Slovenská Technická Univerzita v Bratislave
Fakulta Elektrotechniky a Informatiky

McEliece s LDPC kódmi

Tímový projekt

Študijný program:	Aplikovaná informatika
Číslo študijného odboru:	2511
Názov študijného odboru:	9.2.9. aplikovaná informatika
Školiace pracovisko:	Ústav informatiky a matematiky
Vedúci záverečnej práce:	Mgr. Tomáš Fabšič, Ing. Viliam Hromada, PhD.

Bratislava 2016 Bc. Romana Jamrichová, Bc. Peter Kyseľ, Bc. Michal Jenikovský, Bc. Rastislav Páleník, Bc. Andrea Bečárová

Chceme sa pod'akovat' Mgr. Tomášovi Fabšičovi a Ing. Viliamovi Hromadovi, PhD. za odbornú pomoc a usmernenie pri písaní tejto práce, za cenné rady, pripomienky, ochotu, trpezlivosť a obetovaný čas pri vzniku tejto práce.

We are grateful to the HPC center at the Slovak University of Technology in Bratislava, which is a part of the Slovak Infrastructure of High Performance Computing (SIVVP project, ITMS code 26230120002, funded by the European region development funds, ERDF), for the computational time and resources made available.

Contents

List of Figures

List of Algorithms

Kapitola 1

Úvod

Blížiaca sa hrozba zostrojenia dostatočne veľkého kvantového počítača, ktorá by viedla k zlomeniu kryptosystémov založených na probléme faktorizácie a diskrétneho logaritmu, motivuje ľudí k štúdiu a analýze postkvantovej kryptografie. Postkvantová kryptografia je rezistentná voči útoku pomocou kvantového počítača.

V súčasnej dobe sa výskum postkvantovej kryptografie zameriava predovšetkým na týchto 5 rôznych oblastí:

- hašovacie funkcie
- kódovacie funkcie
- grupy mrežových bodov
- viacpremenné kvadratické rovníce
- supersingulárne eliptické krivky

V tejto práci sa budeme venovať kryptosystémom, založeých na dekódovacom probléme, ktorý je vo všeobecnosti NP-úplný problém. Tieto kryptosystémy by mohli nahradiť súčasne používané bezpečnostné štandardy, akým je napríklad RSA. Jedným z takýchto kryptosystémov je McElieceov kryptosystém, založený na lineárnych kódoch. V pôvodnom návrhu sa používali Goppa kódy, ktoré vytvárali veľmi veľké a prakticky nepoužiteľné kľúče. Preto vznikli alternatívne návrhy McElieceovho kryptosystému s cieľom znížiť veľkosť kľúčov so zachovaním bezpečnosti. Takýmito alternatívnymi návrhmi sú QC-MDPC McEliece a QC- LDPC McEliece.

Oba návrhy sú si veľmi podobné a namiesto Goppových kódov, používajú LDPC kódy. Dňa 9.9.2016 bol zverejnený nový útok na QC-MDPC McElieceov kryptosystém. Tento útok je významným pokrokom v oblasti kryptoanalýzy tohto systému a poukazuje na jeho vážne nedostatky. Cieľom tohto projektu je vykonanie užitočných experimentov pre podporu podobného útoku na QC-LDPC McEliece.

1.1 Ponuka

1.1.1 Úvod

V rámci predmetu Tímový projekt sa uchádzame o tému McEliece s LDPC kódmi. Za účelom pridelenia témy sme vypracovali túto ponuku, v ktorej sa v stručnosti pokúsime zdôvodniť, prečo práve náš tím má najlepšie predpoklady na jej úspešné vypracovanie.

V prvej časti dokumentu sa nachádza zadanie témy, v druhej časti predstavenie členov tímu, v poslednej časti dokumentu sme popísalil motiváciu, prečo sme sa rozhodli pre zvolenie tejto témy a hrubý návrh riešenia daného problému.

1.1.2 Zadanie

Vznik efektívneho kvantového počítača by umožnil útoky na mnohé zo súčasných kryptosystémov vrátane RSA. Je preto potrebné budovať kryptosystémy, ktoré sú voči kvantovému počítaču odolné, a ktoré by mohli RSA nahradiť. Jedným z takýchto kryptosystémov je kryptosystém McEliece. Je to kryptosystém založený na lineárnych kódoch. V pôvodnom návrhu tohto kryptosystému sa používali Goppa kódy. Pôvodný návrh má ale nevýhodu, že kľúče v ňom sú veľmi veľké pre praktické účely. Preto vznikli alternatívne návrhy kryptosystému McEliece s cieľom znížiť veľkosť kľúčov. Takýmito alternatívnymi návrhmi sú aj QC-LDPC McEliece a QC-MDPC McEliece.

Tieto dva návrhy sú si navzájom veľmi podobné a namiesto Goppa kódov, sa v nich používajú LDPC kódy. Dňa 9.9.2016 bol zverejnený nový útok na kryptosystém McEliece s QC-MDPC kódmi. Tento útok je významným pokrokom v oblasti kryptoanalýzy kryptosystém McEliece s QC-MDPC kódmi a poukazuje na závažné nedostatky tohto kryptosystému. Útok bol vykonaný skupinou výskumníkov okolo Thomasa Johanssona zo Švédska. Tomáš Fabšič v súčasnosti spolupracuje s Thomasom Johanssonom na vykonaní podobného útoku na McEliece s QC-LDPC kódmi. Cieľom projektu by bolo vykonať počítačové experimenty užitočné pre takýto útok na McEliece s QC-LDPC kódmi.

Štúdiom kryptosystému McEliece sa na FEI zaoberá viacero ľudí. Pavol Zajac viedol diplomové prace a tímové projekty, zamerané na implementáciu kryptosystému McEliece. Výsledným produktom týchto prác, je knižnica BitPunch obsahujúca implementácie kryptosystému McEliece. Knižnica je písaná v jazyku C a je voľne dostupná na GitHube. Obsahuje implementácie McEliece s Goppa kódmi a tiež s QC-MDPC kódmi.V minulom akademickom roku viedli Tomáš Fabšič a Viliam Hromada tímový projekt, ktorý na báze knižnice BitPunch vytvoril implementáciu McEliece s QC-LDPC kódmi.

1.1.3 Tím

Bc. Romana Jamrichová

Názov pozície: Analytik a vedúci projektu
Súvisiace predmety: Základy kódovanie, Základy kryptografie

Je absolventkou bakalárskeho štúdia na Fakulte elektrotechniky a informatiky Slovenskej Technickej Univerzity v Bratislave, v študijnom programe Aplikovaná informatika, odbor Bezpečnosť informačných systémov. Bakalárske štúdium ukončila vypracovaním bakalárskej práce s názvom: Multivariate cryptography. Pri vypracovaní bakalárskej práce získala praktické skúsenosti z oblasti konečných polí a postkvantovej kryptografie. Vzhľadom k záujmu o štúdium rôznych matematických oblastí a postkvantovej kryptografie bola zvolená za analytika projektu. Vďaka organizačným a komunikačným schopnostiam bola vybraná za vedúceho tohto projektu.

Bc. Andrea Bečárová

Názov pozície: Analytik
Súvisiace predmety: Základy kódovania, Základy kryptografie

Je absolventkou bakalárskeho štúdia na Fakulte elektrotechniky a informatiky Slovenskej Technickej Univerzity v Bratislave, v študijnom programe Aplikovaná informatika, odbor Bezpečnosť informačných systémov. Bakalárske štúdium ukončila vypracovaním bakalárskej práce s názvom: Dekódovací problém. Pri vypracovaní bakalárskej práce získala praktické skúsenosti z oblasti lineárnych kódov a postkvantovej kryptografie založenej na teórii kódovania. Vzhľadom k záujmu o štúdium rôznych matematických oblastí a postkvantovej kryptografie bola zvolená za analytika projektu.

Bc. Michal Jeníkovský

Názov pozície: Main Developer
Súvisiace predmety: Programovacie techniky, Algoritmizácia a programovanie

Je absolventom bakalárskeho štúdia na Fakulte elektrotechniky a informatiky Slovenskej Technickej Univerzity v Bratislave, v študijnom programe Aplikovaná informatika, odbor Modelovanie a simulácia udalostných systémov. Bakalárske štúdium ukončil vypracovaním bakalárskej práce s názvom: Implementácia hardvérového modulu SHA512 pre FPGA. Pri vypracovaní bakalárskej práce získal okrem iného aj skúsenosti s kryptosystémom McEliece. Vďaka skúsenostiam s programovaním veľkých projektov vo firme, v ktorej popri štúdiu naberá praktické vedomosti bol zvolený za hlavného developera projektu.

Bc. Rastislav Pálenik

Pozícia: Developer
Súvisiace predmety: Programovacie techniky, Algoritmizácia a programovanie

Je absolventom bakalárskeho štúdia na Fakulte elektrotechniky a informatiky Slovenskej Technickej Univerzity v Bratislave, v študijnom programe Aplikovaná informatika, odbor Modelovanie a simulácia udalostných systémov. Bakalárske štúdium ukončil vypracovaním bakalárskej práce s názvom: Programová podpora predmetu Paralelné programovanie. Pri vypracovaní bakalárskej práce získal najmä skúsenosti z programovania . Vďaka zapáleniu pre programovanie bol zvolený za developera projektu.

Bc. Peter Kyseľ

Pozicia: Developer
Súvisiace predmety: Programovacie techniky, Algoritmizácia a programovanie

Je absolventom bakalárskeho štúdia na Fakulte elektrotechniky a informatiky Slovenskej Technickej Univerzity v Bratislave, v študijnom programe Aplikovaná informatika, odbor Bezpečnosť informačných systémov. Bakalárske štúdium ukončil vypracovaním bakalárskej práce s názvom: Inteligentné riešenia pre moderné bezpečnostné systémy v aplikáciách riadenia a správy prístupov. Pri vypracovaní bakalárskej práce získal najmä skúsenosti z programovania . Vďaka radosti z programovania bol zvolený za developera projektu.

1.1.4 Motivácia

Blížiaca sa hrozba zostrojenia dostatočne veľkého kvantového počítača a skutočnosť, že celý internet je postavený na šifrovaní, ktoré by bolo po takomto objave prelomené v konečnom čase, motivuje ľudí k štúdiu a analýze postkvantovej kryptografie.

Postkvantová kryptografia je rezistentná voči útokom za pomoci kvantového počítača. Jedna z možných tried takýchto kryptosystémov je založená na teórii kódovania. Dekódovací problém, ktorý vďaka náročnosti riešenia (kategória NP-úplných problém) tvorí základ bezpečnosti takýchto systémov. Jedným z nich je systém McEliece, ktorý sa stal kandidátom na vytvorenie bezpečnostného štandardu. V súčasnej dobe sa venuje pozornosť kryptosystému McEliece s QC-MDPC a QC-LDPC kódmi. Dôvodom je veľkosť kľúčov a dosiahnutá bezpečnosť pri útokoch.

Dňa 9.9.2016 bol zverejnený nový útok na QC-MDPC McElieceov kryptosystém. Tento útok je významným pokrokom v oblasti kryptoanalýzy kryptosystému QC-MDPC McEliece a poukazuje na závažné nedostatky tohto kryptosystému. Tento útok nám

poskytol dôvod k štúdiu QC-LDPC McEliece systému a jeho odolnosti voči takémuto útoku.

1.1.5 Organizácia a riešenie projektu

Zadelenie pozícii v projekte bolo spravené na základe predispozícii a schopností jednotlivých členov. Pre správu a organizáciu úloh, sme sa rozhodli pre použitie kolaboračného nástroja Trello. Po spojení rozvrhov jednotlivých členov sme sa dohodli na pravidelných stretnutiach a to vždy v stredu o 12:00 spolu so zadávateľmi tímového projektu Ing. Viliamom Hromadom, PhD. a Mgr. Tomášom Fabšičom. Zároveň sme sa dohodli na stretnutiach tímu v pondelok o 20:00 v priestoroch Netlabu, kde z každého stretnutia bude vypracovaná zápisnica. Všetky zápisnice spolu s aktuálnymi informáciami o projekte je možné nájsť na webovej stránke tímového projektu.
Hrubý návrh riešenia zatiaľ spočíva v:

- Naštudovaní princípov fungovania QC-LDPC a QC-MDPC McEliece kryptosystému.
- Oboznámení sa s útokom Thomasa Johanssona na QC-MDPC McEliece kryptosystému.
- Oboznámení sa s knižnicou BitPunch.
- Implementácií jednotlivých krokov útoku na QC-LDPC.

1.2 Riadenie projektu a tímová organizácia

Hlavnou úlohou tohto projektu počas zimného semestra 2016/2017 bolo zoznámenie sa s danou problematikou. Keďže sa jedná o náročnú tému, bolo nevyhnutné naštudovanie potrebnej teórie z oblasti kódovania a postkvantovej kryptografie. Následne sme sa zoznámili s knižnicou bitPunch, v ktorej sa nachádza implementácia kryptosystému McEliece.

Na prelome zimného a letného semestra sme prešli na praktickú časť, kde sme implementovali do spomínanej knižnice funkcie, potrebné k vykonaniu experimentov pre podporu útoku na QC-LDPC McEliece. Na záver sme vykonali niekoľko testov ďalej popísaných v tejto dokumentácii.

Komunikácia členov tímu s pedagógmi prebiehala počas pravidelných stretnutí v zimnom semestri vždy v stredu od 12:00 do 13:00 a v letnom semestri pravidelne vo štvrtok od 9:00 do 10:00, zápisnice z týchto stretnutí sú súčasťou príloh. Pre rozdelenie a správu úloh sme používali kolaboračný nástroj Trello, všetky aktuality a stretnutia sme zaznamenávali na našej webovej stránke projektu.

Kapitola 2

McEliece

V tejto kapitole sú popísané základné definície, ktoré sú potrebné pre lepšie pochopenie tejto práce, d'alej popis QC-MDPC a QC-LDPC McELiece systému.

2.1 Základné pojmy

Definícia 2.1.1 *Množina $\mathcal{C} \subset \mathbb{B}^n$ sa nazýva **lineárny kód**, ak je vektorovým podpriestorom priestoru $\mathbb{B}^n$ nad pol'om $\mathbb{B}$.*

[2]
V našom prípade sme pracovali s binárnymi kódmi, a teda $\mathbb{B} = \mathbb{GF}(2)$.

Definícia 2.1.2 *Nech $\mathcal{C}$ je vektorový priestor. Neprázdna množina $\mathcal{D} \subset \mathcal{C}$ sa nazýva **b**áza lineárneho priestoru $\mathcal{C}$ ak je lineárne nezávislá a generuje $\mathcal{C}$. Počet prvkov v l'ubovolnej báze vektorového priestoru sa nazýva **dimenzia** a označujeme ju dim $\mathcal{C}$.*

[2]

Definícia 2.1.3 ***Generujúca matica** lineárneho kódu $\mathcal{C}$ je l'ubovolná matica G_c, ktorej riadky tvoria bázu $\mathcal{C}$. Kontrolná matica H_c je l'ubovolná matica, ktorej stĺpce tvoria bázu duálneho kódu(ortogonálneho doplnku ku kódu $\mathcal{C}$). Potom vieme že $cH^T = 0$, ak $c \in \mathcal{C}$.*

[2]

Definícia 2.1.4 *Ak $a = (a1, ..., a_m) \in \mathbb{B}^m$, tak počet jednotiek v tomto vektore budeme nazývat' **Hammingova váha** m-tice a a budeme ju označovat' $hwt(a)$.*

[2]

Definícia 2.1.5 *Nech $w, v \in \mathbb{B}^n$. **Hammingova vzdialenost'** slov w a v nazývame číslo $d(w, v) = hwt(w + v)$. Hammingova vzdialenost' udáva počet miest, na ktorých sa n-tice w a v líšia.*

[2]

Definícia 2.1.6 *Nech $\mathcal{C} \subset \mathbb{B}^n$ je lineárny kód. Potom* ***minimálna vzdialenosť*** *lineárneho kódu $\mathcal{C}$, je najmenšia Hammingova váha nenulového kódového slova, t.j. $d = min\{hwt(v) \mid v \in \mathcal{C},\ v \neq 0\}$.*

[2]

Definícia 2.1.7 *Nech $\mathcal{C} \subset \mathbb{B}^n$ je lineárny kód s kontrolnou maticou $\boldsymbol{H}_{\mathcal{C}}$ a nech $s \in \mathbb{B}^n$. Pre každé slovo w definujeme* ***syndrómom slova*** *s ako $s_w = w \times {H_c}^T$.*

[2]

Definícia 2.1.8 ***Kvázi-cyklický*** *(QC) kód $\mathcal{C}$ je taký lineárny kód, že pre $n_0 \in \mathbb{N}$ je každý cyklický posun kódového slova o n_0 opäť kódovým slovom, je teda uzavretý vzhľadom k cyklickému posunu.*

[2]

Definícia 2.1.9 ***LDPC/ MDPC*** *kód je lineárny samooopravný kód, používajúci riedke kontrolné matice. LDPC a MDPC kódy sa líšia len vo váhe riadkov, keďže LDPC kódy sú riedke(s váhami menej ako 10). Ak sú tieto kódy zároveň kvázicyklické nazývame ich QC-LDPC a QC-MDPC kódy.*

[3]

Poznámka 1 ***Generujúca a kontrolná matica cyklického kódu*** *je reprezentovaná len jedným riadkom(vektorom) z tejto matice, keďže zvyšné riadky sú len cyklickým posunom*

2.2 McEliece

McEliece je asymetrický kryptosystém s verejným kľúčom, založený na dekódovacom probléme, jeho súkromným kľúčom býva generujúca matica a verejným kľúčom transformácia generujúcej matice. Bezpečnosť tohto systému je založená na dekódovaní dostatočne veľkého lineárneho kódu, bez viditeľnej algebraickej štruktúry, čo je vo všeobecnosti považované za NP-úplný problém. V pôvodnom návrhu sa používali Goppa kódy, no tieto systémy boli prakticky nepoužiteľné, kvôli veľkým verejným kľúčom a nízkej prenosovej rýchlosti.

Základom tohto kryptosystému je dekódovací problém, kde ide o hľadanie chybového vektora prenášaného slova, kde odčítaním tohto vektora od prijatého slova získame slovo odoslané. Tento problém sa používa v našom McEliece kryptosystéme. [1]

2.2.1 Popis algoritmu s Goppa kódmi

[6]Majme **G** generujúcu maticu goppovho kódu $\mathcal{C}$ o veľkosti $k \times n$, ktorý je schopný opravit' najviac **t** chýb. Ďalej nech **S** je ľubovolná regulárna binárna matica o veľkosti $k \times k$ a **P** je permutačná matica, ktorá vznikla permutáciou riadkov a stĺpcov jednotkovej matice, typu $n \times n$.

Verejným kľúčom je matica **G**', ktorá vznikne zamaskovaním matice **G**, tak že $G' = S.G.P$. Súkromným kľúčom sú matice **S, G** a **P**.

1. Ak chcú spolu Alica a Bob komunikovat', Alica najprv vygeneruje verejný kľúč **G**' a zverejní ho spolu s informáciou o počte opraviteľných chýb **t**.
2. Následne Bob zašifruje svoju správu ako: $\mathrm{x} = \mathrm{u.G'} + \mathrm{e}$, kde **e** je náhodne vygenerovaný vektor s váhou $\mathrm{w_H} \leq \mathrm{t}$
3. Alica obdrží Bobovu zašifrovanú správu **x** vynásobí ju maticou P^{-1}
   $\mathrm{x.P}^{-1} = (\mathrm{u.S.G.P} + \mathrm{e}).\mathrm{P}^{-1} = (\mathrm{u.S}).\mathrm{G} + \mathrm{e.P}^{-1}$
4. Zvoleným dekódovacím algoritmom odstráni šum $\mathrm{e.P}^{-1}$
5. Vypočíta $(\mathrm{u.S}).\mathrm{S}^{-1} = \mathrm{u}$

Majme transponovanú kontrolnú maticu H^{T} a syndróm $\mathrm{s} = \mathrm{xH}^{\mathrm{T}}$, kde t je maximálny počet chýb, ktorý je kód schopný opravit' a z chybový vektor. Potom je bezpečnost' systému založená na riešení úlohy, nájst' také riešenie z rovnice, kde $\mathrm{s} = \mathrm{xH}^{\mathrm{T}} = (\mathrm{y} + \mathrm{z})\mathrm{H}^{\mathrm{T}} = \mathrm{zH}^{\mathrm{T}}$, kde y je kódové slovo, ktoré má minimálnu váhu a to je NP-úplný problém.

2.2.2 Popis algoritmu s použitím QC-MDPC

[3]Tento návrh McElieceovho kryptosystému narozdiel od pôvodného nepoužíva matice Q a S, no používa kódy, ktoré dovoľujú iteratívne dekódovanie. Pre zachovanie bezpečnosti sú váhy pre kontrolu parity väčšie ako obvykle v takýchto LDPC kódoch a nazývajú sa MDPC kódy (Moderate Density Parity Check). Kvázi cyklická varianta týchto kódov umožňuje reprezentáciu kódu pomocou jedného riadka generujúcej matice. Keďže generujúca matica je verejným kľúčom, tým pádom niekoľkonásobne zmenšíme veľkosť verejného kľúča.

Generovanie verejného kľúča

1. Nech p = n − k, zvolíme si QC-MDPC kód s parametrami [n, n − p], náhodne si zvolíme kontrolnú maticu H $\in \mathbb{F}_2^{p \times n}$, kde n = n_0 × p, tak aby
   $H = (H_0 H_1 \cdots H_{n_0-1})$
   Pričom každý jeden z blokov H_i je cyklická matica $p \times p$ s váhou w_i.

2. Vygenerujeme si verejný kľúč G $\in \mathbb{F}_2^{(n-p)\times n}$ z matice H nasledovne: $G = (IP)$,

$$\text{kde } P = \begin{pmatrix} P_0 \\ P_1 \\ \vdots \\ P_{n_0-2} \end{pmatrix} = \begin{pmatrix} (H^{-1}{}_{n_0-1} H_0)^T \\ (H^{-1}{}_{n_0-1} H_1)^T \\ \vdots \\ (H^{-1}{}_{n_0-1} H_{n_0-2})^T \end{pmatrix}$$

a I je jednotková matica, ktorá spolu s P dáva verejný kľúč G.

Šifrovanie

K zašifrovaniu správy $m \in \mathbb{F}_2^k$ si vygenerujeme náhodný chybový vektor $e \in \mathbb{F}_2^k$ s váhou najviac t a vypočítame $x = m.G + e$. Vypočítané x je zašifrovaná správa.

Dešifrovanie

Pomocou iteratívneho dekódovacieho algoritmu, pri zachovaní maximálneho počtu chýb z prijatej správy, dostaneme $m.G$. Keďže G je v systematickej forme dostaneme správu m z prvých k pozícií $m.G$

2.2.3 Popis algoritmu s použitím QC-LDPC

[5]Tento návrh McElieceovho kryptosystému používa kvázicyklické lineárne kódy, kde kvázicyklickosť zabezpečuje zmenšenie verejného kľúča. Rozdiel medzi QC-MDPC systémom je v použití riedkej kontrolnej matice.

Generovanie kľúča

1. Náhodne si zvolíme kód s parametrami (n_0, d_v, p).
2. Zvolíme si $\mathrm{H} = \mathrm{H}_0, \mathrm{H}_1, \cdots, \mathrm{H}_{\mathrm{n_0}-1}$, pričom každý z blokov $\mathrm{H_i}$ je binárny cirkulantný blok o veľkosti p s váhou riadkov $\mathrm{d_v}$.
3. Pomocou kontrolnej matice si vygenerujeme generujúcu maticu G, tak aby

   $$G = IP \text{ , kde } P = \begin{pmatrix} {H^{-1}}_{n_0-1}H_0)^T \\ {H^{-1}}_{n_0-1}H_1)^T \\ \vdots \\ {H^{-1}}_{n_0-1}H_{n_0-2})^T \end{pmatrix}$$

   Súčast'ou generujúcej matice G je jednotková matica $I_{[k\times k]}$, kde $k = (n_0 - 1) * p$.
4. Následne si vygenerujeme binárnu cirkulantnú invertibilnú maticu S, tak že matica S pozostáva z $(n_0 \times n_0)$ blokov $p \times p$ cirkulantných matíc
5. Ďalej si vygenerujeme a invertibilnú maticu Q pozostávajúcu z $(n_0 \times n_0)$ blokov $p \times p$ cirkulujúcich matíc
6. Vygenerujeme si verejný kľúč: $G' = S^{-1}.G.Q^{-1}$, keďže bloky S a Q sú cirkulantné, matica G si zachováva kvázicyklickú formu.

Súkromným kľúčom sú matice H, S a Q a verejným kľúčom je matica G'.

Šifrovanie

K zašifrovaniu správy m si vygenerujeme náhodný chybový vektor e s váhou najviac t a vypočítame $x = m.G' + e$. Vypočítané x je zašifrovaná správa.

Dešifrovanie

Pre odšifrovanie zašifrovanej správy x si vypočítame $x' = x.Q = u.S^{-1}.G + e.Q$, keďže $x' = u.S^{-1}.G + e.Q$ vieme, že x' je kódové slovo z kódu generovaného maticou G pozmenené o chybový vektor $e.Q$. Pomocou matice H a použitím vhodného dekódovacieho algoritmu pre LDPC kód a následného vynásobenia s maticou S, vieme danú správu dešifrovat'.

Kapitola 3

Útok

3.1 Útok na MDPC

V tejto časti si priblížime útok na QC-MDPC McElieceov kryptosystém. Tento útok je významným pokrokom v oblasti kryptoanalýzy QC-MDPC McElieceovho kryptosystému a poukazuje na závažné nedostatky tohto kryptosystému. Útok bol vykonaný skupinou výskumníkov okolo Thomasa Johanssona zo Švédska.[4]

3.1.1 Popis

Základný scenár útoku je nasledovný. Alica nepretržite posiela správy Bobovi s použítím QC-MDPC systému a Bobovho verejného kľúča. Príležitostne nastane dekódovacia chyba, ktorá sa prejaví odlišnou reakciou Boba. Alica zdeteguje nastatie dekódovacej chyby a túto informáciu si uloží. Po viacnásobnom opakovaní tohto procesu, bude Alica schopná spätne zrekonštruovať Bobov súkromný kľúč pomocou tohto útoku.

Pokiaľ ide o definíciu bezpečnostného modelu, tento útok sa nazýva *reakčný útok*.

Reakčný útok je model útoku, ktorý vyžaduje iba reakciu dešifrovacieho zariadenia (či nastala dešifrovacia chyba) a nie výsledok dešifrovania.

Predpokladáme, že pomer kódu je $R = k/n = 1/2$, čo zodpovedá $n_0 = 2$. A nech $w_0 = w_1 = w$. Útoky s inými parametrami by boli obdobné.

Útok spätnou rekonštrukciou kľúča na QC-MDPC bol dosiahnutý nájdením tajnej matice H_0 pri znalosti matice verejného kľúča P. So znalosťou H_0 a P, je možné ľahko získať zvyšnú časť H, použítím jednoduchej lineárnej algebry. Ak hovoríme o cyklickej matici, rekonštrukcia H_0 je ekvivalentná s odhalením vektora, zodpovedajúceho prvému riadku tejto matice, označeného h_0.

Kľúčovou myšlienkou bolo preverenie procesu dekódovania pre rôzne chybové vzory. Predovšetkým je pre nás zaujímavé, ak Alica vyskúša chybové vzory zo špeciálnej podmnožiny. Nech Ψ_d je množina všetkých binárnych vektorov dĺžky $n = 2r$, ktorá má práve t jednotiek, a kde všetkých t jednotiek je umiestnených v prvej polovici vektora v náhodných $\frac{t}{2}$ pároch, pričom vzdialenosť medzi týmito dvoma jednotkami, ktoré

tvoria pár, je práve d. Druhá polovica vektora sú samé nuly.

Formálne, vyberieme z množiny Ψ_d, ktorá zaručuje opakujúce sa jednotky vo vzdialenosti d aspoň $\frac{t}{2}$-krát, kde

$$\Psi_d = \{v = (e, f) \mid w_H(f) = 0, \text{ a } \exists \text{ rôzne } s_1, s_2, ..., s_t, \text{ také, že } e_{s_t} = 1 \text{ a } s_{2i} = (s_{2i-1} + d) \mod r \text{ for } i = 1, ..., \tfrac{t}{2}\}.$$

Alica teraz pošle M správ Bobovi, použitím QC-MDPC a pridaním chybových vektorov s váhou t vybraných z množiny Ψ_d. Keď nastane pri dekódovaní chyba u Boba, ona si to zaznamená a po M správach bude schopná empiricky vypočítať pravdepodobnosť chyby dekódovania pre podmnožinu Ψ_d. Naviac to bude schopná urobiť pre $d = 1, 2, ..., U$ pre niektoré vhodné horné hranice U.

Jedným z dôsledkov tohto útoku je zistenie silnej korelácie medzi pravdepodobnosťou chyby dekódovania pre chybové vektory z Ψ_d a existenciou vzdialenosti d medzi dvoma jednotkami v H_0.

Takže po poslaní $M \times U$ správ, si všimneme pravdepodobnosť chyby dekódovania pre každé Ψ_d a klasifikujeme každé d, $d = 1, 2, ..., U$ ako "neexistuje v h_0" (nazývaný tiež CASE-0) alebo "existuje v h_0" (nazývaný CASE-1). Toto nám určuje dištančné spektrum pre H_0, označované $D(h_0)$. Je dané

$$D(h_0) = \{d : 1 \leq d \leq U, d \text{ klasifikované ako "existuje v } h_0\text{" }\}.$$

Taktiež, keďže vzdialenosť d sa môže objaviť viackrát v dištančnom spektre pre daný bitový vzor c, budeme rozlišovať násobnosť d v c, a budeme ho označovať $\mu_c(d)$, alebo jednoducho $\mu(d)$, ak je c jasne definované v kontexte.
Napríklad, pre bitový vzor $c = 0011001$, máme $U = 3$ a

$$D(c) = \{1, 3\},$$

s násobnosťami vzdialeností $\mu(1) = 1, \mu(2) = 0, \mu(3) = 2$.

Postup pre výpočet dištančného spektra je uvedený v Algoritme 1.

Algorithm 1: Výpočet dištančného spektra

Vstup: parametre n, r, w a t základného QC-MDPC kódu, počet dekódovacích pokusov M pre jednu vzdialenosť
Výstup: dištančné spektrum $D(h_0)$
  for all vzdialenosti d **do**
    skús M dekódovacích pokusov použitím navrhnutého chybového vektora;
    vykonaj štatistický test na určenie násobnosti $\mu(d)$;
    if $\mu(d) > 0$ **then**
      pridaj d s násobnosťou $\mu(d)$ do dištančného spektra $D(h_0)$;
    end if
  end for

Posledným krokom zostáva vykonať rekonštrukciu h_0 z už známeho dištančného spektra $D(h_0)$. To je možné vykonať pomocou iteratívnej procedúry. Začneme s umiestnením prvých dvoch jednotiek na dĺžku vektora i_0 na pozíciu 0 a i_0, kde i_0 je najmenšia hodnota v $D(h_0)$. Ďalej umiestníme tretiu jednotku na pozíciu a testujeme, či všetky vzdialenosti, ktoré sa momentálne vo vektore nachádzajú, patria do dištančného spektra. Ak nie, skúšame ďalšiu pozíciu pre tretí bit. Ak áno, prejdeme na umiestňovanie štvrtého bitu a testovanie jej vzdialenosti od predošlých troch jednotiek, atď. Po rekonštrukcii, máme vyskladané h_0. Proces rekonštrukcie je detailne popísaný v Algoritme 2.

Algorithm 2: Rekonštrukcia kľúča pomocou dištančného spektra

Vstup: dištančné spektrum $D(h_0)$, čiastočný súkromný kľúč h_0, aktuálna(current) hĺbka l.

Výstup: zrekonštruovaný súkromný kľúč h_0 alebo správa "Neexistuje žiaden príslušný súkromný kľúč".

Initial recursion parameters: dištančné spektrum $D(h_0)$, prázdna množina pre súkromný kľúč, aktuálna hĺbka 0;

if $l = 1$ **then**
  return "Neexistuje žiaden príslušný súkromný kľúč";
end if
if $l = w$ **then**
  return h_0 `"/*nájdený súkromný kľúč*/"`;
end if
for all potenciálne kľúčové bity i **do**
  if všetky vzdialenosti ku kľúčovému bitu i sa nachádzajú v $D(h_0)$ **then**
    pridaj kľúčový bit i do súkromného kľúča h_0;
    sprav rekurzívne volanie s parametrami $D(h_0)$, h_0 a $l + 1$;
  end if
end for
odstráň posledný kľúčový bit i zo súkromného kľúča h_0
sprav rekurzívne volanie s parametrami $D(h_0)$, h_0 a $l - 1$;

Proces rekonštrukcie môže nájsť nejaké kľúčové vzory h_0 s rovnakým dištančným spektrom $D(h_0)$ ako h_0. Algoritmus to v tomto prípade zamietne a rekurzívne vyskúša ďalšie kľúčové vzory, ktoré poskytuje proces rozsiahleho vyhľadávania.

3.2 Útok na LDPC

V tejto časti si bližšie popíšeme nami realizovaný útok na QC-LDPC McElieceov kryptosystém. Tento útok je inšpirovaný Thomasom Johanssonom a jeho tímom, ktorí v roku 2016 vykonali útok na QC-MDPC McElieceov kryptosystém. Tento útok bol podnetom pre Mgr. Tomáša Fabšiča, ktorý v súčasnosti spolupracuje s Thomasom Johanssonom na vykonaní podobného útoku na QC-LDPC McEliece. Cieľom nášho projektu je vykonať počítačové experimenty užitočné pre takýto útok na QC-LDPC McEliece. V tejto časti si taktiež zhrnieme motiváciu k vykonaniu takýchto experimentov a načrtneme, čo by nám mohli tieto experimenty priniesť a ako by sme ich výsledky mohli následne ďalej využiť.

3.2.1 Parametre

Majme nasledovné parametre systému:
- $n_0 = 3$
- $p = 8192$
- Každá matica H_i bude mať v riadku 13 jednotiek.
- $m = 11$
- $t' = t/m = 48$

kde n_0 je počet cirkulantných blokov matice H, p je veľkosť týchto cirkulantných blokov (teda H_i je matica o veľkosti $p \times p$), m je konštantná riadková a stĺpcová váha matice Q a t je počet chýb, ktoré dokáže takýto LDPC kód opraviť.

Ďalej vyžadujeme, aby každý blok matice S bol hustý. Q je skonštruované permutovaním riadkových blokov a stĺpcových blokov kvázicyklickej matice, ktoré majú takú vlastnosť, že bloky na diagonále majú riadky s váhou 3 a bloky mimo diagonály majú riadky s váhou 4.

3.2.2 Útok na H - motivácia

Útok na H je možný aj bez toho, aby sme poznali maticu Q. Vieme totiž, že je riedka a kvázicyklická. Predpokladajme, že e je konštruované nasledovne.

1. dĺžka e je 3×8192
2. e obsahuje 13 vzorov "11". Tieto "11"-vzory sú umiestnené nasledovne.
3. Pre každý "11"-vzor zvolíme jeden z troch 8192-bitových blokov v e rovnomerne náhodne.
4. Ďalej priradíme každému "11"-vzoru pozíciu v rámci svojho bloku, taktiež rovnomerne náhodne. Vyberáme len z pozícií, ktoré neboli predtým priradené.
5. Po umiestnení všetkých "11"-vzorov, umiestnime jednu 1 na jednu z neobsadených pozícií. Túto pozíciu vyberieme rovnomerne náhodne.

6. Na všetky neobsadené pozície na záver doplníme nuly.

O súčine $e \cdot Q$ môžme uvažovať ako o pridaní vybraných riadkov Q. Riadky, ktoré budú pridané sú vyberané podľa pozícií jednotiek vo vektore e. Ak e obsahuje vzor "11" vo svojom prvom 8192-bitovom bloku, bude to mať za následok pridanie dvoch po sebe nasledujúcich riadkov z prvých 8192 riadkov Q. Vzhľadom k tomu, že Q je kvázicyklická a váha jej riadkov je 11, potom sa nám vytvorí pätnásť "11"-vzorov v $e \cdot Q$. Keďže Q je riedka, predpokladáme, že pridanie ďalších riadkov Q, nebude prekážať už pridaným "11"-vzorom. Takže očakávame $13 \times 11 = 143$ "11"-vzorov v $e \cdot Q$.

Podobne, ak budeme konštruovať e tak, aby obsahoval 13 "101"-vzorov, namiesto 13 "11"-vzorov, budeme očakávať, že $e \cdot Q$ bude obsahovať 143 "101"-vzorov, atď.

Teda, pre každé k vieme skonštruovať e tak, že budeme očakávať 143 párov jednotiek v $e \cdot Q$ s tým, že každé dve jednotky v páre budú oddelené práve k nulami.

Pomocou posielania takýchto vektorov e Bobovi a pozorovania pravdepodobnosti chyby dekódovania, môžeme zistiť, ktoré vzdialenosti sa nachádzajú v dištančnom spektre H. Rozdiel od útoku Thomasa Johanssona spočíva v tom, že teraz zisťujeme dištančné spektrum pre celé H, nie pre blok H_i. Takže naše dištančné spektrum bude obsahovať vzdialenosti prislúchajúce k rôznym blokom H_i a nemôžeme s istotou povedať, ku ktorému bloku daná vzdialenosť patrí.

Každý blok H_i obsahuje 13 jednotiek v riadku a H obsahuje 3 takéto bloky. Tak získame multimnožinu obsahujúcu $3 \times \binom{13}{2} = 234$ vzdialeností. (Niektoré vzdialenosti sa môžu v multimnožine nachádzať viac ako raz.)

Môžeme získať viac informácií o pozíciach v H. Získame riadok r_1 spustením Algoritmu 2 s dištančným spektrom ako úplnou multimnožinou obsahujúcou 234 vzdialeností a s $w = 13$. Algoritmus môže nájsť viac ako jedného kandidáta na r_1. Pre každého kandidáta, ktorého algoritmus našiel, spustíme algoritmus znova, teraz s dištančným spektrom obsahujúcim iba vzdialenosti, ktoré neboli použité pri vytváraní kandidáta pre r_1. Takže tentokrát bude Algoritmus 2 bežať s dištančným spektrom s veľkosťou $234 - \binom{13}{2}$ a $w = 13$. Ak tento algoritmus vráti výstup, označíme ho ako r_2. Opäť nám algoritmus môže vrátiť viac kandidátov na r_2. Postupujeme analogicky: Pre každého kandidáta na r_2 spúšťame Algoritmus 2 s dištančným spektrom s veľkosťou $234 - \binom{13}{2} - \binom{13}{2}$ a $w = 13$. Dostaneme kandidátov na r_3.

Pre každý r_i vieme, že predstavuje riadok v bloku H_j v matici H. Tiež vieme, že každý r_i predstavuje riadok v inom bloku. Ale nevieme presne, ktorému bloku vektor r_i prislúcha a na ktorý riadok v tomto bloku patrí.

3.2.3 Experimenty na H

Nami vykonané experimenty sú zhrnuté v Algoritme 3. Výsledkami experimentov by mala byť prehľadná tabuľka, ktorá ku každej vzdialenosti d priradí pravdepodobnosť chyby dekódovania, teda pravdepodobnosť, s ktorou pri našich experimentoch zlyhal nami použitý dekódovací algoritmus a nami posielaná správa sa teda nepodarila dekódovať.

Algorithm 3: Experimenty na H

Vstup: inštancia QC-LDPC McELiece s parametrami z kapitoly 3.3.2.
Výstup: tabuľka pravdepodobností pre vybrané vzdialenosti d.

zober prvý riadok matice H a rozdeľ ho na tri bloky po 8192 bitov;
for all h_i **do**
  zisti vzdialenosti medzi jednotkami vrámci bloku aj s ich násobnosťami;
end for
for all riadky blokov matice $Q : Q_j$ **do**
  zober prvý riadok Q_j a rozdeľ ho na tri bloky po 8192 bitov;
  for all q_i **do**
    zisti vzdialenosti medzi každými dvoma jednotkami vrámci bloku aj s ich násobnosťami;
  end for
end for
for $m = 1$ **to** $m \leq 10$ **do**
  zober d také, ktoré sa v H nachádza práve raz a v Q ani raz;
  for $i = 1$ **to** $i \leq 1000$ **do**
    v_i inicializuj ako nulový vektor dĺžky $3 \cdot 8192$;
    for $j = 1$ **to** $j \leq 24$ **do**
      vyber číslo $k \in 0, 1, 2$ rovnomerne náhodne;
      vyber číslo $l \in 0, 1, ..., 8191$ rovnomerne náhodne;
      vygeneruj vektor w dĺžky $3 \cdot 8192$, ktorý má jednotky na pozíciách $k \cdot 8192 + l$ a $k \cdot 8192 + (l + d \mod 8192)$ a inde samé nuly;
      $v_i \leftarrow v_i + w$
    end for
    dešifruj vektor v_i a poznač si, či nastala chyba dekódovania;
  end for
  odhadni pravdepodobnosť, že nastane chyba dekódovania pre takto zvolené vektory v_i;
end for
for $m = 1$ **to** $m \leq 10$ **do**
  zober d také, ktoré sa v H ani v Q nenachádza ani raz;
  for $i = 1$ **to** $i \leq 1000$ **do**
    v_i inicializuj ako nulový vektor dĺžky $3 \cdot 8192$;
    for $j = 1$ **to** $j \leq 24$ **do**
      vyber číslo $k \in 0, 1, 2$ rovnomerne náhodne;
      vyber číslo $l \in 0, 1, ..., 8191$ rovnomerne náhodne;
      vygeneruj vektor w dĺžky $3 \cdot 8192$, ktorý má jednotky na pozíciach $k \cdot 8192 + l$ a $k \cdot 8192 + (l + d \mod 8192)$ a inde samé nuly;
      $v_i \leftarrow v_i + w$
    end for
    dešifruj vektor v_i a poznač si, či nastala chyba dekódovania;
  end for
  odhadni pravdepodobnosť, že nastane chyba dekódovania pre takto zvolené vektory v_i;
end for

3.2.4 Útok na Q - motivácia

Nech Ψ_d je množina všetkých binárnych vektorov dĺžky $n = 3 \times 8192$ s práve 48 jednotkami, pričom je týchto 48 jednotiek umiestnených v náhodných pároch vo vzdialenosti d v prvej tretine vektora.

Nech $e \in \Psi_d$. Znova chápme súčin $e \cdot Q$ ako pridanie vybraných riadkov matice Q. To, ktoré riadky Q budú pridané, je určené pozíciami jednotiek v e. Pokiaľ je Q riedka matica, budeme normálne očakávať, že súčin $e \cdot Q$ bude obsahovať práve $hwt(e) \times m$ jednotiek. To však nebude platiť v prípade, že prvý riadok Q obsahuje jeden pár jednotiek takých, že obe jednotky z tohto páru, patria do rovnakého bloku a vzdialenosť medzi nimi je d. V takomto prípade by nastalo 24 vykrátení pri tomto násobení (jedno krátenie pre každý pár jednotiek v e, ktoré sú od seba vo vzdialenosti d). Očakávaný počet jednotiek v $e \cdot Q$ bude teda $hwt(e) \times m - 24$. Následný proces opravy chýb bude teda musieť opraviť o 24 chýb menej ako v normálnom prípade. Preto je možné očakávať výrazne nižšiu pravdepodobnosť chyby dekódovania.

Podobne, ak prvý riadok Q obsahuje k párov jednotiek tak, že pre každý pár sú obe jednotky v tom istom bloku a vzdialenosť medzi nimi je d, potom očakávame $k \times 24$ vykrátení v súčine $e \cdot Q$ a očakávaný počet jednotiek v $e \cdot Q$ bude $hwt(e) \times m - k \times 24$. Táto analýza je platná vždy, za predpokladu, že Q je riedka, a to bez ohľadu na to, či v zvolenej variante väčšina blokov matice Q obsahuje len 2 jednotky v riadku.

Preto môžeme očakávať, že odhadom pravdepodobností chyby dekódovania, dokážeme určiť dištančné spektrum prvého riadku Q. Takže sme v tej istej situácií, ktorá nastala v prípade útoku na H: poznáme vzdialenosti, ale nevieme, do ktorých blokov patria a na ktorú pozíciu v bloku patria. Rovnako ako v prípade matice H, môžeme použiť Algoritmus 2 na rekonštrukciu riadkov $r_1, ..., r_3$.

3.2.5 Experimenty na Q

Po vykonaní experimentov na H, sme postupovali podobne aj pri matici Q. Priebeh týchto experimentov je zachytený v Algoritme 4. Očakávame, že výsledkami experimentov teda budú užitočné informácie o vybraných vzdialenostiach d.

Algorithm 4: Experimenty na Q

Vstup: inštancia QC-LDPC McELiece s parametrami z kapitoly 3.3.2.
Výstup: tabuľka pravdepodobností pre vybrané vzdialenosti d.

```
for m = 1 to m ≤ 10 do
    zober d také, ktoré sa v Q nachádza práve raz a v H ani raz;
    for i = 1 to i ≤ 1000 do
        v_i inicializuj ako nulový vektor dĺžky 3 · 8192;
        for j = 1 to j ≤ 24 do
            vyber číslo l ∈ 0, 1, ..., 8191 rovnomerne náhodne;
            vygeneruj vektor w dĺžky 3 · 8192, ktorý má jednotky na pozíciach l a l + d
            mod 8192 a inde samé nuly;
            v_i ← v_i + w
        end for
        dešifruj vektor v_i a poznač si, či nastala chyba dekódovania;
    end for
    odhadni pravdepodobnosť, že nastane chyba dekódovania pre takto zvolené
    vektory v_i;
end for
for m = 1 to m ≤ 10 do
    zober d také, ktoré sa v H ani v Q nenachádza ani raz;
    for i = 1 to i ≤ 1000 do
        v_i inicializuj ako nulový vektor dĺžky 3 · 8192;
        for j = 1 to j ≤ 24 do
            vyber číslo l ∈ 0, 1, ..., 8191 rovnomerne náhodne;
            vygeneruj vektor w dĺžky 3 · 8192, ktorý má jednotky na pozíciach l a l + d
            mod 8192 a inde samé nuly;
            v_i ← v_i + w
        end for
        dešifruj vektor v_i a poznač si, či nastala chyba dekódovania;
    end for
    odhadni pravdepodobnosť, že nastane chyba dekódovania pre takto zvolené
    vektory v_i;
end for
```

Kapitola 4

Praktická časť

Praktická časť tohto tímového projektu v prvom rade zahrňovala analýzu kódu BitPunch. Po analýze sme podľa nášho útoku postupne doplnili vlastnú implementáciu kódu. Našu implementáciu, teda praktickú časť, si vieme rozdeliť na štyri časti - BitPunch, ukladanie kľúčov, dištančné spektrum a generovanie chybového vektora.

4.1 BitPunch

BitPunch je knižnica, ktorá slúži na pracovanie s postkvantovou kryptografiou. Obsahuje objekty a funkcie pomocou ktorých môžeme kódovať, dekódovať, šifrovať či dešifrovať. Všetky tieto funkcionality je možné zároveň aplikovať na Goppa, QC-LDPC a QC-MDPC kódy. Taktiež disponuje s rôznymi funkciami na prácu kvázi-cyklických matíc Ako príklad chcem spomenúť ich ukladanie, pretože ako z mnoha funkcií nám to výrazne pomohlo pri získaní relevantných výsledkov. Ukladať kvázi-cyklické matice je možné dvoma spôsobmi. Prvý spôsob je bežný, kedy sa ukladá prvý riadok matice a zvyšok sa uloží dopočítaním a to posunom doprava o hodnotu rovnú poradového čísla riadku. Druhý spôsob je ukladanie takzvaných *Sparse* matíc. Vtedy sa uložia len pozície jednotiek z prvého riadku matice. Presnejšie *Sparse* ukladanie sme využili pri našich testoch. Celkovo BitPunch knižnica nám výrazne pomohla pri práci s QC-LDPC kódmi a je výsledkom niekoľko bakalárskych a diplomových prác.

4.2 Ukladanie kľúčov

Aby naše výsledky boli relevantné, potrebovali sme si ukladať kľúče (súkromný, verejný), aby sme mohli robiť experimenty s rovnakými kľúčmi. Na ukladanie súkromného kľúča sme si potrebovali uložiť nie je len samotné matice S, Q a H ale aj patričné premenné - $\_m$, $\_w$ a $\_n0$. Pred ukladaním sme štruktúru matíc nepotrebovali nijako upravovať ani nejakým spôsobom meniť, ukladali sme tzv. *raw data*.
Ukladanie verejného kľúča sme implementovali obdobne. Taktiež pri ukladaní sme

Premenné	Popis	Hodnota
_m	veľkosť bloku	8192
_w	váha	39
_n0	počet blokov	3

Table 4.1: Premenné

nepotrebovali meniť štruktúru matíc. Potrebovali sme si uložiť maticu G maskovanú a premenné $\_m$ a $\_w$. K všetkým týmto maticiam a premenným sme pristupovali cez štruktúru $BPU\_T\_Mecs\_Ctx$. Taktiež sme priamo do tejto štruktúry ukladali načítané matice a premenné zo súboru. Čo predstavujú premenné a aké hodnoty sme používali je uvedené v tabuľke .

4.3 Dištančné spektrum

Keďže chceme útočiť na matice H a Q, tak si potrebujeme vypočítať dve dištančné spektrá. Dištančné spektrum vypočítané z matice H a dištančné spektrum vypočítané z matice Q. Ná výpočet dištančného spektra sme museli získať prvé riadky z blokov matice, keďže matice sú kvázi-cyklické. To nám zabezpečila funkcia z BitPunch $BPU\_gf2SparseQcMatrixGetRow()$. Vstupné argumenty tejto funkcie sú *matrix*, *row* a index riadku matice v danom bloku. Do argumentu *row* sa nám uložil riadok z daného bloku. Tento riadok je ale špeciálny a to vďaka tomu, že obsahuje pozície jednotiek z daného riadku. Následne sme implementovali funkciu $calculateRowDistanceSpectrumFromSparsePoly()$, ktorej vstupné argumenty sú smerník na vopred vytvorený objekt dištančného spektra, ktorý sme naplnili vypočítanými vzdialenosti a riadok matice - *row*, z ktorého sme dané vzdialenosti vypočítali.
V prvom kroku sme zabezpečili získanie všetkých možných vzdialeností. Prvú hodnotu pozície z *row* sme začali odpočítavať od všetkých ďalších, potom druhú hodnotu z *row* sme začali odpočítavať od všetkých ďalších a tak ďalej. Ďalší krok znamenal skontrolovať, či vypočítaná vzdialenosť d je menšia alebo rovná ako je polovica veľkosti bloku. To sme zabezpečili jednoduchým výpočtom: $d = \_min(d, 8190 - d)$. Naše spektrum predstavuje pole integerov o veľkosti polovice 8192. Na začiatku sú všetky hodnoty nastavené na 0 . Po výpočte d inkrementujeme hodnotu o 1 na pozícii d v spektre a tým získame početnosť vzdialeností - $spektrum[d]{+}{+}$. Po skončení tohto algoritmu sme úspešne získali dištančné spektrum aj s početnosťami vzdialeností.

```
for (unsigned int i = 0; i < row->weight-1; i++) {
        for (unsigned int j = i + 1; j < row->weight; j++) {
                int d = row->index[j] - row->index[i] - 1;
                d = __min(d, 8190 - d);
                spectrum[d]++;
        }
```

}

Našou úlohou tímového projektu bolo získať pravdepodobnosti zlyhania dekódovania v rôznych prípadoch. Nasledujúcu implementáciu kódu zaradím ako podsekcie dištančného spektra, keďže sme s ním museli priamo pracovať. Vytvorili sme si tri funkcie - $getD\_notInQ\_notInH()$, $getD\_notInQ\_oneInH()$ a $getD\_oneInQ\_notInH()$.

4.3.1 Vzdialenosť ani raz v matici Q a ani raz v H

Potrebovali sme získať pravdepodobnosť zlyhania dekódovania v situácii, kedy vzdialenosť nepatrila do dištančného spektra z matice Q a ani do dištančného spektra z matice H. Na získanie takejto vzdialenosti sme implementovali funkciu $getD\_notInQ\_notInH()$. Stačilo si zadefinovať hodnotu začínajúc od 0 a v cykle ju stále inkrementovať o 1. V cykle sme kontrolovali, či inkrementovaná hodnota sa nachádza ako vzdialenosť v spektrách z matice Q a H. Kontrola spočívala v tom, či na danej pozícii, čo predstavovala hodnota inkrementovanej premennej sa nachádza 0. Ak áno, vzdialenosť mala nulovú početnosť, čo znamená že daná vzdialenosť sa nenachádza v dištančnom spektre matice.

4.3.2 Vzdialenosť ani raz v matici Q a raz v H

V tomto prípade sme potrebovali získať vzdialenosť, ktorej početnosť v dištančnom spektre z matice H bola práve 1 a 0 v dištančnom spektre z matice Q. Na túto funkcionalitu sme si implmentovali funkciu $getD\_notInQ\_oneInH()$. V cykle sme inkrementovali potrebnú premennú, ktorá mala začiatočnú hodnotu 0, pokým nedosiahla maximálnu možnú hodnotu vzdialenosti, čo je polovica z premennej $\_m = 8192$. V cykle sme zároveň kontrolovali, či hodnota premennej má početnosť práve 1 v dištančnom spektre z matice H a 0 v dištančnom spektre z matice Q.

4.3.3 Vzdialenosť raz v matici Q a ani raz v H

Tento prípad je presne opačný ako posledne spomínaný. Úloha bola získať vzdialenosť, ktorá sa nachádza raz v dištančnom spektre z matice Q a ani raz v dištančnom spektre z matice H. Funkciu sme nazvali $getD\_oneInQ\_notInH()$ a jej algoritmus bol podobný ako v poslednej opísanej funkcii. Rozdiel bol v tom, že sme vymenili spektrá a tak sme získali potrebnú vzdialenosť.

4.4 Generovanie chybového vektora

V predošlej sekcii sme popísali ako sme získali potrebné vzdialenosti. Tieto vzdialenosti sme potrebovali na generovanie chybových vektorov. Vytvorili sme si funkciu s názvom $create\_error\_vector()$, ktorej vstupom bola požadujúca vzdialenosť d a vektor typu $BPU\_T\_GF2\_Vector$ (štruktúra v knižnici $BitPunch$). Na začiatku algoritmu

sme pre chybový vektor alokovali pamäť a inicializovali premennú *odd*, do ktorej sme uložili hodnotu 1 alebo 0. Túto premennú sme potrebovali na kontrolu, či počet jednotiek v chybovom vektore bol párny alebo nepárny. Ak premenná mala hodnotu 1, tak počet jednotiek bol nepárny, ak bola 0, tak počet jednotiek bol párny. Počet jednotiek v chybovom vektore predstavovala premenná $\_t$. Ďalej nasledoval samotný cyklus v algoritme.
V cykle sme inicializovali premenné k, l, p, pd a t. Premenná k predstavovala rovnomerne náhodný index bloku v chybovom vektore vynásobený veľkosťou bloku - $_m$, l náhodnú pozíciu prvej jednotky a mohla mať hodnotu od 0 po 8192, p presnú pozíciu prvej jednotky v danom bloku, čiže súčet premenných k a l, pd pozíciu párovej jednotky, čiže hodnotu inkrementovanej premennej p o vzdialenosť d a zmodulovanú premennou $_m$, aby pozícia pd nemala vyššiu hodnotu ako je veľkosť bloku a premenná t predstavovala polovicu z celkového počtu jednotiek chybového vektora. V cykle sme kontrolovali, či na pozícii p je 0, ak áno, tak sme dekrementovali premennú t a nastavili na pozíciu p jednotku. Nasledovne sme nastavili párovú jednotku na pozíciu pd. Nastavenie párovej jednotky prebiehalo len v prípade, ak celkový počet jednotiek v chybovom vektore bol párny alebo ak dekrementovaná hodnota premennej t nebola 0. Túto kontrolu sme potrebovali doimplementovať, aby sme sa vyhli nastaveniu párovej jednotky k poslednej jednotke v prípade nepárneho počtu jednotiek chybového vektora.

Kapitola 5

Výsledky experimentov

V tejto kapitole si ukážeme výsledky našich experimentov pri útoku na QC-LDPC MCEliece

5.1 GRID počítač

Centrum výpočtovej techniky (CVT) STU sa špecializuje na vysokovýkonné počítanie celouniverzitného pracoviska Slovenskej technickej univerzity v Bratislave a zabezpečuje prevádzku výpočtového klastra IBM iDataPlex. Pre vykonanie našich zložitých experimentov potrebných na následné zrekonštruovanie útoku bolo nevyhnutné získať prístup k takémuto superpočítaču. Pomocou neho sme dokázali urýchliť beh experimentov a teda získať výsledky skôr a vedieť v našom projekte efektívnejšie napredovať. Zatiaľ čo na osobnom počítači by nejaký experiment mohol bežať týždeň, na tomto superpočítači nám dokázal poskytnúť výsledky do 24 hodín.

Centrum výpočtovej techniky je súčasne okrem iného pracoviskom zabezpečujúcim kooperovaný prístup na ďalšie superpočítače v rámci projektu Slovenská infraštruktúra pre vysokovýkonné počítanie (SIVVP) na Slovensku. Tento projekt umožňuje realizáciu veľmi zložitých numerických výpočtov pre oblasť výskumu a vývoja a poskytuje rovnocenné pripojenie do existujúcich európskych vysokovýkonných výpočtových systémov.

Klaster pozostáva z 52 výpočtových uzlov IBM iDataPlex dx360 M3, z toho 4 sú osadené dvoma GPU akcelerátormi NVIDIA Tesla. K dispozícii je 624 cpu + 3584 cuda jadier, 2,5TB ram, disková raw kapacita zdieľaného úložiska dát 115TB a ďalších 104TB na lokálnych diskoch. Jeho výpočtový výkon cpu je 6,76 TFLOPS (Linpack benchmark), čo je 92% teoretického výkonu. Maximálny príkon spolu s dátovým úložiskom a obslužnými zariadeniami vrátane chladenia je 40kW.

Pri testovaní sme na spomínanom gride spotrebovali dohromady 35 489h strojového času, ktorý je zobrazený v obrázku č.5.1

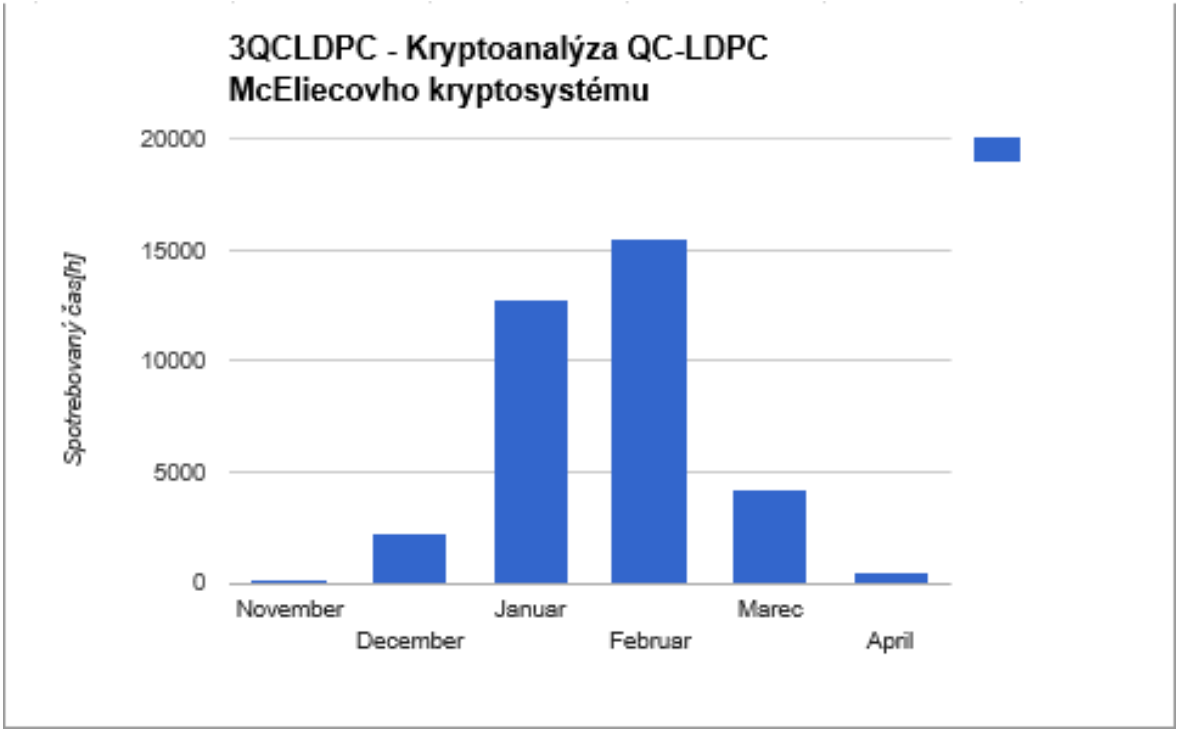

Figure 5.1: Využitie GRID-u

5.2 Naše výsledky

Po úspešnej implementácii potrebných funkcií sme prešli k experimentom potrebných pre útok na QC-LDPC. Použitie iteratívneho dekódovanie pri dešifrovaní v kryptosystéme McEliece QC-LDPC, má za následok občasné zlyhanie. Cieľom našich experimentov bolo dokázať závislosť medzi súkromným kľúčom a zlyhaním v dekódovaní. Pri vykonávaní experimentov sme vyberali jednotlivé vzdialenosti d z dištančného spektra Ψ_d a pre každú vzdialenosť sme generovali 1000 vektorov, ktoré sme posielali na dešifrovanie.

Následne sme z výsledkov sledovali pravdepodobnosť dekódovacej chyby. Otázkou bolo, či dokáže útočník zistiť vzdialenosti jednotiek v matici H a Q, ktoré sú zároveň súkromným kľúčom, čím by získal cennú informáciu, a to vzťah súkromného kľúča s pravdepodobnosťou dekódovacej chyby.

Experimenty sme vykonávali na troch typoch dát, podľa toho či sa vzdialenosti z dištančného spektra nachádzali v matici H, alebo v matici Q, alebo sa takéto vzdialenosti nenachádzali v ani jednej z matíc. Z výsledného grafu 5.2 vidíme že ak sa vzdialenosť z dištančného spektra nachádzala v matici Q, pravdepodobnosť zlyhanie je veľmi nízka, v intervale $< 0,002; 0,022 >$ s priemernou hodnotou 0,00805.

V strednom pásme sa nachádzajú hodnoty, kedy bola vzdialenosť zo spektra v matici H, v intervale $< 0,037; 0,13 >$ s priemernou hodnotou 0,08. A najčastejšia chyba pri dekódovaní nastala, pokiaľ sa v maticiach H a Q nenachádzali vzdialenosti z dištančného spektra, pravdepodobnosť zlyhania sa nachádza v intervale $< 0,106; 0,234 >$ s priemernou hodnotou 0,1695. Pre každú jednu vzdialenosť, sme testovali dešifrovanie na vzorke 1000 vektorov.

Na základe vykonaných experimentov teda môžeme predpokladať, že v danom kryptosystéme by mohol útočník vykonať nasledovné kroky:

1. zvolenie d - teda potencionálnych vzdialeností jednotiek v maticiach H a Q
2. vygenerovanie 1000 vektorov pre každé zvolené d
3. posielanie takto špeciálne skonštruovaných správ
4. na základe príslušnej dekódovacej chyby pre jednotlivé typy zašifrovaných textov rozdeliť jednotlivé vzdialenosti do 3 kategórií: vzdialenosti, ktoré sa pravdepodobne vyskytujú v matici H; vzdialenosti, ktoré sa pravdepodobne vyskytujú v matici Q a vzdialenosti, ktoré sa pravdepodobne nevyskytujú ani v matici H, ani v matici Q

Na základe týchto informácií sa následne môže pokúsiť o rekonštrukciu týchto matíc, t.j. o získanie súkromného kľúča príslušného kryptosystému. Tieto experimenty ukazujú teoretickú možnosť rekonštrukcie súkromného kľúča, a teda možnosť vykonania reakčného útoku na kryptosystém McEliece s QC-LDPC kódmi. Výsledky týchto experimentov dokazujú silnú koreláciu medzi štruktúrou matíc súkromného kľúča a pravdepodobnosťou zlyhania dekódovania.

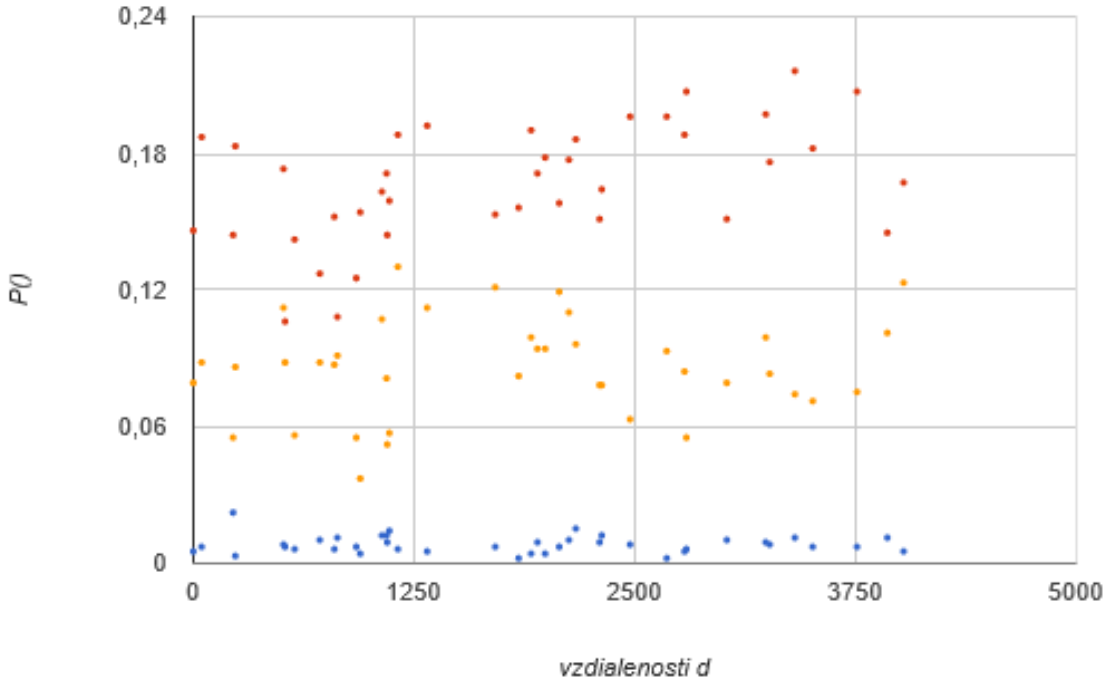

Figure 5.2: Výsledky experimentu

Kapitola 6

Záver

Cieľom tohto tímového projektu bolo vykonanie experimentov pre podporu útoku na QC-LDPC McEliece. Keďže ide o náročnú téme, tak sme v teoretickej časti vysvetlili základné pojmy pre pochopenie danej problematiky. Na konci tejto časti sme popísali základne konštrukcie kryptosystému McEliece a to s Goppa kódmi, s MDPC a LDPC kódmi.

Následne sme sa venovali popisu útoku, ktorý vykonal Thomas Johansson na systéme s MDPC kódmi a k jeho analógii a popisu podobného útoku na McEliece s použitím QC-LDPC kódov. V praktickej časti sme doimplementovali potrebné funkcie do knižnice bitPunch a spúšťali potrebné experimenty. Výsledky experimentov sú relevantné na tento kryptosystém.

Bibliography

[1] Grošek, O., et al. Základy kryptografie (Elementary cryptography), Bratislava, 2006. ISBN 80-227-2415-7.

[2] ČIPKOVÁ, RNDr Karla. ZS 2015/2016–FIIT. KATALOG PREDMETOV Fakulta informatiky a informačných technológií, 77.

[3] Misoczki, Rafael, et al. "MDPC-McEliece: New McEliece variants from moderate density parity-check codes." Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on. IEEE, 2013.

[4] Guo, Qian, Thomas Johansson, and Paul Stankovski. "A key recovery attack on MDPC with CCA security using decoding errors." Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22. Springer Berlin Heidelberg, 2016.

[5] Baldi, Marco, Marco Bodrato, and Franco Chiaraluce. "A new analysis of the McEliece cryptosystem based on QC-LDPC codes." International Conference on Security and Cryptography for Networks. Springer Berlin Heidelberg, 2008.

[6] MacWilliams, Florence Jessie, and Neil James Alexander Sloane. The theory of error-correcting codes. Elsevier, 1977.

Appendix A

Obsah elektronického média

/tpDokumentacia.pdf .....elektronická verzia dokumentu

/program/BitPunch ..... zdrojové kódy programu

/vysledkyExperimentu.csv .....výsledky experimentu