Problem diskretnog logaritma i Diffie-Hellmanov protokol

Josipović, Iwan

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni diplomski studij matematike i računarstva

Iwan Josipović

Problem diskretnog logaritma i Diffie-Hellmanov protokol

Diplomski rad

Osijek, 2020

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni diplomski studij matematike i računarstva

Iwan Josipović

Problem diskretnog logaritma i Diffie-Hellmanov protokol

Diplomski rad

Mentor: izv.prof.dr.sc. Ivan Matić

Osijek, 2020

Sadržaj

1 Uvod **1**

2 Početci kriptosustava s javnim ključem **3**
2.1 Otkriće kriptosustava s javnim ključem 3
2.2 Kripotsustavi s javnim ključem kao "tabu" tema 3
2.3 Osnovne definicije 3

3 Probem diskretnog logaritma **6**
3.1 Diffie-Hellmanov protokol za razmjenu ključeva 9
3.2 Primjena Diffie-Hellmanovog protokola 11
3.3 Napad s čovjekom u sredini na Diffie-Hellmanov protokol 12
3.4 ElGamalov kriptosustav s javnim ključem 13
3.5 Koliko je težak problem diskretnog logaritma? 18
3.6 Algoritam sudara za problem diskretnog logaritma 19

4 Zaključak **22**

Literatura **23**

Sažetak **24**

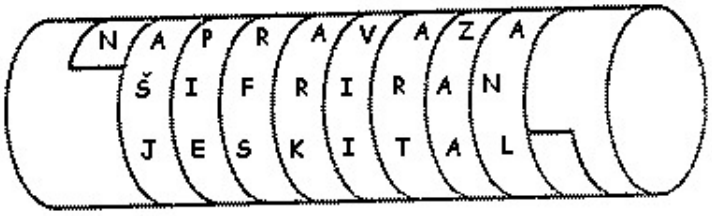
Summary **25**

Životopis **26**

1 Uvod

Kriptografija je znanstvena disciplina koja se bavi proučavanjem metoda za slanje poruka u takvom obliku da ih samo onaj kome su namijenjene može pročitati. Sama riječ kriptografija je grčkog podrijetla i mogla bi se doslovno prevesti kao tajnopis.

Neki elementi kriptografije bili su prisutni već kod starih Grka. Naime, Spartanci su u 5. stoljeću prije Krista upotrebljavali napravu za šifriranje zvanu skital sa slike 1. To je bio drveni štap oko kojeg se namotavala vrpca od pergamenta, pa se na nju okomito pisala poruka. Nakon upisivanja poruke, vrpca bi se odmotala, a na njoj bi ostali izmiješani znakovi koje je mogao pročitati samo onaj tko je imao štap jednake debljine.



Slika 1: Ilustracija skitala

Osnovni zadatak kriptografije je omogućiti dvjema osobama (zvat ćemo ih pošiljalac i primalac - u kriptografskoj literaturi su za njih rezervirana imena Alice i Bob, ali mi ćemo ih zvati Ana i Ivan) komuniciranje preko nesigurnog komunikacijskog kanala (telefonska linija, računalna mreža,...) na način da treća osoba (njihov protivnik - u literaturi se najčešće zove Eva ili Oskar, u našem slučaju Lea), koja može nadzirati komunikacijski kanal, ne može razumjeti njihove poruke. Poruku koju pošiljalac želi poslati primaocu zvat ćemo otvoreni tekst (engl. plaintext). To može biti tekst na njihovom materinjem jeziku, numerički podatci ili bilo što drugo. Pošiljalac transformira otvoreni tekst koristeći unaprijed dogovoreni ključ. Taj postupak se naziva šifriranje, a dobiveni rezultat šifrat (engl. ciphertext) ili kriptogram. Nakon toga pošiljalac pošalje šifrat preko nekog komunikacijskog kanala. Protivnik prisluškujući može doznati sadržaj šifrata, ali ne može odrediti otvoreni tekst. Za razliku od njega, primalac koji zna ključ kojim je šifrirana poruka može dešifrirati šifrat i odrediti otvoreni tekst.

Za razliku od dešifriranja, kriptoanaliza ili dekriptiranje je znanstvena disciplina koja se bavi proučavanjem postupaka za čitanje skrivenih poruka bez poznavanja ključa. S druge strane, kriptologija je grana znanosti koja obuhvaća kriptografiju i kriptoanalizu. Kako bismo u potpunosti razumjeli iduće poglavlje, uvesti ćemo sljedeće pojmove:

Definicija 1.1. Kriptosustav je uređena petorka $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ za koju vrijedi:

1. $\mathcal{P}$ je konačan skup svih mogućih osnovnih elemenata otvorenog teksta.
2. $\mathcal{C}$ je konačan skup svih mogućih osnovnih elemenata šifrata.
3. $\mathcal{K}$ je prostor ključeva, tj. konačan skup svih mogućih ključeva.
4. Za svaki $K \in \mathcal{K}$ postoji funkcija šifriranja $e_K \in \mathcal{E}$ i odgovarajuća funkcija dešifriranja $d_K \in \mathcal{D}$. Pritom su $E_K : \mathcal{P} \longrightarrow \mathcal{C}$ i $d_K : \mathcal{C} \longrightarrow \mathcal{P}$ funkcije sa svojstvom da je $d_K(e_K(x)) = x$ za svaki otvoreni tekst $x \in \mathcal{P}$.

Najvažnije svojstvo u definiciji 1.1 je $d_K(e_K(x)) = x$. Iz njega slijedi da funkcije e_K moraju biti injekcije. Zaista, ako bi bilo

$$e_K(x_1) = e_K(x_2) = y,$$

za dva različita otvorena teksta x_1 i x_2, onda primalac ne bi mogao odrediti trebali li y dešifrirati u x_1 ili x_2, tj. $d_K(y)$ ne bi bilo definirano. U skladu s tim imamo da, ako je $\mathcal{P} = \mathcal{C}$, onda su funkcije e_K permutacije.

Kripotsustavi se dijele prema:

- **Tipu operacije za šifriranje** (supstitucijske šifre, transpozicijske šifre)
- **Načinu obrađivanja otvorenog teksta** (blokovne šifre, protočne šifre)
- **Tajnost i javnost ključa** (simetrični kriptosustavi, kriptosustavi s javnim ključem)

U ovom radu baviti ćemo se kriptosustavima s javnim ključem, počevši od povijesti sve do detaljne razrade načela na kojima se temelje.

2 Početci kriptosustava s javnim ključem

2.1 Otkriće kriptosustava s javnim ključem

1976. godine, Whitfield Diffie i Martin Hellman objavili su rad pod nazivom "New Directions in Cryptography." U tom radu opisali su koncept enkripcije s javnim ključem i došli do revolucionarnih otkrića u tom području. Nešto ranije, nezavisno o njima, Ralph Merkle je u sklopu projekta na Berkleyu smislio postupak konstrukcije javnih ključeva, ali tada je to otkriće još uvijek bilo neshvaćeno, te je objavljeno tek 1982. u radu pod nazivom "Secure communication over insecure channels."
Ipak, smatra se da je koncept enkripcije s javnim ključem prvi otkrio James Ellis dok je radio za Sjedište za Komunikaciju Britanske Vlade (GCHQ). Ellisova otkrića su 1969. godine označena kao državna tajna, te nisu objavljena do 1997. godine, nakon njegove smrti. Poznato je da su dva druga istraživača u GCHQ-u, Malcolm Williamson i Clifford Cocks, otkrili Diffie-Hellmanov algoritam razmjene ključeva i RSA sustav enkripcije s javnim ključem još prije nego što su ih Diffie, Hellman i suradnici javno objavili.

2.2 Kripotsustavi s javnim ključem kao "tabu" tema

Rad koji su Diffie i Hellman objavili bio je iznimno značajan. Postavio je temeljne definicije i ciljeve nove grane matematike i računarstva, grane čije je postojanje nezavisno o tada tek napravljenom digitalnom računalu.
Naravno, njihov rad izazvao je nemire. Prije nego što su objavili rad, istraživanja na temu kriptografije u SAD-u provodila je Nacionalna Sigurnosna Agencija (NSA), stoga su sve informacije u na tu temu bile tajne. Sve do sredine 90-tih godina 20. stoljeća vlada SAD-a tretirala je kriptografske algoritme kao municiju, a to je značilo da je zabranjeno i kažnjivo izvoženje istih u druge države. Naravno, vlasti su ubrzo shvatile da nema smisla braniti uporabu i govorenje o tim kriptografskim sustavima. Stoga su zabranili izvoz samo onih kriptografskih algoritama koji su strojno čitljivi, s ciljem da kompleksni kriptografski programi ne upadnu u ruke potencijalnih neprijatelja SAD-a. To je stvorilo dva nova problema. Prvo, postojanje optičkih skenera brisalo je granicu između "strojno čitljivog" i "ljudskog teksta." U znak protesta ljudi su napisali verziju RSA algoritma u tri linije koda programskoga jezika perl, te printali šalice i limenke s tim linijima koda. Tako su te šalice i konzerve postale "municija". Zanimljivost je da bi tetovaža sljedećeg koda značila da tijelo postaje municija sa zabranom izvoza.

```
1 !/bin/perl -sp0777i<X+d*lMLa^*lN%0]dsXx++lMlN/dsM0<j]dsj
2 $/=unpack('H*',$_);$_=`echo 16dio\U$k"SK$/SM$n\EsN0p[lN*1
3 lK[d2%Sa2/d0$^Ixp"|dc`;s/\W//g;$_=pack('H*',/((..)*)$/)
```

Primjer koda 1: RSA algoritam u 3 linije

Vrijeme je pokazalo da su ovakve mjere bile besmislene, jer u svijetu ima previše matematičara i računaraca, te bi u konačnici netko došao do sličnih otkrića bez obzira na zabrane. Razvoj kriptografije se preselio u druge regije svijeta gdje nije bilo zabranjeno razvijati i javno govoriti o kriptografskim sustavima, te danas gotovo svatko može kupiti kriptografski program koji omogućuje potpuno sigurnu razmjenu podataka.

2.3 Osnovne definicije

Prva i najbitnija stvar koju su Diffie i Hellman definirali je kriptosustav s javnim ključem (eng. PKC, Public Key Cryptosystem) te pripadne jednosmjerne funkcije sa stupi-

com.

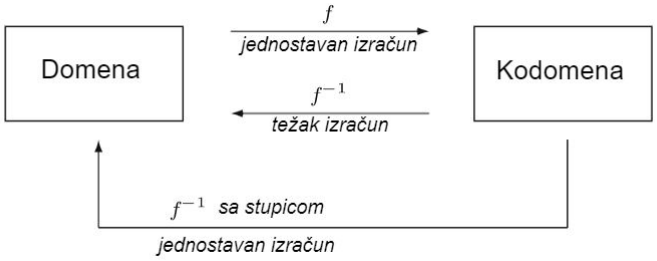
Definicija 2.1. Kriptosustav s javnim ključem sastoji se od dviju familija e_K i d_K funkcija za šifriranje i dešifriranje (gdje K prolazi skupom svih mogućih korisnika) sa svojstvom:

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.

Primjer. Ako pošiljatelj A želi poslati poruku x primaocu B, tada B najpije pošalje A svoj javni ključ e_B. Tada A šifrira svoju poruku pomoću e_B i pošalje primaocu B šifrat $y = e_B(x)$. Konačno, B dešifrira šifrat kopristeći svoj tajni ključ d_B i dobiva $d_B(y) = d_B(e_B(x)) = x$.

Jednosmjerne funkcije su invertibilne funkcije koje je jednostavno izračunati, ali je njihov inverz "teško" izračunati. Naravno, postavlja se pitanje što znači "teško" za izračunati. Izračun smatramo teškim ako svaki algoritam za računanje treba puno vremena za obavljanje zadatka.
Sigurni kriptosustavi s javnim ključem koriste jednosmjerne funkcije sa stupicom. Stupica je dodatna informacija pomoću koje možemo lako izračunati inverz jednosmjerne funkcije. Ta ideja prikazana je na slici 2. Treba napomenuti da nije lako doći od ideje jednosmjerne funkcije sa stupicom do konstrukcije takve funkcije.



Slika 2: Ilustracija jednosmjerne funkcije sa stupicom

Ključ u kriptosustavu s javnim ključem sastoji se od dva elementa, privatnog ključa k_{priv} i javnog ključa k_{pub}, gdje je k_{pub} najčešće dobiven primjenom algoritma za generiranje ključeva na k_{priv}. Za svaki par privatnog i javnog ključa (k_{priv}, k_{pub}) postoji enkripcijski algoritam $e_{k_{pub}}$ i pripadni dekripcijski algoritam $d_{k_{priv}}$. Enkripcijski algoritam $e_{k_{pub}}$ pripada ključu k_{pub}, javno je poznat te se lako računa. Slično dekripcijski algoritam $d_{k_{priv}}$ mora biti lako izračunljiv ukoliko posjedujemo informaciju k_{priv}, ali trebao bi biti gotovo nemoguće izračunati ukoliko posjedujemo samo informaciju k_{pub}.

Zanimljivo je da usprkos godinama istraživanja još uvijek nije poznato postoje li jednosmjerne funkcije. Zapravo, kada bi dokazali postojanje jednosmjernih funkcija istovremeno bi riješili slavni problem $\mathcal{P} = \mathcal{NP}$ iz teorije kompleksnosti.[1] Do sada su mnoge funkcije predložene kao kandidati za jednosmjerne funkcije, neke čak i koristimo u modernim kriptosustavima s javnim ključem. Moramo naglasiti da se sigurnost tih kriptosustava oslanja na pretpostavci da je pronaći inverz korištene funkcije (ili pronalazak privatnog ključa iz javnog) težak problem.

Diffie i Hellman dali su nekoliko svojih prijedloga za jednosmjerne funkcije kao što su problem naprtnjače i potenciranje $\bmod q$, ali nisu konstruirali niti jedan kriptosustav s javnim ključem pretežito jer nisu uspjeli pronaći prikladnu informaciju koja bi poslužila kao stupica. Ali, uspjeli su opisati metodu s javnim ključem koja omogućava sigurno komunikaciju koristeći nesiguran kanal. Njihova metoda, koju zovemo Diffie-Hellmanov protokol za razmjenu ključeva, oslanja se na činjenicu da je problem diskretnog logaritma težak za rješavanje. Navedenu metodu i problem obraditi ćemo u nastavku rada.

[1] Problem $\mathcal{P} = \mathcal{NP}$ jedan je od milenijskih problema, rješavanjem bilo kojeg od njih osvaja se nagrada od 1,000,000$.

3 Probem diskretnog logaritma

Problem diskretnog logaritma matematički je problem koji se pojavio u mnogim oblicima, uključujući $\mod p$ inačicu koju ćemo obraditi u ovom poglavlju i npr. varijante s eliptičnim krivuljama u koju nećemo ulaziti ali možete ju pronaći u literaturi [2]. Prva objavljena konstrukcija javnog ključa, od Diffie i Hellmana, temelji se na problemu diskretnog logaritma u konačnom polju $\mathbb{F}_p$, gdje je $\mathbb{F}_p$ polje s prostim brojem elemenata.

Definicija 3.1. Uvodimo i koristimo sljedeće oznake.

Prsten cijelih brojeva modulo m:

$$\mathbb{Z}/m\mathbb{Z} = \{0, 1, 2, \ldots, m-1\}$$

Grupu koja sadrži samo invertibiline elemente iz $\mathbb{Z}/m\mathbb{Z}$ označiti ćemo kao:

$$(\mathbb{Z}/m\mathbb{Z})^* = \{a \in \mathbb{Z}/m\mathbb{Z} : gcd(a, m) = 1\} = \{a \in \mathbb{Z}/m\mathbb{Z} : a \text{ ima inverz modulo } m\}.$$

Oznaku $\mathbb{F}_p$ koristiti ćemo za označavanje polja[2] $\mathbb{Z}/p\mathbb{Z}$ gdje je p prost broj. Analogno ćemo koristiti sljedeću notaciju:

$$\mathbb{F}_p^* = (\mathbb{Z}/p\mathbb{Z})^*.$$

Primjedba 1. Iako koristimo $\mathbb{Z}/p\mathbb{Z}$ i $\mathbb{F}_p$ kako bi označili istu stvar, jednakost elemenata u njima se izražava drugačije. Za $a, b \in \mathbb{F}_p$, jednakost od a i b označavamo s $a = b$, dok za $a, b \in \mathbb{Z}/p\mathbb{Z}$ jednakost elemenata a i b označavamo ekvivalencijom $\mod p$, npr. $a \equiv b \pmod{p}$.

Zbog primjedbe 1, označavat ćemo to polje s $\mathbb{F}_p$ kad koristimo jednakost elemenata, te sa $\mathbb{Z}/p\mathbb{Z}$ kad koristimo kongruencije.

Teorem 1. (Teorem o primitivnom korijenu) Neka je p prost broj. Tada postoji element $g \in \mathbb{F}_p^*$ čije potencije generiraju sve elemente u $\mathbb{F}_p^*$, na primjer:

$$\mathbb{F}_p^* = \{1, g, g^2, g^3, \ldots, g^{p-2}\}.$$

Elemente s ovim svojstvom nazivamo primitivnim korijenima od $\mathbb{F}_p$ ili generatorima od $\mathbb{F}_p^*$. To su elementi iz $\mathbb{F}_p^*$ reda $p-1$.

Teorem 2. Mali Fermatov Teorem. Neka je p prost broj. Ako $p \nmid a$ onda je

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

Neka je p velik prost broj. Teorem 1 kaže da postoji primitivan element g. To znači da je svaki nenul element iz $\mathbb{F}_p$ jednak nekoj potenciji od g. Posebno, $g^{p-1} = 1$ po teoremu 2. Dodatno $p-1$ najmanja je takva potencija za koju to vrijedi. Ekvivalentno, lista:

$$1, g, g^2, g^3, \ldots, g^{p-2} \in \mathbb{F}_p^*$$

sadrži sve elemente iz $\mathbb{F}_p^*$ u nekom poretku.

[2] $\mathbb{Z}/m\mathbb{Z}$ je polje ako i samo ako je m prost broj.

Definicija 3.2. Neka je g primitivni korijen od $\mathbb{F}_p$, te neka je h nenul element u $\mathbb{F}_p$. Problem diskretnog logaritma je pronalaženje vrijednosti x takve da je:

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

Vrijednost x zovemo diskretni logaritam od h u bazi g, te označavamo s $\log_g(h)$.

Primjedba 2. Stariji naziv za diskretni logaritam je indeks, te se označava s $\text{ind}_g(h)$. Izraz indeks i dalje se koristi u teoriji brojeva. Prikladan je jer se može dogoditi zabuna između običnih logaritama i diskretnih logaritama jer se npr. $\log_2$ može pojaviti u oba slučaja.

Definicija 3.3. Kažemo da je problem dobro postavljen ukoliko on zadovoljava sljedeća tri kriterija:

1. Rješenje postoji
2. Rješenje je jedinstveno
3. Rješenje je stabilno (mali pomaci u ulaznim podacima rezultiraju malim pomacima u rješenju, analogno za velike pomake).

Primjedba 3. Problem diskretnog logaritma nije dobro postavljen, to jest problem pronalaska cjelobrojne potencije x takve da je $g^x = h$. Jer ako postoji jedno rješenje, tada postoji beskonačno mnogo rješenja jer po malom Fermatovom teoremu vrijedi $g^{p-1} \equiv 1 \pmod{p}$. Stoga, ako je x rješenje za $g^x = h$ tada je $x + k(p-1)$ također rješenje za svaki k jer vrijedi:

$$g^{x+k(p-1)} = g^x \cdot (g^{p-1})^k \equiv h \cdot 1^k \equiv h \pmod{p}$$

Dakle, $\log_g(h)$ definiran je do na zbrajanje i oduzimanje višekratnika od $p-1$. Drugim riječima, $\log_g(h)$ je definiran modulo $p-1$. Uočimo da je $\log_g$ funkcija:

$$\log_g : \mathbb{F}_p^* \longrightarrow \mathbb{Z}/(p-1)\mathbb{Z}.$$

Nekada, zbog konkretnosti, opisujemo diskretni logaritam kao cijeli broj x iz intervala $[0, p-2]$ koji zadovoljava kongruenciju $g^x \equiv h \pmod{p}$.

Primjedba 4. Nije teško dokazati da vrijedi:

$$\log_g(ab) = \log_g(a) + \log_g(b) \quad \text{za svaki } a, b \in \mathbb{F}_p^*$$

Iz tog razloga intuitivno je zvati $\log_g$ logaritmom jer pretvara množenje u zbrajanje na isti način kao i standardna logaritamska funkcija. U matematici diskretni logaritam $\log_g$ je izomorfizam grupa $\mathbb{F}_p^*$ i $\mathbb{Z}/(p-1)\mathbb{Z}$.

Primjer. Broj $p = 56509$ je prost, te je $g = 2$ primitivan korijen modulo p. Kako izračunati diskretan logaritam od $h = 38679$? Prvi način koji nam padne na pamet je:

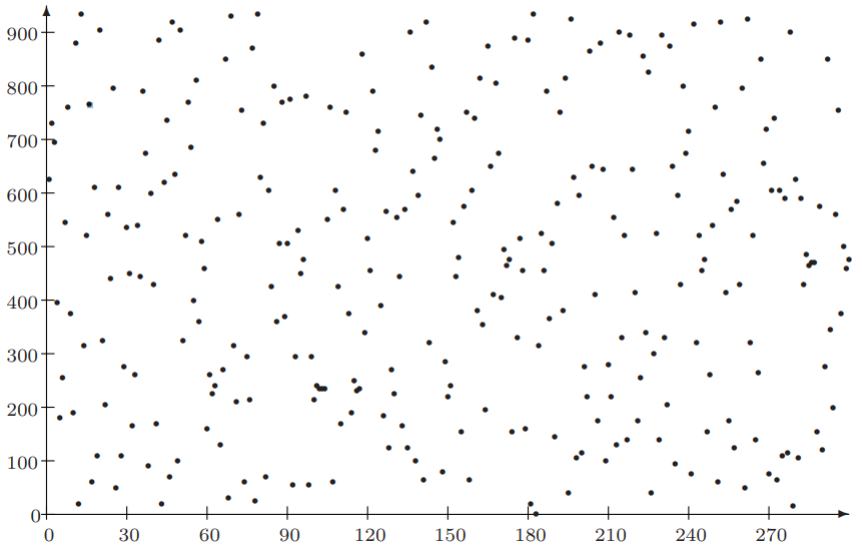
$$2^2, 2^3, 2^4, 2^5, \ldots \pmod{56509}$$

nastavimo sve dok ne nađemo potenciju koja daje 38679. Teško je ovaj postupak raditi ručno, ali koristeći računalo brzo vidimo da je $\log_2(h) = 11235$. Možemo provjeriti tako da izračunamo $2^{11235} \mod 56509$ i vidimo da dobijemo 38679.

Primjedba 5. Na sljedećim primjerima uvidjet ćemo da se diskretni logritam i standardni logaritam koje je definiran na realnim i kompleksnim brojevima ne ponašaju slično. Ime logaritma je ostalo jer su obje funkcije inverzi potenciranja, no potenciranje modulo p ponaša se vrlo nepravilno s obzirom na eksponent, za razliku od običnog potenciranja. Ponašanje potenciranja modulo p izgleda veoma nasumično kao što možemo vidjeti u tablici 1 i slici 3.

Tablica 1 Potenciranje modulo p

n	$g^n \mod p$	n	$g^n \mod p$	h	$\log_g(h)$	h	$\log_g(h)$
1	627	11	878	1	0	11	429
2	732	12	21	2	183	12	835
3	697	13	934	3	469	13	279
4	395	14	316	4	366	14	666
5	182	15	522	5	356	15	825
6	253	16	767	6	652	16	732
7	543	17	58	7	483	17	337
8	760	18	608	8	549	18	181
9	374	19	111	9	938	19	43
10	189	20	904	10	539	20	722



Slika 3: Potencije $627^i \mod 941$ za $i = 1, 2, 3, \ldots$

Primjedba 6. Naša definicija diskretnog logaritma ima pretpostavku da je baza g primitivan korijen modulo p, ali to nije uvijek nužan uvjet. Generalno, za svaki $g \in \mathbb{F}_p^*$ i svaki $h \in \mathbb{F}_p^*$ problem diskretnog logaritma svodi se na određivanje potencije x koja zadovoljava $g^x \equiv h \pmod{p}$, pod uvjetom da takav x postoji.

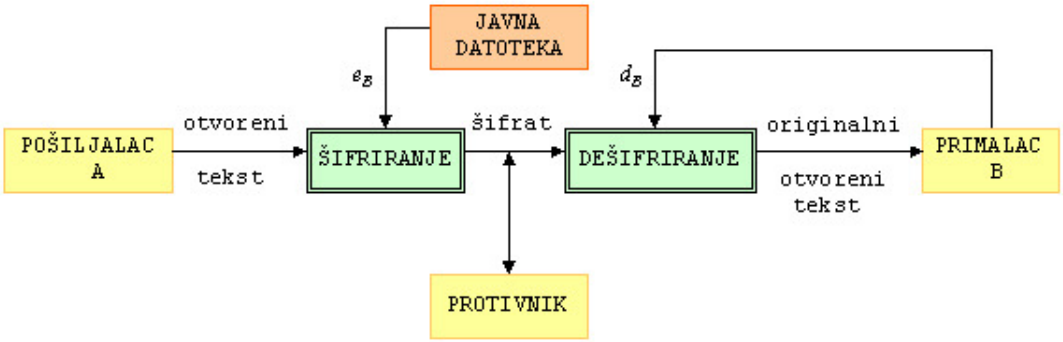
Ovaj problem možemo generalizirati tako da umjesto što uzimamo nenul elemente konačnog polja $\mathbb{F}_p$, množimo i potenciramo ih, možemo uzeti bilo koju grupu i koristiti množenje definirano u njoj. Tako dobijemo generalizirani oblik problema diskretnog logaritma.

Definicija 3.4. Neka je G grupa s pripadnom operacijom $\star$. Problem diskretnog logaritma na G je određivanje vrijednosti x za neke elemente g i h iz G takve da vrijedi:

$$\underbrace{g \star g \star g \star \cdots \star g}_{x \text{ puta}} = h.$$

3.1 Diffie-Hellmanov protokol za razmjenu ključeva

Diffe-Hellmanov protokol koristi se za rješavanje sljedećeg problema. Ivan i Ana žele razmijeniti tajni ključ kako bi ga koristili u simetričnoj šifri, ali njihov jedini način komunikacije nije siguran. Sve što njih dvoje komuniciraju sa strane promatra i osluškuje Lea. Kako Ivan i Ana mogu razmjeniti ključeve bez da budu dostupni Lei? Diffie i Hellman su veoma inovativno iskoristili teškoću problema diskretnog logaritma na $\mathbb{F}_p^*$ kako bi osigurali navedenu komunikaciju.



Slika 4: Prikaz komunikacije kroz nesiguran kanal

Prvi korak je da Ivan i Ana odaberu veliki prost broj p i prirodan broj g modulo p. Ivan i Ana javno objavljuju vrijednosti p i g; npr. objave ih na svojim Facebook profilima kako bi ih i Lea saznala.

Idući korak je da Ana odabere tajni cijeli broj a koji ne otkriva nikome. Istovremeno Ivan bira takav tajni broj b. Ivan i Ana koriste svoje tajne brojeve kako bi izračunali:

$$\underbrace{A \equiv g^a \pmod{p}}_{\text{ovo računa Ana}} \text{ i } \underbrace{B \equiv g^b \pmod{p}}_{\text{ovo računa Ivan}}.$$

Nadalje, razmjenjuju izračunate vrijednosti, Ana Ivanu šalje A, te Ivan Ani šalje B. Napomenimo da Lea sada zna vrijednosti A i B jer su poslane nesigurnim putem. Konačno Ivan i Ana računaju sljedeće vrijednosti:

$$\underbrace{A' \equiv B^a \pmod{p}}_{\text{ovo računa Ana}} \; i \; \underbrace{B' \equiv A^b \pmod{p}}_{\text{ovo računa Ivan}}.$$

Izračunate vrijednosti A' i B' jednake su:

$$A' \equiv B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \equiv B' \pmod{p}.$$

Ta vrijednost je sada njihov zajednički ključ. Sažetak Diffie-Hellmanovog protokola pogledajmo u sljedećoj tablici:

Stvaranje javnog parametra ključa	
Izabire se i javno objavljuje veliki prost broj p i cijeli broj G velikog prostog reda u $\mathbb{F}_p^*$.	
Tajni izračuni	
Ana	Ivan
Odabire tajnu vrijednost a.	Odabire tajnu vrijednost b.
Računa $A \equiv g^a \pmod{p}$	Računa $B \equiv g^b \pmod{p}$
Javna razmjena vrijednosti	
Ana šalje A Ivanu $\longrightarrow A$	
$B \longleftarrow$ Ivan šalje B Ani	
Privatni izračun ključa	
Ana	Ivan
Izračuna vrijednost $B^a \pmod{p}$.	Izračuna vrijednost $A^b \pmod{p}$.
Konačna podijeljena tajna vrijednost je $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod{p}$.	

Primjer. Ivan i Ana dogovorili su se da će koristiti prosti broj $p = 941$ i primitivni korjen $g = 627$. Ana odabire tajni ključ $a = 347$ i računa vrijednost $A = 390 \equiv 627^{347} \pmod{941}$. Slično, Ivan odabire tajni ključ $b = 781$ i računa vrijednost $B = 691 \equiv 627^{781} \pmod{941}$. Ana šalje Ivanu broj 390, te Ivan šalje Ani broj 691. Oba prijenosa vrijednosti odvila su se nesigurnim putem. Stoga smatramo da su obje vrijednosti $A = 390$ i $B = 691$ javne, dok brojeve $a = 347$ i $b = 781$ nismo slali te oni ostaju tajni. Sada Ivan i Ana mogu izračunati vrijednost

$$470 \equiv 627^{347 \cdot 781} \equiv A^b \equiv B^a \pmod{941},$$

te je 470 njihova tajna. Ako pretpostavimo da je Lea cijelo vrijeme promatrala, ona može otkriti Aninu i Ivanovu tajnu ukoliko riješi jednu od kongruencija.

$$627^a \equiv 390 \pmod{941} \text{ ili } 627^b \equiv 691 \pmod{941},$$

jer će tada znati jedan od tajnih eksponenata. To je, koliko znamo, jedini način da Lea otkrije Ivanovu i Aninu tajnu.

Naravno, naš primjer koristi brojeve koji su premali da bi osigurali stvarnu sigurnost Ivanu i Ani, jer svako računalo može u vrlo malo vremena provjeriti sve potencije broja 627 modulo 941. Standardno se predlaže da Ivan i Ana izaberu prost broj p koji u memoriji računala zauzima oko 1000 bitova (npr. $p \approx 2^{1000}$) i broj g koji je prostog reda q gdje je $q \approx p/2$. Tako bi Lea dobila uistinu težak zadatak za računanje. Generalno, Lea ima sljedeći problem. Poznate su joj vrijednosti A i B te stoga zna i vrijednosti g^a i g^b. Također zna vrijednosti g i p, stoga, kada bi mogla riješiti problem diskretnog logaritma, mogla bi izračunati a i b, te pomoću njih izračunati tajnu vrijednost g^{ab}. Čini se da se Ivanova i Anina sigurna komunikacija temelji na pretpostavci da Lea ne zna riješiti problem diskretnog logaritma, ali to nije posve točno. Točno je da se njihova tajna vrijednost može saznati rješavanjem problema diskretnog logaritma, ali to nije točno problem koji Lea mora riješiti. Sigurnost Ivanove i Anine komunikacije ovisi o sljedećem, potencijalno jednostavnijem, problemu.

Definicija 3.5. Neka je p prost broj i g prirodan broj. Diffie-Hellmanov problem je problem računanja vrijednosti $g^{ab} \pmod p$ iz poznatih vrijednosti $g^a \pmod p$ i $g^b \pmod p$.

Jasno je da Diffie-Hellmanov problem nije teži nego problem diskretnog logaritma. Ako Eva može riješiti problem diskretnog logaritma, tada može izračunati tajne vrijednosti a i b iz presretnutih vrijednosti $A = g^a$ i $B = g^b$, te joj je tada jednostavno izračunati Ivanovu i Aninu tajnu vrijednost g^{ab}. Vrijedi li obratna situacija? Ako Lea ima efikasan algoritam za rješavanje Diffie-Hellmanovog problema, može li ga iskoristiti efikasno za rješavanje problema diskretnog logaritma? To još uvijek nije poznato.

3.2 Primjena Diffie-Hellmanovog protokola

Najčešća primjena Diffie-Hellmanovog protokola je u mrežnim protokolima koji omogućuju zaštićenu komunikaciju. Neki od njih su:

- autentikacija u mrežama računala,
- u IPsec (eng. IP security) protokolu,
- unutar IKE (eng. Internet Key Exchange) strukture,
- SSH i TLS protokolima te
- u postupku stvaranja digitalnih potpisa i digitalnih certifikata.

Autentikacija je postupak provjere osobe ili računala s kojim se komunicira u smislu da se ne predstavlja lažno. Kompleksni protokoli koji omogućavaju provjeru identiteta udaljenog procesa temeljeni su na kriptografiji. Komunikacija koja koristi takav protokol opisana je u sljedećem primjeru.

Primjer. Ana prva šalje poruku Ivanu. On joj odgovora te nakon toga razmjene još nekoliko poruka. Budući da se komunikacija odvija kroz nesiguran kanal Lea može presresti poruke, izmijeniti ih ili odgovoriti na njih kako bi sabotirala komunikaciju Ane i Ivana. Znamo da se Ani i Ivanu uljez nije ubacio u komunikaciju jer koriste protokol s autentikacijom. U većini slučajeva za sudjelovanje u komunikaciji Lei je potreban sjednički ključ koji samo Ivan i Ana imaju. Sjednički ključ je tajni ključ kojim se kriptiraju poruke koje se razmjenjuju tokom trajanja sjednice. U praksi su svi podaci koji se šalju nesigurnim kanalom kriptirani upotrebom simetričnog kriptosustava pri čemu se razmjena tajnog ključa obavlja Diffie-Hellmanovim protokolom.

IPsec je standard definiran od strane IETF-a[3], a cilj njegove izrade bio je siguran transport informacija preko javnih IP mreža. IPsec uključuje protokole za uspostavu međusobne autentikacije računala ili procesa na početku sjednice i za dogovor kriptografskog sjedničkog ključa koji će se koristiti tokom sjednice. IPsec se može koristiti za zaštitu toka podataka između korisnika i poslužitelja, usmjerivača (eng. router) i vatrozida (eng. firewall). Zaštitni mehanizmi IPsec protokola se između ostalog zasnivaju na protokolima za razmjenu ključeva, kao što je IKE protokol. Ti protokoli obično koriste Diffie-Hellmanov protokol za razmjenu tajnog ključa koji se dalje koristi kao sjednički ključ.

SSH (eng. Secure Shell) je IETF protokol za sigurnu komunikaciju između udaljenih odredišta u nesigurnoj računalnoj mreži kao što je na primjer internet. Oblikovan je prema

[3]Internet Engineering Task Force (IETF) je organizacija koja razvija i promovira Internet standarde

klijent/poslužitelj modelu, a komunikacija je podijeljena u tri sloja: transportni, autentifikacijski i spojni. Sigurnost se u SSH protokolu postiže primjenom kriptografskih algoritama koji štite tajnost i integritet podataka u prometu te osiguravaju autentičnost sudionika komunikacije. SSH protokol kao i IPsec koristi Diffie-Hellmanov protokol za razmjenu ključeva.

Dodatno, Diffie-Hellmanov protokol koristi se u postupku stvaranja digitalnih certifikata i digitalnih potpisa. Digitalnim potpisima provjeravamo autentičnost dokumenata, dok digitalni certifikati služe za utvrđivanje identiteta i autentifikaciju korisnika za vrijeme obavljanja transakcija na internetu.

3.3 Napad s čovjekom u sredini na Diffie-Hellmanov protokol

U poglavlju 3 spomenuli smo obični Diffie-Hellmanov protokol i verziju s eliptičnim krivuljama. Kod obje verzije sigurnost se temelji na pretpostavci da je teško riješiti problem diskretnog logaritma. Nažalost postoji mana Diffie-Hellmanovog protokola, a to je ranjivost na napad s čovjekom u sredini (eng. man in the middle). U takvom napadu uljez presreće poslane poruke od Ivana i Ane, te ih zamjenjuje svojima. Promotrimo detaljno napad s čovjekom u sredini na sljedećem primjeru.

Primjer. Uljez odabire nasumični broj z i poznavajući objavljene brojeve p i g računa:

$$Z = g^z \pmod{p}$$

Ana šalje Ivanu poruku koja sadrži vrijednost X, napadač presretne tu poruku, spremi vrijednost X, te Ivanu umjesto Anine poruke šalje poruku koja sadrži vrijednost Z. Analogno uljez presretne Ivanovu poruku Y, zadrži ju za sebe i Ani pošalje poruku Z. U ovom trenutku Ana i Ivan nisu svjesni da su umjesto pravih poruka dobili poruke koje šalje uljez.
Uljez računa vrijednosti ključeva K_A i K_B :

$$K_A = X^z \pmod{p} = (g^x)^z \pmod{p} = g^{xz} \pmod{p},$$

$$K_B = Y^z \pmod{p} = (g^y)^z \pmod{p} = g^{yz} \pmod{p}.$$

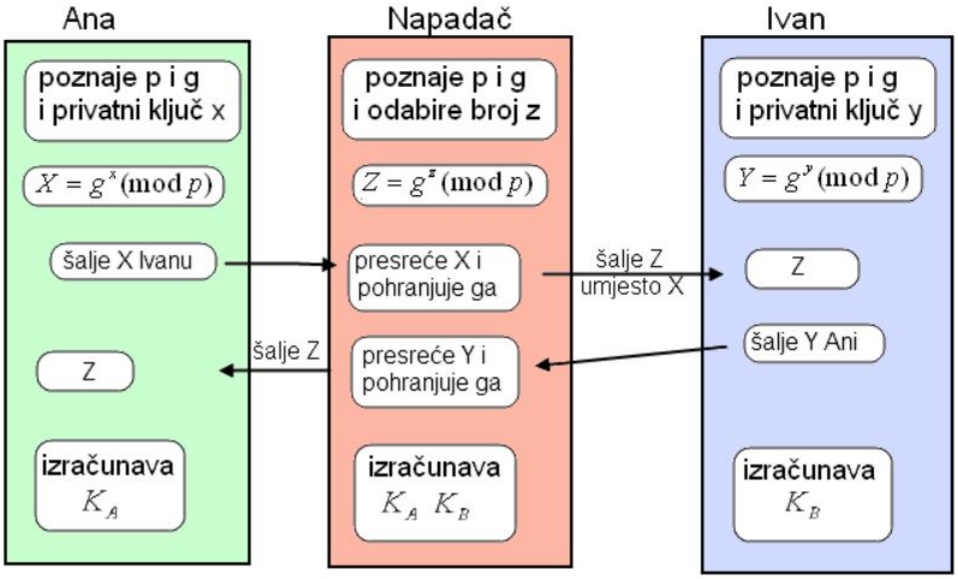
Ana računa vrijednost ključa:

$$K_A = Z^x \pmod{p} = (g^z)^x \pmod{p} = g^{xz} \pmod{p}.$$

Ivan računa vrijednost ključa:

$$K_B = Z^y \pmod{p} = (g^z)^y \pmod{p} = g^{yz} \pmod{p}.$$

Ukoliko nakon ovakve razmjene ključeva Ana i Ivan nastave komunicirati koristeći izračunate tajne ključeve, napadač može narušiti tajnost i autentičnost komuniciranja.



Slika 5: Napad s čovjekom u sredini

Ukoliko se dogodio ovakav napad dolazi do sljedeće situacije. Ana šalje kriptiranu poruku $M_1 = e_{K_A}(T)$, gdje je T otvoreni tekst, napadač ju presretne i dekriptira te saznaje sadržaj poruke $T = d_{K_A}(M_1) = d_{K_A}(e_{K_A}(T))$. Napadač u ovoj situaciji ima mogućnost tekst T ili neki izmišljeni tekst T' kriptirati i poslati Ivanu poruku $M_2 = e_{K_B}(T')$. Ivan misli da je dobio poruku od Ane i svojim ključem K_B dekriptira poruku te dobiva $T' = d_{K_B}(M_2) = d_{K_B}(e_{K_B}(T'))$. Istim postupkom može se ostvariti tok informacija u suprotnom smjeru. Ana i Ivan ne mogu znati da se u njihovu razmjenu poruka umješao napadač, odnosno čovjek u sredini. Zaključujemo kako je ovakve i slične razmjene potrebno dodatno zaštititi postupkom autentikacije sudionika u komunikaciji.

3.4 ElGamalov kriptosustav s javnim ključem

Iako Diffie-Hellmanov protokol za razmjenu ključa daje metodu za razmjenu tajnog nasumičnog ključa, on ne ispunjava sve uvjete da bi bio kriptosustav s javnim ključem jer kriptosustavi omogućavaju razmjenu bilo koje vrste informacija, a ne samo nasumične stringove. Prvi kriptosustav javnog ključa bio je RSA kriptosustav od autora Rivest, Shamir i Adleman. Objavili su ga 1978. godine. RSA kriptosustav bio je i ostao veliko otkriće. Iako je RSA bio prvi svoje vrste, možda nam je logičniji kriptosustav koji je 1985. godine opisao Taher ElGamal jer se nadovezao na rad koji su Diffie i Hellman objavili. ElGamalov algoritam enkripcije s javnim ključem temelji se na problemu diskretnog logaritma te je sličan Diffie-Hellmanovom protokolu. U ovom poglavlju opisati ćemo inačicu ElGamalovog kriptosustava s javnim ključem koji se temelji na problemu diskretnog logaritma za $\mathbb{F}_p^*$, ali postupak je jako sličan problemu diskretnog logaritma na bilo kojoj grupi.

ElGamalov kriptosustav s javnim ključem prvi je kriptosustav koji opisujemo stoga ćemo sve korake detaljno obrazložiti. Ana prvo objavi informaciju koja se sastoji od javnog ključa i algoritma. Javni ključ je broj, dok je algoritam metoda kojom će Ivan šifrirati svoju poruku koristeći Anin javni ključ. Ana ne objavljuje svoj tajni ključ koji je neki drugi broj. Tajni ključ omogućuje samo Ani da dešifrira poruku koja je šifrirana koristeći njen javni ključ.

Ovo je bilo dosta pojednostavljeno objašnjenje koje se može primjeniti na sve kritposustave s javnim ključem. Za ElGamalov kriptosustav s javnim ključem Ana treba veliki prost

broj p za koji je problem diskretnog logaritma na $\mathbb{F}_p^*$ teško riješiti. Također treba element g modulo p velikog prostog reda. Može sama odabrati p i g ili mogu biti unaprijed odabrani pomoću softvera ili od strane vladine agencije.

Ana odabire tajni broj a koji ima funkciju njenog privatnog ključa, te računa vrijednost

$$A \equiv g^a \pmod{p}.$$

Uočimo sličnost s Diffie-Hellmanovim protokolom. Ana objavljuje A, ali tajnim ostaje ključ a.

Nadalje, neka Ivan želi šifrirati poruku koristeći Anin javni ključ A. Pretpostavit ćemo da je Ivanova poruka m koji je cijeli broj iz intervala $[2, p]$. Za pretvaranje tekstualne poruke u broj koristit ćemo sljedeću proceduru:

Neka je poruka koju želimo šifrirati: "Bed bug." uključujući razmake i interpunkciju. Ukoliko pogledamo sljedeću tablicu:

ASCII TABLE

Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]
1	1	1	1	[START OF HEADING]
2	2	10	2	[START OF TEXT]
3	3	11	3	[END OF TEXT]
4	4	100	4	[END OF TRANSMISSION]
5	5	101	5	[ENQUIRY]
6	6	110	6	[ACKNOWLEDGE]
7	7	111	7	[BELL]
8	8	1000	10	[BACKSPACE]
9	9	1001	11	[HORIZONTAL TAB]
10	A	1010	12	[LINE FEED]
11	B	1011	13	[VERTICAL TAB]
12	C	1100	14	[FORM FEED]
13	D	1101	15	[CARRIAGE RETURN]
14	E	1110	16	[SHIFT OUT]
15	F	1111	17	[SHIFT IN]
16	10	10000	20	[DATA LINK ESCAPE]
17	11	10001	21	[DEVICE CONTROL 1]
18	12	10010	22	[DEVICE CONTROL 2]
19	13	10011	23	[DEVICE CONTROL 3]
20	14	10100	24	[DEVICE CONTROL 4]
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]
22	16	10110	26	[SYNCHRONOUS IDLE]
23	17	10111	27	[ENG OF TRANS. BLOCK]
24	18	11000	30	[CANCEL]
25	19	11001	31	[END OF MEDIUM]
26	1A	11010	32	[SUBSTITUTE]
27	1B	11011	33	[ESCAPE]
28	1C	11100	34	[FILE SEPARATOR]
29	1D	11101	35	[GROUP SEPARATOR]
30	1E	11110	36	[RECORD SEPARATOR]
31	1F	11111	37	[UNIT SEPARATOR]
32	20	100000	40	[SPACE]
33	21	100001	41	!
34	22	100010	42	"
35	23	100011	43	#
36	24	100100	44	$
37	25	100101	45	%
38	26	100110	46	&
39	27	100111	47	'
40	28	101000	50	(
41	29	101001	51	)
42	2A	101010	52	*
43	2B	101011	53	+
44	2C	101100	54	,
45	2D	101101	55	-
46	2E	101110	56	.
47	2F	101111	57	/
48	30	110000	60	0
49	31	110001	61	1
50	32	110010	62	2
51	33	110011	63	3
52	34	110100	64	4
53	35	110101	65	5
54	36	110110	66	6
55	37	110111	67	7
56	38	111000	70	8
57	39	111001	71	9
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	A
66	42	1000010	102	B
67	43	1000011	103	C
68	44	1000100	104	D
69	45	1000101	105	E
70	46	1000110	106	F
71	47	1000111	107	G
72	48	1001000	110	H
73	49	1001001	111	I
74	4A	1001010	112	J
75	4B	1001011	113	K
76	4C	1001100	114	L
77	4D	1001101	115	M
78	4E	1001110	116	N
79	4F	1001111	117	O
80	50	1010000	120	P
81	51	1010001	121	Q
82	52	1010010	122	R
83	53	1010011	123	S
84	54	1010100	124	T
85	55	1010101	125	U
86	56	1010110	126	V
87	57	1010111	127	W
88	58	1011000	130	X
89	59	1011001	131	Y
90	5A	1011010	132	Z
91	5B	1011011	133	[
92	5C	1011100	134	\
93	5D	1011101	135	]
94	5E	1011110	136	^
95	5F	1011111	137	_
96	60	1100000	140	`
97	61	1100001	141	a
98	62	1100010	142	b
99	63	1100011	143	c
100	64	1100100	144	d
101	65	1100101	145	e
102	66	1100110	146	f
103	67	1100111	147	g
104	68	1101000	150	h
105	69	1101001	151	i
106	6A	1101010	152	j
107	6B	1101011	153	k
108	6C	1101100	154	l
109	6D	1101101	155	m
110	6E	1101110	156	n
111	6F	1101111	157	o
112	70	1110000	160	p
113	71	1110001	161	q
114	72	1110010	162	r
115	73	1110011	163	s
116	74	1110100	164	t
117	75	1110101	165	u
118	76	1110110	166	v
119	77	1110111	167	w
120	78	1111000	170	x
121	79	1111001	171	y
122	7A	1111010	172	z
123	7B	1111011	173	{
124	7C	1111100	174	|
125	7D	1111101	175	}
126	7E	1111110	176	~
127	7F	1111111	177	[DEL]

Slika 6: ASCII tablica

te za svaki znak uzmemo njegovu pripadnu binarnu vrijednost zapisanu u 8 bitova (ukoliko je zapis kraći nadopunimo nulama na početku).

B	e	d		b	u	g	.
66	101	100	32	98	117	103	46
01000010	01100101	01100100	00100000	01100010	01110101	01100111	00101110

Zatim sve binarne zapise spojimo redom u jedan string. Dakle, poruku "Bed bug." zapišemo kao:

$$01000010011001010110010000100000011000100111010101100111001011 10$$

Ukoliko taj binarni zapis pretvorimo u decimalni dobijemo broj $m = 4784340269404612398$.

Kako bi šifrirao m, Ivan prvo nasumično odabire broj k modulo p. Ivan koristi k kako bi šifrirao točno jednu poruku, zatim taj k odbacuje. Broj k zovemo nasumičnim elementom te mu je svrha da se koristi u šifriranju točno jedne poruke.

Ivan odabire svoju poruku m, nasumični element k i Anin javni ključ A te ih koristi da bi izračunao sljedeće dvije vrijednosti

$$c_1 \equiv g^k \pmod{p} \text{ i } c_2 \equiv mA^k \pmod{p}.$$

Prisjetimo se da su g i p javni, stoga Ivan zna njihove vrijednosti. Ivanov šifrat npr. šifrat od m je uređeni par (c_1, c_2) koji šalje Ani. Postavlja se pitanje kako će Ana dešifrirati Ivanov šifrat. Budući da Ana zna a, može izračunati vrijednost

$$x \equiv (c_1^a)^{-1} \pmod{p}.$$

To može učinit na sljedeći način. Prvo izračuna $c_1^1 \pmod{p}$ koristeći algoritam brzog potenciranja opisanog u primjedbi 7, zatim izračuna inverz koristeći prošireni Euklidov algoritam.

Primjedba 7. Algoritam brzog potenciranja temelji se na sljedećem svojstvu.

$$\text{Za pozitivan } n \text{ vrijedi: } x^n = \begin{cases} x(x^2)^{\frac{n-1}{2}}, & \text{za neparan } n \\ (x^2)^{\frac{n}{2}}, & \text{za paran } n \end{cases}$$

Metoda brzog potenciranja koristi binarni zapis eksponenta n i zajednički faktor x^2 kako bi u manje koraka izračunali x^n. Na primjer, želimo izračunati x^{13} koristeći navedenu metodu. Eksponent je 13, te je njegov binarni zapis 1101. Bitove gledamo s lijeva na desno. Eksponent ima 4 bita, stoga imamo 4 koraka algoritma.

Početno postavimo $r = 1(= x^0)$.

1. $r = r^2(= x^0)$; Prvi bit jednak je 1, stoga računamo $r = r \cdot x(= x^1)$
2. $r = r^2(= x^2)$; Drugi bit jednak je 1, stoga računamo $r = r \cdot x(= x^3)$
3. $r = r^2(= x^6)$; Treći bit jednak je 0 dakle, ne množimo r sa x kao u prethodnim koracima nego ostaje $r = r(= x^6)$
4. $r = r^2(= x^{12})$; Četvrti bit jednak je 1, stoga računamo $r = r \cdot x(= x^{13})$

Alternativno, može jednostavno koristiti brzo potenciranje kako bi izračunala $c_1^{p-1-a} \pmod{p}$. Ana onda množi c_2 i x te time dobiva početni tekst m. Da bismo vidjeli kako, probat ćemo raspisati

$$
\begin{aligned}
x \cdot c_2 &\equiv (c_1^a)^{-1} \cdot c_2 && \pmod{p}, && \text{jer } x \equiv (c_1^a)^{-1} \pmod{p},\\
&\equiv (g^{ak})^{-1} \cdot (mA^k) && \pmod{p}, && \text{jer } c_1 \equiv g^k, c_2 \equiv mA^k \pmod{p},\\
&\equiv (g^{ak})^{-1} \cdot (m(g^a)^k) && \pmod{p}, && \text{jer } A \equiv g^a \pmod{p},\\
&\equiv m && \pmod{p}, && \text{jer se izrazi } g^{ak} \text{ pokrate.}
\end{aligned}
$$

Sažeti ElGamalov kriptosustav s javnim ključem možemo vidjeti u sljedećoj tablici:

Stvaranje javnog parametra ključa	
Povjerljiva osoba bira i objavljuje veliki prosti broj p i element $g \mod p$ koji je velikog prostog reda.	
Ana	**Ivan**
Stvaranje ključa	
Odabire tajni ključ $1 \leq a \leq p-1$. Računa $A = g^a \pmod{p}$. Objavljuje javni ključ A	
Šifriranje	
	Odabire vrijednost m. Odabire nasumični element k. Koristi Anin javni ključ A kako bi izračunao $c_1 \equiv g^k \pmod{p}$ i $c_2 \equiv mA^k \pmod{p}$. Šalje šifrat (c_1, c_2) Ani.
Dešifriranje	
Računa $(c_1^a)^{-1} \cdot c_2 \pmod{p}$. Ta vrijednost jednaka je m	

Kako Lea može dešifrirati poruku? Lea zna parametre p i g, također zna vrijednost $A \equiv g^a \pmod{p}$, jer je Anin ključ A javan. Ako Lea može riješiti ovaj problem diskretnog logaritma, tada može izračunati a i dešifrirati poruku. Preciznije, dovoljno je da Lea riješi Diffie-Hellmanov problem. Pogledajmo na primjeru s malim brojevima.

Primjer. Ana odabire prosti broj $p = 467$ i primitivni korijen $g = 2$. Odabire $a = 153$ kao svoj privatni ključ, te računa javni ključ:

$$A \equiv g^a \equiv 2^{153} \equiv 224 \pmod{467}.$$

Ivan odlučuje Ani poslati poruku $m = 331$. Odabire nasumični element $k = 197$, te računa sljedeće dvije vrijednosti:

$$c_1 \equiv 2^{197} \equiv 87 \pmod{467} \text{ i } c_2 \equiv 331 \cdot 224^{197} \equiv 57 \pmod{467}.$$

Uređeni par $(c_1, c_2) = (87, 57)$ je šifrat koji Ivan šalje Ani.

Budući da Ana zna vrijednost $a = 153$, prvo računa

$$x \equiv (c_1^a)^{-1} \equiv c_1^{p-1-a} \equiv 87^{313} \equiv 14 \pmod{467}.$$

Konačno, računa

$$c_2 x \equiv 57 \cdot 14 \equiv 331 \pmod{467}$$

te tako dobije originalni tekst m.

Primjedba 8. U ElGamalovom kriptosustavu, otvoreni tekst je prirodan broj m između 2 i $p-1$, dok se šifrat sastoji od dva cijela broja c_1 i c_2 iz istog intervala kao i m. Stoga, zapisivanje šifrata zauzima 2 puta više bitova nego zapisivanje originalnog teksta. Zato kažemo da ElGamalov kriptosustav ima 2-na-1 ekspanziju poruke.

Sada se postavlja pitanje, jesmo li s ElGamalovim kriptosustavom stvorili teži problem za Leu nego što je to bio slučaj sa Diffie-Hellmanovim problemom? Moderna kriptografija bavi se identificiranjem teških problema kao što je npr. Diffie-Hellmanov problem koji se nalaze u pozadini kriptosustava kao npr. ElGamalov, te dokazivanjem da je proboj kriptosustava jednako težak ili teži od rješavanja pozadinskog problema.

Pokazat ćemo da svatko tko može dešifrirati šifrat ElGamalovog kriptosustava kojeg smo opisali, također mora moći riješiti Diffie-Hellmanov problem. Točnije, dokazat ćemo sljedeću tvrdnju:

Propozicija 1. Neka su p prosti broj i g baza koje ćemo koristiti u ElGamalovom kriptosustavu. Pretpostavimo da Lea ima pristup stroju koji dešifrira šifrat dobiven ElGamalovim kriptosustavom. Tada može iskoristiti taj stroj da riješi Diffie-Hellmanov problem.

Dokaz. Umjesto sažetog i formalnog dokaza, dokazat ćemo prethodnu propoziciju na malo opširniji način kako bismo demonstrirali korištenje stroja koji dešifrira Elgamanov kriptosustav za rješavanje Diffie-Hellmanovog problema. Prisjetimo se da kod Diffie-Hellmanovog problema Lea zna sljedeće dvije vrijednosti

$$A \equiv g^a \pmod{p} \quad \text{i} \quad B \equiv g^b \pmod{p}$$

te mora izračunati vrijednost $g^{ab} \pmod{p}$. Imajući na umu da zna vrijednosti A i B, ali ne zna niti a niti b.

Pretpostavimo da se Lea može koristiti navedenim strojem. To znači da Lea može u njega unijeti prosti broj p, bazu g, javni ključ A i šifrat (c_1, c_2). U skladu s našim opisom Elgamanovog kriptosustava, stroj vraća vrijednost

$$(c_1^a)^{-1} \cdot c_2 \pmod{p}.$$

Da bi Lea riješila Diffie-Hellmanov problem, kako treba izabrati vrijednosti c_1 i c_2? Dobar odabir bi bio $c_1 = B = g^b$ i $c_2 = 1$ jer tada stroj vraća $(g^{ab})^{-1} \pmod{p}$. Tada Lea može uzeti inverz modulo p od vrijednosti koje je stroj vratio, te tako dobije $g^{ab} \pmod{p}$, što je rješenje Diffie-Hellmanovog problema.

No što ako je stroj zaštićen na način da je isprogramiran da ne vraća ništa ukoliko je $c_2 = 1$ Lea i dalje može prevariti stroj. Može odabrati poroizvoljnu vrijednost c_2, te u stroj unijeti javni ključ A i šifrat (B, c_2). Stroj vraća otvoreni tekst m za koji vrijedi:

$$m \equiv (c_1^a)^{-1} \cdot c_2 \equiv (B^a)^{-1} \cdot c_2 \equiv (g^{ab})^{-1} \cdot c_2 \pmod{p}.$$

Nakon toga Lea jednostavno izračuna

$$m^{-1} \cdot c_2 \equiv g^{ab} \pmod{p}$$

kako bi dobila vrijednost $g^{ab} \pmod{p}$. □

Važno je razlikovati da je Lea koristeći stroj izračunala $g^{ab} \pmod{p}$ bez da zna vrijednosti a i b, stoga je riješila Diffie-Hellmanov problem, a ne problem diskretnog logaritma.

3.5 Koliko je težak problem diskretnog logaritma?

Neka je G grupa, te neka su $g, h \in G$. Problem diskretnog logaritma podrazumijeva pronalazak vrijednosti x za koju vrijedi $g^x = h$. Koliko je ovaj problem stvarno težak, te kako možemo uopće definirati "težak"?

Intuitivno težinu možemo procijeniti brojem operacija koje su potrebne čovjeku ili računalu da riješi problem najefikasnijom poznatom metodom. Na primjer, možemo riješiti problem diskretnog logaritma računajući $g, g^2, g^3, ...$ dok ne naiđemo na vrijednost koja je jednaka h. Ako je g reda n, tada ovaj algoritam uvijek pronalazi rješenje u najviše n množenja. Problem nastaje ako je n jako velik, npr. $n > 2^{80}$, tada navedeni algoritam, uz današnju tehnologiju, ne možemo uzeti u obzir.

Alternativno, možemo uzimati nasumične vrijednosti x, računati g^x, te provjeravati vrijedi li $g^x = h$. Koristeći metodu brzog potenciranja, za izračun g^x potrebno nam je $t \cdot \log_2 x$ operacija množenja. Ako su n i x brojevi koji se zapisuju u k bitova, to jest, oba iznose otprilike 2^k, tada ova metoda zahtijeva otprilike $k \cdot 2^k$ množenja. Ako računamo nad grupom $\mathbb{F}_p^*$ i smatramo operaciju zbrajanje modulo p kao osnovnu operaciju, tada množenje modulo p dvaju k-bitnih brojeva zahtjeva otprilike k^2 osnovnih operacija, stoga, za rješavanje problema diskretnog logaritma koristeći ovu metodu potrebno nam je $t \cdot k^2 \cdot 2^k$ osnovnih operacija.

Koliko iznosi t koji smo nedavno spomenuli u nekoliko navrata? Vrijednost parametra t zapravo nije bitna, bitno je jedino da je on konstanta. Ta konstanta se svakako gubi u asimptotskoj notaciji složenosti algoritama koju ćemo sada definirati.

Definicija 3.6. Neka su $g(x)$ i $f(x)$ funkcije. Kažemo da je $g(x) = \mathcal{O}(f(x))$ ako postoje pozitivne konstante $c, N \in \mathbb{R}$, takve da za svaki $n \geq N$ vrijedi

$$g(n) \leq c \cdot f(n).$$

Recimo da postoji konstanta $A \geq 0$, neovisna o veličini podatka koji prosljeđujemo funkciji, takva da za svaki ulazni podatak koji je $\mathcal{O}(k)$ bitova dugačak, te da je potrebno $\mathcal{O}(k^A)$ koraka za rješavanje problema. Tada kažemo da je problem rješiv u polinomijalnom vremenu. Za $A = 1$ problem je rješiv u linearnom vremenu, za $A = 2$ problem je rješiv u kvadratnom vremenu. Algoritme koji se izvršavaju u polinomijalnom vremenu smatramo brzim algoritmima.

Nadalje, ako postoji konstanta $c > 0$ takva da za ulaz od $\mathcal{O}(k)$ bitova i algoritam koji rješava problem u $\mathcal{O}(e^{ck})$ koraka, tada je problem rješiv u eksponencijalnom vremenu. Algoritme s eksponencijalnim vremenom izvršavanja smatramo sporim algoritmima.

Nepisano pravilo u kriptografiji je da problem koji je rješiv u polinomijalnom vremenu smatramo "lakim", dok problem koji je rješiv u eksponencijalnom vremenu smatramo "teškim". Treba imati na umu da ova klasifikacija teškog i lakog problema uveliko ovisi o veličini ulazne varijable i veličini konstante.

Primjer. Pogledajmo originalni problem diskretnog logaritma $g^x = h$ u $G = \mathbb{F}_p^*$. Ako je prosti broj p između 2^k i 2^{k+1}, tada zapis od g, h i p ima najviše k bitova, dakle, problem možemo izraziti kao $\mathcal{O}(k)$ bitova. Imajmo na umu da je $\mathcal{O}(k)$ isto što i $\mathcal{O}(\log_2 p)$.

Probamo li riješiti problem diskretnog logaritma metodom pokušaja i promašaja koju smo opisali ranije u ovom poglavlju, tada nam je potrebno $\mathcal{O}(p)$ koraka za rješavanje. Jer je $\mathcal{O}(p) = \mathcal{O}(2^k)$ ovaj algoritam rješava u eksponencijalnom vremenu.

Naravno, postoje bolji načini za rješavanje problema diskretnog logaritma na $\mathbb{F}_p^*$, neki od njih su jako brzi, ali rade samo za neke proste brojeve. Dok su drugi sporiji, ali rade za sve proste brojeve. Na primjer, Pohlig-Hellmanov algoritam govori da ukoliko rastav broja $p-1$ na proste faktore sadrži samo male faktore tada problem diskretnog logaritma možemo brzo riješiti.

Primjer. Promotrimo sada problem diskretnog logaritma u grupi $G = \mathbb{F}_p$ s operacijom zbrajanja. Problem diskretnog logaritma u ovom slučaju svodi se na traženje vrijednosti x za kongruenciju:

$$x \cdot g \equiv h \pmod{p},$$

gdje su g i h dani elementi iz $\mathbb{Z}/p\mathbb{Z}$. Metodom opisanom na 29. strani u literaturi [2] možemo ovu kongruenciju riješiti proširenim Euklidovim algoritmom tako da izračunamo $g^{-1} \pmod{p}$ i vrijednost $x \equiv g^{-1} \cdot h \pmod{p}$. Ovaj postupak zahtjeva $\mathcal{O}(\log p)$ koraka, stoga postoji algoritam koji u linearnom vremenu rješava problem diskretnog logaritma u aditivnoj grupi $\mathbb{F}_p$. To je jako brzi algoritam, stoga problem diskretnog logaritma u aditivnoj grupi $\mathbb{F}_p$ nije dobar kandidat za jednosmjernu funkciju u kriptografiji.

Važno je imati na umu da problem diskretnog logaritma nije jednako težak za rješavanje u svim grupama. Dakle, problem diskretnog logaritma u aditivnoj grupi $\mathbb{F}_p$ rješava se u linearnom vremenu, dok je najbolji algoritam za grupu $\mathbb{F}_p^*$ sa množenjem u subeksponencijalnom vremenu.

Definicija 3.7. Neka je $T(x)$ vrijeme izvršavanja algoritma s ulazom x. Kažemo da se taj algoritam izvršava u subeksponencijalnom vremenu, ako za svaku bazu $b > 0$ vrijedi:

$$T(x) < b^x.$$

3.6 Algoritam sudara za problem diskretnog logaritma

Sada ćemo opisati algoritam za rješavanje problema diskretnog logaritma koji je konstruirao Shanks. Ovaj algoritam radi na proizvoljnoj grupi, ne samo $\mathbb{F}_p^*$, i dokaz da radi za bilo koju grupu nije težak, stoga ćemo ga provesti. Za početak prisjetimo se koja je složenost *brute-force* algoritma za rješavanje problema diskretnog logaritma.

Propozicija 2. Neka je G grupa, te neka je $g \in G$ element reda N. To znači da je $g^N = e$, gdje je N najmanji prirodan broj za koji ovo svojstvo vrijedi. Tada problem diskretnog logaritma:

$$g^x = h$$

možemo riješiti u $\mathcal{O}(N)$ koraka koristeći $\mathcal{O}(1)$ memorije, gdje se svaki korak sastoji od množenja elementom g.

Dokaz. Jednostavno izračunamo $g, g^2, g^3, \ldots$, gdje je svaki element u tom nizu dobiven na način da njegovog prethodnika pomnožimo sa g, stoga u svakom trenutku u memoriji moramo pohraniti samo 2 elementa. Ako rješenje problema $g^x = h$ postoji, tada će se vrijednost h pojaviti prije nego dođemo do g^N. □

Primjedba 9. U grupi $\mathbb{F}_p^*$, svako računanje $g^x \pmod p$ zahtjeva $\mathcal{O}((\log p)^k)$ računskih operacija. Tada je ukupan broj računskih koraka to jest vrijeme izvršavanja $\mathcal{O}(N(\log p)^k)$. U praksi veličina $\mathcal{O}((\log p)^k)$ je zanemarivo mala, pa ćemo složenost izraziti kao $\mathcal{O}(N)$.

Temeljna ideja Shanksovog algoritma je generiranje dviju listi, te traženje elementa koji se nalazi u obje liste.

Propozicija 3. Shanksov Babystep-Giantstep algoritam. Neka je G grupa te neka je $g \in G$ element reda $N \geq 2$. Sljedeći algoritam rješava problem diskretnog algoritma $g^x = h$ u $\mathcal{O}(\sqrt{N} \cdot \log N)$ koraka, koristeći $\mathcal{O}(\sqrt{N})$ memorije.

1. Neka je $n = 1 + \lfloor \sqrt{N} \rfloor$, posebno, $n > \sqrt{N}$
2. Generirajmo dvije liste:
$$\text{Lista 1: } e, g, g^2, g^3, \ldots, g^n,$$
$$\text{Lista 2: } h, h \cdot g^{-n}, h \cdot g^{-2n}, h \cdot g^{-3n}, \ldots, h \cdot g^{-n^2}.$$
3. Naći zajednički element prethodnih listi. Neka je to na primjer $g^i = hg^{-jn}$.
4. Tada je $x = i + jn$ rješenje problema $g^x = h$.

Dokaz. Za početak uočimo sljedeće: Dok generiramo listu 2, počinjemo s računanjem vrijednosti $u = g^{-n}$ te nakon toga računamo $h, h \cdot u, h \cdot u^2, \ldots, h \cdot u^n$. Stoga je za stvaranje tih dviju lista potrebno je otprilike $2n$ množenja. Nadalje, pretpostavimo da postoji zajednički element navedenih listi. Tada ga možemo pronaći u otprilike $n \log(n)$ koraka koristeći standardne algoritme za sortiranje i pretraživanje što znači da je za 3. korak potrebno $\mathcal{O}(n \log(n))$ koraka. Dakle, ukupno vrijeme izvršavanje ovog algoritma je $\mathcal{O}(n \log(n)) = \mathcal{O}(\sqrt{N} \log N)$. U posljednjem koraku iskoristili smo činjenicu da je $n \approx \sqrt{N}$, stoga vrijedi

$$n \log n \approx \sqrt{N} \log \sqrt{N} = \frac{1}{2} \sqrt{N} \log N.$$

Uočimo da su liste iz koraka 2 duljine n, te zauzimaju $\mathcal{O}(\sqrt{N})$ memorije.

Kako bismo pokazali da algoritam radi, moramo pokazati da Lista 1 i Lista 2 uvijek imaju zajednički element. U tu svrhu, neka je x rješenje problema $g^x = h$, te zapišimo x u obliku

$$x = nq + r \quad \text{gdje je } 0 \leq r < n.$$

Znamo da je $1 \leq x < N$, dakle

$$q = \frac{x - r}{n} < \frac{N}{n} < n \quad \text{budući da je } n > \sqrt{N}.$$

Zato možemo zapisati $g^x = h$ kao

$$g^r = h \cdot g^{-qn} \quad \text{gdje su } 0 \leq r < n \text{ i } 0 \leq q < n.$$

Dakle, g^r nalazi se u Listi 1 i $h \cdot g^{-qn}$ nalazi se u Listi 2, a to znači da liste imaju zajednički element. □

Primjer. Pokažimo kako Shanksov Babystep-Giantstep algoritam funkcionira na primjeru. Neka je dan problem diskretnog logaritma:

$$g^x = h \quad \text{na} \quad \mathbb{F}_p^* \quad \text{gdje su} \quad g = 9704, \quad h = 13896, \quad \text{i} \quad p = 17389.$$

k	g^k	$h \cdot u^k$	k	g^k	$h \cdot u^k$	k	g^k	$h \cdot u^k$	k	g^k	$h \cdot u^k$
1	9704	347	9	15774	16564	17	10137	10230	25	4970	12260
2	6181	13357	10	12918	11741	18	17264	3957	26	9183	6578
3	5763	12423	11	16360	16367	19	4230	9195	27	10596	7705
4	1128	13153	12	13259	7315	20	9880	13628	28	2427	1425
5	8431	7928	13	4125	2549	21	9963	10126	29	6902	6594
6	16568	1139	14	16911	10221	22	15501	5416	30	11969	12831
7	**14567**	6259	15	4351	16289	23	6854	13640	31	6045	4754
8	2987	12013	16	1612	4062	24	15680	5276	32	7583	**14567**

Slika 7: Babystep-Giantstep za rješavanje $9704^x \equiv 13896 \pmod{17389}$

Broj 9704 je reda 1242 u $\mathbb{F}^*_{17389}$. Neka je $n = \lfloor\sqrt{1242}\rfloor + 1 = 36$ i $u = g^{-n} = 9704^{-36} = 2494$. Tablica sa slike 7 pokazuje vrijednosti od g^k i $h \cdot u^k$ za $k = 1, 2, \ldots$ pa u tablici možemo pronaći koliziju

$$9704^7 = 14567 = 13896 \cdot 2494^{32} \quad \text{u } \mathbb{F}_{17389}.$$

Koristeći činjenicu da je $2494 = 9704^{-36}$, izračunamo

$$13896 = 9704^7 \cdot 2494^{-32} = 9704^7 \cdot (9704^{36})^{32} = 9704^{1159} \quad \text{u } \mathbb{F}_{17389}.$$

Dakle, $x = 1159$ rješava problem $9704^x = 13896$ u $\mathbb{F}_{17389}$.

4 Zaključak

Protokol za razmjenu ključa, koji su Diffie i Hellman objavili, omogućio je razvoj jedne sasvim nove grane kriptografije - kriptografije s javnim ključem. Uz današnju tehnologiju, ukoliko se odaberu dovoljno veliki brojevi, nije moguće izvesti uspješan napad na Diffie-Hellmanov protokol jer problem diskrentog logaritma koji se krije u pozadini protokola ima eksponencijalnu vremensku složenost. Nažalost veliki nedostatak Diffie-Hellmanovog protokola je mogućnost napada s čovjekom u sredini. Stoga bi trebalo uvesti autentikaciju sudionika komunikacije kako bi se ogradili od takve vrste napada. Iako se ovaj protokol može činiti jednostavan, on ima široku primjenu u zaštiti komunikacije putem otvorenih komunikacijskih kanala kao što je na primjer Internet.

Iako se godinama i dalje provode istraživanja na temu sigurne razmjene ključeva, Diffie-Hellmanov protokol i dalje se koristi kao jedino rješenje za razmjenu tajnih ključeva u kriptografskim sustavima. Možemo reći da je Diffie-Hellmanova razmjena ključeva jedna od najznačajnijih primjena teorije brojeva vezana za sigurnost kriptosustava.

Literatura

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

[2] J. Hoffstein, J. Pipher, J. Silverman, *An Introduction to Mathematical Cryptography*, Springer, 2008.

[3] G. Kessler, *An Overview of Cryptography*, `https://www.garykessler.net/library/crypto.html`, zadnje posjećeno 22.9.2020.

[4] *Nacionalni CERT*, `https://www.cis.hr/www.edicija/LinkedDocuments/NCERT-PUBDOC-2009-12-284.pdf`, zadnje posjećeno 21.10.2020.

Sažetak

Cilj ovoga rada je upoznavanje s osnovama kriptografskih sustava s javnim ključem. Od povijesne perspektive pa do prvih formalnih definicija koje su postavile temelje nove grane znanosti.

Kroz primjere vidjeti ćemo takvu komunikaciju iz tri različite perspektive: pošiljaoca, primatelja i "protivnika".

Početi ćemo sa Diffie-Hellmanovim protokolom za razmjenu ključeva, jer je ono postavilo temelje moderne kriptografije. Zatim ćemo objasniti prvi službeni kriptosustav pod nazivom ElGamalov kriptosustav.

Nakon toga baviti ćemo se analizom kompleksnosti matematičkih problema koji se kriju u pozadini obrađenih kriptosustava. Dodatno spomenuti ćemo kako odabrati javne ključeve na način da sustav bude što sigurniji.

Ključne riječi Diffie-Hellmanov protokol, kriptografija, problem diskretnog logaritma, kongruencije

Summary

The aim of this paper is getting to know the basics of public key cryptography. We will start with the history of it, and work our way through the formal definitions that laid the foundations of this new branch of science

Through examples we will see communication from three different perspectives: sender, receiver and the "enemy".

We will start off with the Diffie-Hellman key exchange, because it is one of the most important principles in public key cryptography. Later on, we will learn about the first formal cryptosystem called ElGamalov cryptosystem.

On the end, we will analyse the time complexity of the underlaying math problems of mentioned cryptosystems. Additionaly, we will see how public keys should be picked to make the cryptosystem as safe as possible.

Keywords Diffie-Hellman protocol, criptography, the discrete logarithm problem, congruences

Životopis

Zovem se Iwan Josipović i rođen sam 15. lipnja 1996. godine u Nijmegenu, Nizozemska. Pohađao sam Osnovnu školu Vladimira Nazora u Čepinu i III. gimnaziju u Osijeku. 2015. godine upisao sam preddiplomski studij Matematike na Odjelu za matematiku u Osijeku koji sam završio 2018. godine s temom završnog rada „Primjena interpolacije u digitalnoj obradi slika" pod mentorstvom izv.prof.dr.sc. Domagoja Matijevića. Svoje obrazovanje nastavio sam na Odjelu za matematiku u Osijeku te sam 2018. godine upisao diplomski studij, smjer: Matematika i računarstvo.
Tijekom obrazovanja sudjelovao sam na IEEE grupnim natjecanjima u programiranju. Osim toga, postao sam stipendist tvrtke Atos Convergence Creators, gdje sam trenutno zaposlen.