Slovenská technická univerzita
Fakulta elektrotechniky a informatiky
Ústav informatiky a matematiky

Homomorfné kryptosystémy

(Dizertačná práca)

Michal Mikuš

Školiteľ:
Prof. RNDr. Otokar Grošek, PhD.

Bratislava, máj 2012

Prehlasujem, že prácu som vypracoval samostatne, iba s pomocou literatúry uvedenej v zozname a konzultácií s mojim školiteľom. .....

Chcel by som sa poďakovať prof. Grošekovi za odbornú pomoc, množstvo rád a nápadov, ako aj za prejavenú ochotu pri vedení práce.

Contents

Chapter 1

Úvod

Chapter 2

Prehľad problematiky

V tejto kapitole uvedieme definície potrebných pojmov a to najprv kryptosystémov, ich homomorfných vlastností a bezpečnosti a následne aj krátky súhrn teórie ohľadom diskrétnych mriežok a redukčných algoritmov, keďže jadro práce tvorí kryptosystém založený na tejto algebraickej štruktúre.

Potom uvedieme vybrané plne homomorfné kryptosystémy, ich stručný popis a prípadnú kryptoanalýzu, ak bola publikovaná.

2.1 Základné pojmy

2.1.1 Homomorfný kryptosystém

Homomorfný kryptosystém je taký kryptosystém, ktorý poskytuje určitú funkcionalitu navyše oproti obyčajným kryptosystémom. Umožňuje počítanie so šifrovými textami tak, že výsledok po dešifrovaní zodpovedá nejakej operácii nad príslušnými otvorenými textami. Táto vlastnosť na jednej strane rozširuje jeho možnosti aplikácie, na druhej strane poskytuje útočníkovi väčšie možnosti pri získavaní tajných parametrov.

V literatúre sa uvádzajú viaceré definície homomorfných kryptosystémov. My budeme používať definíciu zavedenú v práci [29], pretože v roku 2009 sa vďaka Gentryho výsledkom zovšeobecnil pohľad na operáciu homomorfizmu a doterajšie definície nevystihujú najnovšie výsledky. Definície vlastností homomorfných kryptosystémov sme prebrali z [11], [12] a [29].

Definícia 2.1.1. *[29] Pod homomorfnou šifrovacou schémou ξ budeme rozumieť päticu $(KeyGen_\xi, E_\xi, D_\xi, Eval_\xi, \lambda)$, kde:*

- $\lambda \in \mathbb{N}$ *je bezpečnostný parameter*
- $KeyGen_\xi(1^\lambda) \rightarrow (\mathcal{P}_\xi, \mathcal{C}_\xi, pk, sk)$ *je znáhodnený polynomiálny algoritmus pre generovanie priestoru otvorených textov $\mathcal{P}_\xi$, priestoru šifrových textov $\mathcal{C}_\xi$, verejného kľúča pk a súkromného kľúča sk. Priestor otvorených textov tvorí buď grupu,*

alebo okruh. [1]

- $E_\xi(pk, m) \to c$ *je znáhodnený polynomiálny šifrovací algoritmus,* $m \in \mathcal{P}_\xi$, $c \in \mathcal{C}_\xi$
- $D_\xi(sk, c) \to m$ *je polynomiálny dešifrovací algoritmus,* $c \in \mathcal{C}_\xi$, $m \in \mathcal{P}_\xi$
- $Eval_\xi(pk, O, c_1, \ldots, c_k) \to c$ *je polynomiálny algoritmus,* $c, c_1, \ldots, c_k \in \mathcal{C}_\xi$, O *je obvod nad* $\mathcal{P}_\xi$

Poznámka 2.1.1. Pod obvodom O nad štruktúrou $\mathcal{P}_\xi$ budeme rozumieť funkciu $O : \mathcal{P}_\xi^k \to \mathcal{P}_\xi$ pre nejaké $k \in \mathbb{N}$ skladajúcu sa z elementárnych operácií príslušnej štruktúry. Napríklad pre okruh sú to operácie sčítania, násobenia a inverzného prvku (vzhľadom na sčítanie).

Označenie 2.1.1. Ak $\mathcal{P}_\xi$ je aditívna (polo)grupa, potom hovoríme o aditívne homomorfnom kryptosystéme (skrátene aditívnom homomorfizme), príslušný algoritmus $Eval_\xi()$ definujeme len nad dvojicou šifrových textov a skrátene označujeme ako A_+. V prípade, že $\mathcal{P}_\xi$ je multiplikatívna (polo)grupa, tak hovoríme o multiplikatívnom homomorfizme a príslušnom algoritme $A_\times(pk, c_1, c_2)$.

Predtým, ako uvedieme zvyšné definície týkajúce sa vlastností homomorfných kryptosystémov, ukážeme príklady homomorfných kryptosystémov, ktoré umožňujú vyhodnocovanie len jednej operácie nad otvorenými textami.

Príklad 2.1.1. Príkladom aditívne homomorfného kryptosystému je systém ElGamal založený eliptických krivkách. V tomto príklade budeme body krivky označovať veľkými písmenami P, Q, R, sčitovanie bodov klasickým $+$ a pre ľubovoľné $m \in \mathbb{Z}$ $mP = P + \ldots + P$, sčítané m-krát.

Nech $(E, +)$ je grupa bodov eliptickej krivky $\mathcal{E}$ definovanej nad $\mathbb{Z}_p$, $p > 3$. Nech ďalej E obsahuje cyklickú podgrupu H, v ktorej je diskrétny logaritmus tažký problém. Potom priestor otvorených textov $\mathcal{P} = E$, priestor zašifrovaných textov $\mathcal{C} = E \times E$ a priestor kľúčov je $\mathcal{K} = \{(\mathcal{E}, B, j, C) : C = j \cdot B\}$, kde bod $B \in E$ je dostatočne veľkého rádu a j je náhodné celé číslo menšie ako p. Parameter j je súkromný, body B, C sú verejné.

Šifrovanie bodu $P \in E$ prebieha v dvoch krokoch. Užívateľ zvolí k, náhodné číslo menšie ako rád bodu B a následne vypočíta body Q, R podľa predpisu:

$$\begin{aligned} Q &= k \cdot B \\ R &= P + k \cdot C \end{aligned}$$

Dešifrovanie šifrového textu (Q, R) prebieha podľa vzorca:

$$P = R - j \cdot Q$$

[1] Keďže časová zložitosť algoritmov sa formálne určuje od dĺžky vstupu, tak sa v definícii používa unárny zápis 1^λ namiesto λ.

Homomorfné sčítanie možno definovať sčítaním zašifrovaných textov po súradniciach. Nech $(Q_1, R_1) = E(P_1)$ a $(Q_2, R_2) = E(P_2)$, potom homomorfným sčítaním týchto dvoch zašifrovaných texov dostávame $(Q_1 + Q_2, R_1 + R_2)$, čo by mal byť zašifrovaný text zodpovedajúci $P_1 + P_2$. Po dosadení do dešifrovacieho predpisu dostávame:

$$\begin{aligned} D((Q_1 + Q_2, R_1 + R_2)) &= R_1 + R_2 - j \cdot (Q_1 + Q_2) \\ &= P_1 + k_1 \cdot C + P_2 + k_2 \cdot C - (jk_1 \cdot B + jk_2 \cdot B) \\ &= P_1 + P_2 + k_1 j \cdot B + k_2 j \cdot B - (jk_1 \cdot B + jk_2 \cdot B) \\ &= P_1 + P_2, \end{aligned}$$

z čoho vyplýva korektnosť homomorfného algoritmu.

Príklad 2.1.2. Najjednoduchším príkladom multiplikatívneho homomorfizmu je známy RSA systém. V ňom $\mathcal{P} = \mathcal{C} = \mathbb{Z}_N$. $E_e(m) = m^e \mod N$ a $D_d(c) = c^d \mod N$. Keďže platí $m_1^e \cdot m_2^e = (m_1 m_2)^e$ tak algoritmus $A_\times$ môžeme definovať jednoducho $A_\times(c_1, c_2) = c_1 \cdot c_2 \mod N$, kde $c_1, c_2 \in \mathcal{C}$.

Definícia 2.1.2. *Hovoríme, že homomorfná schéma ξ je korektná, ak pre ľubovoľné $(\mathcal{P}_\xi, \mathcal{C}_\xi, pk, sk) \leftarrow KeyGen_\xi(1^\lambda)$ platí:*

$$\forall c \in \mathcal{C}_\xi, m \in \mathcal{P}_\xi : c = E_\xi(pk, m) \Rightarrow m = D_\xi(sk, c)$$

Definícia 2.1.3. *Hovoríme, že homomorfná schéma ξ je kompaktná, ak zložitosť jej dešifrovacieho algoritmu D_ξ závisí iba od bezpečnostného parametra λ.*

Definícia 2.1.4. *Množina prípustných obvodov $\mathcal{O}_\xi$ je množina tých obvodov, ktoré je schéma ξ schopná homomorfne vyhodnotiť.*

$$\mathcal{O}_\xi = \{O | \forall m_1, \ldots, m_k \in \mathcal{P}_\xi : D_\xi(Eval_\xi(O, E_\xi(m_1), \ldots, E_\xi(m_k))) = O(m_1, \ldots, m_k)\}$$

Hovoríme, že ξ je korektná pre obvody z $\mathcal{O}_\xi$.

Definícia 2.1.5. *Hovoríme, že schéma ξ je algebraicky homomorfná, ak $\mathcal{P}_\xi$ je okruh a ξ je korektná a kompaktná pre všetky obvody nad $\mathcal{P}_\xi$.*

Označenie 2.1.2. Algebraicky homomorfný kryptosystém sa v literatúre nazýva aj plne homomorfný kryptosystém, resp. plne homomorfné šifrovanie (fully homomorphic encryption). Budeme v práci používať aj toto skrátené pomenovanie, prípadne skratky FHS, FHE.

Okrem uvedených príkladov je známych viacero homomorfných kryptosystémov, ktoré sú buď aditívne, alebo multiplikatívne. Dlho sa však kryptológom nedarilo nájsť algebraický kryptosystém, ktorý by bol zároveň bezpečný aj efektívny. Prvý realizovateľný a (dosiaľ sa verí, že) bezpečný plne homomorfný kryptosystém bol prezentovaný v [12]. Tento kryptosystém nevyhovuje formálne uvedenej definícii 2.1.5, preto bol zavedený trochu slabší pojem a to stupňovito homomorfná trieda kryptosystémov.

Definícia 2.1.6. *Trieda schém* $\{\xi^d | d \in \mathbb{N}\}$ *s rovnakým parametrom* λ *a* $KeyGen$ *algoritmami generujúcimi rovnaký okruh otvorených textov* $\mathcal{P}$ *je stupňovito plne homomorfná, ak každá schéma* ξ^d *je korektná a kompaktná pre obvody nad* $\mathcal{P}$ *hĺbky najviac* d *a všetky jej algoritmy majú zložitosť polynomiálnu od* λ, d *a (v prípade* $Eval_{\xi^d}$*) veľkosti vyhodnocovaného obvodu.*

Na záver uvedieme jednoduchý príklad algebraického homomorfizmu (príklad č.3 z [25]), ktorý nespĺňa základné požiadavky na bezpečnosť. Bol predložený v úvodnej priekopníckej práci hlavne ako motivačný príklad[2] jednoduchého plne homomorfného kryptosystému.

Príklad 2.1.3. [25] Nech $n = p \cdot q$ a p, q sú dve dostatočne veľké utajené prvočísla. Množina otvorených textov nech je $\mathbb{Z}_n$. Označme $+_n, \times_n$ operácie sčítania a násobenia modulo n. Množina zašifrovaných textov bude $\mathbb{Z}_n \times \mathbb{Z}_n$.

Definujme symetrický kľúč $k = (p, q)$ a $E_k(m) = (m \bmod p,\ m \bmod q)$. Algoritmy A_+ (resp. $A_\times$) definujeme ako sčítanie (resp. násobenie) po zložkách, modulo n. Dešifrovanie definujeme spätným počítaním m pomocou Čínskej zvyškovej vety.

Ako bolo ukázané v [4], tento symetrický kryptosystém má bezpečnostné slabiny. Napríklad otvorené texty $m < p, q$ vôbec nezašifruje. Navyše je jednoduché realizovať útok so znalosťou otvoreného textu, ktorý má veľkú pravdepodobnosť odhalenia kľúča.

2.1.2 Definície bezpečnosti

Presne definovať pojem bezpečnosť kryptosystému sa dá len pre špecifickú situáciu. Poznáme niekoľko typických scenárov útoku, kde sa útočník snaží získať nejakú informáciu o otvorenom texte. Rozdeľujeme ich na:

- CPA (Chosen Plaintext Attack) – útočník má k dispozícii iba verejný kľúč schémy
- CCA1 (Chosen Ciphertext Attack) – útočník mal pred útokom možnosť dešifrovať niekoľko šifrových textov podľa jeho výberu
- CCA2 (Adaptive Chosen Ciphertext Attack) – útočník má neobmedzený prístup ku dešifrovaciemu orákulu s podmienkou, že si nemôže nechať dešifrovať cieľový šifrový text – výzvu.

Definujeme preto bezpečnosť aspoň proti týmto typom útokov. Tieto situácie sú zahrnuté v nasledujúcej všeobecnej hre medzi dvoma účastníkmi, vyzývateľom a útočníkom $\mathcal{A}$, ktorá pozostáva z niekoľkých fáz. Popíšeme najprv hru, ktorá vystihuje CCA2-útok:

Inicializácia. Vyzývateľ vytvorí kryptosystém ξ a kľúče $(pk, sk) \leftarrow KeyGen_\xi(\lambda)$. Útočníkovi $\mathcal{A}$ zverejní verekný kľúč pk a vynuluje hodnotu c^*.

[2]Doslova: "The results presented here give a basis for some optimism about finding useful privacy homomorphisms; the examples given here are suggestive if not very practical."

Dotazy. Útočník $\mathcal{A}$ môže posielať požiadavky na dešifrovanie zašifrovaných textov $c_i \neq c^*$. Vyzývateľ odpovie výstupom z $D_\xi(sk, c_i)$. Útočník má právo na požiadavky pred aj po fáze Výzva.

Výzva. $\mathcal{A}$ vyberie dva otvorené texty p_0, p_1 rovnakej dĺžky a pošle vyzývateľovi. Ten vyberie náhodný bit $b \in \{0, 1\}$ a pošle útočníkovi $c^* = E_\xi(pk, p_b)$.

Odhad. Útočník pošle $b' \in \{0, 1\}$ vyzývateľovi. Hovoríme, že útočník vyhral, ak $b = b'$.

Pri scenári CCA1 útoku definujeme, že útočník $\mathcal{A}$ má právo na dešifrovacie dotazy iba pred Výzvou. Pri scenári CPA nemá útočník právo na žiadne dešifrovacie dotazy. V každom prípade definujeme útočníkovu výhodu oproti schéme ξ ako

$$\mathrm{Adv}(\mathcal{A}, \xi, \lambda) = |Pr[b = b'] - 1/2|,$$

čo znamená, že útočníkova výhoda je to, o koľko dokáže mať lepšiu pravdepodobnosť výhry oproti náhodnému tipovaniu. Vždy platí, že $0 \leq \mathrm{Adv} \leq 1/2$.

Definícia 2.1.7. *Hovoríme, že schéma ξ je semanticky odolná voči (CPA, CCA1, CCA2)-útoku, ak neexistuje útočník, ktorý by v polynomiálnom čase vedel získať viac ako zanedbateľnú výhodu v príslušnej (CPA,CCA1,CCA2)-hre.*

Definícia pre homomorfnú schému sa nelíši od klasickej. Pre širšie možnosti útokov, ako aj iné ciele útočníka odporúčame [1].

2.2 Diskrétne mriežky

V tejto časti zhrnieme základné pojmy z teórie diskrétnych mriežok a ideálových mriežok, pretože na tejto štruktúre sú založené najdôležitejšie plne homomorfné kryptosystémy popísané v časti 2.4. Taktiež spomenieme základné ťažké problémy, na ktorých je založená bezpečnosť týchto plne homomorfných systémov. Problematika je tu popísaná stručne, pre detailnejšie štúdium odrúčame napr. [22].

Značenie. V tejto práci budeme narábať iba s riadkovými vektormi označenými malými písmenami, napr. $\overrightarrow{b_1}, \overrightarrow{b_2}, \overrightarrow{b_3}$ a maticu bude teda tvoriť stĺpec riadkových vektorov $B = (b_1, b_2, b_3)^T$. Označením $\| \overrightarrow{b} \|$ rozumieme štandardnú Euklidovskú normu vektora $\overrightarrow{b}$.

Pre ľubovoľné $a, b \in \mathbb{Z}$ bude $[a]_b$ označovať redukciu a modulo b zobrazenú do polootvoreného intervalu $[-b/2, b/2)$. Napríklad $8 \bmod 3 = 2$, ale $[8]_3 = -1$.

Pre ľubovoľné $x \in \mathbb{R}$ budeme značiť najbližšie celé číslo ku x ako $\lfloor x \rceil$. Toto značenie prirodzene rozširujeme aj na vektory, prípadne matice, kde $\lfloor \cdot \rceil$ je aplikované na každý prvok danej štruktúry.

2.2.1 Definície

Definícia 2.2.1. *Nech $n, m \in \mathbf{N}$, $n \leq m$, a nech $\overrightarrow{b_1}, \ldots, \overrightarrow{b_n} \in \mathbb{R}^m$ je n lineárne nezávislých vektorov a $B = (\overrightarrow{b_1}, \ldots, \overrightarrow{b_n})^T$ zodpovedajúca matica. Potom mriežka $\mathcal{L}$ generovaná vektormi $\overrightarrow{b_1}, \ldots, \overrightarrow{b_n}$ je definovaná ako:*

$$\mathcal{L} = L(B) = \left\{ \sum_{i=1}^{n} x_i \overrightarrow{b_i} : x_i \in \mathbb{Z} \right\}$$

a hovoríme, že mriežka $\mathcal{L}$ má dimenziu n a bázu B.

Mriežka môže mať viac ako jednu bázu. Pre ľubovoľné dve bázy B_1, B_2 tej istej mriežky platí, že $B_1 = U_1 \cdot B_2$ a $B_2 = U_2 \cdot B_1$ pre nejaké dve celočíselné matice U_1 a U_2. Z týchto rovností vyplýva, že $det(B_1) = det(U_1) \cdot det(B_2) = det(U_1) \cdot det(U_2) \cdot det(B_1)$ a keďže $det(U_i)$ musí byť celočíselný, tak $det(U_i) = \pm 1$. Z toho vyplýva, že $|det(B_1)| = |det(B_2)|$.

Determinant mriežky $L(B)$ definujeme ako $\sqrt{det(BB^T)}$. Ak pre mriežku $\mathcal{L}$ platí $n = m$, tak hovoríme, že $\mathcal{L}$ má plnú hodnosť a $det(\mathcal{L}) = |det(B)|$ pre ľubovoľnú bázu B mriežky $\mathcal{L}$.

Definícia 2.2.2. *Nech $B = \{b_{i,j} \in \mathbb{R}^{n \times n}\}$. Hovoríme, že matica B je v Hermiteho normálnom tvare (označujeme HNF(B)), ak platí:*

- *B je v dolnom trojuholníkovom tvare, teda $\forall i > j : b_{i,j} = 0$*
- *všetky prvky $b_{i,j}$ sú kladné a v každom stĺpci je najväčší prvok na diagonále, teda $\forall i > j : 0 \leq b_{i,j} \leq b_{j,j}$*

Každú celočíselnú maticu vieme previesť do Hermiteho normálneho tvaru pomocou jednoduchého efektívneho algoritmu využívajúc pritom iba mriežkové operácie, t.j. pričítanie celočíselného násobku vektora k inému vektoru. Navyše pre ľubovoľné dve bázy B_1 a B_2 tej istej mriežky platí, že $\mathrm{HNF}(B_1) = \mathrm{HNF}(B_2)$, takže HNF je idálna voľba pre verejný kľúč ľubovoľnej mriežky plnej hodnosti.

Definícia 2.2.3. *Nech $\mathcal{L}$ je mriežka a B jej báza. Potom základný rovnobežnosten bázy B je definovaný takto:*

$$\mathcal{P}(B) = \left\{ \sum_{i=1}^{n} x_i \overrightarrow{b_i} : x_i \in \langle -1/2, 1/2) \right\}$$

Objem základného rovnobežnostena je totožný s determinantom mriežky, takže je konštantný pre ľubovoľnú bázu. Pre rôzne bázy tej istej mriežky sa ale mení tvar ich základného rovnobežnostena a podľa tohto tvaru sa posudzuje "kvalita" konkrétnej bázy. Formálne kvalitu bázy posudzujeme podľa toho, ako veľmi sú jej vektory na seba kolmé, vyjadruje to tzv. odchýlka od kolmosti[3] definovaná nasledovne:

[3] z angl. orthogonality defect.

Definícia 2.2.4. *Nech* $B = (\overrightarrow{b_1}, \ldots, \overrightarrow{b_n})^T$ *je báza mriežky. Potom odchýlka od kolmosti* $\delta(B)$ *je definovaná vzťahom:*

$$\delta(B) = \frac{\prod \| \overrightarrow{b_i} \|}{|det(L(B))|}$$

Platí, že $\delta(B) \geq 1$ a rovnosť nastáva vtedy a len vtedy, ak sú vektory $\overrightarrow{b_i}$ na seba navzájom kolmé.

Pre mriežku $\mathcal{L} \subset \mathbb{R}^n$ plnej hodnosti (s bázou B) platí, že vektorový priestor $\mathbb{R}^n$ sa dá rozložit na rovnobežnosteny rovnakého tvaru, ktoré sú len posunutím základného rovnobežnostena $\mathcal{P}(B)$ o nejaký mrežový vektor. Formálne vieme definovať zobrazenie $\phi : \mathbb{R}^n \to \mathcal{P}(B)$, ktoré vektoru $\overrightarrow{x} \in \mathbb{R}^n$ priradí také $\overrightarrow{y} \in \mathcal{P}(B)$, že existuje $\overrightarrow{v} \in \mathcal{L}$, pre ktoré platí $\overrightarrow{v} = \overrightarrow{x} - \overrightarrow{y}$. Vektor $\overrightarrow{y}$ sa dá efektívne vypočítať predpisom:

$$\overrightarrow{y} = \overrightarrow{x} - B \cdot \lfloor B^{-1} \cdot \overrightarrow{x} \rceil.$$

Vektor $\overrightarrow{y}$ je definovaný jednoznačne a zapisuje sa niekedy aj ako $\overrightarrow{x}$ mod B.

Definícia 2.2.5. *Nech* $\mathcal{L}$ *je mriežka a* d *je jej dimenzia. Potom pre* $1 \leq i \leq d$ *definujeme* $\lambda_i(\mathcal{L})$ *ako najmenšie číslo* $r \in \mathbb{R}$ *také, že v mriežke* $\mathcal{L}$ *existuje* i *lineárne nezávislých vektorov veľkosti najviac* r*.*

V kryptografii nás špeciálne zaujíma najkratší vektor mriežky, pretože väčšinou vieme zostrojiť mriežku, ktorej najkratší vektor (prípadne jeho násobok) reprezentuje nejaký tajný parameter ktyptosystému. Nájdenie $\overrightarrow{v} \in \mathcal{L}$ tak, aby $\| \overrightarrow{v} \| = \lambda_1(\mathcal{L})$ patrí medzi ťažké problémy popísané v časti 2.3.

2.2.2 Ideálové mriežky

Ideálová mriežka[4] je pojem, ktorý zaviedol C.Gentry vo svojej práci [12]. Je to vlastne ideál špeciálneho okruhu $R = \mathbb{Z}[x]/(f(x))$, pre nejaký ireducibilný polynóm $f(x) \in \mathbb{Z}[x]$. Na ľubovoľný prvok tohto ideálu sa totiž môžeme pozerať ako na vektor jeho koeficientov a tieto vektory tvoria diskrétnu mriežku, ktorú Gentry nazval ideálová. Formálnu definíciu tohto pojmu uvádzame z práce [29].

Definícia 2.2.6. *Nech* $f(x) \in \mathbb{Z}[x]$ *je ireducibilný monický polynóm stupňa* n *a* $R = \mathbb{Z}[x]/(f(x))$ *je okruh. Ďalej nech* $S = \{g(x) \ mod \ f(x) | g(x) \in \mathbb{Z}[x]\}$ *je množina štandardných reprezentantov tried rozkladu okruhu* R*, izomorfizmus* $h_1 : R \to S$ *zobrazuje triedu rozkladu na jej reprezentanta a izomorfizmus* $h_2 : S \to \mathbb{Z}^n$ *zobrazuje polynóm na vektor koeficientov, teda*

$$h_1(A) = g(x) \ mod \ f(x), A \in R, g(x) \in A$$

$$h_2(g_{n-1}x^{n-1} + \ldots + g_1 x + g_0) = (g_{n-1}, \ldots, g_0)$$

Potom izomorfizmus $h : R \to \mathbb{Z}^n$ *definovaný ako* $h(A) = h_2(h_1(A))$ *je izomorfizmus vzhľadom na sčítanie a mriežku* $\mathcal{L} \subseteq \mathbb{Z}^n$ *nazveme ideálová, ak* $\mathcal{L} = h(I)$ *pre nejaký ideál* $I \subseteq R$*.*

[4] z angl. *ideal lattice*.

Označenie 2.2.1. Pri ideálových mriežkach budeme podobne ako v [12], [11] a [29] zjednodušovať označenie a prvky z R nazývať aj polynómy aj vektory. Ak budeme používať operácie z okruhu R na vektory z $\mathcal{L}$, vždy sa tým myslia operácie na príslušných prvkoch z okruhu R. Teda zápis $\overrightarrow{v} \cdot x^i$ bude znamenať $h^{-1}(v) \cdot h_1^{-1}(x^i)$ pre nejaký vektor $\overrightarrow{v} \in \mathbb{Z}^n$.

Definícia 2.2.7. *Nech $\overrightarrow{v} \in R$. Potom $B_{\overrightarrow{v}} = \{\overrightarrow{v_i} | \overrightarrow{v_i} = \overrightarrow{v} \cdot x^i, 0 \leq i < n\}$ nazývame rotačná báza mriežky $L(B_{\overrightarrow{v}})$ prislúchajúca ku $\overrightarrow{v}$.*

Tvrdenie 2.2.1. *[29] Nech $\overrightarrow{v} \in R$ a $I = (\overrightarrow{v})$ je hlavný ideál. Potom rotačná báza prislúchajúca $\overrightarrow{v}$ je bázou ideálovej mriežky $h(I)$.*

Proof. □

Pre čiastočne homomorfné schémy založené na ideálových mriežkach je pre daný okruh dôležité ohraničenie nárastu vektora pri sčítaní alebo násobení. Z trojuholníkovej nerovnosti vidieť, že $\|\overrightarrow{u} + \overrightarrow{v}\| \leq \|\overrightarrow{u}\| + \|\overrightarrow{v}\|$. Dá sa dokázať (podobne ako v [29]), že $\|\overrightarrow{u} \cdot \overrightarrow{v}\| \leq \|\overrightarrow{u}\| \cdot \overrightarrow{v}\| \cdot \gamma_{\text{Mult}}(R)$, pre nejakú konštantu $\gamma_{\text{Mult}}(R)$, ktorá závisí iba od konkrétneho okruhu R. V prípade schémy 2.4.5 je $\gamma_{\text{Mult}}(R) = \sqrt{n}$.

Definícia 2.2.8. *Nech $R = \mathbb{Z}[x]/(f(x))$ a $\mathcal{L} \subseteq R$ je ideálová mriežka. Inverznú mriežku $\mathcal{L}^{-1}$ ku $\mathcal{L}$ definujeme*

$$\mathcal{L}^{-1} = \{\overrightarrow{u} \in \mathbb{Q}[x]/(f(x)) | \forall \overrightarrow{v} \in \mathcal{L} : \overrightarrow{v} \cdot \overrightarrow{u} \in R\}$$

Lema 2.2.2. *[29] Nech B je rotačná báza ideálovej mriežky $\mathcal{L}$ plnej dimenzie prislúchajúca vektoru $\overrightarrow{v}$. Potom B^{-1} je rotačná báza inverznej mriežky $\mathcal{L}^{-1}$ prislúchajúca k vektoru $(\overrightarrow{v})^{-1}$.*

2.3 Ťažké problémy

Definícia 2.3.1. *(Nájdenie $\gamma(n)$-najkratšieho vektora ($\gamma(n)$-SVP)). Daná je mriežka $\mathcal{L}$ dimenzie n a jej báza B. Problém $\gamma(n)$-SVP spočíva v nájdení nenulového vektora $v \in \mathcal{L}$, pre ktorý platí $\|v\| \leq \gamma(n) \cdot \lambda_1(\mathcal{L})$.*

Pre konštantný aproximačný faktor patrí $\gamma(n)$-SVP medzi NP-úplné problémy. Túto hypotézu vyslovil už v 1981 van Emde Boas, ale dôkaz bol sformulovaný až v 2001 v práci [19].

Existuje aj verzia $\gamma(n)$-SVP, ktorej cieľom je nájsť vektor $v \in \mathcal{L}$, pre ktorý platí:

$$\|v\| \leq \gamma(n) \cdot det(\mathcal{L})^{1/n}$$

Tento problém sa označuje ako Hermitov-SVP a používa sa v prípadoch, keď dĺžka najkratšieho vektora nie je známa a potrebujeme nejako vyčísliť dosiahnutý aproximačný faktor.

Definícia 2.3.2. *(Aproximačný problém NSD (AGCDP)). Vstupné sú štyri parametre* (k, η, ρ, γ)*. Vygeneruje sa náhodné* η*-bitové prvočíslo* p *a* k *čísel* $x_1, \ldots, x_k$*, kde*

$$x_i = pq_i + r_i, \text{ pričom } q_i \in_R [0, 2^\gamma/p) \text{ a } r_i \in_R (-2^\rho, 2^\rho)$$

pričom všetky q_i *a* r_i *sú celé čísla. Problém spočíva v nájdení* p *pre dané* (k, η, ρ, γ) *a* $x_1, \ldots, x_k$*.*

Tento problém bol prezentovaný v [14], je na ňom postavená bezpečnosť kryptosystému [28]. V [5] Chen a Nguyen publikovali útok, ktorý rieši daný problém v čase $\sqrt[4]{(T^3)}$, kde T je čas úplného prehľadávania.

Definícia 2.3.3. *(Existencia riedkej podmnožiny s daným súčtom (SSSP*[5]*)). Dané sú tri čísla* $s, t, q \in \mathbb{Z}_+$*. Zvolí sa náhodne* $b \in \{0, 1\}$*. Pre* $b = 0$ *sa vygeneruje množina* $T = \{a_1, a_2, \ldots, a_t\}$*, kde* $a_i \in [-q/2, q/2)$ *s podmienkou, že existuje množina indexov* $S \subseteq \{1, \ldots, t\}$*,* $|S| = s$ *a* $\sum_{i \in S} a_i = 0 \ mod \ q$*. Pre* $b = 1$ *sa vygeneruje množina* T*, ale bez obmedzujúcej podmienky. Problém spočíva v určení* b*, ak je dané* (T, s, q)*.*

Tento problém bol analyzovaný v [16], [20], [21], [18] a útoky sú známe len pre nízke hodnoty parametrov s, t, q. Nguyen a Shparlinski [20] prezentovali útok pomocou redukcie diskrétnych mriežok pre hodnotu $t \leq \log(q)$. Bezpečnosť schémy [13] sa redukuje na tento problém.

Na vektorovej verzii tohto problému je postavená bezpečnosť Gentryho plne homomorfnej schémy [12] (popísaná v časti 2.4.5.

Definícia 2.3.4. *(Existencia riedkej podmnožiny vektorov s daným súčtom (SVSSP)). Dané sú dve čísla* $s, t \in \mathbb{Z}_+$ *a báza* B *mriežky* $\mathcal{L}$*. Zvolí sa náhodne* $b \in \{0, 1\}$*. Pre* $b = 0$ *sa vygeneruje množina* $T = \{\overrightarrow{v_1}, \overrightarrow{v_2}, \ldots, \overrightarrow{v_t}\}$*, kde* $\overrightarrow{v_i} \in \mathcal{P}(B)$ *s podmienkou, že existuje množina indexov* $S \subseteq \{1, \ldots, t\}$*,* $|S| = s$ *a* $\sum_{i \in S} \overrightarrow{v_i} = 0 \ mod \ B$*. Pre* $b = 1$ *sa vygeneruje množina* T*, ale bez obmedzujúcej podmienky. Problém spočíva v určení* b*, ak je dané* (T, s, q)*.*

2.4 Prehľad navrhovaných algebraických homomorfizmov

V tejto časti zhrnieme publikované plne homomorfné kryptosystémy. Prelomový bol piaty (v časti 2.4.5), kde C.Gentry publikoval ideu čiastočne homomorfnej schémy a aj postup ako z nej za určitého predpokladu zostrojiť plne homomorfný kryptosystém. Nasledujúce tri popísané kryptosystémy sa držali jeho myšlienky a posledný z nich 2.4.8 prvýkrát úspešne realizoval plne homomorfnú schému. Na záver už len vymenujeme najnovšie kryptosystémy, ktoré boli publikované nedávno.

Pre kompletnosť uvedieme, že v práci [25] boli navrhnuté štyri homomorfné kryptosystémy. Algebraický bol jeden z nich a je uvedený v príklade 2.1.3.

[5] z angl. Sparse Subset Sum Problem.

2.4.1 J.D.Ferrer - 1996

Autor v [10] prezentoval symetrický kryptosystém, ktorý bol vylepšením systému z príkladu 2.1.3. Tento kryptosystém bol navrhnutý tak, aby odolal útoku so znalosťou otvoreného textu (tzv. known plaintext attack – KPA). Pozostáva z verejných parametrov a súkromných. Verejné slúžia na to, aby nekompetentná osoba mohla vykonávať nad zašifrovanými textami príslušné operácie.

Popis kryptosystému:
Množina otvorených textov $OT = \mathbb{Z}_n$ a operácie nad otvorenými textami, ktoré bude možno počítať pomocou zašifrovaných textov sú sčítanie a násobenie modulo n (ozn. $+_n$, resp. $\times_n$).

Množina zašifrovaných textov je $ZT = (\mathbb{Z} \times \mathbb{Z})^d$ a operácie nad ZT budú popísané neskôr.

- *Súkromný kľúč:* dve dostatočne veľké prvočísla p a q, číslo $r_p \in \mathbb{Z}_p$, ktoré generuje dostatočne veľkú multiplikatívnu podgrupu $(\mathbb{Z}_p \setminus \{0\}, \cdot)$ a číslo $r_q \in \mathbb{Z}_q$ s analogickou vlastnosťou vzhľadom na $(\mathbb{Z}_q \setminus \{0\}, \cdot)$.

- *Verejné parametre:* celé nezáporné čísla n a d, kde d môže byť ľubovoľné[6] a $n = pq$.

- *Šifrovanie:* správu $a \in \mathbb{Z}_n$ rozdelíme náhodne na d častí $a_1, \ldots, a_d \in \mathbb{Z}_n$ tak, aby $a = \sum_{j=1}^{d} a_j \bmod n$.[7] Potom vypočítame páry $E_j(a) = (a_j \cdot r_p^j \bmod p,\ a_j \cdot r_q^j \bmod q)$. Zašifrovaná správa $E(a)$ bude rovná $(E_1(a) \ldots, E_d(a))$ – vektoru dĺžky $2d$. Kvôli prehľadnosti označíme $E_j(a) = (b_j, c_j)$ a $\mathcal{E}_p(a) = (b_1, \ldots, b_d)$ a $\mathcal{E}_q(a) = (c_1, \ldots, c_d)$.

- *Dešifrovanie:* najprv vypočítame skalárny súčin po zložkách $E_j(a)$ s (r_p^{-j}, r_q^{-j}), čím dostaneme $a_j \bmod p$ a $a_j \bmod q$. Tieto dvojice potom sčítame a dostaneme $a \bmod p$ a $a \bmod q$. Pomocou Čínskej zvyškovej vety jednoznačne dopočítame $a \in \mathbb{Z}_n$.

- *Výpočet* $Eval(a_1 +_n a_2)$: je definovaný ako súčet zašifrovaných textov po zložkách nad $\mathbb{Z}_n$ – tým sa koeficienty pri rovnakých mocninách r_p^j a r_q^j sčítajú a pri dešifrovaní dostaneme $a_1 + a_2 \bmod n$.

- *Výpočet* $Eval(a_1 \times_n a_2)$: je definovaný podobne ako násobenie polynómov – časti vektorov $\mathcal{E}_p(a_1)$ a $\mathcal{E}_p(a_2)$ sa roznásobia ako polynómy a výsledkom bude vektor $\mathcal{E}_p(a_1 \times_n a_2)$ o dĺžke rovnej súčtu dĺžok príslušných častí. Časti $\mathcal{E}_q(a_1)$ a $\mathcal{E}_q(a_2)$ sa roznásobia analogicky.

Poznámka 2.4.1. Násobenie dvoch zašifrovaných textov je možné vykonať aj bez znalosti p a q, poradie zložiek vektora $\mathcal{E}_p(a)$ (resp. $\mathcal{E}_q(a)$) udáva exponent čísla r_p (resp. r_q). Výsledný vektor bude už mať koeficienty nad $\mathbb{Z}_n$.

[6] Ako vyplynie z následnej kryptoanalýzy, parameter d ovplyvňuje efektívnosť a bezpečnosť, pri znalosti d párov OT-ZT je možné vypočítať súkromný kľúč.

[7] Kvôli presnosti dodávame, že by nemal nastať prípad a_i rovné p resp. q.

Poznámka 2.4.2. Bezpečnosť kryptosystému je založená (aj) na obtiažnosti problému faktorizácie, preto bola voľba verejných a súkromných parametrov popísaná neformálnym spôsobom. Všimnime si, že voľbou parametrov $d = 1$, $r_p = 1$, $r_q = 1$ dostávame systém z príkladu **??**.

Poznámka 2.4.3. Verejný parameter n sa môže kvôli zvýšenej bezpečnosti utajiť. V práci [10] sa nepíše, ako utajenie zlepší bezpečnosť, ale autor uvádza, že zverejnenie n zlepší efektívnosť výpočtov nad ZT, pretože zložky vektorov možno po každom výpočte redukovať modulo n. Pri utajení n by všetky výpočty so zašifrovanými textami prebiehali nad $\mathbb{Z}$.

Kryptoanalýza

V práci [10] bolo ukázané, že kryptosystém nie je bezpečný pre voľbu $d = 1$. Autor taktiež sám analyzuje bezpečnosť celého systému z hľadiska počtu známych OT-ZT párov. V prípade, že útočník pozná l OT-ZT párov (pri ktorých bol použitý ten istý kľúč (p, q, r_p, r_q)), tak pozná správy $a_1, \ldots, a_l$ a príslušné vektory $\mathcal{E}_p(a_1), \ldots, \mathcal{E}_p(a_l)$.

Neznáme je preňho rozdelenie správ a_i na časti $a_{i1} + a_{i2} + \ldots + a_{il}$ modulo n a parametre p a r_p^j. Útočník však vie, že $a_{ij} \cdot r_p^j = b_{ij} + k_{ij} \cdot p$ pre každé $i = 1, \ldots, d$ a $j = 1, \ldots, l$.

J.D.Ferrer v pôvodnom článku považoval za neznáme aj k_{ij} a systém považoval za riešiteľný iba v prípade ak útočník vedel faktorizáciu $n = pq$ a $l > d$. O pár rokov neskôr však J.H.Cheon a kol. v článkoch [6] a [7] ukázali spôsob ako tento systém zjednodušiť a jednoducho eliminovať premenné k_{ij} bez znalosti faktorizácie čísla n.

Obe analýzy sú založené na rovnakom princípe, preto uvedieme len analýzu z [7]. Pri verejnom parametri n uvádzajú postup, ako môže útočník s pravdepodobnosťou blížiacou sa k jednej, za podmienky $l \geq d + 1$, určiť najprv parametre p, q a následne r_p a r_q.

Pre nájdenie parametra p môže útočník dosadiť $a_{ij} = b_{ij} \cdot r_p^{-j} + k_{ij} \cdot p$ do rovnice $a_i = a_{i1} + a_{i2} + \ldots + a_{il} \bmod p$ pre každé $j = 1, \ldots, l$ a dostane sústavu:

$$\begin{aligned} b_{11} \cdot r_p^{-1} + b_{12} \cdot r_p^{-2} + \ldots + b_{1d} \cdot r_p^{-d} - a_1 &\equiv 0 \bmod p, \\ b_{21} \cdot r_p^{-1} + b_{22} \cdot r_p^{-2} + \ldots + b_{2d} \cdot r_p^{-d} - a_2 &\equiv 0 \bmod p, \\ &\ldots \\ b_{l1} \cdot r_p^{-1} + b_{l2} \cdot r_p^{-2} + \ldots + b_{ld} \cdot r_p^{-d} - a_l &\equiv 0 \bmod p. \end{aligned}$$

Keďže hodnoty r_p^j sú neznáme, tak si ich môžme substituovať za neznáme X_j a dostávame lineárnu sústavu kongruencií. Pravdepodobnosť udalosti, že vybrané OT-ZT páry sú lineárne nezávislé [8], je väčšia ako $e^{-\frac{4}{p-1}}$ – presný dôkaz je v prácach [7] aj [6]. V tom prípade je ale determinant matice tejto sústavy nenulový modulo n.

Na druhej strane homogénna sústava 2.1 má netriviálne riešenie modulo p, a teda jej determinant musí byť nulový modulo p. Z uvedeného vyplýva, že $GCD(det(M), n) = p$.

[8] Jeden OT-ZT pár uvažujeme ako vektor z $\mathbb{Z}_n^{d+1}$.

Parameter q dopočítame ako n/p. Pri znalosti p už z uvedenej sústavy nad $\mathbb{Z}_p$ vieme vypočítať neznámu $X_1 = r_p$.

V článkoch [7] a [6] je takisto uvedený postup, ako môže útočník pomocou $(d+2)$ *OT-ZT* párov vypočítať tajný kľúč s pravdepodobnosťou idúcou k jednej pre dostatočne veľké p.

Záver
Uvedený algebraický homomorfizmus je bezpečný len v prípade, že útočník nepozná viac ako d dvojíc *OT-ZT*. V opačnom prípade môže s vysokou pravdepodobnosťou vypočítať súkromný kľúč a dešifrovať ľubovoľný *ZT*. Navyše dĺžka zašifrovaných textov stúpa s každým vykonaným homomorfným násobením na dvojnásobnú, takže pri "bezpečnej voľbe" d bude prakticky nepoužiteľný.

2.4.2 J.D.Ferrer - 2002

Druhý návrh od D.Ferrera z [8] bol vylepšením prvého, pričom hlavná myšlienka ostala taká istá. Správa sa aj v tomto prípade náhodne rozdelí na súčet niekoľkých častí a zašifruje sa ako vektor koeficientov nejakého polynómu.

Popis kryptosystému:

- *Verejné parametre:* dostatočne veľké $m \in \mathbb{Z}$, ktoré má viacero malých deliteľov a celé číslo $d > 2$.
- *Súkromný kľúč:* $m' \in \mathbb{Z}$, ktoré je malým deliteľom m a $r \in \mathbb{Z}_m^*$.

Na základe týchto parametrov definujeme okruh $\mathbf{R} \stackrel{\text{def}}{=} \mathbb{Z}_m[X]$. Zašifrované texty budú prvky $\mathbf{R}$. Otvorené texty budú prvky okruhu $\mathbb{Z}_{m'}$ s operáciami modulárneho sčítania a násobenia.

- *Šifrovanie:* pre správu $a \in \mathbb{Z}_{m'}$ vyberieme náhodne polynóm $p(x) \in \mathbf{R}$, pre ktorý platí $p(0) = 0$ a $p(1) = a$. Vypočítame polynóm $q(x) = p(r \cdot x)$. $E_{m',r}(a) \stackrel{\text{def}}{=} q(x)$. Zašifrovaný text tvoria koeficienty polynómu $q(x)$.
- *Dešifrovanie:* vypočítame $D_{m',r}(q(x)) = q(r^{-1}) \bmod m'$.
- *Výpočet* $A(a_1 + a_2)$: je definovaný ako súčet vektorov po zložkách.
- *Výpočet* $A(a_1 \times a_2)$: je definovaný ako klasický súčin polynómov modulo m. Poznamenajme, že každým súčinom rastie stupeň výsledného polynómu a teda aj dĺžka zašifrovaného textu.

Poznámka 2.4.4. Aj keď to v práci nebolo explicitne uvedené, tento kryptosystém je tiež symetrický.

V práci [8] sa uvádza aj dôkaz bezpečnosti prezentovaného kryptosystému. Bezpečnosť je žiaľ postavená na tvrdení, že útočník nedokáže zistiť tajný kľúč z ľubovoľnej ohraničenej množiny *OT-ZT* párov. Toto tvrdenie je síce pravdivé – nie je známy spôsob, ako zistiť tajný kľúč – ale nevyplýva z neho bezpečnosť spomínaného algebraického kryptosystému. D.Wagner totiž v [30] uvádza spôsob akým je pomocou malej množiny *OT-ZT* párov možné rozšifrovať ľubovoľný *ZT* bez toho, aby útočník musel počítať tajný kľúč.

Kryptoanalýza

D.Wagner v práci [30] popísal útok, ktorý rozdelil do dvoch krokov. Nájdenie m' a nájdenie $r \bmod m'$ – tu je rozdiel oproti D.Ferrerovej definícii, lebo útočník nepotrebuje hodnotu $r \in \mathbb{Z}_m$, stačí mu jej zvyšok po delení m'.

Nájdenie m'. Predpokladajme, že útočník má prístup k malej množine párov *OT-ZT*,$(a_i, q_i(x))$, kde $q_i(x) = E_{m',r}(a_i)$. Pre každé i definujme polynóm $f_i(x) = q_i(x) - a_i$. Teraz platí $f_i(r^{-1}) \equiv 0 \bmod m'$. Takže polynómy f_i redukované modulo m' majú spoločný koreň r^{-1}. Môžeme teda počítať $R_i = Res(f_i, f_{i+1})$ pre všetky nepárne i, kde funkcia $Res(f, g)$ predstavuje rezultant polynómov f a g. Vieme, že hodnota $R_i \in \mathbb{Z}_m$ a že je náhodným násobkom m', t.j. $R_i \equiv 0 \bmod m'$. Je tomu tak preto, lebo f_i a f_j majú spoločný koreň.

Je známe, že dve náhodne zvolené celé čísla majú najväčší spoločný deliteľ rovný 1 s pravdepodobnosťou $6/\pi^2$. Z toho vyplýva, že $GCD(R_i, R_j) = m'$ s takou istou pravdepodobnosťou $6/\pi^2$. Podľa [8], Lemy 4, platí[9] $P[GCD(R_1, \ldots, R_n) = m'] \approx \frac{1}{\zeta(n)}$ pre $m \to \infty$. Keďže $\frac{1}{\zeta(10)} \doteq 0.999006$, tak útočníkovi stačí 20 párov na to, aby s pravdepodobnosťou 0.999 získal m'.

Nájdenie $r \bmod m'$. Najjednoduchšie riešenie je skúšanie všetkých $r < m'$. V prípade, že m' je príliš veľké a prehľadávanie všetkých $r < m'$ príliš zložité, tak je možné získať $r \bmod m'$ nasledovným spôsobom: uvažujme d známych *OT-ZT* párov $(a_i, q_i(x))$. Keďže $q_i(r^{-1}) \equiv a_i \pmod{m}'$, tak neznámou v tejto sústave sú hodnoty r^{-j} pre $j = 1, \ldots, d$. Substitúciou Y_j za X^j v každej rovnici $q_i(X) \equiv a_i \pmod{m}'$ dostávame už lineárnu sústavu d rovníc o d neznámych. Vieme, že dosadenie $Y_j = r^{-j}$ je správnym riešením tejto sústavy. Štandardnou Gaussovou eliminačnou metódou vieme nájsť jej riešenie v polynomiálnom čase $O(d^3(\log m')^2)$, za predpokladu, že matica tvoriaca sústacu je invertibilná. Táto pravdepodobnosť je nezávislá od d a nie je zanedbateľná, takže niekoľkonásobným opakovaním tohto postupu dostaneme riešenie $Y_1, \ldots, Y_d \in \mathbb{Z}_{m'}$. Prvok Y_1 je potom riešením pre $r' = r \bmod m'$.

Záver

Pri "rozumnej" voľbe parametrov odporúčanej v [8] je možné aplikovať Wagnerov útok. Jediná konfigurácia systému, kedy spomínaný útok nie je aplikovateľný je m' dostatočne veľké, aby sa nedalo prehľadať r hrubou silou a počet známych *OT-ZT* párov n je menší ako d. Tieto nastavenia majú však drastický dopad na efektívnosť systému: ak by sme chceli bezpečne zašifrovať $n = 100$ správ, potrebovali by sme $d > 100$, čiže zašifrované správy by boli 100-násobne dlhšie. Po niekoľkých násobeniach by

[9] $\zeta(n) = \sum_{i=1}^{\infty} i^{-n}$ je Riemanova ζ-funkcia.

dĺzka ZT dosiahla rádovo 1000-násobok otvoreného textu, čo je v praxi nepoužiteľné.

2.4.3 D.Boneh, J.Goh, K.Nissim - 2005

Autori v [2] prezentujú asymetrický kryptosystém založený na Paillierovej konštrukcii [23], ktorý je aditívny a ktorý umožňuje navyše jedno násobenie. Tento kryptosystém zatiaľ nebol narušený, okrem jediného povoleného násobenia má však ešte jeden nedostatok: po vyhodnotení všetkých operácií nesmie výsledná hodnota presiahnuť vopred zvolený interval[10].

Kryptosystém využíva dve multiplikatívne konečné grupy $\mathbb{G}$ a $\mathbb{G}_1$ rádu $n = q_1q_2$ a bilineárne zobrazenie z $\mathbb{G}$ do $\mathbb{G}_1$. Pred popisom samotného kryptosystému ukážeme postup, ako skonštruovať tieto grupy a nájsť príslušné zobrazenie. Pripomenieme, že zobrazenie $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_1$ je bilineárne, ak platí: $\forall a, b, c \in \mathbb{G} : e(a\cdot b, c) = e(a, c)\cdot_1 e(b, c)$ a $e(a, b \cdot c) = e(a, b) \cdot_1 e(a, c)$. Operácia $\cdot_1$ označuje príslušnú operáciu v grupe $\mathbb{G}_1$.

Konštrukcia $\mathbb{G}$, $\mathbb{G}_1$ a e.
Vstupom nech je zložené číslo $n = q_1q_2$, kde $q_i > 3$.

1. Nájdeme najmenšie $l \in \mathbb{N}$ tak, aby $p = ln - 1$ a pre prvočíslo p platilo $p \equiv 2 \bmod 3$.

2. Uvažujme teraz grupu bodov (super-singulárnej) eliptickej krivky nad $\mathbb{F}_p$ danej rovnicou $y^2 = x^3 + 1$. Keďže $p \equiv 2 \bmod 3$, tak krivka má $p + 1 = ln$ bodov v $\mathbb{F}_p$. Z Lagrangeovej vety vyplýva, že existuje podgrupa bodov krivky, ktorá má n bodov - označíme ju $\mathbb{G}$.

3. Nech $\mathbb{G}_1$ označuje podgrupu $\mathbb{F}^*_{p^2}$ takú, že $\mathbb{G}_1$ má rád n. Weilova párovacia funkcia potom definuje požadované bilineárne zobrazenie $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$. Autori len odkazujú na ďalšiu literatúru a poznamenávajú, že táto párovacia funkcia sa dá efektívne vypočítať.

Popis kryptosystému: Popíšeme teraz jednotlivé algoritmy tohto asymetrického kryptosystému:

- *Generovanie kľúčov*: nech τ udáva požadovanú bezpečnosť. Vygenerujeme dve náhodné τ-bitové prvočísla q_1 a q_2. Nech $n = q_1q_2$. Vyššie popísaným postupom zostrojíme grupy $\mathbb{G}$, $\mathbb{G}_1$ rádu n a zobrazenie e. Nech g, u sú dva náhodné generátory grupy $\mathbb{G}$. Definujeme $h = u^{q_2}$, potom h bude generovať podgrupu grupy $\mathbb{G}$, ktorá bude mať rád q_1.
  $pk = (n, \mathbb{G}, \mathbb{G}_1, e, g, h)$. $sk = q_1$.
  $\mathcal{P}_\xi = \{0, 1, \ldots, T\}$, kde $T < q_2$. $\mathcal{C} = \mathbb{G} \cup \mathbb{G}_1$.

- *Šifrovanie*: Pre danú správu $m \in \mathbb{Z}_{T+1}$ zvolíme náhodne $r \in \{0, 1, \ldots, n - 1\}$ a vrátime zašifrovaný text c daný predpisom:

$$c = g^m h^r \in \mathbb{G}$$

[10]Množina otvorených textov nemôže byť veľmi veľká - vyplýva to z obmedzenia pri dešifrovaní.

- *Dešifrovanie*: využijeme rovnosť

$$c^{q_1} = (g^m h^r)^{q_1} = (g^{q_1})^m$$

Nech $g^* = g^{q_1}$. Na získanie m potrebujeme vypočítať diskrétny logaritmus c^{q_1} so základom g^*, ktorý spočítame pomocou Pollardovej λ-metódy. Vďaka podmienke $0 < m < T$ bude časová náročnosť dešifrovania rovná $O(\sqrt{T})$.

- *Výpočet* $A(pk, c_1 + c_2)$: zvolíme náhodne $r_3 \in \{0, 1, \ldots, n-1\}$ a vrátime $c_3 = c_1 c_2 h^{r_3} = g^{m_1+m_2} h^{r_1+r_2+r_3}$.

- *Výpočet* $A(pk, c_1 \times c_2)$: musí platiť $c_1, c_2 \in \mathbb{G}$ a výsledok bude z $\mathbb{G}_1$, takže "hĺbka" násobení musí byť maximálne 1. Na spočítanie výsledku zvolíme náhodne $r_3 \in \{0, 1, \ldots, n-1\}$ a vrátime $c_3 = e(c_1, c_2) h^{r_3}$

Overenie korektnosti: Korektnosť aditívnej vlastnosti vyplýva z vyššieuvedenej rovnosti. Na overenie korektnosti súčinu si najprv označíme dôležité prvky z $\mathbb{G}_1$: nech $g_1 = e(g, g)$ a $h_1 = e(g, h)$. Potom g_1 je rádu n a h_1 je rádu q_1. Nech ďalej $h = g^\alpha$ pre nejaké neznáme α. Rozpíšeme $c_3 = e(c_1, c_2) h^{r_3}$ využijúc bilineárnosť zobrazenia e:

$$\begin{aligned} c_3 = e(g^{m_1} h^{r_1}, g^{m_2} h^{r_2}) h^{r_3} &= e(g,g)^{m_1 m_2} e(g,h)^{m_1 r_2 + m_2 r_1} e(h,h)^{r_1 r_2} \\ &= g_1^{m_1 m_2} h_1^{m_1 r_2 + m_2 r_1} e(g^\alpha, h)^{r_1 r_2} \\ &= g_1^{m_1 m_2} h_1^{m_1 r_2 + m_2 r_1 + \alpha r_1 r_2} \\ &= g_1^{m_1 m_2} h_1^{r^*} \in \mathbb{G}_1 \end{aligned}$$

Pre korektnosť schémy ešte treba dodefinovať analogickú dešifrovaciu funkciu pre $c \in \mathbb{G}_1$. Keďže grupy $\mathbb{G}$ a $\mathbb{G}_1$ majú rovnakú štruktúru, tak dešifrovacia funkcia pomocou g_1 a q_1 je zrejmá. Kryptosystém je navyše aditívne homomorfný aj v grupe $\mathbb{G}_1$.

Poznámka 2.4.5. Keďže desifrovanie zaberie asymptoticky polynomiálny čas od veľkosti priestoru otvorených textov, tak kryptosystém sa dá použiť iba na šifrovanie krátkych správ. Autori poznamenávajú, že predpočítaním tabuľky mocnín g^* (a ich vhodným uložením) sa dá redukovať časová zložitosť na konštantnú – ale za cenu nárastu pamäťovej.

Záver
Tento kryptosystém má pomerne obmedzené možnosti aplikácie, ale na rozdiel od predošlých poskytuje jasnú bezpečnosť (ktorá je podrobnejšie analyzovaná v [2]) a vyhodnocovaním nestúpa dĺžka zašifrovaných textov. Je to prvý kryptosystém, ktorý dokázal skĺbiť sčitovanie s násobením a odolal doterajšej kryptoanalýze.

2.4.4 C.Melchor, P.Gaborit, J.Herranz - 2008

Autori v [17] predstavujú teoretickú konštrukciu, ktorá je zovšeobecnením kryptosystému od Boneha a kol. z časti 2.4.3. Ich cieľom je zostrojenie viacerých aditívnych

kryptosystémov, ktoré bude možné zreťaziť a dosiahnuť tým (podobne ako v predchádzajúcej časti) bezpečnú schému pre spočítanie niekoľkých násobení.

Ich konštrukcia je založená na nasledujúcej myšlienke. Predpokladajme, že pre *ľubovoľné* $s \in \mathbb{Z}^+$ vieme zostrojiť aditívno-homomorfný kryptosystém so šifrovacou funkciou $\varepsilon : \mathbb{Z}_s \to \mathbb{Z}_{s'}$. Potom si zostrojíme dva kryptosystémy také, že pre šifrovacie funkcie bude (okrem aditívneho homomorfizmu) platiť:

$$\varepsilon_1 : \mathbb{Z}_{s_1} \to \mathbb{Z}_{s_2}$$

$$\varepsilon_2 : \mathbb{Z}_{s_2} \to \mathbb{Z}_{s_3}$$

Ak označíme $\varepsilon = \varepsilon_2 \circ \varepsilon_1$, tak bude platiť:

$$\begin{aligned} \forall a_1, a_2 \in \mathbb{Z}_{s_1} : \varepsilon(a_1) + \varepsilon(a_2) &= \varepsilon_2(\varepsilon_1(a_1)) + \varepsilon_2(\varepsilon_1(a_2)) \\ &= \varepsilon_2(\varepsilon_1(a_1) + \varepsilon_1(a_2)) \\ &= \varepsilon_2(\varepsilon_1(a_1 + a_2)) \\ &= \varepsilon(a_1 + a_2) \end{aligned}$$

Navyše:

$$\begin{aligned} \forall a_1 \in \mathbb{Z}_{s_1}, a_2 \in \mathbb{Z}_{s_2} : \varepsilon_1(a_1) \cdot \varepsilon_2(a_2) &= \varepsilon_2(a_2 \cdot \varepsilon_1(a_1)) \\ &= \varepsilon_2(\varepsilon_1(a_1 \cdot a_2) = \varepsilon(a_1 \cdot a_2) \end{aligned}$$

Tieto vlastnosti sa dajú považovať za (nedokonalú) formu algebraického homomorfizmu. Presnejšie pojmy definujú autori v článku. Pokračujú taktiež nástrojmi na "upravovanie" definičných oborov a oborov hodnôt šifrovacích funkcií tak, aby sa tieto dali skladať do požadovanej reťaze. Taktiež ukazujú, že týmito úpravami sa zachováva úroveň bezpečnosti pôvodného kryptosystému. Pri skladaní šifrovacích funkcií potom platí, že ak uroveň bezpečnosti jednej z použitých šifrovacích funkcií bola napríklad IND-CPA, tak výsledná reťaz bude taktiež IND-CPA.

Okrem teoretickej konštrukcie, autori uvádzajú príklad, ako zostrojiť takúto zloženú schému. Vychádzajú z kryptosystému založeného na diskrétnej mriežke od Kawachi, Tanaka a Xagawa [15] a schémy od Boheh, Goh, Nissim. Ich konštrukcia bude schopná bezpečne vyhodnocovať boolovské výrazy v DNF, ktoré môžu mať hĺbku až 3 násobenia (operácie AND) a 10^9 operácií sčitovania (resp. OR). Množinu otvorených textov tvorí $\mathbb{Z}_{10^9+1}$, množinu zašifrovaných textov $(\mathbb{G}, \times)^{80000}$, kde $\mathbb{G}$ je grupa popísaná v 2.4.3 pre $n = 10^9 + 1$. Prvok grupy $\mathbb{G}$ je reprezentovaný pomocou 2048 bitov, zašifrovaný text bude mať teda dĺžku niekoľko megabajtov.

2.4.5 C.Gentry - 2009

V tejto časti popíšeme konštrukciu z [12], ktorá prezentuje asymetrický algebraický homomorfný kryptosystém. Táto konštrukcia bola podrobnejšie popísaná v práci [11]. Prínos tejto prelomovej práce sa dá zhrnúť do troch bodov:

- konštrukcia čiastočne homomorfnej schémy ξ_1

- zníženie stupňa dešifrovacieho obvodu ξ_1 pomocou tzv. "squashing" techniky (vznikne schéma ξ_2)
- použitie schémy ξ_2 na konštrukciu plne homomorfnej schémy

V ďalšom popíšeme stručne jednotlivé body počínajúc čiastočne homomorfným kryptosystémom ξ_1. Tento je založený na myšlienke, že šifrový text bude zložený z dvoch častí – nejakého bodu mriežky a chybového vektora, ktorý nejakým spôsobom kóduje otvorený text. Dešifrovanie spočíva v nájdení najbližšieho vektora mreže, čo je v prípade znalosti dobrej bázy ako súkromného kľúča efentívne riešiteľný problém. Tento chybový vektor bude oveľa menší ako je veľkosť najmenšieho vektora mreže, takže systém umožňuje niekoľko sčítaní a násobení vektorov, kým dĺžka chybového vektora presiahne danú hranicu. Tento kryptosystém je popísaný všeobecne, konkrétna implementácia bola ponechaná na ďalší výskum.

Poznámka 2.4.6. Tretí bod sa označuje aj ako "bootstrapping", ide o spôsob znižovania šumu v zašifrovanom texte tak, aby šum stále kódoval ten istý otvorený text. Gentry ukázal, ako to dosiahnuť za predpokladu, že schéma ξ dokáže homomorfne vyhodnotiť svoj vlastný dešifrovací obvod (rozšírený o jednu NAND-vrstvu).

Čiastočne homomorfný kryptosystém ξ_1
Parametre: schéma pracuje nad okruhom $R = \mathbb{Z}[x]/(f(x))$, pričom polynóm $f(x)$ je monický a má stupeň n, kde n je vstupný parameter. Používa pevne zvolený „malý" ideál I s bázou $\mathbf{B_I}$. Zvolí sa parameter $r_{Enc} \in \mathbb{R}$, ktorého význam vysvetlíme neskôr. Priestor otvorených textov je daný triedami rozkladu okruhu R podľa ideálu I. Všetky menované parametre sú verejné.

- $KeyGen_{\xi_1}$: zvolí ideál $J \subset \mathbb{R}$ tak, aby bol nesúdeliteľný s I (t.j. $I + J = \mathbf{R}$) a vráti súkromnú („dobrú") a verejnú („zlú") bázu (B_{sk}, B_{pk}) ideálu J.
- $E_{\xi_1}(B_{pk}, m)$: najprv $m' \in_R (m + I) \cap \mathcal{B}(r_{Enc})$. Vráti $c = m' \bmod B_{pk}$.
- $D_{\xi_1}(B_{sk,c})$: vráti $m = (c \bmod B_{sk}) \bmod B_I$.
- $Eval_{\xi_1}(B_{pk}, \circ, c_1, c_2)$: vráti $c = c_1 \circ c_2 \bmod B_{pk}$, pre $\circ \in \{+, \cdot\}$ z okruhu R.

Postup šifrovania je formálne založený na dvoch náhodných algoritmoch, prvý zobrazí správu m na náhodný prvok z tej istej triedy rozkladu podľa I. Potom sa vyberie náhodný vektor zo sféry $\mathcal{B}(\mathbf{0}, r_{Enc})$ a pripočíta sa ku m. Výsledný vektor sa nakoniec zobrazí do základného rovnobežnostena daného verejnou bázou B_{pk}. Parameter r_{Enc} teda udáva, nakoľko sa môžeme pri šifrovaní vzdialiť od bodu mriežky.

Pri dešifrovaní zobrazíme bod pomocou súkromnej bázy naspäť. V prípade, že sme sa od bodov mriežky nevzdialili viac ako r_{Dec}, tak jednoduchým zaokrúhlením súradníc dostaneme pôvodný bod mriežky. Maximálna hodnota r_{Dec} je teda daná súkromnou bázou B_{sk} ako polomer najväčšej sféry, ktorá leží celá v rovnobežnostene tvorenom

bázovými vektormi. Pri konkrétnej implementácii je zvolená menšia hodnota r_{Dec}, aby mal algoritmus dešifrovania ešte menšiu hĺbku.

Homomorfné vlastnosti schémy
Je intuitívne jasné, že pre $r_{Enc} = 0$ by sme mohli takto sčitovať a násobiť korektne. Keďže ale pri šifrovaní sme sa vzdialili od bodu mriežky o nejakú vzdialenosť, tak pri sčítaní dvoch vektorov sa táto „chybová" vzdialenosť sčíta tiež. Ak sa nám stane, že chyba bude väčšia ako r_{Dec}, tak nie sme schopní dešifrovať korektne. Množina prípustných obvodov je definovaná tak, aby výsledok obvodu nebol nikdy ďalej od mriežky ako r_{Dec}. Formálne

$$C \in C_\xi \Leftrightarrow \forall(c_1, \ldots, c_t) : C(c_1, \ldots, c_t) \in \mathcal{P}(B_{sk})$$

Pri násobení sa chyba zväčšuje viac. Pre zvolený okruh $\mathbf{R}$ ale existuje konštanta $\gamma_{Mult}(\mathbf{R})$, ktorej veľkosť je polynomiálna od n a platí: $||c_1 \times c_2|| \leq \gamma_{Mult}(\mathbf{R}) \cdot ||c_1|| \cdot ||c_2||$. Autor pre túto schému ukáže, že prípustná hĺbka obvodov, zložených z hradiel Add a $Mult$ je $\log\log r_{Dec} - \log\log(\gamma_{Mult}(\mathbf{R}) \cdot r_{Enc}) \approx k \cdot \log(n)$, kde $k < 1$. To ešte nezahŕňa dešifrovací obvod takejto mriežkovej schémy, takže v nasledujúcom načrtneme, ako znížiť hĺbku dešifrovacieho obvodu.

"Zjednodušenie" dešifrovania
Gentry ukázal, že dešifrovací predpis $m = (c \bmod B_{sk}) \bmod B_I$ sa dá zjednodušiť na $m = (\overrightarrow{c} - \lfloor \overrightarrow{c} \times \overrightarrow{v_{sk}} \rceil) \bmod B_I$, kde $\overrightarrow{v_{sk}}$ je nejaký vektor ideálu J^{-1}. Druhá modifikácia spočíva v znížení r_{Dec} na polovicu. To síce zníži hĺbku prípustných obvodov, ale umožní neskôr počítanie s oveľa menšou presnosťou, takže výrazne zníži hĺbku dešifrovacieho obvodu.

Obe popísané modifikácie nestačili na to, aby $D_\xi \in \mathcal{O}_\xi$, preto Gentry previedol podstatnú časť výpočtu $\overrightarrow{c} \times \overrightarrow{v_{sk}}$ do šifrovacieho predpisu. Dosiahol to tým, že do verejného kľúča pridal množinu γ_{set} vektorov $\overrightarrow{v_i}$, pre ktoré platí, že súčet nejakých γ_{sub} z nich je práve $\overrightarrow{v_{sk}}$. Parameter γ_{set} zvolí dostatočne veľký ($\approx \omega(n)$) a γ_{sub} primerane malý ($\approx \omega(\log(n))$) na to, aby zodpovedajúci SVSSP bol dostatočne ťažký. Súkromný kľúč potom obsahuje binárny vektor $A \in \{0,1\}^{\gamma_{set}}$ taký, že $A_i = 1$ práve vtedy, ak $\overrightarrow{v_i}$ patrí do malej podmnožiny veľkosti γ_{sub}.

Šifrovanie v takto zjednodušenej schéme ξ_2 začne rovnako ako v ξ_1 $\overrightarrow{c} = E_{\xi_1}(pk', m)$ a potom sa vypočítajú $\overrightarrow{c_i} = \overrightarrow{c} \cdot \overrightarrow{v_i}$. Šifrový text schémy ξ_2 je vektor $(\overrightarrow{c}, \overrightarrow{c_1}, \ldots, \overrightarrow{c_{\gamma_{set}}})$.

Evaluácia prebieha tiež podľa schémy ξ_1 a na konci sa z $\overrightarrow{c}$ predpočíta rovnako celý vektor $(\overrightarrow{c}, \overrightarrow{c_1}, \ldots, \overrightarrow{c_{\gamma_{set}}})$.

Dešifrovanie sa zjednodušilo na nasledovné tri kroky:

1. $\overrightarrow{x_i} = A_i \cdot \overrightarrow{c_i}$, pre každé $i \leq \gamma_{set}$
2. $\overrightarrow{c^+} = \sum_{1 \leq i \leq \gamma_{set}} \overrightarrow{x_i}$
3. $m = (\overrightarrow{c} - \lfloor \overrightarrow{c^+} \rceil) \bmod B_I$

Podrobný dôkaz, že takto upravený postup dešifrovania sa dá vyhodnotiť s obvodom logaritmickej hĺbky od n vynechávame.

"Bootstrapping"
Táto technika je založená na pozorovaní, že ak dešifrovací obvod D_ξ nejakej korektnej homomorfnej schémy ξ patrí do prípustnej množiny obvodov, tak sme schopní "meniť" verejný kľúč, ktorým je zašifrovaná ľubovoľná správa bez toho, aby sme ju museli najprv dešifrovať. Ak chceme zmeniť $E_\xi(pk_1, m)$ na $E_\xi(pk_2, m)$, potrebujeme na to iba vlastniť sk_1 zašifrovaný[11] pomocou pk_2 – $E_\xi(pk_2, sk_1)$ – a šifrový text $E_\xi(pk_1, m)$ zašifrovať druhýkrát pomocou kľúča pk_2. Nech $EE(m) = E_\xi(pk_2, E_\xi(pk_1, m))$ označuje dvakrát zašifrovanú správu m. Potom stačí vypočítať:

$$c = Eval_\xi(pk_2, D_\xi, E_\xi(pk_2, sk_1), EE(m))$$

Z definície korektnosti vyplýva, že po dešifrovaní c z predchádzajúcej rovnosti dostaneme to isté ako by sme dostali vyhodnotením obvodu D_ξ na tých istých vstupoch len nezašifrovaných pomocou pk_2. Teda $D_\xi(sk_2, c) = D_\xi(sk_1, E_\xi(pk_1, m)) = m$, z čoho vyplýva, že $c = E_\xi(pk_2, m)$.

Definícia 2.4.1. *Nech Γ je nejaká množina hradiel (napríklad iba NAND). Pre hradlo $g \in \Gamma$ definujeme g-rozšírenie dešifrovacieho obvodu ako hradlo g, ktorého každý vstup je spojený s výstupom dešifrovacieho obvodu D_ξ. Počet vstupov g udáva počet kópií D_ξ, ktoré sa zapoja pred hradlo g.*
Symbolom $D_\xi(\Gamma)$ budeme označovať množinu g-rozšírení D_ξ pre každé $g \in \Gamma$.

Popíšeme teraz konštrukciu stupňovito plne homomorfnej schémy ξ_3 pomocou schémy ξ_2, ktorá korektne vyhodnocuje svoj Γ-rozšírený dešifrovací obvod. Presnejšie požadujeme, aby $D_{\xi_2}(\Gamma) \in \mathcal{O}_{\xi_2}$. Schéma ξ_3 potom bude schopná vyhodnocovať obvody hĺbky d pozostávajúce z hradiel patriacich do Γ. Definujeme štyri požadované algoritmy schémy ξ_3:

- $KeyGen_{\xi_3}(\lambda, d)$: d je preddefinovaná hĺbka obvodov, λ bezpečnostný parameter. Algoritmus spočíta:

$$\begin{aligned} (sk_i, pk_i) &= KeyGen_{\xi_2}(\lambda) \quad \text{pre } i \in [0, d] \\ \overline{sk_i} &= E_{\xi_2}(pk_{i-1}, sk_i) \quad \text{pre } i \in [1, d] \end{aligned}$$

  a vráti $sk^{(d)} = sk_0$ ako súkromný kľúč a $pk^{(d)} = (\langle pk_i \rangle, \langle \overline{sk_i} \rangle)$ ako vektor verejných kľúčov.

- $E_{\xi_3}(pk^{(d)}, m)$: vráti jednoducho $c = E_{\xi_2}(pk_d, m)$.

- $D_{\xi_3}(sk^{(d)}, c)$: predpokladáme, že c bude už výsledkom $Eval_{\xi_3}$ a teda je šifrovaný pomocou pk_0. Výsledok dešifrovania je definovaný ako $D_{\xi_2}(sk_0, c)$.

- $Eval_{\xi_3}(pk^{(d)}, O_d, cc)$: predpokladá vstupný obvod O hĺbky presne d a vektor zašifrovaných textov $cc = (c_1, \ldots, c_t)$. Pre každú úroveň obvodu, t.j. pre každé hradlo g v obvode O sa (pomocou schémy ξ_2) vykoná jeho vyhodnotenie spolu

[11] Takže sa ani nepotrebujeme dozvedieť súkromný kľúč sk_1.

s prešifrovaním na nasledujúci verejný kľúč. Využívame pri tom, že g-rozšírenie dešifrovacieho obvodu patrí do prípustnej množiny obvodov schémy ξ_2. Vyhodnotenie δ (pre $\delta = d, \ldots, 1$) úrovne obvodu O je teda definované v dvoch krokoch:

$$\begin{aligned}(O^{\dagger}_{\delta-1}, cc^{\dagger}_{\delta-1}) &= Augment_{\xi_3}(pk^{(\delta)}, O_\delta, cc_\delta)\\ (O_\delta, cc_{\delta-1}) &= Reduce_{\xi_3}(pk^{(\delta-1)}, O^{\dagger}_{\delta-1}, cc^{\dagger}_{\delta-1})\end{aligned}$$

- $Augment_{\xi_3}(pk^{(\delta)}, O_\delta, cc_\delta)$: každé hradlo na úrovni δ obvodu O_δ rozšíri pomocou obvodu D_{ξ_2} a výsledný obvod označí $O^{\dagger}_{\delta-1}$. Ďalej každý zašifrovaný text c_i vstupujúci do rozšíreného hradla zamení za dvojicu $\langle\langle\overline{sk_\delta}\rangle, \langle\overline{c_i}\rangle\rangle$, kde $\overline{c_i} = E_{\xi_2}(pk_{\delta-1}, c_i)$. Výsledný vektor zašifrovaných textov označíme $cc^{\dagger}_{\delta-1}$.

- $Reduce_{\xi_3}(pk^{(\delta)}, O^{\dagger}_{\delta}, cc^{\dagger}_{\delta})$: vstupný obvod $O^{\dagger}_{\delta}$ je výstupom $Augment_{\xi_3}(\delta+1)$, takže pozostáva z $\delta + 2$ úrovní, kde prvá úroveň pozostáva z D_{ξ_2} a druhá obsahuje pôvodné hradlá g. Označme tieto dve úrovne obvodu $O^{\dagger}_{\delta}$ ako $O^{\ddagger}$ a zvyšok obvodu ako O^{δ}. Obvod $O^{\ddagger}$ tvoria g-rozšírenia dešifrovacieho obvodu schémy ξ_2 a patria do prípustnej množiny obvodov schémy ξ_2 – taký bol predpoklad bootstrappingu. Vieme teda zavolať algoritmus $Eval_{\xi_2}(pk_\delta, O^{\ddagger}, cc^{\dagger}_{\delta})$, ktorý ho vyhodnotí a označiť výsledné šifrové texty ako cc_δ. Výsledkom bude teda zvyšných δ úrovní pôvodného obvodu a korektne vyhodnotené šifrové texty, ktoré sú zašifrované kľúčom pk_δ.

Takto dostaneme zašifrovaný text pomocou súkromného kľúča sk_0 a teda algoritmus $D_{\xi_3}()$ je definovaný korektne. Efektívnosť evaluácie obvodov je pomerne nízka, ale poskytuje veľmi všeobecný nástroj. Efektívnosť sa zvyšuje, ak zložitejšie obvody (nie len jednoduché hradlá) patria do Γ – tým sa umožní vyhodnotenie väčšej časti obvodu na jedno "prešifrovanie".

Ak predpokladáme, že zverejnenie súkromného kľúča zašifrovaného verejným ($\langle sk_i\rangle_{pk_i}$) nijako neovplyvní bezpečnosť schémy (tzv. KDM-bezpečnosť schémy ξ_2), tak nemusíme generovať d-inštancií schémy ξ_2, ale stačí jedna a dostávame plne homomorfnú schému schopnú vyhodnocovať obvody ľubovoľnej hĺbky.

2.4.6 Dijk, Gentry, Halevi, Vaikuntanathan - 2010

Jednoduchý príklad plne homomorfnej schémy ξ_1, ktorej bezpečnosť je založená na aproximačnom probléme najväčšieho spoločného deliteľa (z časti 2.3). Definujeme najprv čiastočne homomorfnú schému.

Popis kryptosystému:
Vstupné parametre:

- λ – bezpečnostný parameter
- γ – bitová dĺžka čísel vo verejnom kľúči

- $t + 1$ – počet čísel vo verejnom kľúči
- η – bitová dĺžka súkromného kľúča
- ρ – bitová dĺžka šumu (pri šifrovaní)

Tieto parametre sú verejné, odporúčaná voľba je $\rho = \lambda$, $\eta = O(\lambda^2)$, $\gamma = O(\lambda^5)$ a $t = \gamma + \lambda$.

- $KeyGen(\lambda)$:

  1. zvolí sa nepárne číslo $p \in_R (2\mathbb{Z} + 1) \cap [2^{\eta-1}, 2^\eta)$.
  2. pre $i = 0, \ldots, t$ sa generuje $q_i \in_R [0, 2^\gamma/p)$ a $r_i \in_R (-2^\rho, 2^\rho)$ a vypočíta $x_i = q_i \cdot p + r_i$
  3. jednotlivé x_i sa preznačia tak, aby x_0 bolo najväčšie

  Tento postup sa opakuje dovtedy, kým x_0 je nepárne a zároveň $[x_0]_p$ je párne. Výstupom je $pk = (x_0, \ldots, x_t)$ a $sk = p$. Okruh otvorených textov je $\mathcal{P} = \mathbb{Z}_2$ a okruh zašifrovaných textov je $\mathbb{Z}_{x_0}$.

- $E(pk, m)$: vyberie sa náhodne podmnožina indexov $S \subseteq \{1, \ldots, t\}$ a $r \in_R (-2^\rho, 2^\rho)$. Výstupom je $c = [m + 2r + \sum_{i \in S} x_i]_{x_0}$.
- $D(sk, c)$: výstup je $m = [[c]_p]_2$.
- *Výpočet* $Eval(pk, \circ, c_1, c_2)$: $c_3 = c_1 \circ c_2$, kde $\circ \in \{+, \cdot\}$ nad celými číslami.

Poznámka 2.4.7. Všimnime si, že veľkosť šifrových textov v takto definovanej schéme rastie pri každej evaluácii. Autori navrhujú dva spôsoby ako tento problém riešiť. Prvým z nich je pridanie postupnosti čísel x'_i do verejného kľúča, kde $x'_i = 2(q'_i + r'_i)$ a $q'_i \in_R [2^{\gamma+i-1}/p, 2^{\gamma+i}/p)$ a $r'_i \in_R (-2^\rho, 2^\rho)$. Počas homomorfnej evaluácie, vždy keď veľkosť šifrového textu presiahne 2^γ, tak sa zredukuje postupne x'_γ, $x'_{\gamma-1}$, až x'_0.

Druhý spôsob riešenia problému je v algoritme $KeyGen$ priamo definovať x_0 ako presný násobok p. Aj keď zodpovedajúci AGCD problém v tomto prípade vyzerá jednoduchšie, jeho obtiažnosť je podľa [5] ekvivalentná ku všeobecnému AGCD problému (popísanému v časti 2.3).

Homomorfné vlastnosti

Priamo po šifrovaní je veľkosť šumu maximálne $2^{2\rho+2}$. Dešifrovanie je korektné, pokiaľ šum nepresiahol hodnotu $p/2$, ale kvôli jednoduchšiemu dešifrovaciemu predpisu autori požadujú hranicu $p/8$. Presnejšie ukážu, že schéma dokáže vyhodnotiť polynóm $f(x_1, \ldots, x_t)$ v t premenných, ktorý je stupňa d, pokiaľ platí

$$d \leq \frac{\eta - 4 - \log_2 |f|_1}{2\rho + 2},$$

kde $|f|_1$ je súčet absolútnych hodnôt koeficientov polynómu f.

Plne homomorfná schéma
Keďže autori nenašli spôsob ako vyjadriť dešifrovaciu funkciu polynómom prípustného stupňa, použili Gentryho techniku "squashing"-u na zjednodušenie dešifrovania za cenu nárastu veľkosti verejného kľúča ako aj postupu šifrovania. Verejný kľúč horeuvedenej čiastočne homomorfnej schémy ξ_1 bude rozšírený o niekoľko čísel, v ktorých bude existovať malá podmnožina, ktorej súčet bude tajný kľúč p. Schéma ξ_2 vyzerá nasledovne:

Popis kryptosystému ξ_2:
Vstupné parametre:

- γ_{set} – počet pridaných čísel do verejného kľúča
- γ_{sub} – veľkosť malej podmnožiny, s predpísaným súčtom
- κ – bitová presnosť pridaných čísel

Tieto parametre sú tiež verejné, autori volia $\kappa = \gamma\eta/(2\rho)$, $\gamma_{set} = \omega(\kappa \cdot \log(\lambda))$ a $\gamma_{sub} = \lambda$.

- $KeyGen_{\xi_2}$:

  1. vygeneruje sa $pk*, sk^*) \leftarrow Keygen_{\xi_1}(\lambda)$.
  2. vypočíta sa $x_p = \lfloor 2^\kappa/p \rceil$ a zvolí sa náhodný binárny vektor $\overrightarrow{s} = \langle s_1, \ldots, s_{\gamma_{set}} \rangle$ s hammingovou váhou γ_{sub}. Označíme $S = \{i | s_i = 1\}$.
  3. zvolia sa náhodné $u_i \mathbb{Z} \cap [0, 2^{\kappa+1})$ s podmienkou, že $\sum_{i \in S} u_i = x_p \bmod 2^{\kappa+1}$.
  4. spočítajú sa $y_i = u_i/2^\kappa$, pre $\overrightarrow{y}$ platí $[\sum_{i \in S} y_i]_2 = 1/p + \triangle_p$, kde $| \triangle_p | < 2^{-\kappa}$.
  5. súkromný kľúč je $sk = \overrightarrow{s}$ a verejný $pk = (pk^*, \overrightarrow{y})$.

- $E_{\xi_2}(pk, m)$ a $Eval_{\xi_2}(pk, \circ, c_1, c_2)$: c^* sa vypočíta zo schémy ξ_1 a potom sa spočíta $z_i = [c^*]_2$, kde sa použije bitová presnosť $n = \lceil \log(\gamma_{sub}) \rceil + 3$ bitov. Výstupom je c^* aj $\overrightarrow{z} = \langle z_1, \ldots, z_{\gamma_{set}} \rangle$.
- $D_{\xi_2}(sk, c)$: výstupom je $m = [c^* - \lfloor \sum_i z_i \cdot s_i \rceil]_2$.

Autori dokážu, že táto dešifrovacia funkcia sa dá naozaj vyjadriť ako dostatočne "plytký" polynóm od vstupných bitov a že schéma naozaj spĺňa definíciu bootstrappingu. Vďaka tomu sa dá použiť na záverečný krok zostrojenia plne homomorfnej schémy ξ_3, postupom popísaným v 2.4.5.

2.4.7 Smart, Vercauteren - 2010

Prvý kryptosystém, ktorý do detailu doplnil Gentryho čiastočne homomorfnú schému z [12]. Autorom sa ešte nepodarilo zjednodušiť schému natoľko, aby mohli uplatiniť bootstrapping a skonštruovať aj plne homomorfný kryptosystém. Popíšeme však aspoň čiatočne homomorfnú verziu.

Popis kryptosystému:
Vstupné parametre:

- N – stupeň polynómu vytvárajúceho okruh
- η – veľkosť koeficientov generovaného polynómu
- μ – veľkosť koeficientov šumového polynómu

Odporúčaná voľba je N mocnina dvojky, $\eta \approx 2^{\sqrt{n}}$ a $\mu \approx \sqrt{n}$.

- $KeyGen(n)$:

  1. $f(x) = x^N + 1$, pre prípad, že N je mocnina dvojky. Inak za $f(x)$ zvolia monický ireducibilný polynóm nad $\mathbb{Z}[x]$.
  2. $s(x)$ sa zvolí ako náhodný polynóm stupňa $N-1$ nad $\mathbb{Z}[x]$, s koeficientami do $\eta/2$. Následne $g(x) = 2 \cdot s(x) + 1$ a $p = rezulatant(g(x), f(x))$. Tento krok sa opakuje, kým p nie je prvočíslo.
  3. $d(x) = NSD(d(x), f(x))$ nad $\mathbb{F}_p[x]$.
  4. do α sa priradí jediný koreň polynómu $d(x)$.
  5. vypočíta sa $z(x)$ ako inverzný polynóm ku $g(x)$ pomocou rozšíreného NSD algoritmu nad $\mathbb{Q}[x]$, takže

     $$z(x) \cdot g(x) = p \bmod f(x)$$

  6. $b = z_0 \bmod 2p$
  7. $pk = (p, \alpha)$ a $sk = (p, b)$.
  8. $\mathcal{P} = \{0, 1\}$ a $\mathcal{C} = \mathbb{Z}_p$.

- $E(pk, m)$: zvolí sa $r(x)$ ako náhodný polynóm stupňa $N-1$ s koeficientami do $\mu/2$ a potom $c(x) = m + 2 \cdot r(x)$. Výsledok je $c = c(\alpha) \bmod p$.
- $D(sk, c)$: $m = (c - \lfloor c \cdot B/p \rceil) \bmod 2$
- *Výpočet* $Eval(pk, \circ, c_1, c_2)$: $c_3 = c_1 \circ c_2 \bmod p$, kde $\circ \in \{+, \cdot\}$.

Záver
Autorom sa nepodarilo dokončiť zjednodušenie dešifrovania natoľko, aby schéma splnila podmienku na bootstrapping, takže ostáva čiastočne homomorfná schéma schopná vyhodnocovať multiplikatívne obvody hĺbky $\log\log(\frac{\sqrt{N} \cdot \eta}{2 \cdot N} - \log\log(N \cdot \mu))$. Pre autormi implementovanú voľbu parametrov N a μ je to medzi 0.3 (pre $N = 256$) a 2.5 (pre $N = 2^{13}$).

2.4.8 Gentry, Halevi - 2010

Prvá fungujúca implementácia plne homomorfnej šifrovacej schémy. Čiastočne homomorfná časť je veľmi podobná predchádzajúcej konštrukcii, rozdiel je v jednoduchšom generovaní kľúčov a zjednodušenej reprezentácii mriežok a vektorov.

Popis kryptosystému: Vstupné parametre:

- N – mocnina čísla dva, stupeň polynómu vytvárajúceho okruh
- t – bitová dĺžka koeficientov polynómu v $KeyGen()$
- q – reálne číslo $q \in (0, 1)$ udáva veľkosť šumu pri šifrovaní
- $f(x) = x^n + 1$

Tieto parametre sú verejné, odporúčaná voľba je $N = \{2048, 8192, 32768\}$, pre "malú", "strednú" a "vysokú" bezpečnosť. Spoločné odporúčanie je $t = 390$ a $q = 1 - 20/N$.

- $KeyGen(\lambda)$:
  1. zvolí sa náhodný polynóm $v(x)$ stupňa $N - 1$, veľkosť koeficientov do 2^t.
  2. $d = rezultant(v(x), f(x))$
  3. spočíta sa $w(x)$ tak, že $w(x) \cdot v(x) = d \bmod f(x)$.
  4. overí sa, či $w(x)$ je vhodného tvaru a d nepárne. V prípade potreby sa postup opakuje od kroku 1.
  5. označíme V rotačnú bázu polynómu $v(x)$.
  6. vhodný tvar polynómu $w(x)$ znamená, že HNF(V) závisí len od dvoch čísel d a r, preto $pk = (d, r)$.
  7. $sk = w_{i_0}$, kde i_0 je index koeficientu $w(x)$, ktorý je nepárny.
  8. $\mathcal{P} = \{0, 1\}$ a $\mathcal{C} = \mathbb{Z}_d$.
- $E(pk, m, q)$:
  1. zvolí sa $u(x)$ ako náhodný polynóm stupňa $N - 1$ s koeficientami z množiny $\{-1, 0, 1\}$, kde $P[u_i = 0] = q$ a $P[u_i = 1] = P[u_i = -1] = 1 - q/2$.
  2. $\overrightarrow{a} = m \cdot \overrightarrow{e_1} + 2 \cdot \overrightarrow{u}$
  3. $\overrightarrow{c} = \overrightarrow{a} \bmod \text{HNF}(V)$
  4. vďaka pšeciálnemu tvaru HNF(V) sa dá ukázať, že $\overrightarrow{c} = \langle c, 0, \ldots, 0 \rangle$ a navyše $c = [a(r)]_d$.
- $D(sk, c)$: $\overrightarrow{a} = \overrightarrow{c} \bmod V$, $(\overrightarrow{c} - \overrightarrow{a}) \bmod 2 = m$, ale opäť sa dá ukázať, že $m = [c \cdot w_{i_0}]_d] \bmod 2$.

m-♯-premenných t-bitová-dĺžka	$m = 64$	$m = 96$	$m = 128$	$m = 192$	$m = 256$
$t = 64$	13	12	11	11	10
$t = 128$	33	28	27	26	24
$t = 256$	64	76	66	58	56
$t = 384$	64	96	128	100	95

Table 2.1: [13] Najvyšší podporovaný stupeň v závislosti od počtu premenných m a bitovej dĺžky t.

Parameter	Význam
$S = 1024, 1024, 4096$	mohutnosť velkých množín
$s = 15$	veľkosť riedkej podmnožiny (= počet veľkých množín)
$R = 2^{51}, 2^{204}, 2^{850}$	pomer medzi prvkami veľkej podmnožiny
$p = 4$	bitová presnosť (= počet bitov čísel z_i)
$c = \lceil 2\sqrt{S} \rceil$	parameter pre znižovanie veľkosti verejného kľúča

Table 2.2: [13] Parametre plne homomorfnej schémy. Rôzne hodnoty zodpovedajú rôznym navrhnutým úrovniam bezpečnosti.

- *Výpočet* $Eval(pk, \circ, c_1, c_2)$: $c_3 = c_1 \circ c_2 \bmod \mathrm{HNF}(V)$, kde $\circ$ je násobenie alebo sčítanie v okruhu $\mathbb{Z}[x]/(f(x))$. Vďaka špeciálnemu tvaru $\mathrm{HNF}(V)$ vieme dokázať, že $c_3 = [c_1 \diamond c_2]_d$, kde $\diamond$ je obyčajné celočíselné sčítanie resp. násobenie.

Dôkaz, že šifrovanie, dešifrovanie a homomorfné operácie sa dajú zjednodušiť na popísaný tvar, sa nachádza v [13].

Homomorfné vlastnosti
Autori nevyjadrili explicitne počet násobení, ktoré schéma dokáže vyhodnotiť korektne. Homomorfné vlastnosti popísali najvyšším stupňom elementárneho symetrického polynómu o m premenných, ktorý daná inštancia dokáže vyhodnotiť korektne. Popísanou voľbou šifrovacieho parametra $q = 1 - 20/N$ dosiahli počiatočnú veľkosť šumu v šifrovom texte vždy okolo $2 \cdot \sqrt{20} =\approx 9$ a teda homomorfné vlastnosti schémy nezávisia od dimenzie N. Najvyšší podporovaný stupeň experimentálne zistili pre $m = 64, 96, 128, 192, 256$ a výsledok sumarizovali v tabuľke 2.1.

Plne homomorfná schéma
Konštrukciu plne homomorfnej schémy nebudeme detailne popisovať, keďže naša práca je zameraná na čiastočne homomorfnú schému. Spomenieme len, že plne homomorfná schéma je založená piatich dodatočných parametroch s, S, R, p, c. Ich význam a odporúčané hodnoty pre "malú", "strednú" a "vysokú" bezpečnosť sú uvedené v tabuľke 2.2.

Zjednodušene povedané, verejný kľúč plne homomorfnej schémy je rozšírený o $S \cdot s$ šifrových textov. Tieto šifrové texty sú organizované do s množín $\mathfrak{B}_1, \ldots, \mathfrak{B}_s$ po S

textoch tak, že existuje množina s textov, z každej $\mathfrak{B}_i$ práve jeden, ktorých súčet je práve w_{i_0} modulo d.

Vhodnou reprezentáciou týchto množín autori výrazne zredukovali veľkosť verejného kľúča. Navyše znížili akceptovateľnú veľkosť šumu v schéme na $1/(s+1)$ pôvodnej veľkosti, čo sa prejaví tým, že zodpovedajúca čiastočne homomorfná schéma dokáže prinajhoršom vyhodnotiť o jedno násobenie menej. Vďaka tomu dokázali ohraničiť počet násobení počas dešifrovania výrazom $s \cdot 2^{p-1} + \sum_{k=1}^{p-1}(s+k) \cdot 2^{p-k} = O(s^2)$.

Kryptoanalýza

V [**?**] bol popísaný jednoduchý CCA1-útok, ktorý možno uplatniť na implementáciu tejto čiastočne homomorfnej schémy. Útok je založený na predpoklade, že útočník má prístup ku dešifrovaciemu orákulu, ktoré pre vstupný šifrový text $c \in \mathbb{Z}$ vždy vráti výstup $[c \cdot w_{i_0}]_d \bmod 2$ a analogický útok je možné uplatniť aj na čiastočne homomorfnú schému od Smart-Vercauteren (popísanú v časti 2.4.7). Cieľom útoku je určiť súkromný kľúč w_{i_0}. Nasleduje algoritmus útoku:

$L := 0$; $U := d - 1$
while $U - L > 1$ **do**

1. $c := \lfloor d/(U-L) \rfloor$
2. $b := \mathfrak{O}_{decr}(c)$
3. $q := (b + c) \bmod 2$
4. $k := \lfloor Lc/d + 1/2 \rfloor$
5. $M := (k + 1/2)d/c$
6. **if** $(k \bmod 2 = q)$ **then**
7. $\quad U := \lfloor M \rfloor$
8. **else**
9. $\quad L := \lceil M \rceil$

return L

Myšlienka útoku je nasledovná. V každej iterácii útoku platia tieto fakty:

i Všetky možné hodnoty tajného parametra w_{i_0} ležia v intervale $[L, U]$.

ii Ak by $w_{i_0} = L$ ležalo na kraji intervalu, tak dešifrovacia funkcia redukuje k-krát číslom d a platí $-d/2 \leq cL - kd < d/2$. Z toho sa dá vyjadriť $k = \lfloor c \cdot L/d + 1/2 \rfloor$.

iii Všetky možné hodnoty výrazu $c \cdot w_{i_0}$ ležia v intervale dĺžky d. Vyplýva to z $c \cdot U - c \cdot L \leq d$.

iv Z (iii) vyplýva, že existuje práve jedno (racionálne) číslo B v intervale $[L, U]$ tak, že $B = d/2 + i \cdot d$, kde i je celé. Konkrétne $B = (k + 1/2) \cdot d/c$.

v Pri dešifrovaní kľúčom w_{i_0} sa redukuje číslom d presne q-krát, kde $q = k$ alebo $q = k + 1$. Vyplýva to z (i) a (ii).

vi Keďže platí $b = (c \cdot w_{i_0} - qd) \bmod 2 \equiv (c - q) \bmod 2 \equiv (c + q) \bmod 2$, tak vieme vyjadriť paritu čísla q. Keďže poznáme k, tak vieme porovnať paritu q a k a rozhodnúť sa, ktorý z prípadov $q = k$ a $q = k + 1$ nastal pri dešifrovaní textu c. Takže vieme zúžiť interval na $[L, \lfloor B \rfloor]$, alebo $[\lceil B \rceil, U]$.

Autori tvrdia, že umiestnenie skutočného kľúča w_{i_0} je v každom intervale $[L_j, U_j]$ náhodné a že voľba menšieho podintervalu je tiež náhodná. Dĺžka intervalu sa teda každou iteráciou zredukuje priemerne na polovicu, takže útok bude vyžadovať približne $\log(d)$ iterácií. Píšu, že útok je v praxi vysoko efektívny a odhalí tajný kľúč pre všetky odporúčané veľkosti parametrov z [13] rádovo v sekundách. Autori ďalej v[**?**]ukázali, ako upraviť ich schému 2.4.7 tak, aby bol tento útok neúspešný. Či a ako sa dá upraviť Gentry-Halevi implementácia ostáva otvorená otázka.

V diplomovej práci [26] je popísaný útok na čiastočne homomorfnú schému, ktorého cieľom je z verejných parametrov (d, r) určiť súkromný kľúč. Postup útoku je taký, že z d, r sa skonštruuje matica $\mathrm{HNF}(V)$, ktorej vektory sú skoro rovnobežné a táto matica sa následne zredukuje nejakým redukčným algoritmom. V [26] boli popísané tri varianty útoku, podľa použitej redukčnej stratégie, pričom najuspešnejší bol základný LLL-algoritmus. Redukcia bola považovaná za úspešnú, ak sa pomocou výslednej matice dalo korektne dešifrovať, teda bola odhalená matica V. Útok bol realizovaný na viacero kombinácií parametrov čiastočne homomorfnej schémy, ale úspech bol dosiahnutý len po dimenziu 128, teda menej ako "nízka" odporúčaná bezpečnosť.

2.4.9 Najnovšie kryptosystémy, založené na LWE

TODO: ešte vygooglit a zhrnut, uz su aspon tri clanky:

1. Zvika Brakerski and Craig Gentry and Vinod Vaikuntanathan: Fully Homomorphic Encryption without Bootstrapping

2. Zvika Brakerski and Vinod Vaikuntanathan: Efficient Fully Homomorphic Encryption from (Standard) LWE

3. Zvika Brakerski and Vinod Vaikuntanathan: Fully homomorphic encryption from ring-lwe and security for key dependent messages

4. Vadim Lyubashevsky and Chris Peikert and Oded Regev: On Ideal Lattices and Learning with Errors Over Rings – základný článok

5. Junfeng Fan and Frederik Vercauteren: Somewhat Practical Fully Homomorphic Encryption

Chapter 3

Implementácia Gentryho schémy

Keďže jedným z cieľov práce je praktická analýza schémy navrhnutej v [12], tak prvým krokom je nutne implementácia tejto schémy. V článku [13], ktorého prvá verzia bola publikovaná v októbri 2010, sa uvádza pomerne detailný opis implementácie a boli v ňom testované aj homomorfné vlastnosti schémy, zdrojové kódy jej programu však neboli o tom čase zverejnené[1]. Ako sme písali v časti 2.4.5, plne homomorfná schéma je postavená na čiastočne homomorfnej využívajúc metódu tzv. bootstrappingu, takže popíšeme najprv implementáciu čiastočne homomorfnej schémy (SHS).

3.1 Požiadavky na aplikáciu SHS

Keďže cieľom implementácie nie je aplikácia pre širokú verejnosť, tak celkové požiadavky boli nasmerované na testovanie tejto schémy, dôraz sa kládol na jednoduchosť a prispôsobivosť celého programu, aby sa dal po častiach otestovať a ukázať jeho korektnosť.

Okrem pôvodnej schémy uvedenej v [13], ktorá uvažovala iba jednu voľbu ideálu $I = (2)$, sme doplnili požiadavku na všeobecnú voľbu ideálu $I = (p)$, kde p bude zadané prvočíslo. Táto voľba zároveň znamená veľkosť priestoru otvorených textov, takže v schéme ju označujeme ako `PT_BOUND`.

Z popisu kryptosystému je zrejmé, že aplikácia bude pracovať s veľkými číslami, z toho vyplýva nutnosť použitia nejakej knižnice pre vedecké výpočty. Pre rozšírenosť a jednoduchosť použitia sme zvolili jazyk C++ a knižnicu NTL [27], konkrétne verziu 5.5.2.

3.2 Analýza a návrh

Výsledný kryptosystém musí pozostávať z piatich algoritmov a to generovanie kľúčov, šifrovanie, dešifrovanie a homomorfné sčítanie a násobenie. Vstupné parametre kryptosystému sú `PT_BOUND` – prvočíselná veľkosť I, `N` – dimenzia mriežky a `t` – bitová

[1] O existencii zdrojových kódov sme sa dozvedeli v júli 2011, najneskôr mesiac od poslednej kontroly ich existencie. Naše kódy boli o tom čase už hotové.

veľkosť koeficientov použitého polynómu $v(x)$. Kryptosystém je daný dvojicou kľúčov `(PK,SK)`, kde verejný kľúč je vlastne štvorica celých čísel `(PT_BOUND,N,t,d,r)` a súkromný kľúč je celé číslo `w`, ktoré predstavuje jeden nepárny koeficient polynómu $w(x)$.

Jednotlivé algoritmy kryptosystému majú teda nasledovné vstupno-výstupné rozhranie:

- generovanie kľúčov: vstupom sú parametre `PT_BOUND`, `N`, `t`, výstupom bude trojica (d, r, w), kde prvé tri čísla sú časťou verejného kľúča, posledné je súkromný kľúč.
- šifrovanie: vstupom je verejný kľúč a otvorený text, výstupom je zašifrovaný text.
- dešifrovanie: vstupom je verejný, súkromný kľúč, zašifrovaný text a výstupom je zodpovedajúci otvorený text.
- homomorfné sčítanie: vstupom je verejný kľúč a dvojica zašifrovaných textov, výstupom je zašifrovaný text.
- homomorfné násobenie: rovnako ako sčítanie

Pre jednoduchosť budú zodpovedajúce funkcie v našom programe mať všetky parametre vstupno-výstupné, funkcie budú vracať ako návratovú hodnotu chybové hlásenie, prípadne nebudú vracať nič (void).

3.3 Zdrojové kódy

Z dôvodu realizácie praktických experimentov na nami implementovanej schéme považujeme za nutné zverejniť zdrojové kódy. Nachádzajú sa v prílohe A.

3.4 Testovanie

Otestovanie implementovanej schémy nebolo jednoduché, keďže nie sú známe žiadne vzorové PT-CT páry, ani svojice verejných a súkromných kľúčov. Testovali sme teda tieto vlastnosti schémy:

- korektnosť — $\forall p \in P : Dec(Enc(p, PK), SK) = p$
- podporovaný počet násobení — aký je najväčší počet ciphertextov, ktoré po vynásobení a následnom dešifrovaní vrátia správny výsledok?
- najväčší podporovaný stupeň — aký je najväčší stupeň symetrického polynómu nad c_i, ktorý po vyhodnotení a dešifrovaní vráti správny výsledok?

Posledné dve z uvedených vlastností boli testované aj autormi kryptosystému a v [13] boli zverejnené očakávané hodnoty, ktoré schéma priemerne dosahuje. Pri nižšie popísaných experimentoch 4 sme dosiahli zhruba rovnaké hodnoty[2], čo považujeme za

[2]Keďže sa jedná o pravdepodobnostný kryptosystém, tak drobné odchýlky boli očakávané.

indikáciu správnosti.

Prvá vlastnosť – korektnosť – je samozrejme hlavnou požiadavkou na každý kryptosystém a štatisticky je jednoducho overiteľná. Zo všetkých testovaných OT-ZT párov (presnejšie 10^3 pre každú testovanú kombináciu parametrov) sme zaznamenali korektné dešifrovanie, takže túto vlastnosť považujeme za štatisticky overenú. Testované parametre boli

- `N` $\in \{2^6, 2^7, \ldots, 2^1 1\}$, `t` $= \{N/2, N\}$ pre `PT_BOUND`$= 2$
- `N` $= 256$, `t` $= 128$ pre `PT_BOUND`$= 3, 5, 7, 11$.

3.5 Originálne zdrojové kódy FHS

Zdrojové kódy, ktoré zverejnili C.Gentry a S.Halevi na svojej webstránke majú viacero významných rozdielov oproti našim. Sú komplexnejšie, obsahujú aj funkcie pre plne homomorfnú schému, sú optimalizované na rýchlosť – silne využívajú obmedzenie $I = (2)$, teda jednobitový priestor kľúčov. Boli ale zverejnené neskôr a sú zložitejšie na použitie ako naše, takže v našich experimentoch sme využili našu implementáciu čiastočne homomorfnej schémy.

Chapter 4

Experimentálna časť

V tejto kapitole popíšeme výsledky experimentov, ktoré ukazujú homomorfné vlastnosti nami implementovaného kryptosystému. V druhej časti tejto kapitoly ukazujeme ako by kryptosystém vyzeral pre iné voľby priestoru otvorených textov a porovnáme takto zmenenú schému s pôvodnou. Na základe výsledkov navrhujeme jednoduchý spôsob, ako by bolo možné pomocou čínskej zvyškovej vety vytvoriť čiastočne homomorfnú schému pre ľubovoľne veľký priestor otvorených textov.

4.1 Homomorfné vlastnosti schémy pre $I = (2)$

V nasledujúcej časti sa venujeme originálnej verzii schémy s voľbou ideálu $I = (2)$. Popíšeme najprv sériu experimentov, ktoré ukazujú nakoľko je implementovaná schéma homomorfná, t.j. koľko operácií dokáže homomorfne vyhodnotiť bez toho, aby nastala chyba pri dešifrovaní. Keďže navrhovaný kryptosystém je čiastočnou verziou algebraicky homomorfného kryptosytému, tak tieto operácie zahŕňajú sčítanie zašifrovaných textov, ich násobenie a aj rôzne kombinácie týchto dvoch operácií, čiže ľubovoľné polynomiálne funkcie.

Nasledujú postupy, ktorými sme prakticky zisťovali, koľko operácií dokáže kryptosystém korektne vyhodnotiť. Jeden z týchto experimentov (z časti 4.1.3) bol realizovaný už v [13], kde sú aj výsledky, takže tieto výsledky budú určitým overením správnosti našej implementácie. Tento experiment sme rozšírili väčším množstvom parametrov a sledovali sme aj závislosť od ďalšieho parametra q. Výsledky z tejto časti boli publikované v [**?**].

4.1.1 Aditívnosť

Metodika

Na overenie, koľko zašifrovaných textov dokáže kryptosystém vyhodnotiť korektne, sme zvolili nasledovný postup. Pre dané parametre sme vygenerovali konkrétnu inštanciu krypstosystému (dvojicu verejný a súkromný kľúč) a potom opakujeme tento postup: generujeme náhodne otvorený text a vypočítame k nemu zodpovedajúci zašifrovaný

text. Spočítame dve čísla: s_1 – súčet otvorených textov a s_2 – homomorfný súčet zašifrovaných textov. Overíme, či dešifrovanie s_2 je totožné s s_1 a ak áno, tak zvýšime počítadlo úspešnosti pokračujeme. Ak nevyjde rovnosť, tak skončíme a počet korektne sčítaných zašifrovaných textov bude rovný doterajšiemu počítadlu. Tento postup opakujeme, kým nevyjde chyba pri dešifrovaní, alebo nedosiahneme nejakú hornú hranicu zadanú vopred (v našich experimentoch to bolo nakoniec[1] 5000 iterácií).

Keďže toto je znáhodnený proces a výsledok by mohol závisieť od šťastnej, či nešťastnej voľby konkrétneho zašifrovaného textu, tak celý experiment opakujeme niekoľkokrát[2] a za výsledok prehlásime minimum zo získaných čísel.

Voľby parametrov a výsledky

Popísaný experiment sme vykonali pre parameter `PT_BOUND` $= 2$ a všetky kombinácie `N` $= 128, 256$ a `t` $\in \{32, 42, 52, \ldots, 252\}$. V každej inštancii vyšiel počet korektne sčítaných zašifrovaných textov 5000, čo bola horná hranica opakovaní.

Tieto výsledky sú napriek tomu v súlade s teóriou, pretože šifrovanie prebieha ako pripočítanie nejakého náhodného chybového vektora ku vektoru z mriežky danej ideálom J (maticou B_{pk}). Pri sčítovaní dvoch zašifrovaných textov sa aj tieto chyby sčítavajú, pričom súčet týchto chybových vektorov má strednú hodnotu nulovú.

4.1.2 Multiplikatívnosť

Metodika

Pri zisťovaní, aký maximálny stupeň monomiálu nad zašifrovanými textami dokáže schéma ešte korektne vyhodnotiť sme použili rovnaký postup, ako v predchádzajúcej časti. Najprv sme vygenerovali dva OT-ZT páry a spočítali súčin otvorených textov s_1 a homomorfný súčin zašifrovaných textov s_2. V prípade, že dešifrovanie výsledku s_2 sa rovnalo súčinu s_1, tak sme vygenerovali ďalší OT-ZT pár, prepočítali súčiny a znova porovnali výsledky. Opakovali sme dovtedy, kým nevyšli rôzne výsledky, alebo sme nedosiahli hornú hranicu opakovaní. Za maximálny počet násobení sme vrátili najvyšší počet OT-ZT párov, ktoré ešte schéma vyhodnotila korektne.

Tento postup sme zopakovali 30-krát a za skutočný maximálny počet korektných násobení sme prehlásili minimum zo získaných výsledkov.

[1] Najprv sme zvolili oveľa menšie číslo, ale po dosiahnutí maximálnej hranice sme ho zvyšovali postupne až na 50000 pre jednu kombináciu parametrov. Z dôvodu veľkej časovej náročnosti sme túto hranicu nepoužili pri všetkých voľbách parametrov, no domnievame sa, že túto hranicu by sme dosiahli aj pri ostatných voľbách.

[2] Konkrétny počet opakovaní sme volili za behu a postupne sme ho zvyšovali až na 10. Keďže výsledky ukazovali, že počet sčítaní je neobmedzený, tak sme nevideli význam opakovať experiment viackrát.

Voľby parametrov a výsledky

Na rozdiel od sčitovania, tento experiment už nedosiahol hornú hranicu počtu opakovaní. Výsledky záviseli od parametra `t`, naopak nezáviseli na voľbe parametra N (pre testované `N` $= 128, 256$. Zobrazujeme preto výsledky pre `N` $= 128$, `PT_BOUND` $= 2$ a pre všetky možnosti `t` $\in \{64, 68, 72, \ldots, 128\}$. Výsledky sú zobrazené na grafe 4.1.

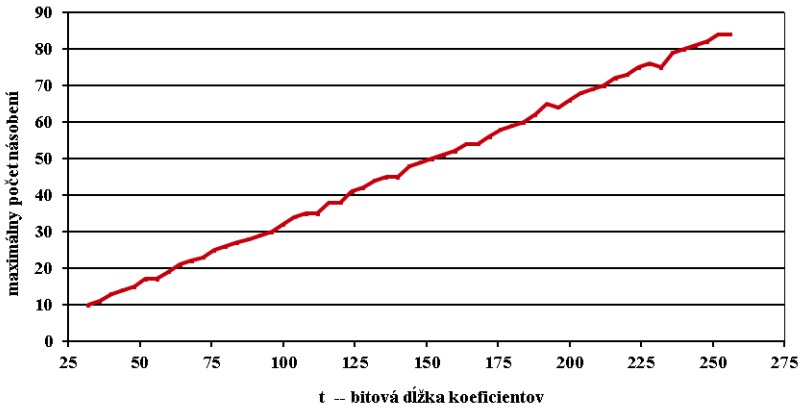

Figure 4.1: Maximálny počet násobení pre $N = 128$ a rôzne parametre t.

Keďže schéma je navrhnutá tak, že pri homomorfnom násobení dvoch zašifrovaných textov sa chybové vektory tiež vynásobili (násobenie prvkov v okruhu R), tak výsledná veľkosť chybového vektora je rovná zhruba súčinu dĺžok pôvodných chybových vektorov, čo potvrdzujú získané výsledky.

Zo zobrazených výsledkov je vidieť, že už táto čiastočne homomorfná schéma ďaleko presiahla doterajšie výsledky v oblasti homomorfných kryptosystémov, pretože poskytuje možnosť spočítania pomerne veľkého počtu násobení, alebo prakticky neobmedzeného počtu sčítaní.

4.1.3 Polynomiálne funkcie

Metodika

Pri vyhodnocovaní, aké všeobecné polynómy sa dajú danou schémou korektne vyhodnotiť sa opierame o fakt, že spomedzi všetkých polynómov stupňa d bude najväčšia chyba pri elementárnom symetrickom polynóme d-teho stupňa. Je to preto, že tento obsahuje všetky možné monomiály stupňa d a všetky monomiály menšieho stupňa ako d budú mať rádovo menšie chybové vektory, takže vo výslednom chybovom vektore budú zanedbateľné.

Postup, ktorým zisťujeme maximálny podporovaný stupeň polynómov, ktoré schéma homomorfne vyhodnotí korektne, realizovali Gentry a Halevi v [13]. Pre dané parametre `N` a `t` zvolili počet premenných m, nad ktorými budú polynómy vyhodnocované a náhodne vygenerovali m párov OT-ZT. Potom postupne vyhodnotili elementárny polynóm nad otvorenými textami a nad zašifrovanými textami a porovnali, či sa výsledky po dešifrovaní zhodujú. Pre každú voľbu m zapísali najvyššiu hodnotu stupňa polynómu, ktorý ešte vrátil korektné výsledky. Tento postup opakovali 12-krát.

Voľby parametrov a výsledky

V originálnom článku [13] boli zvolené hodnoty $\texttt{N} \in \{128, \ldots, 2048\}$, $\texttt{t} \in \{64, 128, 256, 384\}$ a $m \in \{64, 96, 128, 192, 256\}$. Pre všetky hodnoty `N` dosiahli veľmi podobné výsledky. Závery boli, že maximálny podporovaný stupeň závisí lineárne od `t` a pomaly klesá s rastúcim m, pričom nezávislosť od `N` bola očakávaná kvôli voľbe $q = 1 - 20/N$, kde q ovplyvňuje množstvo šumu, teda veľkosť chybového vektora čerstvého zašifrovaného textu v schéme.

Našim cieľom bolo okrem porovnania správnosti našej implementácie aj potvrdenie týchto záverov, hlavne lineárnej závislosti maximálneho stupňa od `t`, keďže v pôvodnom článku boli na grafe len štyri hodnoty. Zvolili sme preto iba jeden parameter $\texttt{N} = 128$ a 24 rôznych $\texttt{t} \in \{64, 72, \ldots, 256\}$, pre tri rôzne $m \in \{64, 80, 96\}$. Výsledky sú zobrazené na grafe 4.2.

Konštatujeme, že pre štyri konfigurácie parametrov, ktoré sa zhodujú s [13] sme dosiahli dve presne rovnaké hodnoty maximálnych stupňov, a dve o jedno vzdialené od pôvodných výsledkov. Vzhľadom na to, že chybové vektory sú generované náhodne a výsledný maximálny podporovaný stupeň závisí od konkrétnych hodnôt chybových vektorov to považujeme za zhodné výsledky s pôvodným článkom.

Pri podrobnejšom zostrojení grafu môžeme hovoriť o lineárnej závislosti maximálneho stupňa od t, aj keď priebehy grafov pre $m = 64$ a 80 naznačujú, že by sa mohlo skôr jednať o lomenú čiaru so zlomom okolo hodnôt $t = 150$ resp. $t = 180$. (TODO: buď doplniť reálne podrobnejšie výsledky, alebo teoreticky spochybniť tú lineárnu závislosť!).

Z výsledkov si môžeme všimnúť, že maximálny podporovaný stupeň polynómu je pomerne blízky podporovanému počtu násobení, čo sa dalo očakávať z dôvodu väčšieho

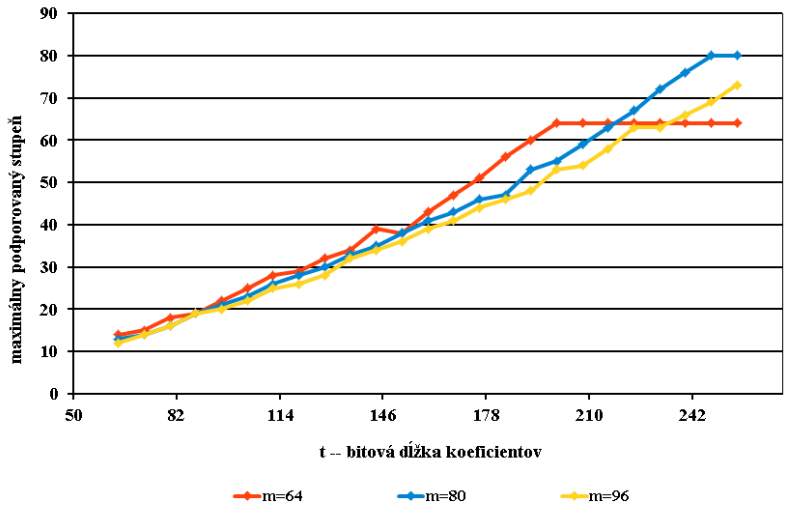

Figure 4.2: Závislosť maximálneho podporovaného stupňa od t pre voľby $N = 128$ a $m = 64, 80, 96$.

nárastu chyby pre násobenie ako pre sčitovanie. Prakticky nám takáto čiastočne homomorfná schéma umožňuje spočítanie veľkého množstva polynomiálnych výrazov aj bez transformácie na plne homomorfnú schému pomocou bootstrappingu, čo je už samo o sebe rádovo lepší výsledok, ako bol dosiahnutý v predchádzajúcich prácach.

4.2 Rôzne voľby ideálu I

Výsledky z prechádzajúcej časti nás motivovali k modifikácii priestoru otvorených textov v originálnej schéme, čo zodpovedá zmene ideálu I. Pôvodná schéma totiž umožňovala vyhodnotenie pomerne zložitých výrazov, ale iba nad premennými, ktorých hodnoty sú nula alebo jedna. Pre praktické výpočty to znamená, že celé čísla je nutné reprezentovať napríklad v binárnej sústave a podľa toho s nimi aj vykonávať homomorfné operácie.

4.2.1 Homomorfné výpočty v binárnej sústave

Klasické sčitovanie a násobenie binárnych čísel však nie je jednoduchá operácia, ak každý bit je reprezentovaný nejakým zašifrovaným textom. Pri sčítaní dvoch m-

prvé číslo:				a_2	a_1	a_0
druhé číslo:			$\times$	b_2	b_1	b_0
prenos z -2:		c_4				
prenos z -1:	d_5	d_4	d_3	d_2		
				x_2	x_1	x_0
			y_3	y_2	y_1	
		z_4	z_3	z_2		
výsledok:	w_5	w_4	w_3	w_2	w_1	w_0

Table 4.1: Schéma pre "ručné" násobenie dvoch trojbitových čísel.

bitových čísel je nutné realizovať $O(m)$ sčítaní, $O(m)$ násobení a výpočet najvýznamnejšieho bitu výsledného čísla vyžaduje vyhodnotenie polynómu stupňa m nad zašifrovanými bitmi pôvodných čísel.

Pri klasickom školskom násobení dvoch m bitových čísel je to ešte komplikovanejšie. Kvôli prehľadnosti uvedieme v Tab.4.2.1 príklad násobenia dvoch trojbitových čísel. Tu je nutné počítanie tzv. prenosov cez rád. Prenos cez i rádov (teda bit, ktorý sa prenesie zo stĺpca k stĺpcu $k+i$[3]) sa v tomto binárnom prípade spočíta ako elementárny symetrický polynóm stupňa 2^i nad bitmi stĺpca k. Keďže každý z bitov x_i, y_i, z_i je súčinom príslušných bitov a_k a b_j, teda ich stupeň je už dva, tak prenosový bit d_2 je už stupňa 4 (súčin x_1 a y_1). Bitu c_4 – prenos z druhého do štvrtého stĺpca – je rovný súcinu všetkých šiestich bitov pôvodných čísel a z toho vyplýva, že stupeň polynómu potrebného na vyhodnotenie bitov w_4 a w_5 je 6.

4.2.2 Všeobecná voľba $I = (p)$

Z predchádzajúcej časti je zrejmé, že binárna reprezentácia čisel nie je vhodná na homomorfné počítanie, pretože stupeň výrazov potrebných na homomorfné výnásobenie dvoch m-bitových čísel je $2m$, takže napríklad schéma s parametrami $N = 128$, $t = 128$ (maximálny podporovaný stupeň je 31) by zvládla nanajvýš jedno vynásobenie dvoch 15-bitových čísel. Z tohto dôvodu sme testovali rôzne iné voľby ideálu I, konkrétne $I = (p)$, kde p je prvočíslo.

Pri inej voľbe ako $I = (2)$ však musíme poznamenať, že zatiaľ nie je známy spôsob, ako by sa takáto čiastočne homomorfná schéma dala transformovať na plne homomorfnú, pretože konštrukcia z [13] silne využívala binárne operácie, špeciálne vlastnosti operácie XOR.

Homomorfné vlastnosti zmenenej schémy

Pre každú voľbu p sme rovnakým spôsobom ako v časti4.1 určovali homomorfné vlastnosti. Pre porovnanie sme zvolili už len $N = 128$ a podobne sme skúšali rôzne voľby

[3]Číslujeme tak, že poradové čísla stĺpcov rastú smerom doľava.

t. Počet homomorfných sčítaní opäť vyšiel neobmedzený, takže na nasledujúcom grafe 4.3 zobrazujeme podporovaný počet násobení pre p menšie ako 15.

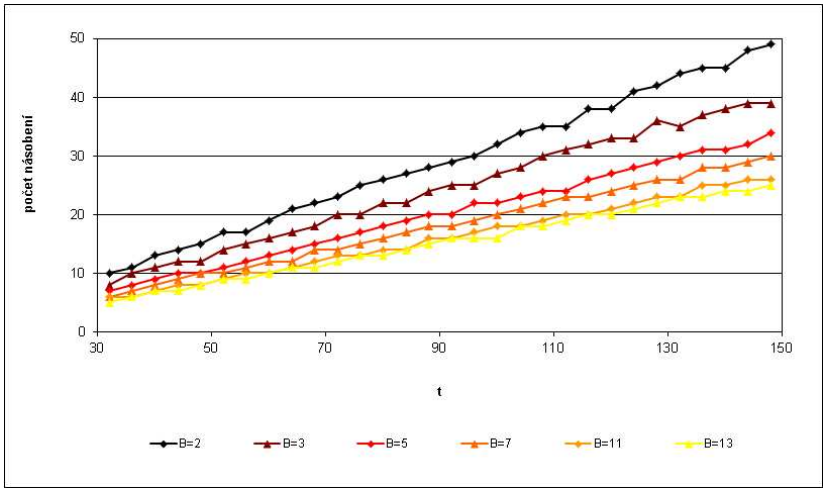

Figure 4.3: Maximálneho počet násobení v závislosti od t pre $N = 128$ a rôzne voľby $I = (B)$.

Maximálny podporovaný stupeň polynómu sme testovali v rovnakej dimenzii $N = 128$ a pre prehľadnosť sme zvolili už len počet premenných $m = 80$. Na grafe 4.4 sú výsledky pre rôzne hodnoty prvočísla p.

Z grafu je vidieť, že výsledky sú podobné pre obe sledované vlastnosti zmenenej schémy. Maximálny podporovaný počet násobení je pre $p = 13$ zhruba polovičný ako pre $p = 2$ a v prípade maximálneho podporovaného stupňa polynómu sú hodnoty pre $p = 13$ ostro väčšie ako polovica z hodnôt pre $p = 2$. Z toho vyplýva, že keby sme mali realizovať čiastočne homomorfnú schému pre čísla do 8, tak sa viac oplatí použiť schému s $I = (11)$, ako pôvodnú $I = (2)$ a reprezentovať čísla troma bitmi v binárnej sústave.

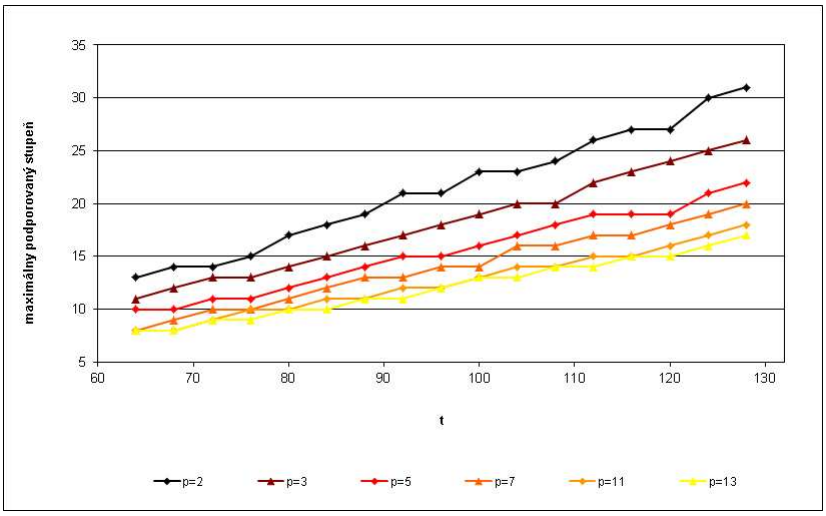

Figure 4.4: Závislosť maximálneho podporovaného stupňa od t pre voľby $N = 128$, $m = 80$ a rôzne $I = (p)$.

4.2.3 Čiastočne homomorfná schéma pre ohraničene veľké čísla

Využívajúc výsledky z predchádzajúcej časti 4.2.2 a čínsku zvyškovú vetu môžeme skonštruovať jednoduchý čiastočne homomorfný kryptosystém s ľubovoľne veľkým priestorom otvorených textov B.

Postupovať budeme tak, že B rozložíme na súčin prvočísel p_i pre $i = 1, \ldots, k$ a pre každé z nich definujeme osobitný kryptosystém s ideálom $I = (p_i)$. Každý otvorený text $m \in B$ sa dá jednoznačne rozložiť na časti m_i a tejto bude prislúchať zašifrovaný text c_i. Každá homomorfná operácia sa vykoná po zložkách modulo každé prvočíslo p_i a pri dešifrovaní sa dešifrujú jednotlivé časti m'_i a pomocou čínskej zvyškovej vety sa dopočíta naspäť otvorený text z B.

Nevýhodou takéhoto kryptosystému je, že jeden otvorený text sa zašifruje na k zašifrovaných textov a pamäťová aj časová náročnosť každej operácie vzrastie k-násobne.

Appendix A

Zdrojové kódy

Keďže práca obsahuje viaceré praktické experimenty, ktorých význam je založený na výstupoch našej aplikácie, tak považujeme za nutné poskytnúť tieto zdrojové kódy a zdôvodniť ich korektnosť. Uvedieme len najdôležitejšie časti programu a to hlavné funkcie čiastočne homomorfnej schémy na generovanie kľúčov, šifrovanie, dešifrovanie a homomorfné operácie.

A.1 Keygen funkcia

V tejto časti ukážeme správne fungovanie generovania kľúčov. Pre rozsiahlosť kódov sme niektoré riadky s kontrolnými výpismi a komentármi vynechali.

```
1  int shs_keygen_CIC(long PT_BOUND, long N, long t,
2                     ZZ& d, ZZ& r, ZZ& w, int verbose)
3     /*
4     parameters for keygen_CIC:
5          PT_BOUND - prvocislo (=2 pre gentryho original) idealu I
6        N - dimenzia
7        t - bitova velkost koeficientov vx
8        d,r - PK
9        w - SK
10       verbose - 0/1 -- 1 ci pisat kontrolne vypisy
11    */
12 {
13    ZZX fx,vx,vx1,temp;
14    ZZ coef,coef0,coef1,x,rN, r_inv, d_temp;
15    int HNF=0,i,iter=0;
16    ofstream fout("shs_keygen_CIC.log",ofstream::out);
17
18    fx.SetMaxLength(N+1);
19    SetCoeff(fx,0,1);
20    SetCoeff(fx,N,1);
21    if (verbose==1) fout << "fx: " << fx << endl << endl;
```

```
22
23      while (HNF==0)
24      {
25          // set vx:
26          for (i=0;i<N;i++)
27          {
28              coef=RandomBits_ZZ(t);
29              SetCoeff(vx,i,coef);
30          }
31          if (verbose==1) fout << "vx: " << vx << endl << endl;
32
33          coef0=compute_inverse_coeff(N,vx,fx,d,0);
34          if (d%2==1)
35          {
36             rem(vx1, LeftShift(vx,1), fx);
37             coef1=compute_inverse_coeff(N,vx1,fx,d_temp,0);
38             if (d<0)
39             {
40                d=d*(-1);
41                coef0*=-1;
42                coef1*=-1;
43             }
44             if (verbose==1) fout << "coef0: " << coef0 << endl;
45             if (verbose==1) fout << "coef1: " << coef1 << endl;
46             if (verbose==1) fout << "d: " << d << endl;
47
48             if (GCD(coef1,d)==1)
49             {
50                x=InvMod(coef1%d,d);
51                r=(coef0*x)%d;
52                if (verbose==1) fout << "r : " << r << endl;
53                rN = PowerMod(r,N,d);
54                if (verbose==1) fout << "rN: " << rN << endl;
55                if (rN == (d-1))
56                   HNF=1;
57             }
58          }
59          iter++;
60      } // end while HNF==0;
61      if ((coef0%PT_BOUND)==1)
62      {
63         w=coef0;
64      }
65      else if ((coef1%PT_BOUND)==1) {
66         w=coef1;
67      } else
```

```
68      {
69          r_inv = InvMod(r,d);
70          for (i=2;i<N;i++)
71          {
72              coef=(coef1*r_inv)%d;
73              if (coef>=d/2) coef-=d;
74              if ((coef%PT_BOUND)==1)
75              {
76                  w=coef;
77                  break;
78              }
79              coef1=coef;
80          }
81      }
82      return iter;
83  }
```

A.1.1 Počítanie inverzných koeficientov

Funkcia `compute_inverse_coeff` vráti koeficient pri lineárnom člene polynómu $w(x)$ inverzného ku $v(x)$, kde $w(x) \cdot v(x) = d \bmod f(x)$.

TODO - upravit kody (odstranit komentare) a zdovodnit spravnost. Podla [13].

```
1   ZZ compute_inverse_coeff(int N, ZZX vx, ZZX fx, ZZ& d, int verbose)
2   {
3       ZZ g_0,g_1;
4       ZZX U,V,A,B,fnx, U1,V1;
5       int nj, i, j, logN;
6
7       U.SetMaxLength(N+1);
8       U1.SetMaxLength(N+1);
9       V.SetMaxLength(N+1);
10      V1.SetMaxLength(N+1);
11      A.SetMaxLength(N+1);
12      B.SetMaxLength(N+1);
13
14      U=1; V=vx;
15      logN = (log(1.0*N)/log(2.0));
16      nj=N;
17      fnx=fx;
18      j=0;
19      V1=V;
20      for (i=1;i<=deg(V1);i+=2)
21          SetCoeff(V1,i,(-1)*coeff(V,i));
22      U1=U;
23      for (i=1;i<=deg(U1);i+=2)
```

```
24          SetCoeff(U1,i,(-1)*coeff(U,i));
25      rem(A,(V*V1),fnx);
26      rem(B,(U*V1+U1*V),fnx);
27
28      for (j=1;j<=logN;j++)
29      {
30          SetCoeff(fnx,nj,0);
31          nj=nj/2;
32          SetCoeff(fnx,nj,1);
33          clear(U); for (i=0;i<=nj-1;i++) SetCoeff(U,i,coeff(B,2*i));
34          clear(V); for (i=0;i<=nj-1;i++) SetCoeff(V,i,coeff(A,2*i));
35
36          V1=V;
37          for (i=1;i<=deg(V1);i+=2)
38              SetCoeff(V1,i,(-1)*coeff(V,i));
39          U1=U;
40          for (i=1;i<=deg(U1);i+=2)
41              SetCoeff(U1,i,(-1)*coeff(U,i));
42          rem(A,(V*V1),fnx);
43          rem(B,(U*V1+U1*V),fnx);
44
45      }
46      g_0 = ConstTerm(V);
47      d=g_0;
48      g_1 = ConstTerm(U);
49      return g_1/N;
50  }
```

A.2 Encrypt

```
1   void shs_encrypt(long PT_BOUND, ZZ& CT, int PT, int N,
2                    ZZ d, ZZ r, double q, int verbose)
3   {
4       vec_ZZ rN;
5       int i, logN;
6       double ran,q1;
7       ZZ ur,d_half;
8       vec_ZZ ux;
9       ofstream fout("shs_encrypt.log",ofstream::out);
10
11      logN = (log(1.0*N)/log(2.0));
12      rN.SetLength(logN+1);
13      rN[1]=r;
14      for (i=2;i<=logN;i++)
15      {
```

```
16          rN[i]=(rN[i-1]*rN[i-1])%d;
17          if (verbose==1) fout << ", " << rN[i];
18      }
19      if (verbose==1) fout << endl;
20
21      q1=(1-q)/2+q;
22      ux.SetLength(N);
23      do {
24          for (i=0;i<N;i++)
25          {
26              ran=RandomBnd(1583)/1583.0;
27              if (ran < q) ux[i]=to_ZZ(0);
28              else if (ran < q1) ux[i]=to_ZZ(1);
29              else ux[i]=to_ZZ(-1);
30          }
31      } while (IsZero(ux));
32      if (verbose==1) fout << "ux: " << ux << endl;
33
34      //logN = (log(1.0*N)/log(2.0));
35      ur=eval_poly(ux,0,N-1,logN,rN,d,0);
36      CT=(PT_BOUND*ur+PT)%d;
37      if (CT>=d/2) CT=CT-d;
38      if (verbose==1) fout << "encrypt output: " << CT << endl << endl;
39  }
```

A.2.1 Vyhodnotenie polynómu v bode r

Funkcia `eval_poly()` vyhodnotí vstupný polynóm v bode r, pričom predpokladá stupeň polynómu $N-1$, kde N je mocnina dvojky a ako vstupy predpočítané hodnoty r^e, kde e sú mocniny dvojky. Funkcia funguje rekurzívne, rozdelí polynóm dna dve polovice a vyhodnotí ho ako súčet prvej a druhej. Výhoda rozdelenia na polovice je, že obe časti sú vyhodnocované ako polynóm stupňa $N/2-1$ a druhá časť je prenásobená hodnotou $r^{N/2}$, ktorá už bola predpočítaná a je vstupom do funkcie.

```
1   ZZ eval_poly(vec_ZZ ux, int a, int b, int depth, vec_ZZ rN,
2                ZZ d, int verbose)
3   /*Parametre:
4         ux - polynom
5         a - dolny rozsah mocnin koeficientov x^i
6         b - horny rozsah mocnin koeficientov x^i
7         depth - vyjadruje, ktoru mocninu dvojky prace pocitame
8         rN - vektor mocnin r^i
9         d - verejny parameter, pocitame modulo d
10        verbose - ci mame vypisovat kontrolne medzivysledky
11  */
12  {
13      ZZ temp;
```

```
14      if (depth==1)
15      {
16          return (ux[a]+ux[b]*rN[1])%d;
17      }
18      else
19      {
20          temp = eval_poly(ux,a,a+(b+1-a)/2-1,
21                                  depth-1,rN,d,verbose);
22          temp = (temp + eval_poly(ux,a+(b+1-a)/2,b,depth-1,
23                                          rN,d,verbose)*rN[depth])%d;
24          return temp;
25      }
26  }
```

A.3 Decrypt

Funkcia `shs_decrypt()` je už jednoduchá a podľa [13] je jej predpis $pt = [ct \cdot w]_d \bmod 2$. V našej všeobecnej verzii sme konkrétny rozsah ideálu I nahradili všeobecným prvočíslom `PT_BOUND` a tak záverečná operácia je modulovanie týmto číslom.

```
1  int shs_decrypt( long PT_BOUND, ZZ CT, ZZ d, ZZ w )
2  {
3      ZZ temp;
4      temp = (CT*w)%d;
5      if (temp>=d/2) temp-=d;
6      return temp%PT_BOUND;
7  }
```

Bibliography

[1] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, *Relations among notions of security for public-key encryption schemes*, in CRYPTO, H. Krawczyk, ed., vol. 1462 of Lecture Notes in Computer Science, Springer, 1998, pp. 26–45.

[2] D. Boneh, E.-J. Goh, and K. Nissim, *Evaluating 2-dnf formulas on ciphertexts*, in TCC, J. Kilian, ed., vol. 3378 of Lecture Notes in Computer Science, Springer, 2005, pp. 325–341.

[3] D. Boneh and R. J. Lipton, *Algorithms for black-box fields and their application to cryptography (extended abstract)*, in CRYPTO, N. Koblitz, ed., vol. 1109 of Lecture Notes in Computer Science, Springer, 1996, pp. 283–297.

[4] E. F. Brickell and Y. Yacobi, *On privacy homomorphisms (extended abstract)*, in EUROCRYPT, D. Chaum and W. L. Price, eds., vol. 304 of Lecture Notes in Computer Science, Springer, 1987, pp. 117–125.

[5] Y. Chen and P. Q. Nguyen, *Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers*. Cryptology ePrint Archive, Report 2011/436, 2011. `http://eprint.iacr.org/2011/436/`.

[6] J. Cheon, W. H. Kim, and H. Nam, *Known-plaintext analysis of the domingo-ferrer algebraic privacy homomorphism scheme*, Information Processing Letters, 97 (2006), pp. 118–123.

[7] J. H. Cheon and H. S. Nam, *A cryptanalysis of the original domingo-ferrer's algebraic privacy homomophism*, Cryptology ePrint Archive, Report 2003/221. Available online at:`http://eprint.iacr.org/2003/221` .

[8] J. Domingo-Ferrer, *A provably secure additive and multiplicative privacy homomorphism*, in ISC, A. H. Chan and V. D. Gligor, eds., vol. 2433 of Lecture Notes in Computer Science, Springer, 2002, pp. 471–483.

[9] M. Fellows and N. Koblitz, *Combinatorial cryptosystems galore!*, Contemporary Mathematics, vol. 168 of Finite Fields: Theory, applications and Algorithms (1993), pp. 51–61.

[10] J. D. Ferrer, *A new privacy homomorphism and applications*, Information Processing Letters, 60 (1996), pp. 277–282.

[11] C. Gentry, *A fully homomorphic encryption scheme*, PhD thesis, Standford University, 2009.

[12] ——, *Fully homomorphic encryption using ideal lattices*, in Proceedings of the 41st annual ACM symposium on Theory of computing (STOC'09), 2009, pp. 169–178.

[13] C. Gentry and S. Halevi, *Implementing gentry's fully-homomorphic encryption scheme*, in Proceedings of the 30th Annual international conference on Theory and applications of cryptographic techniques: advances in cryptology, EUROCRYPT'11, Berlin, Heidelberg, 2011, Springer-Verlag, pp. 129–148.

[14] N. Howgrave-graham, *Approximate integer common divisors*, in CaLC 2001, Lecture Notes in Computer Science, Springer-Verlag, 2001, pp. 51–66.

[15] A. Kawachi, K. Tanaka, and K. Xagawa, *Multi-bit cryptosystems based on lattice problems*, in Public Key Cryptography – PKC 2007, T. Okamoto and X. Wang, eds., vol. 4450 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2007, pp. 315–329.

[16] T. Matsumoto, K. Kato, and H. Imai, *Speeding up secret computations with insecure auxiliary devices*, in Advances in Cryptology — CRYPTO' 88, Lecture Notes in Computer Science, Springer-Verlag, 1990, pp. 497–506.

[17] C. Melchor, P. Gaborit, and J. Herranz, *Additively homomorphic encryption with t-operand multiplications*, Cryptology ePrint Archive, Report 2008/378, (2008). Available online at:`http://eprint.iacr.org/2008/378` .

[18] J. Merkle, *Multi-round passive attacks on server-aided rsa protocols*, in ACM Conference on Computer and Communications Security, D. Gritzalis, S. Jajodia, and P. Samarati, eds., ACM, 2000, pp. 102–107.

[19] D. Micciancio, *The shortest vector problem is NP-hard to approximate to within some constant*, SIAM Journal on Computing, 30 (2001), pp. 2008–2035. Preliminary version in FOCS 1998.

[20] P. Q. Nguyen and I. Shparlinski, *On the insecurity of a server-aided rsa protocol*, in ASIACRYPT, C. Boyd, ed., vol. 2248 of Lecture Notes in Computer Science, Springer, 2001, pp. 21–35.

[21] P. Q. Nguyen and J. Stern, *The béguin-quisquater server-aided rsa protocol from crypto '95 is not secure*, in ASIACRYPT, K. Ohta and D. Pei, eds., vol. 1514 of Lecture Notes in Computer Science, Springer, 1998, pp. 372–379.

[22] P. Q. Nguyen and B. Vallée, *The LLL algorithm, Survey and Applications*, Information Security and Cryptography, Springer-Verlag, 2010.

[23] P. Paillier, *Public-key cryptosystems based on composite degree residuosity classes*, in EUROCRYPT, J. Stern, ed., vol. 1592 of Lecture Notes in Computer Science, Springer, 1999, pp. 223–238.

[24] D. K. Rappe, *Homomorphic cryptosystems and their applications*, PhD thesis, University of Dortmund, 2004. Available online at: `www.rappe.de/doerte/Diss.pdf`.

[25] R. Rivest, L. Adleman, and M. Dertouzos, *On data banks and privacy homomorphisms*, Foundations of Secure Computation, (1978), pp. 169–177.

[26] P. Schmidt, *Fully homomorphic encryption - overview and cryptanalysis*, master's thesis, University of Dortmund, 2011.

[27] V. Shoup, *A library for doing number theory, v.5.5.2.* New York University, New York, Available online at: `http:shoup.net/ntl/`.

[28] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, *Fully homomorphic encryption over the integers*, in EUROCRYPT, H. Gilbert, ed., vol. 6110 of Lecture Notes in Computer Science, Springer, 2010, pp. 24–43.

[29] J. Vaňo, *Plne homomorfné šifrovacie schémy.* bakalárska práca. Univerzita Komenského Bratislava, Slovakia, 2012.

[30] D. Wagner, *Cryptanalysis of an algebraic privacy homomorphism*, in ISC, C. Boyd and W. Mao, eds., vol. 2851 of Lecture Notes in Computer Science, Springer, 2003, pp. 234–239. Available online at: `www.cs.berkeley.edu/~daw/papers/` (revised version).