SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Evidenčné číslo: FEI-5384-86222

SKUPINOVÁ PODPISOVÁ SCHÉMA ZALOŽENÁ NA MPKC

DIPLOMOVÁ PRÁCA

2021 **Bc. Daniela Leščinská**

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Evidenčné číslo: FEI-5384-86222

SKUPINOVÁ PODPISOVÁ SCHÉMA ZALOŽENÁ NA MPKC

DIPLOMOVÁ PRÁCA

Študijný program:	Aplikovaná informatika
Číslo študijného odboru:	2508
Názov študijného odboru:	Informatika
Školiace pracovisko:	Ústav informatiky a matematiky
Vedúci záverečnej práce:	Ing. Viliam Hromada, PhD.

Bratislava 2021 **Bc. Daniela Leščinská**

Slovenská technická univerzita v Bratislave
Ústav informatiky a matematiky

Fakulta elektrotechniky a informatiky
Akademický rok: 2020/2021
Evidenčné číslo: FEI-5384-86222

ZADANIE DIPLOMOVEJ PRÁCE

Študentka:	**Bc. Daniela Leščinská**
ID študenta:	86222
Študijný program:	aplikovaná informatika
Študijný odbor:	informatika
Vedúci práce:	Ing. Viliam Hromada, PhD.
Miesto vypracovania:	Ústav informatiky a matematiky

Názov práce: **Skupinová podpisová schéma založená na MPKC**

Jazyk, v ktorom sa práca vypracuje: slovenský jazyk

Špecifikácia zadania:

V rámci NATO projektu G5448, ktorého riešiteľom je aj UIM FEI STU, sa rieši úloha skupinovej výmeny kľúča medzi viacerými účastníkmi. Paralelne s takouto skupinovou komunikaciou je zaujímavá otázka aj skupinovej podpisovej schémy, ktorá umožňuje, aby účastník dohodnutej skupiny podpísal anonymne dokument v jej mene. Navyše je atraktívna možnosť, že by táto schéma bola založená na algoritmoch postkvantovej kryptografie, napríklad algoritmoch založených na problematike riešenia sústavy nelineárnych polynómov viacerých neurčitých (MPKC).

Úlohy:

1. Naštudujte problematiku skupinového elektronického podpisu a MPKC.
2. Implementujte schému podľa zvolenej literatúry.
3. Riešenie vyhodnoťte.

Zoznam odbornej literatúry:

1. Malo, P. – Hromada, V. *Implementácia postkvantového protokolu na skupinovú výmenu kľúča.* Diplomová práca. Bratislava : 2020. 45 s.
2. MOHAMED, Mohamed Saied Emam; PETZOLDT, Albrecht. RingRainbow–an efficient multivariate ring signature scheme. In: International Conference on Cryptology in Africa. Springer, Cham, 2017. p. 3-20.

Riešenie zadania práce od: 15. 02. 2021

Dátum odovzdania práce: 14. 05. 2021

Bc. Daniela Leščinská
študentka

Dr. rer. nat. Martin Drozda
vedúci pracoviska

prof. Ing. Pavol Zajac, PhD.
garant študijného programu

SÚHRN

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Študijný program:	Aplikovaná informatika
Autor:	Bc. Daniela Leščinská
Diplomová práca:	Skupinová podpisová schéma založená na MPKC
Vedúci záverečnej práce:	Ing. Viliam Hromada, PhD.
Miesto a rok predloženia práce:	Bratislava 2021

Práca sa zaoberá konštrukciou prstencových podpisových schém nad kryptosystémami založenými na systéme polynómov viacerých neurčitých. Dôvodom výberu tohto typu kryptosystémov je, že patria medzi kandidátov na postkvantové algoritmy. V našej práci sa venujeme hlavne prstencovým podpisovým schémam, kde vystupuje ľubovoľný počet účastníkov, ktorí spoločne tvoria skupinu, pričom každý člen tohto zoskupenia dokáže vygenerovať podpis v mene skupiny. Tieto prstencové podpisové schémy zaručujú každému členovi anonymitu. Výsledkom práce je testovacia aplikácia vybraných prstencových schém nad kryptosystémami založenými na systéme polynómov viacerých neurčitých. Pre naše účely sme si vybrali algoritmy Rainbow a GeMSS, ktoré sú finalistami v súťaži NIST Post Quantum Cryptography, ktorej cieľom je výber vhodných kandidátov na štandardy postkvantovej kryptografie. Na základe experimentov sme zistili, že súčinová prstencová podpisová schéma je výhodnejšia oproti súčtovej podpisovej schéme, pretože parametre sústavy sa s pribúdajúcim počtom účastníkov nemenia, taktiež pri použití súčtovej podpisovej schémy nad Rainbow a GeMSS, bolo výhodnejšie použitie schémy Rainbow, avšak pri 50 užívateľoch sa situácia vymenila. Testovacia aplikácia umožňuje vygenerovať podpis, ktorý je overiteľný akýmkoľvek držiteľom verejného kľúča. Tieto jednotlivé prstencové podpisové schémy s rôznymi podpisovými algoritmami porovnáme, pričom sa pri experimentoch zameriame na veľkosti verejných kľúčov, súkromných kľúčov, veľkosti podpisov a na dobu vykonávania jednotlivých operácií generovania podpisu, verifikácie a generovania kľúčov.

Kľúčové slová: kryptosystémy založené na systéme polynómov s viacerými neurčitými, prstencové podpisové schémy, Rainbow, GeMSS, digitálny podpis

ABSTRACT

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA
FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

Study Programme:	Applied Informatics
Author:	Bc. Daniela Leščinská
Diploma Thesis:	A group signature scheme based on MPKC
Supervisor:	Ing. Viliam Hromada, PhD.
Place and year of submission:	Bratislava 2021

This thesis deals with multivariate signature schemes. The main reason why we choose this type of algorithms is fact, that multivariate cryptography is one of the candidates of post-quantum cryptosystems. Ring signature schemes are designed for an arbitrary number of users, which creates an aggregation, where every member can generate the signature on behalf of the group. Ring signature schemes ensure anonymity for each member of the ring. The result of this thesis is an application which implements the multivariate ring signature scheme over different underlying multivariate signature schemes. For our work, we choose algorithms Rainbow and GeMSS, which are candidates in NIST PQC competition. The results of experiments show, that multiply ring signature scheme is more convenient against sum signature scheme, because parameters of the polynomial system is constant with increasing number of users. With usage of sum ring signature scheme over Rainbow and GeMSS, usage of Rainbow was more convenient, but with system of 50 users the situation has changed. Functionality of the application consists of signature generation, which will be verifiable by every keeper of public key. In the conclusion we compare each of the ring signature scheme with different digital signature algorithms. By testing, we focus on the size of public keys, size of secret keys, size of signatures and we will measure execution time of the signature generation, signature verification and generation of keys.

Keywords: multivariate public key cryptogrpahy, ring signature schemes, Rainbow, GeMSS, digital signature

Poďakovanie

Chcela by som poďakovať za podporu, pomoc a každú cennú radu môjmu vedúcemu práce Ing. Viliamovi Hromadovi, PhD.

Obsah

Zoznam obrázkov a tabuliek

Zoznam skratiek a značiek

MPKC - Multivariate Public Key Cryptography
SME - Simultaneous Multivariate Equations
MQ - Multivariate Quadratic
UOV - Unbalanced Oil and Vinegar
MIA - Matsumo-Imai Scheme
HFE - Hidden Field Equations
STS - Stepwise Triangular System
GeMSS - Great Multivariate Short Signature
GF - Galois Field

Zoznam algoritmov

Úvod

Hrozba kvantových počítačov sa nenávratne blíži a preto dnešní kryptografi stoja pred neľahkou úlohou. Majú viacero možností, buď budú skúmať nové algoritmy alebo oprášia staršie, no už dobre známe matematické problémy, ktoré sú odolné voči útokom na kvantových počítačoch. Jedným z týchto starších matematických problémov je hľadanie koreňov sústavy polynómov viacerých neurčitých nad konečným poľom.

Kryptosystémy založené na tomto matematickom probléme sa používajú najmä na tvorbu podpisov a spomedzi mnohých postkvantových algoritmov zaručujú jeden z najkratších podpisov[3], čo je pri bežnom používaní veľmi výhodná vlastnosť. Avšak v našej práci sa nebudeme zaoberať iba bežnými jednoduchými podpismi, budeme skúmať schémy, ktoré sú určené pre viacero užívateľov. Myslíme si, že takéto skupinové podpisy budú v budúcnosti viac rozšírené, pretože skupinové činnosti sa postupne prenášajú do virtuálneho sveta, a sú činnosti ako napríklad politické voľby, pri ktorých musí byť zaručená anonymita každého občana a to nám tieto prstencové podpisové schémy poskytujú. V práci budeme skúmať prstencové podpisové schémy, ktoré zabezpečujú úplnú anonymitu pre každého člena skupiny. Pre implementáciu sme si zvolili prstencové podpisové schémy z článku Mohammeda a Petzolda.[8] Našim zámerom je tieto prstencové podpisové schémy implementovať pre kryptosystémy založené na sústave polynómov s viacerými neurčitými, keďže dané kryptosystémy patria medzi postkvantové algoritmy, budú vhodné aj do budúcnosti. Medzi takéto algoritmy, ktoré používame aj pri našej praktickej práci, patria Rainbow a GeMSS, ktoré sú medzi finalistami súťaže NIST, ktorej cieľom je identifikácia kvantovo bezpečných podpisových schém. Našim cieľom je porovnať, ktoré algoritmy budú pre dané prstencové schémy vhodnejšie a efektívnejšie. Rozhodovacími faktormi budú veľkosti verejných kľúčov, veľkosť podpisu a doba trvania podpisovania a verifikácie.

Štruktúra práce je nasledovná: v prvej kapitole sa venujeme úvodu do asymetrickej kryptografie, v druhej kapitole priblížime kryptosystémy založené na sústave polynómov viacerých neurčitých, pričom sa pozrieme na štruktúru verejného kľúča, súkromného kľúča, tvorbu a verifikáciu podpisu. V tretej kapitole preskúmame jednosmerné funkcie, ktoré sa používajú v kryptosystémoch založených na sústave polynómov viacerých neurčitých, v štvrtej kapitole si popíšeme prstencové podpisové schémy, ktoré budeme v našej práci implementovať. V piatej kapitole si načrtneme ako budeme prácu riešiť a popíšeme požiadavky na vypracovanie zadania a nakoniec v poslednej šiestej kapitole sú prezentované výsledky našich experimentov s prstencovými podpisovými schémami.

1 Asymetrická kryptografia

Kryptografia sa používa celé tisícročia na zabezpečenie utajenej komunikácie medzi subjektmi, ktoré si navzájom dokážu prirodzene dôverovať.[11] Do 70. rokov 20.storočia bola po svete rozšírená iba symetrická kryptografia.[6] Pri tomto druhu kryptografie používa odosielateľ a príjmateľ správy rovnaký tajný kľúč. To znamená, že tajným kľúčom sa správa zašifruje a rovnakým kľúčom sa správa aj dešifruje. Hlavným dôvodom prečo sa vedci snažili vynájsť nový druh kryptografie bol ten, že pri symetrickej kryptografii si účastníci komunikácie musia pred výmenou správy tento tajný kľúč vymeniť. Tento problém viedol k otázke ako si bezpečne vymeniť kľúč aj na veľké vzdialenosti, keď útočník môže číhať kdekoľvek. Preto výskum smeroval k spôsobom akým bezpečne previesť kľúč po sieti alebo vymyslieť taký druh kryptografie, kde by ku výmene tajného kľúča ani nemuselo dôjsť.

Tu na scénu prichádza kryptografia s verejným kľúčom alebo inak *asymetrická kryptografia*. Tento druh kryptografie využíva na šifrovanie a dešifrovanie dva odlišné typy kľúčov a to súkromný kľúč a verejný kľúč. *Súkromný kľúč* ako už z názvu vyplýva je známy iba vlastníkovi kľúča, ktorý správu dešifruje. *Verejný kľúč* je známy verejne a slúži na šifrovanie správy. Obidva tieto kľúče sú späté s určitým ťažkým matematickým problémom ako napríklad faktorizácia alebo problém diskrétneho logaritmu. Tieto ťažké matematické problémy majú spoločné to, že výpočet verejného kľúča zo súkromného je jednoduchý avšak výpočet súkromného kľúča z verejného by mal byť nemožný, takéto funkcie nazývame aj *trapdoor funkcie* alebo jednosmerné funkcie. V našom prípade budeme uvažovať ťažký matematický problém založený na systéme polynómov s viacerými neurčitými.

1.1 Digitálny podpis

Pri našej práci budeme pracovať hlavne s digitálnym podpisom, preto je dôležité vysvetliť si aj tento pojem. Formálna definícia digitálneho podpisu podľa [11] znie:
Podpisová schéma predstavuje päticu $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$, kde platia nasledové podmienky:

1. $\mathcal{P}$ je konečná množina možných správ
2. $\mathcal{A}$ je konečná množina možných podpisov
3. $\mathcal{K}$ predstavuje priestor možných kľúčov
4. Pre každé $K \in \mathcal{K}$, existuje podpisový algoritmus $sig_K \in \mathcal{S}$ a k nemu prislúchajúci verifikačný algoritmus $ver_K \in \mathcal{V}$. Pre každé $sig_K : \mathcal{P} \to \mathcal{A}$ a $ver_K : \mathcal{P} \times \mathcal{A} \to$

$\{true, false\}$ sú funkcie(pre niektoré podpisovacie algoritmy sú znáhodnené), kde pre každú správu $x \in \mathcal{P}$ a pre každý podpis $y \in \mathcal{A}$ platí nasledovný vzťah:

$$ver_K(x, y) = \begin{cases} true & \text{if } y = sig_K(x) \\ false & \text{if } y \neq sig_K(x) \end{cases}$$

Pár (x, y), kde $x \in \mathcal{P}$ a $y \in \mathcal{A}$ predstavuje podpísanú správu.

V podpisovej schéme súkromný kľúč vstupuje do podpisového algoritmu ako parameter, teda odosielateľ ho použije pri podpise správy. Výsledkom tohto podpisovacieho algoritmu je podpis, ktorý závisí na podpisovanej správe, ale aj na kľúči. Podpis sa posiela spolu s podpísanou správou. Na druhej strane príjmateľ, ktorý si chce overiť či dokument obsahuje platný podpis odosielateľa, použije overovací algoritmus, kde vstup predstavuje dokument, podpis dokumentu a verejný kľúč. Výsledkom je odpoveď či je podpis validný alebo nie. Podpisová schéma je určená hlavne na kontrolu integrity a validáciu pravosti podpisu. Podpis môže byť overený kýmkoľvek, kto má prístup k verejnému kľúču, to nám zaručuje, že tvrdenie odosielateľa, že dokument nepodpísal je nevyvrátiteľné, pretože súkromný a verejný kľúč tvoria neoddeliteľnú dvojicu už pri vygenerovaní.[11]

2 Postkvantové kryptosystémy

Existujú rozdielne ťažké matematické problémy, na ktorých sú postavené dnešné asymetrické kryptosystémy. Ako príklad uvedieme kryptosystém RSA, ktorý je postavený na probléme faktorizácie alebo podpisová schéma DSA, ktorá je založená na probléme diskrétneho logaritmu. Avšak v roku 1994, Peter Shor predstavil algoritmus [10], ktorý dokáže riešiť problémy ako diskrétny logaritmus alebo faktorizáciu za polynomiálny čas na kvantovom počítači. Príchod kvantových počítačov bude mať pre dnešné používané algoritmy asymetrickej kryptografie založené na faktorizácii alebo diskrétnom logaritme fatálne následky, preto v posledných rokoch výskum smeruje k návrhom nových algoritmov alebo už existujúcich, tak aby tieto algoritmy boli odolné voči útokom na kvantových počítačoch. Medzi takéto algoritmy patria:

- kryptosystémy založené na mrežových bodoch
- kryptosystémy založené na hašovacích funkciách
- kryptosystémy založené na teórií kódovania
- kryptosystém založený na sústave polynómov viacerých neurčitých (MPKC)

[6]

Ďalšou otázkou zostáva či postkvantové kryptografické systémy dosiahnu rovnakú úroveň výkonu ako dnešné algoritmy. Kryptografia sa v dnešnej dobe používa najviac v komunikácií cez internet a je veľmi dôležité aby sa na tieto algoritmy dalo bezpečnostne spoľahnúť, ale aby hlavne boli aj efektívne.

2.1 MPKC

MPKC systémy sú založené na riešení sústavy polynómov s viacerými neurčitými. Nech $n \in \mathbb{N}$, označuje počet neurčitých a $m \in \mathbb{N}$, označuje počet polynómov a $d \in \mathbb{N}$ predstavuje stupeň polynómov nachádzajúcich sa v sústave. Ďalej $x_1, ..., x_n$ budú predstavovať neurčité nad poľom $\mathbb{F}$, pričom $x_0 = 1$. Pre dané $n, d \in \mathbb{N}$ definujeme

$$\mathcal{V}_n^d := \begin{cases} \{0\} & \text{pre } d = 0 \\ \{v \in \{0, ..., n\}^d : i \leq j \Rightarrow v_i \leq v_j\} & \textit{inak} \end{cases}$$

, kde označujeme komponenty vektora v s $v_1, ..., v_d \in \{0, ..., n\}$. $\mathcal{P}$ bude predstavovať systém m polynómov s n neurčitými s maximálnym stupňom $d \in \mathbb{N}$. Každý polynóm je

v tvare:

$$p_i(x_1, ..., x_n) := \sum_{v \in V_n^d} \gamma_{i,v} \prod_{j=1}^{d} x_{v_j} \quad \text{pre} \quad 1 \leq i \leq m \tag{1}$$

s koeficientami $\gamma_{i,v} \in \mathbb{F}$ a vektormi $v \in \mathcal{V}_n^d$. Teraz môžeme definovať problém *Simultánnych rovníc s viacerými premennými(Simultaneous Multivariate Equations (SME))*. Nech $y_1, ..., y_m \in \mathbb{F}$ sú prvky poľa a polynómy $p_1, ..., p_m$ podľa vzťahu č.1. Potom hľadanie riešenia $x \in \mathbb{F}^n$ pre SME v sústave polynómov $\mathcal{P}$, keď je dané $y \in \mathbb{F}^m$ je nazvané ako SME-problém, a je zobrazené na obrázku č.1. [14]

$$\begin{cases} y_1 &= p_1(x_1, \ldots, x_n) \\ y_2 &= p_2(x_1, \ldots, x_n) \\ &\vdots \\ y_m &= p_m(x_1, \ldots, x_n) \end{cases}$$

Obrázok 1: Príklad SME-problému

Ďalej už budeme iba uvažovať MPKC-kryptosystémy s polynómami s $d = 2$, čo je matematický problém riešenia kvadratických polynómov viacerých neurčitých nad konečným poľom. V tejto dobe ešte neexistuje efektívny algoritmus, ktorý by dokázal takúto sústavu polynómov riešiť.[6]

$$\begin{aligned} p^{(1)}(x_1, ..., x_n) &= \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}^{(1)} x_i x_j + \sum_{i=1}^{n} p_i^{(1)} x_i + p_0^{(1)} \\ p^{(2)}(x_1, ..., x_n) &= \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}^{(2)} x_i x_j + \sum_{i=1}^{n} p_i^{(2)} x_i + p_0^{(2)} \\ &\vdots \\ p^{(m)}(x_1, ..., x_n) &= \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}^{(m)} x_i x_j + \sum_{i=1}^{n} p_i^{(m)} x_i + p_0^{(m)} \end{aligned} \tag{2}$$

V prípade, že sú polynómy kvadratické, problém riešenia takejto sústavy sa nazýva *MQ problém*. MQ problém hovorí, že v prípade, keď máme daných m kvadratických polynómov $p^{(1)}(x), ..., p^{(m)}(x)$ s n neurčitými $x_1, x_2, ..., x_n$ podľa predpisu č.2, hľadáme vektor $x = (x_1, ..., x_n)$ taký, že $p^{(1)}(x) = ... = p^{(m)}(x) = 0$. [2]

Je dokázané, že MQ-problém pre $m \approx n$ pre kvadratické polynómy nad poľom $GF(2)$ je NP-úplný, pričom môže byť dokázané, že je to ekvivalent 3-SAT problému.[6]

2.2 Afinné transformácie

Pre jednoduchšie pochopenie celého kryptografického systému založeného na kvadratických polynómoch s viacerými neurčitými je potrebné spomenúť aj afinné transformácie. Afinná transformácia je zobrazenie vektorového priestoru $\mathbb{F}^n$ na vektorový priestor $\mathbb{F}^m$, pričom toto zobrazenie je dané:

- maticou A_S, o veľkosti $m \times n$, pričom platí, že $A_S \in \mathbb{F}^{m \times n}$.
- vektorom b_S, kde $b_S \in \mathbb{F}^m$

Matica aj vektor sú náhodne vygenerované, spolu tvoria afinné zobrazenie vektora x na vektor y také, že $y = A_S x + b_S$, pričom vektory b_S a y majú m prvkov nad poľom $\mathbb{F}$, t.j. $b_S, y \in \mathbb{F}^m$ a x má n prvkov nad poľom $\mathbb{F}$. Ak navyše platí, že transformácia je zobrazenie vektorového priestoru $\mathbb{F}^n$ na ten istý vektorový priestor a matica A_S rozmerov $n \times n$ je invertovateľná, potom je afinná transformácia invertovateľná a platí, že $x = A_S^{-1}(y - b_S)$. Afinná transformácia je následne bijektívne zobrazenie, ktoré dokáže zobraziť vektor $x \rightarrow y$, a jeho inverzné zobrazenie dokáže zobraziť vektor $y \rightarrow x$. [14]

2.3 MQ systémy - Verejný kľúč

V tomto type kryptosystému verejný kľúč tvorí samotná sústava kvadratických polynómov s viacerými neurčitými. Použiť verejný kľúč prakticky znamená dosadiť otvorený text, respektíve hodnoty podpisu za neurčité sústavy polynómov a hodnota sústavy polynómov predstavuje zašifrovaný text, respektíve haš podpisovanej správy. Verejný kľúč P je sústava m kvadratických polynómov s n neurčitými, pričom vznikol ako zloženie zobrazení: S je afinné zobrazenie $S : \mathbb{F}^n \rightarrow \mathbb{F}^n$, T je afinné zobrazenie $T : \mathbb{F}^m \rightarrow \mathbb{F}^m$ a P' je centrálne zobrazenie $P' = \mathbb{F}^n \rightarrow \mathbb{F}^m$, pričom sa jedná o sústavu kvadratických polynómov, pre ktorý je známy algoritmus hľadania jej koreňov. Verejný kľúč P potom vzniká zložením $P = T \circ P' \circ S := \mathbb{F}^n \rightarrow \mathbb{F}^m$. Afinné transformácie pomáhajú zamaskovať pôvodnú štruktúru sústavy kvadratických polynómov tak aby sa z nej stala náhodne vyzerajúca sústava polynómov, ktorú už môžeme použiť ako verejný kľúč. Neurčité takejto sústavy tvoria bity otvoreného textu a pravé strany, teda hodnoty polynómov tvoria bity zašifrovaného textu(v prípade použitia MQ ako šifrovacej schémy). Ak by sme vedeli efektívne riešiť sústavu kvadratických polynómov viacerých neurčitých nad konečným poľom, potom by sme vedeli aj efektívne invertovať takéto kvadratické zobrazenie P. V kvadratickom zo-

brazení P môže existovať aj nepomer medzi počtom polynómov a neurčitých, v takomto prípade sa otvorený text a zašifrovaný text veľkostne líšia. [14]

Systém P má v praxi tvar matice, v ktorej sú uložené koeficienty členov polynómov systému P vo vopred definovanom usporiadaní, takáto matica sa nazýva Macaulayova matica a má nasledovnú formu [6]:

$$M_P = \begin{pmatrix} p_{11}^1\, p_{12}^1 \dots p_{nn}^1 & p_1^1 \dots p_n^1\, p_0^1 \\ p_{11}^2\, p_{12}^2 \dots p_{nn}^2 & p_1^2 \dots p_n^2\, p_0^2 \\ & \vdots \\ p_{11}^m\, p_{12}^m \dots p_{nn}^m & p_1^m \dots p_n^m\, p_0^m \end{pmatrix} \tag{3}$$

V ďalšej sekcii si popíšeme aké tajomstvo musí legitímny príjemca vedieť aby dokázal vyriešiť toto kvadratické zobrazenie P.

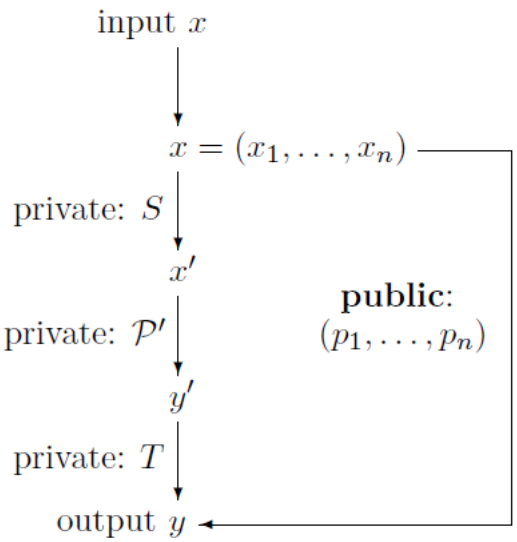
2.4 MQ systémy - Súkromný kľúč

Súkromný kľúč v tomto kryptografickom systéme predstavuje taktiež určitú sústavu kvadratických polynómov. Avšak rozdielom oproti verejnému kľúču je, že táto sústava už nie je náhodná ale je špecificky generovaná, tak aby ju držiteľ súkromného kľúča dokázal riešiť. Majme invertovateľnú afinnú transformáciu $S : \mathbb{F}^n \to \mathbb{F}^n$, invertovateľnú afinnú transformáciu $T : \mathbb{F}^m \to \mathbb{F}^m$ a ľahko invertovateľnú sústavu kvadratických polynómov P', ktorá obsahuje m polynómov s n neurčitými, a každý polynóm má najviac stupeň $d = 2$, pričom môžeme si ju predstaviť aj ako zobrazenie $P' : \mathbb{F}^n \to \mathbb{F}^m$, potom súkromný kľúč je tvorený trojicou (S, P', T). Afinné transformácie budú tvoriť súkromný kľúč spolu s kvadratickým zobrazením P'. Je nevyhnutné aby zobrazenie P' bolo invertovateľné a tým pádom vlastník súkromného kľúča dokáže nájsť riešenie takejto sústavy narozdiel od nelegitímneho príjemcu, ktorý pozná iba verejný kľúč, v ktorom nie sú známe transformácie S, T teda nevie ako vyzerá ľahko-riešiteľná sústava kvadratických rovníc. Na obrázku č.2 je vyobrazená schéma tejto *trapdoorovej funkcie*, kde môžeme vidieť ako funguje súkromný a verejný kľúč v MQ systémoch v prípade šifrovania resp. overenia podpisu(pretože vtedy je x otvorený text na šifrovanie, resp. podpis na overenie). [14]

2.5 Tvorba podpisu

Na vytvorenie podpisu s použitím jednosmernej funkcie, potrebujeme invertovať každý krok tohto algoritmu. Potrebujeme vyrátať vektor $y' = T^{-1}(y)$, pre dané y, ďalej $x' = P'^{-1}(y')$ a nakoniec $x = S^{-1}(x')$.[14]

Ako si môžeme všimnúť z rovníc, je potrebné aby afinné transformácie S a T boli invertovateľné pretože bez toho by sme nedokázali vytvoriť podpis alebo dešifrovať správu.



Obrázok 2: MQ - trapdoor
[14]

Podmienkou použitia určitej transformácie pre tento typ kryptosystému je, že transformácia musí byť invertovateľná teda aj matica, z ktorej je vypočítaná musí byť invertovateľná. Pri P' funkcii je to o niečo zložitejšie. Hovorili sme si, že MPKC kryptosystémy sú založené na ťažkom matematickom probléme, ktorý predpokladá, že je nemožné vyriešiť sústavu kvadratických polynómov s viacerými neurčitými, avšak ako by sme potom mohli takúto sústavu vyriešiť ak je to matematicky nemožné? P' musí byť špeciálne skonštruovaná, pričom bude patriť do triedy ľahko-riešiteľných sústav kvadratických polynómov, ako sú napríklad *UOV, MIA* alebo *Rainbow*. Tieto funkcie nám poskytujú informáciu, s ktorou ľahko vyriešime danú sústavu kvadratických polynómov s viacerými neurčitými.

2.6 Validácia podpisu

Validácia podpisu spočíva vo vyhodnotení polynomiálneho vektoru P s daným podpisom $x \in \mathbb{F}^n$. Ak je výsledok rovnaký ako správa $y \in \mathbb{F}^m$, podpis je validný inak ho odmietame. Pri verifikácii vykonáme tieto kontroly:

$$y_1 \stackrel{?}{=} p_1(x_1, ..., x_n) \ \ldots \ y_m \stackrel{?}{=} p_m(x_1, ..., x_n)$$

Kontrolujeme či zahašovaná správa y je rovnaká ako výsledok funkcie, kde vstupom je podpis, ktorý je dosadený do verejného kľúča, ktorý predstavuje sústava kvadratických polynómov.[14]

2.7 Šifrovanie

Aj keď nebudeme používať pri našej praktickej práci šifrovanie povieme si o ňom pár slov. Šifrovanie si môžeme predstaviť ako zobrazenie $P(x) = y$, pričom x predstavuje otvorený text a y predstavuje zašifrovaný text. Pri šifrovaní sa používa verejný kľúč P, do ktorého je dosadený otvorený text x. Zobrazenie P je vlastne sled zobrazení $T(P'(S(x))) = y$, pričom šifrovateľ nevidí medzivýsledky, len finálny výsledok y. Trojicu *T, P', S* pozná jedine príjemca, pre ktorého je táto trojica súkromným kľúčom. [14]

2.8 Dešifrovanie

Dešifrovanie a tvorba podpisu sú veľmi podobné, avšak dešifrovanie sa líši v tom, že potrebujeme vypočítať všetky možné výsledky $P'(X') = y'$ pre dané $y' \in \mathbb{F}^m$. Potrebujeme si vybrať správne x_i, z množiny všetkých možných riešení. Teda dešifrovanie spočíva v skúšaní rôznych možností, alebo šifrovateľ poskytne dešifrovateľovi informáciu navyše aby mu uľahčil celý proces dešifrovania. Aj pre tento dôvod sa tieto schémy používajú skôr na podpisovanie ako na bežné šifrovanie a dešifrovanie správ.[14]

3 Jednosmerné(trapdoor) funkcie

Hlavným stavebným prvkom MPKC systémov sú jednosmerné alebo *trapdoor* funkcie. Sústavy kvadratických polynómov P' majú špeciálnu konštrukciu, ktorá zaručuje to, že v našom prípade pri tvorení podpisu je sústava ľahko riešiteľná, aj keď na prvý pohľad to nemusí tak vyzerať. Najprv sa pozrieme na jednoduchšie konštrukcie ako *UOV* a *STS* ale spomenieme aj zložitejšie konštrukcie, ktoré používajú rozšírenie nad poľom ako je napríklad *HFE*.

3.1 UOV

Unbalanced oil and vinegar schéma je zovšeobecnením pôvodnej schémy Patarina, ktorý ako prvý vyskúšal balansovanú olejovo-octovú schému. Táto schéma bola neskôr zlomená útokom s využitím lineárnej algebry.[6]

UOV schéma je postavená na jednoduchom konečnom poli $\mathbb{F}$, čo zaručuje pomerne efektívnu implementáciu. Ďalšou vlastnosťou tejto schémy je, že sústava polynómov obsahuje viac neurčitých ako polynómov, čo obmedzuje túto schému na použitie len v podpisových schémach. Definícia hovorí, nech existuje konečné pole $\mathbb{F}$ a $n, m \in \mathbb{N}$, kde $n > m$ a $\alpha'_i, \beta'_{i,j}, \gamma'_{i,j,k} \in \mathbb{F}$ potom polynómy schémy UOV centrálneho zobrazenia P' majú tvar:

$$p_i(x'_1, ..., x'_n) := \sum_{j=1}^{n-m} \sum_{k=1}^{n} \gamma'_{i,j,k} x'_j x'_k + \sum_{j=1}^{n} \beta'_{i,j} x'_j + \alpha'_i \tag{4}$$

Neurčité x'_i pre $1 \leq i \leq n-m$ predstavujú tzv. *octové neurčité* a x'_i pre $n-m < i \leq n$ predstavujú tzv. *olejové neurčité*. Počet olejových neurčitých je $o := m$ a počet octových neurčitých sa označuje ako $v := n - m = n - o$.[14] Pravidlom v tomto systéme je, že v kvadratických termoch sú medzi sebou vynásobené 2 octové neurčité alebo octová a olejová avšak nikdy nie 2 olejové neurčité. Algoritmus riešenia sústavy polynómov v takomto tvare spočíva v priradení náhodných hodnôt do octových neurčitých, po priradení octových neurčitých vzniká lineárna sústava rovníc v olejových neurčitých, ktorá môže byť riešiteľná Gaussovou eliminačnou metódou. Ak sústava nie je riešiteľná zvolíme nové náhodné octové neurčité a opakujeme tento algoritmus kým sústava nie je riešiteľná.[6]

Pri verifikácii podpisu postupujeme nasledovne: $z \in F^n$ predstavuje podpis správy d, správu zahašujeme $w = \mathcal{H}(d)$, pričom $\mathcal{H}$ je hašovacia funkcia. Vypočítame $w' = P(z)$. Ak $w = w'$ potom môžeme konštatovať, že z je skutočne pravý podpis správy d. [6]

3.2 STS

Stepwise Triangular System je ďalší spôsob ako konštruovať invertovateľné centrálne zobrazenie. Táto schéma patrí taktiež do kategórií schém nad jednoduchým konečným poľom $\mathbb{F}$. Centrálne zobrazenie STS schémy tvorí vrstvená sústava polynómov, v ktorej sú polynómy rozdelené do tzv. vrstiev, pričom platí, že polynómy v i-tej vrstve obsahujú vždy aspoň o jednu ďalšiu premennú navyše než polynómy v predchádzajúcich vrstvách. V tejto schéme je počet nových neurčitých a počet nových polynómov určovaný podľa parametra r. V tomto systéme bude počet polynómov predstavovať parameter m, počet neurčitých parameter n a parameter L bude predstavovať počet vrstiev v kvadratickej sústave polynómov.

$$
\begin{array}{ll}
\text{Step 1} & \left\{ \begin{array}{rl} p'_1 & (x'_1, \ldots, x'_r) \\ & \vdots \\ p'_r & (x'_1, \ldots, x'_r) \end{array} \right. \\
\vdots & \\
\text{Step } l & \left\{ \begin{array}{rlll} p'_{(l-1)r+1} & (x'_1, \ldots, x'_r, & \ldots, & x'_{(l-1)r+1}, \ldots, x'_{lr}) \\ & & \vdots & \\ p'_{lr} & (x'_1, \ldots, x'_r, & \ldots, & x'_{(l-1)r+1}, \ldots, x'_{lr}) \end{array} \right. \\
\vdots & \\
\text{Step } L & \left\{ \begin{array}{rlllll} p'_{(L-1)r+1} & (x'_1, \ldots, x'_r, & \ldots, & x'_{(l-1)r+1}, \ldots, x'_{lr}, & \ldots, & x'_{n-r+1}, \ldots, x'_n) \\ & & \vdots & & \vdots & \\ p'_{Lr} & (x'_1, \ldots, x'_r, & \ldots, & x'_{(l-1)r+1}, \ldots, x'_{lr}, & \ldots, & x'_{n-r+1}, \ldots, x'_n) \end{array} \right.
\end{array}
$$

Obrázok 3: Centrálne zobrazenie STS schémy

Schéma tohto systému je zobrazená na obrázku č.3.[12] Nech $r_1, ..., r_L$ je počet neurčitých v jednotlivých vrstvách potom bude platiť, že $r_1 + ... + r_L = n$, kde n sú všetky neurčité sústavy a $m_1, ..., m_L \in \mathbb{N}$ je počet polynómov v jednotlivých vrstvách, platí, že $m_1 + ... + m_L = m$, budú všetky polynómy takejto sústavy. Každá vrstva obsahuje všetky neurčité z predchádzajúcej vrstvy a tie, ktoré práve pribudli. Táto schéma dostala názov podľa tvaru aký tieto kvadratické polynómy vytvárajú.[14]

Invertovanie sústavy polynómov v takomto tvare prebieha v nasledujúcich krokoch:

1. Nájde sa riešenie len pre prvú vrstvu podľa polynómov v prvej vrstve.

2. Následne sa hľadajú riešenie polynómov z druhej vrstvy, pričom do neurčitých z prvej vrstvy sa doplnia už vyriešené hodnoty z predchádzajúceho kroku.

3. Tento postup sa opakuje, dosadzujeme neurčité z predchádzajúcich vrstiev a počítame neurčité len tej vrstvy, ktorú práve riešime až kým nedorazíme na poslednú vrstvu kvadratickej sústavy polynómov.

Neriešime celú sústavu naraz, ale hľadáme riešenia len jednotlivých blokov. Dôležitá úloha je na začiatku algortimu, kde musíme vyriešiť sústavu kvadratických polynómov, pričom výhodou je, že táto sústava neobsahuje všetky neurčité systému. Pri tejto schéme využívame skutočnosť, že rovnice na seba nadväzujú a preto nemusíme riešiť sústavu ako celok, ale môžeme si ju rozdeliť do menších častí.

3.3 HFE

Hidden Field Equations je kryptografická schéma, ktorá využíva polynómy s jednou neurčitou. Bola prvýkrát predstavená Patarinom v roku 1996. Hlavnou myšlienkou tejto schémy je pridanie ďalších termov do *Matsumo-Imai* centrálneho zobrazenia, pričom verejný kľúč zostane kvadratický a centrálne zobrazenie zostane invertovateľné. Samotné HFE nie je odolné voči niektorým útokom avšak v dnešnej dobe slúži najmä ako základ pre náročnejšie schémy.[6]

Základnou ideou tejto schémy je použitie polynómu v nadpoli, čo predstavuje súkromný kľúč a verejný kľúč predstavuje sústava polynómov nad základným konečným poľom.[13] Nech $\mathbb{F}$ je konečné pole, ktoré má $q = |\mathbb{F}|$ elementov, $\mathbb{E}$ bude predstavovať rozšírenie(nadpole) n-tého stupňa poľa $\mathbb{F}$ a $\phi = \mathbb{E} \rightarrow \mathbb{F}^n$ bude predstavovať bijekciu tohto nadpoľa do určeného vektorového priestoru. Nech $P'(X')$ bude jednorozmerný polynóm v nadpoli $\mathbb{E}$ taký, že:

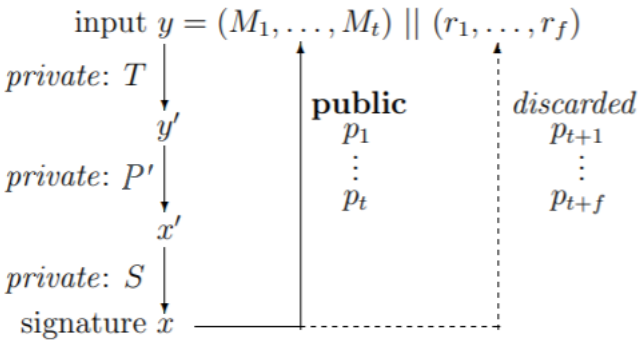
$$P'(X') := \sum_{\substack{0 \leq i,j \leq d \\ q^i + q^j \leq d}} C'_{i,j} X'^{q^i+q^j} + \sum_{\substack{0 \leq k \leq d \\ q^k \leq d}} B'_k X'^{q^k} + A' \tag{5}$$

kde $C'_{i,j}X'^{q^i+q^j}$ pre $C'_{i,j} \in \mathbb{E}$ sú kvadratické termy, $B'_k X'^{q^k}$ pre $B'_k \in \mathbb{E}$ sú lineárne termy a A' pre $A' \in \mathbb{E}$ sú konštantné termy, s $i, j \in \mathbb{N}$ a stupňom $d \in \mathbb{N}$. Platí, že $\mathcal{P}' := \phi \circ P' \circ \phi^{-1}$ sú v HFE tvare a predstavujú kvadratické zobrazenie nad priestorom $\mathbb{F}^n$. Tento algoritmus závisí na veľkosti dimenzie n nadpoľa $\mathbb{E}$ a stupňa d polynómu P', pričom z hľadiska efektívnosti je dôležité aby tieto konštanty boli malé.[14]

Paramater d zaisťuje efektívnu inverziu centrálneho zobrazenia P'. Inverzia zobrazenia P' je vlastne hľadanie koreňov polynómu stupňa d, preto zložitosť celej operácie závisí hlavne na stupni d. Pri riešení tejto inverzie sa používa Berlekampov alebo Cantor-Zassenhaus faktorizačný algoritmus, zložitosť oboch týchto algoritmov je $O(d^3)$. [6]

Na obrázku č. 4 je znázornená schéma tvorby podpisu HFE, pričom vstup je označený ako y a tvorí ho zjednotenie $(M_1, ..., M_t) \in \mathbb{F}^t$, čo predstavuje správu, ktorú sa

chystáme podpísať a $(r_1, ..., r_f) \in \mathbb{F}^f$ je náhodne vybraný vektor. Výstup je označený ako x. Ak poznáme súkromný kľúč $k = (S, P', T)$ problém nájdenia riešenia x pre dané



Obrázok 4: HFE - tvorba podpisu
[13]

y, je redukovaný na hľadanie riešenia $P'(x') = y'$, kde P' je polynóm stupňa d. Keďže zobrazenie P' vo všeobecnosti nemusí byť surjektívne, môže sa stať, že sa pri jeho inverzii nenájde riešenie x' pre y'. V takom prípade sa pre správu $(M_1, M_2, ..., M_t)$ zvolia iné náhodné náhodné hodnoty $(r_1, r_2, ..., r_f)$ a postup sa opakuje.[13]

3.4 Rainbow

V tomto zobrazení sa autori [5] snažia zovšeobecniť UOV konštrukciu verejného kľúča. Myšlienkou *Rainbow* zobrazenia je viacvrstvový UOV systém. Používa sa teda kombinácia STS a UOV prístupu. Môžeme povedať, že *Rainbow* systém používa konštrukciu STS s UOV jednosmernou funkciou. Pri tejto schéme sa používa nebalansovaná OV schéma, pretože bolo ukázané, že balansované OV nie je dostatočne bezpečné.[6] Pre nebalansované OV je potrebné aby parameter o, nebol príliš veľký (< 100) potom $v - o$ by malo byť dostatočne veľké na zaistenie bezpečnosti tohto systému.[5] Je potrebné ďalej poznamenať, že dokument, ktorý je v UOV schéme podpisovaný je vektor o veľkosti k^{o+v}, kde k je počet prvkov poľa $\mathbb{F}$. Keďže podpisovaný dokument predstavuje vektor z vektorového priestoru k^o a výsledný podpis vektor k^{o+v} vidíme, že podpis je najmenej dvakrát taký veľký ako podpisovaný dokument. S pribúdajúcou veľkosťou parametrov $v+o$ sa stáva systém menej efektívny. Práve konštrukciou, ktorá UOV rozkladá do viacerých vrstiev je v konečnom dôsledku podpis len o niečo väčší ako je veľkosť podpisovaného dokumentu. Keďže sa

táto schéma skladá z vrstiev autori videli asociáciu s dúhou aj práve preto je táto schéma nazvaná ako *Rainbow*.[5]

Rainbow podpisová schéma je jedna z najviac študovaných podpisových schém čo sa týka kvadratických polynómov s viacerými neurčitými. Nech $\mathbb{F} = \mathbb{F}_q$ je konečné pole s q elementami, $n \in \mathbb{N}$ a $0 < v_1 < v_2 < ... < v_l < v_{l+1} = n$ je sekvencia celých čísel. Nech $m = n - v_1$, $O_i = \{v_i + 1, ..., v_{i+1}\}$ a $V_i = \{1, ..., v_i\}(1 = 1, ..., l)$. Práve tieto hodnoty nám určujú ako sú rozdelené octové a olejové neurčité. Množina V_i predstavuje octové neurčité v i-tej vrstve a množiny O_i predstavujú indexy olejových neurčitých v i-tej vrstve. Ďalšou vlastnosťou tohto systému je, že v každej vrstve je toľko polynómov, koľko má daná vrstva olejových neurčitých. Všetkých polynómov je v zobrazení $|O_1| + |O_2| + ... + |O_l| = (v_{l+1} - v_1) = n - v_1$, potom parameter m udávajúci počet polynómov je $m = n - v_1$. Zobrazenie pozostáva z l vrstiev, každá vrstva obsahuje iné octové a olejové neurčité. V prvej vrste sa nachádzajú na začiatku určené octové a olejové neurčité. V druhej vrstve sú octové neurčité všetky neurčité z predchádzajúcej vrsty a olejové sú tie, ktoré v tejto vrstve pribudli.[8]

3.4.1 Generovanie kľúča

Podľa článku [8] súkromný kľúč sa skladá z dvoch invertovateľných afinných zobrazení $S : \mathbb{F}^m \to \mathbb{F}^m$ a $T : \mathbb{F}^n \to \mathbb{F}^n$, ďalej nech je centrálne zobrazenie označené ako $F(x) = (f^{(v_1+1)}(x), ..., f^{(n)}(x)) : \mathbb{F}^n \to \mathbb{F}^m$. Polynómy $f^{(i)}(i = v_1 + 1, ..., n)$ sú v tvare s koeficientami náhodne vybraných z $\mathbb{F}$

$$f^{(i)} = \sum_{k,l \in V_j} \alpha_{k,l}^{(i)} \cdot x_k \cdot x_l + \sum_{k \in V_j, l \in O_j} \beta_{k,l}^{(i)} \cdot x_k \cdot x_l + \sum_{k \in V_j \cup O_j} \gamma_k^{(i)} \cdot x_k + \eta^{(i)} \tag{6}$$

,kde polynóm $f^{(i)}$ patrí do j-tej vrstvy. Verejný kľúč predstavuje zložené zobrazenie $P = S \circ F \circ T : F^n \to F^m$. Z rovnice č.6 jednotlivé členy predstavujú:

- Kvadratické = Kombinácia $x_k x_l$, kde x_k a x_l sú octové neučité. Toto sú členy s koeficientom $\alpha_{k,l}$
- Kvadratické = Kombinácia $x_k x_l$, kde x_k sú octové neurčité a x_l sú olejové neučité. Toto sú členy s koeficientom $\beta_{k,l}$
- Lineárne = V tvare x_k, kde x_k sú olejové alebo octové neurčité. Členy s koeficientom γ_k
- Absolútne = Koeficient η

3.4.2 Generovanie podpisu

Pri generovaní podpisu dokumentu d, sa použije hašovacia funkcia $\mathcal{H} : \{0,1\}^* \to \mathbb{F}^m$, s ktorou sa následne dokument zahašuje $w = \mathcal{H}(d) \in \mathbb{F}^m$ a vypočítajú sa inverzné hodnoty $x = S^{-1}(w)$, $y = P^{-1}(x)$, $z = T^{-1}(y)$. $P^{-1}(x)$ znamená nájsť riešenie sústavy kvadratických polynómov. Inými slovami hľadáme také x aby platilo $P(x) = y$, ak poznáme y. Keďže táto sústava je špeciálne skonštruovaná dokážeme efektívne invertovať zobrazenie P, využijeme delenie na vrstvy a UOV vlastnosti polynómov. Náhodne zvolíme hodnoty octových neurčitých v prvej vrstve, po ich dosadení sa zmenia polynómy v prvej vrstve na lineárne a tie riešime napríkad Gaussovou eliminációu. Po ich nájdení dosadíme všetky hodnoty neurčitých do druhej vrstvy a postup opakujeme. Môže sa stať, že po dosadení do druhej vrstvy nenájdeme riešenie sústavy, v takomto prípade sa musíme vrátiť na začiatok a vybrať si iné hodnoty octových neurčitých v prvej vrstve. Podpis dokumentu d je $z \in \mathbb{F}^n$.

3.4.3 Verifikácia podpisu

Pri verifikácii podpisu z dokumentu d je naprv potrebné skontrolovať či $z \in \mathbb{F}^n$, následne sa vypočíta haš z dokumentu d, $w = \mathcal{H}(d)$ a $w' = P(z) \in \mathbb{F}^m$. Ak $w' = w$ potom je podpis validný inak ho zamietame.

3.4.4 Praktický príklad Rainbow trapdoor funkcie

Uvažujeme konečné pole $GF(3)$. Nech tento systém obsahuje 2 vrstvy, $l = 2$. Nech $v_1 = 2, v_2 = 4, v_3 = 6$. Z toho vyplýva, že:

- Budeme mať 6 neurčitých $(x_1, x_2, x_3, x_4, x_5, x_6)$
- $V_1 = \{1, 2\}$, $O_1 = \{3, 4\}$
- $V_2 = \{1, 2, 3, 4\}$, $O_2 = \{5, 6\}$
- Budeme mať 4 polynómy:
  - V prvej vrstve sú 2 polynómy $f^{(3)}, f^{(4)}$, v ktorých sú octové neurčité x_1, x_2 a olejové x_3,x_4
  - V druhej vrstve sú 2 polynómy $f^{(5)}, f^{(6)}$, v ktorých sú octové neurčité x_1,x_2,x_3,x_4 a olejové x_5,x_6

V ďalšom kroku je potrebné polynómy náhodne vygenerovať tak aby spĺňali naše podmienky a aby dodržiavali vlastnosti UOV polynómov. Môžeme uvažovať následné

polynómy v prvej vrstve:

$$f^{(3)} = x_1^2 + 2x_1x_2 + 2x_1x_3 + x_1x_4 + 2x_2x_4 + x_1 + x_3 + 2x_4 + 1$$
$$f^{(4)} = 2x_1^2 + 2x_2^2 + x_1x_4 + x_2x_3 + 2x_2x_4 + 2x_1 + 2x_2 + x_4 + 2$$

V druhej vrstve môžeme uvažovať tieto náhodne vygenerované polynómy:

$f^{(5)} = x_1^2 + x_1x_2 + x_1x_3 + 2x_1x_4 + 2x_2x_4 + x_3^2 + x_4^2 + x_1x_5 + 2x_2x_6 + 2x_3x_6 + x_4x_5 + x_1 +$
$x_4 + x_6 + 2$
$f^{(6)} = 2x_1^2 + 2x_2^2 + x_3x_4 + x_4^2 + x_1x_5 + 2x_1x_6 + 2x_3x_5 + x_3x_6 + x_4x_6 + x_1 + x_5 + x_6$

Centrálne zobrazenie P je tvorené 4 polynómami so 6 neučitými a to:

$f^{(3)} = x_1^2 + 2x_1x_2 + 2x_1x_3 + x_1x_4 + 2x_2x_4 + x_1 + x_3 + 2x_4 + 1$
$f^{(4)} = 2x_1^2 + 2x_2^2 + x_1x_4 + x_2x_3 + 2x_2x_4 + 2x_1 + 2x_2 + x_4 + 2$
$f^{(5)} = x_1^2 + x_1x_2 + x_1x_3 + 2x_1x_4 + 2x_2x_4 + x_3^2 + x_4^2 + x_1x_5 + 2x_2x_6 + 2x_3x_6 + x_4x_5 + x_1 +$
$x_4 + x_6 + 2$
$f^{(6)} = 2x_1^2 + 2x_2^2 + x_3x_4 + x_4^2 + x_1x_5 + 2x_1x_6 + 2x_3x_5 + x_3x_6 + x_4x_6 + x_1 + x_5 + x_6$

Ďalší krok je invertovanie tohto zobrazenia. Nech hodnoty polynómov sú napríklad $(0, 2, 1, 1)$, $f^{(3)} = 0, f^{(4)} = 2, f^{(5)} = 1, f^{(6)} = 1$. Najprv riešime prvú vrstvu sústavy, v našom prípade prvé dve rovnice:

$$x_1^2 + 2x_1x_2 + 2x_1x_3 + x_1x_4 + 2x_2x_4 + x_1 + x_3 + 2x_4 + 1 = 0$$
$$2x_1^2 + 2x_2^2 + x_1x_4 + x_2x_3 + 2x_2x_4 + 2x_1 + 2x_2 + x_4 + 2 = 2$$

Pri riešení náhodne zvolíme octové neurčité (x_1, x_2), dosadíme do prvej vrstvy a snažíme sa vyrátať olejové neurčité (x_3, x_4).

$$x_3 + x_4 + 1 = 0$$
$$x_3 = 2$$

Dostávame hodnoty olejových neurčitých v prvej vrstve $x_3 = 2, x_4 = 0$. Octové a olejové neurčité majú teda hodnoty $(x_1, x_2, x_3, x_4) = (0, 1, 2, 0)$. Následne riešime druhú vrstvu:

$x_1^2 + x_1x_2 + x_1x_3 + 2x_1x_4 + 2x_2x_4 + x_3^2 + x_4^2 + x_1x_5 + 2x_2x_6 + 2x_3x_6 + x_4x_5 + x_1 +$
$x_4 + x_6 + 2 = 1$
$2x_1^2 + 2x_2^2 + x_3x_4 + x_4^2 + x_1x_5 + 2x_1x_6 + 2x_3x_5 + x_3x_6 + x_4x_6 + x_1 + x_5 + x_6 = 1$

x_1, x_2, x_3, x_4 už poznáme, doplníme vyriešené hodnoty z prvej vrstvy a v tejto druhej vrstve dostávame 2 lineárne rovnice o dvoch neznámych:

$$x_6 + 2 = 1$$
$$2x_5 + 2 = 1$$

Dostávame hodnoty olejových neurčitých v druhej vrstve $x_5 = 1, x_6 = 2$. Vypočítali sme hodnoty všetkých neurčitých $x_1 = 0, x_2 = 1, x_3 = 2, x_4 = 0, x_5 = 1, x_6 = 2$ Ak sa na sústavu polynómov pozeráme ako na zobrazenie $GF(3)^6 \rightarrow GF(3)^4$, potom $F(0,1,2,0,1,2) = (0,2,1,1)$, to isté platí aj pre inverzné zobrazenie $F^{-1}(0,2,1,1) = (0,1,2,0,1,2)$.

V tomto príklade je ukázaný algoritmus výpočtu inverzie centrálneho zobrazenia avšak oproti skutočnému verejnému kľúču nám ešte chýbajú afinné transformácie, ktoré celý výsledok zamaskujú.

3.5 GeMSS

Great Multivariate Short Signature je schéma postavená na modifikovanom HFE kryptosystéme s tzv. mínusovými a octovými modifikátormi *HFEv-*. Ako už názov napovedá táto schéma vytvára podpisy malej dĺžky, má rýchly verifikačný proces a stredne veľké veľkosti verejného kľúča. [4] HFEv- využíva stupeň n nadpoľa $\mathbb{E}$ z poľa $\mathbb{F}$ a izomorfizmus $\phi : \mathbb{F}^n \rightarrow \mathbb{E}$. Oproti základnému HFE, HFEv- využíva ďalšie dva parametre a to:

- a: počet mínusových polynómov
- v: počet octových neurčitých

HFE- sa líši od bežného HFE tým, že a polynómov z verejného kľúča nie je známych. Ponechanie niektorých polynómov tajných z verejného kľúča zvyšuje celkovú bezpečnosť systému. Avšak veľké množstvo týchto polynómov nemôže byť odstránené. Utajiť jeden polynóm znamená vziať $log_2 q$ bitov z informácie $M' := HFE(M)$. Pre korektné invertovanie takéhoto zobrazenia HFE- je potrebné vyskúšať všetky verzie zašifrovanej správy s chýbajúcimi q zložkami v prípade, že uvažujeme HFE- ako šifrovaciu schému.[13]

HFEv mení štruktúru polynómov súkromného kľúča. Namiesto toho aby sa použil jeden súkromný polynóm P, táto schéma nám dovoľuje použiť q^v súkromných polynómov $P_1, ..., P_{q^v}$, kde $v \in \mathbb{N}$, predstavuje počet octových neurčitých $z_1, ..., z_v$. Octové neurčité $(z_1, ..., z_v) \in \mathbb{F}^v$ sú inicializované náhodnými hodnotami. Pri šifrovacom procese nie je jednoduché použiť túto octovú modifikáciu, pretože každá z q^v možností výberu octových neurčitých je rovnako pravdepodobná. Avšak oproti HFE- v tomto prípade sa zmení celý

polynóm a tým pádom pre podpisové schémy možeme použiť aj väčšie množstvo octových neurčitých.[13]

3.5.1 Generovanie kľúčov

Centrálne zobrazenie $\mathcal{F} : \mathbb{E} \times \mathbb{F}^v \to \mathbb{E}$ má pri schéme HFEv- nasledovnú formu:

$$\mathcal{F}(X, x_1, ..., x_v) = \sum_{i,j\geq 0}^{q^i+q^j\leq D} \alpha_{i,j} X^{q^i+q^j} + \sum_{i\geq 0}^{q^i\leq D} \beta_i(x_i, ..., x_v) X^{q^i} + \gamma(x_i, ..., x_v) \tag{7}$$

Koeficienty $\alpha_{i,j}$ sú vybrané náhodne z poľa $\mathbb{E}$, ďalej $\beta_i : \mathbb{F}^v \to \mathbb{E}$ a $\gamma : \mathbb{F}^v \to \mathbb{E}$ sú lineárne respektíve kvadratické zobrazenia. Kvôli utajeniu štruktúry centrálneho zobrazniea $\mathcal{F}$ vo verejnom kľúči sú použité 2 lineárne alebo afinné zobrazenia $S : \mathbb{F}^n \to \mathbb{F}^{n-a}$ a $T : \mathbb{F}^{n+v} \to \mathbb{F}^{n+v}$.

Verejný kľúč predstavuje kvadratické zobraznie $\mathcal{P} := \mathbb{F}^{n+v} \to \mathbb{F}^{n-a}$. Súkromný kľúč pozostáva z 3 zobrazení a to $S, \mathcal{F}, T$. Keďže verejný kľúč HFEv- obsahuje viac neurčitých ako polynómov, tým pádom táto schéma môže byť použitá iba ako podpisová schéma. [6]

3.5.2 Generovanie podpisu

Pre vygenerovanie podpisu pre správu $d \in \{0,1\}^*$, použijeme hašovaciu funkciu $\mathcal{H} : \{0,1\}^* \to \mathbb{F}^{n-a}$ na výpočet hašu správy $w = \mathcal{H}(d) \in \mathbb{F}^{n-a}$ a následne:

- Nájdeme vzor $x \in \mathbb{F}^n$ zahašovaného dokumentu w pod afinnou transformáciou S a výsledok prenesieme do nadpoľa $\mathbb{E}$. Výsledok tejto operácie označíme ako X.
- Vyberieme si náhodne hodnoty octových neurčitých $x_1, ..., x_v$ a dosadíme ich do centrálneho zobrazenia $\mathcal{F}$ aby sme dostali parametrizované zobrazenie $\mathcal{F}_v : \mathbb{E} \to \mathbb{E}$.
- Nájdeme koreň polynómu s jednou neurčitou $\mathcal{F}_v(\hat{Y}) = X$ pomocou Berlekampovho algoritmu. Koreň označíme ako $Y \in \mathbb{E}$. Ak nenájdeme riešenie musíme si zvoliť nové octové neurčité a zopakovať tento krok.
- Vypočítame $y' = \phi^{-1}(Y) \in \mathbb{F}^n$, doplníme octové neurčité $x_1, ..., x_v$ a dostaneme sústavu $y = (y' \parallel x_1 \parallel \cdots \parallel x_v) \in \mathbb{F}^{n+v}$ a vypočítame podpis správy $z \in \mathbb{F}^{n+v}$ ako $z = T^{-1}(y)$.

[6]

3.5.3 Overenie podpisu

Na overenie pravosti podpisu $z \in \mathbb{F}^{n+v}$, najprv overovateľ použije hašovaciu funkciu na vygenerovanie $w = \mathcal{H}(d) \in \mathbb{F}^{n-a}$ a následne použije verejný kľúč na výpočet

$$w' = \mathcal{P}(z) \in \mathbb{F}^{n-a} \tag{8}$$

Ak $w' = w$ potom overovateľ podpis akceptuje avšak ak sa tieto dve hodnoty nerovnajú potom podpis zamieta. [6]

3.5.4 Praktický príklad HFEv- trapdoor funkcie

Nasledujúci príklad riešenia HFEv- kryptosystému sme prevzali z knihy Multivariate Public Key Cryptography [6]. Pri tomto príklade použijeme pole $GF(4)$ so 4 elementami a parametre $(n, D, a, v) = (4, 17, 1, 1)$. Ireducibilný polynóm, ktorý použijeme na vygenerovanie nadpoľa bude $f(x) = X^4 + X^2 + \alpha X + 1$. Vyberieme si afinné zobrazenia $S : \mathbb{F}^4 \to \mathbb{F}^3$ a $T : \mathbb{F}^5 \to \mathbb{F}^5$. Centrálne zobrazenie HFEv- $\mathcal{F} : \mathbb{E} \times \mathbb{F} \to \mathbb{E}$ je dané:

$$\begin{aligned}\mathcal{F}(\hat{X}, x_5) = \beta_{17}\hat{X}^{4^2+4^0} + \beta_8\hat{X}^{4^1+4^1} + \beta_5\hat{X}^{4^1+4^0} + \beta_2\hat{X}^{4^0+4^0} \\ +\gamma 16(x_5)\hat{X}^{4^2} + \gamma 4(x_5)\hat{X}^{4^1} + \gamma 1(x_5)\hat{X}^{4^0} + \delta(x_5)\end{aligned}$$

$$\begin{aligned}\beta_{17} &= X^3 + \alpha X^2 + \alpha^2 X + \alpha^2 \\ \beta_8 &= \alpha X^3 + \alpha^2 X^2 + \alpha^2 X + \alpha^2 \\ \beta_5 &= X^3 + \alpha^2 X^2 + \alpha X + \alpha \\ \beta_2 &= \alpha X^2 + X \\ \gamma_{16}(x_5) &= \alpha^2 X^2 + (\alpha x_5 + \alpha^2)X^2 + (\alpha x_5 + \alpha)X + \alpha x_5 \\ \gamma_4(x_5) &= (\alpha^2 x_5 + \alpha^2)X^3 + (\alpha x_5 + \alpha^2)X^2 + (\alpha^2 x_5 + 1)X + x_5 \\ \gamma_1(x_5) &= (x_5 + \alpha^2)X^3 + \alpha x_5 X + \alpha x_5 + 1 \\ \delta(x_5) &= \alpha^2 X^3 + (\alpha^2 x_5^2 + 1)X^2 + (\alpha^2 x_5 + 1)X + \alpha x_5 + \alpha^2\end{aligned}$$

Verejný kľúč je zložený z troch polynómov $P = (p^{(1)}, p^{(2)}, p^{(3)})$ a to nasledovne

$$\begin{aligned}p^{(1)} &= \alpha^2 x_1^2 + x_1 x_3 + \alpha^2 x_1 x_4 + \alpha x_1 x_5 + x_2^2 + \alpha x_2 x_3 + \alpha x_2 x_4 + \alpha x_2 \\ &+ \alpha^2 x_3^2 + x_3 x_4 + \alpha^2 x_3 x_5 + x_3 + x_4^2 + x_4 x_5 + \alpha^2 x_4 + \alpha x_5^2 + \alpha^2 x_5 \\ p^{(2)} &= \alpha x_1^2 + x_1 x_3 + \alpha^2 x_1 + x_2^2 + x_2 x_3 + x_2 x_5 + \alpha^2 x_2 + \alpha^2 x_3^2 + \alpha^2 x_3 x_4 \\ &+ x_3 x_5 + \alpha^2 x_4 x_5 + \alpha x_4 + x_5 + \alpha \\ p^{(3)} &= x_1^2 + \alpha x_1 x_2 + \alpha x_1 x_3 + \alpha^2 x_1 + \alpha^2 x_2 x_3 + x_2 x_5 + \alpha^2 x_2 + \alpha x_3^2 \\ &+ \alpha x_3 + \alpha^2 x_4^2 + x_4 x_5 + x_4 + \alpha^2 x_5 + \alpha\end{aligned}$$

Generujeme podpis pre hodnotu $w = (0, 0, \alpha^2) \in \mathbb{F}^3$. Aplikovaním inverznej afinnej transformácie S dostávame

$$x = (\alpha^2, 0, 0, \alpha) \in \mathbb{F}^4.$$

Ďalej výsledok prenesieme do nadpoľa $\mathbb{E}$

$$\hat{X} = \alpha X^3 + \alpha^2.$$

Ďalej si zvolíme hodnotu octovej neurčitej $x_5 = 0$. x_5 následne doplníme do centrálneho zobrazenia, do koeficientov γ_i a δ, dostávame

$$\gamma_{16} = \alpha^2 X^3 + \alpha^2 X^2 + \alpha X$$
$$\gamma_4 = \alpha^2 X^3 + \alpha^2 X^2 + X$$
$$\gamma_1 = \alpha^2 X^3 + 1$$
$$\delta = \alpha^2 X^3 + X^2 + X + \alpha^2$$

Snažíme sa vyriešiť rovnicu $\mathcal{F}_0(Y) = \hat{X}$ použitím Berlekampovho algoritmu. Ak by rovnica nemala riešenie museli by sme vybrať inú octovú neurčitú x_5. Táto rovnica má nasledovné riešenie

$$Y = \alpha X^3 + X^2 + \alpha X.$$

Y prenesieme do priestoru $\mathbb{F}^4$ a pridaním octovej neurčitej a aplikovaním afinnej transformácie T nám vzniká podpis

$$z = (\alpha, \alpha^2, \alpha^2, 1, 0).$$

Pri overení použitím verejného kľúča, kde doplníme podpis z dostávame našu pôvodnú zahašovanú správu

$$w = \mathcal{P}(z) = (0, 0, \alpha^2).$$

4 Prstencové podpisové schémy

Prstencové podpisové schémy (Ring signature schemes) dovoľujú účastníkovi, ktorý chce podpísať určitú správu deklarovať ľubovoľnú množinu možných signatárov vrátane seba samého a vygenerovať podpis pomocou svojho súkromného kľúča a verejných kľúčov ostatných účastníkov prstenca $\mathcal{R} = \{u_1, ..., u_k\}$. Príjemca podpísanej správy môže skontrolovať či správa bola podpísaná členom prstenca, avšak nemôže odhaliť konkrétnu identitu signatára. Signatár nepotrebuje súhlas alebo asistenciu ostatných členov prstenca, potrebuje iba poznať ich verejné kľúče.

4.1 Všeobecný popis prstencových skupinových schém

Prstencové schémy sú zjednodušené skupinové schémy, ktoré nemajú manažérov skupiny, ale iba účastníkov. Pri skupinových podpisových schémach manažér skupiny dokáže spojiť skupinový podpis so signatárom, ktorý vygeneroval podpis v mene skupiny. Naopak pri prstencových podpisových schémach je signatárom zaručená úplná anonymita. Skupinové schémy sa používajú ak skupina užívateľov chce spolupracovať a prstencové podpisové schémy sa využívajú ak skupina nechce spolupracovať.[9]

Formálne môžeme definovať prstencovú podpisovú schému nasledovne. Nech $\mathcal{R} = \{u_1, ..., u_k\}$ je skupina alebo prstenec používateľov. Prstencová podpisová schéma pozostáva z troch algoritmov *KeyGen*, *RingSign* a *Verify*:

- *KeyGen*(1^λ): Pravdepodobnostný algoritmus *KeyGen* berie na vstup bezpečnostný parameter λ a výstupom takejto funkcie je pár kľúčov (sk, pk) poskytujúci λ bitovú bezpečnosť. V prstencovej podpisovej schéme vykoná tento algoritmus každý účastník skupiny.

- *RingSign*$(d, sk_i, \{pk_1, ..., pk_k\})$: Do algoritmu *RingSign* vstupuje správa d, ktorá má byť podpísaná, tajný kľúč užívateľa sk_i, ktorý sa chystá správu podpísať a zoznam verejných kľúčov všetkých účastníkov $\{pk_1, ..., pk_k\}$ skupiny $\mathcal{R}$. Výstupom algoritmu je skupinový podpis σ správy d v mene skupiny $\mathcal{R}$.

- *Verify*$(d, \sigma, \{pk_1, ..., pk_k\})$: Deterministický algoritmus, do ktorého vstupuje pár správy a vygenerovaného podpisu d, σ a zoznam verejných kľúčov $\{pk_1, ..., pk_k\}$ skupiny $\mathcal{R}$. Výstupom je booleovská hodnota **TRUE** ak podpis σ správy d je správny a **FALSE** ak je to inak.

Základné bezpečnostné kritéria, ktoré prstencové podpisové schémy spĺňajú sú zaistenie *anonymity* a *nefalšovateľnosť*. Podmienka anonymity hovorí o tom, že útočník ne-

dokáže povedať, ktorý člen skupiny vygeneroval daný podpis. Nefalšovateľnosť deklaruje skutočnosť, že útočník, ktorý nepatrí do skupiny nie je schopný sfalšovať pravý skupinový podpis.[8]

Definícia anonymity: Daná je prstencová podpisová schéma $(Gen, RingSign, Verify)$ a útočník $\mathcal{A}$ útočiaci s algoritmami v polynomiálnom čase, potom uvažujeme nasledujúcu hru:

1. Kľúčové páry(sk_i, pk_i) sú generované algoritmom $Gen(1^\lambda)$ a zoznam verejných kľúčov$\{pk_1, ..., pk_k\}$ je daný útočníkovi $\mathcal{A}$.

2. Útočníkovi $\mathcal{A}$ je daný prístup k podpisovaciemu orákulu, ktoré vracia validný podpis σ v mene skupiny $\mathcal{R} = \{u_1, ..., u_k\}$.

3. $\mathcal{A}$ vygeneruje výstupnú správu d^* a dva odlišné indexy i_0 a $i_i \in \{1, ..., k\}$. Následne $\mathcal{A}$ dostane podpis $\sigma \leftarrow RingSign(d^*, sk_{i_b}, \{pk_1, ..., pk_k\})$, kde b je náhodne zvolené z $\{0, 1\}$.

4. Útočník $\mathcal{A}$ určuje bit b', vyhráva ak $b = b'$.

Skupinová podpisová schéma poskytuje anonymitu ak výhoda útočníka

$$Adv_{\mathcal{A}} = 2 \cdot Pr[b' = b] - 1$$

je pre každého útočníka v polynomiálnom čase zanedbateľná.[1]

Definícia nefalšovateľnosti: Daná je prstencová podpisová schéma *(Gen, RingSign, Verify)* a útočník $\mathcal{A}$ útočiaci s algoritmami v polynómialnom čase, potom uvažujeme nasledujúcu hru:

1. Kľúčové páry(sk_i, pk_i) sú generované algoritmom $Gen(1^\lambda)$ a zoznam verejných kľúčov$\{pk_1, ..., pk_k\}$ je daný útočníkovi $\mathcal{A}$.

2. Útočníkovi $\mathcal{A}$ je daný prístup k podpisovaciemu orákulu, ktoré vracia validný podpis σ pre správu d v mene skupiny $\mathcal{R} = \{u_1, ..., u_k\}$.

3. Útočníkovi $\mathcal{A}$ je daná vyzývacia správa d^*. Je víťazom hry ak dokáže vytvoriť validný prstencový podpis σ^* pre d^* v mene skupiny $\mathcal{R}$.

Skupinová podpisová schéma poskytuje nemožnosť falšovania ak pravdepodobnosť úspechu

$$Pr_{\mathcal{A}}[succ] = Pr[Verify(d^*, \sigma^*, \{pk_1, ..., pk_k\}) = \mathbf{TRUE}]$$

je pre každého útočníka v polynomiálnom čase zanedbateľná.[1]

4.2 Súčtová prstencová podpisová schéma

V tejto podkapitole bližšie popíšeme prstencovú podpisovú schému z článku Petzolda a Mohammeda [8], ktorú používame pri praktickej časti našej práce. Autori článku tvrdia, že schéma, ktorú navrhli môže byť použitá na rozšírenie ľubovoľnej podpisovej schémy ako napríklad UOV, Rainbow alebo Gui. Popíšeme si jednotlivé časti podpisovania a to generovanie kľúčov, generovanie podpisu a verifikáciu podpisu.

Generovanie kľúčov. Každý používateľ u_i si musí vopred vygenerovať svoj pár súkromného a verejného kľúča $((S_i, F_i, T_i), P_i)$. Verejný kľúč skupiny je spojenie všetkých verejných kľúčov každého účastníka $P = P_1||P_2||\cdots||P_k$, pričom súkromné kľúče zostávajú nezverejnené, teda v súkromí svojich vlastníkov.

Generovanie podpisu. Pri podpisovaní dokumentu d, skupinou $\mathcal{R}$, užívateľ u_i, ktorý generuje podpis, použije hašovaciu funkciu $\mathcal{H}$ na výpočet hašu správy $w = \mathcal{H}(d) \in \mathbb{F}^m$. Ďalej si vyberie náhodné vektory $z_1, ..., z_{i-1}, z_{i+1}, ...z_k \in \mathbb{F}^n$ a vypočíta

$$\widetilde{w} = w - \sum_{\substack{i=1 \\ j\neq i}}^{k} P_j(z_j) \in \mathbb{F}^m. \tag{9}$$

Následne použije svoj súkromný kľúč na výpočet vektora $z_i \in \mathbb{F}^n$ a to podľa $P(z_i) = \widetilde{w}$. Podpis správy d následne tvoria vektory $(z_1, ..., z_k) \in \mathbb{F}^{k\cdot n}$.

Overenie podpisu. Pri verifikácii, overovateľ kontroluje či vektory $(z_1, ..., z_k) \in \mathbb{F}^{k\cdot n}$ tvoria platný podpis správy d, pričom najprv použije hašovaciu funkciu $\mathcal{H}$ na výpočet hašu správy $w = \mathcal{H}(d) \in \mathbb{F}^m$ a následne použije verejné kľúče všetkých účastníkov $P_1||P_2||\cdots||P_k$ a vypočíta

$$\widehat{w} = \sum_{j=1}^{k} P_j(z_j) \tag{10}$$

Ak platí, že $\widehat{w} = w$, potom môže byť overovateľ spokojný pretože vie, že podpis je pravý. V rovnici č.11 je ukázané ako je možné, že ak je podpis správy validný musí platiť $\widehat{w} = w$.

$$\widehat{w} = \sum_{j=1}^{k} P_j(z_j) = \sum_{\substack{i=1 \\ j\neq i}}^{k} P_j(z_j) + P_i(z_i) = w - \widetilde{w} + \widetilde{w} = w \tag{11}$$

Podľa autorov článku [8] najväčšou nevýhodou súčtovej podpisovej schémy je zväčšovanie sa parametrov so zvyšujúcim sa počtom používateľov z dôvodov bližšie rozobraných v kapitole 4.4.

4.3 Súčinová prstencová podpisová schéma

Okrem predchádzajúcej prstencovej schémy článok Petzolda a Mohammeda [8], opisuje ešte jednu alternatívnu prstencovú podpisovú schému, ktorá má svoje výhody oproti súč-

tovej prstencovej podpisovej schémy. Ako už sme spomenuli v kapitole 4.2, pri súčtovej prstencovej podpisovej schéme je hlavným nedostatkom zväčšovanie sa parametrov so zvyšujúcim sa počtom používateľov. Táto schéma namiesto súčtu podpisov používa operáciu násobenia. Autori tvrdia, že týmto počinom je súčinová prstencová podpisová schéma odolná voči útokom, ktoré vznikajú keď v schéme je oveľa viac premenných ako rovníc, čo vzniká so zvyšujúcim sa počtom účastníkov.

Generovanie kľúčov. Každý používateľ u_i si vygeneruje svoj vlastný pár kľúčov $((S_i, F_i, T_i), P_i)$. Verejný kľúč skupiny tvorí množina všetkých verejných kľúčov každého účastníka $P = P_1||P_2||\cdots||P_k$, pričom každý účastník si uschová svoj súkromný kľúč (S_i, F_i, T_i).

Generovanie podpisu. Pri podpisovaní dokumentu d, skupinou $\mathcal{R}$, užívateľ u_i, ktorý generuje podpis, použije hašovaciu funkciu $\mathcal{H}$ na výpočet hašu správy $w = \mathcal{H}(d) + 1^m \in \mathbb{F}^m$, pričom 1^m predstavuje vektor jednotiek, ktorý je pričítaný ku hašu správy d. Ďalej si vyberie náhodné vektory $z_1, ..., z_{i-1}, z_{i+1}, ...z_k \in \mathbb{F}^n$ pričom každý vektor musí spĺnať nasledovnú podmienku

$$(P_j(z_j))_s \neq 0 \quad j \in \{1, ..., k\} \setminus \{i\}, s \in \{1, ..., m\}.$$

Potom vypočíta

$$\widetilde{w} = w \cdot (\prod_{\substack{i=1 \\ j \neq i}}^{k} P_j(z_j))^{-1} \in \mathbb{F}^m \tag{12}$$

Následne použije svoj súkromný kľúč na výpočet vektora $z_i \in \mathbb{F}^n$ a to podľa $P(z_i) = \widetilde{w}$. Podpis správy d následne tvoria vektory $(z_1, ..., z_k) \in \mathbb{F}^{k \cdot n}$.

Overenie podpisu. Pri verifikácii, overovateľ kontroluje či vektory $(z_1, ..., z_k) \in \mathbb{F}^{k \cdot n}$ tvoria platný podpis správy d, pričom najprv použije hašovaciu funkciu $\mathcal{H}$ na výpočet hašu správy $w = \mathcal{H}(d) \in \mathbb{F}^m$ a následne použije verejné kľúče všetkých účastníkov $P_1||P_2||\cdots||P_k$ a vypočíta

$$\widehat{w} = \prod_{j=1}^{k} P_j(z_j) \tag{13}$$

Ak platí, že $\widehat{w} = w$, potom môžeme tvrdiť, že podpis je pravý. V rovnici č.14 je ukázané ako je možné, že ak je podpis správy validný musí platiť $\widehat{w} = w$.

$$\widehat{w} = \prod_{j=1}^{k} P_j(z_j) = \prod_{\substack{i=1 \\ j \neq i}}^{k} P_j(z_j) \cdot P_i(z_i) = w \cdot \widetilde{w}^{-1} \cdot \widetilde{w} = w \tag{14}$$

4.4 Voľba parametrov pri použití prstencových podpisových schém

V tejto kapitole sme čerpali z webstránky môjho vedúceho práce pána Ing. Viliama Hromadu, PhD. [7] Pri použití skupinovej podpisovej schémy skonštruovanej podľa kapitoly 4.2 v našej práci, je potrebné podľa autorov článku [8] uvažovať nasledovný útok generovania falošného podpisu. Aby útočník sfalšoval podpis správy w a vydával sa za účastníka skupiny $\mathcal{R} = \{u_1, u_2, ..., u_k\}$, musí nájsť taký podpis $z_1, z_2, ..., z_k$, pre ktorý platí vzťah:

$$\mathcal{P}_1(z_1) + \mathcal{P}_2(z_2) + ... + \mathcal{P}_k(z_k) = w. \tag{15}$$

Existujú 2 spôsoby ako by to mohol dosiahnuť:

- Útočník náhodne vygeneruje $k-1$ hodnôt $z_1, z_2, ..., z_{k-1} \in \mathbb{F}^n$, vypočíta $\tilde{w} = w - \sum_{i=1}^{k-1} \mathcal{P}_i(z_i)$ a pokúsi sa nájsť riešenie z_k systému $\mathcal{P}_k(z_k) = \tilde{w}$.
- Útočník sa pokúsi priamo vyriešiť sústavu podľa rovnice č.15.

Bezpečnosť prvej uvedenej situácie je ekvivalentná zlomeniu jednej inštancie použitej podpisovej schémy. Bezpečnosť druhej uvedenej situácie, t.j. riešenie sústavy č. 15 *nie je tak ťažké* ako riešenie jednej inštancie podpisovej schémy. Je to z toho dôvodu, že sústava č.15 obsahuje oveľa viac premenných než rovníc, a teda ide o nedourčenú sústavu rovníc. Z článku [8] autori tvrdia, že pri riešení takejto sústavy rovníc platí:

1. Ak pre počet premenných n a počet rovníc m nedourčenej sústavy $\mathcal{P}$ platí, že $n = \omega m$, potom riešenie sústavy $\mathcal{P}$ je tak ťažké, ako riešenie sústavy $m - \lfloor \omega \rfloor + 1$ rovníc o rovnakom počte premenných.
2. Ak pre počet premenných n sústavy $\mathcal{P}$ o m rovniciach platí $n \geq \frac{m(m+3)}{2}$, potom je možné sústavu $\mathcal{P}$ riešiť v polynomiálnom čase.

Z týchto tvrdení vyplýva, že je potrebné nastaviť parametre použitých podpisových schém tak, aby:

- Každá inštancia podpisovej schémy každého používateľa spĺňala minimálne požadovanú úroveň bezpečnosti.
- Výsledný skupinový verejný kľúč predstavoval systém polynómov, ktorého hľadanie koreňov má zložitosť minimálne na požadovanej úrovni bezpečnosti.

V článku [8] uvádzajú autori aj odporúčané parametre pre podpisový algoritmus Rainbow, tak aby bola dosiahnutá požadovaná úroveň bezpečnosti s použitím súčtovej aj súčinovej prstencovej podpisovej schémy, vzhľadom na počet účastníkov 5, 10, 20 a 50. V prípade Rainbow sú to tri veličiny, ktoré predstavujú počet olejových neurčitých v prvej vrstve $o1$, počet octových neurčitých v prvej vrstve $v1$ a počet octových neurčitých v druhej vrstve $v2$. Avšak pre podpisovú schému GeMSS, tieto nastavenia parametrov pre rôzny počet účastníkov chýbajú. Z dostupných informácií o bezpečnosti systému GeMSS [4], sme sa rozhodli určiť tieto nastavenia parametrov, tak aby bola zachovaná 128 bitová bezpečnosť:

- Každej inštancie GeMSS každého účastníka prstencovej podpisovej schémy na úrovni 128 bitov.
- Výsledného skupinového verejného kľúča na úrovni 128 bitov, z hľadiska útoku hľadania riešenia sústavy č.15.

4.4.1 Voľba parametrov GeMSS

GeMSS pracuje nad poľom $GF(2)$, základné parametre sú:

- Stupeň rozšírenia poľa n
- Počet odstránených polynómov (modifikátor mínus) Δ
- Počet octových neurčitých v HFE polynóme v

Pre uvedené parametre je potom verejný kľúč jedného používateľa sústava polynómov nad konečným poľom $GF(2)$, ktorá obsahuje:

- Počet polynómov verejného kľúča: $m = n - \Delta$
- Počet neurčitých v polynómoch verejného kľúča: $n + v$

V prípade, že uvažujeme prstencovú podpisovú schému, ktorej sa zúčastňuje k používateľov, tak vzťah č15 predstavuje sústavu polynómov, ktorá obsahuje

- Počet polynómov v súčte danom vzťahom č.15: $m = n - \Delta$
- Počet neurčitých v polynómoch vzťahu č.15: $k(n + v)$

V článku [4] v kapitole 8.3.1 sa nachádza vzťah, ktorý udáva asymptotickú zložitosť riešenia kvadratickej sústavy m rovníc o m premenných nad konečným poľom $GF(2)$

ako $O(2^{0.792 \cdot m})$. Aby sme dosiahli požadovanú úroveň bezpečnosti 128 bitov, $m \geq 162$, keďže zložitosť riešenia sústavy $m = 162$ rovníc o 162 premenných je podľa tohto vzťahu $O(2^{128.3})$. Preto aj verzia podpisovej schémy GeMSS so 128 bitovou bezpečnosťou GeMSS128 pre jedného používateľa, ktorá je súčasťou návrhu [4], je navrhnutá tak, aby $m = 162$.

Ak sa použije schéma GeMSS s parametrami (n, Δ, v) na tvorbu prstencovej podpisovej schémy, tak sústava daná vzťahom č.15 obsahuje m rovníc a $k(n + v)$ premenných. V takom prípade dochádza k tomu, že riešenie tejto sústavy je tak ťažké, ako riešenie sústavy $(n - \Delta) - \lfloor \frac{k(n+v)}{(n-\Delta)} \rfloor + 1$ rovníc o rovnakom počte premenných [8].

Preto *základné pravidlo* voľby parametrov GeMSS (n, Δ, v) pre použitie v prstencovej schéme k používateľov, ktorá dosahuje 128-bitovú bezpečnosť je, aby pre parametre platilo:

$$(n - \Delta) - \lfloor \frac{k(n + v)}{n - \Delta} \rfloor + 1 \geq 162 \tag{16}$$

Ak spĺňajú parametre vzťah č.16, potom možno predpokladať, že zložitosť riešenia sústavy č.15 je aspoň 2^{128}.

Samozrejme, parametre (n, Δ, v) musia zároveň spĺňať predpoklady zachovania 128-bitovej bezpečnosti pre jednotlivé inštancie systému GeMSS. V kapitole 8.7 článku [4] autori uvádzajú spôsob výpočtu parametrov, aby bola schéma GeMSS bezpečná na požadovanej úrovni λ. V našom prípade $\lambda = 128$. Ďalší dôležitý parameter GeMSS je stupeň použitého HFE polynómu D. V našom prípade $D = 513$

1. Pre počet polynómov verejného kľúča m musí platiť

$$m \geq 1.26\lambda = 161.28 \tag{17}$$

2. Pre stupeň regularity D_{reg} sústavy polynómov verejného kľúča musí platiť

$$O\binom{m}{D_{reg}}^2 \geq 2^{\lambda} \tag{18}$$

3. Zároveň v kapitole 8.5 článku [4] autori uvádzajú, že pre počet $\Delta + v$ by malo platiť

$$\Delta + v = 3 \times (D_{reg} - D_{reg}^{HFE}), \tag{19}$$

D_{reg}^{HFE} je stupeň regularity centrálneho HFE zobrazenia bez modifikátorov.

4. Hodnota D_{reg}^{HFE} sa dá podľa [4] aproximovať

$$D_{reg}^{HFE} \approx 2,03 + 0,36 \log_2(D) \approx 6. \tag{20}$$

Na základe uvedených vzťahov sme určili hodnoty parametrov (n, Δ, v) pre rôzne počty účastníkov prstencovej podpisovej schémy, $k = 5, 10, 20, 50$. Hodnoty boli zvolené tak, aby výsledná prstencová podpisová schéma spĺňala vzťah č. 16 a zároveň, aby každá inštancia GeMSS s danými parametrami spĺňala vzťahy č.17, č.18, č.19, č.20. Parametre boli odvodené od schémy GeMSS128 pre jedného používateľa, pre ktorú platí $(n, \Delta, v) = (174, 12, 12)$ [4]. Vo všetkých prípadoch sa jedná o schémy nad *konečným poľom* $GF(2)$, *stupeň HFE polynómu je* $D = 513$.

Hodnoty parametrov uvádzame v tabuľke č.1. Parametre sú platné pre uvedený počet účastníkov k v záhlaví stĺpca, alebo pre prípadný menší počet účastníkov.

128-bitová bezpečnosť	$k = 5$	$k = 10$	$k = 20$	$k = 50$
Parametre (n, Δ, v)	(178,12,12)	(184,12,12)	(194,11,11)	(227,11,11)
$m = n - \Delta$	166	172	183	216

Tabuľka 1: Navrhované hodnoty parametrov GeMSS pre prstencovú podpisovú schému

Parametre boli zvolené tak, aby v uvedených prípadoch $k = 5, 10, 20, 50$ príslušná schéma vo vzťahu č.16 mala hodnotu ľavej strany práve 162. Z hľadiska bezpečnosti jednotlivých inštancií GeMSS :

1. Z tabuľky č.1 vidieť, že vo všetkých prípadoch je nerovnosť č.17 splnená.

2. Na základe parametra m je požadovaný stupeň regularity D_{reg} podľa vzťahu č.18 a na základe D_{reg} a vzťahu č.19 pre parametre $\Delta + v$ musia platiť hodnoty z tabuľky č.2.

	$k = 5$	$k = 10$	$k = 20$	$k = 50$
Parametre (n, Δ, v)	(178,12,12)	(184,12,12)	(194,11,11)	(227,11,11)
D_{reg}	14	14	13	13
$\Delta + v$	24	24	21	21

Tabuľka 2: Požadované hodnoty D_{reg} a pre uvedené verzie GeMSS

V prípade schém pre $k = 5$ a $k = 10$ účastníkov je tento vzťah splnený. V prípade schém pre $k = 20$ a $k = 50$ sme pristúpili k voľbe parametrov kde $\Delta + v = 22$. Je to z toho dôvodu, že autori v článku [4] odporúčajú v kapitole 8.5 aby $\Delta = v$.

5 Návrh riešenia práce

V tejto kapitole si bližšie popíšeme aké základné ciele a požiadavky sme si stanovili pri tvorbe našej práce. Následne si načrtneme aj návrh riešenia zadania práce a popíšeme si algoritmy, ktoré sme použili pri riešení.

5.1 Požiadavky a popis zadania práce

Hlavnou úlohou našej práce je implementovať 2 prstencové podpisové schémy na už existujúce implementácie MPKC systémov. Keďže v čase, keď túto prácu píšeme ešte prebieha súťaž NIST, ktorej cieľom je výber nových algoritmov aj pre vytvorenie digitálneho podpisu, medzi ktorými sú algoritmy, ktoré sú založené na probléme riešenia sústavy polynómov s viacerými neurčitými, vybrali sme si 2 algoritmy, na ktorých vyskúšame implementácie prstencových podpisových schém. Medzi finalistov súťaže, ktoré sú založené na probléme MPKC sú systémy Rainbow a GeMSS, ktoré použijeme v našej práci na experimenty s prstencovými podpisovými schémami. Cieľom práce je preskúmať aplikovateľnosť, reprodukovateľnosť a porovnať efektívnosť vybraných prstencových schém. Myšlienka riešenia zadania je rozšírenie existujúcej implementácie vybraného MPKC systému o možnosť použitia tohto algoritmu viacerými účastníkmi, pričom použijeme ideu prstencovej podpisovej schémy z kapitoly 4.2 a 4.3. Kľúčovou požiadavkou v tejto práci je vytvorenie softvéru, ktorý bude slúžiť na testovanie prstencových schém, pričom budeme tieto prstencové schémy skúmať a zaujíma nás, ktorá schéma je vhodnejšia pre daný algoritmus a z výsledkov predstavíme naše závery a odporúčania. Riešenie by malo byť schopné generovať podpisy pre ľubovoľný počet účastníkov a taktiež následne by mal byť podpis aj ľahko overiteľný. Z našich experimentov budeme sledovať veľkosti verejných kľúčov, veľkosť výsledného podpisu a aj dobu trvania overenia a vygenerovania podpisu s viacerými účastníkmi.

5.2 Návrh riešenia

Pri návrhu riešenia si predstavíme implementácie oboch prstencových schém. Postupne si popíšeme algoritmy pre vygenerovanie prstencového podpisu a overenie prstencového podpisu. Algoritmus generovania kľúčov nebudeme meniť, rozšírime ho len o možnosť vygenerovať si kľúče pre viacerých účastníkov.

5.2.1 Riešenie generovania a overenia podpisu súčtovej schémy

Algoritmus 1 *CryptoSumSign* zobrazuje pseudokód metódy, ktorá generuje súčtový skupinový podpis podľa kapitoly 4.2, pričom do tejto metódy vstupujú premenné ako

verejné kľúče, správa, ktorú sa chystáme podpísať, súkromný kľúč, počet účastníkov a poradie účastníka, ktorý vykonáva proces podpisovania. Výstupom z tejto funkcie sú vektory, ktoré spolu tvoria celkový podpis správy. V úvode funkcie si najprv vygenerujeme náhodné vektory, pričom vynechávame vektor účastníka, ktorý dokument podpisuje. *Zeros* metóda naznačuje, že na začiatku výpočtov je potrebné pole, kde budeme ukladať výsledky inicializovať nulami. Metóda *CryptoSignOpen* predstavuje dosadenie hodnôt v vektorov $vectors_i$ do polynómov verejného kľúča pks_i a vracia hodnoty tohto polynomiálneho systému. *CryptoSign* vypočíta podpis hodnoty w za pomoci súkromného kľúča používateľa i.

Algoritmus 1 Vygenerovanie prstencového podpisu

1: **procedure** CryptoSumSign($pks, message, sk, users, order_i$)
2: $digest \leftarrow \mathcal{H}(message)$
3: **for** i from 0 to $users$ **do**
4: **if** $i == order_i$ **then**
5: $do\_nothing()$.
6: **else**
7: $vectors_i \leftarrow \mathsf{rand}()$
8: $result \leftarrow zeros(n)$ ▷ zeros metóda inicializuje pole nulami
9: **for** i from 1 to $users$ **do**
10: $result \leftarrow result + \mathsf{CryptoSignOpen}(pks_i, vectors_i)$
11: $w \leftarrow digest - result$
12: $z_i \leftarrow \mathsf{CryptoSign}(pks[order_i], sk, w)$
13: $vectors[order_i] \leftarrow z_i$
14: **return** $vectors$

Pri verifikácii potrebujeme do funkcie dodať verejné kľúče všetkých účastníkov, správu ktorá bola podpísaná, počet účastníkov prstenca a samotný podpis. Sčítame výsledky verifikačnej funkcie všetkých vektorov, ktoré sú súčasťou podpisu, môžeme si všimnúť, že poradie účastníka, ktorý správu podpisoval už nie je potrebné. Následne porovnáme či je výsledok zhodný s hašom správy. Ak áno, úspešne sme overili podpis, ak nie podpis zamietame.

Algoritmus 2 Overenie prstencového podpisu

```
1: procedure SumVerify(pks, message, users, signature)
2:     digest ← H(message)
3:     result ← zeros(n)                    ▷ zeros metóda inicializuje pole nulami
4:     for i from 0 to users do
5:         result ← result + CryptoSignOpen(pks_i, signature_i)
6:     if result == digest then
7:         return 0
8:     else
9:         return 1
```

5.2.2 Riešenie generovania a overenia podpisu súčinovej schémy

Algoritmus 3 *CryptoSignMultiply* znázorňuje pseudokód, ktorý vytvára skupinový podpis podľa súčinovej podpisovej schémy podľa kapitoly 4.3. Na začiatku správu zahašujeme, následne pomocou metódy *ones* naplníme pole výsledkov s jednotkami, pretože k nim budeme postupne násobiť ďalšie hodnoty. Výsledok operácie *CryptoSignOpen* nemôže obsahovať 0, preto musíme generovať náhodné vektory dovtedy, kým sa nám nepodarí mať všetky výsledky nenulové. Túto skutočnosť popisujú riadky 8 - 10. Metóda *gf256inv* zinvertuje výsledok násobenia, aby sme mohli podpis verifikovať. Všetky tieto operácie prebiehajú v poli $GF(256)$. Premenné $result$, $result_i$, $digest$ a inv_result sú vektory hodnôt nad $GF(256)$, preto násobenia na riadku 11 a 15 sú násobenia po zložkách a taktiež inverzia na riadku 12 je inverziou po zložkách vektora $result$. Je potrebné zabezpečiť aby haš dokumentu bol taktiež nenulový. Autori článku [8] tento problém vyriešili pripočítaním vektora jednotiek, ale my sme sa rozhodli pre iné riešenie. Haš správy postupne hašujeme ďalej až kým haš nie je nenulový. Sme si vedomí toho, že môžu vznikať kolízie, čo môže viesť k bezpečnostnému riziku avšak myslíme si, že toto riešenie je vhodnejšie ako riešenie autorov. Následne už len vynásobime nenulový haš správy so zinvertovaným výsledkom násobenia a tento výsledok použijeme pri vypočítaní vektora z_i pomocou podpisového algoritmu, ktorého vstupom je aj súkromný kľúč. Celkový podpis správy je množina náhodne vygenerovaných vektorov a vypočítaný vektor z_i.

Algoritmus 3 Vygenerovanie prstencového podpisu

1: **procedure** CRYPTOSIGNMULTIPLY($pks, message, sk, users, order_i$)
2: $digest \leftarrow \mathcal{H}(message)$
3: $result \leftarrow ones(n)$ ▷ ones metóda, ktorá inicializuje pole s jednotkami
4: **for** i from 0 to $users$ **do**
5: **if** $i == order_i$ **then**
6: $do\_nothing()$.
7: **else**
8: **while** 0 in $result_i$ **do**
9: $vectors_i \leftarrow rand()$
10: $result_i \leftarrow \mathsf{CryptoSignOpen}(pks_i, vectors_i)$
11: $result \leftarrow result * result_i$
12: $inv\_result = \mathsf{gf256inv}(\mathsf{result})$
13: **while** 0 in $digest$ **do**
14: $digest \leftarrow \mathcal{H}(digest)$
15: $w \leftarrow digest * inv\_result$
16: $z_i \leftarrow \mathsf{CryptoSign}(pks[order_i], sk, w)$
17: $vectors[order_i] \leftarrow z_i$
18: **return** $vectors$

Algoritmus 4 *MultiplyVerify* overuje či vstupný podpis je pravým podpisom odosielateľa. Prechádzame cez všetkých užívateľov pričom *CryptoSignOpen* je metóda, ktorá dosadí hodnoty vstupného vektora do neurčitých verejného kľúča, ten vyhodnotí a výsledný vektor hodnôt násobime už s predchádzajúcimi výsledkami. Taktiež, ešte musíme overiť, či haš správy nie je nenulový ak je nulový zahašujeme správu znovu a tento proces opakujeme dokým nebude haš správy nenulový. Presne takým istým spôsobom sme postupovali aj pri generovaní podpisu. Posledným krokom je už len overenie, či haš je rovný výsledku vypočítanému pomocou verejných kľúčov. Ak sa tieto dve hodnoty rovnajú program vráti 0 a ak nie, je vrátená 1.

Algoritmus 4 Overenie prstencového podpisu

```
1: procedure MultiplyVerify(pks, message, users, signature)
2:     digest ← H(message)
3:     result ← ones(n)                ▷ ones metóda, ktorá inicializuje pole s jednotkami
4:     for i from 0 to users do
5:         result ← result * CryptoSignOpen(pks_i, signature_i)
6:     while 0 in digest do
7:         digest ← H(digest)
8:     if result == digest then
9:         return 0
10:    else
11:        return 1
```

5.2.3 Metodika experimentov

Vrámci experimentovania s našou implementáciou je našim cieľom medzi sebou porovnať algoritmy GeMSS a Rainbow. Experimentovať budeme s parametrami algoritmov, ktoré zaručujú 128-bitovú bezpečnosť. Našej pozornosti neuniknú veľkosti verejných kľúčov a súkromných kľúčov či veľkosť vygenerovaného podpisu. Hlavným aspektom, ktorý nás bude zaujímať je čas za aký sa podpis vygeneruje a verifikuje. Keďže pri súčinovej podpisovej schéme musíme generovať náhodne vektory dovtedy, kým výsledky z riešenia sústavy nebudú nenulové, budeme pozorovať ako to ovplyvní celkový čas výpočtu oproti súčtovej schéme, kde sa budú parametre so zvyšujúcim počtom účastníkov zväčšovať. Taktiež budeme meniť počet účastníkov a sledovať aký vplyv bude mať táto skutočnosť na zvolenú prstencovú podpisovú schému, pričom pri súčinovej prstencovej schéme sa so zvyšujúcim počtom účastníkov parametre nemenia, a tým pádom veľkosť súkromného kľúča bude rovnaká a veľkosť verejného kľúča sa bude meniť iba podľa počtu účastníkov.

5.3 Návrh testovacej aplikácie

Testovacia aplikácia je jednoduchá aplikácia, ovládaná z príkazového riadku. Pozostáva z troch príkazov a to z:

- príkazu na vygenerovanie kľúčov (genkey)
- príkazu na vygenerovanie podpisu (sign)
- príkazu na verifikáciu podpisu (verify)

GenKey potrebuje ako vstupné parametre názvy súborov, kde uloží verejné kľúče a súkromný kľúč užívateľa. Pre testovacie účely aplikácia generuje verejné kľúče všetkých účastníkov do jedného súboru. Počet účastníkov je konfigurovateľný priamo v zdrojovom kóde. Do súboru súkromného kľúča uschová len jeden kľúč daného účastníka, v testovacej aplikácii uvažujeme, že daný účastník je prvý v poradí.
Sign potrebuje na vykonanie svojej funkcie 3 vstupné parametre. Potrebuje textový súbor, ktorý sa chystáme podpísať, súbor so súkromným kľúčom a súbor s verejným kľúčom. Výsledkom tohto príkazu je súbor, kde je uložený podpis dokumentu.
Verify očakáva na vstupe textový súbor, ktorý bol podpísaný, ďalej potrebuje súbor s daným podpisom a súbor s verejnými kľúčmi všetkých účastníkov. Výsledkom tejto funkcie je odpoveď, či vstupný podpis je pravý alebo falošný.

6 Výsledky a testovanie

V tejto kapitole uvedieme výsledky našich experimentov s vytvorenými implementáciami prstencových podpisových schém. Postupne budeme porovnávať súčtovú podpisovú schému so súčinovou podpisovou schémou s použitím algoritmu Rainbow. Parametre Rainbow podpisovej schémy sú určené pre 128 bitovú bezpečnosť. Následne overíme efektivitu súčtovej schémy nad GeMSS a súčtovej schémy nad Rainbow s použitím parametrov potrebných pre zaistenie 128 bitovej bezpečnosti.

6.1 Špecifikácia zvolenej platformy

Riešenie sme realizovali v jazyku *C*, s použitím kompilátora *gcc* verzie 10.2.0. Pre kompiláciu na operačnom systéme Windows 10 sme použili platformu *Cygwin 3.2.0*. Pri implementácii prstencovej podpisovej schémy sme využili voľne dostupné zdrojové kódy podpisových schém *Rainbow* a *GeMSS* zo súťaže pod záštitou NIST organizácie. Použili sme implementovaný algoritmus GeMSS z 3.kola súťaže dostupný na webstránke `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions` zo dňa 6.5.2021, a algoritmus Rainbow z 2.kola súťaže dostupný na webstránke `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions` zo dňa 6.5.2021. Dôvodom výberu Rainbow z 2.kola bolo pridanie bezpečnostných opatrení v najnovšej verzii, ktoré obmedzovali ľubovoľný výber parametrov, čo je pri našich experimentoch kľúčové. Tieto implementácie zvolených algoritmov dokážu vygenerovať pár kľúčov, vygenerovať podpis na zvolenú správu, a overiť či daný podpis s daným verejným kľúčom nie sú falošné. Na realizáciu experimentov používame osobný počítač s procesorom *Intel Core m3-6Y30*, s grafickým procesorom *Intel HD Graphics 515* a s 8GB RAM.

6.2 Výsledky experimentov

Postupne sme s našou testovacou aplikáciou vykonali 100 experimentov pre každý prípad v tabuľkách č.3, č.4 a č.6, pričom sme pri každom experimente merali čas výpočtu podpisovacieho algoritmu, čas výpočtu verifikačného algoritmu, čas výpočtu generovania kľúčov a veľkosti verejných, súkromných kľúčov a veľkosť podpisu. Experimenty sme vykonávali postupne pre 5, 10, 20 a 50 účastníkov, pričom časové výsledky sú spriemerované a uvedené v tabuľkách. Textový súbor, ktorý sme podpisovali bol pri každom experimente rovnaký a to o veľkosti 14 B. Parametre Rainbow schémy meníme podľa článku [8] a parametre GeMSS sme určili podľa našich výpočtov z kapitoly 4.4.1.

V tabuľke č.3 sú výsledky experimentov s použitím súčtovej podpisovej schémy nad

Rainbow, v tabuľke č.4 sú výsledky meraní s použitím súčinovej podpisovej schémy nad Rainbow a v tabuľke č.6 sú uvedené výsledky meraní súčtovej podpisovej schémy nad GeMSS128. V tabuľke č.5 sú uvedené počty vygenerovaných náhodných vektorov pri použití súčinovej a súčtovej podpisovej schémy nad Rainbow.

	$k = 5$	$k = 10$	$k = 20$	$k = 50$
Parametre (v_1, o_1, o_2)	(36, 23, 20)	(34, 26, 23)	(32, 33, 29)	(30, 64, 58)
Veľkosť verejného kľúča(kB)	679,4	1708,14	5536,6	70930,8
Veľkosť súkromného kľúča(kB)	95,862	115,178	173,072	738,942
Veľkosť podpisu(B)	475	990	2200	8400
Generovanie podpisu(ms)	412,63	918,24	2757,37	32435,24
Verifikácia podpisu(ms)	368,4	875,57	2857,71	33032,95
Generovanie kľúčov(ms)	487,95	1121,88	3522,44	49572,36

Tabuľka 3: Súčtová podpisová schéma nad Rainbow

	$k = 5$	$k = 10$	$k = 20$	$k = 50$
Parametre (v_1, o_1, o_2)	(36, 21, 22)	(36, 21, 22)	(36, 21, 22)	(36, 21, 22)
Veľkosť verejného kľúča(kB)	679,4	1358,8	2717,6	6794
Veľkosť súkromného kľúča(kB)	96,32	96,32	96,32	96,32
Veľkosť podpisu(B)	475	950	1900	4750
Generovanie podpisu(ms)	470,24	868,66	1486,63	3406,12
Verifikácia podpisu(ms)	446,02	842,42	1424,17	3447,64
Generovanie kľúčov(ms)	615,34	1057,74	1822,88	4238,95

Tabuľka 4: Súčinová podpisová schéma nad Rainbow

	$k = 5$	$k = 10$	$k = 20$	$k = 50$
Súčinová schéma	480	1077	2321	5483
Súčtová schéma	400	900	1900	4900

Tabuľka 5: Súčet vygenerovaných vektorov zo 100 opakovaní

	$k = 5$	$k = 10$	$k = 20$	$k = 50$
Parametre (n, Δ, v)	(178,12,12)	(184,12,12)	(194,11,11)	(227,11,11)
Veľkosť verejného kľúča(kB)	1882,65	4151,01	9660,58	38396,95
Veľkosť súkromného kľúča(B)	16	16	16	16
Veľkosť podpisu(B)	165	340	680	1900
Generovanie podpisu(ms)	2245,75	4253,51	7956,7	21566,12
Verifikácia podpisu(ms)	1084,06	1915,55	4681,12	18488,61
Generovanie kľúčov(ms)	988,09	2000,97	5606,47	20340,11

Tabuľka 6: Parametre (n, Δ, v) pre uvedené verzie GeMSS

6.3 Diskusia

V tejto kapitole prezentujeme výsledky našich pokusov s testovacou aplikáciou, pričom používame tzv. boxploty, ktoré sú vhodné na zobrazenie priemerov jednotlivých časov experimentov. Spodná hranica boxplotu v grafe označuje prvý kvartil a zhora ho ohraničuje tretí kvartil, pričom medzi nimi je hodnota mediánu. Šírka boxplotov bude stále rovnaká, pretože určuje veľkosť testovacieho súboru a to je v našom prípade 100. Najprv porovnávame súčtovú schému so súčinovou nad schémou Rainbow a následne súčtovú podpisovú schému nad Rainbow a GeMSS.

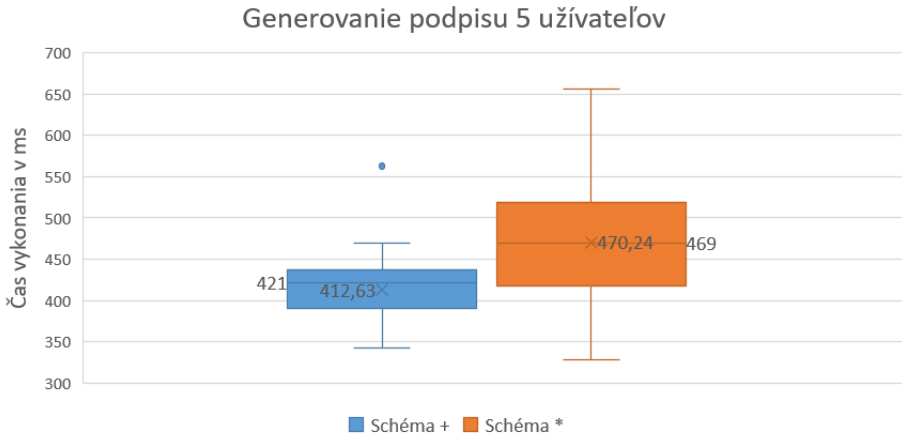
6.3.1 Porovnanie operácií súčtovej a súčinovej schémy s 5 užívateľmi

Na obrázku č.5 je znázornená dvojica boxplotov, ktorá predstavuje súbor časov generovania podpisu s 5 užívateľmi, pričom modrou farbou je označená súčtová schéma a oranžovou farbou je označená súčinová schéma. Krížikom je označený priemer a čiarou, ktorá prechádza cez box je vyznačený medián alebo inak stredná hodnota. Z grafu č.5 je zrejmé, že súčtová schéma je pre 5 užívateľov pri generovaní podpisu rýchlejšia, keďže dosahuje rýchlejší priemerný čas ako súčinová schéma a nachádza sa pod boxplotom, ktorý znázorňuje súčinovú schému.

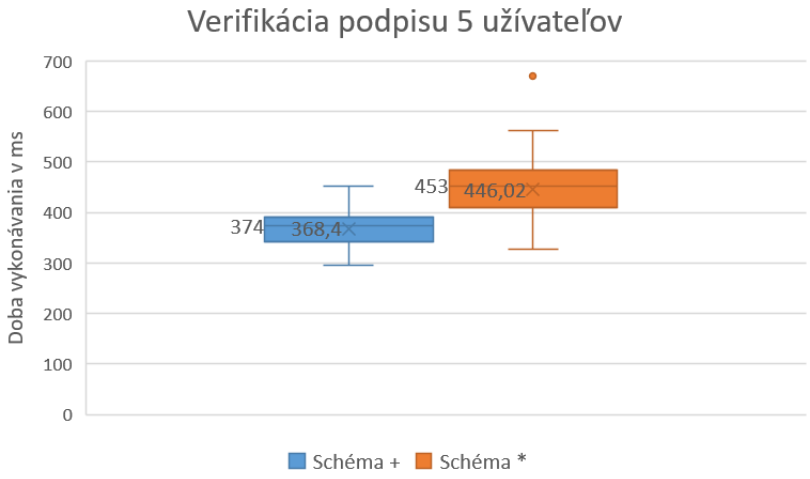
Pri verifikačnom algoritme, ktorý je znázornený na grafe č.6 vyplýva, že súčtová podpisová schéma má nižšiu priemernú rýchlosť aj v tomto prípade. Môžeme si všimnúť, že oproti generovaniu podpisu sa pri verifikácii jednotlivé boxy už neprekrývajú, teda rozdiely v časoch sú ešte väčšie.

Pri generovaní kľúčov s 5 užívateľmi bola rýchlejšia súčtová schéma, čo mohlo spôsobiť mierne menší súkromný kľúč v prípade súčtovej podpisovej schémy, výsledky potvrdzuje aj graf č.7.

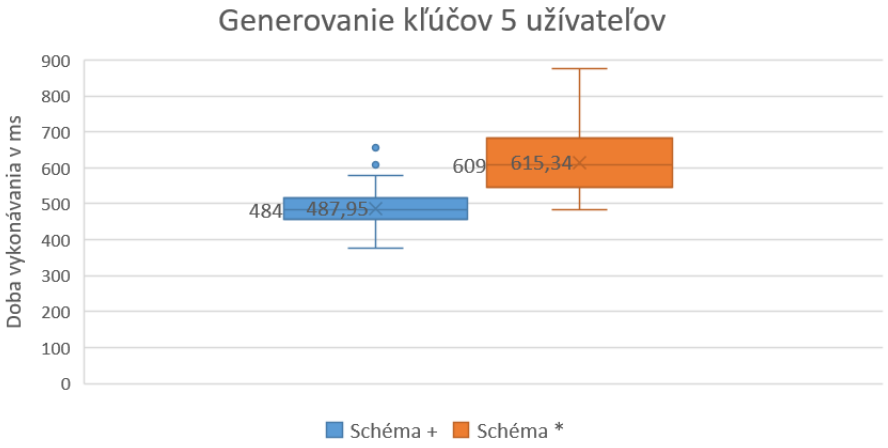
Keďže veľkosti verejných kľúčov sú v oboch prípadoch rovnaké a to o rozmeroch 79



Obrázok 5: Generovanie podpisu s 5 užívateľmi s použitím súčtovej a súčinovej schémy



Obrázok 6: Verifikácia podpisu s 5 užívateľmi s použitím súčtovej a súčinovej schémy

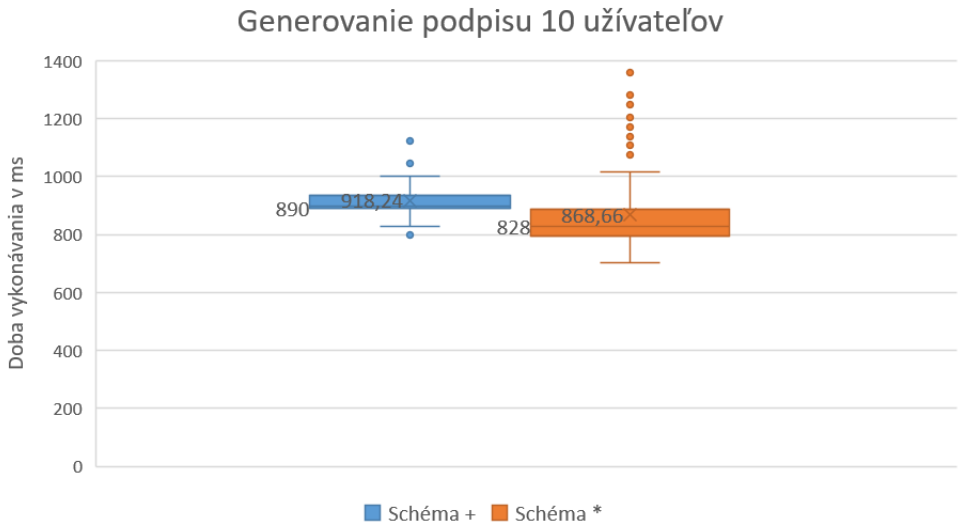


Obrázok 7: Generovanie kľúčov s 5 užívateľmi s použitím súčtovej a súčinovej schémy

neurčitých a 43 polynómov, rozdiel v priemerných časoch môže byť spôsobený najmä tým, že súčtová schéma používa operáciu sčítanie nad $GF(256)$, čo sa dá realizovať jednoduchou operáciou XOR, avšak pri súčinovej schéme sa vykonáva násobenie, čo je časovo náročnejšia operácia ako súčet. Ďalej inverzia, ktorá pri súčtovej schéme chýba a nakoniec súčinová schéma ešte musí čakať na vygenerovanie vektorov so samými nenulovými hodnotami.

6.3.2 Porovnanie operácií súčtovej a súčinovej schémy s 10 užívateľmi

S 10 účastníkmi, už pozorujeme opačnú situáciu. Súčinová schéma sa ukázala byť efektívnejšia ako súčtová, uvedenú skutočnosť môžeme vidieť aj na grafe č.8. Na grafoch si môžeme všimnúť aj to, že jednotlivé boxy majú nižšiu výšku, čo znamená, že dáta sa na seba viac podobajú, nie sú rozptýlené.

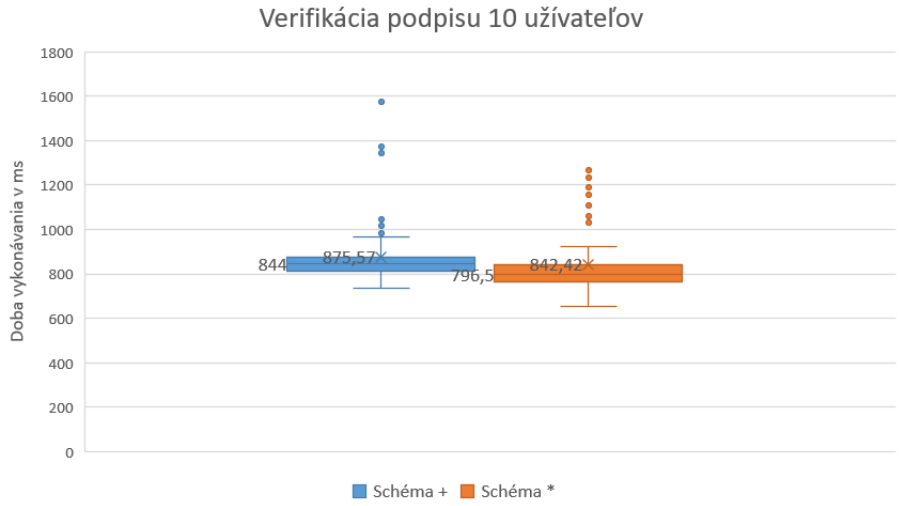


Obrázok 8: Generovanie podpisu s 10 užívateľmi s použitím súčtovej a súčinovej schémy

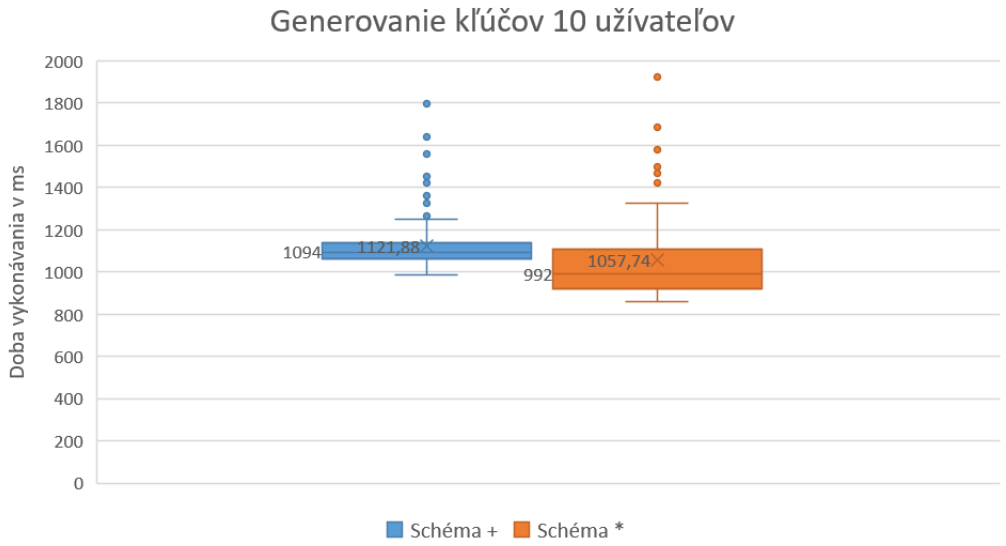
Pri verifikácii sú výsledky súčinovej a súčtovej schémy veľmi podobné, čo je vidno aj na grafe č.9, môžeme vidieť, že sa čiastočne prekrývajú. Pri tomto jedinom príklade sú dáta najviac podobné, avšak rýchlejšia je aj v tomto prípade súčinová schéma.

Generovanie kľúčov je efektívnejšie s použitím súčinovej schémy, vyplýva to aj z grafu č.10. Môžeme pozorovať, že veľkosť boxu súčinovej schémy sa mierne zväčšila, čo znamená, že dáta vrámci súčinovej schémy sa od seba viac líšia.

S použitím 10 účastníkov je efektívnejšia súčinová schéma, dôvodom sú menšie parametre, čo znamená, že schéma pracuje s menším počtom polynómov a s menším počtom



Obrázok 9: Verifikácia podpisu s 10 užívateľmi s použitím súčtovej a súčinovej schémy

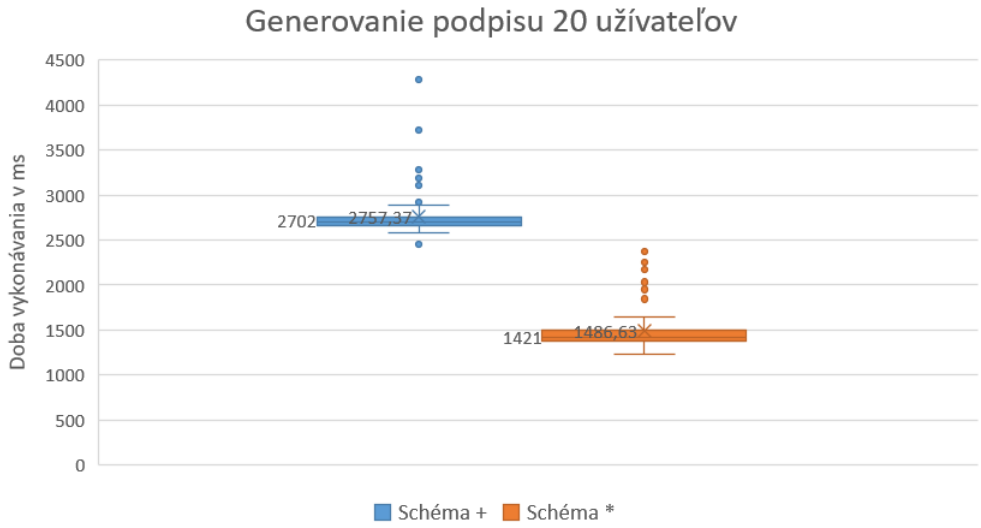


Obrázok 10: Generovanie kľúčov s 10 užívateľmi s použitím súčtovej a súčinovej schémy

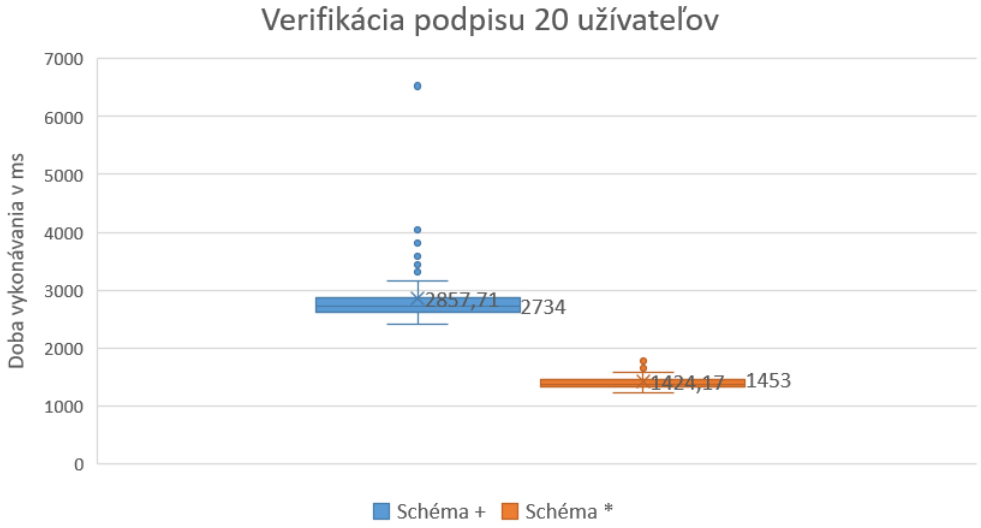
neurčitých oproti súčtovej schéme, čo sa odrazilo aj na veľkosti verejných a súkromných kľúčov.

6.3.3 Porovnanie operácií súčtovej a súčinovej schémy s 20 užívateľmi

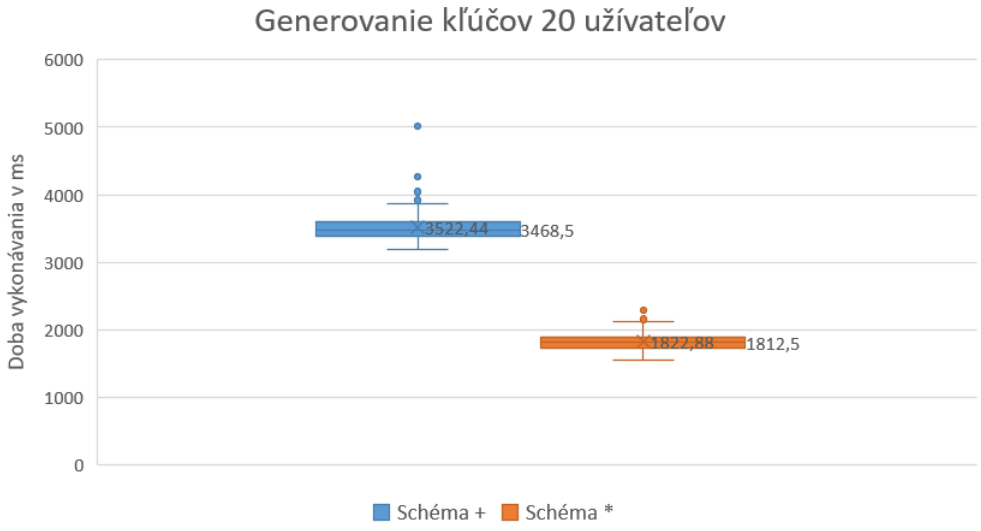
Z grafov č.11, č.12 a č.13 je zreteľne vidieť, že rozdiely v rýchlostiach medzi súčinovou a súčtovou schémou sa ešte viac prehĺbili. Medzi množinami dát sú priepastné rozdiely, čo spôsobilo zväčšovanie sa parametrov pri súčtovej schéme, pričom pri súčinovej schéme parametre zostali rovnaké.



Obrázok 11: Generovanie podpisu s 20 užívateľmi s použitím súčtovej a súčinovej schémy



Obrázok 12: Verifikácia podpisu s 20 užívateľmi s použitím súčtovej a súčinovej schémy

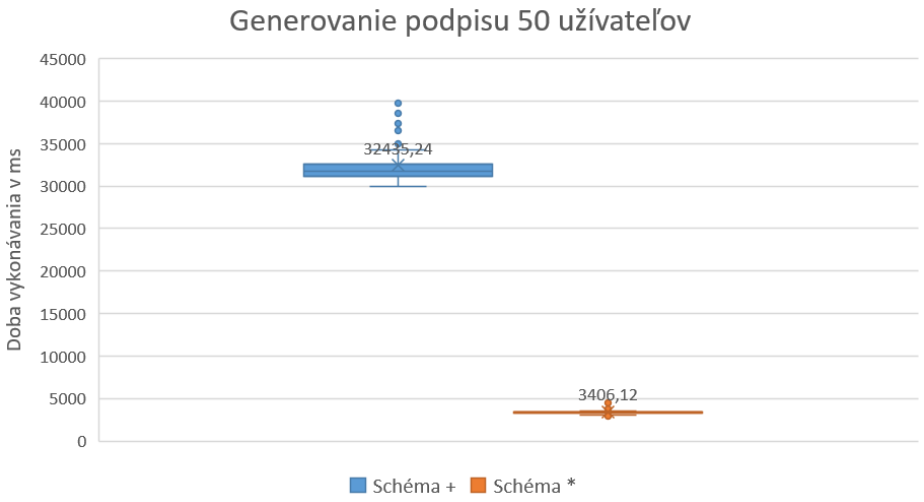


Obrázok 13: Generovanie kľúčov s 20 užívateľmi s použitím súčtovej a súčinovej schémy

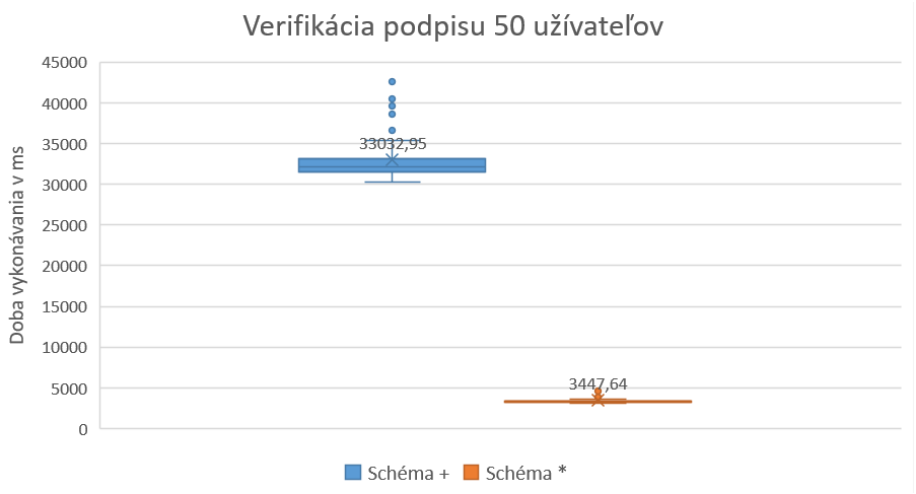
6.3.4 Porovnanie operácií súčtovej a súčinovej schémy s 50 užívateľmi

Pri 50 užívateľoch sú množiny dát súčtovej a súčinovej schémy neporovnateľné, čo je aj vidno z grafov č.14, č.15 a č.16, kde sú vyznačené už iba priemerné hodnoty jednotlivých časov. Môžeme si všimnúť obrovské rozdiely v rýchlosti, všetkých troch operácií. Súčinová metóda ako už aj z predošlých experimentov vyplýva, je rýchlejšia práve pri vyššom počte účastníkov. Je to práve preto, že parametre Rainbow sa nemenia, a tým pádom veľkosti kľúčov aj podpisu sa menia iba z dôvodu pribúdajúcich užívateľov. Generovanie kľúčov je oproti verifikácii a generovaní podpisu časovo náročnejšie, trvá najdlhšie zo všetkých operácií.

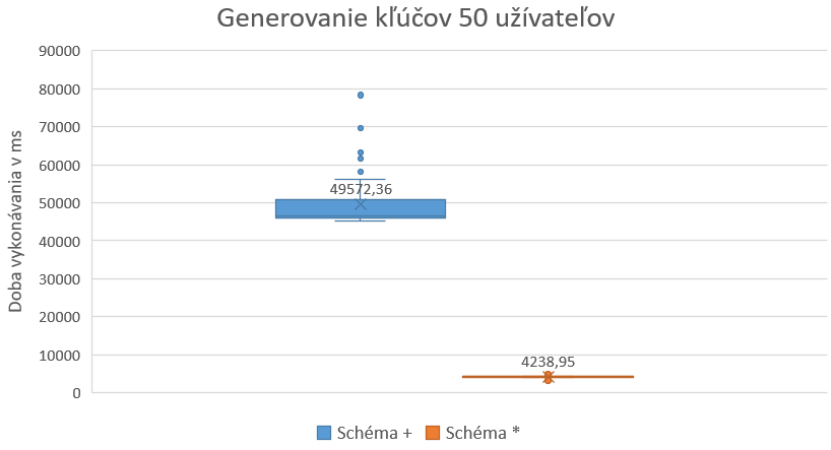
Súčinová metóda mala v implementácii tú nevýhodu, že potrebovala na svoj výpočet nenulové hodnoty, čo záviselo iba na náhodnej funkcii, ktorá generovala náhodné vektory, avšak z výsledkov môžeme konštatovať, že toto spomalenie nemalo na výsledný beh algoritmu skoro žiadny vplyv. Taktiež ďalšou výhodou súčinovej schémy sú veľkosti kľúčov a veľkosť podpisu, ktoré sú pre bežné používanie praktickejšie oproti súčtovej schéme. Veľkosť podpisu je taktiež nižšia pri súčinovej podpisovej schéme, pretože veľkosť podpisu závisí najmä od veľkosti parametrov. V tabuľke č.5 sú zapísané počty vygenerovaných náhodných vektorov, v súčtovej schéme bol každý vektor vygenerovaný práve raz, avšak pri súčinovej schéme pri 5 účastníkoch bol počet vygenerovaných vektorov zvýšený o 20%, pri 10 účastníkoch zvýšený o 19,67%, pri 20 účastníkoch o 22,16% a pri 50 účastníkoch iba o 11,9%.



Obrázok 14: Generovanie podpisu s 50 užívateľmi s použitím súčtovej a súčinovej schémy



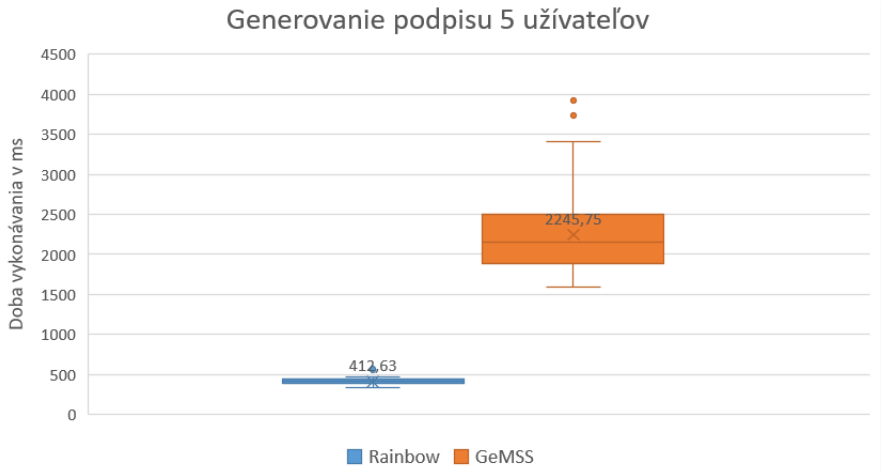
Obrázok 15: Verifikácia podpisu s 50 užívateľmi s použitím súčtovej a súčinovej schémy



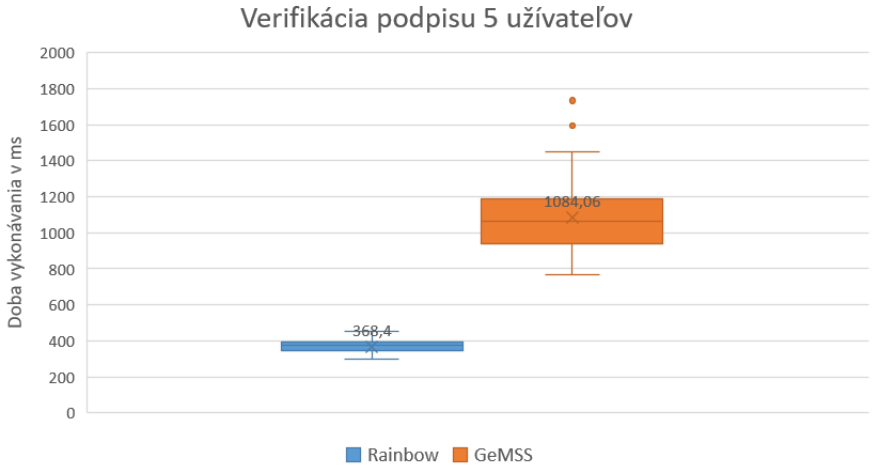
Obrázok 16: Generovanie kľúčov s 50 užívateľmi s použitím súčtovej a súčinovej schémy

6.3.5 Porovnanie súčtovej schémy nad GeMSS a Rainbow s 5 užívateľmi

Pri implementácii súčtovej podpisovej schémy nad GeMSS sme použili knižnicu *gf2x*, ktorá urýchlila výpočty, ktoré sa vykonávajú pri generovaní podpisu. Bez použitia tejto knižnice by GeMSS dosahovalo oveľa pomalšie výsledky ako sú uvedené v tabuľke č.6.



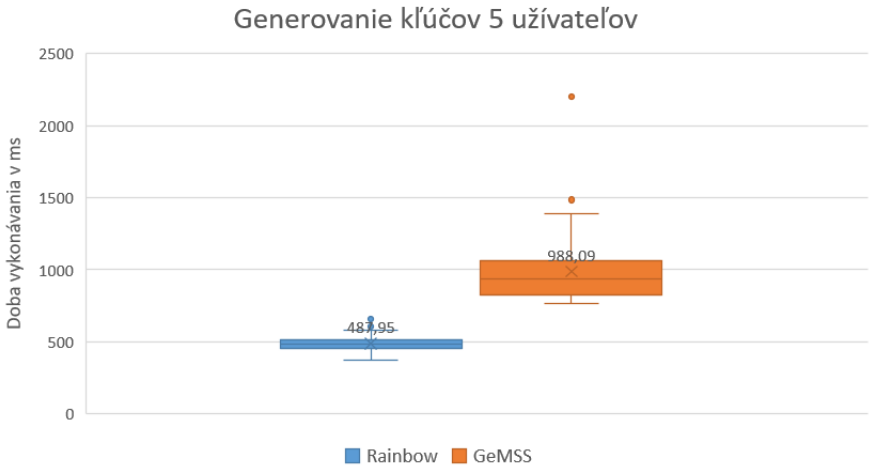
Obrázok 17: Generovanie podpisu s 5 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow



Obrázok 18: Verifikácia podpisu s 5 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow

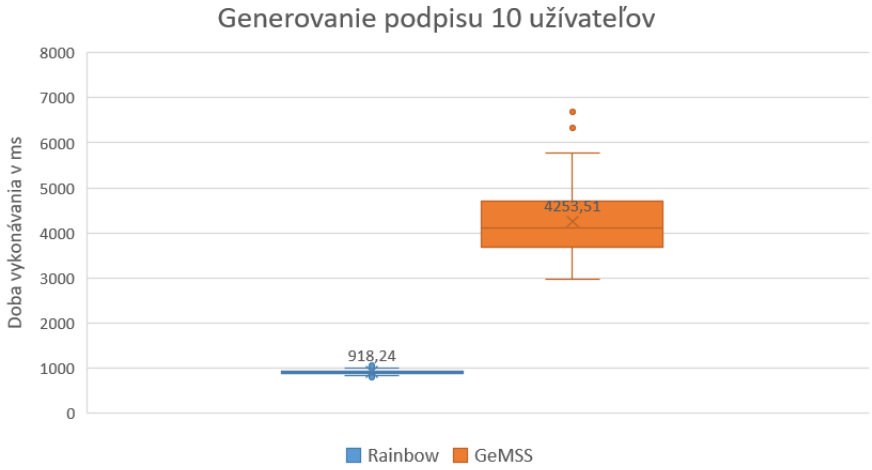
Použitie GeMSS bolo pre nižší počet účastníkov nevýhodné, pretože generovanie podpisu, verifikácia aj generovanie kľúčov prebiehalo s nižšou rýchlosťou ako pri schéme Rainbow. Veľkosť súkromného kľúča sa v GeMSS nemenila, čo je ale spôsobené vlastnosťou implementácie a táto hodnota sa nemení so zmenou parametrov. Na grafoch č.17, č.18 a

č.19 môžeme vidieť, že výrazne efektívnejším algoritmom je vo všetkých troch operáciách Rainbow, pričom v grafoch je zapísaný priemer časov jednotlivých operácií. Reprezentácia celého súboru je znázornená boxplotom, kde modrou farbou je vyznačená súčtová podpisová schéma nad Rainbow a oranžovou farbou je znázornená súčtová schéma nad GeMSS. Z grafov si môžeme všimnúť, že pri použití algoritmu GeMSS je box vyšší ako pri algoritme Rainbow, čo je spôsobené väčšou rozdielnosťou dát.



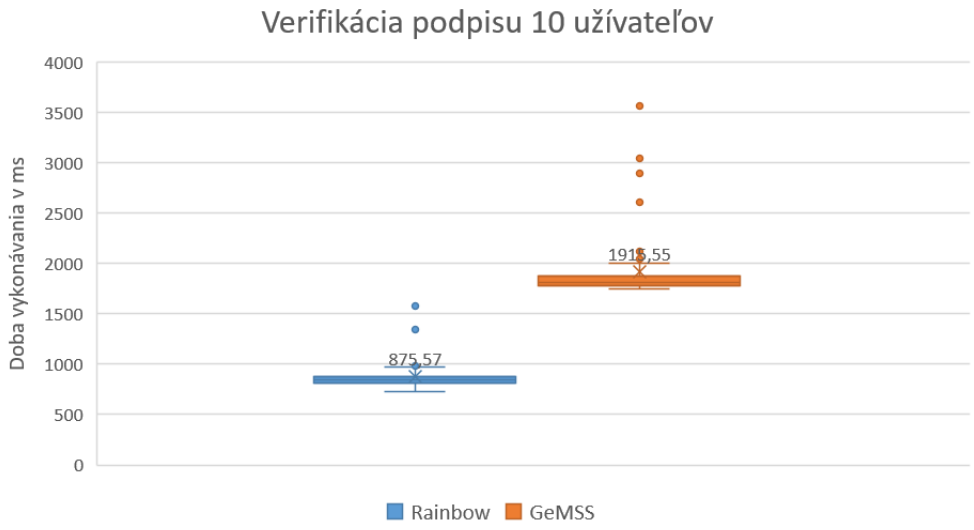
Obrázok 19: Generovanie kľúčov s 5 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow

6.3.6 Porovnanie súčtovej schémy nad GeMSS a Rainbow s 10 užívateľmi

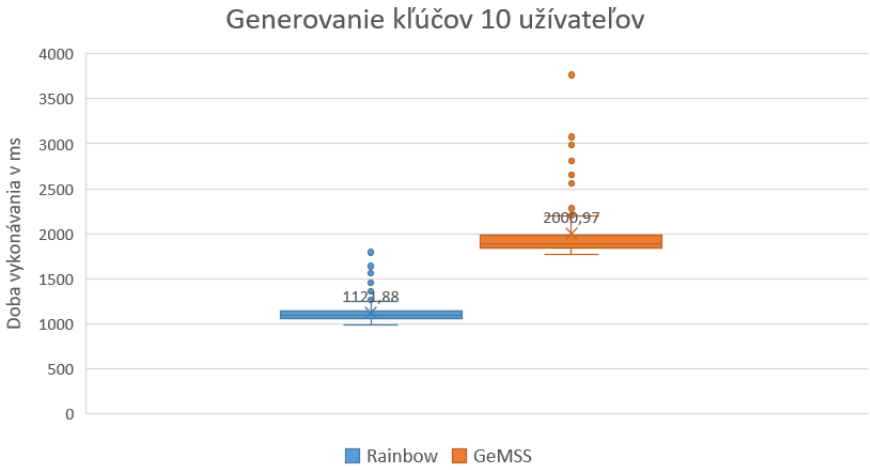


Obrázok 20: Generovanie podpisu s 10 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow

S 10 užívateľmi stále zostáva rýchlejší algoritmus Rainbow, čo naznačujú aj grafy č.20, č.21 a č.22. Na grafoch si môžeme všimnúť, že oproti 5 užívateľom sa boxy v prípade GeMSS viac sploštili.



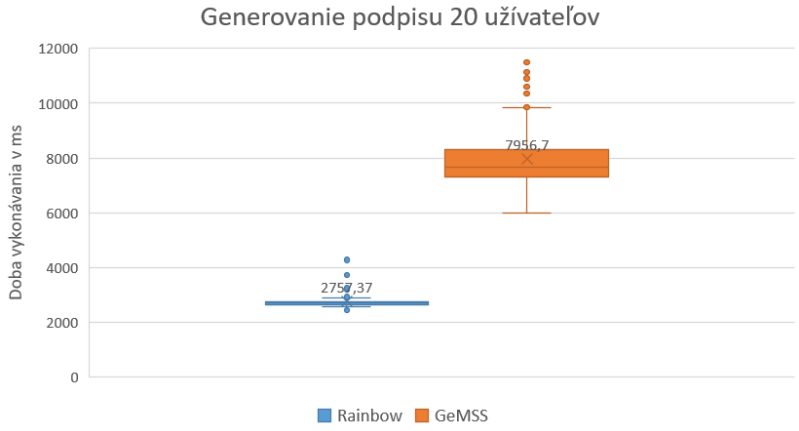
Obrázok 21: Verifikácia podpisu s 10 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow



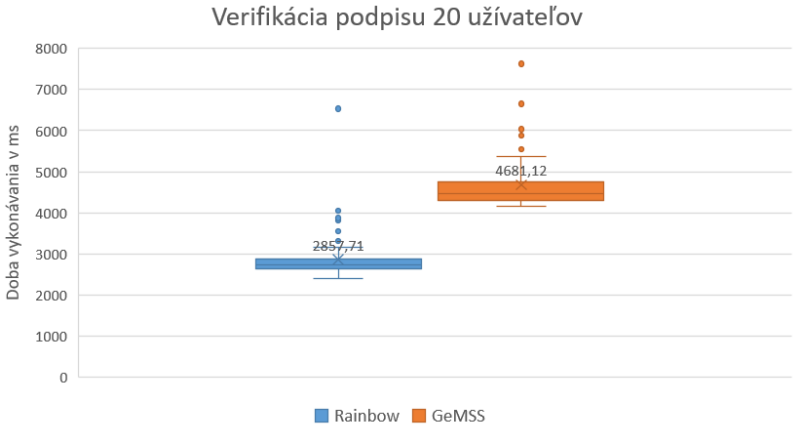
Obrázok 22: Generovanie kľúčov s 10 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow

6.3.7 Porovnanie súčtovej schémy nad GeMSS a Rainbow s 20 užívateľmi

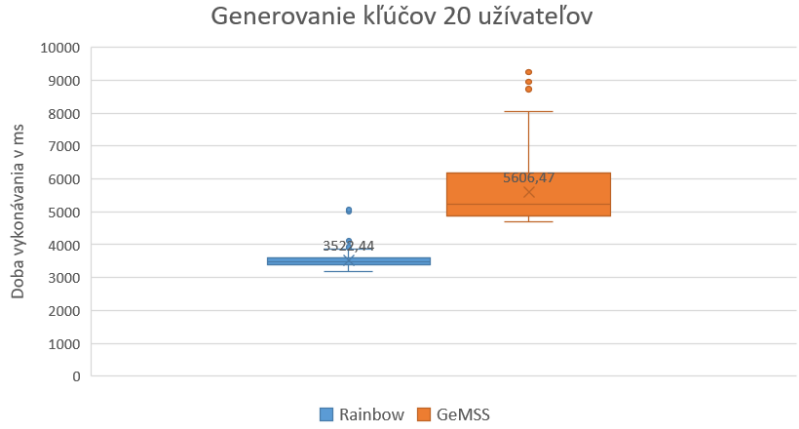
Pri porovnávaní GeMSS a Rainbow s 20 užívateľmi si môžeme všimnúť z grafu č.25, že tieto dva boxy sú bližšie pri sebe v porovnaní s ostatnými prípadmi, teda časy generovania



Obrázok 23: Generovanie podpisu s 20 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow



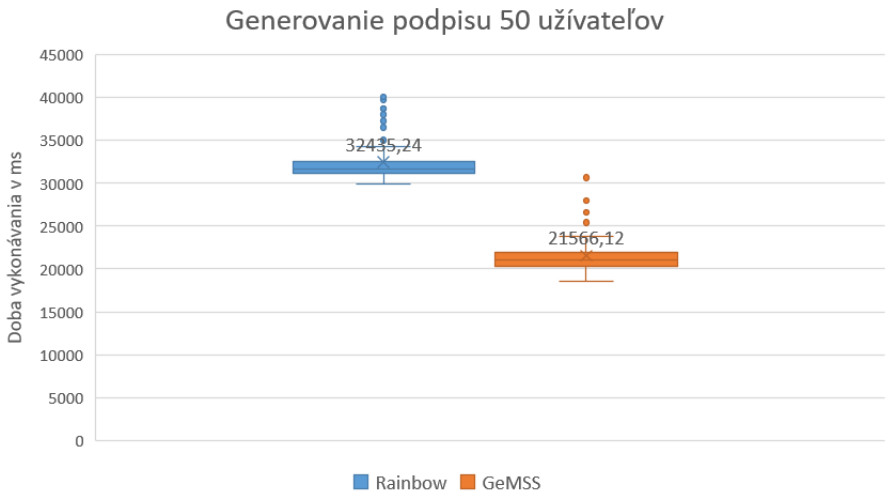
Obrázok 24: Verifikácia podpisu s 20 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow



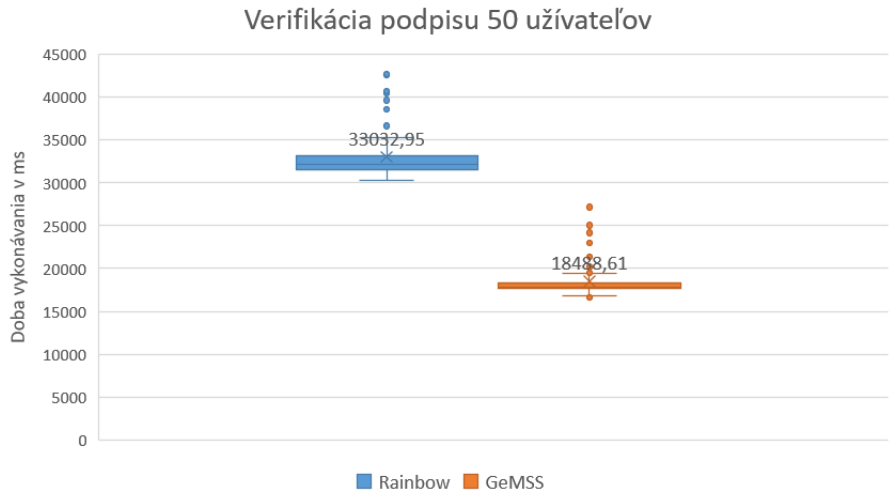
Obrázok 25: Generovanie kľúčov s 20 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow

kľúčov sú pre tento prípad podobné. Z grafov č.24 a č.23 je zrejmé, že medzi Rainbow a GeMSS sú priepastné rozdiely v dobe vykonávania tvorby podpisu a verifikácie podpisu.

6.3.8 Porovnanie súčtovej schémy nad GeMSS a Rainbow s 50 užívateľmi

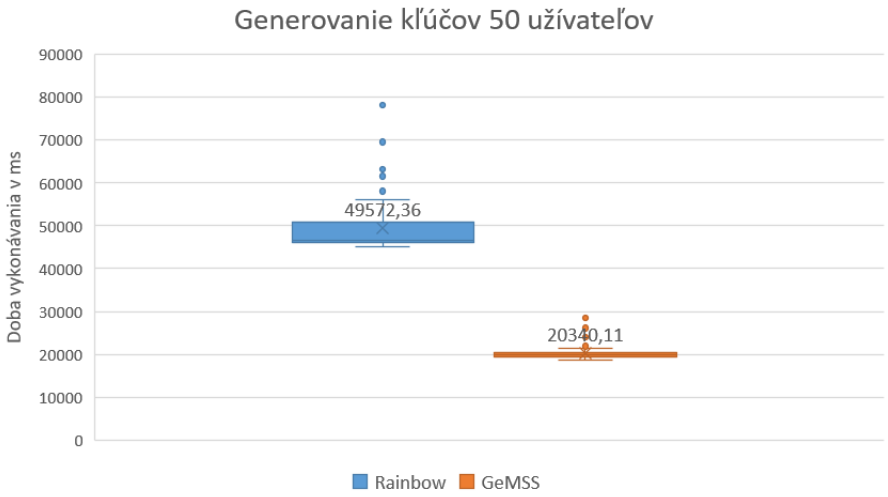


Obrázok 26: Generovanie podpisu s 50 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow



Obrázok 27: Verifikácia podpisu s 50 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow

Pri 50 užívateľoch nastal zlom a priemerné časy GeMSS prekonali Rainbow. Oproti 5, 10 a 20 užívateľom je efektívnejšia súčtová schéma s použitím GeMSS oproti použitiu Rainbow. Na grafoch č.20, č.21 a č.22 je zreteľne vidieť, že GeMSS je efektívnejší vo



Obrázok 28: Generovanie kľúčov s 50 užívateľmi s použitím súčtovej schémy nad GeMSS a Rainbow

všetkých operáciách s 50 užívateľmi, z čoho usudzujeme, že GeMSS je vhodnejšie použiť pre veľký počet účastníkov. Zaujímavé je, že rozdiely medzi časmi sú priepastné a ani v jednom experimente sa k sebe nepriblížili.

Záver

V tejto práci sme skúmali použitie prstencových podpisových schém s použitím kryptosystému Rainbow a GeMSS. Nielen že sme experimentovali so súčtovou a súčinovou schémou, porovnali sme aj použitie súčtovej podpisovej schémy nad Rainbow a GeMSS, pričom tieto algoritmy patria do kategórie kryptosystémov založených na sústave polynómov viacerých neučitých.

Z výsledkov vyplýva, že súčinová podpisová schéma nad Rainbow bola pre viac účastníkov efektívnejšia, čo sa týka doby vykonania generovania podpisu a verifikácie, ale taktiež aj vo veľkosti verejných kľúčov a vygenerovaných podpisov. Hlavným dôvodom úspechu boli nemenné veľkosti parametrov Rainbowu pri súčinovej podpisovej schéme, pretože celková bezpečnosť súčinovej podpisovej schémy sa nezmení ani s pribúdajúcim počtom účastníkov, čo ale neplatí pri súčtovej podpisovej schéme. Aj tento fakt prispieva k tomu, že súčinová podpisová schéma je výhodnejšia pre použitie do praxe, keďže sa parametre Rainbowu nemusia prepočítavať pri zmene počtu účastníkov. Súčtová podpisová schéma dosahuje výborné výsledky hlavne pri nižšom počte účastníkov (tzn. do 10), ale so zvyšujúcim sa počtom účastníkov sa výkonnostne parametre schémy zhoršujú.

Vrámci porovnania použitia súčtovej podpisovej schéme nad Rainbow a GeMSS, výrazne efektívnejšie výsledky podala súčtová podpisová schéma nad Rainbow, avšak pri 50 účastníkoch GeMSS výrazne prekonalo Rainbow aj v dobe vykonávania všetkých operácií, ale aj vo veľkosti verejných kľúčov. Z tohto pokusu preto vyplýva, že súčtová podpisová schéme je efektívnejšia s väčším počtom účastníkov pre GeMSS, pretože s použitím Rainbow výrazne zaostáva. Je dôležité poznamenať, že bez knižnice gf2x by schéma nad GeMSS nepodala také efektívne výsledky pri viacerých účastníkoch ako to bolo pri schéme Rainbow.

Do budúcnosti by sme chceli skúmať aj použitie súčinovej podpisovej schéme nad GeMSS, ale aj experimentovať s inými prstencovými podpisovými schémami, ktoré budú aplikovateľné na akýkoľvek podpisovací algoritmus.

Zoznam použitej literatúry

[1] Bender, A., Katz, J., and Morselli, R. Ring signatures: Stronger definitions, and constructions without random oracles. In *Theory of Cryptography* (Berlin, Heidelberg, 2006), S. Halevi and T. Rabin, Eds., Springer Berlin Heidelberg, pp. 60–79.

[2] Bernstein, Daniel J. and Buchmann, Johannes and Dahmen, Erik. *Post Quantum Cryptography*. 1st. Springer Publishing Company, Incorporated.

[3] Bogdanov, A., Eisenbarth, T., Rupp, A., and Wolf, C. Time-area optimized public-key engines: -cryptosystems as replacement for elliptic curves?. pp. 45–61.

[4] Casanova, A., and et.al. A great multivariate short signature. Submission to NIST PQC "competition" Round-3.

[5] Ding, J., and Schmidt, D. Rainbow, a new multivariable polynomial signature scheme. vol. 3531, pp. 164–175.

[6] DING Jintai, PETZOLDT Albrecht, SCHMIDT S. Dieter. *Multivariate Public Key Cryptography*. 2. Springer US, 2020. ISBN: 978-1-0716-0987-3.

[7] Hromada, V. Voľba parametrov podpisovej schémy gemss pre použitie v prstencovej podpisovej schéme. `https://uim.fei.stuba.sk/wp-content/uploads/2021/03/GeMssSkupina.pdf`.

[8] Mohamed, M., and Petzoldt, A. Ringrainbow – an efficient multivariate ring signature scheme. pp. 3–20.

[9] Rivest, R. L., Shamir, A., and Tauman, Y. How to leak a secret. In *Advances in Cryptology — ASIACRYPT 2001* (Berlin, Heidelberg, 2001), C. Boyd, Ed., Springer Berlin Heidelberg, pp. 552–565.

[10] Shor, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing 26*, 5 (Oct 1997), 1484–1509.

[11] Stinson, Dougles Robert and Paterson, Maura. *Cryptography : Theory and Practice*. 4. Boca Raton : Chapman and Hall/CRC, 2018.

[12] Wolf, C., Braeken, A., and Preneel, B. On the security of stepwise triangular systems.

[13] Wolf, C., and Preneel, B. Asymmetric cryptography: Hidden field equations. Cryptology ePrint Archive, Report 2004/072, 2004. https://eprint.iacr.org/2004/072.

[14] Wolf, C., and Preneel, B. Taxonomy of public key schemes based on the problem of multivariate quadratic equations, 2005. Christopher.Wolf@esat.kuleuven.be 13132 received 12 Mar 2005, last revised 15 Dec 2005.

Prílohy

A Používateľská príručka

Pre jednoduchú inštaláciu a rozbehanie testovacej aplikácie odporúčame užívateľovi postupovať podľa nasledovného návodu:

1. **Prerekvizity**

   - Knižnica *openssl*
   - Knižnica *XKCP*, toto je statická knižnica dostupná na `https://github.com/XKCP/XKCP`
   - (Voliteľná) Knižnica *gf2x*, používa ju implementácia GeMSS je pribalená už priamo v zdrojovom kóde aj s návodom

2. **Práca s aplikáciou**

   V diplomovej práci sú dve prílohy zdrojových kódov. V súbore *Rainbow.zip* sú zdrojové kódy súčtovej a súčinovej prstencovej podpisovej schémy určené na generovanie kľúčov, podpisu a verifikácie nad schémou Rainbow a v súbore *GeMSS.zip* sú zdrojové kódy súčtovej prstencovej podpisovej schémy určené generovanie kľúčov, podpisu a verifikácie nad schémou GeMSS.
   Kompilácia sa spustí príkazom `make`. Vymazanie .o súborov sa spustí príkazom `make clean`.
   Pri spustení aplikačných súborov(.exe) bez parametrov, aplikácia vypíše pomocnú informáciu ohľadom parametrov.

   Spustiteľné súbory v aplikácii Rainbow:

   - `rainbow-ring-genkey pk sk [random_seed_file]`
     - Príkaz na vygenerovanie páru kľúčov s použitím Rainbow
     - *pk* - názov súboru, kde má byť uložený vygenerovaný verejný kľúč
     - *sk* - názov súboru, kde má byť uložený vygenerovaný súkromný kľúč
     - *random_seed_file* - voliteľný parameter, súbor s inicializačným vektorom
   - `rainbow-ring-sign message_file_name sk pk | tee signature`
     - Príkaz na vygenerovanie podpisu s použitím súčtovej podpisovej schémy nad Rainbow
     - *message_file_name* - textový súbor so správou, ktorá bude podpísaná

  - *pk* - súbor s verejným kľúčom
  - *sk* - súbor so súkromným kľúčom
  - *signature* - názov súboru, kde má byť uložený vygenerovaný podpis
- `rainbow-ring-verify signature_file_name message_file_name pk`
  - Príkaz na verifikáciu podpisu s použitím súčtovej podpisovej schémy nad Rainbow
  - *message_file_name* - textový súbor so správou, ktorá bola podpísaná
  - *pk* - súbor s verejným kľúčom
  - *signature_file_name* - súbor s vygenerovaným podpisom
- `rainbow-multiply-sign message_file_name sk pk | tee signature`
  - Príkaz na vygenerovanie podpisu s použitím súčinovej podpisovej schémy nad Rainbow
  - *message_file_name* - textový súbor so správou, ktorá bude podpísaná
  - *pk* - súbor s verejným kľúčom
  - *sk* - súbor so súkromným kľúčom
  - *signature* - názov súboru, kde má byť uložený vygenerovaný podpis
- `rainbow-multipy-verify signature_file_name message_file_name pk`
  - Príkaz na verifikáciu podpisu s použitím súčinovej podpisovej schémy nad Rainbow
  - *message_file_name* - textový súbor so správou, ktorá bola podpísaná
  - *pk* - súbor s verejným kľúčom
  - *signature_file_name* - súbor s vygenerovaným podpisom

  Parametre (v_1, o_1, o_2) sú konfigurovateľné v súbore `rainbow_config.h`. Počet užívateľov je definovaný makrom `USERS` a nachádza sa v súbore `api.h`.

Spustiteľné súbory v aplikácii GeMSS:

- `gemss_ring_keypair pk sk`
  - Príkaz na vygenerovanie páru kľúčov s použitím GeMSS
  - *pk* - názov súboru, kde má byť uložený vygenerovaný verejný kľúč
  - *sk* - názov súboru, kde má byť uložený vygenerovaný súkromný kľúč
- `gemss_ring_sign message_file_name sk pk`

  - Príkaz na vygenerovanie podpisu s použitím súčtovej podpisovej schémy nad GeMSS, vygenerovaný podpis bude uložený do súboru s názvom *signature*
  - *message_file_name* - textový súbor so správou, ktorá bude podpísaná
  - *pk* - súbor s verejným kľúčom
  - *sk* - súbor so súkromným kľúčom
- `gemss_ring_verify message_file_name signature_file_name pk`
  - Príkaz na verifikáciu podpisu s použitím súčtovej podpisovej schémy nad GeMSS
  - *message_file_name* - textový súbor so správou, ktorá bola podpísaná
  - *pk* - súbor s verejným kľúčom
  - *signature_file_name* - súbor s vygenerovaným podpisom

  Parametre (n, Δ, v) sú konfigurovateľné v súbore `parameters_HFE.h`. Počet užívateľov je definovaný makrom `USERS` a nachádza sa v súbore `api.h`.

B Zdrojové súbory

Pri implementácii sme vychadzali zo zdrojových kódov Rainbow dostupný na `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions` zo dňa 6.5.2021.
Naše explicitné zmeny v implementácii Rainbow spočívajú v:

- rainbow-ring-genkey.c
- rainbow-ring-sign.c
- rainbow-ring-verify.c
- rainbow-multiply-sign.c
- rainbow-multiply-verify.c
- v súbore sign.c funkcie:
  - crypto_sign_ring
  - crypto_sign_multiply
  - rainbow_verify_ring
  - rainbow_verify_multiply
- v súbore rainbow.c funkcie:
  - rainbow_sign_ring
  - rainbow_sign_multiply
  - rainbow_verify_mul

Pri GeMSS sme vychádzali zo zdrojových kódov dostupných na `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions` zo dňa 6.5.2021.
Naše explicitné zmeny v implementácii GeMSS spočívajú v:

- gemss_ring_keypair.c
- gemss_ring_sign.c

- gemss_ring_verify.c
- v súbore sign.c funkcie:
  - crypto_sign_ring
  - crypto_sign_open_ring
- v súbore signHFE.c funkcia: signHFE_FeistelPatarin_ring