Pregled post-kvantnih kriptografskih sustava

Čurla, Kristian

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO–MATEMATIČKI FAKULTET
MATEMATIČKI ODSJEK

Kristian Čurla

PREGLED POST-KVANTNIH KRIPTOGRAFSKIH SUSTAVA

Diplomski rad

Voditelj rada:
Izv. prof. dr. sc. Matija Kazalicki

Zagreb, Srpanj 2019

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

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

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Diplomski rad napravljen je u sklopu aktivnosti Projekta KK.01.1.1.01.0004 - Znanstveni centar izvrsnosti za kvantne i kompleksne sustave te reprezentacije Liejevih algebri.

Sadržaj

Poglavlje 1

Uvod

1.1 Kriptografija i sigurnost

Još od ranih dana ljudi su imali potrebu za tajnosti. Ova potreba je danas jača nego ikada s obzirom na to koliko se prijenos informacija prebacio u digitalni svijet te koliko su te informacije bitne. Komunikacija, internet bankarstvo, elektronsko glasanje, razmjena osjetljivih podataka su samo neki od procesa na koje smo se počeli jako oslanjati i uzimati ih zdravo za gotovo, a koji bi bili jednostavno nemogući bez nekog sistema koji garantira sigurnost podataka u pitanju.

Kako izgleda takav sustav?

Kriptosustav

Imamo poruku p koju šaljemo. Procesom enkripcije pretvaramo našu poruku p u šifrat s. Iz šifrata s je nemoguće rekonstruirati poruku p osim ako znamo proces dekripcije. Procesom dekripcije pretvaramo šifrat s natrag u poruku p.

Definicija 1 (Kriptosustav)**.** *Kriptosustav je uređena petorka* $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *gdje su:*

- $\mathcal{P}$ *skup mogućih poruka*
- $\mathcal{C}$ *skup svih mogućih šifrata*
- $\mathcal{K}$ *skup svih ključeva*
- $\mathcal{E} = \{E_k : k \in \mathcal{K}\}$ *skup funkcija* $E_k : \mathcal{P} \to \mathcal{C}$*. Njene elemente zovemo "enkripcijske funkcije"*
- $\mathcal{D} = \{D_k : k \in \mathcal{K}\}$ *skup funkcija* $D_k : \mathcal{C} \to \mathcal{P}$*. Njene elemente zovemo "dekripcijske funkcije"*

Koji je točno svrha kriptosustava, tj. koje kvalitete želimo da naš kriptosustav posjeduje? Ovdje nabrajamo njegove fundementalne ciljeve:

- Tajnovitost - osiguranje da nitko ne može pročitati poruku osim onog kome je poruka namjenjena, bez obzira na to koliko je sam kanal komunikacije siguran.
- Integritet - onemogučavanje izmjene poslane poruke prije nego li je primljena
- Autentičnost - mogučnost dokazivanja identiteta
- Neosporavanje - mehanizam za dokazivanje da je pošiljatelj zaista poslao poruku

Ovo su idealni ciljevi no je li svaki kriptosustav koji ih zadovoljava dobar? Nažalost ne, neki kriptosustavi jednostavno nisu praktični za realnu implementaciju (na primjer, ključevi koji zahtjevaju previše memorije ili enkripcijska shema koja je prespora). Mi ćemo se u ovom radu fokusirati na sigurnost, no spomenuti ćemo i ostale kvalitete kriptosustava.

Sigurnost

Kako izvesti to da je nemoguće nekome tko je presreo poruku da ju pročita? Nažalost, garantirati da je nešto apsolutno sigurno je jednostavno neizvedivo, netko tko ima pristup efektivno beskonačnim komputacijskim resursima će moći probiti i najsofisticiraniji kriptosustav.

Ono što možemo napraviti je učiniti taj proces što skupljim možemo. Ako promatramo naš proces enkripcije kao funkciju, ono što želimo je učiniti tu funkciju što "teže" invertibilnom. Pronalazak takvih funkcija je ključna ideja kriptografije.

Što je to općenito "teško" za računala? U Teoriji Izračunljivosti problemi se dijele na dvije velike klase: P i NP. Za probleme iz klase P znamo da postoje algoritmi koji ih rješavaju u polinomijalnom vremenu, dok za probleme iz klase NP ne znamo postoje li polinomijalni algoritmi. Probleme iz klase P smatramo "lakima" dok su problemi iz klase NP "teški". Nekako je prirodno pretpostaviti da onda samo odaberemo neki problem iz klase NP, konstruiramo funkciju koja ga odlučuje i oko nje tvorimo kriptosustav. Dapače, svi dokazi sigurnosti kriptosustava se svode na dokazivanje da je neki problem na kojem baziramo kriptosustav NP-težak. Bitno je istaknuti da ovaj način dokazivanja nikako ne dokazuje ne postojanje efikasnog algoritma, već se samo "naslonimo" na težinu nekog drugog problema kojeg smatramo teškim te pokažemo da za postojanje efikasnog algoritma za prvi problem mora postojati efikasan algoritam za drugi problem.

Recimo da smo našli adekvatan problem i dokazali da je NP-težak. Nažalost, svijet nije tako jednostavan i posao ovdje nije gotov. Naime, da bi kriptosustav bio

iskoristiv funkciju treba biti lako izračunati no teško invertirati i mnogi NP teški problemi su teški za izračunati u oba smjera. Funkcije koje posjeduju svojstvo koje opisujemo zovemo jednosmjernim funkcijama.

Definicija 2 (Jednosmjerna funkcija)**.** *Funkcija $f : \{0,1\}^* \to \{0,1\}^*$ je jednosmjerna ako je f izračunljiva u polinomijalnom vremenu ali bilo koji polinomijalni randomizirani algoritam F koji pokuša invertirati f uspjeva sa zanemarivom vjerojatnošću.*

Je li nam dovoljna jednosmjerna funkcija? Nažalost, često nije.

Naime, jednosmjerne funkcije nam omogučavaju da enkriptiramo poruku tako da ju je napadačima teško dekriptirati. Mi također želimo da osoba koja treba dobiti poruku ju lako može dekriptirati tj. invertirati funkciju dekripcije, a jednosmjerne funkcije to nužno ne omogućavaju. Trebamo pronaći način da omogućimo laganu dekripciju samo specifičnim akterima. Upravo za tu svrhu koristimo ključeve.

Ključ je neka informacija koja kada ju posjedujemo problem dekripcije poruke postaje za nas lagan. Postoji podskup jednosmjernih funkcija koje imaju svojstvo da postoji neka informacija koja ih čini lako invertibilnima. Funkcije sa ovim svojstvom zovu se Trapdoor Funkcije.

Definicija 3 (Trapdoor funkcija)**.** *Za kolekciju funkcija $\{f_k : D_k \to R_k\}\,(k \in K)$, gdje su K, D_k, R_k podskupovi od $\{0,1\}^*$, kažemo da je Trapdoor ako vrijedi:*

- *Postoji polinomijalni probabilistički (PP) algoritam Gen takav da:* $\mathrm{Gen}\,(1^n) = (k, t_k)$ *gdje je* $\mathrm{k} \in \mathrm{K} \cap \{0,1\}^n$ *i* $t_k \in \{0,1\}^*$ *koji zadovoljava* $|t_k| < p(n)$ *za neki polinom p. Svaki t_k se naziva trapdoor-om za pridruženi k, te ga je lako uzorkovati.*
- *Za dani input $k \in K$, postoji neki PP algoritam koji izbacuje $x \in D_k$. Drugim riječima, svaki D_k se lako uzorkuje.*
- *Za svaki $k \in K$, postoji neki PP algoritam koji efikasno izračunava f_k.*
- *Za svaki $k \in K$, postoji neki PP algoritam A takav da za svaki $x \in D_k$, postoji $y = A\,(k, f_k(x), t_k)$ takav da $f_k(y) = f_k(x)$. Drugim riječima, ako imamo trapdoor, funkciju je lagano invertirati.*
- *Za svaki $k \in K$, bez pristupa trapdooru t_k, svakom PP algoritmu je vjerojatnost da uspješno invertira f_k zanemariva.*

Pitanje koje je ostalo u zraku je kako predati ključ osobi koja ga treba imati? Postoje dva pristupa: simetrični ključevi i asimetrični ključevi. Ako koristimo simetrične ključeve princip djeljenja klučeva je jako osjetljiv te se stoga taj pristup slabo koristi u današnje vrijeme, iako je u pravilu jednostavniji. Drugi pristup, koji se

alternativnim imenom zove javni ključevi, je onaj koji je pretežno u upotrebi danas. Njegova praktičnost se skriva u tome što ključ koji se dijeli nije osjetljive prirode pa ne moramo paziti na sigurnost njegove distribucije, što je dosta "pipljava" tema.

Kriptosustav s javnim ključem funkcionira na sljedeći način: Ante želi Borisu poslati poruku. Boris sam sebi generira ključ koji zovemo privatni ključ. Zatim on generira javni ključ koji je zapravo recept po kojem će Ante zakodirati poruku tako da ključ koji je potreban da se ona dekodira je upravo taj koji je Boris generirao pri početku procesa. Ante kodira poruku te ju Boris dekriptira koristeći svoj privatni ključ.

Kao što primjećujemo, jedini ključ koji se šalje je javni ključ i on ne sadrži nikakve osjetljive informacije te upravo ta činjenica čini ovaj sustav tako praktičnim za upotrebu. Pošto je pretpostavka da napadač ima javni ključ ono na što treba paziti je da bude jako teško otkriti koji je privatni ključ pridružen tom konkretnom javnom ključu.

Treba paziti da ključ koji šaljemo, dakle javni, zauzima malo memorije kako bi komunikacija bila efikasna. Zna se dogoditi slučaj da je veličina ključa potrebna da bi kriptosustav bio dovoljno siguran bude prevelika i da to bude jedini razlog zašto odbijemo neki kriptosustav.

Najpoznatiji i najšire korišteni kriptosustav s javnim ključem zove se **RSA** i bazira se na činjenici da je teško rastaviti ogromne brojeve na velike proste faktore. Ima jako male javne ključeve (od samo 2048 bitova), brzu i sigurnu enkripciju baziranu na problemu koji je jako izučavan.

Za kraj ove sekcije, istaknimo da je cijela priča oko kriptografije fokusirana na probleme koji su teško rješivi računalima. Makar to nismo eksplicitno spomenuli do sada smo mislili na računala klasične prirode kakve većina ljudi danas ima u svojim domovima te kada smo pričali o teškim problemima podrazumjeva se da su to teški problemi u klasičnom smislu, dakle teški za klasična računala. 1959. godine Richard Feynman komentira kako bi se kvantna mehanika mogla koristiti pri izračunavanju, a Yuri Manin 1980. godine motiviran time predlaže alternativnu prirodu računala pod imenom Kvantno računalo. Motivacija za ova računala je mogučnost za izvršavanje operacija koje klasična računala čine jako sporo. Kako se sigurnost kriptosustava bazira na tome da su neki problemi teško izračunljivi (dakle sporo) ovo znači da su potencijalno naši dosadašnji kriptosustavi u opasnosti da budu lako probijeni od strane kvantnih računala. No kako su u to vrijeme kvantna računala bila samo potencijalna te je njihova upotreba bila čisto predviđana, nije bilo razloga za paniku.

1994. godine Peter Shor pronalazi kvantni algoritam (dakle algoritam koji samo mogu provesti kvantna računala) koji efikasno pronalazi velike proste faktore velikih brojeva, te efektivno probija RSA.

2011. godine dokazano je da je moguće izgraditi kvantno računalo bazirano na

Von Neumann-ovoj arhitekturi računala.

Razloga za paniku, čini se, ipak ima.

1.2 Kvantno računanje

Što su to točno kvantna računala i po ćemu se ona razlikuju od klasičnih?

Prisjetimo se, klasična računala operiraju na *bitovima*, apstraktnim komadima informacija koji mogu biti 0 ili 1. Takva stanja fizički stvaramo unutar računala *tranzistorima* koji ili propuštaju elektrone i tako stvaraju stanje 1, ili ih ne propuštaju stvarajući stanje 0. Tranzistore spajamo u *logičke sklopove* koji reprezentiraju osnovne logičke operacije poput AND i XOR. Sa velikim nizom logičkih sklopova definiriamo što to sve naše računalo može. Na kraju, mi našem računalu dajemo input od n bitova i ono nam daje output od k bitova koji ili direktno koristimo ili dajemo kao input nekom sljedećem nizu logičkih sklopova.

Kvantna računala razlikuju se upravo u ovim fundementalnim gradivnim blokovima.

Umijesto bitova operiraju nad *qubitima* koji nisu u stanju 0 ili 1 već u superpoziciji ta dva stanja te prilikom procesa mjerenja kolabiraju u jedno od ta dva stanja. Da ne zalazimo duboko u kvantnu mehaniku dajemo matematičku definiciju superpozicije qubita.

Definicija 4 (Stanje qubit-a)**.** *Označimo s* $|0\rangle$ *stanje qubita koje prilikom mjerenja uvijek daje stanje 0 te s* $|1\rangle$ *stanje koje uvijek daje 1 (standardna ket notacija). Ako ih zapišemo kao vektor dobijemo sljedeću notaciju* $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ *i* $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$*. Ova dva vektora su* ***baza Hilbertovog prostora*** *za qubite i njima možemo opisati sva njegova stanja. Općenito stanje qubita se definira kao*

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Gdje su α *i* β *vjerojatnosne amplitude izražene kao kompleksni brojevi za koje vrijedi*

$$|\alpha|^2 + |\beta|^2 = 1$$

Veoma je bitno sada spomenuti sljedeću činjenicu: ako nam je dan proizvoljan qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ **ne** postoji nikakav način da eksplicitno dobijemo α i β. Dapače, ovo je jedna od fundementalnih činjenica kvantne mehanike općenito.

No ipak možemo izvući neke informacije iz qubita. Proces kojim to činimo se zove mjerenje u računskoj bazi.

Definicija 5 (Mjerenje u računskoj bazi)**.** *Mjerenje u komputacijskoj bazi je probabilistička funkcija koja za dani input qubit $|\psi\rangle$ daje vrijednost 0 s vjerojatnošću $|\alpha|^2$ i 1 s vjerojatnošću $|\beta|^2$ te pritom nepovratno uništava stanje superpozicije qubita $|\psi\rangle$.*

Sada se lako vidi otkud dolazi zahtjev $|\alpha|^2 + |\beta|^2 = 1$.

Za razliku od transistora, fizička inkarnacija qubita može biti uglavnom bilo što što posjeduje kvantno ponašanje, no kako je stanje superpozicije jako teško postići i održati, te postaje sve teže i teže što je više qubita u istom sustavu sama fizička konstrukcija je izrazito kompleksna.

Način na koji qubiti mogu komunicirati zove se kvantno zapetljanje.

Umjesto logičkih sklopova imamo kvantna vrata koja utječu na superpoziciju qubita ili niza qubita koji se promatraju kao jedno kvantno stanje.

Definicija 6 (Kvantna logička vrata)**.** *Kvantna logička vrata koja operiraja nad n quibita je $2n \times 2n$ unitarna matrica U. Njeno djelovanje opisujemo kao $U|a_1\rangle \otimes \cdots \otimes |a_n\rangle$.*

Isto kao i u klasičnim računalima, večina najkorištenijih kvantnih logičkih vrata su ona koja kao input imaju mali broj qubita (pretežno jedan, dva i tri). Također, zgodno je primjetiti da je definicija kvantnih logičkih vrata kao unitarnih matrica zapravo nužnost da se održi zahtjev na amplitude qubita.

Primjer 1 (Kvantna logička vrata NOT)

Kao što i samo ime sugerira, ova vrata su generalizacija klasičnog logičkog sklopa NOT. Dakle sigurno želimo $|0\rangle \to |1\rangle$ i $|1\rangle \to |0\rangle$. Veoma prirodno, linearno ga proširujemo i opće ponašanje vrata opisujemo kao

$$\alpha|0\rangle + \beta|1\rangle \to \alpha|1\rangle + \beta|0\rangle$$

Tj u matričnom zapisu $N = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

Nakon ovog jako jednostavnog primjera promotrimo malo kompleksniji primjer.

Primjer 2 (Hadamardova vrata)

Ponovno, prvo definirajmo ponašanje vrata na baznim stanjima.

$$|0\rangle \to \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$|1\rangle \to \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Ako ovo raspišemo po liearnosti dobijemo matrični zapis koji je

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

. Uz raspisivanje i ponešto linearnog računa lako se uvjeriti da je H involuntarna matrica, baš kao i NOT. Ono što je bitno naglasiti i to će nam biti od važnosti nešto kasnije, je činjenica da je ovo zapravo verzija kvantne Fourierove transformacije na jednom qubitu.

Sljedeći primjer je prvi primjer kvantnih vrata koji operira na vše od jednog qubita pa treba definirati kako izgleda stanje od više qubita. Veoma prirodno proširujemo vektorski prikaz qubita da prikažemo stanje od dva qubita i to tako da ga definiramo kao

$$|ab\rangle = |a\rangle \otimes |b\rangle$$

Gdje $\otimes$ predstavlja uobičajeni tenzorski produkt dva vektora.

Primjer 3 (Kontrolirana vrata)

Kontrolirana vrata su verzija kvantnih vrata koja uzimaju kao ulaz više od jednog qubita te vrše operaciju samo ako je kontrolni qubit u stanju 1.

Uzmimo za primjer kontrolirana NOT vrata:

Za kontrolni bit koristiti ćemo prvi qubit a na drugom vršimo operaciju NOT. Tada imamo

$$|00\rangle \mapsto |00\rangle \qquad |01\rangle \mapsto |01\rangle$$

$$|10\rangle \mapsto |11\rangle \qquad |11\rangle \mapsto |10\rangle$$

Kvantno zapetljanje

Objasnili smo da se generalno stanje od više qubita predstavlja tenzorskim produktom, no možemo li tako opisati sva stanja od dva ili više qubita? Odgovor je ne, oni mogu biti u jednom posebnom stanju koje zovemo *kvantno zapetljanje* i takvo stanje se ne može opisati tenzorskim produktom. Promotrimo najjednostavniji primjer takvog stanja na dva qubita pod nazivom *zapetljano Bell stanje* koje ćemo označiti s B i definirati na sljedeći način:

$$|B\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

Dakle, B može samo biti u stanju $|00\rangle$ i $|11\rangle$ i to se vjerojatnošću 1/2 za oba, a nikada nije u stanju $|01\rangle$ niti $|10\rangle$ tj. u tim stanjima je s vjerojatnošću 0.

Pokušajmo ovo stanje opisati pomoću tenzorskog produkta dva qubita:

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}} = \alpha \cdot \gamma \cdot |00\rangle + \alpha \cdot \delta \cdot |01\rangle + \beta \cdot \gamma \cdot |10\rangle + \beta \cdot \delta \cdot |11\rangle$$

Iz čega slijedi da je nužno

$$\alpha = 0 \text{ ili } \delta = 0$$

No to je u kontradikcija s time da je $\alpha\gamma = 1/2$ i da je $\beta\delta = 1/2$. Iz čega zaključujemo da je nemoguće opisati ovo stanje preko prirodnog tenzorskog produkta.

Intuitivna motivacija za ime "zapetljani" dolazi od činjenice da je sudbina naša dva qubita povezana. Ako prvog izmjerimo i dobijemo 0 tada drugi nužno mora isto biti u stanju 0, te analogno vrijedi i obrnuto.

Ovo stanje možemo smatrati "srži" kvantnog računanja jer je konstatirano (iako još nije rigorozno dokazano) da bilo koji kvantni algoritam koji ne koristi zapetljano stanje može efikasno biti izveden i na klasičnim računalima.

Za kraj valja još jednom naglasiti da je kvantno računanje probabilističke prirode, tj. da izlazno stanje ima više stanja sa svojim vjerojatnosnim težinama i veoma su rijetki kvantni algoritmi koji stvaraju izlaze u kojima samo jedno stanje ima težinu 1 a ostala 0.

No, u praksi nam to nije problem dokle god algoritam sa dovoljno velikom vjerojatnošću daje pravi izlaz, jednostavno ćemo ponoviti računanje dovoljno puta da budemo sigurni.

Još je otvoreno pitanje (tzv. *quantum supremacy*) jesu li kvantna računala striktno superiornija od klasičnih ili su samo adekvatnija za neke zadatke. I za njih vrijede neka fundementalne konstatacije iz teorije izračunljivosti poput Church–Turing teze koja kaže da svaka funkcija nad prirodnim brojevima se može efikasno izračunati ako i samo ako ju može izračunati neki Turingov stroj. Dapače, Turingov stroj može simulirati rad kvantnog računala. Stoga kvantna računala ne proširuju raspon toga što računala mogu. Na primjer, kvantna računala ne mogu odlučiti halting problem. Ali što se same efikasnosti tiče, postoje algoritmi koje kvantna računala izračunavaju daleko efikasnije od klasičnih računala.

Bez sumnje najbitniji takav algoritam je Shorov algoritam.

1.3 Shorov algoritam

Rastavljanje broja na njegove proste faktore je proces koji je jako težak za klasična računala i pretežno se svodi na neku formu pogađanja što vodi u eksponencijalna

vremena. Uspjeli smo pronaći neka sofisticirana ubrzanja koja to dovode u subeksponentne vode (dakle, bolje od eksponencijalnog, gore od polinomnog) no ništa u polinomijalnom vremenu.

Shorov algoritam je kvantni algoritam klase **BQP** (*eng*, **B**ounded error **Q**uantum **P**olynomial time). To znači da je njegovo vrijeme izvršavanja polinomijalno, te da griješi s vjerojatnošću najviše 1/3. Ovo je zapravo kvantni analogon klasi **PP** (*eng*, **P**robabilistic **P**olynomial time) koja se odnosi na probabilističke algoritme u polinomijalnom vremenu što je nadskup klase **P** (je li pravi nadskup je otvoreno pitanje kako u novije vrijeme otkrivamo determinističke načine za izvesti algoritme koje smo smatrali da trebamo rješavati probabilistički). Ono što je nama bitno je da je Shorov algoritam polinomijalan te je samim time *efikasan*.

Predstavljanje problema

Neka je $N \in \mathbb{N}$ veliki složen broj. Treba pronaći broj koji ga djeli. Najzanimljiviji primjer ovoga je kada su djelitelja broja N veliki prosti brojevi.

Algoritam

Primjetimo da ne moramo baš pronaći broj koji ga djeli, dovoljno je pronaći broj g za koji vrijedi $gcd(g, N) > 1$ jer ako imamo taj broj možemo Euklidovim algoritmom doći do djelitelja broja N (sam Euklidov algoritam je polinomijalan).

Kao što smo spomenuli, općenito ne možemo bolje nego pogađati. No, ako nismo pogodili, možemo li taj pokušaj nekako poboljšati i sa određenom sigurnošću tvrditi da je taj poboljšani pokušaj onaj koji tražimo? Odgovor je da, i to je prva ideja Shorovog algoritma.

Naime, pretpostavimo da $gcd(g, N) = 1$ (između ostalog, ako nije, već smo gotovi). U tom slučaju, prirodni brojevi manji od N koji nemaju zajedničkog djelitelja s N formiraju konačnu multiplikativnu Abelovu grupu modulo N, nazovimo ju G, te je $g \in G$. To znači (kako je grupa konačna) da $\exists p \in \mathbb{N}$ takav da

$$g^p = m \cdot N + 1, \text{za neki m} \in \mathbb{N}$$

$$g^p - 1 = m \cdot N$$

$$\left(g^{p/2} + 1\right) \cdot \left(g^{p/2} - 1\right) = m \cdot N$$

Ova zadnja jednadžba izgleda divno jer ukazuje da možda $\left(g^{p/2} + 1\right)$ ili $\left(g^{p/2} - 1\right)$ djele N ili imaju zajedničkog djelitelja s N, što smo komentirali da nam je jednako dobro. $g^{p/2} \pm 1$ je upravo taj poboljšani pokušaj koji smo najavili. Nažalost, ovaj nezgodni "možda" označava par nepovoljnih ishoda.

- Jedan od faktora s lijeve strane može biti umnožak od N.
- p može biti neparan broj, pa onda $p/2$ nije cijeli broj i naš poboljšani pokušaj je neupotrebljiv.

Još gore, od ovih problema nam nema spasa. No, ti problemi ipak ispadaju nebitni jer u 37.5% slučajeva, neće se dogoditi niti jedno od toga dvoje. To znači da ako ponovimo algoritam 10 puta, šansa da smo zaista dobili bolji pokušaj je preko 99%. Ponavljanje polinomijalnog algoritma fiksni broj puta i dalje ga čini polinomijalnim.

Preostaje nam ključno pitanje: kako pronaći p? Možemo ga probati pronaći na klasičan način no nećemo se usrećiti jer najbolje što možemo je pogađati ga. Kvantna računala, s druge strane, mogu to izvesti efikasno.

Redukcija problema na pronalazak perioda funkcije

Promotrimo sljedeću dosta jednostavnu konstataciju. Neka je

$$g^x = m \cdot N + r$$

$$g^p = m \cdot N + 1$$

Množenjem ove dvije jednadžbe i supstitucijom velikog umnoška s m_2 dobijemo

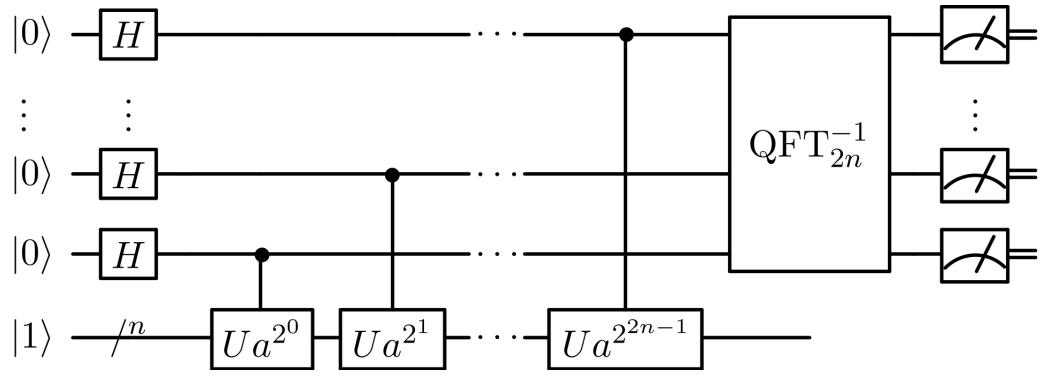
$$g^{x+p} = m_2 \cdot N + r$$

Ovo znači da je p period funkcije f koja x-u pridružava r.

Ono što smo upravo napravili je reducirali problem traženja broja p na traženje perioda neke funkcije f. Probleme ove klase nazivamo *problem skrivene grupe* (**HSP**) i možemo ga smatrati "srcem" Shorovog algoritma. Sve do sada smo mogli izvesti i na klasičnim računalima, ali da efikasno rješimo ovu instancu HSP-a morati ćemo koristiti kvantno računanje.

Izrazito je bitno za naglasiti da je rješenje HSP-a generalnije od same faktorizacije što čini sve kriptosustave koji se mogu reducirati na ovakvu instancu HSP-a u opasnosti, ne samo one koji se baziraju na težini faktorizacije.

Shema kvantnog djela Shorovog algoritma



Slika 1.1: Kvantni sklop Shorovog algoritma

Prva stvar koju moramo napraviti je dovesti ulazne qubite u stanje superpozicije, tj. inicijalizirati ih. To ćemo napraviti primjenom Hadamardovih vrata na sve njih paraleleno. Naglašavamo da ovo ovisi o broju N i broju g, dakle nije univerzalno za sve instance problema.

Sljedeći dio implementacije je izvedba funkcije f kao kvantne transformacije. Tu koristimo tzv. *proces ponavljajućeg kvadriranja* gdje ulazne qubite koristimo kao kontrolne qubite.

Zadnji dio kvantnog sklopa je *inverz kvantne Fourierove tranformacije* realizirane pomoću vrata kontrolirane rotacije i Hadamardovih vrata.

Opišimo sad malo dublje ove korake ali radi jednostavnosti ćemo neke detalje preskočiti, poglavito kod mjerenja:

Prvo, odaberimo q takav da za $Q = 2^q$ vrijedi $N^2 \leq Q < 2N^2$. Inicijaliziramo q nezavisnih qubita kao ulazne registre na

$$\frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} |x\rangle = \left(\frac{1}{\sqrt{2}} \sum_{x_1=0}^{1} |x_1\rangle \right) \otimes \cdots \otimes \left(\frac{1}{\sqrt{2}} \sum_{x_q-0}^{1} |x_q\rangle \right)$$

U drugom koraku konstruiramo funkciju f spomenutu u prošloj sekciji kao kvantnu funkciju i primjenimo ju na naš ulaz od q qubita. Tada dobijemo

$$\frac{1}{\sqrt{Q}} \sum_{x=0}^{Q-1} |x, f(x)\rangle$$

Bitno je primjetiti kako u sumi nemamo tenzorski produkt, već pišemo samo jedno stanje. To proizlazi iz toga što su naših ulaznih q qubita i izlaznih n qubita u stanju

kvantne zapetljanosti i njihova mjerenja utječu jedna na druge. Zato ih moramo pisati kao jedno stanje.

Zadnji korak nam daje sljedeće stanje

$$U_{\mathrm{QFT}}(|x\rangle) = \frac{1}{\sqrt{Q}} \sum_{y=0}^{Q-1} \omega^{xy} |y\rangle$$

gdje je $\omega = e^{\frac{2\pi i}{Q}}$. Ovo stanje možemo malo drugačije zapisati kao

$$\frac{1}{Q} \sum_{z=0}^{N-1} \sum_{y=0}^{Q-1} \left[\sum_{x \in \{0,\ldots,Q-1\}; f(x)=z} \omega^{xy} \right] |y, z\rangle$$

Zadnji dio algoritma je mjerenje. Kao što smo spomenuli, preskačemo konkretne detalje implementacije i komentiramo da je vjerojatnost stanja veća za ona stanja za koja je $\frac{yp}{Q}$ bliže prirodnom broju. Ako je $\frac{yp}{Q}$ blizu prirodnom broju c tada je $\frac{y}{Q}$ blizu $\frac{c}{r}$. Provodimo razvoj u verižni razlomak za $\frac{y}{Q}$ i dobijemo aproksimaciju $\frac{d}{s}$ za koju vrijedi

$$s < N$$

$$\left| \frac{y}{Q} - \frac{d}{s} \right| < \frac{1}{2Q}$$

Iz čega zaključujemo da ako je $\frac{d}{s}$ ireducibilan tada je s veoma vjerojatno period p ili barem njegov djelitelj.

Provjerimo sve kandidate i ako nismo pogodili, algoritam počinje iz početka s novim pokušajem g.

Za kraj, dašak realnosti: Shorov algoritam zahtjeva 5900 qubita da bi uspio pronaći djelitelje brojeva koji se koriste u RSA kriptosustavu, a najveće kvantno računalo koje smo do sada uspjeli osposobiti posjeduje samo 160 qubita.

No rast i razvoj kvante teorije je brz i nepredvidljiv. Tko zna, možda smo samo jednu spoznaju do stabilnog kvantnog računala sa dovoljnim brojem qubita potrebnih za Shora, no jednako tako, možda nikada nećemo uspjeti složiti sistem od toliko qubita jer posao postaje izrazito teži što ih je veći broj u pitanju.

1.4 Što dalje

Postoji nekoliko kriptosustava koji nisu bazirani na težini faktorizacije brojeva niti se mogu svesti na neku instancu HSP-a te time nisu ugroženi od strane Shorovog algoritma.

U drugom poglavlju pričati ćemo o jednom takvom sustav baziranom na rešetkama. Prvo ćemo se upoznati s time što su rešetke, dokazati neka bitnija svojstva te objasniti jedan od kriptosustava koji se bazira na njima pod imenom GGH.

U trećem i zadnjem poglavlju reći ćemo nešto o kriptosustavima baziranim na kodovima. Početi ćemo s motivacijom za postojanje kodova te ćemo kroz primjer proći sva bitnija njihova svojstva. Zatim ćemo opisati McEliece-ov kriptosustav koji se bazira na sposobnosti kodova da korigiraju greške.

Pri završetku će biti jasno da iako ima razloga za paniku, sigurno ima i nade.

Poglavlje 2

Sustavi bazirani na rešetkama

2.1 Uvod u rešetke

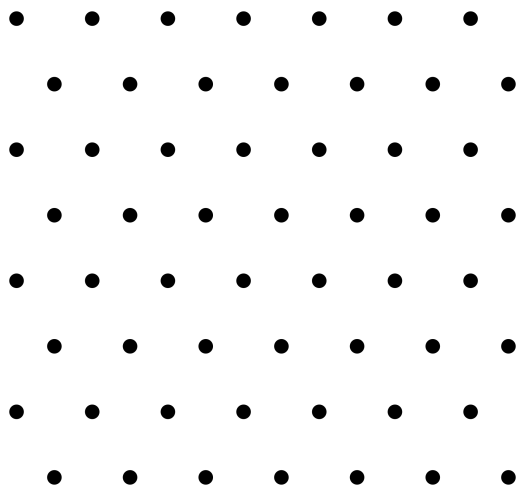
Od svih kriptografskih sustava koji bi mogli odoljevati kvantnim napadima, oni bazirani na rešetkama pokazuju zasada najviše potencijala. Ova konstatacija potkrijepljena je činjenicama da ti sustavi često imaju snažne dokaze sigurnosti, težinu komputacije bazirane na najgore-teškom slučaju (SIS Ajtai 96), relativno jednostavnu implementaciju sustava te intuitivnosti ideja koje su prisutne. Nadalje, dan danas nije pronađen kvantni algoritam koji probija sustave bazirane na rešetkama niti su prenosive već poznate kvantne metode, posebno Schor-ov algoritam.

Što je to rešetka i kako se ona definira? Opisno, zamislimo da imamo n dimenzionalan prostor i u njemu n nezavisnih vektora. Rešetka je skup svih točaka u tom prostoru dobiven cijelobrojnim linearnim kombinacijama spomenutih vektora. Kao što možemo vidjeti na slici 2.1, očito je da samo ime za ovaj objekt dolazi iz njene periodičke rigidne strukture.

Definicija 7. *Rešetka L je skup definiran kao:*

$$L(b_1, ..., b_n) = \left\{ \sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z} \right\}$$

Gdje su $b_1, ..., b_n$ linearno nezavisni vektori iz $\mathbb{R}^n$.

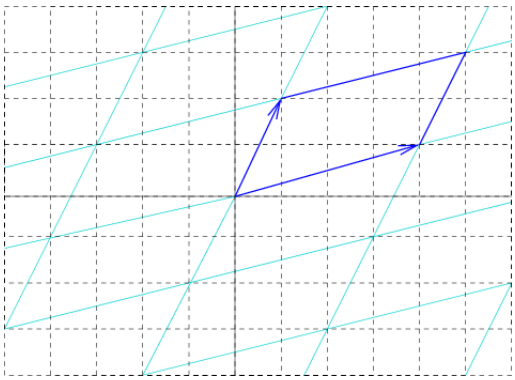


Slika 2.1: Dvo-dimenzionalna rešetka

Vektore $b_1, ..., b_n$ zovemo **bazom** rešetke L. Kao što znamo, baza vektorskog prostora nije jedinstvena već postoji mnogo baza koje definiraju jedno te isti vektorski prostor. Vrijedi li nešto slično za rešetke?

Ako bazu zapišemo kao matricu $\mathbf{B} = (b_1, ..., b_n)$ (dakle matricu kojoj su stupci vektori baze rešetke) rešetku možemo definirati kao $L(\mathbf{B}) = \{\mathbf{B}x : x \in \mathbb{Z}^n\}$

Definicija 8. *Fundementalno područje za danu rešetku $L \subseteq \mathbb{R}^n$ je n-dimenzionalan paralelopiped definiran vektorima baze L.*

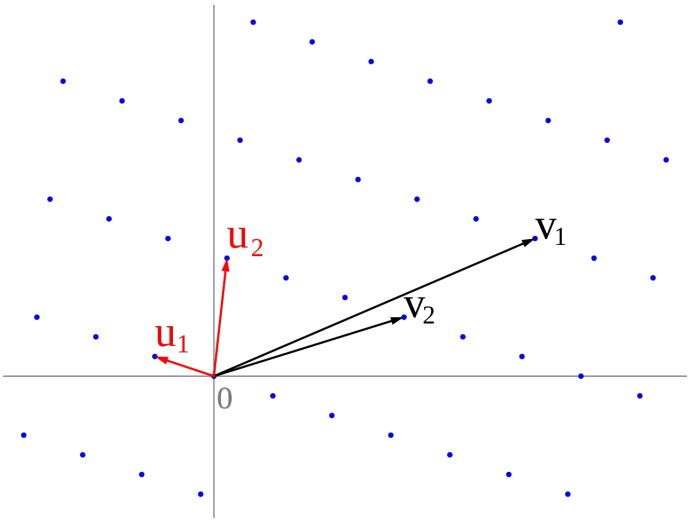


Slika 2.2: Fundementalno područje

Sada možemo dosta lako pokazati vezu između mogućih različitih baza rešetke.

Teorem 2.1.1. *Neka je B n-dimenzionalna matrica i neka je $L(B)$ rešetka te neka joj je B baza. Neka je U proizvoljna n-dimenzionalna unimodularna matrica. Tada vrijedi $L(B) = L(BU)$.*

Iz ovog teorema slijedi da ima beskonačno mnogo baza za neku konkretnu rešetku. Na slici 2.3 možemo vidjeti dvije takve baze. Nadalje, slijedi također i da dvije različite baze koje defniraju istu rešetku imaju fundementalno područje iste površine (ovo dobijemo iz činjenice da je det(U)=1). Već u ovim ranim konstatacijama o bazama dolazimo do suštine težine problema koji okružuju rešetke a to su razlike između baza iste rešetke.



Slika 2.3: Dvije različite baze koje definiraju istu rešetku

Definicija 9 (Dualni Defekt Ortogonalnosti)**.** *Neka je B nesingularna $n \times n$ matrica. Njen dualni defekt ortogonalnosti je definiran kao: $do^*(B) := \frac{\prod_i \|b_i^*\|}{|\det(B^{-1})|} = |\det(B)| \prod_i \|b_i^*\|$ gdje su b_i^* i-ti redak matrice B^{-1}.*

Naime, baza rešetke može biti *dobra* ili *loša*. Za bazu kažemo da je loša ako joj je dualni ortogonalni defekt velik, dok su dobre baze one s niskim dualnim ortogonalnim defektom. Intuitivno, dobrom bazom smatramo onu koja se sastoji od kratkih, gotovo ortogonalnih vektora kao što su u_1 i u_2 na slici. V_1 i V_2 primjeri su loše baze. Ako nam je rešetka zadana nekom lošom bazom, mnogo naigzled benignih problema postaje izrazito teško za riješiti.

Za kraj ovog poglavlja dokazati ćemo jedno svojstvo rešetka koje opravdava postavljanje problema koje ćemo postaviti u sljedećem podpoglavlju, tj. garantira njihovu riješivost.

Postojanje malog vektora

Teorem 2.1.2 (Blichfeld)**.** *Neka je B n-dimenzionalna matrica i neka je L(B) rešetka kojoj je B baza. Neka je $S \subseteq \mathbb{R}^n$ proizvoljan takav da $V(S) < det(B)$, gdje V(S) označava volumen skupa S a det(B) volumen fundementalnog područja rešetke L.*

Tada postoji $z_1, z_2 \in S$, $z_1 \neq z_2$ takvi da je $z_1 - z_2 \in L$.

Ovaj teorem je samo predjelo za sljedeći teorem u kojemu se skriva svojstvo koje tražimo.

Teorem 2.1.3 (Minkowski)**.** *Neka je B baza za proizvoljnu rešetku L i konveksan ishodištno-simetričan skup S volumena $> 2^n det(B)$.*

Postoji točka rešetke unutar skupa S.

Korolar 2.1.4 (Minkowski)**.** *Za svaku rešetku L postoji vektor $v \in \mathbb{R}^n$ takav da vrijedi* $||v|| \leq \sqrt{n} \cdot \det(B)^{1/n}$

Gdje je B baza rešetke L.

Dakle opisno, ovaj korolar nam kaže da za danu rešetku postoji "dovoljno mali" vektor te još imamo i gornju ogradu na njega. Kao što smo najavili, postojanje kratkih vektora nam je ključno za većinu problema vezanih za rešetke, te ovakvo svojstvo garantira da su ti problemi riješivi.

Kratka opaska, nismo definirali normu u prethodnom korolaru. Mnoge norme se mogu promatrati u ovom području, no detalji oko norma nam nisu od velike važnosti i koristiti ćemo najintuitivniju, euklidsku normu.

U sljedećem podpoglavlju analizirati ćemo neke od najbitnijih problema nad rešetkama. Prije kraja treba napomenuti da iako su u primjerima korištene dvodimenzionalne rešetke radi jednostavnosti, za potrebe kriptografije nas će zanimati rešetke izrazito velikih dimenzija.

2.2 Problemi nad rešetkama

Krenimo sa par problema koji su prirodni za postaviti ali su ultimativno izračunjlivo lagani.

Najprirodniji problem je pitati se za zadanu bazu B i neki vektor v $\in \mathbb{R}^n$ je li vektor $v \in L(B)$. Ovaj problem se trivijalno brzo rješava Gaussovom eliminacijom.

Sljedeće što možemo tražiti je ako su zadane baze B_1 i B_2 jesu li njihove rešetke iste tj je li $L(B_1) = L(B_2)$. Argumentirati laganost (algebarski problem množenja matrica).

Mana ovih problema što su algebarske prirode i računalima lako probavljivi. Geometrijski problemi, u drugu ruku se čine izrazito teški. Shodno tome, svi problemi na

koje se oslanja kriptografija bazirana na rešetkama su u suštini geometrijske prirode. Slijedi pregled najbitnijih.

SVP

Problem kratkog vektora (*eng*, **S**hortest **V**ector **P**roblem) definiran je na sljedeći način:

Dana nam je baza rešetke (loša baza) te treba pronaći najkraći ne-nul vektor u rešetci.

Za ovu varijantu problema SVP-a, dakle egzaktno rješenje, najbolji algoritmi do danas imaju vremensku složenost $2^{\mathcal{O}(n)}$, i to bez ikakvog većeg pomaka od 1982 kada je predstavljen tzv. LLL algoritam (Lenstra, Lenstra, Lovasz) koji ne rješava egzaktni SVP već u polinomijalnom vremenu pronalazi aproksimaciju najkraćeg vektora koji je otprilike za faktor $2^{\mathcal{O}(n)}$.

Makar činjenica da do sada nije pronađen efikasni algoritam ne uljeva sigurnost jer se možda već sutra otkrije algoritam koji je efikasan za ovaj problem treba biti svijestan da to vrijedi za svaki problem i da sporost kojom se napreduje na polju efkisanosti ovih algoritama je ozbiljan razlog za nadanje da su ovi problem izrazito izračunljivo teški. Kao kontrast ovome spomenimo da se na području faktorizacije brojeva (na kojem se, prisjetimo, bazira najrasprostranjeniji kriptosustav danas: RSA) kroz godine došlo do ozbiljnih ubrzanja na algoritmima, od kojih je najbitnije spomenuti Sito Polja brojeva (GNFS).

Često za potrebe kriptosustava ovakvi egzaktni problemi su krajnje nepraktični. U našem slučaju SVP-a javni ključevi postanu preglomazni za praktičnu upotrebu, stoga se promatra blaža varijanta ovog problema koji zahtjeva da nađemo vektor koji je najviše γ veći od najkraćeg vektora. Odabir ovog faktora je od velike važnosti i uvelike utječe na težinu problema. Ako za faktor γ uzmemo nešto veće od $\sqrt{n/\log(n)}$ problem nije NP-težak. SVP je NP-težak samo za faktore manje od $n^{\mathcal{O}(1/loglog(n))}$.

CVP

Problem najbližeg vektora (*eng*, **C**losest **V**ector **P**roblem) definiran je na sljedeći način: Dana nam je baza rešetke $\mathbf{B}$ i vektor $\mathbf{t}$ koji nije nužno element rešetke. Treba pronaći točku rešetke $\mathbf{v} \in L(B)$ koja je najbliža vektoru $\mathbf{t}$.

Emde Boas je dokazao da je ovaj problem NP-težak, a Babai je osmislio algoritam koji riješava aproksimaciju ovog problema.

SIVP

Problem najkraćih nezavisnih vektora (*eng*, **S**hortest **I**ndependent **V**ectors **P**roblem): Zadana je baza B n-dimenzionalne rešekte L. Zadatak je pronaći n međusobno nezavisnih vektora $s_1, ..., s_n$ ($s_i \in L(B)$ za svaki i) tako da minimiziramo vrijednost $M = max_i\|s_i\|$.

SBP

Problem najmanje baze (*eng*, **S**mallest **b**ases **P**roblem). Zadana nam je loša baza B i treba pronaći bazu D koja ima najmanji defekt ortogonalnosti.

Za ovaj problem ne postoji polinomijalni algoritam, te najbolje što za sada postoji je LLL koji riješava dosta ograničenu verziju ovog problema (na faktor aproksimacije $2^{O(n^2)}$) u polinomijalnom vremenu.

2.3 GGH kriptosustav s javnim ključem

Kriptosustav koji ćemo ovdje prezentirati nosi ime po svojim kreatorima: Goldreich, Goldwasser i Halevi, koji su ga 1996. godine prezentirali kao potencijalnu alternativu RSA-u. U svojem članku objavili su 5 numeričkih izazova sve većeg sigurnosnog faktora. Spomenuti faktori su $n = 200, 250, 300, 350, 400$. Naravno, ruku uz ruku s većim sigurnosnim faktorom ide i veličina javnog ključa koji je jedan od ključnih praktičnih problema koji kriptosustav s javnim ključem treba zadovoljiti. U našem slučaju veličina ključa se kreće od 330 Kb (što je više od 1000 puta više nego što je potrebno za RSA sustav) za male n-ove, pa sve do 2 Mb za velike n-ove što je izrazito nepraktično. Ova činjenica proizlazi iz toga što je javni ključ GGH kriptosustava $n \times n$ matrica. No još gore od toga, 2009. godine Nguyen i Regev u svojem članku "Learning a Parallelepiped" prezentiraju napad koji probija GGH za sve faktore osim $n = 400$. To ga ne čini "probijenim" ali znatno umanjuje njegovu kredibilnost kao mogućoj praktičnoj alternativi RSA-u u dogledno vrijeme.

No usprkos ovih veoma teških mana koje GGH kriptosustav ima, i dalje ga ima smisla proučavati jer sadrži neke fundementalne ideje koje se koriste u mnogim kriptosustavima koji se grade nad njime i nemaju gore navedene probleme.

Nadalje, GGH posjeduje i neke prednosti. Naravno, za nas najbitnije, odoljeva kvantnim napadima, tj. još uvijek ne postoji kvantni algoritam koji ga efikasno riješava. Zatim, asimptotski je efikasniji od RSA koristeći modularne eksponente (to znači da su vremena potrebna za enkripciju, dekripciju, potpisivanje te verifikaciju manje, kvadratne za razliku od kubnih u faktoru sigurnosti), te za kraj ima prirodnu

shemu potpisa o kojoj nećemo ovdje puno pričati no znatiželjnog čitatelja upučujemo na popis referenci.

Slijedi pregled cijelog kriptosustava prije nego li se fokusiramo na detalje.

Pregled kriptosustava

Generacija javnog ključa

- Kreiramo *dobru* bazu B.
- Transformiramo bazu B u *lošu* bazu Q.
- Objavljujemo bazu Q kao javni ključ a čuvamo bazu B kao privatni ključ.

Enkripcija

- Vektor **v** koji je naša poruka množimo s lijeva matricom Q te mu dodajemo vektor pogreške **p**. Tako dobiveni vektor nazovimo **c**.
- Šaljemo c kao enkriptiranu poruku.

Dekripcija

- Koristeći privatnu bazu, izračunamo pomoću Babai-evog algoritma vektor rešetke koji je najbliži vektoru **c**. To je upravo $c - p$.
- Dobijemo poruku **v** tako da vektor $c - p$ množimo s lijeva sa Q^{-1}.

Sigurnosne pretpostavke

- Lako je izračunati *lošu* bazu iz *dobre* ali obrnuto je teško.
- Čak i s *lošom* bazom lako je napiknuti nasumični vektor iz rešetke.
- Lako je pronaći najbliži vektor iz rešetke ako imamo *dobru* bazu, a teško ga je pronaći ako nam je dana *loša* baza.

Prva pretpostavka je upravo SBP, a za zadnju proces pronalaska najbližeg vektora s *dobrom* bazom opisati ćemo koristeći već spomenuti Babai-ev algoritam zaokruživanja, a pronalazak istog vektora *lošom* bazom je CVP.

Kriptosustav

Generacija ključeva

Počinjemo s definicijom dimenzije baze B. Što je veća dimenzija to je veća sigurnost, no prostor javnog ključa i brzina dekripcije rastu barem za faktor $\Omega(n^2)$. Nakon Nguyen-ovog napada znamo da n mora biti barem 400 da garantiramo sigurnost.

Zatim generiramo bazu B. Biramo nasumičnu matricu, ali trebamo definirati kako, tj. iz koje distribucije. Biramo matricu uniformno iz $\{-l, \cdots, +l\}^{n\times n}$, gdje su l-ovi (eksperimentalno pokazano) mali, točnije do 4.

Da bi dobili bazu Q bazu B ćemo množiti s nizom unimodularnih matrica, dakle $Q = B \cdot T_1 \cdot T_2 \cdots$. T_i dobijemo kao produkt matrica L_i i U_i gdje je gdje su L i U donja tj. gornja trokutasta matrica gdje su elementi dijagonale ± 1 a ne-nul elementi iz $\{-1, 0, 1\}$. Što se broja množenja tiče, eksperimentalno je pokazano da nam je dovoljno množiti s 4 T-a. Primjetimo da nam teorem 2.1.1 garantira da ove dvije baze daju istu rešetku.

Enkripcijska funkcija

Imamo neka ograničenja na to kakav vektor p treba biti, jer treba paziti da nemamo značajne greške tijekom dekripcije. Pokazati ćemo da elementi vektora pogreške trebaju biti iz $[-\sigma, \sigma]$ za neki $\sigma \in \mathbb{R}$. Što možemo reći o σ?

Lema 2.3.1. *Neka je B privatna baza korištena za dekripciju. Greška prilikom dekripcije se događa ako i samo ako* $\lceil B^{-1}\mathrm{e} \rfloor \neq \overline{\mathbf{0}}$.

Dokaz. □

Teorem 2.3.2. *Neka je B privatna baza korištena tijekom dekripcije, te neka je maksimum L_1 norme redaka B^{-1} označen s m. Tada dok je god $\sigma < 1/(2m)$, neće doći do pogreške tijekom dekripcije.*

Dekripcijska funkcija

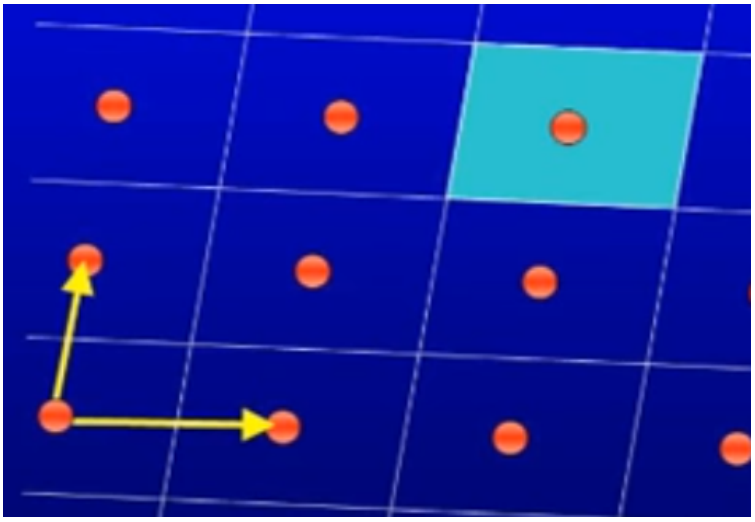
Da zaokružimo cjelinu analizirati ćemo Babai-ev Algoritam Zaokruživanja. Slijedi opis algoritma a kasnije dokaz da on zaista pronalazi najbliži vektor.

Algoritam: Neka je naš input vektor $x = \sum_{i=1}^{n} \beta_i b_i$ gdje su b_i vektori javne baze Q, te neka je α_i najbliži cijeli broj realnom broju β_i. Izlazni vektor je $w = \sum_{i=1}^{n} \alpha_i b_i$.

Ono što radimo u algoritmu je izrazimo naš dani vektor preko baze rešetke i onda za svaki koeficijent u tom izrazu tražimo njemu najbliži cijeli broj jer znamo da je rešetka definirana na cjelobrojnim kombinacijama te se *nadamo* da smo tako pogodili vektor rešetke koji nam treba. Sada treba jako dobro promotriti kada ovaj algoritam

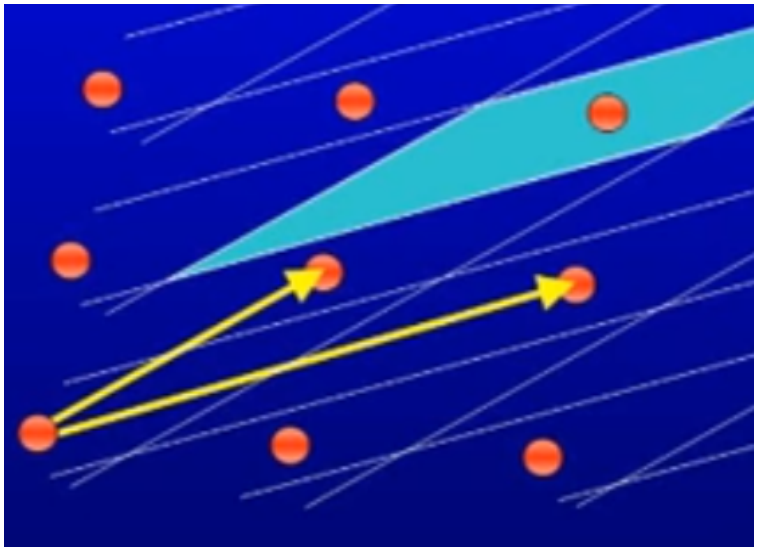
pogriješi tj. daje krivi vektor (ovo je sigurno moguće jer nemamo nikakvu garanciju da će ova procedura baš pogoditi vektor rešetke, pogledati sliku) te staviti ograničenja da se to ne dogodi, jer nadati se da smo pogodili jednostavno nije dovoljno dobro. Promotrimo kada algoritam uspjeva a kada ne.

Na slici 2.4. vidimo rešetku sa dobrom bazom. Područje koje je označeno označava sve točke koje će se cjelobrojno zaokružiti na vektor u sredini tog područja. Ovo je zapravo translatirano fundementalno područje. Primjetimo da je ovo dobra situacija i da za koju god točku piknemo iz označenog područja imamo jako dobre šanse da ćemo dobiti pravit vektor zaokruživanjem.



Slika 2.4: Kada zaokruživanje uspjeva, dobra baza

Na slici 2.5. vidimo istu rešetku, ali sada izraženu u lošoj bazi. Područje je označeno istom logikom. Odmah uočavamo da ako odaberemo vektor pri samom ljevom rubu područja biti će zaokružen na vektor u sredini područja umjesto na vektor koji mu je bliži.



Slika 2.5: Kada zaokruživanje ne uspjeva, ista rešetka ali loša baza

Ključna dva teorema reći će nam koja su to ograničenja da bi algoritam uspio te koliko greška mu je moguća. Prije samog iskaza teorema treba definirati što je to Lovacz-reducirana baza te istaknuti činjenicu da naše dobre baze ili jesu Lovasz-reducirane ili veoma lako postanu.

Definicija 10 (Lovacz-reducirana baza)**.** *Neka su $b_1, \ldots, b_n \in \mathbf{R}^n$ linearno nezavisni vektori i neka je L rešetka definirana s bazom $B = \{b_1, \ldots, b_n\}$. Neka su $b_1^*, \ldots, b_n^*$ ortogonalizirane sekvence pridružene $b_1, \ldots, b_n$. Tada imamo:*

$$b_i = \sum_{j=1}^{a} \mu_{ij} b_j^* = b_i^* + \sum_{1 \leq j < i} \mu_{ij} b_j^*$$

gdje su

$$\mu_{ij} = 0 \quad za \quad 1 \leq i < j \leq n$$

$$\mu_{il} = 1 \quad (i = 1, \ldots, n)$$

Za bazu B kažemo da je Lovacz reducirana ako sljedeća dva uvjeta vrijede:

- $|\mu_{i,j}| \leq \frac{1}{2}$ *za svaki* $i < j$
- $|b_i^*| \geq \frac{|b_{i-1}^*|}{\sqrt{2}} \quad za \quad i = 2, \ldots, n$

Teorem 2.3.3 (O obliku Lovacz-reduciranog paralelopipeda)**.** *Neka je B Lovacz reducirana baza. Neka je θ_k kut između b_k i linearnog podprostora $U_k = \sum_{j \neq k} \mathbf{R} b_j$. Tada za svaki $k (1 \leq k \leq n)$ vrijedi:* $\sin \theta_k \geq (\sqrt{2}/3)^n$.

Teorem 2.3.4 (Babai-ev algoritam zaokruživanja)**.** *Ako je baza B Lovasz-reducirana, onda algoritam zaokruživanja daje točan vektor rešetke w koji je najbliži zadanom vektoru x do na faktor $C'_n = 1 + 2n(9/2)^{n/2}$.*

Unaprijeđenja GGH kriptosustava

Za generaciju baze Q Micciancio je našao efikasniju metodu od originalne kada je predložio da jednostavno iskoritimo Hermitsku normalnu formu matrice B. Prisjetimo se, za svaku matricu postoji njoj unikatna matrica H koja je njena HNF i dobijena je kao UA = H gdje je U unimodularna matrica. Po Teoremu 2.1.1. znamo da H i B definiraju istu rešetku.

Ovo unaprijeđenje smanjuje veličinu javnog ključa za otprilike faktor 10.

2.4 Zaključak

Vidjeli smo da u svijetu rešetki postoji mnogo izračunljivo teških problema za koje još ne postoje kvantni algoritmi, te smo prokomentirali neke konkretne već predložene kriptosustave. Prednost koju do sada nismo spomenuli je da svi sustavi bazirani na rešetkama koriste jako jeftine komputacijske operacije, u pitanju je samo niz zbrajanja.

Nadalje, činjenica da većina kriptosustava uživaju u svojstu average-case težine dodatno utvrđuje činjenicu da su ovi problemi adekvatni za kriptografiju.

Sve navedene činjenice upučuju na jako svijetlu budućnost kriptografije bazirane na rešetkama, pogotovo u slučaju adventa kvantnih računala.

Poglavlje 3

Sustavi baziran na kodovima

Kodovi su prvenstveno osmišljeni kao sustav koji osigurava čitljivost poruke u komunikacijskim kanalima gdje je moguće da dođe do greške u poruci prilikom njenog prijenosa. Na primjer, recimo da se sa vjerojatnošću p događa greška u svakom bitu poruke. Želimo smanjiti tu šansu.

Preciznije, ako šaljemo neku poruku od k bitova $\boldsymbol{u} = \boldsymbol{u}_1\boldsymbol{u}_2, \ldots, \boldsymbol{u}_k$ zakodirati ćemo ju u poruku (kodnu riječ) $\boldsymbol{x} = x_1x_2 \ldots x_n$ gdje je $n \geqslant k$. Takva poruka bi trebala imati manju šansu da bude nečitka ako se u njoj nalazi greška. Promatrati ćemo linearne kodove koji mogu poslužiti toj svrsi.

3.1 Linearni kod

Kao primjer uzmimo neku najjednostavniju ideju: Prvi dio poruke ostavimo isti i na njega konkataniramo $n - k$ bitova koji ovise o originalnoj poruci. Dakle:

$$x_1 = u_1, \quad x_2 = u_2, \quad \ldots, \quad x_k = u_k$$

Sljedeći bitovi, $x_{k+1}, \ldots, x_n$, su odabrani tako da kodna riječ zadovoljava:

$$H \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = Hx^{\mathrm{tr}} = 0$$

H je $(n - k) \times n$ matrica koju nazivamo *matrica provjere pariteta.* Ona je oblika:

$$H = [A|I_{n-k}]$$

Gdje je A neka $(n-k) \times k$ matrica, a I_{n-k} $(n-k) \times (n-k)$ matrica identiteta. Praktično nam je definirati što je tzv. *generacijska matrica*. To je matrica koja opisuje kako iz poruke doći do koda, tj. $\boldsymbol{x} = \boldsymbol{u}G$, gdje je G spomenuta generacijska matrica. Nju ćemo izvući preko gore navedene jednadžbe matrice provjere pariteta.

$$[A|I_{n-k}] \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = 0$$

$$\begin{pmatrix} x_{k+1} \\ \vdots \\ x_n \end{pmatrix} = -A \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix}$$

$$= -A \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix}$$

Odakle dobijemo $G = [I_k| - A^w]$. Primjetimo da kako u ovom konkretnom slučaju radimo s binarnom aritmetikom zapravo je $-A = A$.

Parametri linearnog koda

Naš kod ima blok duljine n. Primjetimo da ako H ima $n-k$ linearno nezavisnih vektora, u našem slučaju postoji 2^k kodnih riječi. k ćemo zvati dimenzijom koda. Notacija koju ćemo koristiti je $[n, k]$ kod.

Primjetimo da je kodna riječ uvijek veća od same poruke pa ima smisla promatrati efikasnost koda koja se definira kao $R = k/n$.

Nakon što smo promotrili ovaj jednostavni binarni linearni kod, spremni smo definirati općeniti.

Definicija 11 (Linearni kod)**.** *$[n, k]$ linearni kod definiran na konačnom polju $\mathbb{F}_q$ je k-dimenzionalni linearni podprostor $C \subseteq \mathbb{F}_q^n$. Postoji $k \times n$ dimenzionalna generacijska matrica G te $(n-k) \times n$ dimenzionalna matrica H provjere parnosti za koje vrijedi:*

$$C = \left\{uG | u \in \mathbb{F}_q^k\right\} = \left\{x \in \mathbb{F}_q^n | Hx^T = \mathbf{0}\right\}$$

Primanje poruke

Vratimo se na naš originalni primjer.

Poslana nam je kodna riječ $x = x_1 \cdot \cdot x_n$, no zbog moguće greške, primili smo $y = y_1 \cdots y_n$. Definirajmo *vektor greške* kao:

$$e = y - x = e_1 \cdots e_n$$

Na nama je da za dani y pokušamo pronaći vektor e te tako dobijemo x i iz njega poruku u. Nažalost, kako ne možemo biti sigurni u vektor e, naša strategija će biti da pronađem e koji je najvjerojatniji.

Definicija 12 (Hammingova udaljenost)**.** *Hammingova udaljenost između dva vektora $x = x_1 \cdots x_n$ i $y = y_1 \cdots y_n$ je definirana kao broj mjesta gdje se x i y razlikuju.*

$$hu(x, y) = card(\{x_i | x_i \neq y_i\})$$

Definicija 13 (Hammingova težina)**.** *Hammingova težina vektora $x = x_1 \cdots x_n$ definirana je kao broj ne-nul mjesta.*

$$ht(x) = card(\{x_i | x_i \neq 0\})$$

Zgodno je primjetiti da $hu(x, y) = ht(x - y)$.

Nadalje, primjetimo sljedeću konstataciju: Za proizvolji vektor v s $ht(v) = a$, tada za vektor greške vrijedi

$$\mathbb{P}(\{e = v\}) = p^a(1 - p)^{n-a}$$

A kako je p u slučaju kodova $< 1/2$ zaključujemo da su najvjerojatniji vektori greške oni s malim ht. Ovaj pristup zovemo dekodiranje najbližeg susjeda.

Za samo dekodiranje možemo jednostavno primitivno usporediti poruku sa svim mogućima i uzeti najbližu, ali ovo postaje izrazito nezgodno za iole velike k-ove. Jedan od glavnih faktora u evaluaciji kodova je koliko efikasan možemo naći algoritam dekodiranja.

Još jedan bitana parametar koda je njegova minimalna Hammingova udaljenost.

Definicija 14 (Minimalna Hammingova udaljenost koda)**.** *Za dani kod C definiramo minimalnu Hammingovu udaljenost d kao*

$$d = \min_{w \in C, w \neq 0} ht(w)$$

Kako to da ne moramo provjeravati udaljenost između svaka dva člana koda? Naime, za proizvoljni $x, y \in C$ vrijedi $x - y \in C$. Veoma se lako uvjeriti u ovo preko jednadžbe za provjeru pariteta $H(x+y)^{\text{tr}} = Hx^{\text{tr}} + Hy^{\text{tr}} = 0$. Odavde naposljetku i ime *linearni kod*. Samo još iskoristimo vezu između udaljenosti i težine i dobijemo gore navedenu formulaciju. Često koristimo notaciju $[n, k, d]$ da naglasimo minimalnu Hammingovu udaljenost. Sada možemo dokazati jedno zgodno svojstvo.

Teorem 3.1.1. *Kod C sa minimalnom Hammingovom udaljenosti d može korigirati $\left\lceil \frac{1}{2}(d-1) \right\rfloor$ grešaka. Ako je d paran, kod može simultano korigirati $\frac{1}{2}(d-2)$ grešaka i detektirati $d/2$ grešaka.*

U zadnjem teoremu spomenuli smo detekciju grešaka. Ovo svojstvo može biti od velike praktične važnosti, jer ako možemo tražiti da se poruka pošalje ponovno, možemo samo detektirati jesu li se dogodile greške (ili je li se dogodilo previše grešaka) te tražiti ponovno slanje poruke. Ovo je uglavnom jednostavniji problem, i ako komunikacija nije skupa postaje jako primamljiva.

Definicija 15 (Koset koda)**.** *Neka je C $[n, k]$ linearni kod. Za svaki vektor a definiramo skup koji nazivamo njegovim kosetom ili translatom kao:*

$$a + C = \{a + x : x \in C\}$$

Zašto nam je bitna ova naočigled nebitna definicija? Promotrimo sljedeću konstataciju: primili smo vektor y koji mora pripadati nekom kosetu, recimo $y = a + x(x \in C)$. Pitamo se koji su mogući vektori greške e? Recimo da je poslana poruka $m \in C$, tada imamo

$$e = y - m = a_i + x - m = a_i + x'' \in a_i + C$$

Što će reći, ne moramo tražiti vektor greške po svim vektorima, vektor greške se sigurno nalazi u kosetu vektora y.

Teorem 3.1.2. *Dva koseta ili nemaju presjek ili su jednaki.*

Dokaz. Neka su $a + C$ i $b + C$ dva koseta. Uzmimo vektor $v \in (a + C) \cap (b + C)$. Tada vrijedi

$$\boldsymbol{v = a + x = b + y}$$

Gdje su x i y oboje iz C. Dalje vrijedi

$$b = a + x - y = a + x' \, (x' \in C)$$

Iz čega slijedi da je $b + C \subseteq a + C$, a istim principom dobijemo obratnu tvrdnju iz čega slijedi $b + C = a + C$, tj. ako dva koseta imaju presjek onda oni nužno moraju biti isti. □

Pomalo nalik na rastav skupa na klase ekvivalencije možemo skup svih vektora F^n raspisati kao

$$F^n = C \cup (a_1 + C) \cup (a_2 + C) \cup \cdots \cup (a_i + C)$$

Gdje je $t = q^{n-k} - 1$. a_i se biraju tako da je njihova težina najmanja moguća i sada vidimo koliko nam to olakšava posao prilikom dekodiranja.

Za kraj dva teorema koji zaokružuju priču.

Teorem 3.1.3 (Singleton ograda)**.** *Ako je $C\,[n, k, d]$ kod tada vrijedi $n - k \geqslant d - 1$*

n-k je zapravo rank matrice H i označiti ćemo ga s r. Kodovi za koje vrijedi $r = d - 1$ zovu se *maksimalno separirani kodovi* (**MDS**) i od posebne su važnosti za upotrebu.

Teorem 3.1.4 (Gilbert-Varshamova ograda)**.** *Postoji binarni linearni kod C duljine n, s matricom pariteta ranga najviše r i minimalnoj udaljenosti d ako vrijedi*

$$1 + \begin{pmatrix} n-1 \\ 1 \end{pmatrix} + \cdots + \begin{pmatrix} n-1 \\ d-2 \end{pmatrix} < 2^r$$

Značaj ovog teorema je u tome što nam garantira postojanje "dobrih" linearnih binarnih kodova.

3.2 Problemi nad kodovima

BDD

Dekodiranje s ogradom na udaljenost (*eng*, **B**ounded **D**istance **D**ecoding) je problem definiran na sljedeći način: dan nam je linearni kod $C \subseteq F_a^n$, dan nam je njegov element $y \in F_a^n$ te Hammingova udaljenost $t \in \mathbb{N}$. Cilj je pronaći x takav da $x \in C$: $\mathrm{hu}(x, y) \leq t$.

1978. godine Berlekamp dokazuje da je BDD NP-težak problem.

3.3 McEliece kriptosustav s javnim ključem

Kriptosustav koji ćemo prezentirati objavljen je od strane Robert J. McEliece godine 1978. Čak nakon 4 desetljeća, nije pronađen adekvatni napad na ovaj sustav, kako klasični tako kvantni, iako je postalo potrebno korigirati neke parametre. No ponovno susrećemo isti problem na koji smo naišli s GGH kriptosustavom u prethodnom poglavlju: javni ključ je ponovno velika matrica. Takav veliki javni ključ je glavna stvar koja priječi ovaj kriptosustav od svijetske slave.

Pregled kriptosustava

Generacija ključeva

- Odaberemo neki kod Γ s generacijskom matricom G koji može korigirati t pogrešaka.
- Odaberemo ne-singularnu matricu S i nasumičnu permutacijsku matricu P, te izračunamo $G' = SGP$.
- Šaljemo (G', t) kao javni ključ a (S, G, P) čuvamo kao privatni ključ.

Enkripcija

- Zakodiramo svoju poruku x te joj dodamo vektor pogreške e s težinom t. Šaljemo poruku rezultat kao poruku y.

Dekripcija

- Množimo y s P^{-1}.
- Dekodiramo koristeći algoritam za dekodiranje danog koda da dobijemo xS. Množimo se inverzom od S i dobijemo poruku.

Sigurnosne pretpostavke

- BDD problem je težak.
- Lako je generirati G' iz G ali obratno je teško.

Kriptosustav

Generacija ključeva

Odaberimo $m \in \mathbb{N}, n = 2^m, t < \frac{n}{m}$ te binarno ireducibilni Goppa kod Γ $[n, k]$ takvi da je $k \geq n - mt$ koji može korigirati t pogrešaka. Ovdje nećemo objašnjavati detalje vezane uz same Goppa kodove, jer za ideju kriptosustava konkretna familija kodova je od manje važnosti, ali je bitno naglasiti da za samu sigurnost nikako nije.

Znamo da Γ ima generacijsku matricu G dimenzije $k \times n$. Generiramo $k \times k$ dimenzionalnu matricu S takvu da $\det(S) \neq 0$ i nasumičnu $n \times n$ dimenzionalnu matricu permutacije P te zamaskiramo našu generacijsku matricu u matricu G' tako da $G' = SGP$.

Kao javni ključ šaljemo matricu G' i broj grešaka t, a kao privatni ključ čuvamo matrice S, G i P.

Enkripcija

Našu poruku $x \in \mathbb{F}_2^k$ množimo s G' s desna te joj dodajemo nasumičan vektor pogreške $e \in \mathbb{F}_2^n$ pazeći da ht(e)=t. Dakle poruka koju šaljemo je oblika

$$y = xG' + e$$

Dekripcija

Izračunamo

$$yP^{-1} = xSG + eP^{-1}$$

Ovdje nam je bitno da je P permutacijska matrica jer tada vrijedi

$$ht(eP^{-1}) = ht(e) = t$$

I znamo da možemo dekodirati poruku. Kako je xSG kodna riječ koda Γ možemo ju dekodirati i dobiti poruku xS. Kako je S invertibilna možemo pomnožiti s njenim inverzom i dobiti našu poruku x.

3.4 Zaključak

Prvo, volio bih konstatirati da ako promatramo dovoljno apstraktkno, dva kriptosustava koja smo obradili u ovom radu su zapravo gotovo analogna. U oba slučaja imamo neku strukturu, odaberemo jedan element iz te strukture te se pomaknemo od njega za neki vektor greške i pretpostavljamo da se teško vratiti iz te pomaknute točke natrag u element strukture.

Vidjeli smo primjer jednog kriptosustava, i vidimo da zapravo možemo dobiti dosta analognih kriptosustava ako odaberemo drugu klasu kodova, iako većina njih nije tako sigurna kao što su to Goppa kodovi.

Sam proces enkripcije i dekripcije, ponovno kao i u slučaju rešetaka je otprilike duplo brži nego kod RSA, što je nezanemariva informacija.

Činjenica da kroz sve ove godine nije pronađen niti jedan efikasni napad na McEliece-ov kriptosustav niti njegove varijante uljeva jako puno sigurnosti u ovu granu kriptografije.

Glavni i zasad jedini problem je veličina javnog ključa, i kroz sve ove godine nismo ga uspjeli dovoljno smanjiti. Makar ovo čini kriptosustave bazirane na kodovima dosta nepraktičnima, jednom kada kvantna računala postanu stvarnost oni su među zadnjim zidovima obrane koje do sada poznajemo.

Bibliografija

[1] László Babai, *On Lovász'lattice reduction and the nearest lattice point problem*, Combinatorica **6** (1986), br. 1, 1–13.

[2] Bhaskar Biswas i Nicolas Sendrier, *McEliece cryptosystem implementation: Theory and practice*, International Workshop on Post-Quantum Cryptography, Springer, 2008, str. 47–62.

[3] Erik Dahmen Daniel J. Bernstein, Johannes Buchmann, *Post-Quantum Cryptography*, Springer-Verlag, 2009.

[4] Oded Goldreich, Shafi Goldwasser i Shai Halevi, *Public-key cryptosystems from lattice reduction problems*, Advances in Cryptology — CRYPTO '97 (Berlin, Heidelberg) (Burton S. Kaliski, ur.), Springer Berlin Heidelberg, 1997, str. 112–131, ISBN 978-3-540-69528-8.

[5] Vadim Lyubashevsky i Daniele Micciancio, *On bounded distance decoding, unique shortest vectors, and the minimum distance problem*, Annual International Cryptology Conference, Springer, 2009, str. 577–594.

[6] Robert J McEliece, *A public-key cryptosystem based on algebraic*, Coding Thv **4244** (1978), 114–116.

[7] Phong Q. Nguyen i Oded Regev, *Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures*, Journal of Cryptology **22** (2009), br. 2, 139–160, ISSN 1432-1378, `https://doi.org/10.1007/s00145-008-9031-0`.

[8] Seong Hun Paeng, Bae Eun Jung i Kil Chan Ha, *A lattice based public key cryptosystem using polynomial representations*, International Workshop on Public Key Cryptography, Springer, 2003, str. 292–308.

[9] Claus Peter Schnorr, *A hierarchy of polynomial time lattice basis reduction algorithms*, Theoretical computer science **53** (1987), br. 2-3, 201–224.

[10] Violetta Weger, *A code-based cryptosystem using GRS codes*, Disertacija, Verlag nicht ermittelbar, 2016.

[11] Wikipedia, *One way function*, `https://en.wikipedia.org/wiki/One-way_function`.

[12] _____, *Quantum computing*, `https://en.wikipedia.org/wiki/Quantum_computing`.

[13] _____, *Quantum logic gate*, `https://en.wikipedia.org/wiki/Quantum_logic_gate#Controlled_(cX_cY_cZ)_gates`.

[14] _____, *Qubit*, `https://en.wikipedia.org/wiki/Qubit`.

[15] _____, *Shor's algorithm*, `https://en.wikipedia.org/wiki/Shor%27s_algorithm`.

[16] _____, *Trapdoor function*, `https://en.wikipedia.org/wiki/Trapdoor_function`.

Sažetak

Ovaj rad bavi se alternativnim kriptosustavima koji odoljevaju napadima kvantnih računala. Rad je podjeljen u 3 poglavlja.

U prvom poglavlju dajemo motivaciju za promatranje takvih sustava. Objašnjavamo neke osnovne pojmove iz kriptografije te ističemo neke ključne karakteristike koje čine neki kriptosustav dobrim. Zatim provodimo kratki uvod u kvantno računanje, gdje objašnjavamo razliku između klasičnog i kvantnog računala te pokazujemo kako funkcionira računanje u kvantnom okruženju. Za kraj prvog poglavlja ukratko objašnjavamo najbitniji kvantni algoritam: Shorov algoritam. Shorov algoritam nam omogućuje da lako faktoriziramo brojeve i time probijemo većinu danas korištenih kriptosustava. Upravo iz tog razloga pojavljuje se potreba za kriptosustavima koji se baziraju na drugačijoj klasi problema.

U drugom poglavlju prezentiramo prvu klasu takvih problema, problemi definirani nad rešetkama, te objašnjavamo primjer jednog kriptosustava baziranog na tim problemima, GGH kriptosustav s javnim ključem. U tom poglavlju definiramo rešetke te pokazujemo neka ključna svojstva. Zatim opisujemo većinu bitnih problema nad rešetkama poput SVP-a i CVP-a. Za kraj, definiramo GGH kriptosustav, objašnjavamo kako radi i koje su mu mane i prednosti.

U zadnjem, trećem poglavlju, prezentiramo drugu klasu problema koji trenutačno odoljevaju kvantnim napadima, problemi definirani nad kodovima. Prvo objašnjavamo što su kodovi i kroz primjer objašnjavamo njihova bitna svojstva. Ključno od svih tih svojstava je mogučnost korekcije pogreške i upravo oko tog svojstva je Mceliece dizajnirao svoj kriptosustav. Pred kraj poglavlja objašnjavamo ideju Mceliece-vog kriptosustava te komentiramo njegove kvalitete za praktičnu primjenu.

Summary

This paper deals with alternative cryptosystems which resist the attacks of quantum computers. The work is divided into 3 chapters.

In the first chapter we give the motivation to observe such systems. We explain some basic concepts from cryptography and point out some key features that make a given cryptosystem good. Then we give a short introduction to quantum computing, explaining the difference between a classical and a quantum computer, in addition to showing how computation works in a quantum environment. For the end of the first chapter we briefly explain the most important quantum algorithm: Shor's algorithm. Shor's algorithm allows us to easily factorize numbers and thus break most of the cryptosystems used today. For this reason, there is a need for cryptosystems based on a different class of problems.

In the second chapter we present the first class of such problems, the problems defined over lattices, and we explain an example of a cryptosystem based on these problems, the GGH cryptosystem with a public key. At the beginning this chapter, we define lattices and show some key features. Then we describe most of the major problems on lattices such as SVP and CVP. To conclude, we define the GGH cryptosystem, explain how it works and what are the drawbacks and advantages.

In the last, third chapter, we present another class of problems that are currently resisting quantum attacks, code based problems. First we explain what the codes are and using an example code explain their essential properties. The most important of all these features is the ability to correct errors and precisely around this feature has Mceliece designed his cryptosystem. At the end of the chapter, we explain the idea of the Mceliece cryptosystem and comment on its quality for practical application.

Životopis

Rođen sam 09. 06. 1994. godine u Zagrebu. 2001. godine upisujem osnovnu školu Gračani koju završavam 2009. godine i upisujem XV. Gimnaziju u Zagrebu. Preddiplomski studij matematike, smjer inžinjerski, upisujem 2013. godine na Matematickom odsjeku Prirodoslovno-matematičkog fakulteta u Zagrebu. Akademski naziv prvostupnika matematike stječem 2017. godine i upisujem diplomski studij Računarstvo i matematika na istom odsjeku. Za cijelo vrijeme studiranja profesionalno se bavimo dizajnom igara, što društvenih što računalnih, za razne inozemne i domaće firme.