kako na internetu tako i općenito. Email može biti privatnog ili poslovnog sadržaja, te je taj sadržaj obično nezaštićen te bilo tko sa pristupom poslužitelju preko kojeg se poruke šalju je u mogućnosti pročitati njegov sadržaj. Kako bi se izbjegla zloupotreba, potrebno je elektroničku poštu kriptirati, tj. šifrirati sadržaj te ga u obliku šifrata prenijeti od pošiljatelja do primatelja. Imamo više metoda za šifriranje poruka elektroničke pošte:

PGP (Pretty Good Privacy) je standard koji pomoću metode sažimanja, kompresije i šifriranja omogućuje zaštitu sadržaja pošte.

S/MIME (Secure/Multipurpose Internet Mail Extension) je standard koji se koristi simetričnim i asimetričnim kriptosustavima za šifriranje pošte.

STARTTLS je protokol (nadogradnja SMTP protokola koji se koristi za razmijenu elektroničke pošte) koji se koristi kako bi se onemogućilo prisluškivanje na poslužiteljima.

Postupak šifriranja se odvija na standardan način:

Primatelj kreira svoj javni i tajni ključ;

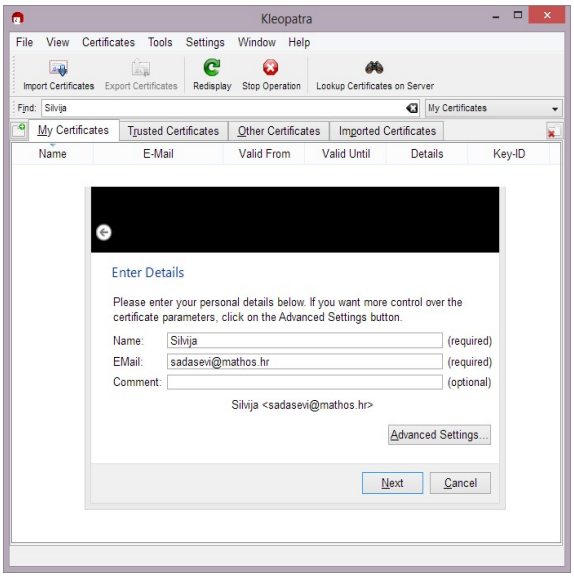
Objavljuje svoj javni ključ;

Pošiljatelj šifrira poruku javnim ključem primatelja;

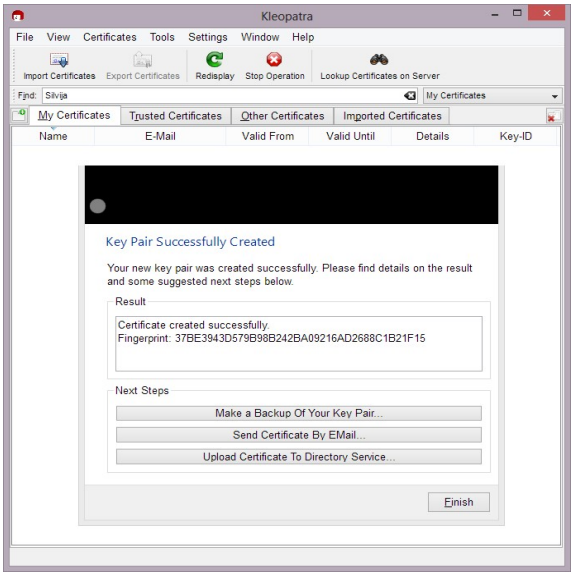
Šalje ju primatelju;

Primatelj dešifrira poruku svojim tajnim ključem.

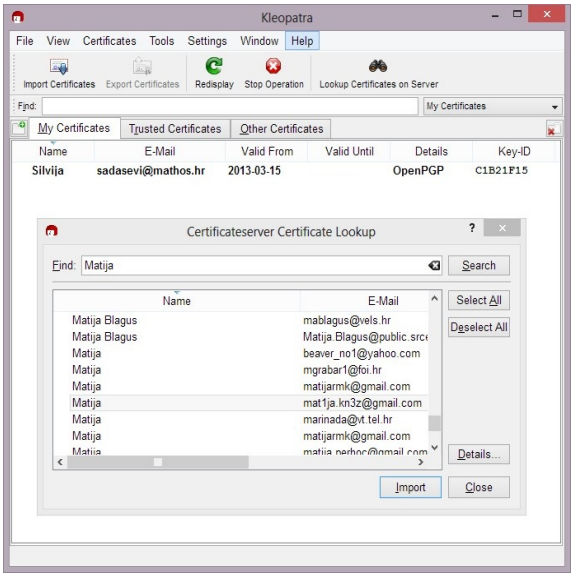
Napomenimo kako kada pošiljatelj naslovi poštu primatelju pomoću javnog ključa, samo primatelj može pročitati tu poruku. Za dodatnu sigurnost može se dodati potpisivanje pošte, te se na taj način dobiva neporecivost te integritet poruke.
Postupak šifriranja i potpisivanja započinje generiranjem ključeva, koji mogu biti šifrirani pomoću raznih kriptosustava, među kojima su i ELGamalov i RSA kriptosustav. Pogledajmo kako to izgleda u praksi koristeći GPG4win (alat za šifriranje elektroničke pošte).



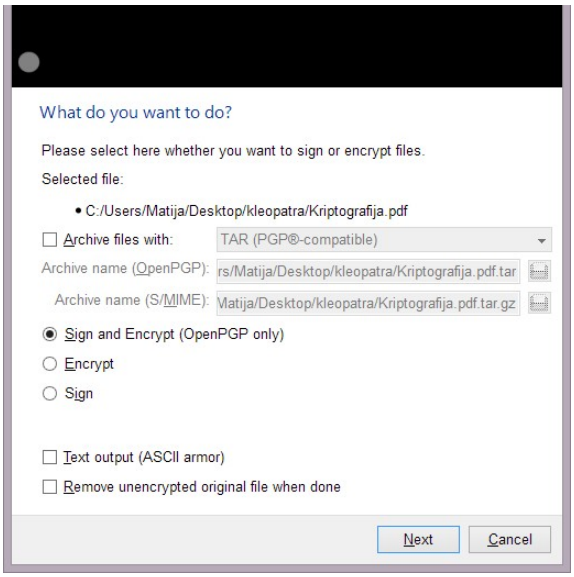
Slika 4. Najprije upisujemo svoju email adresu (u skupu alata GPG4win koristi se program Kleopatra za generiranje novog certifikata, ključeva, te za šifriranje i dešifriranje).



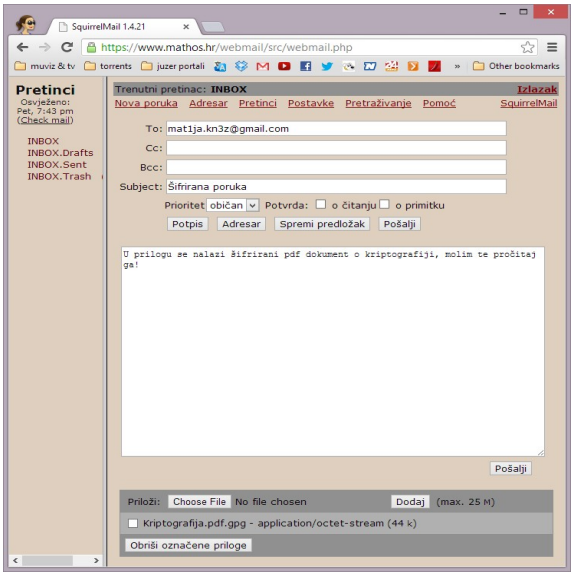
Slika 5. Zatim se obavlja generiranje javnog i tajnog ključa.



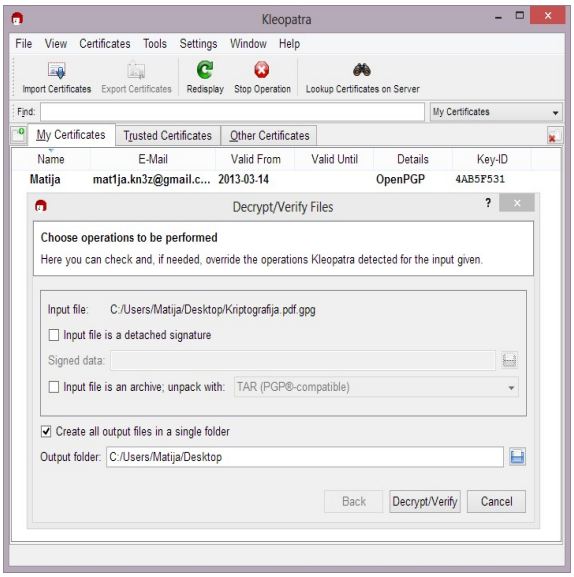
Slika 6. Javni ključ primatelja se može dobaviti sa nekog od servera, ako ga je primatelj objavio.



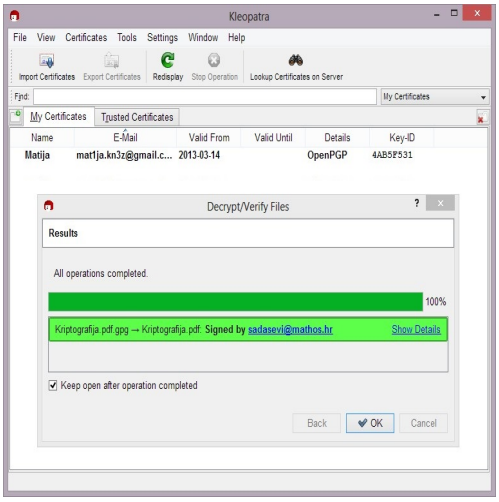
Slika 7. Nakon toga je potrebno šifrirati poruku koju želimo poslati. To obavlja programski alat Kleopatra, a koristi se najčešće jednim od kriptosustava s javnim ključem, u ovom slučaju to je RSA kriptosustav).



Slika 8. Poruku ili dokument koji smo šifrirali šaljemo u običnom privitku email poruke.



Slika 9.



Slika 10. Primatelj zaprima email poruku, te njezin privitak dešifrira istim programskim alatom uz poznavanje javnog ključa pošiljatelja (Slika 9, Slika 10).

Literatura

[1] Andrej Dujella, M. Maretić, *Kriptografija*, Element, Zagreb, 2007.

[2] Andrej Dujella, *Uvod u teoriju brojeva*, skripta, PMF - Matematički odjel, Sveučilište u Zagrebu, Zagreb, 2009.

[3] Alfred J. Menezes, Paul C. van Oorschott, Scott A. Vanstone, *Handbook of applied cryptography*, CRC Press, 2011.

[4] D. Žubrinić, *Diskretna matematika*, Element, Zagreb, 2002.

[5] http://www.cis.hr/sigurosni-alati/kriptiranje-elektronicke-poste.html

Sažetak

Kriptografija kao znanstvena disciplina ima zadatak osigurati razmijenu informacija tako da informacije mogu razumijeti samo osobe kojima su one namijenjene. Ljudi su od najstarijih vremena imali potrebu za sigurnom razmijenom informacija. Danas se za to koriste različiti kriptosustavi.
Sustavi koji se koriste asimetričnom kriptografijom, tj. koriste kriptografiju javnog ključa su sporiji od simetričnih sustava. Stoga se oni pogodniji za šifriranje manjih jedinica podataka.
Kriptosustavi kojima smo se bavili u radu svaki posjeduju svoje specifičnosti zbog kojh su sigurni za prijenos podataka.
RSA kriptosustav sigurnost temelji na teškoći faktorizacije velikih prirodnih brojeva.
Rabinov kriptosustav svoju sigurnost temelji na teškoći računanja kvadratnih korijena po fiksnom složenom modulu.
ElGamalov kriptosustav se oslanja na problem računanja s diskretnim logaritmom.
Konkretna primjena ovih sustava se može pronaći u šifiranju raznih baza podataka, razmijeni elektroničke poste, internet bankarstvu i slično.

Public key cryptography in practice

Cryptography as a science discipline has the task of ensuring the exchange of information so that the information can be understood only by persons for whom they are intended.
People of ancient times had the need for a secure information exchange. Today to ensure that, we use several cryptosystems.
The systems that use the asymmetric cryptography, that is using public key cryptography are slower than symmetric systems. Therefore, they are more suitable for small units of data encryption.
Cryptosystem that were discussed in the paper each have their own specificity due to are safe to transfer data.
RSA cryptosystem its security based on the difficulty of factorization of large numbers.
Rabin Cryptosystem its security based on the difficulty of calculating the square root of the complex at a fixed module.
ElGamalov cryptosystem relies on the problem of computing discrete logarithms.
The specific application of these systems can be found in a variety of databases, exchange of emails, internet banking, etc.

Životopis

Rođena sam 29.09.1985. u Sremskoj Mitrovici. Godine 1992. upisala sam osnovnu školu, koju sam završila 2000. godine sa priznanjem Školske knjige za najboljeg učenika Osnovne škole A. Starčevića u Viljevu. Od petog do osmog razreda sudjelovala sam na više natjecanja, od kojih su zapaženi rezultati bili iz područja povijesti, zemljopisa i biologije. Godine 2000. upisala sam Isusovačku klasičnu gimnaziju u Osijeku, koju završavam 2004. Iste godine upisujem preddiplomski studij matematike na Odjelu za matematiku u Osijeku. 2007. mi je odobrena promjena smjera na sveučilišni nastavnični studij matematike i informatike. Tijekom studiranja radila sam kao nastavnik matematike u Oš Tenja na zamjeni.