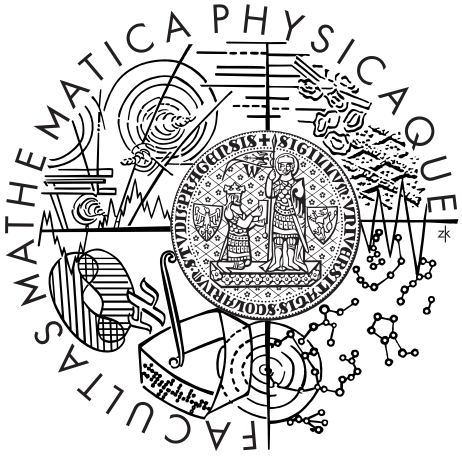
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Miloslav Homer

Útok na Wieschebrinkovu verziu Niederreteirovho systému

Katedra algebry

Vedoucí bakalářské práce:	Prof. RNDr. Aleš Drápal, CSc., DSc.
Studijní program:	Matematika
Studijní obor:	Matematické metody informační bezpečnosti

Praha 2015

Ďakujem prof. RNDr. Alešovi Drápalovi, CSc., DSc., za trpezlivosť s akou viedol moju prácu, za ochotu pri konzultáciách, za cenné rady, ako aj za poskytnutie odbornej literatúry.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Název práce: Útok na Wieschebrinkovu verzi Niederreiterova systému

Autor: Miloslav Homer

Katedra: Katedra algebry

Vedoucí bakalářské práce: Prof. RNDr. Aleš Drápal, CSc., DSc., Katedra algebry

Abstrakt: V práci je popsán útok na Wieschebrinkovu verzi Niederreterova kryptosystému využívajícího GRS kódy podle Couvreur et. al. z roku 2014. Jsou zde definovány relevantní pojmy teorie samoopravných kódů, McEliecovo schéma, Niederreiterovo schéma a dále Wieschebrinkovy modifikace obou schémat. Následuje popis samotného útoku obsahující distinguishera podle Couvreura et. al. založeného na vlastnostech součinu kódů po složkách a vlastnostech zkrácených kódů. Také je představen Sidelnikův-Šestakův útok na Niederreiterovo schéma a s ním spojené pojmy teorie grup a problémy implementace. Útok je poté shrnut a je odhadnuta jeho časová složitost. V poslední kapitole jsou uvedeny reálné časy útoků naměřené pomocí implementace v jazyce C++. Přílohou práce je program včetně dokumentace, který implementuje kryptosystém i útok na něj.

Klíčová slova: McEliecovo schéma, Niederreiterův kryptosystém veřejného klíče, GRS kódy, Wieschebrinkovo šifrovací schéma, součin kódů po složkách

Title: An attack upon Wieschebrink's version of Niederreiter system

Author: Miloslav Homer

Department: Department of Algebra

Supervisor: Prof. RNDr. Aleš Drápal, CSc., DSc., Department of Algebra

Abstract:

In this work an attack upon Wieschebrink's version of Niederreiter cryptosystem using GRS codes by Couvreur et. al. from 2014 is described. Relevant notions of error-correcting code theory are presented, definitions of McEliece scheme, Niederreiter scheme and their respective Wieschebrink's modifications are shown. A description of the attack using distinguisher as described by Couvreur et. al. Based on componentwise code products and shortened codes properties follows, as does Sidelnikov-Shestakov attack on Niederreiter scheme with relevant group theory notions. Implementation details are also outlined. The attack is then summarized and its complexity is mentioned. The attack duration measured by the C++ implementation is presented in the last chapter. The program implementing the cryptosystem as well as the attack is located in the appendix with the program documentation.

Keywords: McEliece scheme, Niederreiter public key cryptosystem, GRS codes, Wieschebrink's encryption scheme, componentwise code products

Názov práce: Útok na Wieschebrinkovu verziu Niederreiterovho systému

Autor: Miloslav Homer

Katedra: Katedra algebry

Vedúci bakalárskej práce: Prof. RNDr. Aleš Drápal, CSc., DSc., Katedra algebry

Abstrakt: V práci je popísaný útok na Wieschebrinkovu verziu Niederreterovho kryptosystému využívajúceho GRS kódy podľa Couvreur et. al. z roku 2014. Sú vyložené relevantné pojmy teórie samoopravných kódov, definície McEliecovej schémy, Niederreiterovej schémy, ako aj Wieschebrinkových modifikácii daných schém. Nasleduje popis samotného útoku obsahujúci distinguishera podľa Couvreura et. al. založeného na vlastnostiach súčinu kódov po zložkách a vlastnostiach skrátených kódov.Taktiež je predvedený Sidelnikov-Šestakov útok na Niederreiterovo schéma a s ním spojené pojmy teórie grúp a taktiež sú spracované implementačné detaily. Útok je potom zhrnutý a je spomenutá jeho časová zložitosť. V poslednej kapitole sú uvedené reálne časy útokov namerané pomocou jeho implementácie v jazyku C++. V prílohách sa vyskytuje program implementujúci ako kryptosystém tak aj útok, vrátane dokumentácie programu.

Kľúčové slová: McEliecovo schéma, Niederreitov kryptosystém verejného kľúča, GRS kódy, Wieschebrinkovo šifrovacie schéma, súčin kódov po zložkách

Obsah

Úvod

V roku 1978 McEliece publikoval článok [1], v ktorom predstavil asymetrický kryptosystém založený na teórii samoopravných kódov a na probléme dekódovania všeobecného kódu. Vo svojom článku navrhol použiť Goppa kódy a v tejto podobe jeho kryptosystém ostáva nezlomený dodnes, aj keď sa podarilo prelomiť pár špecifických prípadov [2], [3]. Taktiež je to jeden z mála kryptosystémov odolných voči kvantovým algoritmom [4], čo bohužiaľ nie je prípad väčšiny dnes používaných kryptosystémov (napr. RSA alebo systémy založené na teórii eliptických kriviek). Neskôr, v roku 1986, prišiel Niederreiter v [5] s podobným kryptosystémom, avšak namiesto generujúcej matice a správ v podobe kódových slov navrhol použiť maticu kontrolnú a chybové slová. Bolo ukázané, že za predpokladu rovnakého použitého kódu, sú tieto kryptosystémy ekvivalentne bezpečné [6]. Avšak Niederreiter vo svojom systéme navrhol použiť namiesto Goppa kódov GRS kódy, a v tejto podobe daný systém prelomili Sidelnikov a Šestakov v [7].

Obranu proti ich útoku navrhol Wieschebrink [8], ale Couvreur et al. [9] prelomili aj tento pokus využiť GRS kódy v kryptografii. Útok založili na spojení úvah o odlíšiteľnosti GRS kódov od náhodných, na vlastnostiach skrátených kódov, ako aj na pôvodnom útoku Sidelnikova a Šestakova.

Týmto útokom sa zaoberá táto práca, ktorá si dáva za cieľ sprehľadnenie útoku, ako aj kompletnú implementáciu kryptosystému a útoku.

V úvode sú uvedené relevantné pojmy teórie samoopravných kódov, ako aj popis kryptosystémov. Nasleduje samotný popis útoku – distinguisher schopný odlíšiť GRS kód od náhodného, Sidelnikov-Šestakov algoritmus vrátane relevantných výsledkov teórie grúp, ako aj problémy nematematického rázu zabraňujúce implementácii útoku.

Kapitola 1

Samoopravné kódy

1.1 Základné pojmy

Definícia 1.1. *Nech je $\mathbb{F}_q$ konečné teleso, nech $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}_q^n$. Hammingovou vzdialenosťou (ďalej len vzdialenosťou) rozumieme počet súradníc, na ktorých sa $\boldsymbol{x}$ líši od $\boldsymbol{y}$, značíme $d(\boldsymbol{x},\boldsymbol{y})$. Hammingovou váhou (ďalej len váhou) rozumieme počet súradníc $\boldsymbol{x}$ s nenulovou hodnotou, značíme $w(\boldsymbol{x})$.*

Definícia 1.2. *Nech je $\mathbb{F}_q$ konečné teleso. Lineárnym $[n,k]_q$ ($[n,k]$) kódom nazveme ľubovoľný podpriestor $\mathcal{C} \subseteq \mathbb{F}_q^n$ dimenzie k. Ak navyše platí*

$$\min_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{C}\colon \boldsymbol{x}\neq\boldsymbol{y}} d(\boldsymbol{x},\boldsymbol{y}) = d$$

hovoríme o lineárnom $[n,k,d]_q$ ($[n,k,d]$) kóde. Tiež hovoríme, že minimálna vzdialenosť kódu je d (pre úplnosť definujeme minimálnu vzdialenosť jednoprvkového kódu ako n). Prvky $\mathbb{F}_q^n$ (riadkové) nazveme slová a prvky $\mathcal{C}$ nazveme kódové slová.

Poznámka 1.1. Ak nie je hodnota q uvedená, predpokladáme, že hodnota q je rovná 2.

Lemma 1.1. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód a $\mathbf{0}$ je nulový vektor. Potom*

$$d = \min_{\boldsymbol{x}\in\mathcal{C}\colon \boldsymbol{x}\neq\mathbf{0}} w(\boldsymbol{x}).$$

Dôkaz. Pretože $\mathcal{C}$ je podpriestor, tak platí

$$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{C}\colon \boldsymbol{x} - \boldsymbol{y} \in \mathcal{C}.$$

Ďalej platí z definície 1.1

$$\min_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{C}\colon \boldsymbol{x}\neq\boldsymbol{y}} d(\boldsymbol{x},\boldsymbol{y}) = \min_{\boldsymbol{x},\boldsymbol{y}\in\mathcal{C}\colon \boldsymbol{x}\neq\boldsymbol{y}} w(\boldsymbol{x}-\boldsymbol{y}) = \min_{\boldsymbol{z}\in\mathcal{C}\colon \boldsymbol{z}\neq\mathbf{0}} w(\boldsymbol{z}).$$

❑

Definícia 1.3. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód. Generujúcou maticou kódu $\mathcal{C}$ označíme maticu G nad $\mathbb{F}_q$ typu $k \times n$, ktorej riadky tvoria bázu $\mathcal{C}$.*

Definícia 1.4. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód. Kontrolnou maticou kódu $\mathcal{C}$ označíme maticu H nad $\mathbb{F}_q$ typu $r \times n$, ktorá spĺňa*

$$\boldsymbol{x} \in \mathcal{C} \Leftrightarrow H\boldsymbol{x}^T = 0.$$

Poznámka 1.2. Ľahko nahliadneme, že $\text{rank}(H) = n - k$, a teda ak matica H obsahuje $n - k$ riadkov, tak sú lineárne nezávislé. Vskutku,

$$\text{rank}(H) = n - \dim(\ker(H)) = n - \dim(\mathcal{C}) = n - k.$$

Pretože $\text{rank}(H) = n - k$, tak výberom $n - k$ stĺpcov H, dostaneme regulárnu maticu rozmerov $(n-k) \times (n-k)$. Ak predpokladáme $n-k$ riadkov, tak sú tieto riadky lineárne nezávislé. Ďalej v texte budem pri kontrolnej matici predpokladať $n - k$ riadkov, pokiaľ nie je napísané inak.

Lemma 1.2. *Nech $\mathcal{C} \neq \{\mathbf{0}\}$ je lineárny $[n,k]_q$ kód s kontrolnou maticou H. Potom každá $(d-1)$-tica stĺpcov H je lineárne nezávislá práve vtedy keď minimálna vzdialenosť kódu $\mathcal{C}$ je rovná d (a to d je najväčšie také na oboch stranách ekvivalencie).*

Dôkaz. Pre každé slovo $\boldsymbol{x}$ z $\mathcal{C}$ platí: $H\boldsymbol{x}^T = \mathbf{0}$.
„⇐“ Sporom.
Z lemmy 1.1 platí, že najmenšia možná váha nenulového slova $\boldsymbol{x}$ je $w(\boldsymbol{x}) = d$. Ak nájdeme $d-1$ závislých stĺpcov, vieme túto závislosť vyjadriť ako vektor $\boldsymbol{y}$ a na zvyšné pozície môžme dať nuly. Práve sme našli vektor váhy $d-1$, ktorý patrí kódu $\mathcal{C}$, spor s lemmou 1.1.
„⇒“
Pretože existuje d-tica stĺpcov H, ktorá je lineárne závislá, analogicky zostrojíme vektor váhy d a teda minimálna vzdialenosť $\mathcal{C}$ je najviac d. Ak by bola minimálna vzdialenosť kódu $\mathcal{C}$ menšia ako d, z lemmy 1.1 nájdeme vektor váhy $d-1$, ktorý vyjadruje lineárnu závislosť $(d-1)$-tice stĺpcov, dostávame spor. Váha kódu $\mathcal{C}$ je teda rovná d.

❑

Veta 1.3 (Singletonov odhad)**.** *Pre každý lineárny $[n,k,d]_q$ kód platí:*

$$d \leq n - k \ + 1.$$

Dôkaz. Kontrolná matica H obsahuje regulárnu maticu rozmerov $(n-k)\times(n-k)$. Z lemmy 1.2 platí $d - 1 \leq n - k$.

❑

Definícia 1.5. *Lineárny $[n,k,d]_q$ kód nazveme MDS kódom (Maximum Distance Separable) ak pre neho nastáva rovnosť v Singletonovom odhade (tj. $d = n-k+1$).*

Definícia 1.6. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód s generujúcou maticou G. Duálnym kódom k $\mathcal{C}$ označíme kód $\mathcal{C}^\perp$, definovaný ako:*

$$\mathcal{C}^\perp = \{\boldsymbol{x} \in \mathbb{F}_q^n \colon \boldsymbol{x}G^T = \mathbf{0}\}.$$

Pozorovanie 1.3. Ak je $\mathcal{C}$ ako v predchádzajúcej definícii, potom G je kontrolná matica kódu $\mathcal{C}^\perp$.

Lemma 1.4. *Ak $\mathcal{C}$ je MDS, potom aj $\mathcal{C}^\perp$ je MDS.*

Dôkaz. Vieme, že $\mathcal{C}^\perp$ je $[n,n-k]_q$ kód. Z lemmy 1.2 máme, že každá množina stĺpcov kontrolnej matice H kódu $\mathcal{C}$ veľkosti $n-k$ je lineárne nezávislá. Teda každá množina stĺpcov veľkosti $n-k$ tvorí regulárnu maticu. Z toho plynie, že netriviálna kombinácia riadkov nemôže obsahovať $n-k$ a viac núl – váha takéhoto vektoru je aspoň $k+1$. Z pozorovania 1.3 máme, že H je generujúcou maticou $\mathcal{C}^\perp$, a teda minimálna vzdialenosť kódu $\mathcal{C}^\perp$ je $k+1$, čo sme chceli dokázať. ❑

Definícia 1.7. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód s generujúcou maticou G a kontrolnou maticou H. Povieme, že G je v štandardnom tvare ak $G = (I_k|A\ )$, kde I_k je identická matica rozmeru $k \times k$ a A je matica typu $k \times (n-k)$.*

Definícia 1.8. *Nech $\mathcal{C}$ je ako v definícii 1.7. Povieme, že $\mathcal{C}^*$ s generujúcou maticou G^* a kontrolnou maticou H^* je permutačne ekvivalentný kód kódu $\mathcal{C}$, ak platí:*

- *$\mathcal{C}^*$ je lineárny $[n,k,d]_q$ kód.*
- *Existuje permutácia $\Pi \in \boldsymbol{S}_n$, pre ktorú platí: $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \mathcal{C} \Leftrightarrow \boldsymbol{c^*} = (c_{\Pi(1)}, c_{\Pi(2)}, \ldots, c_{\Pi(n)}) \in \mathcal{C}$.*

Lemma 1.5. *Nech $\mathcal{C}$ je $[n,k,d]_q$ kód.*

1. *Existuje permutačne ekvivalentný kód $\mathcal{C}^*$, ktorý má generujúcu maticu v štandardnom tvare.*
2. *Ak $G = (I_k|A)$ je generujúca matica, potom matica $H = (-A^T|I_{n-k})$ je kontrolná matica.*

Dôkaz. 1) Vidno, že použitím ekvivalentných riadkových úprav na generujúcu maticu nezmeníme kódové slová (iba dostaneme inú, ekvivalentnú bázu). Preto môžeme použiť na maticu G Gauss-Jordanovu elimináciu a získame maticu generujúcu maticu G' v tvare:

$$G' = (\boldsymbol{e}_1^T, \ldots, \boldsymbol{e}_2^T, \ldots, \boldsymbol{e}_n^T, \ldots),$$

kde $\boldsymbol{e}_i$ je i-ty kanonický vektor. Ak označíme číslom l_j stĺpcovú pozíciu $\boldsymbol{e}_j$, tak permutáciou ktorú hľadáme je $\Pi = (n\ l_n) \circ ((n-1)\ l_{n-1}) \circ \ldots \circ (1\ l_1)$. To plynie z toho, že $\boldsymbol{e}_j$ na pozícii l_j sa v štandardnom tvare objaví na pozícii j. A teda index l_j chcem „prepermutovať“ na index j. Tak dostanem vyššie uvedenú permutáciu, definujúcu permutačne ekvivalentný kód.

2) Očividne $\operatorname{rank}(H) = n-k$ a H je typu $(n-k) \times n$. Pretože kódové slová z $\mathcal{C}$ sú generované riadkovými vektormi z G, stačí ukázať, že $HG^T = \mathbf{0}$, kde $\mathbf{0}$ je nulová matica. Ale:

$$(-A^T|I_{n-k}) \times \begin{pmatrix} I_k \\ A^T \end{pmatrix} = -A^T I_k + I_{n-k} A^T = \mathbf{0}.$$

❑

Poznámka 1.4. Pozrime sa teraz na kódovanie informačných slov (vo forme vektorov z $\mathbb{F}_q^k$). Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód s generujúcou maticou G. Informačné slovo $\boldsymbol{u} \in \mathbb{F}_q^k$ zakódujeme na kódové slovo použitím zobrazenia $\phi_G \colon \boldsymbol{u} \to \boldsymbol{u}G$. Pretože $\text{rank}(G) = k$ jedná sa o bijekciu. Ak je G v štandardnom tvare ako v definícii 1.7, tak $\phi_{(I_k|A\ )} \colon \boldsymbol{u} \to (\boldsymbol{u} \mid \boldsymbol{u}A)$.

Definícia 1.9. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód s kontrolnou maticou H typu $(n-k) \times n$. Nech $\boldsymbol{y} \in \mathbb{F}_q^n$. Syndrómom $\boldsymbol{y}$ rozumieme vektor $\boldsymbol{s} \in \mathbb{F}_q^{n-k}$, ktorý spĺňa $\boldsymbol{s} = H\boldsymbol{y}^T$. Rozkladovou triedou $\boldsymbol{y}$ rozumieme množinu $\boldsymbol{y} + \mathcal{C} = \{\boldsymbol{y} + \boldsymbol{c} \mid \boldsymbol{c} \in \mathcal{C}\}$.*

Tvrdenie 1.6. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód. Potom vieme opraviť t chýb práve vtedy, keď $2t < d$.*

Dôkaz. Ekvivalenciu dokážem dvakrát nepriamo.

„Neviem opraviť t chýb $\Rightarrow 2t \geq d$.“

Nech $\boldsymbol{u} \in \mathbb{F}_q$ je slovo s t chybami. Pretože tieto chyby neviem opraviť, existujú aspoň dve slová $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{C}$ tak , že

$$d(\boldsymbol{c}_1,\boldsymbol{u}) \leq t \ \& \ d(\boldsymbol{c}_2,\boldsymbol{u}) \leq t.$$

Ale Hammingova vzdialenosť spĺňa trojuholníkovú nerovnosť, takže dostávam:

$$2t \geq d(\boldsymbol{c}_1,\boldsymbol{u}) + d(\boldsymbol{c}_2,\boldsymbol{u}) \geq d(\boldsymbol{c}_1, \boldsymbol{c}_2) \geq d.$$

„$2t \geq d \Rightarrow$ neviem opraviť t chýb.“

Nech $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{C}$ a nech $d(\boldsymbol{u},\boldsymbol{v}) = d$. Zmením t hodnôt $\boldsymbol{u}$ na hodnoty $\boldsymbol{v}$ (a dostanem slovo $\boldsymbol{w}$) tak, aby som zmenšoval ich vzdialenosť. Pretože $2t \geq d$, tak $t \geq d - t$ a teda vzdialenosť $\boldsymbol{v}$ od $\boldsymbol{w}$ je tiež menšia alebo rovná t, a teda slovo $\boldsymbol{w}$ neviem opraviť.

❑

Poznámka 1.5. Z definície 1.4 nahliadneme, že ak $\boldsymbol{y}_1,\boldsymbol{y}_2 \in \mathbb{F}_q^n$ tak :

$$\boldsymbol{y}_1 - \boldsymbol{y}_2 \in \mathcal{C} \Leftrightarrow H\boldsymbol{y}_1^T = H\boldsymbol{y}_2^T,$$

a teda $\boldsymbol{y}_1,\boldsymbol{y}_2$ sa nachádzajú v rovnakej rozkladovej triede práve vtedy, ak sa ich syndrómy rovnajú. Takže dekódovať môžeme nasledujúcim spôsobom:

1. Nájdenie syndrómu: Pre prijaté $\boldsymbol{y} \in \mathbb{F}_q^n$ spočítaj $\boldsymbol{s} = H\boldsymbol{y}^T$.

2. Nájdenie chybového vektoru: Nájdi vektor $\boldsymbol{e} \in \mathbb{F}_q^n$ spĺňajúci

$$\boldsymbol{s} = H\boldsymbol{e}^T \quad \& \quad w(\boldsymbol{e}) = \min_{\boldsymbol{y} \in \mathcal{C} \colon H\boldsymbol{y}^T = \boldsymbol{s}} (w(\boldsymbol{y})).$$

Pretože ak $2w(\boldsymbol{e}) < d$, tak $\boldsymbol{e}$ je určené jednoznačne (tvrdenie 1.6) a teda daný algoritmus funguje. Dôsledkom je taktiež, že každý kód má jednoznačne určenú hornú hranicu počtu chýb, ktoré vie jednoznačne opraviť. Bohužiaľ, druhý krok je pre všeobecnú maticu H výpočtovo náročný problém (NP úplný) [10]. Našťastie, ako sa ukáže v ďalšej sekcii, pre špeciálne prípady existuje efektívny dekódovací algoritmus.

1.2 Zovšeobecnené Reed-Solomonove kódy (GRS)

Definícia 1.10. *Nech $\mathbb{F}_q$ je teleso, nech $\alpha_1, \alpha_2, \ldots, \alpha_n$ sú rôzne nenulové prvky $\mathbb{F}_q$, a nech $v_1, v_2, \ldots, v_n$ sú nenulové (nemusia byť rôzne) prvky $\mathbb{F}_q$. Zovšeobecneným Reed-Solomonovým (Generalized Reed-Solomon – GRS) kódom nad $\mathbb{F}_q$ nazveme lineárny $[n,k,d]_q$ kód $\mathcal{C}_{GRS}$ nad $\mathbb{F}_q$, definovaný kontrolnou maticou:*

$$H_{GRS} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{n-k-1} & \alpha_2^{n-k-1} & \cdots & \alpha_n^{n-k-1} \end{pmatrix} \begin{pmatrix} v_1 & & & \\ & v_2 & & \\ & & \ddots & \\ 0 & & & v_n \end{pmatrix}.$$

Prvky α_i nazveme lokátory kódu a prvky v_i nazveme multiplikátory kódu.

Z definície GRS kódu plynie, že $n \leq q - 1$.

Lemma 1.7. *GRS kódy sú MDS a teda ak $\mathcal{C}_{GRS}$ je $[n,k,d]_q$ kód, tak $d = n - k + 1$.*

Dôkaz. Nech $\mathcal{C}_{GRS}$ je GRS kód ako v definícii 1.10. Pretože vynásobenie stĺpcov nenulovými hodnotami v_i nezmení ich lineárnu závislosť/nezávislosť, môžeme predpokladať, že sú všetky rovné 1 a teda je kontrolná matica kódu $\mathcal{C}_{GRS}$ rovná:

$$H_{GRS} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{n-k-1} & \alpha_2^{n-k-1} & \cdots & \alpha_n^{n-k-1} \end{pmatrix}.$$

Pre $n-k$ rôznych stĺpcov platí, že tvoria $(n-k) \times (n-k)$ Vandermondovu maticu a teda regulárnu maticu (z Vandermondovho determinantu a rôznosti prvkov α_i). Z toho okamžite plynie, že $n - k$ rôznych stĺpcov je lineárne nezávislých a teda z lemmy 1.2 a vety 1.3 dostávame požadovanú rovnosť.

❑

Lemma 1.8 ([11] str. 148)**.** *Nech $\mathcal{C}_{GRS}$ je $[n,k,d]_q$ GRS kód ($k < n$). Potom duálny kód $\mathcal{C}_{GRS}^{\perp}$ je $[n,n-k]_q$ GRS kód a $\forall i \leq n$: $\alpha_i = \alpha_i^{\perp}$.*

Z lemmy 1.8 a pozorovania 1.3 plynie dôsledok:

Dôsledok 1.9. *Nech $\mathcal{C}_{GRS}$ je $[n,k,d]_q$ GRS kód ($k < n$). Existuje generujúca matica G_{GRS} kódu $\mathcal{C}_{GRS}$ v tvare:*

$$G_{GRS} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \cdots & \alpha_n^{k-1} \end{pmatrix} \begin{pmatrix} v_1' & & & \\ & v_2' & & \\ & & \ddots & \\ 0 & & & v_n' \end{pmatrix}.$$

Poznámka 1.6. Užitočný býva nasledujúci náhľad na GRS kódy. Nech $\mathcal{C}_{GRS}$ je ako v definícii 1.10. Zakódujme slovo $\boldsymbol{u} \in \mathbb{F}_q^k$ ako v poznámke 1.4. Ak sa pozrieme na slovo $\boldsymbol{u} = (u_0, u_1, \ldots, u_{k-1})$ ako na polynóm $u(x) = u_0 + u_1 x + \ldots u_{k-1}x^{k-1}$, potom

$$\boldsymbol{u}G_{GRS} = (v_1'u(\alpha_1), v_2'u(\alpha_2), \ldots, v_n'u(\alpha_n)),$$

z čoho plynie, že kódové slovo získame vyhodnotením polynómu v kódových lokátoroch a následným vynásobením upravenými kódovými multiplikátormi.

Poznámka 1.7. Pohľad na vektory ako na polynómy vieme s výhodou využiť aj pri dekódovaní GRS kódov, ako je uvedené podrobne v [11]. Nech $\mathcal{C}_{GRS}$ je ako v definícii 1.10. Nech $\boldsymbol{y} \in \mathbb{F}_q^n$, nech $\boldsymbol{s} = (s_0, s_1, \ldots, s_{n-k-1})$ je syndróm tohto slova, nech $\boldsymbol{e} \in \mathbb{F}_q^n$ je chybový vektor, ktorým je $\boldsymbol{y}$ zaťažené, a nech J je množina indexov nenulových prvkov $\boldsymbol{e}$. Predpokladáme, že nenastalo viac ako $(n-k-1)/2$ chýb, tj. $|J| \leq (n-k-1)/2$. Definujme polynóm $S(x)$ ako polynóm s koeficientami syndrómu. Potom (z pozn. 1.5):

$$S(x) = \sum_{l=0}^{n-k-1} s_l x^l = \sum_{l=0}^{n-k-1} x^l \sum_{j \in J} e_j v_j \alpha_j^l = \sum_{j \in J} e_j v_j \sum_{l=0}^{n-k-1} (\alpha_j x)^l.$$

Ďalej definujme *lokalizačný polynóm*

$$\Lambda(x) = \prod_{j \in J}(1 - \alpha_j x).$$

A *evaulačný polynóm*

$$\Gamma(x) = \sum_{j \in J} e_j v_j \prod_{m \in J \setminus \{j\}} (1 - \alpha_m x).$$

Všimnime si, že pre lokalizačný polynóm platí

$$\Lambda(\alpha_k^{-1}) = 0 \Leftrightarrow k \ \in J.$$

Ak nájdeme korene lokalizačného polynómu, odhalíme pozície chýb. Rovnice

$$\begin{aligned} &\mathrm{NSD}(\Lambda(x), \Gamma(x)) = 1, \\ &\deg(\Gamma) < \deg(\Lambda) \leq (n-k-1)/2, \\ &\qquad \Lambda(x)S(x) \equiv \Gamma(x) \pmod{x^{n-k-1}} \end{aligned}$$

tvoria tzv. *kľúčovú rovnicu*. V [11] je ukázané, že sa dá vyriešiť Euklidovým algoritmom, ako aj Berlekamp-Massey-ho algoritmom, čím získame $\Lambda(x), \Gamma(x)$. Chienovým vyhľadávaním ďalej nájdeme korene $\Lambda(x)$ a teda množinu J. Nakoniec použitím Forneyovho algoritmu, získame hodnotu vektoru $\boldsymbol{e}$ ako:

$$\forall k \ \in J \colon e_k = -\frac{\alpha_k \Gamma(\alpha_k^{-1})}{v_k \Lambda'(\alpha_k^{-1})}.$$

Definícia 1.11. *Nech $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{F}_q^n$, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n), \boldsymbol{y} = (y_1, y_2, \ldots, y_n)$. Hviezdový produkt $\boldsymbol{x} \star \boldsymbol{y}$ definujeme ako:*

$$\boldsymbol{x} \star \boldsymbol{y} = (x_1 y_1, x_2 y_2, \ldots, x_n y_n).$$

Definícia 1.12. *Nech $\mathcal{C}_1$ je $[n,k_1]$ kód a $\mathcal{C}_2$ je $[n,k_2]$ kód. Hviezdovým produktom kódov $\mathcal{C}_1, \mathcal{C}_2$ označíme kód $\mathcal{C}_1 \star \mathcal{C}_2$, definovaný:*

$$\mathcal{C}_1 \star \mathcal{C}_2 = \langle \{ \boldsymbol{x} \star \boldsymbol{y} \mid \boldsymbol{x} \in \mathcal{C}_1, \boldsymbol{y} \in \mathcal{C}_2 \} \rangle.$$

Ak $\mathcal{C}_1 = \mathcal{C}_2$ hovoríme o hviezdovej mocnine kódu $\mathcal{C}_1$ a značíme $\mathcal{C}_1^2$.

Tvrdenie 1.10. *Nech $\mathcal{C}_1$ je $[n,k_1]$ kód a $\mathcal{C}_2$ je $[n,k_2]$ kód. Potom:*

$$\dim(\mathcal{C}_1 \star \mathcal{C}_2) \leq \min\{k_1 k_2, n\} \text{ a } \dim(\mathcal{C}_1^2) \leq \min\left\{ \binom{k_1+1}{2}, n \right\}.$$

Dôkaz. Nech $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_{k_1}$ je báza $\mathcal{C}_1$, a nech $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_{k_2}$ je báza $\mathcal{C}_2$. Vidno, že každý vektor v hviezdovom produkte kódov je generovaný vektormi hviezdového produktu dvojíc $\boldsymbol{a}_i, \boldsymbol{b}_j$. Tých je $k_1 k_2$. V prípade, že sa jedná o hviezdovú mocninu kódu, zo symetrie zložkového súčinu berieme do úvahy $k(k-1)/2$ dvojíc rôznych vektorov a k dvojíc „rovnakých“ vektorov – druhé mocniny bázových vektorov. V súčte je to $k(k+1)/2$ vektorov.

❑

Tvrdenie 1.11. *Nech $\mathcal{C}_{GRS}$ je určený generujúcou maticou G ako v dôsledku 1.9. Nech $k \leq (n+1)/2$. Potom $\mathcal{C}_{GRS}^2$ je GRS kód s rovnakými lokátormi ako $\mathcal{C}_{GRS}$, dimenzia $\mathcal{C}_{GRS}^2 = 2k-1$ a pre multiplikátory $v_1'', v_2'', \ldots, v_n''$ pri generujúcej matici kódu $\mathcal{C}_{GRS}^2$ platí vzťah:*

$$\forall i \ \in \mathbb{N}, i \ \leq n \colon v_i'' = (v_i')^2.$$

Dôkaz. Z polynomiálneho náhľadu na GRS kódy dostávame, že súčin dvoch prvkov bázy:

$$\boldsymbol{a}, \boldsymbol{b} \in \mathcal{C}_{GRS}, \boldsymbol{a} = (v_1' p(\alpha_1), v_2' p(\alpha_2), \ldots, v_n' p(\alpha_n)), \boldsymbol{b} = (v_1' q(\alpha_1), v_2' q(\alpha_2), \ldots, v_n' q(\alpha_n))$$

je v tvare:

$$\boldsymbol{a} \star \boldsymbol{b} = ((v_1')^2 p(\alpha_1) q(\alpha_1), (v_2')^2 p(\alpha_2) q(\alpha_2), \ldots, (v_n')^2 p(\alpha_n) q(\alpha_n)).$$

$p(\alpha_i)q(\alpha_i)$ je taktiež vyhodnotenie polynómu v bode α_i. Z $\deg(p) \leq k-1, \deg(q) \leq k-1$ plynie $\deg(pq) \leq 2k-2$ a teda prvky hviezdovej mocniny bázy nenagenerujú prvok, ktorý by zodpovedal polynómu s vyšším stupňom. Pretože ľubovoľný polynóm stupňa najviac $2k-2$ sa dá napísať ako lineárna kombinácia súčinov polynómov stupňov menších, alebo rovných $k-1$, a pretože každý prvok v hviezdovom produkte kódov je generovaný vektormi zložkového súčinu dvojíc prvkov bázy, tak každému polynómu a, $\deg(a) \leq 2k-2$ viem priradiť jedno kódové slovo a teda dimenzia kódu $\mathcal{C}_{GRS}^2$ je rovná $2k-1$.

❑

Definícia 1.13. *Nech $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód, nech $\mathcal{I} \subsetneq \{1, 2, \ldots, n\}$. Prepichnutým kódom $\mathcal{C}_{\mathcal{I}}$ rozumieme kód:*

$$\mathcal{C}_{\mathcal{I}} = \{(c_{p_1}, \ldots, c_{p_l}) \mid \{p_1, \ldots, p_l\} = \{1, \ldots, n\} \setminus \mathcal{I}, p_1 < \ldots < p_l, (c_1, \ldots, c_n) \in \mathcal{C}\}.$$

Lemma 1.12. *Nech $\mathcal{C}$ je ako v definícii 1.10, nech $\mathcal{I} \subsetneq \{1, 2, \ldots, n\}$. Prepichnutý kód $\mathcal{C}_\mathcal{I}$ je GRS kód.*

Dôkaz. Tvrdenie plynie z tvaru generujúcej matice GRS kódu – ak vynecháme dané stĺpce dostávame znova generujúcu maticu GRS kódu.

❑

Definícia 1.14. *Nech $\mathcal{C}$ je $[n,k,d]_q$ kód, nech $\mathcal{I} \subsetneq \{1, 2, \ldots, n\}$. Označme:*

$$\mathcal{C}(\mathcal{I}) = \{\boldsymbol{c} \in \mathcal{C} \mid \forall i \ \in \mathcal{I} \colon c_i = 0\}.$$

Potom skráteným kódom $\mathcal{C}^\mathcal{I}$ rozumieme kód $\mathcal{C}(\mathcal{I})_\mathcal{I}$.

Lemma 1.13 ([12], Th. 1.5.7)**.** *Nech $\mathcal{C}$ je $[n,k,d]_q$ kód, nech $\mathcal{I} \subsetneq \{1, 2, \ldots, n\}$. Potom $(\mathcal{C}^\perp)^\mathcal{I} = (\mathcal{C}_\mathcal{I})^\perp$ a $(\mathcal{C}^\perp)_\mathcal{I} = (\mathcal{C}^\mathcal{I})^\perp$.*

Dôkaz. Dokazujeme rovnosť množín, dokážeme teda dve inklúzie.
„$(\mathcal{C}^\perp)^\mathcal{I} \subseteq (\mathcal{C}_\mathcal{I})^\perp$: “
Nech $\boldsymbol{c} \in \mathcal{C}^\perp$, nech $\boldsymbol{c}$ je nulové na pozíciách $\mathcal{I}$. Potom $\boldsymbol{c}'$, ktoré získame z $\boldsymbol{c}$ prepichnutím v $\mathcal{I}$, spĺňa $\boldsymbol{c}' \in (\mathcal{C}^\perp)^\mathcal{I}$. Pre $\boldsymbol{x} \in \mathcal{C}$ (a $\boldsymbol{x}' \in \mathcal{C}_\mathcal{I}$) máme

$$\boldsymbol{x} \cdot \boldsymbol{c} = 0 \Rightarrow \boldsymbol{x}' \cdot \boldsymbol{c}' = 0 \Rightarrow \boldsymbol{c}' \in (\mathcal{C}_\mathcal{I})^\perp \Rightarrow (\mathcal{C}^\perp)^\mathcal{I} \subseteq (\mathcal{C}_\mathcal{I})^\perp.$$

„$(\mathcal{C}_\mathcal{I})^\perp \subseteq (\mathcal{C}^\perp)^\mathcal{I}$: “
Ak vezmeme $\boldsymbol{c}' \in (\mathcal{C}_\mathcal{I})^\perp$, vieme z neho odvodiť $\boldsymbol{c}^*$ pridaním núl na pozície z $\mathcal{I}$. Potom pre $\boldsymbol{x} \in \mathcal{C}$ (a $\boldsymbol{x}' \in \mathcal{C}_\mathcal{I}$) máme

$$\boldsymbol{c}' \cdot \boldsymbol{x}' = 0 \Rightarrow \boldsymbol{c}^* \cdot \boldsymbol{x} = 0 \Rightarrow \boldsymbol{c}^* \in \mathcal{C}^\perp \Rightarrow \boldsymbol{c}' \in (\mathcal{C}^\perp)^\mathcal{I} \Rightarrow (\mathcal{C}_\mathcal{I})^\perp \subseteq (\mathcal{C}^\perp)^\mathcal{I}.$$

Druhú rovnosť získame z faktu $(\mathcal{C}^\perp)^\perp = \mathcal{C}$ a aplikáciou prvého prípadu.

❑

Lemma 1.14. *Nech $\mathcal{C}$ je $[n,k,d]_q$ MDS kód (tj. $d = n - k + 1$). Ak skrátime kód $\mathcal{C}$ o i-tu súradnicu, dostaneme $[n-1,k-1,n-k+1]_q$ MDS kód.*

Dôkaz. Najprv ukážeme, že existuje prvok $\mathcal{C}$, ktorý má na i-tej súradnici nenulovú hodnotu. Sporom – potom sa v generujúcej matici na i-tej súradnici nachádza nulový stĺpec. Pretože duálny kód k MDS kódu je tiež MDS (lemma 1.4), tak platí, že v generujúcej matici je každá k-tica stĺpcov lineárne nezávislá – spor.

Označme teda prvok, ktorý má na i-tej súradnici nenulovú hodnotu $\boldsymbol{b}_1$. Doplníme ho na bázu $\mathcal{C}$ prvkami $\boldsymbol{b}_l$, pre $l = 2 \ldots k$, ktoré majú na i-tej súradnici hodnotu 0. Označme túto bázu B. Nech $\boldsymbol{a} \in \mathcal{C}$, potom:

$$\boldsymbol{a} = \sum_{l=1}^{k} x_l \boldsymbol{b}_l \Rightarrow (\boldsymbol{a} \in \mathcal{C}^{\{i\}} \Leftrightarrow x_1 = 0).$$

Teda ľubovoľný prvok skráteného kódu nagenerujeme z $k-1$ vektorov. Súčasne každá lineárna kombinácia prvkov $\boldsymbol{b}_2, \boldsymbol{b}_3, \ldots, \boldsymbol{b}_k$ bude mať na i-tej pozícii hodnotu 0.

Ostáva ukázať, že tento kód je MDS, využijeme lemmu 1.2. Vezmeme generujúcu maticu G danú prvkami B, generujúcu maticu skráteného kódu G' získame škrtnutím j-teho riadku a i-teho stĺpca. Tvrdíme, že každá $(k-1)$-tica stĺpcových vektorov je lineárne nezávislá – sporom. Nech existuje $(k-1)$-tica lineárne závislých stĺpcových vektorov v G'. Potom by ale v G táto $(k-1)$-tica spolu so stĺpcom i, ktorý je nulový až na j-tu súradnicu, tvorila lineárne závislú množinu – spor.

❑

Kapitola 2

Kryptografický systém

2.1 McEliece-ov a Niederreiterov kryptografický systém

Po vzniku novej vetvy kryptografie – asymetrickej kryptografie, McEliece prezentoval v [1] asymetrický kryptografický systém založený na probléme dekódovania všeobecného lineárneho kódu (NP-úplný problém, viď [10]). Bohužiaľ nebolo dokázané, že prelomenie McEliecovho kryptosystému je NP-úplný problém (viď [13]). Výhodami tohto kryptosystému sú:

- Efektívne šifrovanie, dešifrovanie (viď [1]);
- odolnosť voči kvantovým algoritmom (viď [4]);
- neexistencia efektívneho útoku na tento kryptosystém za predpokladu, že využíva Goppa kódy.

Nanešťastie, tomuto kryptosystému je vlastná rada nevýhod, ktoré v súčasnosti zabraňujú jeho použitiu v praxi. Hlavnými sú:

- Veľkosť súkromého kľúča,
- nemožnosť použitia kryptosystému ako podpisovej schémy.

Tieto nedostatky sa pokúsil odstrániť Niederreiter v [5]. Prvou zmenou bolo reprezentovanie správy ako chybového vektoru namiesto informačného slova a náhodnej chyby. Ďalej, namiesto Goppa kódov použil práve GRS kódy, čím znížil nielen veľkosť kľúča, ale aj bezpečnosť kryptosystému, ako ukázali Sidelnikov a Šestakov v [7]. Na druhú stranu, v [6] bolo ukázané, že sú tieto dva systémy rovnako bezpečné za predpokladu zhody použitého kódu $\mathcal{C}$. Pre úplnosť treba spomenúť, že existujú aj návrhy na vylepšenie tohto kryptosystému ([14], [15] – proti tomuto bol publikovaný útok: [16]). Špeciálne návrhu [8] sa venuje táto práca a prezentuje útok na tento kryptosystém podľa článku [9].

Nasleduje formálny popis kryptosystémov.

Definícia 2.1. *Nech* t,$n \in \mathbb{N}, t \ll n$. *McEliecovým kryptosystémom rozumieme päticu* $(\boldsymbol{P}, \boldsymbol{C}, \boldsymbol{K}, \boldsymbol{E}, \boldsymbol{D})$, *kde:*

- $\boldsymbol{P} = \mathbb{F}_q^k$ *je množina otvorených textov,*

- $\boldsymbol{C} = \mathbb{F}_q^n$ *je množina šifrových textov,*
- $\boldsymbol{K}$ *je množina kľúčov. Každý kľúč sa skladá z verejnej a súkromnej zložky, súkromná zložka je v tvare:*
  - S *: náhodná regulárna matica typu* $k \times k$ *nad* $\mathbb{F}_q$*,*
  - G *: generujúca matica kódu* $\mathcal{C}$ *typu* $k \times n$*, kde* $\mathcal{C}$ *je lineárny* $[n,k,d]_q$ *kód, ktorý opraví až* t *chýb, a pre ktorý poznáme efektívny dekódovací algoritmus* $\mathcal{D}_\mathcal{C}$*,*
  - P *: náhodná permutačná matica typu* $n \times n$*.*

  Verejným kľúčom je potom matica $G^{pub} = SGP$ *a číslo* $t \geq 1$*.*
- $\boldsymbol{E} = \{e_{key} \mid key \in \boldsymbol{K}\}$ *je množina šifrovacích algoritmov, kde* $e_{key} \colon \boldsymbol{P} \to \boldsymbol{C}$ *je definované vzťahom* $e_{key}(\boldsymbol{p}) = \boldsymbol{p}G_{key}^{pub} + \boldsymbol{e}$*, pre* $\boldsymbol{e} \in \mathbb{F}_q^n$ *náhodný vektor váhy menšej ako* t*.*
- $\boldsymbol{D} = \{d_{key} \mid key \in \boldsymbol{K}\}$ *je množina dešifrovacích algoritmov, kde* $d_{key} \colon \boldsymbol{C} \to \boldsymbol{P}$ *je definované touto postupnosťou krokov:*
  1. *Pre šifrový text* $\boldsymbol{c}$ *(a pôvodnú správu* $\boldsymbol{m}$*) najprv spočítame* $\boldsymbol{c}P_{key}^{-1} = \boldsymbol{m}S_{key}G_{key} + \boldsymbol{e}P_{key}^{-1}$*.*
  2. *Použijeme dekódovací algoritmus* $\mathcal{D}_{\mathcal{C}_{key}}$ *a získame* $\boldsymbol{m}S_{key}$*.*
  3. *V závere spočítame* $\boldsymbol{m} = \boldsymbol{m}S_{key}S_{key}^{-1}$*.*

Lemma 2.1. *Dešifrovanie v McElieceovom systéme funguje.*

Dôkaz. Pre šifrový text $\boldsymbol{c}$ a pôvodnú správu $\boldsymbol{m}$ máme:

$$\boldsymbol{c}P_{key}^{-1} = \boldsymbol{m}G_{key}^{pub}P_{key}^{-1} + \boldsymbol{e}P_{key}^{-1} = \boldsymbol{m}S_{key}G_{key} + \boldsymbol{e}P_{key}^{-1}.$$

Váha vektoru $\boldsymbol{e}$ sa permutáciou nezmení, a pretože predpokladáme že bola menšia alebo rovná ako t, efektívnym dekódovacím algoritmom $\mathcal{D}_{\mathcal{C}_{key}}$ získame $\boldsymbol{m}S_{key}$. Pretože maticu S_{key} sme volili regulárnu, existuje k nej inverzná matica a teda $\boldsymbol{m} = \boldsymbol{m}S_{key}S_{key}^{-1}$.

❑

Definícia 2.2. *Nech* $t,n \in \mathbb{N}, t \ll n$*. Niederreiterovým kryptosystémom rozumieme päticu* $(\boldsymbol{P}, \boldsymbol{C}, \boldsymbol{K}, \boldsymbol{E}, \boldsymbol{D})$*, kde:*

- $\boldsymbol{P} = \{\boldsymbol{e} \mid \boldsymbol{e} \in \mathbb{F}_q^n, w(\boldsymbol{e}) \leq t\}$ *je množina otvorených textov (správa je reprezentovaná ako chybový vektor váhy menšej, alebo rovnej* t*),*
- $\boldsymbol{C} = \mathbb{F}_q^{n-k}$ *je množina šifrových textov,*
- $\boldsymbol{K}$ *je množina kľúčov, každý kľúč sa skladá z verejnej a súkromnej zložky, súkromná zložka je v tvare:*
  - M *: regulárna matica typu* $(n-k) \times (n-k)$ *nad* $\mathbb{F}_q$ *s náhodnými prvkami,*

– *H : kontrolná matica kódu $\mathcal{C}$ typu $(n-k)\times n$, kde $\mathcal{C}$ je lineárny $[n,k,d]_q$ kód, ktorý opraví až t chýb a pre ktorý poznáme efektívny dekódovací algoritmus $\mathcal{D}_\mathcal{C}$,*

– *P : náhodná permutačná matica typu $n \times n$.*

Verejným kľúčom je potom matica $H^{pub} = MHP$ a číslo t.

- $\boldsymbol{E} = \{e_{key} \mid key \in \boldsymbol{K}\}$ *je množina šifrovacích algoritmov, kde* $e_{key}\colon \boldsymbol{P} \to \boldsymbol{C}$ *je definované vzťahom* $e_{key}(\boldsymbol{e}) = H^{pub}_{key}\boldsymbol{e}^T$,
- $\boldsymbol{D} = \{d_{key} \mid key \in \boldsymbol{K}\}$ *je množina dešifrovacích algoritmov, kde* $d_{key}\colon \boldsymbol{C} \to \boldsymbol{P}$ *je definované touto postupnosťou krokov:*
  1. *Pre šifrový text $\boldsymbol{c}$ (a pôvodnú správu $\boldsymbol{e}$) najprv spočítame $M^{-1}_{key}\boldsymbol{c} = H_{key}P_{key}\boldsymbol{e}^T$.*
  2. *Použijeme dekódovací algoritmus $\mathcal{D}_{\mathcal{C}_{key}}$ a získame $P_{key}\boldsymbol{e}^T$.*
  3. *Spočítame $\boldsymbol{e}^T = P_{key}P^{-1}_{key}\boldsymbol{e}^T$.*

Poznámka 2.1. Dešifrovanie v Niederreiterovom systéme funguje. Máme totiž pre šifrový text $\boldsymbol{c}$ a pôvodnú správu $\boldsymbol{e}$:

$$M^{-1}_{key}\boldsymbol{s} = M^{-1}_{key}H^{pub}_{key}\boldsymbol{e}^T = M^{-1}_{key}M_{key}H_{key}P_{key}\boldsymbol{e}^T = H_{key}P_{key}\boldsymbol{e}^T.$$

Ďalej, použijeme efektívny dekódovací algoritmus $\mathcal{D}_{\mathcal{C}_{key}}$ a získame $P_{key}\boldsymbol{e}^T$, pretože permutácia daná maticou P_{key} neovplyvní váhu $\boldsymbol{e}^T$. Nakoniec nájdeme inverznú permutáciu P^{-1}_{key} a získame $\boldsymbol{e} = P^{-1}_{key}P_{key}\boldsymbol{e}^T$.

2.2 Wieschebrinkova modifikácia

V [8] Wieschebrink navrhol spôsob, akým Niederreiterov kryptosystém „ubrániť“ proti Sidelnikov-Šestakovmu útoku, a to pomocou nasledujúceho problému (tzv. EPC problém – Equivalent Punctured Code).

Problém 2.2. *Nech je daná matica M typu $k \times n$ a matica H typu $k \times m$ kde $m \leq n$. Existuje regulárna matica T typu $k \times k$ a množina $I \subsetneq \{1,2,\ldots,n\}$, $|I| = n - m$ tak , že platí:*

$$(TM)_I = TM_I = H?$$

Wieschebrink v [8] tiež ukázal, že tento problém je NP-úplný. Najprv si všimnime, ako tento problém využil pri modifikácii McEliecovho kryptosystému.

Definícia 2.3. *Nech $KS_{McEliece} = \{\boldsymbol{P}, \boldsymbol{C}, \boldsymbol{K}, \boldsymbol{E}, \boldsymbol{D}\}$ a $\mathcal{C}$,t,n ako v definícii 2.1. Wieschebrinkovou modifikáciou McEliecovho kryptosystému budeme rozumieť päticu $\{\boldsymbol{P}_W, \boldsymbol{C}_W, \boldsymbol{K}_W, \boldsymbol{E}_W, \boldsymbol{D}_W\}$, kde:*

- $\boldsymbol{P}_W = \boldsymbol{P}$,
- $\boldsymbol{C}_W = \mathbb{F}_q^{n+r}$,
- $\boldsymbol{K}_W$ *analogicky ako v definícii 2.1, s týmito zmenami:*

– *Zvoľme* $r \in \mathbb{N}$, *zvoľme* $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$ *náhodné vektory dĺžky* k *a zvoľme* $p_1, \ldots, p_r \in \mathbb{N}$ *tak , že* $1 \leq p_1 < p_2 < \ldots < p_r \leq n + r$ *(pozície na vloženie stĺpcových vektorov* $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$*).*

– *Namiesto generujúcej matice* G *kódu* $\mathcal{C}$*, použijeme generujúcu maticu* G'*, ktorú získame z* G *vložením vektorov* $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$ *na pozície* $p_1, p_2, p_3, \ldots, p_r \in \mathbb{N}$*. Označme tento nový kód* $\mathcal{C}'$*.*

– P *nech je náhodná permutačná matica typu* $(n + r) \times (n + r)$*. Maticu* G^{pub} *získame ako* $G^{pub} = SG'P$*.*

– *Do súkromného kľúča pribudnú hodnoty* $r, p_1, p_2, \ldots, p_r$*.*

- $\boldsymbol{E}_W$ *analogicky, ako v definícii 2.1 s maticou* G^{pub}_{key}*.*

- $\boldsymbol{D}_W$ *analogicky, medzi kroky 1 a 2 pridáme krok:*

  – *Z vektoru* $\boldsymbol{c}P'^{-1}_{key}$ *vynechaj zložky* $p_1, p_2, \ldots, p_r$ *určené kľúčom key.*

Poznámka 2.2. Môžeme si všimnúť niekoľko vecí:

- Množina otvorených textov sa nezmenila – to vyplýva z toho, že pridávame len „nadbytočné“ zložky vektoru, na zmätenie útočníka a nemeníme kód ako taký.

- Ak vynecháme „nadbytočné“ zložky vektoru, je možné použiť efektívny dekódovací algoritmus $\mathcal{D}_{\mathcal{C}_{key}}$ pôvodného kódu $\mathcal{C}_{key}$.

- t ostáva nezmenené: nemôžeme predpokladať, že náhodný chybový vektor bude nenulový v niektorých „nadbytočných“ zložkách, pretože pozície náhodných „nadbytočných“ vektorov sú tajné (ostatne na ich utajení a hľadaní je problém 2.2 založený).

- Dešifrovanie funguje, lebo prepichnutím slova $\boldsymbol{c}P'^{-1}_{key}$ na pozíciách $p_1, p_2, \ldots, p_r$ získame slovo, z ktorého vieme zrekonštruovať správu efektívnym dekódovacím algoritmom $\mathcal{D}_{\mathcal{C}_{key}}$ (a potom už len použijeme inverz regulárnej matice rovnako ako v prípade McElieceovho kryptosystému). To plynie z tvaru generujúcej matice a faktu, že kód $\mathcal{C}_{key}$ zvládne opraviť t chýb a prepichnutím slova tento počet nezvýšime, skôr naopak.

Teraz sa pozrime, ako vieme tento problém využiť na úpravu Niederreiterovho kryptosystému. Postupujeme podobne: zvolíme číslo r, vektory $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$ pridáme ako stĺpce na náhodné pozície (nech sú to pozície $p_1, p_2, \ldots, p_r$) do *generujúcej* matice G kódu $\mathcal{C}$, čím vznikne matica G' kódu $\mathcal{C}'$. Pomocou lineárnej algebry vieme dopočítať kontrolnú maticu H' tohto kódu typu $(n-k+r) \times (n+r)$. To vedie k definícii Wieschebrinkovej verzie Niederreiterovho systému.

Definícia 2.4. *Nech* $KS_{Niederreiter} = \{\boldsymbol{P}, \boldsymbol{C}, \boldsymbol{K}, \boldsymbol{E}, \boldsymbol{D}\}$ *a* $\mathcal{C}$*,*t*,*n *ako v definícii 2.2. Wieschebrinkovou modifikáciou Niederreiterovho kryptosystému budeme rozumieť päticu* $\{\boldsymbol{P}_W, \boldsymbol{C}_W, \boldsymbol{K}_W, \boldsymbol{E}_W, \boldsymbol{D}_W\}$*, kde:*

- $\boldsymbol{P}_W = \{\boldsymbol{e} \in \mathbb{F}_q^{n+r} \mid w(\boldsymbol{e}) \leq t\}$,

- $\boldsymbol{C}_W = \mathbb{F}_q^{n+r}$,

- $\boldsymbol{K}_W$ *analogicky ako v definícii 2.2, s týmito zmenami:*
  - *Zvoľme* $r \in \mathbb{N}$*, zvoľme* $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$ *náhodné vektory dĺžky* k *a zvoľme* $p_1, \ldots, p_r \in \mathbb{N}$ *tak , že* $1 \leq p_1 < p_2 < \ldots < p_r \leq n + r$ *(pozície na vloženie stĺpcových vektorov* $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$*).*
  - *Spočítame generujúcu maticu* G'*, ktorú získame z* G *vložením vektorov* $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$ *na pozície* $p_1, p_2, \ldots, p_r \in \mathbb{N}$ *do matice* G*. Označme tento nový kód* $\mathcal{C}'$*. Spočítame jeho kontrolnú maticu* H'*.*
  - *Zvolíme maticu* M *ako náhodnú reguláru typu* $(n-k+r) \times (n-k+r)$*.*
  - *Namiesto kontrolnej matice kódu* $\mathcal{C}$*, použijeme maticu* H'*.*
  - P *nech je náhodná permutačná matica typu* $(n+r) \times (n+r)$*. Maticu* H^{pub} *získame ako* $H^{pub} = MH'P$*.*
  - *Do súkromného kľúča pribudnú hodnoty* $r, p_1, p_2, \ldots, p_r$ *a matica* G'*.*
- $\boldsymbol{E}_W$ *analogicky ako v definícii 2.2 s maticou* H^{pub}*.*
- $\boldsymbol{D}_W$ *analogicky, použijeme nasledujúci algoritmus:*
  1. *Pre šifrový text* $\boldsymbol{c}$ *(a pôvodnú správu* $\boldsymbol{e}$*) najprv spočítame* $M_{key}^{-1}\boldsymbol{c} = H'_{key}P_{key}\boldsymbol{e}^T$*.*
  2. *Použitím lineárnej algebry nájdeme* $P_{key}\boldsymbol{e}'$ *také, že* $H'_{key}P_{key}\boldsymbol{e}'^T = H'_{key}P_{key}\boldsymbol{e}^T$*.*
  3. *Prepichneme* $P_{key}\boldsymbol{e}'^T$ *v zložkách* $p_1, p_2, \ldots, p_r$ *a dostaneme* $\boldsymbol{e}^*$*.*
  4. *Použijeme dekódovací algoritmus* $\mathcal{D}_{\mathcal{C}_{key}}$ *na* $\boldsymbol{e}^*$ *a získame* $\boldsymbol{a}$ *– informačné slovo ku kódovému slovu k prijatému slovu* $\boldsymbol{e}^*$*.*
  5. *Spočítame* $P_{key}\boldsymbol{e}^T$ *ako* $P_{key}\boldsymbol{e}' - \boldsymbol{a}G'_{key} = P_{key}\boldsymbol{e}^T$*.*
  6. *Nakoniec spočítame* $\boldsymbol{e}^T = P_{key}^{-1}P_{key}\boldsymbol{e}^T$*.*

Poznámka 2.3. Prvé tri body predchádzajúcej poznámky platia aj v tomto prípade (s výnimkou tvrdenia o otvorených textoch, stále ale opravujeme len t chýb). Ostáva ukázať, že dešifrovanie funguje. Vidno (pre šifrový text $\boldsymbol{c}$ a pôvodnú správu $\boldsymbol{e}$):

$$M_{key}^{-1}\boldsymbol{c} = H'_{key}P_{key}\boldsymbol{e}^T.$$

Vektor $\boldsymbol{e}'$ (z kroku 2) sa nám podarí nájsť, pretože pracujeme nad telesom a vieme Gaussovou eliminačnou metódou vyriešiť sústavu $H'_{key}P_{key}\boldsymbol{e}'^T = M_{key}^{-1}\boldsymbol{c}$. Z definície 1.9 a poznámky za ňou platí:

$$P_{key}\boldsymbol{e}'^T = \boldsymbol{a}G'_{key} + P_{key}\boldsymbol{e}^T.$$

Prepichnutím $P_{key}\boldsymbol{e}'$ na $\boldsymbol{e}^*$ a následnou aplikáciou $\mathcal{D}_{\mathcal{C}_{key}}$ získame $\boldsymbol{a}$ ako informačné slovo ku kódovému slovu k prijatému slovu $\boldsymbol{e}^*$. Spočítame $P_{key}\boldsymbol{e}^T = P_{key}\boldsymbol{e}'^T - \boldsymbol{a}G'_{key}$, a potom už len aplikujeme inverznú permutáciu.

Kapitola 3

Útok na Wieschebrinkovu variantu Niederreitovho systému

Útok sa bude venovať pôvodnej variante - ako samoopravný kód bude použitý GRS kód. Cieľom útoku je schopnosť dešifrovať šifrované správy - získame kľúč, ktorý definuje rovnaké zobrazenia e_{key} a d_{key}, aj keď sa môže od použitého kľúča líšiť. Útok bude spadať do kategórie CPA - Chosen Plaintext Attack, tj. útočník má prístup k šifrovému textu pre ľubovoľný ním vybraný otvorený text. Tento predpoklad je rozumný, nakoľko sa jedná o asymetrický kryptografický systém.

3.1 Pozície náhodných vektorov

Lemma 3.1. *Nech KS je ako v definícii 2.4, nech key je kľúč tohto kryptosystému. Potom existuje kľúč key′ tak , že platí:*

1. $e_{key} = e_{key'}$ *a* $d_{key} = d_{key'}$*;*
2. $P_{key'}$ *je matica identity typu* $(n + r) \times (n + r)$*.*

Dôkaz. Permutačná matica P_{key} permutuje stĺpce matice H'_{key} – po permutácii získame kontrolnú maticu permutačne ekvivalentného kódu, tj. tiež GRS-kódu s r náhodnými vektormi. Pretože násobenie matíc (na vektor sa môžeme pozerať ako na maticu typu $1 \times n$) je asociatívne, môžeme predpokladať že pracujeme s permutačne ekvivalentným kódom a matica P_{key} je identita.

❑

V nasledujúcom texte je popísaný útok na ľubovoľný pevne zvolený kľúč – pre jednoduchosť a prehľadnosť nebude znázorňovaná príslušnosť ku kľúču. Aj keď útokom získame kľúč, ktorý sa môže líšiť od skutočného kľúča použitého pri konštrukcii kryptosystému, pre jednoduchosť a prehľadnosť budú ich zložky značené rovnako a to podľa definície 2.4.

Z asociativity násobenia matíc vyplynie, že prenásobenie regulárnou maticou M nezmení jadro kontrolnej matice H^{pub} – je možné spočítať jeho bázu a získame tak generujúcu maticu G', ktorá generuje rovnaký kód ako matica G s r pridanými náhodnými vektormi (tento kód je v definícii 2.4 označený $\mathcal{C}'$). Označme túto bázu $\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_k$. Navyše, kód daný touto generujúcou maticou „kontroluje“

matica H^{pub} (v zmysle – kódové slová sa zobrazia na nulový vektor, o nenulových syndrómoch veľa povedať nevieme).

Stĺpce G' rozdelíme na dve skupiny:

Definícia 3.1. *Povieme, že i-ty stĺpec matice G' je kódový, ak sa v matici G na i-tej pozícii nevyskytoval náhodný vektor. Analogicky povieme, že i-ty stĺpec matice G' je náhodný, ak sa v matici G na i-tej pozícii náhodný vektor vyskytoval (ide práve o hodnoty pozícii $p_1, p_2, \ldots, p_r$ z definície 2.4).*

Ostáva zistiť, ktoré vektory sú náhodné a ktoré nie. Pri hľadaní pozícií náhodných vektorov vyjdeme z článku [9].

Definícia 3.2. *Nech A,B sú matice typu $m \times n$, $k \times n$. Symbolom $A \star B$ budeme značiť maticu typu $l \times n$, ktorá je generujúcou maticou hviezdového produktu kódov určených maticami A a B.*

Definícia 3.3. *Nech G je generujúca matica $[n,k,d]_q$ kódu $\mathcal{C}$, nech $A \subseteq \{1,2,\ldots,n\}$. Potom symbolom G^A bude značená generujúca matica kódu $\mathcal{C}^A$ (skráteného kódu) a symbolom G_A bude značená generujúca matica kódu $\mathcal{C}_A$ (prepichnutého kódu).*

Tvrdenie 3.2. *Nech $\mathcal{C}$ je $[n+r,k]$ kód s generujúcou maticou G', ktorá vznikla pridaním náhodných vektorov $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$ na pozície $p_1, p_2, \ldots, p_r$ matice G - generujúcej matice GRS kódu. Nech $k < (n-r)/2$. Potom:*

$$2k - 1 \leq \dim(\mathcal{C}^2) \leq 2k + r - 1.$$

Dôkaz. Bez ujmy na všeobecnosti, nech $p_1, p_2, \ldots, p_r$ sú pozície $n+1$ až $n+r$ (permutácia stĺpcov nemení hodnosť matice). Vezmime generujúcu maticu kódu $\mathcal{C}^2$, označme ju $G^\star$. Jej hodnosť je dimenziou kódu $\mathcal{C}^2$.

Pretože GRS kód je daný nenulovými prvkami, a z tvrdenia 1.11 má dimenziu rovnú $2k-1$, tak ak budú náhodné vektory všetky nulové, bude hodnosť minimálna a to $2k-1$.

Naopak, pretože prvých n stĺpcov matice $G^\star$ má hodnosť rovnú $2k-1$, tak celková hodnosť matice nemôže presiahnuť $2k-1+r$.

❑

Couvreur et. al. v [9] ďalej experimentálne zistili, že platí $\dim(\mathcal{C}^2) = 2k+r-1$ skoro vždy. Na tomto výsledku založili útok na kryptosystém, ktorý by zlyhal bez tohto predpokladu. V nasledujúcej hypotéze je tomuto experimentálnemu výsledku venovaná väčšia pozornosť.

Hypotéza 3.3. *Nech $\mathcal{C}'$ je $[n+r,k]_q$ kód určený generujúcou maticou G', ktorá vznikla vložením náhodných vektorov $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_r$ na pozície $p_1, p_2, \ldots, p_r$ generujúcej matice G GRS kódu $\mathcal{C}$. Nech $k < (n-r)/2$. Potom $\dim((\mathcal{C}')^2) = 2k-1+r$ s pravdepodobnosťou blížiacou sa k 1.*

Úvahy o možnom dôkaze. Máme danú generujúcu maticu G', môžeme bez ujmy na všeobecnosti predpokladať, že náhodné stĺpce sa nachádzajú na pozíciách $n+1, n+2, \ldots, n+r$ (pretože zmena poradia stĺpcov nemení hodnosť matice) a označme ich maticu R a jej riadky $\boldsymbol{r}_1$ až $\boldsymbol{r}_k$. Ďalej môžeme bez ujmy

na všeobecnosti predpokladať, že generujúca matica kódu $\mathcal{C}$ označená G je v tvare ako v dôsledku 1.9.

Dimenziu kódu $(\mathcal{C}')^2$ hľadáme nasledujúcim spôsobom – vezmeme každú dvojicu riadkov, spočítame ich hviezdový produkt a ten umiestnime do novej matice, označme ju A. Jej rank je potom dimenziou kódu $(\mathcal{C}')^2$. Pretože máme dokopy $\binom{k}{2}$ dvojíc a k „štvorcov“, tak A je typu $\left(\binom{k}{2}+k\right) \times (n+r)$.

Vieme, že i-ty riadok (indexujeme od nuly) matice G_{GRS} je v tvare:

$$(\alpha_1^i v_1 \quad \alpha_2^i v_2 \quad \dots \quad \alpha_n^i v_n).$$

Z toho plynie tvar matice $G \star G$:

$$G \star G = \begin{pmatrix} v_1^2 & v_2^2 & \dots & v_n^2 \\ \alpha_1 v_1^2 & \alpha_2 v_2^2 & \dots & \alpha_n v_n^2 \\ & & \ddots & \\ \dots & \dots & & \dots \\ \alpha_1^{2k-2} v_1^2 & \alpha_2^{2k-2} v_2^2 & \dots & \alpha_n^{2k-2} v_n^2 \end{pmatrix},$$

ako aj jej typ, $(2k-1) \times n$. Ľahko nahliadneme, že hviezdový produkt ľubovoľnej dvojice riadkov je jeden z týchto $2k-1$ vektorov. Označme ich $\boldsymbol{b}_0, \boldsymbol{b}_1, \dots \boldsymbol{b}_{2k-2}$ a ich množinu označme B. Dostaneme práve maticu $G \star G$ aj keď s prepermutovanými riadkami (bez ujmy na všeobecnosti nech je v tvare ako vyššie). Každý ďalší vektor potom na prvých n súradniciach vynulujeme tak , že od neho odpočítame správny prvok B.

Podobne definujme $\boldsymbol{v}_0, \boldsymbol{v}_1, \dots \boldsymbol{v}_{2k-2}$, ale pre stĺpcové súradnice $n+1$ až $n+r$, ich množinu označme B_r. Po vynulovaní prvých n stĺpcových pozícií pre riadky $2k$ až $\binom{k+1}{2}$ pre stĺpcové pozície $n+1$ až $n+r$ (označme ich riadkové vektory $\boldsymbol{z}_{i,j}$) platí:

$$\boldsymbol{z}_{i,j} = \boldsymbol{r}_i \star \boldsymbol{r}_j - \boldsymbol{v}_{\lambda_{i,j}},$$

kde $\lambda_{i,j}$ je vhodné z množiny $\{0,1,\dots 2k-2\}$.

Takto sme upravili prvých n stĺpcov matice A tak, aby sa v prvých $2k-1$ riadkoch nachádzala matica $G \star G$ a na zvyšných nula. Ako sme nahliadli vyššie, týmito úpravami docielime, že v pravom dolnom rohu (tj. stĺpce $n+1$ až $n+r$ a riadky $2k$ a vyššie) sa nachádza práve matica Z, ktorá je tvorená riadkami $\boldsymbol{z}_{i,j}$.

Bolo by treba dokázať, že matica Z má hodnosť r s pravdepodobnosťou limitne blížiacou sa k jednej. To sa mi dokázať nepodarilo. Avšak konštrukcia matice Z naznačuje, že problém jej hodnosti nemusí závisieť na tom, že pôvodná matica G' je odvodená z GRS kódu.

Pre dôkaz hypotézy je nutné sa viac venovať problematike náhodných kódov, čo sú otázky idúce iným smerom ako zameranie tejto práce.

To vedie k postupu, akým rozlíšiť medzi náhodným a kódovým stĺpcom pre danú generujúcu maticu.

Algoritmus 3.4. *Detekcia náhodných vektorov - prípad* $2k-1+r < n$.

Vstup: G', $n+r$, k, i, t.
Výstup: Odpoveď na otázku: Je vektor na mieste i náhodný?

$d \leftarrow \dim(G' \star G')$;
if $\dim(G'_{\{i\}} \star G'_{\{i\}}) = d-1$ **then return** true;

else if $\dim(G'_{\{i\}} \star G'_{\{i\}}) = d$ **then return** false; ▷ S vysokou pr.
end if

Algoritmus 3.5. *Detekcia náhodných vektorov - prípad* $2k - 1 + r \geq n$.

Vstup: G', $n + r$, k, i, t.
Výstup: Odpoveď na otázku: Je vektor na mieste i náhodný?

---

zvoľ a tak , že $a < k$ & $a > d - n$;
zvoľ $A \subset \{1, 2, \ldots, n + r\}$ tak , že $|A| = a$ & $i \notin A$;
$d \leftarrow \dim((G')^A \star (G')^A)$;
if $\dim(((G')^A)_{\{i\}} \star ((G')^A)_{\{i\}}) = d - 1$ **then return** true;
else if $\dim(((G')^A)_{\{i\}} \star ((G')^A)_{\{i\}}) = d$ **then return** false; ▷ S vysokou pr.
end if

Algoritmus 3.6. *Detekcia náhodných vektorov.*

Vstup: G', $n + r$, k, t.
Výstup: Zoznam náhodných pozícií alebo null (v prípade neplatnosti hypotézy 3.3).

$n_1 \leftarrow 2t + k$; $r_1 \leftarrow (n + r) - n_1$;
$n_2 \leftarrow 2t + k + 1$; $r_2 \leftarrow (n + r) - n_2$;
$P \leftarrow \emptyset$;
if $(n + r) < 4t + 1$ **then**
  for $i \leftarrow 1,2, \ldots, (n + r)$ **do**
    Použi algoritmus 3.4 na i;
    if i je náhodný **then**
      Do zoznamu P pridaj i;
    end if
  end for
else if $(n + r) \geq 4t + 3$ **then**
  for $i \leftarrow 1,2, \ldots, (n + r)$ **do**
    Použi algoritmus 3.5 na i;
    if i je náhodný **then**
      Do zoznamu P pridaj i;
    end if
  end for
else
  $P_1 \leftarrow \emptyset$; $r_{check1} \leftarrow 0$;
  $P_2 \leftarrow \emptyset$; $r_{check2} \leftarrow 0$;
  for $i \leftarrow 1,2, \ldots, (n + r)$ **do**
    Použi algoritmus 3.4 na i;
    if i je náhodný **then**
      $r_{check1} \leftarrow r_{check1} + 1$; Do zoznamu P_1 pridaj i;
    end if
    Použi algoritmus 3.5 na i;
    if i je náhodný **then**

$r_{check2} \leftarrow r_{check2} + 1$; Do zoznamu P_2 pridaj i;
end if
end for
if $|P_1| = r_{check1}$ **then**
$P \leftarrow P_1$;
else if $|P_2| = r_{check2}$ **then**
$P \leftarrow P_2$;
else return null;
end if
end if
return P;

Lemma 3.7. *Nech $\mathcal{C}$ je GRS kód s r pridanými náhodnými vektormi dĺžky $n+r$ dimenzie k. Potom algoritmus 3.6 s podalgoritmami 3.4, 3.5 fungujú s vysokou pravdepodobnosťou.*

Dôkaz. Hodnota n je neznáma, ale známa je maximálna hodnota t, ktorá spĺňa

$$t < \frac{n-k+1}{2},$$
$$2t < n-k\ +1.$$

Z toho plynie, že ak $n-k$ je párne, tak $t = (n-k)/2$ a v opačnom prípade $t = (n-k-1)/2$. Dostávame dve možnosti pre hodnotu n, a to n_1 a n_2:

$$n_1 = 2t + k,$$
$$n_2 = 2t + k + 1.$$

Zo znalosti hodnoty $n+r$ tak vieme zistiť príslušné r_1, r_2.

Zaujíma nás platnosť nerovnosti

$$2k - 1 + r < n,$$
$$2k - 1 + (n+r) < 2n,$$
$$(n+r) < 2n - 2k + 1.$$

Teraz využijeme hotnotu t:

$$n_1 \colon (n+r) < 4t + 1,$$
$$n_2 \colon (n+r) < 4t + 3.$$

Vidno, že ak $(n+r) < 4t+1$, alebo $(n+r) \geq 4t+3$, tak vieme ktorý algoritmus máme na detekciu náhodných vektorov použiť. Ak nenastane ani jeden z týchto prípadov, použijeme obidva algoritmy a porovnáme s očakávanými hodnotami r_1, r_2.

Ostáva dokázať funkčnosť algoritmov 3.4, 3.5. Pozrime sa najprv na prípad $2k - 1 + r \leq n$:

Ak i označuje pozíciu náhodného vektoru, očakávame s vysokou pravdepodobnosťou, že prepichnutie kódu v tejto pozícii bude mať za dôsledok

$$\dim(\mathcal{C}^2_{\{i\}}) = \dim(\mathcal{C}^2) + r - 1 = 2k - 2 + r,$$

pretože $\mathcal{C}_{\{i\}}$ je GRS kód s náhodne vloženými náhodnými vektormi o počte $r-1$.

Ak i označuje pozíciu kódového vektoru, potom $\mathcal{C}_{\{i\}}$ je GRS kód dimenzie k s r náhodne vloženými náhodnými vektormi a očakávaná dimenzia $\dim(\mathcal{C}^2_{\{i\}}) = 2k-1+r$ (prepichnutie MDS kódu v jednej súradnici nezníži jeho dimenziu).

V opačnom prípade:

Je možné vybrať také a, pretože $k > a \ > d-n \Leftrightarrow k \ > d-n+1 \Leftrightarrow k \ > 2k+r-n \Leftrightarrow n > k \ +r$, čo nutne platí. Pre a platí $a = a_0 + a_1$, kde a_0 je počet náhodných pozícii v našom výbere a a_1 je počet kódových pozícii v našom výbere.

Vznikol $[n-a_1,k-a_1]$ skrátený GRS kód (lemma 1.14), ku ktorému je pridaných $r-a_0$ náhodných vektorov. Ostáva ukázať, že podmienka na prvú vetvu algoritmu je splnená. To platí, pretože:

$$\begin{aligned} 2k-1+r-n &\leq a, \\ 2k-1+r-n &\leq a_0+a_1, \\ 2(k-a_1)-1+(r-a_0) &\leq n-a_1, \end{aligned}$$

a teda sme to previedli na prvý prípad.

❑

3.2 Zistenie GRS kódu - Sidelnikov-Šestakov algoritmus

Momentálne máme k dispozícii generujúcu maticu G kódu $\mathcal{C}'_R = \mathcal{C}$, kde R je množina pozícii náhodných vektorov – je to generujúca matica GRS kódu, aj keď nie v tvare ako v dôsledku 1.9. My ale chceme maticu v tomto tvare získať (označme ju G_{GRS}), presnejšie chceme hodnoty $\alpha_1, \alpha_2, \ldots, \alpha_n$ a $v'_1, v'_2, \ldots, v'_n$. Hľadáme teda maticu prechodu medzi bázou ktorú máme, a bázou ktorú očakávame. Navyše platí:

Poznámka 3.1.

$$G = S\ G_{GRS},$$

kde S je regulárna matica – to plynie z toho, že riadky G sú bázou rovnakého kódu (rovnakého podpriestoru) ako riadky G_{GRS} a pre každé dve bázy daného podpriestoru platí, že existuje regulárna štovrcová matica (S) prevádzajúca jednu bázu na druhú.

Ostáva nájsť maticu S a maticu G_{GRS}. Našťastie, spôsob na získanie týchto matíc je popísaný v článku [7].

Nasleduje stručný dôkaz správnosti Sidelnikov-Šestakovho algoritmu. Podrobnejšie vysvetlenie sa nachádza v práci [17].

Definícia 3.4. *Nech $\mathbb{F}_q$ je teleso. Projektívnou priamkou nad $\mathbb{F}_q$ rozumieme projektívny priestor dimenzie 1, teda množinu všetkých podpriestorov $\mathbb{F}_q^2$ dimenzie 1, značíme $\mathbb{P}^1(\mathbb{F}_q)$. Prvky tejto množiny nazveme projektívne body. Ak $a \in \mathbb{P}^1(\mathbb{F}_q)$ a $(x,y) \in a$ nenulový, značíme prvok a symbolom $[x:y]$.*

Vidno, že pre všetky $\lambda \in \mathbb{F}_q^*$ platí $[x : y] = [\lambda x : \lambda y]$ a teda skoro každý projektívny bod môžeme jednoznačne zapísať ako $[x' : 1]$. Jedinú priamku, ktorú takto nemôžeme zapísať označíme $[1 : 0]$. Ďalšie stotožnenie je množiny $\mathbb{F}_\infty = \mathbb{F}_q \cup \{\infty\}$ s množinou $\mathbb{P}^1(\mathbb{F}_q)$. Najprv na $\mathbb{F}_\infty$ čiastočne dodefinujme aritmetiku nasledujúcim spôsobom:

$$\forall x \in \mathbb{F}_q^*\colon \frac{x}{0} = \infty,\ x \cdot \infty = \infty;$$
$$\forall x \in \mathbb{F}_q\colon \frac{x}{\infty} = 0,\ x + \infty = \infty.$$

Vidíme, že možno stotožniť $x \in \mathbb{F}_q$ s prvkom $[x : 1]$ a ∞ s prvkom $[1 : 0]$.

Definícia 3.5. *Nech $\mathcal{C}$ je GRS kód s kontrolnou maticou H v tvare ako v definícii 1.10. Ak $\alpha_i = \infty$, potom i-ty stĺpec matice H je tvaru*

$$(0,0,\ldots,0v_i)^T.$$

Ak $\alpha_i = 0$, potom i-ty stĺpec matice H je tvaru

$$(v_i,0,\ldots,0)^T.$$

Definícia 3.6 ([18], str. 102)**.** *Nech $\boldsymbol{G} = (G, \cdot, ^{-1}, 1)$ je grupa a X je neprázdna množina. Pôsobením grupy $\boldsymbol{G}$ na množine X nazveme homomorfizmus $\pi\colon G \to \boldsymbol{S}(X)$.*

Definícia 3.7. *Permutačnou grupou množiny X rozumieme ľubovoľnú podgrupu množiny $\boldsymbol{S}(X)$.*

Definícia 3.8. *Symbolom $\boldsymbol{GL}(n, q)$ rozumieme grupu regulárnych štvorcových matíc typu $n \times n$ nad telesom $\mathbb{F}_q$.*

Lemma 3.8. *Grupa $\boldsymbol{GL}(n, q)$ pôsobí na $\mathbb{P}^1(\mathbb{F}_q)$ a zobrazenie $\pi\colon \boldsymbol{GL}(n, q) \to \mathbb{P}^1(\mathbb{F}_q)$ je definované:*

$$\forall M \in \boldsymbol{GL}(2, q), M = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \forall x \in \mathbb{P}^1(\mathbb{F}_q)\colon \pi(M)(x) = \begin{cases} \frac{ax+b}{cx+d} & x \in \mathbb{F}_q, \\ \frac{a}{c} & x = \infty. \end{cases}$$

Dôkaz. Potrebujeme dokázať, že $\pi(M)$ je permutácia. Na to nám stačí ukázať, že sa jedná o prosté zobrazenie (pretože vedie z konečnej množiny do konečnej množiny). Nech $x_1, x_2 \in \mathbb{P}^1(\mathbb{F}_q)$ a nech $\pi(M)(x_1) = \pi(M)(x_2)$. Ak $x_1 = \infty$, potom nutne $x_2 = \infty$ a naopak, pretože neexistuje také $x \in \mathbb{F}_q$, že:

$$\frac{ax + b}{cx + d} = \frac{a}{c}.$$

Máme teda rovnosť:

$$\frac{ax_1 + b}{cx_1 + d} = \frac{ax_2 + b}{cx_2 + d}.$$

Ak $cx_1 + d = 0$ potom $\frac{ax_1+b}{cx_1+d} = \infty$ a teda $\frac{ax_2+b}{cx_2+d} = \infty$, z čoho plynie $cx_2 + d = 0$. Potom ale $cx_1+d = cx_2+d$ a teda $x_1 = x_2$. V opačnom prípade môžeme prenásobiť

obidve strany rovnosti menovateľmi a dostávame:

$$\begin{aligned}(ax_1+b)(cx_2+d) &= (ax_2+b)(cx_1+d)\\ acx_1x_2+bcx_2+adx_1+bd &= acx_1x_2+bcx_1+adx_2+bd\\ (ad-bc)x_1 &= (ad-bc)x_2\\ x_1 &= x_2.\end{aligned}$$

Posledný krok dostávame z faktu, že M je regulárna matica s nenulovým determinantom, ktorý je rovný práve $ad-bc$.

❑

Nahliadnime, že $\mathbf{Im}(\pi)$ pôsobí prirodzene na projektívnu priamku. Túto grupu označme $\boldsymbol{PLG}(2,\mathbb{F}_q)$.

Definícia 3.9. *Permutačná grupa $\boldsymbol{G}$ na X sa nazýva ostro 3-tranzitívna, ak platí:*

$$\forall(a_1,a_2,a_3),(b_1,b_2,b_3)\in X^3 \text{ kde } a_i\neq a_j, b_i\neq b_j \text{ ak } \ i\ \neq j\ \exists!\phi\in\boldsymbol{G}\colon \phi(a_i)=b_i.$$

Tvrdenie 3.9 ([17] str. 5)**.** *Grupa $\boldsymbol{PGL}(2,q)$ na $\mathbb{F}_\infty$ je ostro 3-tranzitívna.*

Hľadáme riešenie rovnice

$$G = SG_{GRS}(\alpha_1,\alpha_2,\ldots,\alpha_n,v'_1,v'_2,\ldots,v'_n), \tag{3.1}$$

kde neznámymi sú matica S a hodnoty $\alpha_1,\alpha_2,\ldots,\alpha_n,v'_1,v'_2,\ldots,v'_n$. Podotknime, že nám postačia ľubovoľné hodnoty spĺňajúce túto rovnicu. Nemusí sa jednať o hodnoty použité pri generovaní kľúča. Pre jednoduchosť a prehľadnosť však budú značené rovnako.

Tvrdenie 3.10. *Nech n,k,$m\in\mathbb{N}, n>k$, nech A je matica typu $k\times n$ nad $\mathbb{F}_q$ ranku k. Nech B je matica nad $\mathbb{F}_q$ typu $m\times n$. Nech M je matica nad $\mathbb{F}_q$ typu $m\times k$. Nech $MA=B$. Potom ak poznáme A,B vieme zistiť M.*

Dôkaz. Nájdeme regulárnu submaticu matice A typu $k\times k$ (označme ju A') – tú získame vynechaním $n-k$ stĺpcov, ich množinu indexov označme Σ. Vynechaním stĺpcov matice B na indexoch z Σ získame maticu B'. Platí:

$$\begin{aligned}B &= MA\\ B' &= MA'\\ B'A'^{-1} &= M.\end{aligned}$$

❑

Z tvrdenia 3.10 mimo iného plynie, že ak máme dané nejaké hodnoty

$$\alpha_1,\alpha_2,\ldots,\alpha_n,v'_1,v'_2,\ldots,v'_n,$$

tak matica S je určená jednoznačne.

Môžeme teda označiť symbolom $\mathcal{S}$ množinu všetkých riešení

$$(\alpha_1,\alpha_2,\ldots,\alpha_n,v'_1,v'_2,\ldots,v'_n)\in\mathbb{F}_\infty^n\times\mathbb{F}_q^{n\,*}$$

rovnice (3.1). Ak teraz položíme

$$\mathcal{X} = \{(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_\infty^n; \exists (v'_1, \ldots, v'_n) \in \mathbb{F}_q^{n\,*} \colon (\alpha_1, \ldots, \alpha_n, v'_1, \ldots, v'_n) \in \mathcal{S}\},$$

potom platí nasledujúce tvrdenie:

Tvrdenie 3.11 ([17] str. 20)**.** *Grupa* $\boldsymbol{PLG}(2, \mathbb{F}_q)$ *pôsobí na* $\mathcal{X}$ *a zobrazenie* $\pi(\phi) \colon \mathcal{X} \to \mathcal{X}$ *pre* $\phi \in \boldsymbol{PLG}(2, \mathbb{F}_q)$ *je definované:*

$$\pi(\phi)(\alpha_1, \alpha_2, \ldots, \alpha_n) = (\phi(\alpha_1), \phi(\alpha_2), \ldots, \phi(\alpha_n)).$$

A spolu s tvrdením 3.9 získavame nasledujúci dôsledok:

Dôsledok 3.12. *Existuje prvok patriaci* $\mathcal{S}$ *tvaru:*

$$(1, 0, \infty, \alpha_4^*, \ldots, \alpha_n^*, v_1^*, v_2^*, \ldots, v_n^*),$$

kde $(\alpha_4^*, \alpha_5^* \ldots, \alpha_n^*) \in \mathbb{F}_q \setminus \{0, 1, \infty\}$ *a* $v_1^*, v_2^*, \ldots, v_n^* \in \mathbb{F}_q^*$.

Poznámka 3.2. Bohužiaľ, pri dekódovaní sa neuspokojíme s kódovými lokátormi, medzi ktorými sa vyskytuje 0 alebo ∞. Z ostrej 3-tranzitivity však plynie, že sme schopní zvoliť takú transformáciu, aby sme sa vrátili naspať do $\mathbb{F}_q \setminus \{0\}$ (je zhrnutá v algoritme 3.16). A teda nájdeme taký prvok $\mathcal{S}$, kde sú všetky zložky z $\mathbb{F}_q^*$.

Nájdenie matice G_{GRS} urobíme v dvoch krokoch. Najprv nájdeme kódové lokátory $\alpha_1, \alpha_2, \ldots, \alpha_n$ a s touto znalosťou potom nájdeme $v'_1, v'_2, \ldots, v'_n$ a maticu S. Kódové lokátory nájdeme použitím algoritmu 3.13.

Algoritmus 3.13. *Sidelnikov-Šestakov: Nájdenie kódových lokátorov.*

Vstup: Matica G typu $k \times n$, ktorá spĺňa $G = S\, G_{GRS}$, kde G_{GRS} je generujúca matica GRS kódu s lokátormi $\alpha_1, \alpha_2, \ldots, \alpha_n$ a S je regulárna matica.
Výstup: Hodnoty $\alpha_1, \alpha_2, \ldots, \alpha_n$.

$\alpha_1 \leftarrow 1$; $\alpha_2 \leftarrow 0$; $\alpha_3 \leftarrow \infty$;
while $\exists i \ \in \{1,2,\ldots,n\}$: nepoznám hodnotu α_i **do**
Nájdi takú množinu $J \subseteq \{4,5,\ldots,n\}$, že $|J| = k - 2$ a pre maximum $m \in J$ platí: α_m poznám;
Nájdi $\boldsymbol{u}_1, \boldsymbol{u}_2 \in \mathbb{F}_q^k$ tak , že:

- $\forall j \in J \colon \sum_{i=0}^{k-1} u_{l,i} g_{i,j} = 0$ pre $l \in \{1, 2\}$;
- $\sum_{i=0}^{k-1} u_{1,i} g_{i,0} = 0$ & $\sum_{i=0}^{k-1} u_{2,i} g_{i,1} = 0$;
- $\boldsymbol{u}_1, \boldsymbol{u}_2$ sú nenulové;

$\boldsymbol{c}_1 \leftarrow \boldsymbol{u}_1 G$;
$\boldsymbol{c}_2 \leftarrow \boldsymbol{u}_2 G$;
$t \leftarrow \frac{c_{1,2}}{c_{2,2}}$;
for all $j \in \{4,5,\ldots,n\} \setminus J$ **do**
$\alpha_j \leftarrow t / \left(t - \frac{c_{1,j-1}}{c_{2,j-1}}\right)$;
end for

end while

Lemma 3.14. *Nech G je generujúca matica GRS kódu typu $k \times n$. Nech $J \subseteq \{1,2,\ldots,n\}$, nech $|J| = k-1$. Potom vieme odvodiť nenulové informačné slovo $\boldsymbol{u}$ tak, že k nemu kódové slovo $\boldsymbol{c}$ má na pozíciách z J nuly. Navyše, polynóm $g_u(x)$ prislúchajúci informačnému slovu $\boldsymbol{u}$ je v tvare:*

$$g_u(x) = \lambda \prod_{i \in J, i \neq \infty} (x - \alpha_i).$$

Dôkaz. Hľadáme informačné slovo $\boldsymbol{u}$ také, že kódové slovo $\boldsymbol{c}$ má na daných pozíciách nuly. Pretože kódovanie zodpovedá evaluácii polynómu, získame tak kódové slovo, resp. polynóm, ktorý má korene z množiny J. Z definície násobenia matíc dostávame vzťahy použité v algoritme 3.13 – zodpovedajú tomu, že hľadáme slovo, ktoré má na pozíciách z J hodnotu nula. Riešime homogénnu sústavu $k-1$ lineárnych rovníc o k premenných – má netriviálne riešenie. Ak označíme

$$g_u(x) = \sum_{i=0}^{k-1} u_i f_i(x),$$

kde $f_i(x)$ závisia na S, z polynomiálneho náhľadu na kódové slová dostávame, že

$$c_j = \sum_{i=0}^{k-1} u_i v'_i f_i(\alpha_j) = \sum_{i=0}^{k-1} v'_i g_u(\alpha_j)$$

a stupeň $g_u(x)$ je menší rovný $k-1$. Stupeň $g_u(x)$ bude menší ako $k-1$ práve v prípade, keď bude na niektorej pozícii z J lokátor rovný ∞.

❑

Lemma 3.15. *Algoritmus 3.13 funguje.*

Dôkaz. Použitím lemmy 3.14 nájdeme $\boldsymbol{u}_1, \boldsymbol{u}_2$ a k nim $\boldsymbol{c}_1, \boldsymbol{c}_2$. Ak sa teraz pozrieme na náš konkrétny prípad, spomenieme si, že $\alpha_1 = 1, \alpha_2 = 0, \alpha_3 = \infty$. Lenže $\boldsymbol{u}$ sme volili tak , aby pre $j \in J$ platilo $c_j = 0$ a pretože $|J| = k-1$ a $3 \notin J$ (v algoritme volíme $k-2$ hodnôt, v prvom prípade pridáme hodnotu 0 a v druhom hodnotu 1) tak platí:

$$g_u(x) = \lambda \prod_{i \in J} (x - \alpha_i),$$

a teda stupeň $\deg(g_u(x)) = k-1$. Máme:

$$g_{u_1}(x) = \lambda_1 (x-1) h(x),$$
$$g_{u_2}(x) = \lambda_2 x h(x).$$

Platí $\forall j \notin J \cup \{1,2\}\colon c_j \neq 0$ (polynomiálny náhľad - $g_{u_i}(x)$ má stupeň $k-1$ a korene sú práve prvky J a $|J| = k-1$) a dostávame

$$\forall j \notin J \cup \{1,2\}\colon t_j := \frac{c_{1,j}}{c_{2,j}} = \frac{v'_j g_1(\alpha_j)}{v'_j g_2(\alpha_j)} = \frac{\lambda_1(\alpha_j - 1)}{\lambda_2 \alpha_j}.$$

Ďalej z definície 3.5 získavame pohľad na evaluáciu v ∞

$$t = \frac{v'_j g_1(\infty)}{v'_j g_2(\infty)} = \frac{\lambda_1}{\lambda_2},$$
$$t_j = t_3 \frac{\alpha_j - 1}{\alpha_j} \Rightarrow \alpha_j = \frac{t}{t - t_j}.$$

Tento postup opakujeme dovtedy, kým nenájdeme všetky hodnoty α_j. V ďalšej iterácii zvolíme do množiny J najprv tie prvky, ku ktorým už α_j dopočítané máme, čo nám umožní spočítať α_j pre všetky hodnoty.

❑

Poznámka 3.3. V pôvodnej práci [17], ako aj v [7] sa predpokladalo $2k < n$, čo im umožňovalo vybrať práve dve dvojice informačných (a k nim príslušiacich kódových) slov a získali tak všetky hodnoty α_j. V tejto práci je tento predpoklad vynechaný za cenu mierne zvýšenej komplexity algoritmu. Navyše, hodnoty n,k odporučené v [8] tento predpoklad nespĺňajú.

V nasledujúcom algoritme je zhrnutá transformácia ktorá zaručí, že nájdené riešenie rovnice (3.1) nebude obsahovať hodnoty 0 a ∞ a teda bude možné pomocou neho dešifrovať.

Algoritmus 3.16. *Úprava hodnôt α_i tak , aby neobsahovali hodnoty ∞ a 0.*

Vstup: $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}_q \cup \{\infty\}$ kódové lokátory GRS kódu $\mathcal{C}_{GRS}$.
Výstup: $\alpha_1^*, \alpha_2^*, \ldots, \alpha_n^* \in \mathbb{F}_q \setminus \{0\}$ kódové lokátory GRS kódu $\mathcal{C}_{GRS}$.

Nájdi $a \in \mathbb{F}_q, \forall i \ \in \mathbb{N}, i \ \leq n$: $\alpha_i \neq a$;
for $i \leftarrow 1,2,\ldots,n$ **do**
  $\alpha'_i \leftarrow 1/(a - \alpha_i)$;
end for
Nájdi $a \in \mathbb{F}_q, \forall i \ \in \mathbb{N}, i \ \leq n$: $\alpha'_i \neq a$;
for $i \leftarrow 1,2,\ldots,n$ **do**
  $\alpha_i^* \leftarrow \alpha'_i - a$;
end for

So znalosťou kódových lokátorov môžeme pristúpiť k nájdeniu generujúcej matice G_{GRS}.

Algoritmus 3.17. *Sidelnikov-Šestakov: Nájdenie matíc S a G_{GRS}.*

Vstup: Matica G, ktorá spĺňa $G = S\, G_{GRS}$, kde G_{GRS} je generujúca matica GRS kódu s lokátormi $\alpha_1, \alpha_2, \ldots, \alpha_n$ a S je regulárna matica, hodnoty $\alpha_1, \alpha_2, \ldots, \alpha_n$.
Výstup: Matice S, G_{GRS}.

Nájdi $c_j \in \mathbb{F}_q$ pre $j \in \{1,2,\ldots,k+1\}$ tak , že platí:

- $\sum_{j=1}^{k+1} c_j g_{i,j} = 0$;
- $\exists j \in \{1,2,\ldots,k+1\}$: $c_j \neq 0$;

$v'_1 \leftarrow 1$;
Nájdi $v'_j \in \mathbb{F}_q$ pre $j \in \{2,3,\ldots,k+1\}$ tak , že:

- $\forall i \ \in \mathbb{N} \cup \{0\}, i \ \leq k-1 \colon \sum_{j=1}^{k+1} c_j v'_j \alpha_j^i = 0$;

for all $i \in \mathbb{N} \cup \{0\}, i \ \leq k-1$ **do**
Nájdi $s_{i,0}, s_{i,1}, \ldots, s_{i,k-1}$ spĺňajúce:

- $\forall j \in \mathbb{N}, j \leq k \ : \ \sum_{l=1}^{k-1} s_{i,l} \alpha_j^l = {v'_j}^{-1} g_{i,j}$;

end for
$S \leftarrow (s_{i,j})$;
Nájdi $S^{-1} = (s'_{i,j})$;
for all $j \in \{k+2, k+3, \ldots, n\}$ **do**
$v'_j \leftarrow \sum_{i=0}^{k-1} s'_{0,i} g_{i,j}$;
end for

Lemma 3.18. *Algoritmus 3.17 funguje.*

Dôkaz. Ak budú $(\alpha_1, \alpha_2, \ldots, \alpha_n, v'_1, v'_2, \ldots, v'_n) \in \mathcal{S}$ a k nim bude prislúchať matica $S = (s_{i,j})$, potom pre $a \in \mathbb{F}_q^*$ bude $(\alpha_1, \alpha_2, \ldots, \alpha_n, av'_1, av'_2, \ldots, av'_n) \in \mathcal{S}$ a k nim bude prislúchať matica $S' = (a^{-1} s_{i,j})$ (z násobenia matíc). Z toho plynie, že existuje riešenie, kde $v'_1 = 1$ – také budeme hľadať. Sústava premenných c_j:

$$\forall i, 0 \leq i \leq k-1 \colon \sum_{j=1}^{k+1} c_j g_{i,j} = 0$$

je homogénnou sústavou k rovníc o $k+1$ premenných – má netriviálne riešenie. Ďalej,

$$\forall j \in \mathbb{N}, j \leq k+1 \colon c_j \neq 0,$$

pretože opačný prípad by znamenal k lineárne závislých vektorov generujúcej matice GRS kódu – spor s tým, že je to MDS kód (tvrdenie 1.7). Vieme, že matica G je tvaru

$$G = \begin{pmatrix} v'_1 f_0(\alpha_1) & v'_2 f_0(\alpha_2) & \ldots & v'_n f_0(\alpha_n) \\ v'_1 f_1(\alpha_1) & v'_2 f_1(\alpha_2) & \ldots & v'_n f_1(\alpha_n) \\ \vdots & \vdots & \ddots & \vdots \\ v'_1 f_{k-1}(\alpha_1) & v'_2 f_{k-1}(\alpha_2) & \ldots & v'_n f_{k-1}(\alpha_n) \end{pmatrix},$$

kde $f_i(x) = \sum_{i=0}^{k-1} s_{j,i} x^i$. Môžeme teda sústavu prepísať do nasledujúceho tvaru:

$$\begin{pmatrix} f_0(\alpha_1) & \ldots & f_0(\alpha_{k+1}) \\ f_1(\alpha_1) & \ldots & f_1(\alpha_{k+1}) \\ \vdots & \ddots & \vdots \\ f_{k-1}(\alpha_1) & \ldots & f_{k-1}(\alpha_{k+1}) \end{pmatrix} \begin{pmatrix} c_1 & \ldots & 0 \\ 0 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & c_{k+1} \end{pmatrix} \begin{pmatrix} 1 \\ v'_2 \\ \vdots \\ v'_{k+1} \end{pmatrix} = 0$$

a po prenásobení maticou S^{-1} zľava, získavame rovnosť

$$\begin{pmatrix} \alpha_1^0 & \alpha_2^0 & \dots & \alpha_n^0 \\ \alpha_1^1 & \alpha_2^1 & \dots & \alpha_n^1 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \dots & \alpha_{k+1}^{k-1} \end{pmatrix} \begin{pmatrix} c_1 & 0 & \dots & 0 \\ 0 & c_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & c_{k+1} \end{pmatrix} \begin{pmatrix} 1 \\ v_2' \\ \vdots \\ v_{k+1}' \end{pmatrix} = 0,$$

z čoho po roznásobení plynie:

$$\sum_{j=2}^{k+1} c_j \alpha_j^i v_j' = -c_1 \alpha_1^i.$$

Toto je sústava k rovníc o k premenných. Ak je determinant matice sústavy nenulový, má práve jedno riešenie. To platí, pretože sa jedná o Vandermondov determinant a platí, že $\alpha_i \neq \alpha_j \Leftrightarrow i \ \neq j$. Získali sme $v_1', v_2', \dots, v_{k+1}'$. Vyriešením sústavy s neznámymi $s_{i,l}$ pre pevné i

$$\forall j \in \mathbb{N}, j \leq k \ : \ \sum_{l=0}^{k-1} s_{i,l} \alpha_j^l = {v_j'}^{-1} g_{i,j}$$

získame i-ty riadok matice S. Riešenie existuje a je jednoznačne určené – znova cez Vandermondov determinant. Všimneme si, že opakovane riešime sústavu s rovnakou ľavou stranou – môžme riešiť všetky pravé strany súčasne. Našli sme maticu S, ostáva nájsť hodnoty $v_{k+2}', v_{k+3}', \dots, v_n'$. Vieme ale, že prvý riadok matice G_{GRS} tvoria hodnoty $v_1', v_2', \dots, v_n'$. Tak maticu G prenásobíme zľava maticou S^{-1}, čím dostaneme maticu G_{GRS} a požadované hodnoty.

❑

V tomto bode je možné zistiť konkrétne náhodné vektory, ktoré boli k pôvodnému GRS kódu vložené. Jednoducho prenásobíme zľava maticou S^{-1} maticu G'.

Poznámka 3.4. S náhodnými vektormi na správnych pozíciách získame maticu H'' ako nejakú kontrolnú maticu kódu $\mathcal{C}'$. Vidno, že táto matica nám poslúži rovnako dobre ako pôvodná použitá matica (pretože jadro matice H'' je rovnaké ako matice H' tak sa líšia iba ekvivalentnými úpravami, ktoré sa dajú reprezentovať prenásobením regulárnou maticou zľava – tú „odchytíme“ prinásobenú k M).

Vieme, že existuje regulárna submatica matice H'' rozmeru $(n-k) \times (n-k)$ (z ranku a rozmeru kontrolnej matice lineárneho kódu). Môžeme teda použiť tvrdenie 3.10 a získať tak maticu M.

Tým sme prelomili šifru.

3.3 Implementačné problémy

V tejto sekcii sa nachádzajú algoritmy, ktoré nie sú veľmi zaujímavé z matematického hľadiska, ale pri implementácii je nutné ich uvážiť.

Algoritmus 3.19. *Nájdenie dimenzie hviezdovej mocniny kódu $\mathcal{C}$ daného generujúcou maticou G.*

Vstup: Generujúca matica G kódu $\mathcal{C}$ typu $m \times n$.
Výstup: d, dimenzia kódu $\mathcal{C}^2$.

T buď matica typu $(m(m-1)/2+m) \times n$;
$l \leftarrow 0$;
for $i \leftarrow 0,1,\ldots m$ **do**
  for $j \leftarrow 0,1,\ldots i$ **do**
    for $k \leftarrow 0,1,\ldots n$ **do**
      $t_{l,k} = g_{i,k}g_{j,k}$;
    end for
    $l \leftarrow l+1$;
  end for
end for
$d \leftarrow \mathrm{rank}(T)$;

Lemma 3.20. *Algoritmus 3.19 funguje.*

Dôkaz. Z definície 1.12 vidíme, že prvky hviezdovej mocniny kódu sú generované hviezdovým produktom dvojíc prvkov bázy. Algoritmus 3.19 tieto hviezdové produkty spočíta a umiestni ich do riadkov pomocnej matice T. Spočítaním hodnosti matice zistíme dimenziu priestoru generovaného týmito vektormi – dimenziu hviezdovej mocniny kódu $\mathcal{C}$.

❑

Nasledujúca dvojica algoritmov reprezentuje prevod medzi reťazcami znakov (kde jeden znak je reprezentovaný ôsmimi bitmi) a vektormi dĺžky n o váhe t.

Algoritmus 3.21. *Prevod reťazca znakov na vektory váhy t.*

Vstup: Reťazec znakov s.
Výstup: Matica V typu $n \times k$ nad $\mathbb{F}_{2^m}$, kde stĺpce majú váhu t.

$p_s \leftarrow 0$;
$k \leftarrow 8\frac{s.length}{t(m-1)}+1$;
Zvoľ náhodne t po dvoch rôznych čísel r_j z množiny $\{0,1,\ldots,n-1\}$;
V buď typu $n \times k$;
for $i \leftarrow 0,1,\ldots,k$ **do**
  for $j \leftarrow 0,1\ldots,t$ **do**
    Nastav nultý bit $v_{r_j,i}$ na jednotku;
    for $l \leftarrow 1,\ldots,m$ **do**
      Nastav l-tý bit $v_{r_j,i}$ na hodnotu p_s-tého bitu s;

```
            p_s ← p_s + 1;
        end for
    end for
end for
```

Algoritmus 3.22. *Prevod vektorov na reťazec znakov.*

Vstup: Matica V typu $n \times k$ nad $\mathbb{F}_{2^m}$, kde stĺpce majú váhu najviac t.
Výstup: Reťazec znakov s.

```
x buď reťazec bitov;
for i ← 0,1,...n do
    for j ← 0,1,...k do
        if v_{i,j} ≠ 0 then
            for l ← 1,2,...m do
                x ← x zreťazené s l-tým bitom v_{i,j};
            end for
        end if
    end for
end for
for i ← 0,1,...x.length/8 do
    s_i ← x_{i+0}||x_{i+1}||...||x_{i+7};
end for
s ← s_1||s_2||...||s_μ;
```

Lemma 3.23. *Algoritmy 3.21 a 3.22 fungujú.*

Dôkaz. Algoritmy vychádzajú z reprezentácie prvku telesa $\mathbb{F}_{2^m}$, ktorý je reprezentovaný ako m-tica bitov. Rozdelíme preto vstupný reťazec znakov (ktoré sú reprezentované osmicou bitov) na $(m-1)$-tice bitov, prvý bit v každej zvolenej hodnote nastavíme na jednotku. To nám umožní priamočiaro umiestniť $m-1$ bitov do zvyšných bitov zvolenej hodnoty. Prvý nenulový bit zaručí, že prvok, ktorý takto získame bude nenulový a teda sa bude dať odlíšiť od nulových prvkov ktoré nenesú informáciu.

Keď chceme naopak získať pôvodné bity, nájdeme nenulové hodnoty v danom vektore a pozrieme sa na bity 1 až m (všetko indexujeme od 0). Potom už len stačí dané bity zapísať za sebou, rozdeliť do osmíc a získame pôvodný reťazec s. Ak by aj niekde ostali nevyužité bity (týka sa to posledného nenulového prvku – nemusíme potrebovať využiť $m-1$ bitov), rozdelením do osmíc a vynechaním zvyšku dostaneme pôvodný reťazec.

❑

3.4 Samotný útok

Lemma 3.24. *Časová zložitosť algoritmu 3.6 je $O(k^2(n+r)^3)$ (v operáciách v $\mathbb{F}_q$).*

Dôkaz. Všetky kroky algoritmu okrem počítania dimenzie hviezdového produktu kódu majú konštantnú zložitosť (v operáciách v $\mathbb{F}_q$). Pri počítaní dimenzie hviezdového produktu spočítame hviezdové produkty všetkých dvojíc bázy ($O(k(k-1)(n+r)) = O(k^2(n+r))$) a spočítame hodnosť matice, ktorá vznikne naskladaním týchto produktov do riadkov (použitím Gaussovej metódy: $O(k(k-1)(n+r)^2) = O(k^2(n+r)^2)$). Pretože takto overujeme všetky stĺpce, výsledná časová zložitosť je $O(k^2(n+r)^3)$ (v operáciách v $\mathbb{F}_q$).

❑

Lemma 3.25. *Časová zložitosť algoritmu 3.13 je $O((n-k)^3)$ (v operáciách v $\mathbb{F}_q$).*

Dôkaz. Nájdeniu hodnôt $u_{1,i}, u_{2,i}, u_{3,i}, u_{4,i}$ zodpovedá vyriešenie sústav $n-k-1$ rovníc o $n-k$ premenných (Gauss: $O((n-k)^3)$ operácií v $\mathbb{F}_q$). Nájdeniu hodnôt $c_{1,i}, c_{2,i}, c_{3,i}, c_{4,i}$ zodpovedá výpočet sumy $n-k$ prvkov pre každé i, ktoré prebieha od 3 do n ($O((n-k)n)$ operácii v $\mathbb{F}_q$). Celková časová zložitosť je teda $O((n-k)^3)$.

❑

Lemma 3.26. *Časová zložitosť algoritmu 3.16 je $O(n)$ (v operáciách v $\mathbb{F}_q$).*

Dôkaz. Nájdenie prvku $a \in \mathbb{F}_q$ zaberie najviac n porovnaní (a teda prejdeme najviac $n+1$ prvkov) - $O(n)$. Pre každé i od 1 do n spočítame hodnotu α_i troma operáciami v $\mathbb{F}_q$ - $O(n)$. Druhá transformácia analogicky ($O(n)$), celková zložitosť je teda $O(n)$.

❑

Lemma 3.27. *Časová zložitosť algoritmu 3.17 je $O((n-k)^3 + n(n-k))$ (v operáciách v $\mathbb{F}_q$).*

Dôkaz. Nájdenie hodnôt c_j pozostáva z vyriešenia homogénnej sústavy o $n-k$ neznámych ($O((n-k)^3)$). Pri hľadaní hodnôt v'_j z druhého kroku potrebujeme najprv zostaviť maticu rovníc – zostavíme Vandermondovu maticu z hodnôt α_i ($O((n-k)^2)$) a stĺpce prenásobíme hodnotami c_j ($O((n-k)^2)$) a pokračujeme vyriešením homogénnej sústavy $n-k$ rovníc o $n-k$ neznámych ($O((n-k)^3)$).

Ak si zapamätáme Vandermondovu maticu z predchádzajúceho kroku, môžeme ju s výhodou využiť pri zostavovaní sústavy pre hľadanie hodnôt $s_{i,j}$ ($O((n-k)^2)$), ktorú vyriešime pre všetky pravé strany naraz ($O((n-k)^3)$). Ostáva nájsť inverznú maticu S^{-1} ($O((n-k)^3)$) a dopočítať zvyšné hodnoty v'_i ($O((n-k)n)$).

Celková zložitosť je teda $O((n-k)^3 + n(n-k))$.

❑

Nasleduje zhrnutie celého útoku:

Algoritmus 3.28. *Útok na Wieschebrinkovu verziu Niederreiterovho systému.*

Vstup: Verejný kľúč Key_{pub} Wieschebrinkovej verzie Nederreiterovho systému.
Výstup: Súkromný kľúč Key_{priv} taký, že Key_{pub} je k nemu kľúč verejný.

1: Zistíme maticu G typu $k \times m$, generujúcu maticu kódu, ktorému je H^{pub} kontrolnou maticou;
2: $P \leftarrow$ zoznam pozícii daných výstupom algoritmu 3.6;
3: Získame maticu G^* prepichnutím matice G na pozíciách z P;
4: $n \leftarrow m - |P|$;
5: Určíme kódové lokátory $\alpha_1, \alpha_2, \ldots, \alpha_n$ použitím algoritmu 3.13;
6: Určíme G_{GRS} a R tak , že platí $RG_{GRS} = G^*$ a G_{GRS} je v tvare dôsledku 1.9 použitím algoritmu 3.17;
7: Pomocou R^{-1} určíme maticu $G' = R^{-1}G$; ▷ Na pozíciách z P sa vyskytujú práve náhodné vektory, prepichnutím v týchto pozíciách získame maticu G_{GRS}
8: Pomocou G' zistíme H', ktorá spĺňa $MH' = H^{pub}$ pre vhodnú regulárnu maticu M (pozn. 3.4);
9: Získame M aplikovaním tvrdenia 3.10;

Lemma 3.29. *Časová zložitosť algoritmu 3.28 je $O(k^2(n+r)^3)$ (v operáciách v $\mathbb{F}_q$).*

Dôkaz. Algoritmus 3.6 má práve túto zložitosť. Zložitosť zvyšných algoritmov je oproti tomu asymptoticky malá (viď lemmy v tejto sekcii).

❑

Kapitola 4

Výsledky útoku

4.1 Dešifrovanie – meranie času

V tabuľke 4.1 sú ukázané namerané hodnoty pre kryptosystém nad telesom $\mathbb{F}_q$ s GRS kódom typu $[n,k]$ a r náhodnými vektormi. Príslušný čas potom zodpovedá priemernému jednému útoku na kryptosystém.

q	n	k	r	Čas (sekundy)
128	95	61	12	0.075
128	127	79	17	0.299
256	191	119	26	2.312
256	255	161	34	9.919
512	383	243	51	69.815
512	511	325	68	413.713

Tabuľka 4.1: Namerané priemerné časy pri $N = 100$ pokusoch.

Ostáva vysvetliť pôvod hodnôt n,k,r. V [8] bol navrhnutý spôsob, akým určiť k a r, aby sa dosiahlo čo najväčšej bezpečnosti ako proti dekódovacím útokom, tak aj proti útokom hrubou silou (tipovanie množiny náhodných stĺpcov). Pre nájdenie optimálnych k,r som použil softvér SageMath [19].

Pretože pre vyššie hodnoty q,n,k,r útok prebieha dlhší čas, meranie prebiehalo s obmedzením na $N = 5$ pokusov (výsledky sú zhrnuté v tabuľke 4.2).

q	n	k	r	Čas (sekundy)
1024	767	475	106	5878.010
1024	1023	639	140	26733.254

Tabuľka 4.2: Namerané priemerné časy pri $N = 5$ pokusoch.

4.2 Hypotéza 3.3 – meranie pravdepodobnosti úspechu

Sú známe okrajové prípady, ktoré nespĺňajú rovnosť hypotézy 3.3 (napr. všetky náhodné vektory sú nulové). Test pozostával z vygenerovania danej matice, spočítania dimenzie hviezdovej mocniny a overenia nerovnosti. Pre dané q,n,k,r bol tento test zopakovaný $N = 1000$ krát.

Výsledky pre nižšie n sa nachádzajú v tabuľke 4.3. Pre vyššie n (až do 255) a q (až do 256) mali všetky testy úspešnosť 100 percent.

q	n	k	r	Úspešnosť (%)
8	7	2	1	0.0
16	15	5	2	99.9
16	15	5	3	99.9
16	15	6	1	100.0

Tabuľka 4.3: Experimentálne overovanie platnosti hypotézy 3.3 pri $N = 1000$ pokusoch.

Pretože malé hodnoty sú z kryptografického hľadiska nepoužiteľné, z týchto výsledkov plynie, že má dobrý zmysel predpokladať platnosť vety v náhodnom prípade pre väčšie q,n,k,r.

Záver

V práci boli vyložené relevantné poznatky z teórie samoopravných kódov ako aj z teórie grúp. Ďalej boli definované a popísané kryptosystémy založené na samoopravných kódoch, konkrétne McEliece-ov [1], Niederreiterov [5] a ich modifikácie podľa Wieschebrinka [8].

Prínosom práce je podrobný popis útoku. Ďalším prínosom je sprehľadnenie algoritmu 3.13, ako aj odstránenie predpokladu $2(n-k-1) < n$, ktorý bol použitý v prácach [7], [17] a ktorý sa ukázal byť v priamom rozpore s odporúčanými hodnotami n,k tak , ako boli popísané v [8]. Dôkazu hypotézy 3.3 bolo síce venované veľké množstvo úsilia, ale nepodarilo sa ho dokončiť. Každopádne bol naznačený smer dôkazu, ako aj hypotéza, z ktorej pravdivosti by veta plynula. V neposlednej rade treba spomenúť kompletnú implementáciu útoku v jazyku C++ (mimo aritmetiky konečných telies charakteristiky 2, ale vrátane napr. dekódovania GRS kódov), ako aj vyriešenie problémov s útokom spojených, ktoré nie sú z matematického hľadiska až tak zaujímavé.

Experimentálne namerané výsledky podporujú tvrdenie hypotézy 3.3 a to bez výnimky pre $q \geq 32$ a $n \geq 31$ a len s drobnou odchýlkou pre $q = 16$ a $n = 15$. Namerané časy jednotlivých behov potom ilustrujú efektivitu tohto útoku.

Dodatok A

Dokumentácia programu

Program priložený k tejto práci implementuje útok na Wieschebrinkovu verziu Niederreitovho systému. Súčasťou programu sú aj testy, ktorých výsledky sa nachádzajú v kapitole 4.

Program je navrhnutý objektovo a implementovaný v jazyku C++, kompilovaný kompilátorom `g++ version 4.8.3 20140911 (Red Hat 4.8.3-7)` a otestovaný v systéme `Fedora 20`.

A teraz pristúpme k jednotlivým častiam programu.

A.1 Konečné telesá

Aritmetiku nad konečnými telesami implementuje trieda `GaloisField`, knižnice `Galois Field Arithmetic Library (version 0.0.1)`, autora Arash Partow (dostupná na `http://www.partow.net/projects/galois/index.html` pod licenciou Common Public Licence – `http://www.opensource.org/licenses/cpl.php`).

Táto trieda implementuje aritmetiku nad konečnými telesami charakteristiky 2. Objektom je rozšírenie telesa $\mathbb{Z}_2$ dané primitívnym polynómom pri konštrukcii objektu. Samotné prvky telesa sú typu `galois::GFSymbol`, polynóm prislúchajúci prvku je uložený v pamäti ako číslo v dvojkovej sústave.

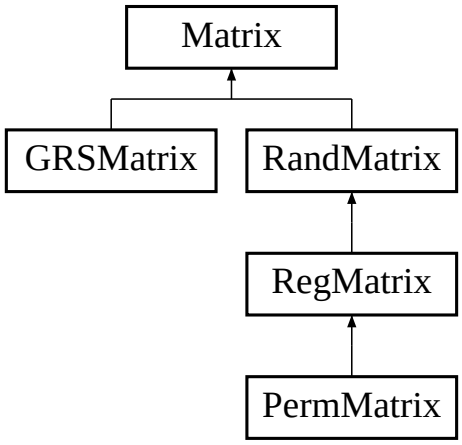
Za zmienku stojí následné využívanie tejto triedy ostatnými objektami. Každá z tried obsahuje `public static` ukazovateľ na objekt `GaloisField`, pôvodne inicializovaný nulou. Na začiatku behu programu sa zvolí primitívny polynóm, vytvorí sa objekt a každá z tried sa inicializuje. To umožňuje objektom jednoduchý prístup k jednému pevne zvolenému telesu.

A.2 Matice

Pre manipuláciu s maticami som implementoval niekoľko pomocných tried, ich hierarchickú štruktúru ukazuje obrázok A.2.

V triede `Matrix` sa nachádzajú základné algoritmy pre prácu s maticami koncipované tak, aby ich návratová hodnota bola opäť matica. Tiež platí, že (až na `setXY`, `plusKTimesRow` a konštruktory) nemenia hodnotu matice. Sú to nasledujúce metódy:

```
inline int getNumRows() const;
inline int getNumCols() const;
```



Obr. A.1: Hierarchia dedičnosti tried matíc.

```
inline galois::GFSymbol getXY(int col, int row) const;
inline void setXY(int col, int row, galois::GFSymbol val);
int compRank();
void plusKTimesRow(galois::GFSymbol k, int rowFrom, int rowTo);
int squareStarCodeDim();
Matrix transpose();
Matrix findKer();
Matrix puncture(std::vector<int>& positions);
Matrix punctureInv(std::vector<int>& positions);
Matrix GaussElimination();
Matrix solve(Matrix& rightSide);
static Matrix multiplication(Matrix& A, Matrix& B);
```

Väčšina z nich je implementovaná priamočiaro, na mieste je však pár poznámok. Metóda `plusKTimesRow` mení hodnotu matice – vezme riadok na pozícii `rowFrom` a pripočíta jeho `k`-násobok k riadku na pozícii `rowTo`. Metóda `squareStarCodeDim` počíta dimenziu hviezdového produktu kódu (určeného danou maticou ako generujúcou maticou kódu) implementujúc algoritmus 3.19. Metóda `puncture` prepichne maticu v stĺpcoch daných vektorom v parametri, predpokladám, že sú zoradené vzostupne. Metóda `punctureInv` prepichne maticu v stĺpcoch, ktorých indexy sa nenachádzajú v parametri, taktiež predpokladám vzostupné zoradenie. Metóda `solve` rieši sústavu rovníc danú touto maticou pre pravú stranu `rightSide`, ktorú predpokladám v tvare matice typu $m \times 1$.

Pokračujme triedou `GRSMatrix`. Táto trieda reprezentuje kontrolnú maticu GRS kódu. Je určená vektormi `locators, multiplicators` v duchu definície 1.10. Jej zaujímavými metódami je dvojica `encode` a `decode`. Metóda `encode` má ako parameter riadkový vektor dĺžky k, kde k je dimenzia daného GRS kódu – vráti kódové slovo reprezentované stĺpcovým vektorom. Metóda `decode` má ako parameter stĺpcovy vektor dĺžky n, kde n je dĺžka daného GRS kódu – vráti stĺpcový chybový vektor dĺžky n implementujúc algoritmus načrtnutý v poznámke 1.7.

Nasleduje zvyšná trojica tried – `RandMatrix`, `RegMatrix`, `PermMatrix`. Ako pozorný čitateľ určite vydedukoval, tieto triedy reprezentujú v poradí náhodnú maticu, regulárnu maticu a maticu permutácie. Spoločnú majú metódu `generate`, ktorá ich náhodne vygeneruje (neprekvapí, že pre každú triedu je implementovaná

inak). Triedy `RegMatrix` a `PermMatrix` ďalej majú spoločnú metódu `inverse`, ktorá vráti inverznú maticu. Trieda `PermMatrix` obsahuje metódu `ChangePos`, ktorá pre daný index i vráti hodnotu indexu, na ktorý ho daná permutácia zobrazí.

A.3 Polynómy

Nasleduje trieda `Polynomial` implementujúca polynómy nad konečným telesom. Jedná sa o zoskupenie väčšieho množstva pomerne priamočiarych metód, takže pri väčšine uvediem len názov:

```
inline int getDeg() const;
inline void setDeg(int n);
inline void setA(int i, galois::GFSymbol val);
inline galois::GFSymbol getA(int i) const;

inline Polynomial plus(const Polynomial& b) const;
inline Polynomial minus(const Polynomial& b) const;
inline Polynomial constMul(galois::GFSymbol x) const;
inline Polynomial shift(int a) const;
Polynomial mul(const Polynomial& b) const;
Polynomial divBy(const Polynomial& b) const;
Polynomial modBy(const Polynomial& b) const;

Polynomial derivative() const;
void EuclidForDecoding(int d, Polynomial & lambda, Polynomial & gamma) const;
galois::GFSymbol eval(galois::GFSymbol x) const;
void write() const;
```

Algoritmy aritmetiky polynómov sú školské. Metóda `shift` sa dá interpretovať ako prenásobenie polynómu hodnotou x^a. Za zmienku stojí metóda `EuclidForDecoding`, ktorá je súčasťou algoritmu riešenia kľúčovej rovnice – implementuje Euklidov algoritmus pre potreby dekódovania tak , ako je popísaný v knihe [11] (str. 191).

A.4 Kryptosystém

Trieda `Cryptosystem` implementuje Wieschebrinkovu verziu Niederreiterovho systému tak, ako bol popísaný v definícii 2.4. Pri konštrukcii objektu treba dodať kontrolnú maticu GRS kódu a počet náhodných vektorov, neskôr je možné vygenerovať kryptosystém s rovnakými q,n,k,r, ale odlišným GRS kódom metódou `regenerate`. Jedna inštancia triedy zodpovedá používaniu jedného kľúča.

Otvorené texty sú reprezentované typom `std::string`, šifrové texty sú reprezentované stĺpcovými vektormi dĺžky $n + r$, kde n je dĺžka GRS kódu podaného konštruktoru a r je počet náhodných vektorov (tiež parameter konštruktoru). Trieda obsahuje pomocné metódy `stringToVectors` a `vectorsToString`, ktoré prevádzajú medzi typom `std::string` a stĺpcovými vektormi dĺžky $n + r$ váhy najviac t, kde t je počet chýb, ktorý je daný GRS kód schopný opraviť. Tieto metódy implementujú algoritmy 3.21 a 3.22, využívajúc pomocné triedy `Buffer`

a `InsertBuffer` na uľahčenie práce s prúdom bitov (prvá z nich prevádza `std::string` na prúd bitov, druhá naopak).

Ostáva popísať dvojicu metód `encrypt`, `decrypt`. V prvej prevedieme reťazec znakov na vektory správnej váhy, a ako v definícii 2.4 prenásobíme sprava maticu H^{pub}. Druhá implementuje algoritmus v danej definícii popísaný, potom prevedie vektory na reťazec znakov.

A.5 Útočník

Trieda `Attacker` implementuje samotný útok na šifru tak , ako je popísaný v sekcii 3.4.

Pri konštrukcii sa dodá kryptosystém, na ktorý chceme útočiť. Metóda `ripntear` vykoná samotný útok. Pre daný ciphertext potom metóda `decrypt` vráti plaintext. Pre opakovanie útoku na iný kryptosystém s rovnakými parametrami môžeme použiť metódu `newCryptosystem`, ktorá pre naalokovaný kryptosystém zavolá metódu `regenerate`.

A.6 Testy

Ostáva trieda `Testin` zodpovedná za meranie dát. Jedná sa o pomerne technickú záležitosť. Dôraz bol kladený na to, aby pri akomkoľvek probléme pri výpočte (ktorý môže prebiehať aj niekoľko hodín) sa zachovali aspoň nejaké výsledky. Preto sa každý test zapíše do separátneho súboru, a pri novom behu sa skontroluje či už daný test nie je hotový (ak áno, druhýkrát sa nerobí).

Dáta merajú až z nej odvodené triedy `DimTests` a `TimeTests`. Prvá meria pravdepodobnosť úspechu hypotézy 3.3 pri náhodných testoch s náhodnými n,k,r. Jeden test pozostáva z testovania `numOfRuns` rôznych matíc pre fixné n,k,r. Druhá meria čas útoku na kryptosystém, pre n,k,r zvolené spôsobom aký bol popísaný v [8]. V každom teste sa pre fixné n,k,r generuje `numOfRuns` rôznych kryptosystémov a na každý z nich je prevedený útok. Do celkového času sa započítava iba čas útoku (tj. nezapočítava sa tam príprava ďalšieho kryptosystému).

Literatúra

[1] Robert J McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN progress report*, 42(44):114–116, 1978.

[2] Alain Couvreur, Ayoub Otmani, and Jean-Pierre Tillich. Polynomial time attack on wild mceliece over quadratic extensions. In *Advances in Cryptology–EUROCRYPT 2014*, pages 17–39. Springer, 2014.

[3] Jean-Charles Faugere, Valérie Gauthier-Umana, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high-rate mceliece cryptosystems. *Information Theory, IEEE Transactions on*, 59(10):6830–6844, 2013.

[4] Hang Dinh, Cristopher Moore, and Alexander Russell. Mceliece and niederreiter cryptosystems that resist quantum fourier sampling attacks. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 761–779. Springer Berlin Heidelberg, 2011.

[5] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. 1986.

[6] Yuan Xing Li, Robert H Deng, and Xin Mei Wang. On the equivalence of mceliece's and niederreiter's public-key cryptosystems. *IEEE Transactions on Information Theory*, 40(1):271–273, 1994.

[7] Vladimir M Sidelnikov and Sergey O Shestakov. On insecurity of cryptosystems based on generalized reed-solomon codes. *Discrete Mathematics and Applications*, 2(4):439–444, 1992.

[8] Christian Wieschebrink. Two NP-complete Problems in Coding Theory with an Application in Code Based Cryptography. In *IEEE International Symposium on Information Theory*, 2006.

[9] Alain Couvreur, Philippe Gaborit, Valérie Gauthier-Umaña, Ayoub Otmani, and Jean-Pierre Tillich. Distinguisher-based attacks on public-key cryptosystems using reed-solomon codes. *arXiv preprint arXiv:1307.6458*, 2013.

[10] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[11] Ron Roth. *Introduction to Coding Theory*. Cambridge University Press, New York, NY, USA, 2006.

[12] W Cary Huffman and Vera Pless. *Fundamentals of error-correcting codes*. Cambridge university press, 2003.

[13] Raphael Overbeck and Nicolas Sendrier. Code-based cryptography. In *Post-quantum cryptography*, pages 95–145. Springer, 2009.

[14] Ernst M Gabidulin. Public_key cryptosystems based on linear codes. *Computers & Security*, 16(3):187–187, 1997.

[15] ThierryP. Berger and Pierre Loidreau. How to mask the structure of codes for a cryptographic use. *Designs, Codes and Cryptography*, 35(1):63–79, 2005.

[16] Christian Wieschebrink. An attack on a modified niederreiter encryption scheme. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 14–26. Springer Berlin Heidelberg, 2006.

[17] Tereza Hrubešová. Klasický strukturální útok na niederreiterův kryptosystém vytvořený nad grs kódy, 2014.

[18] David Stanovský a Univerzita Karlova. Matematicko-fyzikální fakulta. *Základy algebry*. Matfyzpress, 2010.

[19] W. A. Stein et al. *Sage Mathematics Software (Version 6.1.1)*. The Sage Development Team, 2015. `http://www.sagemath.org`.