VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019 Lukáš Matula

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

POST-KVANTOVÁ KRYPTOGRAFIE NA OMEZENÝCH ZAŘÍZENÍCH
POST-QUANTUM CRYPTOGRAPHY ON CONSTRAINED DEVICES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
Lukáš Matula

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Lukáš Malina, Ph.D.

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Informační bezpečnost**
Ústav telekomunikací

Student: Lukáš Matula ***ID:*** 175080
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Post-kvantová kryptografie na omezených zařízeních

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s metodami post-kvantové kryptografie. Zaměřte se především na metody pro ustanovení klíčů a digitálního podpisu. Analyzujte metody post-kvantové kryptografie a zhodnoťte jejich praktickou použitelnost na omezených zařízeních, jako jsou čipové karty. Vyberte vhodné schémata a metody pro omezená zařízení a porovnejte jejich paměťovou a výpočetní náročnost. Připravte verifikační implementaci nejvhodnější metody ve vybraném programovatelném jazyce.
Hlavním cílem bakalářské práce bude funkční implementace postkvantové kryptografické metody na vybrané platformě čipových karet a zhodnocení výpočetní a paměťové náročnosti na čipové kartě.

DOPORUČENÁ LITERATURA:

[1] ALKIM, Erdem, DUCAS, Léo, POPPELMANN, Thomas, SCHWABE, Peter. Post-quantum Key Exchange-A New Hope. USENIX Security Symposium, s. 327-343. 2016.

[2] CHEN, Lily, et al. Report on post-quantum cryptography. US Department of Commerce, National Institute of Standards and Technology, 2016.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Lukáš Malina, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

V posledních letech dochází k velkému technologickému vývoji, který mimo jiné přináší návrhy a realizace kvantových počítačů. V případě využití kvantových počítačů je dle Shorova algoritmu velmi pravděpodobné, že matematické problémy, o které se opírají dnešní kryptografické systémy, budou vypočitatelné v polynomiálním čase. Je tedy nezbytné věnovat pozornost vývoji post-kvantové kryptografie, která je schopna zabezpečit systémy vůči kvantovým útokům. Práce zahrnuje souhrn a porovnání různých typů post-kvantové kryptografie a následně měření a analyzování jejich náročnosti za účelem implementace na omezená zařízení, jako jsou čipové karty. Měřené hodnoty na PC jsou využity na určení nejvhodnější implementace na čipovou kartu a poté je samotná verifikační metoda na čipovou kartu implementována.

KLÍČOVÁ SLOVA

Post-kvantová kryptografie, kvantový počítač, AES, RSA, ECDH, ECDSA, DSA, SHA, NTRU, NP-těžký, SVP, CVP, LWE, Ring-LWE, Rainbow, NewHope, Kyber, Frodo, SIKE, McEliece, Picnic, EEPROM, RAM, verifikace, autentizace, omezené zařízení, čipová karta, implementace, šifrování, dešifrování

ABSTRACT

In recent years, there has been a lot of technological development, which among other things, brings the designs and implementation of quantum computing. Using Shor's algorithm for quantum computing, it is highly likely that the mathematical problems, which underlie the cryptographic systems, will be computed in polynomial time. Therefore, it is necessary to pay attention to the development of post-quantum cryptography, which is able to secure systems against quantum attacks. This work includes the summary and the comparison of different types of post-quantum cryptography, followed by measuring and analysing its levels of difficulty in order to implement them into limited devices, such as smart cards. The measured values on the PC are used to determine the most suitable implementation on the circuit card and then the verification method itself is implemented on it.

KEYWORDS

Post-quantum cryptography, quantum computer, AES, RSA, ECDH, ECDSA, DSA, SHA, NTRU, NP-hard, SVP, CVP, LWE, Ring-LWE, Rainbow, NewHope, Kyber, Frodo, SIKE, McEliece, Picnic, EEPROM, RAM, verification, authentization, constrained device, smart card, implementation, encryption, decryption

MATULA, Lukáš. *Post-kvantová kryptografie na omezených zařízeních*. Brno, 2019, 57 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Lukáš Malina, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Post-kvantová kryptografie na omezených zařízeních“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Lukáši Malinovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Dnešní kryptografie je založena na matematických problémech, jako je výpočet prvočíselného rozkladu nebo diskrétního logaritmu. Tyto matematické problémy jsou v dnešní době nemožné vyřešit v polynomiálním čase, jsou tedy považovány za bezpečné a jsou schopny zabezpečit systémy proti klasickým útokům. V případě využití kvantového počítače dle Shorova algoritmu je však velmi pravděpodobné, že zmíněné problémy, na kterých je založená bezpečnost dnešní kryptografie, budou vypočitatelné v polynomiálním čase. Vzniká tedy post-kvantová kryptografie, která popisuje protokoly a kryptosystémy, které jsou odolné vůči kvantovým počítačům. Zájem odborníků a vědců v posledních letech roste a přispívá tím k budování novějším a efektivnějším protokolům, kterými bude možno zabezpečit zařízení a systémy proti kvantovým počítačům. Budované protokoly odolné vůči kvantovým počítačům jsou založeny na různých typech kryptografie využívající rozličných matematických problémů. Kvůli rozličnosti druhů kryptografií a matematických problémů nelze jasně určit, který z kryptosystémů je nejefektivnější pro jaké zařízení.

V současné době je výzkum post-kvantové kryptografie orientován na implementaci a analyzování protokolů především na PC, noteboocích či serverech, nikoliv však na velmi omezených zařízeních. Práce se zaměřuje na post-kvantovou kryptografii na omezených zařízení, jako jsou čipové karty. Čipové karty jsou hojně využívaná zařízení pro služby, kde je nutná autentizace, jako je například karta bankovního účtu. Kryptografický protokol potom chrání výměnu informací mezi čipovou kartou a autentizačním zařízením, jako je terminál. Na tento typ zařízení je tedy kladen velký důraz z pohledu bezpečnosti, je ovšem výpočetně i paměťově omezený. Tato omezení mohou být překážkou pro implementaci post-kvantových protokolů.

Práce je logicky rozdělena do několika kapitol. První kapitola je věnována samotnému představení termínu post-kvantové kryptografie a popsání typů kryptografie odolné vůči kvantovým počítačům. Ve druhé kapitole je pozornost věnována protokolům, které jsou ve třetí kapitole srovnány a na základě měření je určen typ kryptografie, který je vhodný pro implementaci na omezená zařízení. Třetí kapitola se věnuje porovnání zmíněných protokolů a určení nejvhodnější metody na omezená zařízení s návrhem její modifikace. V poslední kapitole je s ohledem na verifikační metodu vybrána čipová karta, dále jsou zde popsány dílčí funkce metody a jejich implementace na samotné zařízení.

1 Post-kvantová kryptografie

V praxi využívaná asymetrická kryptografie je založena na problémech z teorie čísel. Mezi problémy, využívané asymetrickou kryptografií, patří faktorizace čísel a problém diskrétního logaritmu. Systémy založené na těchto problémech jsou výbornou volbou v mnoha aplikacích a jejich bezpečnost je dobře definována a chápána. Jejich hlavní nevýhodou je však fakt, že v případě dostupnosti kvantových počítačů patřičných kapacit budou zranitelné. Kvantové počítače jsou totiž schopny problémy faktorizace čísel a diskrétního logaritmu vyřešit v polynomiálním čase, čímž jsou algoritmy využívající těchto problémů v budoucnosti nepoužitelné. Toto tvrzení je podloženo důkazem z roku 1994 amerického profesora aplikované matematiky Petera Shora, na základě kterého je popsán Shorův algoritmus. Tento algoritmus je do budoucna schopen vypočítat problémy, které byly doposud považovány za nerešitelné. Proto je studium post-kvantové kryptografie, která popisuje algoritmy, proti nimž není znám žádný efektivní útok některým z kvantových algoritmů, velmi důležité. Post-kvantová kryptografie má obrovskou výhodu, že jí popisující algoritmy jsou použitelné již na stávajících počítačích, tudíž ji lze považovat za takovou „prevenci proti kvantovým počítačům“.

Rozdíl mezi kvantovým počítačem a dnešní výpočetní technikou je takový, že dnešní počítače pracují s čísly, která jsou reprezentována bity. Bity jsou zapsány v binární soustavě, tudíž mohou nabývat pouze hodnot 0 nebo 1. Číslo je tedy možno reprezentovat v rozsahu mezi 0 a $2^n - 1$, kde n je maximální počet bitů. Pokud máme tedy k dispozici 4 bity, můžeme pomocí nich reprezentovat čísla z rozsahu 0 – 15. Kvantové počítače se neváží pouze na hodnoty 0 a 1, namísto toho mají vlastní superpozici skládající se ze dvou hodnot, které nabývají libovolných hodnot mezi 0 a 1 s odlišnou pravděpodobností.

Tab. 1.1: Dopad kvantových výpočtů na běžně využívané kryptografické algoritmy [1]

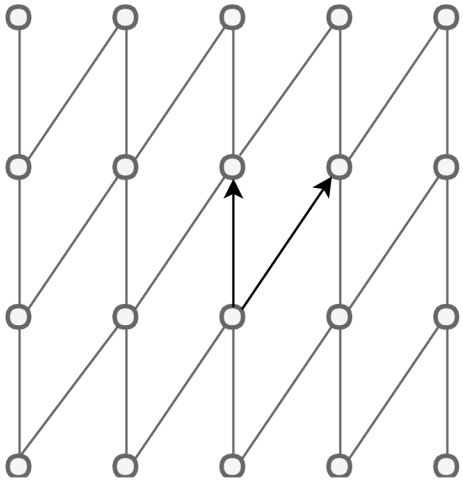
Kryptografický algoritmus	**Typ**	**Účel**	**Dopad kvantového počítače**
AES	Symetrický	Symetrické šifrování	Potřeba větší velikosti klíče
SHA-2, SHA-3	Hashovací funkce	Jednosměrná kompresní funkce	Potřeba použití delšího výstupu
RSA	Asymetrický	Podpis, Ustanovení klíčů	Není bezpečný
ECDSA, ECDH	Asymetrický	Podpis, Ustanovení klíčů	Není bezpečný
DSA	Asymetrický	Podpis, Ustanovení klíčů	Není bezpečný

V tabulce 1.1 jsou srovnány dnes běžně využívané algoritmy s dopadem nasazení

kvantových počítačů. Z tabulky plyne, že algoritmy o které se opírá významná část dnešní kryptografie, jako je digitální podpis a ustanovení klíčů, tedy RSA, ECDSA, ECDH, DSA a další kryptosystémy založeny na podobných matematických problémech, které jsou Shorovým algoritmem vypočitatelné, nejsou do budoucna považovány za bezpečné. V případě algoritmů AES, SHA-2 a SHA-3 lze řešení použitím delšího klíče nebo delšího výstupu považovat za dostatečně bezpečné i navzdory Groverovu algoritmu[1] , který teoreticky nabízí kvantovým počítačům kvadratické zrychlení, díky kterému by delší klíče a výstupy z těchto metod byly možné vypočítat. Kvadratické zrychlení je však z praktického hlediska prozatím nemožné, proto jsou tyto kryptosystémy považovány za bezpečné i proti kvantovým počítačům [1, s. 3-6] [2].

1.1 Kryptografie založená na mřížkách

Kryptografie založená na mřížkách je jeden ze systémů, který je schopen odolat kvantovým útokům. Využívá silné bezpečnostní algoritmy a zároveň je relativně jednoduše a efektivně implementovatelná. Díky těmto parametrům je do budoucna považována za jednu z nejvíce využívaných kryptografických metod.
Jak lze již z názvu odvodit, tato kryptografie využívá mřížky. Mřížka je volně řečeno množina bodů v n-rozměrném prostoru s periodickou strukturou. V případě na obrázku 1.1, tedy 2-dimenzionálního prostoru, si lze danou problematiku představit jako $\mathbb{Z}^2$ množinu bodů.



Obr. 1.1: Příklad mřížky ve dvourozměrném prostoru

[1] Groverův algoritmus se používá k hledání nesetříděné posloupnosti v $\mathcal{O}(n^{\frac{1}{2}})$

Definice 1.1.1 Nechť n je přirozené číslo a $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$ lineárně nezávislé vektory nad $\mathbb{R}$. Pak množinu

$$L = \sum_{i=1}^{n} \mathbb{Z}\mathbf{b}_i = \left\{ \sum_{i=1}^{n} a_i \mathbf{b}_i; a_1,\ldots,a_n \in \mathbb{Z} \right\} \tag{1.1}$$

nazýváme *mřížka* nad vektory $\mathbf{b}_1, \ldots, \mathbf{b}_n$.

Definice 1.1.2 Řekněme, že $\mathbf{b}_1, \ldots, \mathbf{b}_n$ tvoří *bázi mřížky* L, pokud L je mřížka nad těmito vektory. Libovolná podmnožina $M \subseteq \mathbb{R}^n$ je mřížka, pokud existují vektory $\mathbf{b}_1, \ldots, \mathbf{b}_n$ takové, že M je mřížka právě nad $\mathbf{b}_1, \ldots, \mathbf{b}_n$. Číslo n se nazývá *hodnost mřížky* [3].

Při použití této metody se pro zabezpečení používá n jako prvočíslo, které je značeno písmenem q.

Z definice 1.1.2 plyne, že každá mřížka má nekonečně mnoho bází. Neexistuje pravidlo, které by privilegovalo některou z bází před ostatními.

Systémy využívající mřížku jsou považovány za bezpečné z důvodu složitosti problémů s ní spojené. Tyto problémy jsou popsány v následující kapitole.

1.1.1 Využití mřížek v post-kvantové kryptografii

Jak již bylo naznačeno, kryptografie založená na mřížkách je považována za bezpečnou i proti kvantovým útokům, opírá se totiž o dva rozsáhlé matematické problémy, které nejsou ani po delších studiích efektivně řešitelné ani kvantovým počítačem. Jedná se o problém Shortest Vector Problem (dále jen SVP) a Closest Vector Problem (dále jen CVP).

Problém SVP je hlavním problémem týkající se mřížek, je to problém nalezení nejkratšího vektoru báze. Při řešení tohoto problému je zadána mřížka L s libovolnou bází. Cílem je najít nejkratší nenulový vektor, který mřížka obsahuje. [4, s. 370–372]. Hledané řešení je bod mřížky, který nejblíže nule, ale není nulový. Tento princip je využíván například při zabezpečení privátního klíče kryptosystémem NTRU Encrypt[2]. Nejznámějším polynomiálním, široce studovaným algoritmem, zabývajícím se hledáním nejkratšího nenulového vektoru, je např. Lenstra, Lenstra and Lovász algorithm, tedy *LLL algoritmus*, který počítá v čase $2^{n(\log n)^2/\log n}$ anebo Kannanův algoritmus, který počítá v čase $2^{O(n \log n)}$, kde n je dimenze mřížky.

[2]NTRU Encrypt je mřížkově-založená alternativa RSA a kryptografie založené na eliptických křivkách

Problém CVP je problém nalezení nejbližšího vektoru k libovolnému vektoru mřížky. Vektor může být nulový, ale musí splňovat podmínku, že je nejblíže zvolenému bodu. Jelikož s každým vektorem v obsahuje mřížka i vektor -v o stejné velikosti, není řešení SVP jednoznačné. Stejně tak nemusí být jednoznačné ani řešení CVP. V případě obou problémů je známo, že pro velké hodnoty dimenze n jsou v obecném případě NP-těžké [5, s. 2–3].

Oba problémy mají aproximační variantu apprSVP a apprCVP viz [3, s. 7–11]. Jediný z kvantových algoritmů, který by teoreticky mohl zlomit problémy mřížky, je Shorův algoritmus, u kterého to ale nebylo doposud dokázáno. Kryptografie založená na mřížkách je tedy při použití správných parametrů, jak již bylo zmíněno, považována za bezpečnou.

V post-kvantové kryptografii se na základě CVP a SVP zakládají dva konkrétní matematické problémy. Jedná se o problémy LWE (Learning With Errors) a Ring-LWE problémy.

Learning With Errors problém (dále jen LWE) – na základě výsledků zmíněných zde [6, s. 1–7] je kladen vysoký důraz na bezpečnost i proti kvantovým útokům kryptografie založené na LWE, navíc je efektivně implementovatelná a zahrnuje operace s nízkou složitostí (především doplňky). LWE se zakládá na složitosti rozlišit distrubuci ($\mathbf{A}$, $\mathbf{AS} + \mathbf{e}$), kde $\mathbf{A}$ je jednotná náhodná matice v $\mathbb{Z}_q^{m \times n}$, $\mathbf{s}$ jednotný náhodný vektor v $\mathbb{Z}_q^n$ a $\mathbf{e}$ vektor s menšími koeficienty zvolenými z dané distribuce. Náhodný vektor $\mathbf{s}$ může být i z velmi blízké distribuce, jako vektor $\mathbf{e}$, přičemž složitost zůstane zachována [7, s. 16].

Ring-LWE problém přidává k problému LWE problematiku polynomiálních okruhů nad konečným polem. Ring-LWE pracuje na základě ideálních mřížek (LWE na základě generických mřížek), čímž se redukuje výpočetní náročnost. LWE rozšířený o polynomiální okruhy, tedy Ring-LWE, se jeví z hlediska času mnohem efektivnější. Ideální mřížky s sebou přináší další matematické problémy jako SIS (Short integer solution problem) a Ring-SIS dále popsány zde [8]. Ring-LWE je využíván například kryptosystémem NTRU. V této práci jsou představeny protokoly využívající Ring-LWE i čistý LWE problém a jsou mezi sebou srovnány.

Post-kvantových kryptosystémů založených na mřížkách je velké množství, mezi ty nejznámější patří například systémy New Hope, Frodo, Kyber a již zmiňovaný NTRU.

1.2 Kryptografie založená na teorii kódování

Teorie kódování je teorie, která v sobě obsahuje postupy a metody informatiky, matematiky a spojovací techniky. Úlohou teorie kódování je tvorba postupů a metod, které zajišťují bezpečný přenos zpráv komunikačním kanálem a také kontrola, zdali přenesená data byla nějakým způsobem ztracena, nebo znehodnocena. Ve většině případů je kódování prováděno tak, že se k bitům dat přidají další, tzv. kontrolní bity, díky kterým si v kombinaci s bity dat můžeme ověřit, zdali došla zpráva v pořádku. Existují i tzv. „Samoopravné kódy“, které díky různým matematickým funkcím dokáží detekovat chybu, její místo výskytu a následně ji opravit – např. při doručení zprávy zahájí kontrolu pomocí kontrolních bitů, příznak chyby indikuje, že je chyba obsažena, příznak výskytu chyby (vektor bitů) prozradí na jaké pozici je chybný bit a ten opraví tím, že jeho hodnotu logicky zneguje. Mezi samoopravné kódy se řadí například Hammingův kód, který byl stručnou demonstrací představen již výše.

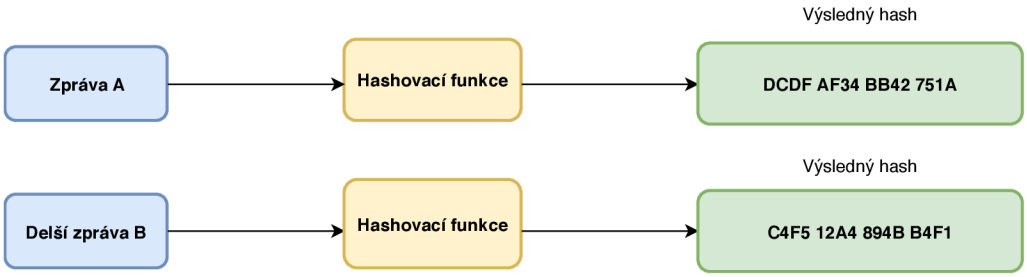
1.2.1 Využití kódování v post-vantové kryptografii

Kryptosystémy využívající teorii kódování, které jsou schopny odolávat vůči útokům kvantových počítačů, používají takzvané Goppa kódy [9][10]. Goppa kód je kód odvozený z algebraických křivek nad konečným tělesem. Proto jsou tyto kódy mnohdy označovány jako *geometrické kódy*. Kryptosystémy tohoto typu jsou bezpečné jen v případě dodržení určitých parametrů. Jednou z hlavních nevýhod těchto systémů je potřeba velkých privátních a veřejných klíčů, které jsou ve formě matice a jejich velikost dosahuje až stovek tisíc bytů. Pro porovnaní lze například uvést, že v případě 1024-bitového modulu RSA je potřeba 256 bytového veřejného klíče [11]. Jejich výhodou je však rychlost, která je vyšší, než u zmíněného RSA modulu. Mezi kryptosystémy založené na teorii kódování určené k zabezpečení proti kvantovým počítačům se především řadí kryptosytémy McEliece, McBits a Niederreitův kryptosystém, vycházející z McEliece.

1.3 Kryptografie založená na hashovacích funkcích

Hashovací funkce je jednosměrná matematická funkce, která mapuje řetězec libvolné délky (zprávu, datový soubor) na řetězec konstantní délky a vytvářejí tak otisk vstupního řetězce viz obrázek 1.2. Výsledný otisk se označuje jako hash a je závislý na všech bitech vstupního řetězce. Tyto funkce se využívají mimo jiné ke kontrole integrity dat, porovnávání dvojice zpráv, k vyhledávání a indexování, ale především se ve světě kryptografie využívají pro tvorbu digitálních podpisů. Každá hashovací

funkce není v principu injektivní (prostá), může tedy existovat více zpráv poskytující stejný hash.



Obr. 1.2: Proces hashovacích funkcí

Formálně jde o funkci h, která převádí vstupní posloupnost bitů na posloupnost pevné délky n bitů.

$$h : D \longrightarrow R, \tag{1.2}$$

kde $|D| >> |R|$.

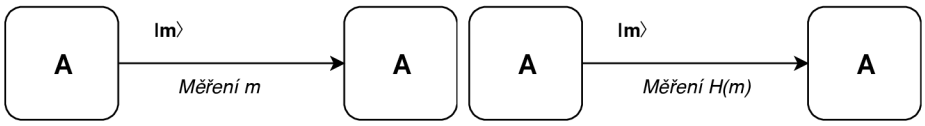
Hashovací funkce musí splňovat následující požadavky:

(a) Funkce musí být jednosměrná, nelze vypočítat x takové, aby $H(x) = c$, tedy lze vypočítat hash ze zprávy, ale nelze vypočítat zpráva z hashe.
(b) Pro výstup x je náročné vypočítat y, aby platilo $H(x) = H(y)$.
(c) Je třeba, aby byl odolný vůči kolizím. Je tedy těžké naléze dvojici zpráv (x, y) tak, aby platilo $H(x) = H(y)$.
(d) Jakékoliv množství vstupních dat generuje vždy stejně dlouhý výstup (konstantní velikosti).

1.3.1 Využití hashovacích funkcí v post-kvantové kryptografii

Kvůli možnostem kvantových počítačů je odolnost hashovacích funkcí založených na nemožnosti nalezení kolize výrazně nižší. Nehledají se totiž dvě různé hodnoty stejného hashe, ale dvě různé superpozice stejného hashe. Tento problém řeší posílená zabezpečovací hashovací funkce, tzv. „collapsed“ hashovací funcke. Collapsed hashovací funkce byla v roce 2016 představena profesorem Unruhem a počítá s tím, že útočník sice může znát superpozice hodnoty m, ale s těmito hodnotami mohou být prováděny dvě různé operace. Buď dochází k měření na základě superpozice m (Obr. 1.3(a)), nebo je vyhodnocována superpozice $H(m)$ (Obr. 1.3(b)). Požadavkem a využitelností této funkce je to, že útočník vlastnící kvanotvý počítač není schopen

v polynomiálním čase rozlišit, zdali je měřeno *H(m)*, nebo *m*, nebo zdali vůbec k nějakému měření dochází. Díky tomuto zabezpečení není možné, aby útočník nalezl původní zprávu. Tyto hashovací funkce jsou tedy i ve srovnání s kvantovými počítači kolizními. Některé ze současných systémů, jako např. SHA-3, tuto podmínku splňují.[12, s. 4–6]



(a) Vyhodnocování zprávy *m*. (b) Vyhodnocování zprávy *H(m)*.

Obr. 1.3: Unhrův návrh vyhodnocování zprávy *m* a *H(m)*.

Obrázek 1.3 záměrně vyobrazuje obě operace identicky, protože není možné určit, která z nich je právě prováděna.

1.4 Kryptografie založená na polynomiálních rovnicích

Kryptografie založená na polynomiálních rovnicích je postavena na algebraické geometrii a opírá se o náročnost řešení soustavy rovnic o více neznámých sestavených nad konečným polem. Jedná se o další z efektivních řešení pro post-kvantovou kryptografii. V této metodě dochází k využívání tzv. jednosměrné funkce se zadními vrátky.

Definice 1.4.1 *Mějme konečné pole k s q počtem prvků. Veřejný klíč je definován funkcí se zadními vrátky* $P : k^n \to k^m$*, kde*

$$P(x_1,\ldots,x_n) = (p_1(x_1,\ldots,x_n),\ldots,p_m(x_1,\ldots,x_n)). \tag{1.3}$$

Jednosměrná funkce se zadními vrátky, která je využívána například v kryptosystému RSA, Rabin aj., hraje velkou roli v problematice násobení prvočísel nabývajících vysokých hodnot. Prvočísla lze velmi jednoduše roznásobit, ale dosažení původního součinu prvočísel je už velmi náročné [13, s. 8–10].

1.4.1 Využití polynomiálních rovnic v post-kvantové kryptografii

V případě post-kvantové kryptografie je využívána tzv. Hidden Field Equations funkce (dále jen HFE) – funkce výpočtu skrytých polí. Funkce počítá s polynomy

nad konečným polem $\mathbb{F}_q$. Tyto polynomy mají různé velikosti, aby byla skryta vazba mezi veřejným a soukromým klíčem. HFE využívá tzv. problém MQ:

Definice 1.4.2 *Mějme m mnohonásobných kvadratických polynomů* $p^1(x),\ldots,p^m(x)$ *a je třeba najít vektor* $\overline{x} = (\overline{x}_1,\ldots,\overline{x}_n)$*, při kterém platí* $p^1(\overline{x}) = \cdots = p^m(\overline{x}) = 0$.

Je dokázáno, že tento problém je NP-těžký, přesto s nevhodně zvolenými hodnotami existují algoritmy, které jsou tento problém schopny vyřešit i na běžném, tedy bitovém počítači [14].

Mezi kryptosystémy využívající polynomiální rovnice lze uvést např. kryptosystémy Rainbow, Unbalanced Oil and Vinegar, nebo Matsumoto-Imai kryptosystém.

1.5 Kryptografie založená na supersingulárních eliptických křivkách

Na využití supersingulárních křivek v oboru kryptografie je kladen větší důraz od doby, kdy je známo, že kvantové počítače budou schopny prolomit algoritmy založené na eliptických křívkách. Eliptické křivky jsou totiž založeny na problému diskrétního logaritmu, který je kvantovým počítačem řešitelný. V případě supersingulárních křivek je $E(\mathbb{F}_{\text{ı}})$ definována jako skupina eliptických křivek, která využívá velké endomorfní okruhy. Jednou z podmínek supersingulárních křivek je, že $\#E(\mathbb{F}_{\text{ı}}) = p - 1$, přičemž typicky $p \geq 5$.

1.5.1 Využití supersingulárních eliptických křivek v post-kvantové kryptografii

Hlavní problém, který přispívá k nevypočitatelnosti metody je ten, že na rozdíl od protokolu ECDH, který používá body na jedné křivce, v post-kvantových protokolech využívajících supersingulárních křivek dochází ke generování křivek během výměny klíčů. Tajné klíče jsou v post-kvantové metodě izogenní. To znamená, že je možné body jedné eliptické křivky zobrazit do druhé a zároveň zachovat vrcholy a strukturu samotné křivky. Využívá se zde tedy toho, že lze namapovat více bodů na jeden samostatný bod [15, s. 1–2][16]. Po namapování všech bodů jedné křivky na druhou vzniknou izogenity, které vedou z jedné křivky na druhou a ty jsou využity jako soukromé klíče. Veřejný klíč je samotná supersingulární eliptická křivka. Protkoly využívající supersingulární eliptické křivky s izogeny jsou velmi výpočetně náročné, avšak generují klíče o malých velikostech. Mezi protokoly využívající výše

zmíněné metody patří například protokol Supersingular isogeny Diffie-Hellman key echange (SIDH) – protokol pro výměnu klíčů, nebo Supersingular isogeny Key Encapsulation (SIKE) – protokol pro ustanovení klíčů.

1.6 Shrnutí

V této kapitole byly popsány základní metody a principy, na kterých jsou postaveny protokoly post-kvantové kryptografie. Existuje mnoho protokolů a metod, které využívají výše zmíněné mechanizmy a v některých případech ke své funkcionalitě využívají i více metod dohromady. V tabulce 1.2 jsou probrané metody kryptografie shrnuty. Tabulka je zaměřena zejména na výhody a nevýhody ve vztahu k paměťové a výpočetní náročnosti a je pouze orientační, protože každý protokol dané metody je jinak implementován a záleží tedy na provedení samotného protokolu, který blíže určuje své výhody či nevýhody. Z tabulky tedy nelze odvodit, které výhody či nevýhody nabízí všechny protokoly daného typu. Tabulka spíše slouží k určení obecného trendu které mezi sebou, na základě použité metody, protokoly sdílí.

Tab. 1.2: Výhody a nevýhody metod post-kvantové kryptografie

Typ kryptografie	**Výhody**	**Neyvýhody**	**Příklady protokolů**
Kryptografie založená na mřížkách	nižší výpočetní náročnost	vyšší paměťová náročnost	NTRU, New Hope, Frodo, Kyber, LIMA
Kryptografie založená na teorii kódování	nižší výpočetní náročnost	vyšší paměťová náročnost, větší velikost klíčů	McEliece, Niederreit
Kryptografie založená na hashovacích funkcích	menší velikost klíčů	vyšší paměťová náročnost	Merkle signature scheme, SHA-256, Picnic
Kryptografie založená na polynomiálních rovnicích	nižší výpočetní náročnost	vyšší paměťová náročnost	Rainbow, LUOV, HiMQ-3
Kryptografie založená na supersingulárních eliptických křivkách	nižší paměťová náročnost, menší velikost klíčů	vyšší výpočetní náročnost	SIDH, SIKE

V další kapitole je práce zaměřena na protokoly využívající výše zmíněných metod. V současnosti je v post-kvantové kryptografii nejvíce využívána kryptografie založená na mřížkách. Důvodem je, že je relativně efektivně implementovatelná a zároveň nabízí prostor pro optimalizaci při využívání LWE a Ring-LWE problémů. Nelze však jednoznačně určit, že je tato metoda nejefektivnější, protože různé protokoly využívající Goppa kódy, hashovací funkce, či izogenity lze také v dnešní době efektivně implementovat a v některých ohledech mohou být výhodnější, nežli protokoly využívající mřížky.

Cílem práce je určení vhodného schématu pro implementaci na omezené zařízení. Omezené zařízení je zařízení, které má omezenou paměťovou i výpočetní kapacitu, hlavně z důvodu jeho velikosti. Při výběru popisovaných protokolů je tedy na základě [17], kde jsou metody srovnány z pohledu potřebné velikosti klíčů a zařazeny

do úrovní tzv. NIST-security, tedy NIST-bezpečnosti uvedené v tabulce 1.3, na tato omezení brán ohled a jsou vybrána pouze známá schémata, která byla již v několika pracích představena a testována.

Tab. 1.3: Znázornění úrovně NIST-bezpečnosti na dnešních algoritmech

NIST úroveň zabezpečení	**Protokol**
1	AES-128
2	SHA256, SHA3-256
3	AES-192
4	SHA384, SHA3-384
5	AES-256

Dále zmíněné protokoly jsou postupem času modifikovány a uzpůsobovány pro jejich širší využití a stále přibývají další, které většinou rozšíří již dříve představené. Vývoj jde velmi rychle dopředu, je tedy možné, že se vybrané metody budou ubírat jiným směrem, avšak jejich hlavní rysy (výhody i nevýhody) zůstanou nejspíše stejné na základě neměnitelného jádra metod. V rámci popisů protokolů budou také zmíněny jejich výhody i nevýhody.

2 Vybrané protokoly post-kvantové kryptografie

Tato kapitola je věnována několika známým protokolům určeným pro digitální podpis a ustanovení klíčů, které jsou zde podrobně popsány a následně porovnány. Jedná se především o protokoly, které jsou vhodnými kandidáty pro implementaci na omezená zařízení. Jsou tedy vyloučeny protokoly s velmi vysokou paměťovou i výpočetní náročností. Další část práce se věnuje srovnání zmíněných metod z hlediska jejich implementace na zvolené omezené zařízení s ohledem na jazyk, ve kterém bude metoda implementována.

2.1 NTRU

Protokol NTRU je post-kvantovou alternativou k protokolu RSA, který byl představen již roku 1996. Na rozdíl od RSA a ECC nevyužívá problému prvočíselného rozkladu či problému diskrétního logaritmu. Jak již bylo zmíněno v kapitole 1.1.1, protokol NTRU je založen na problému SVP, konkrétně na Ring-LWE problému. Jednou z výhod protokolu je nízká náročnost na paměť a vyšší rychlost generování klíčů [18]. V porovnání s RSA při ekvivalentní kryptografické síle provádí nákladné operace se soukromými klíči rychleji NTRU.

NTRUEncrypt je asymetrická metoda určená pro šifrování. Klíče jsou generovány v podobě matice a pro jejich vygenerování je třeba tří parametrů N, p a q.

Definice 2.1.1 *Pro parametry p,q platí*

$$gcd(p, q) \cup q > p.$$

Při volbě polynomů f a g v kruhu R je použit parametr N, přičemž f představuje část soukromého klíče a parametr g na základě soukromého klíče generuje veřejný klíč h. Polynomy jsou zvoleny ze skupiny L_g stupně $n-1$ a koeficienty z množiny {-1,0,1}. Dále je počítána inverzní funkce f

$$f_q \equiv f^{-1} (\mathrm{mod}\, q) \tag{2.1}$$

$$f_p \equiv f^{-1} (\mathrm{mod}\, p) \tag{2.2}$$

Vypočítáním f_q lze dopočítat veřejný klíč h, který se odvíjí od parametru g.

$$h \equiv f^{-1} (\mathrm{mod}\, p) \tag{2.3}$$

Klíče jsou ve tvaru polynomu. Veřejný klíč je polynom $K_{pub}(h)$ a soukromý klíč polynom $K_{priv}(f, f_p)$

Šifrování probíhá tak, že odesílatel rozdělí zprávu na m bloků polynomů a zvolí koeficienty mezi $\{-\frac{p}{2}, \frac{p}{2}\}$. Poté náhodně zvolí polynom ϕ a použije veřejný klíč pro zašifrování bloků polynomů.

$$e \equiv \phi \times h + m(\mathrm{mod}\, q) \tag{2.4}$$

Druhá strana přijme zašifrovanou zprávu e, kterou dešifruje pomocí svého privátního klíče $K_{priv}(f, f_p)$. Vypočítá se parametr a.

$$a \equiv f \times e(\mathrm{mod} q) \tag{2.5}$$

$$m \equiv f_p \times a(\mathrm{mod} p) \tag{2.6}$$

Dále je parametr a použit k dešifrování původní zprávy m.

Síla zabezpečení NTRU závisí na zvolených parametrech N, p a q, kdy při volbě parametrů $(N, p, q) = (503; 3; 256)$ a velikosti klíčů $(K_{pub}, K_{priv}) = (4024\mathrm{b}; 1595\mathrm{b})$ poskytuje nejvhodnější zabezpečení pro post-kvantovou kryptografii. [19]

2.2 NewHope

Protokol New Hope je relativně nový protokol vydaný roku 2016. Jeho úkolem je optimalizovat některé chyby a slabiny již představených protokolů. Vychází především z protokolu BCNS, který se snaží v několika ohledech optimalizovat. Jedná se například o lepší analýzu pravděpodobnosti selhání protokolu, kvůli které je používána mřížka $D4$, díky které je snížen modulus $q = 12289 < 2^{14}$. Díky tomu dochází ke zlepšení efektivity i bezpečnosti. Nepoužívá jinými protokoly často používaný pevně daný parametr, místo něj volí pseudonáhodný parametr volený při každé výměně klíčů. Při distribuci chyb využívá na rozdíl od Gaussova rozložení binomického rozložení, které je efektivnější. Zároveň však protokol nezahrnuje problematiku autentizace. Autoři protokolu uvažují, že je vhodnější zvolit pro výměnu klíčů kryptografii založenou na mřížkách a pro šifrování a digitální podpis využít kryptografii založenou na hashovacích funkcích. Předpokládá se, že dojde k využití ozkoušených post-kvantových podpisů, kdy by případné útoky neměly ohrozit předchozí komunikaci [20].
Proces ustanovení klíčů probíhá tak, že jsou domluveny parametery $q = 12289 < 2^{14}$a$n = 1024$, rozložení chybovosti je ψ_{16}. První strana si vygeneruje tzv. *seed* a na

základě něj pomocí hashovací funkce SHAKE-128 vygeneruje parametr **a**.

$$seed \leftarrow \{0.1\}^{256} \tag{2.7}$$

$$\mathbf{a} \leftarrow \text{Parse (SHAKE-128}(seed)) \tag{2.8}$$

Poté si obě strany na základě binomické distribuce vygenerují chybové stavy. Alice generuje **s,e** $\leftarrow \psi_{16}^n$, Bob $s_1, e_1, e_2 \leftarrow \psi_{16}^n$. Alice poté vypočítá **b**, a to pošle Bobovi společně se *seed*.

$$\mathbf{b{=}as{+}e} \tag{2.9}$$

Výměna probíhá přes NTT (Number-theoretic transform) doménu, která polynomy kódovány o 1792 bytech. Tyto polynomy jsou komprimovány do menšího formátu. *Seed* je kódován do 32 bytového pole a následně je provázaný se zákodovaným **b**. Bob obdrží *seed* a **b**, stejným způsobem, jako Alice, si vygeneruje parametr **a**. Následně Bob vypočítá hodnoty **u, v** pomocí získaných parametrů **a** a **b** a chybového stavu viz 2.11.

$$\mathbf{a} = \text{Parse (SHAKE-128}(seed)) \tag{2.10}$$

$$u = as_1 + e_1, v = bs_1 + e_2 \tag{2.11}$$

Dále Bob využije funkci založenou na hledání nejbližšího vektoru ve 4-dimenzionální mřížce, funkci HelpRec. Funkcí HelpRec je rozdělení 1024 koeficientů parametru **v** do 256 4-dimenzionálních vektorů, tedy $x_i = (v_i, v_{i+256}, v_{i+512}, v_{768})$ pro $i = 0, \ldots, 255$. Poté se vypočítá harmonizační informace r_i z x_i, kde b je náhodný bit a $g - (0{,}5; 0{,}5; 0{,}5)^t$.

$$r_i = \text{HelpRec}(x_i, b) = \text{CVP}_D \left(\frac{2^r}{q}(x_i + bg) \right) \bmod 2^r \tag{2.12}$$

$$r = \text{HelpRec(v)} \tag{2.13}$$

Po vývěru **r** zašle zpět Bob Alici dvojici parametrů **u** a **r**. Alice si na základě toho dopočítá vlastní parametr $\mathbf{v}_1$.

$$\mathbf{v}_1 = \mathbf{us} \tag{2.14}$$

Potom obě strany využijí funkci Rec, která také pracuje se 4-dimenzionálními vektory k obnovení chybových stavů. Bob využije parametry **r,v** a Alice **r**,$\mathbf{v}_1$.

$$v = \text{Rec}(\mathbf{v}_1, \mathbf{r}), v = \text{Rec}(\mathbf{v}, \mathbf{r}) \tag{2.15}$$

Poslední krok k získání finálního klíče je zahashování nového parametru μ.

$$\mu = \text{SHA3-256}(v) \tag{2.16}$$

Výhodou protokolu New Hope je jeho nízká výpočetní a paměťová náročnost, jeho rychlost a jednoduchost jeho implementace. Zároveň zajišťuje vysokou úroveň bezpečnosti. Mezi jeho nevýhody patří malá výše rozptylu chyb a to, že je limitován v případě volby parametrů.

2.3 Kyber

Protokol Kyber byl představen v roce 2016. Je založen na teorii mřížek a je představen později, než protokoly NTRU a New Hope, na jejichž základě se inspiruje a jeho snahou je inovace těchto protokolů. Protokol je rozdělen do několika částí. Kyber zahrnuje schéma Kyber.KE pro výměnu klíčů, i Kyber.AKE pro autentizovanou výměnu klíčů. Dále Kyber.CPA zajišťuje vytvoření schéma šifrování pomocí veřejného klíče. Kyber.Hybrid se potom vyznačuje využitím Fujisaki-Okamoto transformace[21].

Protokol Kyber je také postaven na Ring-LWE problému, kdy všechny polynomy jsou v okruhu $R_q = \mathbb{Z}_q(X)/(X_n + 1)$. Parametr $q = 7681$ je volen, protože se jedná o nejmenší prvočíslo, kde $q \equiv 1 \text{mod}(2n)$, lze potom využít k šifrování NTT jednoduše a efektivně. Kyber, stejně jako New Hope používá binomické rozložení, ale ψ_3. Více o protokolu Kyber a jeho částech zde[21].

Výhodou protokolu Kyber je jeho výpočetní nenáročnost, paměťová nenáročnost a je relativně jednoduše implementovatelný. Může být kritizován za to, že je postaven na Ring-LWE problému, díky kterému však dosahuje časově nenáročných výsledků.

2.4 Frodo

Protokol Frodo byl představen v roce 2016. Opět se jedná o protokol založený na kryptografii mřížek, ale oproti předchozím kandidátům se liší v postavení na jiném matematickém problému, avšak čistého LWE problému. Frodo tedy nevyužívá polynomiálních okruhů. Je tedy časově náročnější, než předchozí protokoly. Pravdou je, že zatím nebylo prokázáno, že by LWE bylo, při použití stejných parametrů, méně bezpečné, než Ring-LWE.[22, s. 1–5].

Doporučené parametry pro tento protokol jsou $n = 752$ a $q = 2^{15}$ a pro chybovou distribuci distribuce χ, založená na Gaussově rozložení na rozdíl od předchozích protokolů, využívajících rozložení binomické.

Na začátku je Alicí generován $seed \leftarrow 0,1^{256}$. Následně si nechá vygenerovat matici $\mathbf{A}$, kterou získá pseudonáhodnou funkcí Gen.

$$\mathbf{A} = \text{Gen}(seed) \tag{2.17}$$

Pomocí χ distribuce si dále vygeneruje $\mathbf{S}$, $\mathbf{E} \in \chi(\mathbb{Z}_q^{n\times m})$. Následně dopočítá veřejný klíč v podobě matice $\mathbf{B}$, kdy $\mathbf{B} \leftarrow \mathbf{AS} + \mathbf{E}$. Stejně, jako v případě New Hope, Bobovi zasílá parametr $\mathbf{B}$ a *seed*.

Bob využije získaný parametr *seed* a znovu pomocí pseudonáhodné funkce Gen si vygeneruje parametr $\mathbf{A} = \text{Gen}\ (seed)$. Dopočítá $\mathbf{S}_1$, $\mathbf{E}_1 \in \chi(\mathbb{Z}_q^{n\times m})$. Dále i své $\mathbf{B}_1$, $\mathbf{B}_1 \leftarrow \mathbf{AS}+\mathbf{E}$. Potom Bob pro ustanovení klíče vygeneruje ještě jeden stav založený na χ distribuci, matici $\mathbf{E}_2$.

$$E_2 \leftarrow \chi(\mathbb{Z}_q^{n\times m}) \tag{2.18}$$

Zároveň pomocí $\mathbf{E}$ dopočítá matici $\mathbf{V}$, $\mathbf{V} \leftarrow \mathbf{AS}_1+\mathbf{E}_2$. Strany poté využijí funkce, které dokáží zharmonizovat vygenerování klíče K. Jedná se o funkce crossrounded function a rounded function viz [22]
Využitím crossrounded funkci Bob vygeneruje marici $\mathbf{C}$, kterou i s $\mathbf{B}_1$ pošle Alici.

$$C \leftarrow \langle \mathbf{V} \rangle_{2^B} \tag{2.19}$$

Alice získává několik parametrů a může použít *rec* funkci, díky níž může vygenerovat společný ustanovený klíč K. Aby Bob vygeneroval klíč K, používá rounded function a dochází ke shodě a finálnímu ustanovení klíčů.

Protokol Frodo využívá narozdíl od ostatních protokolů založených na mřížkách klasický problém LWE, který využívá mnohem více paměťového prostoru a jednotlivé výpočty v jeho případě zaberou více času. Může být ovšem považován za bezpečnější, než systémy založeny na problému Ring-LWE. Jeho nevýhodou je potřeba velké délky klíčů, které vyžadují paměťovou náročnost o mnoho větší, než doposud zmíněná schémata.

2.5 SIKE

Protokol SIKE (Supersingular isogeny Key Encapsulation), představený roku 2017, je založen na supersingulárních křivkách. Protokol SIKE je založen na protokolu

SIDH, využívá však Hofheinzovy transformace pro dosáhnutí CCA bezpečnosti. Protokol využívá takzvané izogenity pro soukromé klíče a eliptické křivky pro klíče veřejné, jak již bylo zmíněno. Jedná se o bezpečný protokol proti generickým útokům, není totiž známý generický útok, který by tento protokol prolomil.[23]

2.6 McEliece

McEliece je první asymetrický šifrovací kryptosystém, vydaný Robertem McEliece v roce 1978 , založený na samoopravných kódech a je považován za hlavního představitele těchto šifer. Zvláštnost této metody mimo jiné spočívá v používání úmyslného zanesení chyby do zakódované zprávy. Přenášená informace ve zprávě je tedy jinými slovy zničena, avšak vlastník soukromého klíče je schopný tyto chyby správným dekódováním odstranit a obnovit tak původní zašifrovanou zprávu.[24, s. 114–116]. Tento kryptosystém v dnešní době není velmi využíván díky několikanásobně větší velikosti klíčů oproti dnešním metodám, avšak jeho popularita roste díky kandidatuře pro post-kvantovou asymetrickou kryptografii.

Protokol tedy pro vlastní funkci potřebuje, jako jiné protokoly, vygenerovat klíče, které počítá tak, že si nejdříve zvolí lineární kód $\mathcal{K}$ s parametry (n,k,t) (opravující t chyb) a $k \times n$ generující maticí G, pro který je znám efektivní dekódovací algoritmus. Dále se vygeneruje náhodná $k \times k$ regulární matice S a náhodná $n \times n$ permutační matice P. Vypočítáním $k \times n$ se získá matice $\hat{G}$=SGP. Čísla k, n a t jsou veřejnými parametry systému, matice $\hat{G}$ představuje veřejný klíč a kód generujicí maticí G včetně matic S a P jsou klíči soukromými.[24]

Šifrování protokolu McEliece probíhá tak, že se jako první vygeneruje náhodný vektor délky n s hammingovou váhou t^2. Šifrovaná zpráva c délky n je vytvořena zakódováním generující maticí $\hat{G}$ a přičtením chybového vektoru z.

$$c = m\hat{\mathrm{G}} + z \tag{2.20}$$

Získání zprávy m je provedeno opačnou operací, tedy dešifrováním. Obdrženou zašifrovanou zprávu c délky n je dešifrována tak, že se nejprve vypočítá vektor $\hat{c}$ délky n, $\hat{c} = cP^{-1}$.

$$m = Dek_{\mathcal{K}}\hat{\mathrm{c}} \tag{2.21}$$

3 Srovnání a analýza vhodnosti vybraných post-kvantových schémat na čipové karty

Předchozí kapitola byla věnována analýze jednotlivých post-kvantových protokolů. Velký důraz z důvodu efektivity implementace i samotné funkcionality a úrovně bezpečnosti byl kladen především na protokoly, které jsou založeny na mřížkách. Mimo jiné byla věnována pozornost i protokolu založeném na supersingulárních křivkách a podpisovým protokolům využívající hashovacích funkcí. Tato kapitola zahrnuje současný stav post-kvantové kryptografie na omezených zařízeních, popis omezeného zařízení, jako je čipová karta, a analýzu post-kvantových metod s ohledem na paměťové, výpočetní a implementační nároky.

3.1 Současný stav post-kvantové kryptografie na omezených zařízeních

S nástupem post-kvantových algoritmů, které jsou zejména paměťově velmi náročné, přichází problém implementace na zařízení, která disponují malou výpočetní i paměťovou sílou. Pro dnešní stolní počítač není počítání post-kvantových metod velký problém, ale existují i méně výpočetně silná zařízení, u kterých je post-kvantová kryptografie velmi důležitá, protože by s nástupem kvantových počítačů nejspíše úplně zanikla, v každém případě nebyla bezpečná. Proto se vývoj post-kvantových metod ubírá směrem k redukování komplexity výpočtů i paměťových nároků, avšak zároveň i zvyšování jejich bezpečnosti.

Při realizaci implementace je velmi důležité, jaká metoda je použita. Ve většině případů platí pravidlo, že je metoda velmi paměťově náročná, ale výpočetně velmi snadná a rychlá, nebo naopak paměťová nenáročnost je zaplacena potřebou vysokých výpočetních zdrojů.

V již realizovaných implementacích na omezená zařízení se zdají velmi populární schémata založena na mřížkách. Je to z důvodu relativně jednoduché implementace, dobrých výsledků, kterých tyto metody dosahují jak po stránce bezpečnosti, tak po stránce šetření výpočetními zdroji. Navíc se metody založeny na mřížkách zdají dobře modifikovatelnými různými funkcemi redukující paměťové či výpočetní nároky. Zejména práce zabývající se úspěšnou implementací metody založené na mřížkách, blíže na problému Ring-LWE na čipovou kartu JavaCard, která disponuje pouze 10KB pamětí RAM viz. [25] zcela dokazuje, že lze i na takto omezená zaří-

zení nahrát verifikační protokol. Ve zmíněné práci je vysvětleno, že na velmi omezená zařízení, jako je JavaCard, se kvůli výpočetnímu omezení může počítat pouze s metodami založenými na problému Ring-LWE díky redukci klíčů, tedy nezanechání klíčů v původní matici, která je pro omezená zařízení velmi náročná na uložení. Zároveň je práce zaměřena na algoritmy, které zajišťují větší efektivitu schématu, jako je například *Montgomeryho alrogitmus (MMM)*, zajišťující rychlý způsob optimalizace násobení díky redukci velmi velkého čísla, který je pro slabý procesor velmi přínosný algoritmus k ulehčení a zrychlení výpočtů. Dále je zde popsána metoda *Rychlá Furierova transformace (FFT)*, která v kombinaci s Montgomeryho algoritmem napomáhá rychlému násobení velkých čísel, ke kterému v případě metod založených na mřížkách dochází. I další práce, jako [26] a [27], které se zabývají implementací, či efektivitě mřížkově-založených kryptografických systémech dokazují, že ve spojení s matematickými funkcemi jako je Furierova transformace, Montgomeryho algoritmus, či diskrétní Gaussovo vzorkování mohou být tyto metody velmi efektivní.

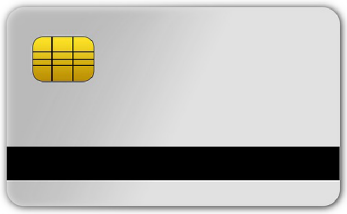
Mezi již realizovanými pokusy či úspěšnými implementacemi lze také zmínit implementaci protokolu McEliece PKC na čipovou kartu viz [28]. Jedná se o implementaci schématu založeného na Goppa kódech. Tato implementace byla použita na zařízení s 16-bitovým procesorem, pamětí 12KB RAM a 504KB NVM pamětí, což je velmi omezené zařízení, ale v porovnání s čipovými kartami se jedná pořád o docela vysoko-kapacitní i výpočetně zdatné zařízení. Hlavní výhodou je relativně velká paměť, která umožňuje ukládání celých matic. I když se jedná o ukládání do paměti NVM, tedy zapisování je asi 20x pomalejší, než zapisování do paměti RAM, jedná se o úspěšnou implementaci protokolu McEliece PKC na omezné zařízení.

Odborné práce zabývající se metodami na omezených zařízeních se tedy ubírají hlavně směrem k vývoji protokolů založených na mřížkách, zejména těch, které jsou založeny na problému Ring-LWE. Metody využívající kódování (jako McEliece) se zdají velmi efektivními co se týče výpočtu, jsou však velmi náročné na ukládání celých matic. Naopak velmi paměťově úsporné protokoly založené na supersingulárních křivkách s výhodou přenášení malé velikosti zpráv se zdají výpočetně nevýhodné, proto o jejich implementaci na omezených zařízeních nelze mnoho nalézt.

3.2 Čipové karty

Čipová karta, viz obrázek 3.1, je hojně využívané médium, které zabezpečuje autentizaci osob, které žádají o přístup k systému, účtu nebo například pro vstup

do budovy, nebo pro jiné účely, kde je třeba rozlišit, zda-li má osoba, která se o přístup hlásí, patřičná oprávnění. Je to omezené zařízení, které obsahuje integrovaný



Obr. 3.1: Čipová karta

obvod. Omezení čipové karty spočívá v její velmi malé paměti a omezených výpočetních zdrojů. Omezené výpočetní zdroje jsou dány mikroprocesorem vykonávajícím aritmetické operace o nízké frekvenci. Možnost spuštění složitých kryptografických algoritmů je tedy jeden z problémů, kterým čipové karty čelí. I když jsou dnešní čipové karty již uzpůsobeny pro podporu kryptosystémů, jako je např. RSA, AES, nebo kryptosystémů založených na eliptických křivkách, s nástupem kvantových počítačů jsou tato zařízení do budoucna bezbranná. Problémem je, že post-kvantové zabezpečení s sebou přináší metody, které využívají, v porovnání s dnešními schématy, mnohonásobně větší klíče. Tyto parametry jsou popsány dále v kapitole 3.3.1

3.2.1 Vybavení čipových karet

Čipové karty jsou charakterizovány většinou 8 až 16-bitovým mikročipem o nízké frekvenci (okolo 30-60MHz), pamětí RAM, pamětí ROM a programovatelnou pamětí EEPROM. Paměť RAM slouží jako operační paměť programu pro ukládání mezivýsledků během chodu aplikace na čipové kartě, zatímco paměť EEPROM (elektricky mazatelná nevolatilní paměť typu ROM-RAM) slouží jako „pevný disk“ karty, tedy úložiště perzistentních dat. Důležité je také poznamenat, že zapisování do paměti EEPROM je omezené počtem zapsání (okolo 200,000) a zároveň je pomalejší a energeticky náročnější, než zapisování do paměti RAM.

3.3 Vhodná schémata k implementaci na čipovou kartu

K tomu, aby byla implementace post-kvantového kryptografického systému možná na tak výpočetně i paměťově omezené zařízení, jako je čipová karta, je nutno zvolit

protokoly s nejnižší možnou úrovní zabezpečení - tedy NIST 1 (128 bit) zabezpečením (viz tabulka 1.3). Volba těchto protokolů je dána velmi omezenými zdroji zařízení, ale stále se jedná o protokoly odolné vůči kvantovým útokům.

Tab. 3.1: Teoretické srovnání vybraných protokolů

Protokol	**Typ kryptografie**	**Využití**	**Matematický problém**
Picnic	křivky	Digitální podpis	Supersingulární křivky
Rainbow	křivky	Digitální podpis	Polynomiální křivky
SIKE	křivky	Ustanovení klíčů	Supersingulární křivky
NewHope	mřížky	Ustanovení klíčů	Ring-LWE
NTRU	mřížky	Ustanovení klíčů, Digitální podpis, Asymetrické šifrování	Ring-LWE
McEliece	kódy	Ustanovení klíčů	Goppa kódy

K podrobnějšímu představení protokolů, které podle předpokladů již vypracovaných prací, uvedených v kapitole 3.1, a vlastními specifikacemi, byly z dostupných představitelů vybrány NewHope (NewHope-512-CCA), NTRU (ntru-kem-443), SIKE (sike503), McEliece (mceliece6960119), Rainbow (Rainbow Ia) a Picnic (picnic1fs), které jsou dostupné v oficiálním adresáři NIST dostupné na [17]. Některé z metod byly již představeny v kapitole 2 a některé byly zvoleny na základě již zmíněných prací, nebo z důvodu srovnání problémů, na kterých jsou postaveny. Jsou vybrány optimalizované verze těchto metod, protože některé již obsahují modifikace, které optimalizují výkon dané metody (většina modifikací se věnuje právě paměťovým a výpočetním nárokům, které jsou pro tuto práci vítány).

3.3.1 Srovnání vybraných schémat z pohledu paměťové a výpočetní náročnosti

V rámci této podkapitoly jsou zmíněné protokoly, tedy zástupci většiny metod, navzájem porovnány z pohledu využívaného místa - paměti RAM, EEPROM a náročnosti početních operací - procesorový čas strávený na počítání metody. Měření proběhlo na osobním počítači s virtualizovaným operačním systémem Ubuntu 18.04 LTS a čtyřjádrovým procesorem o taktu 2,5 Ghz s využitím balíčků Libntl-dev, OpenSSL, Libssl-dev, dev, xsltproc, a libgmp3-dev. Všechny měřené protokoly jsou měřeny s volenými parametry tak, aby splňovaly stejnou úroveň zabezpečení a bylo možné změřit, které metody jsou nejefektivnější. Volená úroveň zabezpečení je, jak již bylo zmíněno, úroveň 1 podle NIST-bezpečnosti, což odpovídá 128b-bezpečnosti, tedy

považovanou za kvantově odolnou. Platí, že čím větší úroveň zabezpečení, tím větší je potřeba velikost klíčů, nebo náročnost výpočtů, což je v našem případě nežádoucí. Velikosti klíčů uvedené v tabulce 3.2 a 3.3 jsou pevně dány parametry a vnitřní logikou metod, jsou to tedy údaje, které jsou neměnné.

Tab. 3.2: Velikosti klíčů metod pro ustanovení klíčů v bytech

Protokol	**SK**	**PK**	**CT**	**Součet**
SIKE	434	379	402	1214
NTRU	701	611	611	1923
NewHope	1888	928	1120	3936
McEliece	13908	1047319	226	1061453

Tab. 3.3: Velikosti klíčů podpisových schémat v bytech

Protokol	**SK**	**PK**	**Součet**
Picnic	49	33	82
Rainbow	100209	152097	252306

Hodnoty „EEPROM“ v tabulkách 3.4 a 3.5 jsou měřeny operačním systémem a jedná se o velikost, kterou představuje program jako takový, s perzistentními daty, které využívá. Měření využití paměti EEPROM je tedy v zásadě orientační, protože se jedná o programy napsané v programovacím jazyce C a jejich překlad do binárního souboru je proveden překladačem, který není k dispozici u implementace na čipové karty, navíc čipové karty mnohdy překládají kódy do jiných kódů, které jsou přizpůsobeny pro chod karty. Lze tedy s jistotou říci, že program napsaný v jazyce C bude mít odlišnou velikost, než velikost programu napsaném v jiném programovacím jazyce, bude-li tedy jiným jazykem implementována. Metody navíc obsahují redundantní funkce, které jsou pro náš účel nepotřebné. Redundance funkcí je ovšem u všech změřených post-kvantových metod velmi podobná, dá se tedy s jistotou říci, že výsledné hodnoty určují, která z metod je nejvhodnější z pohledu paměťové náročnosti pro perzistentní data (tedy náročnosti na paměť EEPROM v rámci čipové karty).

Nejméně místa zabírá protokol NewHope, který i s redundantními funkcemi potřebuje pouze 40,7 kilobytů a je z tohoto hlediska nejvhodnějším kandidátem. Co se týče procesorového času využitého na procesy metod, tak si nejlépe vede protokol Picnic.

Tab. 3.4: Paměťová a výpočetní náročnost metod pro ustanovení klíčů

Protokol	**EEPROM v kB**	**Procesorový čas v sekundách**
NewHope	40,7	0,20
SIKE	124,1	29,81
McEliece	185,3	22,28
NTRU	562,6	0,15

Tab. 3.5: Paměťová a výpočetní náročnost podpisových schémat

Protokol	**EEPROM v kB**	**Procesorový čas v sekundách**
Picnic	692,7	0,12
Rainbow	162,9	62,4

Je ale velmi náročný na paměť, protože obsahuje složitou a komplikovanou implementaci (to se odráží na výsledné velikosti programu). Druhým protokolem, který je nejméně náročným na paměť EEPROM, je protokol SIKE, ten ovšem spotřebuje nejvíce procesorového času, což znamená, že je k jeho vykonání potřeba nejvíce procesorových cyklů značících komplexní matematické operace. V rámci procesorového času jsou dále velmi výhodné protokoly NTRU a NewHope, které díky jednoduché implementaci využívají jednodušší operace pro svůj výpočet.

Dalším důležitým parametrem je využití paměti RAM, tedy využívání operační paměti programem. Měření využívání paměti RAM je ovlivněno, stejně jako u měření paměti EEPROM, stejnými faktory, tedy redundancí funkcí, které jsou pro náš účel nepotřebné, ale je důležité zjistit, která z metod je nenáročná na paměť RAM, protože zejména paměť RAM je v zařízeních, jako je čipová karta, velmi malá. Naměřené hodnoty jsou měřeny nástrojem Valgrind (verze 3.13.0) s jeho dostupnými podnástroji (*memcheck, massif a callgrind*), který byl pro toto měření zvolen díky jeho velmi podrobnému výstupu o zkoumaných programech.

Nástroj Valgrind a jeho výše zmíněné nástroje měří využívání operační paměti programů, zaznamenávají průběh programových alokací paměťových prostředků, jejich uvolňování, chyby v paměti způsobené programem, velikost využité paměti, rozdělení paměti na zásobník a haldu a celkové zhodnocení náročnosti programu na paměti RAM. Velmi potřebnou výhodou tohoto nástroje je, že se spouští z příkazového řádku a zkoumaný program je spuštěn jako parametr nástroje Valgrind

se svými případnými parametry. To zabezpečuje, že nástroj Valgrind zkoumá pouze paměťové prostředky využívané programem, nad kterým je spuštěn. Nástroj jako takový má tedy určitou režii na paměti RAM, ale jako výsledek měření uvádí pouze o hodnoty vyprodukované zkoumanými programy.

Tab. 3.6: Zhodnocení post-kvantových metod z pohledu využití paměti RAM

Protokol	**Počet volání**	**Celkově alokováno v bytech**	**Maximální využití zásobníku v kilobytech**	**Maximální využití haldy v kilobytech**
Picnic	162 842 487	6 383 846	3,0	3 627
Rainbow	492 708 717 650	301 655 112	0	274 250
SIKE	145 237 721 225	5 672	3,0	16,4
NewHope	558 553 891	837 064	12,9	22,2
NTRU	545 427 051	2 935 224	4,3	23
McEliece	41 512	3 125	7,2	8,8

Dle hodnot zaznamenaných v tabulce 3.6 lze říci, že nejvíce náročná metoda pro zpracování z pohledu paměti RAM je protokol Rainbow, který má nejvíce programových volání, alokuje celkově nejvíce bytů v paměti a má nejvyšší maximální využití haldy (paměti pro instance, objekty atp.). Protokoly SIKE a Picnic využívají mnoho volání, avšak malých velikostí zásobníku a haldy, tedy ukládají menší mezivýsledky, ale pracují s nimi častěji, než v případě protokolů založených na mřížkách, tedy NewHope a NTRU. Tyto protokoly trpí využíváním většího prostoru, avšak oproti ostatním zmíněným jen nepatrně. Nejlepší výsledky v náročnosti paměti RAM má jednoznačně protokol McEliece založený na kódování. Využívá velmi malého místa v paměti RAM a nejnižšího počtu volání ze všech uvedených protokolů.

Naměřené hodnoty v tabulce 3.6 jsou orientační z důvodu přidělování systémových prostředků prováděné operačním systémem, který je jiný, než je tomu u čipové karty. Dále také kvůli přidělování prostředků, které má počítač několikanásobně větší, než samotná čipová karta.

3.3.2 Shrnutí a určení metody k implementaci

Předešlé kapitoly se zabývaly měřením post-kvantových protokolů a jejich srovnání na základě jejich paměťové náročnosti (využití pamětí RAM a EEPROM, velikosti klíčů) a jejich výpočetní náročnosti (délce procesorového času). Všechny tyto aspekty jsou při volbě vhodného schéma velmi důležité, avšak jsou důležité například i datové typy, které tyto metody využívají a jejich možná implementace na čipové karty.

V rámci srovnání je nejméně výpočetně i paměťově náročný protokol McEliece, který však generuje klíče o takové velikosti, že jejich samotné uchování, nebo práce s nimi je pro dnešní čipové karty nemožné. Podpisový protokol Rainbow má také velmi velké klíče, navíc jejich generování a samotný protokol je velmi výpočetně náročný díky polynomiálním křivkám, které využívá.

Podpisový protokol je dobrou volbou pro implementaci na zařízení, jako je čipová karta, protože použití podpisu v autentizaci je již hojně využíváno, ovšem druhý podpisový protokol Picnic je i přes velmi malé klíče velmi paměťově náročný jak pro paměť RAM, tak pro paměť EEPROM. Protokol SIKE disponuje malou velikostí klíčů i celkovým využitím programové paměti, avšak je velmi náročný na výpočet díky supersingulárním křivkám, které jsou i z implementačního hlediska složité. Protokoly založené na mřížkách - NewHope a NTRU jsou protokoly, které generují přijatelně velké klíče, ve velmi dobrém čase (jsou tedy i výpočetně nenáročné) a jsou i paměťově srovnatelné s ostatními protokoly.

Měření tedy potvrdilo i dosavadní vývoj implementací na omezená zařízení. Metody založené na kódování mají nepřijatelnou velikost klíčů, metody založené na supersingulárních křivkách jsou velmi implementačně a výpočetně náročné, a metody založené na hashovacích funkcích naopak paměťově náročné. Metody založené na mřížkách jsou kompromisem mezi velikostí klíčů, avšak z hlediska implementace jsou relativně jednoduché a paměťové nároky jsou srovnatelné s ostatními protokoly. Navíc metoda nevyužívá složitých datových typů a struktur a využívá matematických primitiv, která lze stále optimalizovat a tím zefektivnit dané metody.

Z metod založených na mřížkách jsou mnohem efektivnější ty, které využívají problému Ring-LWE. Oproti LWE totiž nevyužívají celé matice, ale vytvářejí polynomiální kruh, kterým jsou velikosti klíčů zredukovány. Klíče uložené do matic by bylo možné linearizovat, avšak za cenu nárůstu požadavků na paměť. Proto se metody založené na mřížkách, blíže na problému Ring-LWE zdají být nejvhodnějšími pro implementaci na čipovou kartu.

Z měřených schémat založených na mřížkách sice vyšel lépe protokol NTRU z pohledu velikosti klíčů, ale jedná se o stovky bytů, které nejsou tak důležité, jako potřeba paměti EEPROM a RAM, kterou má výhodnější protokol NewHope. Procesorový čas obou metod je potom velmi podobný, což vyplývá z velmi podobné složitosti a implementační podobě obou metod. Metoda, která bude implementována na čipovou kartu je tedy NewHope, která se z měření a z pohledu možné implementace zdá nejvhodnější volbou.

S volbou implementované metody je také důležité zvolit zařízení, na které bude tato metoda implementována, a to zejména z pohledu datových typů, matematických operací, aplikačního prostředí a programovacího jazyka nutného pro danou implementaci.

4 Implementace metody NewHope na čipovou kartu

Tato kapitola se věnuje implementaci protokolu NewHope na čipovou kartu. Důraz je kladen především na stručný popis protokolu, určení vhodné karty pro samotnou implementaci a vlastní návrh verifikační metody, která je složena z částí NewHope. S návrhem verifikace je spojena modifikace samotné metody, aby byla paměťově i výpočetně přívětivější, která je v této kapitole také podrobně popsána. Po návrhu verifikační metody pomocí protokolu NewHope jsou pak popsány funkce, které jsou jím využívány. Výstupem kapitoly je pak popis implementace ve vybraném jazyce a problémy spojené se samotným protokolem i jeho implementací.

4.1 Výběr čipové karty

S výběrem implementované metody je úzce spojen výběr čipové karty, na kterou je schéma implementováno. Lze vybírat mezi několika výrobci, kteří vyrábí čipové karty s podobnými účely, avšak s naprosto odlišnými parametry.

Z dnes dostupných karet se jeví jako nejvhodnější uvažovat implementaci na jednu ze tří nejvíce propagovaných čipových karet, mezi které se řadí karty Multos, JavaCard nebo BasicCard.

Při výběru karty je velmi důležité, jaké má karta parametry, s jakými datovými typy dokáže pracovat, což také úzce souvisí s programovacím jazykem, ve kterém jsou programovatelné. Tabulka 4.1 srovnává karty z pohledu programovacích jazyků. Každý z programovacích jazyků má některé výhody i nevýhody. Například jazyk

Tab. 4.1: Jazyky programovatelných čipových karet

Čipová karta	**Programovací jazyk**
BasicCard	Basic
Multos	C
JavaCard	Java

Java se zdá pro náš příklad implementace nevhodný kvůli nedostatku podpory aritmetických operací s velkými čísly. Jazyk C je výhodný z pohledu využitelnosti kvůli jeho rozsáhlosti a přenositelnosti, avšak karty Multos nejsou osazeny příliš velkou pamětí RAM, výrobce sází spíše na velikost EEPROM paměti, která je v našem případě potřebná, avšak karty BasicCard obsahují velmi podobnou paměť a nabízí

více paměti RAM.

BasicCard karty jsou navrhovány přímo pro kryptografické protokoly a disponují množstvím knihoven, které jsou pro implementaci kryptografických metod velmi podpůrné. Jazyk Basic má nevýhodu méně obsáhlé a popsané dokumentace, avšak poskytuje knihovny pro práci s velkými čísly, které jsou v případě protokolu NewHope velmi žádané[29].

4.1.1 BasicCard ZC 7.6 rev D

Pro implementaci post-kvantové verifikační metody byla zvolena čipová karta BasicCard ZC 7.6 rev D z důvodu podpory aritmetických operací, hash funkcí a zároveň přijatelným paměťovým možnostem karty. Čipová karta BasicCard ZC 7.6 rev D se řadí mezi ostatními kartami Basic mezi „profesionální“ karty výrobce. Ostatní karty výrobce se odlišují zejména velikostí pamětí RAM a EEPROM, využití protokolů pro komunikaci s terminálem, možností hashovacích funkcí a možností šifrování, která je karta možná aplikovat. Čipová karta (na obr. 4.1) je programovatelná v

Tab. 4.2: Specifikace karty Basic ZC 7.6 rev D

RAM	**EEPROM**	**ROM**
4,2kB	73,7kB	256kB

jazyce Basic, který je vhodnou volbou pro námi zvolenou metodu, protože podporuje potřebné datové typy a hlavně násobení velkých čísel, které se často v metodě NewHope objevuje v podobě násobení polynomů. Paměti RAM, EEPROM a jejich využití programem je zmíněno již v kapitole 4.1 a paměť ROM je v našem případě využívána pouze operačním systémem karty. Paměť ROM tedy nelze programáto-

Obr. 4.1: Čipová karta BasicCard ZC 7.6 rev D

rem využít k uložení mezivýsledků nebo potřebných hodnot.

Karta je připojena k počítači čtečkou karet (na obr.4.2), která zprostředkovává komunikaci mezi kartou a počítačem, který je v tomto případě terminál. Zajišťuje hlavně zápis a čtení dat z karty a operace, které jsou na kartě volány.

Obr. 4.2: Čtečka čipových karet OMNIKEY

Podrobnější informace o BasicCard kartách, jejich komunikaci s terminálem a podrobnější specifikace jsou vypracovány v práci [29]

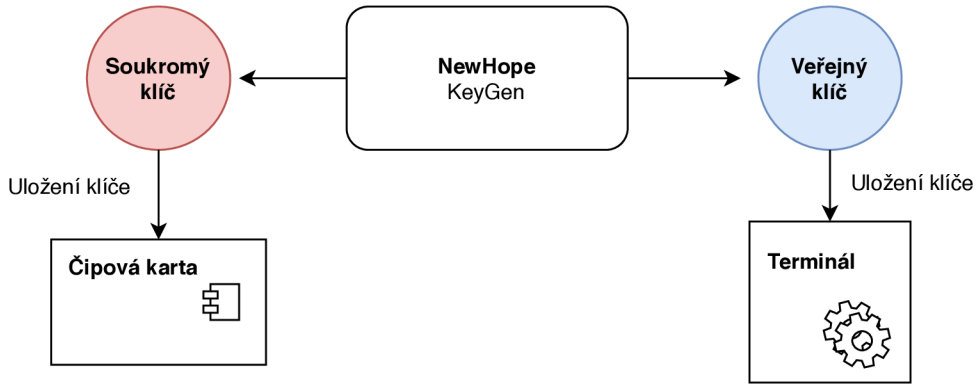
4.2 Modifikace protokolu NewHope a návrh verifikace

Jak bylo zmíněno v kapitole 3.3, paměťové i výpočetní nároky metod jsou ovlivněny redundantními funkcemi v již implementovaných metodách. Verifikační metodu NewHope je tedy nejprve nutné upravit na co nejvíce paměťově i výpočetně nenáročnou tak, aby tuto metodu bylo možné na čipovou kartu vůbec nahrát. Při úpravě verifikační metody je důležité klást důraz na to, které funkce jsou opravdu redundantní, aby odmazáním potřebných funkcí nedošlo k narušení bezpečnosti. Zároveň je důležité implementovat pouze funkce, které jsou pro verifikační metodu důležité. Pro určení těchto funkcí je třeba vysvětlit, jak by měla verifikační metoda na čipové kartě fungovat.

4.2.1 Verifikace karta–terminál

Nejdříve je třeba osvětlit práci verifikační metody na rozhraní karta-terminál, které bude v naší práci využito. Podpisová schémata fungují tak, že jsou pomocí určitého algoritmu vypočítány klíče a soukromý klíč je uložen na kartu a veřejný na terminál. Terminál pošle zprávu m, kterou čipová karta podepíše. Majitel karty se může autentizovat tím, že přiloží kartu k terminálu a terminál ověří, zdali je podpis správný,

nebo ne. V našem případě, tedy v případě šifrovacích metod, jako je NewHope, se také vypočítají klíče (veřejný a soukromý), soukromý klíč je uložen na čipové kartě a veřejný klíč je uložen v terminálu. Počítání klíčů by mělo být z hlediska bezpečnosti vypočítáno na zařízení třetí strany, která by pak představovala autoritu. Při autentizaci majitele karty se potom zpráva m zašifruje, přičemž vznikne šifrovaná zpráva c. Terminál si pamatuje původní zprávu m. Karta přijme od terminálu šifrovanou zprávu c a rozšifruje ji soukromým klíčem. Rozšifrovanou zprávu m karta pošle zpět terminálu, který pouze porovná přijatou zprávu s tou, kterou nejprve šifroval. Tento proces je znázorněn na obrázku 4.4. Pokud jsou zprávy stejné, je díky bezpečnosti protokolu prokazatelné, že majitel karty je např. opravdovým vlastníkem bankovního účtu, ke kterému přistupuje. V našem případě se jedná o metodu



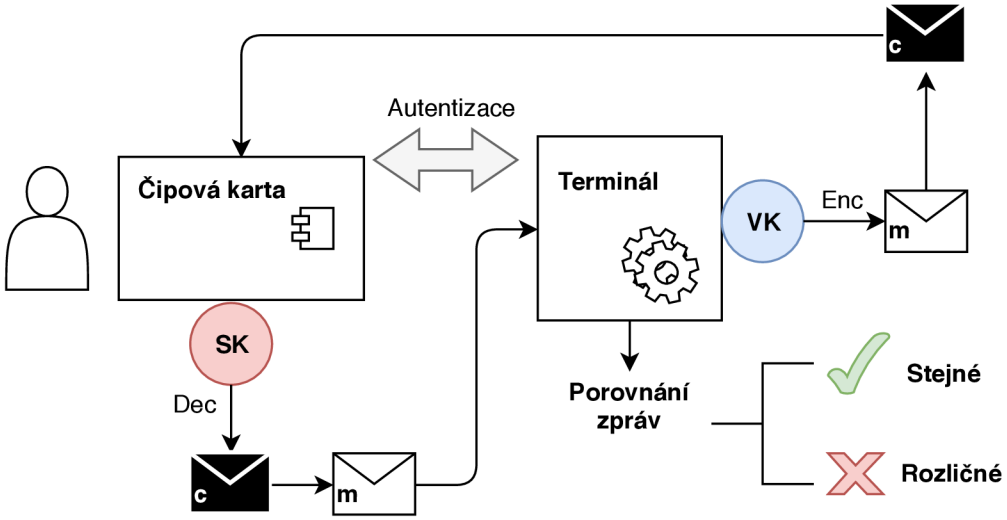
Obr. 4.3: Generování klíčů

NewHope, ale nutné je, aby oba klíče byly vygenerovány stejnou metodou. *KeyGen* značí metodu generování veřejného a soukromého klíče.

V případě obrázku 4.4 je soukromý klíč značen „SK“ a veřejný klíč „VK“. *Enc* je potom označení pro metodu zašifrování zprávy a *Dec* značí metodu opačnou, rozšifrování zprávy. Porovnání zpráv potom jednoduše porovná přijatou a uloženou zprávu. Ta se nemůže rovnat, pokud na kartě není soukromý klíč, který byl vygenerován zároveň s klíčem veřejným.

4.2.2 Redukce verifikační metody a návrh k implementaci

K verifikační metodě nejsou potřeba všechny moduly post-kvantového protokolu NewHope. Některé moduly jsou zde nepotřebné pro verifikaci, a jak z měřených hodnot v kapitole 3.3 a specifikace karty (tabulka 4.2) vyplývá, není možné všechny moduly protokolu NewHope přímo implementovat na námi zvolenou čipovou kartu.



Obr. 4.4: Návrh verifikační metody karta-terminál

Jedním z těchto modulů je například enkapsulace (zapouzdření) klíče a opačný proces, tedy jeho dekapsulace (rozbalení). Dalším faktorem redundantních dat je, že některé funkce potřebné protokolem NewHope jsou již implementovány v knihovnách, které samotná čipová karta obsahuje. Jedná se zejména o generování náhodného čísla, nebo zarovnání velkého čísla do bloků pomocí hash funkce. Tyto funkce lze nahradit až v implementaci na samotnou čipovou kartu, ale pro preventivní přeměření paměťové náročnosti již redukovaného protokolu jej lze tedy rozdělit na námi potřebné funkce, jako je generování klíčů (*KeyGen*), zašifrování zprávy veřejným klíčem (*Enc*) a rozšifrování šifry soukromým klíčem (*Dec*).

V rámci měření na PC jsou pro tyto metody stále využity knihovny jazyka C, zejména pro funkce hashování a generátoru náhodného čísla, které budou využívat jiných výpočetních i paměťových zdrojů. Pro účely měření kapacitních požadavků je však pravděpodobné, že metody implementované na čipové kartě budou paměťově velmi podobné či méně náročné z důvodu implementace na paměťově velmi omezené zařízení. Verifikační protokol je tedy rozdělen do dvou aplikací - ***Terminal*** a ***Karta***. *Terminal* je aplikací pro terminál, která obsahuje metodu generování veřejného a soukromého klíče (*KeyGen*), která může být realizována programem C, dále vytvoření zprávy m a její zašifrování do šifry c (*Enc*), jak je naznačeno na obrázku 4.3 a 4.4, přičemž program v jazyce C provede popsané operace.

Druhá aplikace *Karta* obsahující metodu pro rozšifrování šifry c na zprávu m bude implementována přímo na čipovou kartu. Dílčí metody a jejich aplikace jsou blíže popsány v kapitole 4.3.

Po rozdělení mají aplikace *Terminal* velikost 36kB a *Karta* 22,2kB. V našem případě tedy aplikace na čipové kartě zabírá přibližně třetinu dostupné paměti EEPROM, je zde však ještě potřeba uložit soukromý klíč, který dle tabulky 3.2 čítá 1888 bytů, dohromady je tedy přibližně potřeba 24,5kB paměti EEPROM, což je pod polovinu dostupného místa na kartě. Samotná aplikace napsána v programovacím jazyce Basic bude zabírat odlišnou velikost kvůli uložení globálních proměnných do perzistentní paměti, neměla by se však příliš lišit, její srovnání pak bude vysvětleno v kapitole 4.3.

Tab. 4.3: Procesorový čas trávený nad výpočty dílčích metod na PC

Funkce	**Procesorový čas**
KeyGen	4 µs
Enc	5 µs
Dec	1 µs

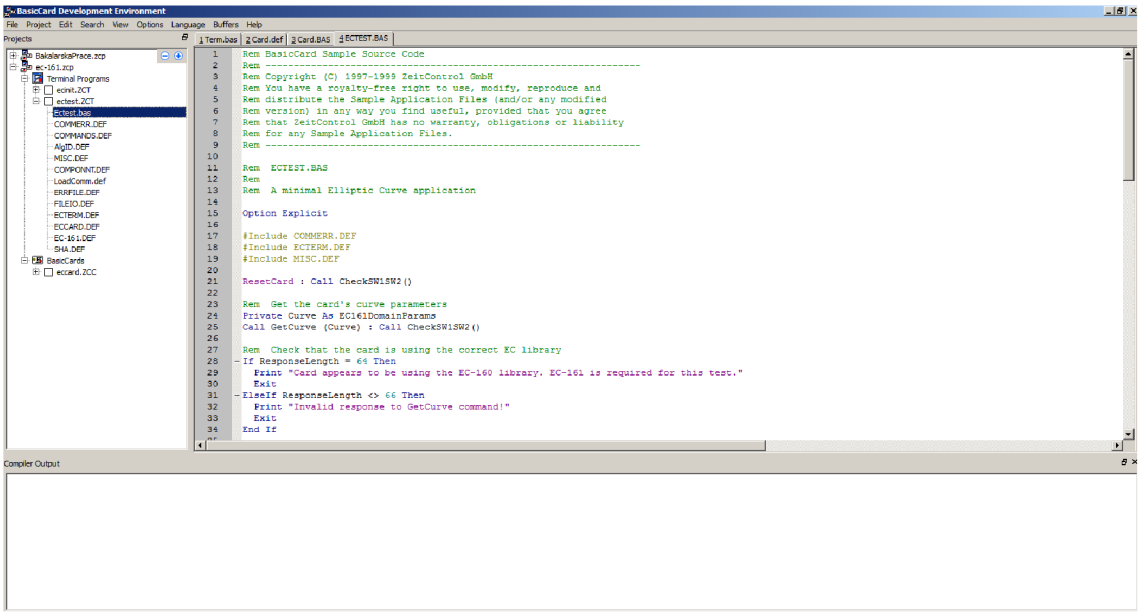
Pro další účely srovnání je vhodné změřit procesorový čas strávený na operacích KeyGen, Enc a Dec, jejichž hodnoty jsou uvedeny v tabulce 4.3. Podle naměřených hodnot lez určit, že funkce dešifrování (Dec) je procesorem nejrychleji zpracována, jedná se tedy o procesorově nejméně náročnou operaci, což je z hlediska implementace na čipovou kartu velmi přínosné.

4.3 Implementace dílčích metod na čipovou kartu

V této kapitole se práce věnuje popisu již dříve zmíněné metody Dec, jejíž funkce jsou na kartu implementovány v jazyce Basic. Verifikační metoda je implementována dle obrázku 4.4, přičemž terminál představuje PC, ke kterému je připojeno zařízení, které je využíváno pro zápis a čtení dat z karty. Čipová karta z obrázku potom představuje samotnou BasicCard ZC7.6 rev D použitou pro implementaci.

K programování čipové karty je využíváno prostředí BasicCard Development Environment (na obr. 4.5), které nabízí mnoho funkcí a možností operací s čipovou kartou, která je čtečkou propojena s počítačem. Tento program je spolu s dokumentací, která je však jediným zdrojem popisu funkce syntaxe a sémantiky námi využívaného upraveného jazyka Basic, dostupná na webových stránkách výrobce[30].

Pro implementaci náročné metody na velmi omezené zařízení je velmi důležité využívat co nejvíce „vnějších zdrojů“, které je třeba využít k výpočtu funkcí, či operací,



Obr. 4.5: Prostředí programu BasicCard Development Environment

které nejsou třeba počítat přímo na kartě. Tím se ušetří výpočetní výkon potřebný pro tento protokol, tedy využití prostředků karty, kterých je pro samotnou funkci protokolu velmi málo. Další důležitou redukcí zatěžování karty je statické zapsání globálních proměnných do paměti EEPROM hned na začátku běhu aplikace karty.

K vygenerování klíčů i samotnému zašifrování zprávy může být tedy využit program C. Z teorie vyplývá, že pokud bude šifrovací metoda využívat stejné funkce, jako dešifrovací (jen opačné), tak je metoda plně funkční. Je tedy třeba dbát na funkčnost dílčích funkcí, které musí počítat stejné hodnoty, jako program napsaný v jazyce C.

4.3.1 Funkce hexToType

Po vygenerování klíčů a zašifrování zprávy do šifry je stěžejním problémem implementace dešifrovací metody na čipovou kartu. Soukromý klíč metody NewHope-512-CCA má 1888 bytů, avšak datové typy karty podporují maximální velikost proměnných 2048 bitů. Pro jeho uložení je tedy potřeba zapsat jej jako statickou proměnnou, inicializovanou na začátku programu karty. Nutnost tohoto řešení je dána velikostí klíče, kvůli které jej nelze zapsat v jiném datovém typu. O převod klíče z vygenerovaného formátu do bytového pole o velikosti 1888 bytů (tedy 15104 bitů) se stará jednoduchá funkce *hexToType*, psaná v jazyce Java, která jako parametr přebírá klíč

v hexadecimální podobě, a po dvojicích oddělené námi potřebnými znaky vrací jako výsledek.

Jazyk Basic s velkými čísly, jako je například náš soukromý klíč, počítá v bytových polích, které jsou uloženy do proměnné String ($). Například hodnota (A15F) je funkcí *hexToType* převedena na řetězec (&Ha1, &H5f), který reprezentuje bytové pole, přičemž každý prvek může nabývat hodnoty 0-255. Takto zapsaná sekvence znaků lze potom přiřadit proměnné v jazyce Basic následovně:

```
private promenna$
promenna$ = Chr$(&Ha1, &5f)
```

Funkce Chr$() zde plní úlohu konverze do datového typu *String* požadovaného pro uložení do proměnné *promenna$*. Po takto provedené funkci je v proměnné *promenna$* námi požadovaná hodnota.

V případě soukromého klíče staticky ukládaného do paměti karty EEPROM nastává problém uložení klíče do proměnné, protože zápis klíče je příliš dlouhý. To je vyřešeno uložením klíče do bytových polí SKa$, SKb$, SKc$a SKd$, které lze poté příkazem

```
SKall$=SKa$+SKb$+SKc$+SKd$
```

Poskládat dohromady, přičemž znaménko „+“ v této operaci neznačí operaci sčítání, avšak konkatenaci (spojení) bytových bloků do jednoho velkého bytového pole. Funkce *hexToType* slouží pouze pro urychlení, aby takto velké hodny nemusely být přepisovány ručně, nebo počítány na samotné kartě.

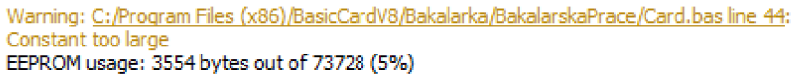
Šifrovaná zpráva *ct* je díky její velikosti také uložena v bytovém poli, avšak bez nutnosti jejího rozdělení do více částí.

4.3.2 Knihovna BigInt

Jednou z nejdůležitějších částí karty BasicCard je její knihovna BigInt. Knihovna obsahující aritmetické, či logické operace nad velkými čísly je jeden z hlavních faktorů výběru karty. Číslo je, jak již bylo zmíněno, ukládáno do bytoých polí, které jsou v hodnotě *String* zapsány do proměnné a je s ním možno provádět všechny aritmetické operace, včetně bitových posunů či konverze čísla do jiných datových typů. Číslo uložené jako bytové pole do hodnoty *String* může nabývat hodnot $0–2^{16384}-1$. Tato vlastnost je vhodná pro naši metodu zejména kvůli možnosti uložení velkého čísla, avšak možnost zápisu pouze pozitivních čísel je následně vyžadována v dalších

metodách, které využívají bitový posun (je nežádoucí se posunem bitů dostat do negativních hodnot). Tato vlastnost umožňuje uložení tzv. „*uint_32*“ čísel známých z jazyka C, která jsou metodou využívána.

Velký problém nastává, když je např. volána funkce, kde parametr představuje několika bytové číslo, které bylo vypočítáno některou z funkcí programu. Toto číslo totiž může být ve tvaru Int, nebo Double a jeho konverze z dekadického tvaru do tvaru bytových polí reprezentovaných Stringem je nutno provádět přes dílčí funkce knihovny, které však nepřijímají takto velké argumenty nutné ke konverzi (obrázek 4.6).



Obr. 4.6: Chyba při konverzi velkých čísel

Je proto nutno všechny funkce uzpůsobit tak, aby počítaly s bytovými poli, které jsou uloženy do hodnoty *String* a dále knihovnu BigInt ke své funkcionalitě využívali.

4.3.3 Funcke Reduce

Další stěžejní funkcí je funkce 4.1. Ta je využívána pro redukci 32 bitového bezznaménkového integeru (rozsah 0 až 4294967296) na 16 bitový (rozsah 0 až 65536). K násobení velkých čísel je zde využito bitového posunu, který šetří nároky výpočtu. Je zde využito vlastnosti, že výsledné číslo, které je funkcí vraceno, je kongruentní s číslem vstupním násobené $R^{-1} mod q$, kde R je 2^{18}, viz ukázka kódu 4.1.

$$a \times R^{-1} \, mod \, q \tag{4.1}$$

Kde a je vstupní parametr a q je globální proměnná určující modulo ve kterém protokol počítá.

Výpis 4.1: Příklad implementace Montgomeryho redukce v jazyce Basic

```
// funkce Reduce
#include BigInt.def
Declare Function Reduce (Hodnota$) As String

Function Reduce (Hodnota$) As String
private qinv$
private rlog%
private u$
private mezivysledek$
private jedna$
private vysledek$
private znamenko%
private konecna$

'inverse mod p,2^18
qinv$ = Chr$(&H2f, &Hff)
rlog% = 8
'aby bylo mozno volat funkci knihovny bigint
jedna$ = Chr$(&H01)
vysledek$ = Hodnota$

u$ = BigIntMul(Hodnota$, qinv$)
mezivysledek$ = BigIntShiftLeft(jedna$, rlog%)
mezivysledek$ = BigIntSub(mezivysledek$, jedna$, znamenko%)
u$ = BigIntAnd(u$, mezivysledek$)
u$ = BigIntMul(u$, newhopeQ$)
vysledek$ = BigIntAdd(vysledek$, u$)
konecna$ = BigIntShiftRight(vysledek$, rlog%)
Reduce = konecna$
end Function
```

Opět je využito předpočítání neměnných hodnot, jako je inverzní prvek k prvku $R \, mod \, q$ (proměnná qinv$).

4.3.4 Předpočítané hodnoty

Následkem omezení volby parametrů protokolu NewHope je možnost uložení dalších předem vypočítaných proměnných do datového typu Array (pole) a indícií, podle kterých je v dalších funkcích s poli zacházeno.

Následující výčet proměnných představují předpočítané hodnoty pro NTT (Number Theoretic Transform).

bitrev_table – pole obsahující bitově obrácené 9-bitové indexy, které jsou poté použity pro opětovné seřazení polynomů před využitím NTT.
omegas_bitrev_montgomery – pole obsahující mocniny n-tého kořene Montgomeryho domény s $R = 2^{18}$, v bitově obráceném pořadí.
omegas_inv_bitrev_montgomery – pole obsahující inverzní prvky mocnin n-tého kořene Montgomeryho domény, seřazeny v bitově obráceném pořadí.
psis_bitrev_montgomery –pole obsahující mocniny n-tého kořene mínus jedné v Montgomeryho doméně, seřazeny v bitově obráceném pořadí.
psis_inv_bitrev_montgomery – pole obsahující inverzní prvky k prvkům pole **psis_bitrev_montgomery**

Všechny předem vypočítané hodnoty jsou počítány s parametry protokolu NewHope, tedy $n = 512,\ q = 12289$ a $R = 2^{18}$.

Z implementačního hlediska je ukládání zmíněných hodnot do polí obsahující elementy typu *Integer* velmi výhodné kvůli nepotřebě konverze prvků. Přístup k prvkům pole má konstantní složitost, což vede ke zrychlení metod. Vybraný prvek typu *Integer* je potom nutno přetypovat přímo v metodě, která jej volá, na bytové pole typu String aby s ním mohly počítat další funkce protokolu.

4.3.5 Polynomy

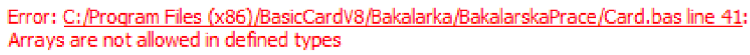
Samotný protokol NewHope pracuje s polynomy, ve kterých počítá nejdůležitější operace, kvůli kterým je protokol NewHope bezpečný a zároveň efektivní. Iplementace polynomů je však v programovacím jazyce Basic velmi náročná. Je nutno vytvořit vlastní datový typ obsahující složky polynomu. K tomu je potřeba využít pole **coeffs**, přičemž jsou jeho pozice obsazeny samotnými koeficienty polynomu. Polynom, jak je naznačeno v kapitole 2.2, je reprezentován prvky
coeffs[0] + X × coeffs[1] + X^2 × coeffs[2] + ... + $X^{(n-1)}$ × coeffs[i].

Pro vytvoření námi zvoleného datového typu slouží v jazyce Basic sekvence příkazů dle ukázky 4.2.

Výpis 4.2: Vytvoření vlastního datového typu poly

```
Type poly                                                    1
  coeffs(0 to 511) As Integer                                2
End Type                                                     3
```

Vytvoření požadovaného datového typu není však podporován viz obrázek č. 4.7.



Obr. 4.7: Chyba při vytvoření datového typu poly

Tento problém je zásadní z hlediska dalších funkcí, které s polynomy počítají. Uložení polynomu do pole je z hlediska logiky, funkcionality i efektivity metod velmi důležité a bez této možnosti jsou další funkce protokolu nefunkční. Další stěžejní funkce protokolu NewHope (konverze polynomu z bytů, dekomprese polynomů, převody polynomů, násobení či odečítání polynomů), jejichž implementace je závislá na datovém typu *poly* by tedy bylo třeba modifikovat, což spadá do velmi obtížných modifikací kódu a narušení struktury programu, které by dále mohlo vést k jeho celkové nefunkčnosti.

4.3.6 Shrnutí

V této kapitole byla zahrnuta implementace některých funkcí verifikační části protokolu NewHope. Aplikace některých datových typů a samotný přepis kódu je velmi obtížný kvůli odlišné syntaxi a sémantice jazyků. Implementované funkce jsou otestovány na menších hodnotách z důvodu složitosti ověření hodnot, které jsou dány parametry protokolu NewHope.

Soukromý klíč, který je třeba uložit do čtyřech bytových polí může mít v návaznosti jeho následné konkatenace za následek špatných výsledků funkce. Tato skutečnost musí být ovšem vyvrácena po vykonání všech ostatních procesů z důvodu složitosti ověření. Celkové počítání s velkými čísly je jeden z předpokladů, proč je metoda složitá na implementaci a ověření její funkčnosti.

Ostatní předem vypočítané hodnoty i jiné parametry byly i přes jejich délku nahrány na čipovou kartu do paměti EEPROM z důvodu šetření paměťových zdrojů

programu. Po načtení implementovaných funkcí, hodnot a parametrů na kartu, zabírá program celkově 8042 bytů, což je přibližně 11% využití paměti EEPROM. Lze tedy předpokládat, že i s chybějícími funkcemi je verifikační část protokolu NewHope pro čipovou kartu paměťově přijatelná, avšak výpočetní náročnost nelze posoudit kvůli necelistvosti programu. Hlavním problémem implementace je potom nemožnost vytvoření vlastního datového typu *poly*, které je stěžejní pro počítání dané metody. Z výše uvedených důvodů se tedy nepodařilo nahrát celou verifikační část metody NewHope na čipovou kartu, což značí obtížnost post-kvantového zabezpečení velmi paměťově i výpočetně omezených zařízení.

Obecně lze říci, že je implementace post-kvantového zabezpečení na čipové karty velmi obtížná, protože daná zabezpečení s sebou přináší komplexnější operace, počítání s mnohonásobně většími čísly a jiné komplikace, které jsou v předešlých kapitolách popsány. Složitost implementace i přes volbu vhodné metody a čipové karty značí, že zabezpečení čipových karet proti kvantovým počítačům je v dnešní době téma, kterému je třeba věnovat pozornost, aby byla bezpečnost těchto zařízení zachována. Jak naznačuje vývoj post-kvantových metod a čipových karet, je možné, že budou postupem času efektivnější metody a vybavenější systémy, jejichž zabezpečení by bylo méně náročné. V dnešní době lze však předpokládat, že s nástupem kvantových počítačů by hojně využívaná zařízení, jako je čipová karta, byla považována za nebezpečná.

5 Závěr

Práce pojednává o nástupu kvantových počítačů a jeho dopadu na zabezpečení běžně využívaných systémů. V teoretické části práce je vysvětlen pojem post-kvantová kryptografie a její metody, které jsou dále analyzovány. Z hlediska vymezení práce byly detailněji popsány protokoly využívající kryptografie založené na kódování, na supersingulárních izogenních eliptických křivkách, na mřížkách a na hashovacích funkcích.

V rámci praktické části se práce zaměřila na analýzu a měření paměťové a výpočetní náročnosti jednotlivých protokolů za účelem implementace na velmi omezené zařízení. Naměřené hodnoty byly srovnány a analyzovány na virtuálním PC s využitím zmíněných knihoven. Na základě naměřených hodnot byla zvolena metoda NewHope, vzhledem její výpočetní i paměťové náročnosti. Její volba byla také zasazena do současného vývoje post-kvantových systémů pro omezená zařízení s potvrzením měření i teoretických poznatků o její efektivitě. S ohledem implementace na omezené zařízení byla metoda modifikována a byl popsán návrh verifikační části protokolu na úrovni karta-terminál. Takto modifikovaná metoda byla měřena z pohledu atomických funkcí, které obsahuje, pro srovnání náročností daných částí protokolu.

Následně práce seznamuje s čipovými kartami a problematikou jejich zabezpečení proti kvantovým počítačům. S ohledem vnitřní logiky modifikované části metody NewHope jsou srovnány výhody i nevýhody dnešních čipových karet za účelem jejich zabezpečení. Čipová karta BasicCard typu Z 7.6D byla vzhledem k jejím vlastnostem zvolena jako nejvýhodnější k aplikaci dané metody. V poslední kapitole je popisována přímá implementace dílčích metod na čipovou kartu spolu s potřebnými nástroji, které jsou k tomu využity. Byly implementovány některé z dílčích funkcí metody a popsány operace, které byly potřebné kvůli šetření výpočetních nároků na čipovou kartu. Podrobněji je popsána úspěšná část implementace i problémy, které při implementaci nastaly a vedly k neúplné aplikaci metody NewHope na čipovou kartu.

Práce si kladla za cíl seznámit čtenáře s problematikou post-kvantové kryptografie a jednotlivými protokoly, které mohou být v současnosti využívány. Zároveň si stanovila za cíl vybrat vhodné schéma pro implementaci na omezené zařízení a navrhnout její verifikační metodu následně implementovanou na vhodné čipové kartě. Cíle byly z velké části splněny. Na základě analýzy byla v práci určena nejvhodnější metoda, která byla následně modifikována a některé její dílčí funkce byly implementovány na vybrané platformě čipových karet. Z neúspěšné implementace lze odvodit,

že čipové karty nejsou zcela připraveny na zabezpečení proti kvantovým počítačům kvůli složitosti kryptografických metod spolu s velkým paměťovým i výpočetním omezením zařízení.

Literatura

[1] CHEN, Lily, et al. *Report on post-quantum cryptography*. US Department of commerce, National Institute of Standards and Technology

[2] MAVROEIDIS, Vasileios. *The Impact of Quantum Computing on Present Cryptography* [online]. Department of Informatics, University of Oslo, Norway, 2018, 3. 11. 2018 [cit. 2018-12-17]. Dostupné z: <`https://arxiv.org/pdf/1804.00200.pdf`>

[3] DIVIŠOVÁ, Jana. *Kryptografie založená na mřížích*. Praha, 2010. Diplomová práce. Univerzita Karlova v Praze. Vedoucí práce RNDr. David Stanovský, Ph.D.

[4] HOFFSTEIN, Jeffrey, Jill Catherine PIPHER a Joseph H. SILVERMAN. *An introduction to mathematical cryptography*. London: Springer, 2008. ISBN 9780387779935.

[5] MICCIANCIO, D. *Lattice-based Cryptography*. Courant Institute of Mathematical Sciences [online], 2008. [cit. 2018-11-10]. Dostupné z:<`https://www.cims.nyu.edu/regev/papers/pqc.pdf`>

[6] REGEV, Oded. *The Learning with Errors Problem* [online]. [cit. 2018-11-17]. Dostupné z: <`https://cims.nyu.edu/regev/papers/lwesurvey.pdf`>

[7] POPELOVÁ, Lucie. *Metody post-kvantové kryptografie*. Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně. Vedoucí práce Ing. Lukáš Malina, Ph.D.

[8] MICCIANCIO, Daniele. *Generalized compact knapsacks, cyclic lattices, and efficient one-way functions* [online]. University of California, San Diego [cit. 2018-11-15]. Dostupné z: <`https://cseweb.ucsd.edu/daniele/papers/Cyclic.pdf`>

[9] BALDI, M. et al. *Soft McEliece: MDPC code-based McEliece cryptosystems with very compact keys through real-valued intentional errors*. Universita Politecnica delle Marche, 2016. ISBN 978-1-5090-1806-2.

[10] WANG, Wen, et al. *FPGA-based Niederreiter Cryptosystem using Binary Goppa Codes*, Germany [online].[cit. 2018-10-15]. Dostupné z: <`https://eprint.iacr.org/2017/1180.pdf`>

[11] KOTIL, Jaroslav. *Goppovy kódy a jejich aplikace*. Praha, 2013. Diplomová práce. Karlova Univerzita v Praze. Vedoucí práce Prof. RNDr. Aleš Drápal, CSc., DSc. [online].[cit. 2018-10-18]. Dostupné z: <https://is.cuni.cz/webapps/zzp/detail/77948/?lang=en>

[12] UNHUR, Dominique. *Collapsing sponges: Post-quantum security of the sponge construction*, University of Tartu [online]. 2017 [cit. 2018-10-19]. Dostupné z: <https://eprint.iacr.org/2017/282.pdf>.

[13] BERNSTEIN, Daniel J. *Post-Quantum Cryptography*. Technische Universität Darmstadt, Darmstadt, Německo: Springer, 2009. ISBN 9783540887027.

[14] WOLF, Christopher a Bart PRENEEL. *ASYMMETRIC CRYPTOGRAPHY: HIDDEN FIELD EQUATIONS* [online]. Leuven-Heverlee, Belgium, 2004 [cit. 2018-11-15]. Dostupné z: <https://eprint.iacr.org/2004/072.pdf>

[15] GALBRAITH, D.S. *Supersingular Curves in Cryptography*. University of London [online], n.d.. [cit. 2018-11-17]. Dostupné z URL: <https://www.iacr.org/archive/asiacrypt2001/22480497.pdf>

[16] HUYNH, A. *An Introduction to Supersingular Elliptic Curves and Supersingular Primes*. University of Washington [online], n.d.. [cit. 2018-11-17]. Dostupné z URL: <https://wstein.org/edu/2011/581g/final/anh-supersingular_elliptic_curves.pdf>

[17] *SAFEcrypto* [online]. [cit. 2018-11-17]. Dostupné z: <https://www.safecrypto.eu>

[18] ANTICI, C.A., et al. *Low-cost implementations of NTRU for pervasive security* . Leuven, Belgium, 2008. ISBN 978-1-4244-1897-8.

[19] HOFFSTEIN, J., et al. *NTRUSign: Digital Signatures Using the NTRU Lattice*. Springer Publishing Company, Incorporated, 2003. ISBN 978-3-540-00847-7.

[20] ALKIM, E. et al. *Post-quantum key exchange – a new hope*. Department of Mathemathics, Ege University, Turkey. [cit. 2018-20-11]. Dostupné z: <https://eprint.iacr.org/2015/1092.pdf>

[21] BOS, J., et al. *CRYSTALS – Kyber: a CCA-secure module-lattice-based KEM* .Microsoft Research, NXP. [cit. 2018-11-21]. Dostupné z: <https://eprint.iacr.org/2017/634.pdf>

[22] BOS, J., et al. Frodo: *Take off the ring! Practical, Quantum-Secure Key Exchange from LWE*. ACM CCS. [cit. 2018-11-25]. Dostupné z: <https://eprint.iacr.org/2016/659>

[23] HESAMIAN, S. *Analysis of BCNS and Newhope Key-exchange Protocols.* University of Wisconsin Milwaukee [online], 2017. [cit. 2018-12-10]. Dostupné z: <https://dc.uwm.edu/cgi/viewcontent.cgi?referer=https://www.google.com/&httpsredir=1&article=2490&context=etd>.

[24] Robert J. McEliece, *A Public-Key Cryptosystem Based on Algebraic Coding Theory*, JPL Deep Space Network Progress Report, 1978. [cit. 2018-11-25]. Dostupné z:<http://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF>

[25] YUAN, Ye., et al. *Memory-Constrained Implementation of Lattice-based Encryption Scheme on Standard Java Card* [online]. Japonsko [cit. 2018-04-18]. Dostupné z: <https://eprint.iacr.org/2018/1238.pdf>

[26] BOORGHANY, Ahmad, JALILI, Rasool, *Implementation and Comparison of Lattice-based Identification Protocols on Smart Cards and Microcontrollers* [online]. Írán, 2014 [cit. 2019-04-21]. Dostupné z: <http://ce.sharif.edu/~boorghany/pubdown/latticeid-v1.pdf>

[27] BOORGHANY, A., SARMIDI, S. B., JALILI, R., *On constrained implementation of lattice-based cryptographic primitives and schemes on smart cards* [online]. Írán, 2014 [cit. 2019-04-21]. Dostupné z: <https://eprint.iacr.org/2014/514.pdf>

[28] STRENZKE, Falko, *A smart card implementation of the McEliece PKC. In IFIP International Workshop on Information Security Theory and Practices* [online] Německo [cit. 2019-04-21]. Dostupné z:<http://opendl.ifip-tc6.org/db/conf/wistp/wistp2010/Strenzke10.pdf>

[29] KONEČNÝ, J. *Silná kryptografie na čipových kartách.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. Ústav telekomunikací, 2017. Diplomová práce. Vedoucí práce: Ing. Lukáš Malina, Ph.D

[30] *BasicCard* [online]. [cit. 2019-04-23]. Dostupné z: <http://www.basiccard.com/index.html?overview.htm>

Seznam symbolů, veličin a zkratek

AES	Advanced Encryption System – Symetrický systém
ARRAY	Datový typ pole – pro seznam prvků
BCNS	Post-kvantový kryptosystém založený na mřížkách
CCA	Choden Cipher Attack – druh útoku
CPA	Chosen Plaintext Attack – druh útoku
CVP	Closest Vector Problem – Problém nejbližšího vektoru
DSA	Digital Signature Algorithm – Asymetrický kryptosystém
ECC	Elliptic Curve Cryptography – Kryptografie založená na eliptických křivkách
ECDH	Elliptic Curve Diffie Hellmann – Asymetrický kryptosystém
ECDSA	Elliptic Cutve Digital Signature Algorithm – Algoritmus asymetrické kryptografie
EEPROM	Electrically Erasable Programmable Read-Only Memory – Elektronicky programovatelná paměť pouze pro čtení)
FS	Fiat Shamir Transformation – Fiat Shamirova transformace
FTT	Fast Furier Transformation – Furierova transformace
GB	Gigabyte
GHz	Gigahertz
HFE	Hidden Fields Equations – Asymetrický kryptosystém
Integer	Celočíselný datový typ
KB	Kilobyte
KEM	Key Encapsulation Mechanism
LLL	Lenstra-Lentstra-Lovász – Algoritmus redukce na mřížkách
LWE	Learning With Errors – Matematický problém využívaného mřížkami
NIST	National Institute of Stanndards and Technologies – Národní institut standardů a technologií
NP	Nondeterministic Polynomial Problém – Problém neřešitelný v polynomiálním čase
NTT	Number Theoretic Transform – funkce číselné transformace
NVM	Non-volatile memory – Nevolatilní paměť
PKC	Public Key Cryptography – Asymetrická kryptografie
RAM	Random Access Memory – Volatilní paměť využívaná programy
ROM	Read Only Memory – Paměť určená pouze ke čtení
SHA	Secure Hash Algorithm – Hashovací algoritmus
SHAKE	Secure Hash Algorithm Key Exchange – Hashovací funkce pro generování parametrů
SIDH	Supersingular Isogeny Diffie-Hellman –Post-kvantový protokol pro

	výměnu klíčů
SIKE	Supersingulare Isogeny Key Exchange –Post-kvantový protokol pro ustanovení klíčů
SIS	Short Integer Solution Problem – Problém v mřížkově založené kryptografii
SK	Soukromý klíč
String	Datový typ řetězce znaků
SVP	Shortest Vector Problem – Problém nejkratšího vektoru
UOV	Unbalanced Oil and Vinegar – Kryptografické schéma
UR	Unhur's transformation – Unhurova transformace
VK	Veřejný klíč
µs	Mikrosekunda

Seznam příloh

A Obsah přiloženého DVD

Na přiloženém médiu se nachází samotná práce v podobě pdf, zdrojové kódy psané v jazyku C, Java i Basic rozděleny do vlastních složek.

```
/ ..... kořenový adresář přiloženého CD
├── Basic ..... zdrojové kódy v Basic
├── C ..... soubory C
│   ├── Modifikace NewHope
│   ├── NewHope
│   └── NewHope Card
├── Elektronická verze ..... pdf verze práce
├── Java ..... program psaný v javě
└── README.txt ..... soubor s popisem obsahu DVD
```

A.1 Basic

Složka Basic obsahuje zdrojové kódy dílčích metod post-kvantové kryptografie NewHope.

A.2 C

Soubory C jsou logicky rozděleny podle toho, jak s nimi bylo manipulováno. V podsložce NewHope jsou tedy zdrojové kódy protokolu NewHope v původní podobě. Podsložka Modifikace NewHope obsahuje zdrojové kódy zredukované o nepotřebné funkce na rozhraní verifikace karta-terminál a NewHope Card obsahuje upravenou iplementaci funkcí, které jsou poté implementovány v jazyce Basic (pouze funkce plnitelné kartou)

Pro překlad souborů C je potřeba překladač GCC (využito gcc verze 9.1)

A.3 Elektronická verze

Složka obsahuje elektronickou verzi bakalářské práce ve formátu pdf.

A.4 README.txt

Soubor popisující obsah DVD v textové formě.