FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání diplomové práce

Název:	Pokročilé algoritmy pro sdílení tajemství
Student:	Bc. Hana Svobodová
Vedoucí:	Ing. Josef Kokeš
Studijní program:	Informatika
Obor / specializace:	Počítačová bezpečnost
Katedra:	Katedra informační bezpečnosti
Platnost zadání:	do konce letního semestru 2022/2023

Pokyny pro vypracování

1) Seznamte se se základními algoritmy pro sdílení tajemství. Zformulujte jejich hlavní nedostatky a možné směry řešení.
2) Nastudujte algoritmy, které tyto nedostatky řeší - ověřitelné sdílení tajemství a multiparty computing.
3) Analyzujte jejich bezpečnostní vlastnosti, zformulujte možné hrozby, zejména ve vztahu k použití kvantového počítače.
4) Nastudujte dále algoritmy pro sdílení tajemství realizované pomocí kvantového počítače.
5) Demonstrujte hlavní popsané algoritmy pomocí vhodné implementace.
6) Diskutujte dosažené výsledky.

Elektronicky schválil/a prof. Ing. Róbert Lórencz, CSc. dne 7. února 2022 v Praze.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Pokročilé algoritmy pro sdílení tajemství

Bc. Hana Svobodová

Katedra informační bezpečnosti
Vedoucí práce: Ing. Josef Kokeš

25. dubna 2022

Poděkování

Ráda bych tímto poděkovala Ing. Josefu Kokešovi za veškeré rady, připomínky a vstřícný přístup při zpracování této práce. Dále bych ráda poděkovala své rodině za trpělivost a podporu při studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracovala samostatně a že jsem uvedla veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu) licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 25. dubna 2022 . . . . . .

Odkaz na tuto práci

Svobodová, Hana. *Pokročilé algoritmy pro sdílení tajemství.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2022.

Abstrakt

Tato práce se zabývá algoritmy pro sdílení tajemství. Konkrétně se zaměřuje na ověřitelná a kvantová schémata pro sdílení tajemství a metodu multiparty computing. Rešeršní část se zaměřuje na popis vybraných schémat a jejich bezpečnost. Náplní praktické části je implementace vybraných algoritmů, konkrétně Shamirova algoritmu pro sdílení tajemství, Pedersenova ověřitelného a Schoenmakersova veřejně ověřitelného schématu a schéma BGW pro multiparty computing. Kvantové schéma pro sdílení tajemství je implementováno pomocí simulátoru. Výstupem je aplikace, která dokáže pomocí výše zmíněných schémat rozdělit nebo rekonstruovat tajemství.

Klíčová slova sdílení tajemství, ověřitelné sdílení tajemství, multiparty computing, kvantové sdílení tajemství

Abstract

This thesis deals with secret sharing algorithms. In particular, it focuses on verifiable secret sharing algorithms, quantum secret sharing and multiparty computation. The survey part focuses on the description of selected algorithms and their security properties. In the second part, Shamir's scheme, Pedersen's verifiable scheme and Schoenmakers' publicly verifiable secret sharing scheme and BGW scheme for multiparty computing are implemented. The quantum secret sharing scheme is implemented on a simulator. The main result is an application which can split and reconstruct secret according to the implemented algorithms.

Keywords secret sharing, verifiable secret sharing, multiparty computing, quantum secret sharing

Obsah

Úvod **1**

1 Cíl práce **3**

2 Základní sdílení tajemství **5**
2.1 Základní definice 5
2.2 Shamirovo schéma pro sdílení tajemství 6
2.3 Slabiny a nečestní účastníci 7
2.3.1 Podvržení dílu u Shamirova sdílení tajemství 7
2.3.2 Obměna dílů 8

3 Ověřitelné sdílení tajemství **9**
3.1 Závazková schémata 9
3.1.1 Naorovo závazkové schéma 10
3.1.2 Pedersenovo závazkové schéma 12
3.1.3 Schéma založené na hashovací funkci 13
3.2 Ověřitelné sdílení tajemství 15
3.2.1 Pedersenovo ověřitelné schéma 16
3.2.2 Schoenmakersovo veřejně ověřitelné schéma 17

4 Bezpečné počítání více stran **21**
4.1 Schéma založené na *garbled circuit* 22
4.1.1 Garbled circuit 22
4.1.2 Oblivious transfer 24
4.1.3 Počítání více stran pomocí garbled circuit 26
4.2 Schéma BGW (Ben-Or, Goldwasserová, Wigderson) 26
4.2.1 Aritmetické operace schématu BGW 27
4.2.2 Počítání více stran pomocí schématu BGW 29

5 Kvantové sdílení tajemství **31**

5.1 Kvantové počítání 32
5.1.1 Notace a základní vlastnosti kvantového počítání 32
5.1.2 Kvantové operace 35
5.2 Schéma Hillery, Bužek, Berthiaume 39

6 Realizace **41**
6.1 Knihovna `vsssLib` 41
6.1.1 Shamirův algoritmus 41
6.1.2 Pedersenovo ověřitelné schéma 43
6.1.3 Schoenmakersovo veřejně ověřitelné schéma 44
6.1.4 BGW schéma 46
6.2 Demo aplikace 46
6.2.1 Práce se soubory 46
6.2.2 Použití knihovny `vsssLib` 48
6.2.3 Testování rychlosti 49
6.3 Obvod realizující kvantové schéma pro sdílení tajemství 50

Závěr **55**

Bibliografie **57**

A Uživatelská příručka **65**
A.1 Demo aplikace 65
A.1.1 Požadavky 65
A.1.2 Spuštění 65
A.2 Skript `hbb.py` 67
A.2.1 Požadavky 67
A.2.2 Spuštění 67

B Obsah přiloženého CD **69**

Seznam obrázků

5.1 Schematické značky kvantových hradel 37
5.2 Vytvoření Bellova a GHZ stavu pomocí kvantových hradel 38

6.1 Aritmetický obvod realizující funkci hlasování s pěti voliči 49
6.2 Aritmetický obvod realizující funkci f se všemi třemi typy hradel . 49
6.3 Čas průběhu funkcí rekonstruujících tajemství 50
6.4 Schéma obvodu pro rozdělení a rekonstrukci tajného qubitu $|S\rangle$ . . 52
6.5 Příklad výstupu spuštění skriptu `hbb.py` s výchozími parametry . 53

Seznam tabulek

4.1 Transformace pravdivostní tabulky hradla AND 23

5.1 Tabulka transformací qubitu na tajný qubit. 40

6.1 Použité transformace k získání tajného qubitu. 52

Seznam algoritmů a výpisů

1 Struktura reprezentující jeden díl v Shamirově sdílení tajemství 42
2 Lagrangeova interpolace v bodě *point* modulo *mod* 42
3 Struktura dílu v Pedersenově ověřitelném sdílení tajemství . . . 43
4 Struktura veřejných parametrů Pedersenova sdílení tajemství . 44
5 Výpočet závazku k dílu v Pedersenově sdílení tajemství 44
6 Struktura hodnot Chaumova-Pedersenova protokolu 45
7 První a druhá fáze hradla násobení v protokolu BGW 47
8 Vytvoření prázdného obvodu 53
9 Vytvoření provázaného GHZ stavu tří qubitů. 53
10 Příklad měření qubitu a podmíněné aplikace hradla X 53

Úvod

Sdílení tajemství je metoda, která řeší problém bezpečného rozdělení tajných hodnot mezi více osob, aby jednotlivci nemohli své díly zneužít a pouze jejich spoluprací je možné tajnou hodnotu zrekonstruovat. To lze využít v případě potřeby bezpečného uložení tajných hodnot, jako je kryptografický klíč, nebo při doručování tajných zpráv, kdy ani kompromitováním jednoho kurýra nedojde k odhalení zprávy.

Samotné rozdělení ale nemusí vždy stačit a je potřeba zaručit, že díly jsou platné, protože by mohly být podvrženy nebo poškozeny. Řešením jsou ověřitelná schémata pro sdílení tajemství. Někdy ale není vhodné, aby při rekonstrukci tajemství byly jednotlivé díly odhaleny ostatním, což řeší multiparty computing – bezpečné počítání více stran.

Hrozbu pro bezpečnost současné kryptografie představují kvantové počítače a jejich možnosti, a sdílení tajemství není výjimkou. Jak je ukázáno v práci, kvantové počítače kromě hrozeb přináší i nové způsoby, jak tajemství sdílet.

Práce se skládá z rešeršní a implementační části. Úvodní kapitola se zabývá základními schématy pro sdílení tajemství a jejich nedostatky. Následující kapitola řeší ověřitelná schémata pro sdílení tajemství a jejich bezpečnost. Následuje část zabývající se bezpečným počítáním více stran, pro které lze využít sdílení tajemství. Poslední kapitola rešeršní části se věnuje kvantovému sdílení tajemství. Implementační část je rozdělena do tří částí, první z nich je implementace knihovny, která obsahuje vybrané algoritmy popsané v této práci. Druhou částí je demonstrační aplikace využívající implementovanou knihovnu a která umožňuje rozdělovat a rekonstruovat tajemství několika různými algoritmy. Třetí část popisuje implementaci kvantového sdílení tajemství.

Kapitola 1

Cíl práce

Cílem rešeršní části práce je seznámení se se základními algoritmy pro sdílení tajemství, popsání jejich hlavních nedostatků a jejich možných řešení. Druhým cílem je nastudování algoritmů řešících nedostatky základních algoritmů pro sdílení tajemství, jako jsou algoritmy pro ověřitelné sdílení tajemství a multi-party computing, analýza jejich bezpečnosti, i ve vztahu k použití kvantových počítačů. Dalším cílem je nastudování algoritmů pro sdíleních tajemství realizovaných pomocí kvantového počítače.

Cílem praktické části je demonstrovat hlavní popsané algoritmy pomocí vhodné implementace.

Kapitola 2

Základní sdílení tajemství

Tato kapitola se zabývá základními schématy pro sdílení tajemství (především Shamirovým polynomiálním schématem) a jejich hlavními nedostatky.

2.1 Základní definice

V této části je zavedeno základní názvosloví, které se s těmito schématy pojí, jsou představeny různé přístupy k řešení problému a popsáno základní schéma, ze kterého vychází mnoho dalších.

Sdílení tajemství umožňuje rozdělit citlivá a důležitá data (jako např. hesla nebo kryptografické klíče) na jednotlivé díly mezi více osob takovým způsobem, že samostatné díly nedávají o rozdělených datech žádné informace a samostatně tak nejsou k užitku. Rozdělená data lze zrekonstruovat pouze složením více dílů. Bez použití sdílení tajemství by všichni museli znát celou rozdělovanou hodnotu a mohli by toho zneužít, což sdílení tajemství eliminuje. První schémata pro sdílení tajemství byla představena již v roce 1979, a to nezávisle na sobě Adi Shamirem [1] a Georgem Blakleym [2]. Shamirovo schéma je založeno na polynomech a jejich bodech zatímco Blakleyho schéma využívá geometrii, konkrétně nadroviny a jejich průnik. Tajemství se dá sdílet i jinými způsoby, např. lze využít Čínskou větu o zbytcích, jak ukázali Asmuth a Bloom [3] nebo je možné tajemství sdílet i vizuálně [4].

Definice 2.1 Schéma pro sdílení tajemství je (k, n)-*prahové*, pokud je tajemství rozděleno na n dílů takovým způsobem, že k jeho rekonstrukci postačuje libovolných k dílů. [1]

Pokud se v takovém schématu zkombinuje méně než k dílů, nemělo by být možné tajemství zrekonstruovat. V prahových schématech mají všechny díly stejnou váhu, žádný z účastníků není zvýhodněn. I s těmito schématy ale lze dosáhnout určité hierarchie, stačí, když vybraní účastníci obdrží více než jeden díl tajemství.

Definice 2.2 Nechť S a T jsou množiny všech možných tajemství, resp. dílů tajemství a nechť $\oplus$ a $\otimes$ jsou binární operace na množině S, resp. T. Každé (k, n) schéma definuje množinu funkcí $F_I : T^k \mapsto S$ pro každou podmnožinu I množiny $\{0, 1, \ldots, n\}$, jejíchž počet prvků je roven k. Tyto funkce umožňují z k dílů sestavit tajemství D: $D = F_I(D_{I_1}, \ldots, D_{I_k})$, kde $I \in i_1, \ldots, i_k$. [5]

Prahové (k, n) schéma je $(\oplus, \otimes)$-homomorfní, platí-li pro všechna $I \in \{i_1, \ldots, i_k\}$

$$D \oplus D' = F_I(D_{I_1} \otimes D'_{I_1}, \ldots, D_{I_k} \otimes D'_{I_k}) \tag{2.1}$$

Definice 2.3 $(\oplus, \otimes)$ schéma je $(\oplus, \otimes)$-kompozitní (k, n) prahové schéma, pokud je m dílčích tajemství $d_+, \ldots, d_m$ rozděleno do $d_{i,j}$ dílů, $i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$ takovým způsobem, aby celkové tajemství $D = d_1 \oplus \ldots \oplus d_m$ bylo rekonstruovatelné pomocí k kombinovaných dílů D_i, $D_i = d_{i,1} \otimes d_{i,2} \otimes \ldots \otimes d_{i,m}$. Zároveň ani znalost všech kombinovaných dílů D_i a nejvýše $k - 1$ dílčích dílů $d_{i,j}$ nedává další informace o žádném z dílčích tajemství d_i. [5]

Homomorfismus dílů je široce využitelná vlastnost. Dá se využít například ke sdílení tajemství bez distributora, kde se homomorfismu dílů používá k vypočtení celkového tajemství z dílčích tajemství [6]. Tato vlastnost se dá využít i v jiných aplikacích, např. pro multiparty computing (podrobněji v kapitole Bezpečné počítání více stran).

2.2 Shamirovo schéma pro sdílení tajemství

Shamirův algoritmus je asi nejznámější schéma pro sdílení tajemství. Toto schéma je (k, n)-prahové schéma a je založeno na polynomiální interpolaci. Tajemství představuje absolutní člen polynomu, díly tajemství jsou jednotlivé body, kterými polynom prochází. Pomocí dostatečného množství bodů lze pomocí interpolace zrekonstruovat původní polynom, tedy i tajemství. [1]

Postup pro sdílení tajemství je následující:

1. Distributor vybere tajemství $D \in \mathbb{Z}_p$, kde p je prvočíslo větší než D.
2. Distributor dále vybere polynom $Q(x)$ nad $\mathbb{Z}_p$ stupně $k - 1$

   $$Q(x) = a_0 + a_1 x + \ldots + a_{k-1} x^{k-1}, \tag{2.2}$$

   kde $a_0 = D$ a $a_1, \ldots, a_{k-1}$ jsou náhodné koeficienty ze $\mathbb{Z}_p$.
3. Distributor účastníkovi i pošle díl D_i, $i \in \{1, \ldots, n\}$:

   $$D_1 = Q(1), D_2 = Q(2), \ldots, D_n = Q(n). \tag{2.3}$$

Pokud chtějí účastníci tajemství rekonstruovat, pak původní polynom lze jednoznačně určit pomocí Lagrangeovy interpolace a alespoň k dílů, neboť

stupeň původního polynomu je $k-1$ a polynom stupně $k-1$ lze jednoznačně určit k různými body [7]:

$$\begin{aligned} Q(x) &= \sum_{i=0}^{k} D_{a_i} l_i(x) \\ l_i(x) &= \frac{(x-a_0)\dots(x-a_{i-1})(x-a_{i+1})\dots(x-a_n)}{(a_i-a_0)\dots(a_i-a_{i-1})(a_i-a_{i+1})\dots(a_i-a_k)} \end{aligned} \tag{2.4}$$

Tajemství se pak spočte jako $D = Q(0)$. [1]

Toto schéma je nepodmíněně bezpečné, protože ani $k-1$ dílů nedává žádné nové informace o tajemství. [1] Díky tomu je bezpečné i vůči kvantovým algoritmům.

Vlastnosti

Shamirovo schéma je $(+,+)$-homomorfní schéma a také $(+,+)$-kompozitní schéma. Součet dílů různých tajemství představuje součet příslušných polynomů, a jelikož tajemství představuje bod na výsledném polynomu, je součtem všech dílčích tajemství.

2.3 Slabiny a nečestní účastníci

V této části jsou popsány zásadní nedostatky základních schémat pro sdílení tajemství.

Základní schémata, jako např. Shamirovo schéma, mají zásadní nedostatek – účastník nemůže ověřit, že jeho díl není poškozený nebo podvržený. Nelze tak odhalit nečestného distributora, který by úmyslně části účastníků poslal podvržené díly, aby nemohli tajemství zrekonstruovat. Při fázi rekonstrukce nelze dále zaručit, že účastníci pošlou pravé díly, a pokud v této fázi ostatním pošlou falešné díly, ostatní nemají jak odhalit, které díly jsou pravé a které podvržené. Pokud s falešným dílem tajemství nepůjde zrekonstruovat, ostatní účastníci alespoň zjistí, že je něco v nepořádku; pokud se podaří nějakou hodnotu zrekonstruovat, nemusí být ihned zřejmé, že není správná. Útočník tak může úmyslně zabránit rekonstrukci správného tajemství, nebo dokonce pomocí dílů ostatních být jediný, kdo může zrekonstruovat správnou hodnotu.

2.3.1 Podvržení dílu u Shamirova sdílení tajemství

Jak ukázali Tompa a Wollová [8], pro Shamirovo sdílení tajemství je možné, aby nečestný účastník při rekonstrukci poskytl podvržený díl, a nebyl přitom odhalen. Princip tohoto útoku je jednoduchý; musí ostatním předložit díl, s kterým bude možné úspěšně rekonstruovat nějakou hodnotu – ale jinou, než původní tajemství.

V Shamirově schématu má každý účastník i jako svůj díl hodnotu $P(i)$. Účastníci $i_1, i_2, \ldots, i_k$ chtějí zrekonstruovat tajemství. Útočníkovi $i_j, 1 \leq j \leq k$ stačí najít polynom Q stupně nejvýše $k-1$, pro který platí:

$$Q(i) = \begin{cases} s', & \text{pro } i = 0 \\ 0, & \text{pro } i = i_1, \ldots, i_{j-1}, i_{j+1}, \ldots, i_k \\ \delta_i & \text{pro } i = i_j \end{cases} \tag{2.5}$$

Jako podvržený díl útočník použije hodnotu $P(i_j) + Q(i_j)$. Protože je Shamirovo schéma $(+,+)$–homomorfní, při rekonstrukci tajemství účastníci polynomiální interpolací získají polynom $P+Q$ místo P. Absolutním členem tohoto polynomu je $P(0) + Q(0)$, tj. $s - s'$, kde s je původní tajemství.

Obrana proti tomuto útoku je jednoduchá, ale není plně účinná. V Shamirově algoritmu se jako díly tajemství používají funkční hodnoty v bodech $1, 2, \ldots, n$. Pokud se místo nich vyberou rovnoměrně a náhodně neopakující se prvky z rozsahu $1, 2, \ldots, p-1$, útok se stane obtížně proveditelným (za předpokladu, že tyto hodnoty nebudou veřejně známé).

2.3.2 Obměna dílů

Určitou nevýhodu může představovat stálost a neměnnost dílů – mohou být platné velice dlouhou dobu, třeba několik let i více. To útočníkovi poskytuje čas postupně získat jednotlivé díly. Obranou je pro stejné tajemství pravidelně generovat nové díly a starší tak pravidelně nahrazovat novými. To ale mj. vyžaduje spolupráci s distributorem. Pokud je schéma homomorfní, dá se této vlastnosti využít. V případě Shamirova algoritmu je tajemství sdíleno jako obvykle, ale v pravidelných intervalech je náhodně zvolen polynom stupně $k-1$, jehož absolutní člen, tj. tajemství, je rovno nule. Toto tajemství je rozděleno na díly dle Shamirova algoritmu a posláno účastníkům. Všichni účastníci si příchozí díly přičtou k původnímu. Hodnota výsledného tajemství se nezměnila, ale hodnoty dílů ano, a proto útočník nemůže využít dříve získané díly. Tento systém se nazývá *proaktivní sdílení tajemství*. [9]

Kapitola 3

Ověřitelné sdílení tajemství

V této kapitole jsou popsána vybraná ověřitelná schémata pro sdílení tajemství, která řeší nedostatky popsané v předešlé kapitole a umožňují odhalit nečestného distributora nebo účastníky. Nejprve jsou popsána závazková schémata, která bývají základním blokem ověřitelného sdílení tajemství.

3.1 Závazková schémata

Závazková schémata (*commitment schemes*) jsou kryptografické algoritmy, které umožňují zavázat se k určité hodnotě. *Zavazující (commiter)* se zaváže k hodnotě, kterou zná jen on, a tuto hodnotu nemůže změnit. Výstupem schématu je *závazek (commitment)* k tajné hodnotě, který je zveřejněn nebo poslán druhé straně. Zavazující může později tajnou hodnotu odhalit, a z jeho závazku lze ověřit, že zveřejněná hodnota je skutečně ta, ke které se původně zavázal.

Pokud ze zveřejněného závazku nelze zjistit nic o tajné hodnotě, ke které se zavazující zavázal, schéma se nazývá *dokonale skrývající (perfectly hiding)*. Pokud není možné, aby zavazující mohl změnit tajnou hodnotu tak, aby již vydaný závazek byl platný i pro novou tajnou hodnotu, schéma je *dokonale svazující (perfectly binding)*. Není možné, aby schéma bylo zároveň dokonale skrývající i dokonale svazující; u dokonale svazujícího schématu stejný závazek může představovat jen jedno tajemství, aby zavazující nemohl svůj závazek změnit. U dokonale skrývajícího schématu stejný závazek odpovídá mnoha tajemstvím, aby nebylo možné určit hodnotu, ke které byl závazek vytvořen. Tyto požadavky jsou tak v rozporu. Schémata také mohou být *statisticky skrývající*, resp. *statisticky zavazující*. U těchto schémat účastník s neomezenými výpočetními prostředky sice může teoreticky podvádět, ale pouze s zanedbatelnou pravděpodobností. Posledním typem jsou schémata, kde je bezpečnost založena na výpočetní obtížnosti některých problémů. V tomto případě účastník s neomezenou výpočetní silou podvádět může. [10]

Závazková schémata mohou být založena na různých principech, například na problému diskrétního logaritmu [6], hashovacích funkcích [10] nebo generátorech pseudonáhodných čísel [11].

Možnosti použití těchto schémat jsou široké; ověřitelná schémata pro sdílení tajemství (popsaná v další sekci), důkazy s nulovou znalostí (*zero-knowledge proofs*) [12] nebo tzv. *coin-flipping* protokoly [13].

3.1.1 Naorovo závazkové schéma

Toto schéma je založeno na generátoru pseudonáhodných čísel a v základní variantě se zavazující zaváže k jednomu bitu.

Definice 3.1 Generátor pseudonáhodných čísel je pravděpodobnostní polynomiální algoritmus G, pro který existuje funkce $l : \mathbb{N} \mapsto \mathbb{N}$, $l(m) > m$ pro všechna $m \in \mathbb{N}$ a $|G(s)| = l(|s|)$ pro všechny bitové řetězce s libovolné délky a jehož výstup musí být v polynomiálním čase nerozlišitelný od skutečně náhodného. [14][11]

Nechť G je generátor pseudonáhodných čísel a n je bezpečnostní parametr takový, že žádný pravděpodobnostní polynomiální algoritmus nedokáže rozlišit výstupy generátoru G s počáteční hodnotou délky n bitů od skutečně náhodného výstupu stejné délky. Nechť $B_i(s)$ je i-tý bit sekvence bitů vytvořené generátorem G s počáteční hodnotou s. Tajný bit zavazujícího je značen b. Naorovo závazkové schéma se skládá z následujících kroků, přičemž $\oplus$ značí bitovou operaci XOR:

1. Protistrana náhodně zvolí vektor $\mathbf{R} = (r_1, r_2, \ldots, r_{3n})$, $r_i \in \{0, 1\}$ pro $i \in \{1, 2, \ldots, 3n\}$
2. Protistrana pošle $\mathbf{R}$ zavazujícímu.
3. Zavazující náhodně zvolí počáteční hodnotu s, $s \in \{0, 1\}^n$ pro generátor pseudonáhodných čísel G
4. Zavazující vypočte vektor $\mathbf{D} = (d_1, d_2, \ldots, d_{3n})$, $d_i \in \{0, 1\}$, kde hodnoty d_i pro $i \in \{1, 2, \ldots, 3n\}$ jsou následující

$$d_i = \begin{cases} B_i(s) & \text{pokud } r_i = 0 \\ B_i(s) \oplus b & \text{pokud } r_i = 1 \end{cases} \tag{3.1}$$

5. Zavazující pošle $\mathbf{D}$ protistraně.

Pokud chce zavazující svůj bit zveřejnit, pošle protistraně svůj bit b a počáteční hodnotu s. Protistrana ověří, že pro všechna i, $i \in \{1, 2, \ldots, 3n\}$, pokud $r_i = 0$, pak $d_i = B_i(s)$, jinak $d_i = B_i(s) \oplus b$. [11]

Bezpečnost

Pokud zavazující najde jinou počáteční hodnotu s' takovou, že výstupy $G_{3n}(s)$ a $G_{3n}(s')$ se shodují na všech pozicích i, $i \in \{1, 2, \ldots, 3n\}$, kde $r_i = 0$ a zároveň se liší na všech pozicích, kde $r_i = 1$, pak může podvrhnout bit b. Toto schéma tedy není dokonale svazující. Pravděpodobnost, že zavazující pro vektor $\mathbf{R}$ najde takovou dvojici s a s', je nejvýše 2^{-n}.

Důkaz. Pokud zavazující našel k s takové s', pak taková dvojice dokáže obelstít pouze jeden vektor $\mathbf{R}$, neboť pro všechna $i \in \{1, 2, \ldots, 3n\}$ platí, že $r_i = B_i(s) \oplus B_i(s')$. Dvojic různých počátečních hodnot je $2^n \times 2^n = 2^{2n}$ a počet možných vektorů $\mathbf{R}$ je 2^{3n}. Z toho vyplývá, že zavazující k vektoru $\mathbf{R}$ najde takovou dvojici s a s' s pravděpodobností nejvýše $\frac{2^{2n}}{2^{3n}} = 2^{-n}$. [11]

Protistrana může odhadnout bit b s pravděpodobností vyšší než $\frac{1}{2} + p(n)$ tehdy, pokud dokáže v polynomiálním čase rozlišit výstup generátoru pseudonáhodných čísel od zcela náhodného výstupu.

Důkaz. Důkaz bude proveden sporem. Protistrana dokáže v polynomiálním čase odhadnout bit b s pravděpodobností větší než $\frac{1}{2} + p(n)$. Zavazující se zaváže k náhodnému bitu b a místo výstupu z generátoru G použije hodnotu x. Protistrana následně odhadne b. Je-li odhad správný, x bude považováno za pseudonáhodné; je-li odhad špatný, bude x považováno za skutečně náhodné. Protistrana tedy dokáže rozlišit zcela náhodný výstup od výstupu generátoru pseudonáhodných čísel, což je spor. [11]

Pokud se na generátory pseudonáhodných čísel bude hledět jako na tabulku počátečních hodnot délky n bitů a odpovídajících vygenerovaných sekvencí, v klasickém pojetí je na nalezení počáteční hodnoty k dané vygenerované sekvenci potřeba $\mathcal{O}(2^n)$ pokusů. S kvantovým počítačem je možné díky Groverovu algoritmu najít vzor pouze s $\mathcal{O}(\sqrt{2^n}) = \mathcal{O}(2^{\frac{n}{2}})$ pokusy [15]. Najít vzor je tak podobně obtížné jako pro generátor pseudonáhodných čísel s počáteční hodnotou délky $\frac{n}{2}$ bez použití kvantového počítače.

Vlastnosti

V původním schématu se zavazující zaváže pouze k jedinému bitu; toto schéma se dá samozřejmě zopakovat pro více bitů, ale je pak značně neefektivní. Závazek pro jeden bit totiž zabírá $\mathcal{O}(n)$ paměti, pro m bitů je to již $\mathcal{O}(mn)$. Pro sdílení více bitů je vhodnější použít upravené schéma – v původním článku je navrženo schéma využívající jednu pseudonáhodnou sekvenci k zavázání se k více bitům najednou, spolu se samoopravnými kódy, které zajišťují, aby zavazující nemohl některý ze svých bitů změnit. [11]

3.1.2 Pedersenovo závazkové schéma

Toto schéma patří mezi nejznámější závazková schémata. V původním článku z roku 1991 bylo využito pro konstrukci ověřitelného schématu sdílení tajemství, které je popsáno v sekci 3.2.1. Schéma je založeno na problému diskrétního logaritmu. [6]

Nechť p a q jsou prvočísla taková, aby q dělilo $p-1$. Podgrupa grupy $\mathbb{Z}_p^*$ řádu q je G_q a g a h jsou takové generátory grupy G_q, aby $\log_g h$ nebyl známý. Zavazující se zaváže k hodnotě $s \in \mathbb{Z}_q$ pomocí náhodného $t \in \mathbb{Z}_q$ a výpočtem a zveřejněním následujícím hodnoty:

$$E(s,t) = g^s h^t \pmod{p} \tag{3.2}$$

Pokud chce zavazující svoji tajnou hodnotu odhalit, rozešle ostatním stranám s a t, díky kterým ostatní mohou ověřit, že pro tyto hodnoty rovnost $E(s,t) = g^s h^t \pmod{p}$ platí. [6]

Bezpečnost

Hodnota $E(s,t)$ neodhaluje nic o s (je dokonale skrývající) a zavazující nemůže podvrhnout s bez znalosti $log_g h \pmod{q}$ (schéma tedy není dokonale svazující – závisí na obtížnosti výpočtu diskrétního logaritmu). [6]

Důkaz. Důkaz nemožnosti podvrhnout s bez znalosti logaritmu $log_g h$ bude proveden sporem. Zavazující našel $s', t' \in \mathbb{Z}_q$ takové, aby $s' \neq s \pmod{q}$ a zároveň $E(s',t') = E(s,t)$. V tom případě i $t' \neq t \pmod{q}$ a platí následující rovnost:

$$log_g h = \frac{s - s'}{t' - t} \pmod{q}. \tag{3.3}$$

Zavazující tedy musí znát i $log_g h$, což je spor.

Důkaz. Důkaz, že je toto schéma dokonale skrývající: Pokud je $t \in \mathbb{Z}_q$ zvoleno náhodně, pro každé $s \in \mathbb{Z}_q$ bude $g^s h^t = E(s,t)$ rozděleno rovnoměrně náhodně v G_q. Vyjádřením $E(s,t)$ pomocí generátoru g vznikne následující vztah:

$$\begin{aligned} g^e = g^s h^t &= g^s g^{x^t} \\ e = s + xt &\pmod{q} \\ x^{-1}(e - s) = t &\pmod{q} \end{aligned} \tag{3.4}$$

Hodnota x nebude nulová, protože g^x je generátor, takže x^{-1} existuje. Z toho plyne, že pro každé možné tajemství $s \in \mathbb{Z}_q$ existuje $t \in \mathbb{Z}_q$ takové, aby vypočtený závazek zůstával neměnný. [6]

Protože problém diskrétního logaritmu je řešitelný na kvantovém počítači v polynomiálním čase díky Shorovu algoritmu [16], s nástupem kvantových počítačů nebude toto závazkové schéma nadále bezpečné. Druhá strana sice

stále nebude moci ze závazku zjistit tajnou hodnotu, ale zavazující by mohl spočítat $log_g h$ a tajnou hodnotu podvrhnout.

Ze závazku k tajemství s lze jednoduše spočítat závazek k $s+1$. Stačí závazek vynásobit generátorem g, neboť $g^s h^t g = g^{s+1} h^t = E(s+1,t)$. Toho se dá zneužít, pokud útočník naruší komunikaci mezi zavazujícím a ostatními, závazek zavazujícího odchytí a pošle místo něj ostatním podvrženou hodnotu. Při odhalení tajné hodnoty opět naruší komunikaci, ověří správnost závazku a ostatním pošle $s+1$, t. Ostatní sice úspěšně ověří platnost podvrhu, ale nedozví se, že je tato tajná hodnota jiná, než byla tajná hodnota zavazujícího.

Vlastnosti

Byla-li tajná hodnota s n-bitová, velikost závazku je $\mathcal{O}(n)$ bitů. Schéma je $(\cdot,+)$ homomorfní – vynásobením dvou různých závazků $E(s_1,t_1)$ a $E(s_2,t_2)$ (při zachování g, h, p a q) vznikne závazek $E(s_1+s_2, t_1+t_2)$, kde součty s_1+s_2 a t_1+t_2 jsou hodnoty modulo q. Obdobně vynásobením $E(s_1,t_1)$ inverzní hodnotou $E(s_2,t_2)$ vznikne nový závazek $E(s_1-s_2, t_1-t_2)$.

Důkaz. Úpravy výrazu jsou přímočaré.

$$E(s_1,t_2)\cdot E(s_2,t_2) = g^{s_1}h^{t_1}g^{s_2}h^{t_2} = g^{s_1+s_2}h^{t_1+t_2} = E(s_1+s_2,t_1+t_2) \pmod p \tag{3.5}$$

Podobně lze k rovnosti dospět v případě rozdílu zavazovaných hodnot

$$\begin{aligned} E(s_1,t_2)\cdot(E(s_2,t_2))^{-1} &= g^{s_1}h^{t_1}(g^{s_2})^{-1}(h^{t_2})^{-1} = g^{s_1-s_2}h^{t_1-t_2} \\ &= E(s_1-s_2,t_1-t_2) \pmod p \end{aligned} \tag{3.6}$$

Tajnou hodnotu s lze také inkrementovat nebo dekrementovat – vynásobením $E(s,t)$ s g, resp. s g^{-1}.

3.1.3 Schéma založené na hashovací funkci

Další možností, jak vytvořit závazkové schéma, je využít vlastností hashovacích funkcí. Schéma, které je rozebráno v této části, popsali Damgård, Pedersen a Pfitzmannová [10], a obdobné schéma později Halevi a Micali [17].

Definice 3.2 Množina H množin hashovacích funkcí $\{H_m\}$, $m \in \mathbb{N}$, je odolná vůči kolizím, pokud splňuje následující vlastnosti:

1. Každá funkce $h \in H_m$ je funkce $\{0,1\}^m \mapsto \{0,1\}^{l(m)}$, kde $l(m)$ je funkce $l(m) : \mathbb{N} \mapsto \mathbb{N}$ a pro všechna m platí $l(m) < m$.

2. Existuje algoritmus, který spočítá výsledek funkce $h \in H_m$ pro vstup $x \in \{0,1\}^m$ v polynomiálním čase.

3. Pro všechna $c > 0$ a pro všechny pravděpodobnostní polynomiální algoritmy A_H je pravděpodobnost, že A_H nalezne dva různé prvky $x, y \in \{0,1\}^m$ takové, že $h(x) = h(y)$ nejvýše m^{-c}, za předpokladu, že h i A_H jsou náhodně vybrané. [10]

Hashovací funkce tedy přijímá na vstup nějakou hodnotu, kterou lze spočíst v polynomiálním čase a jejím výstupem je hodnota fixní délky. Protože množina vstupních hodnot může být (a bývá) značně větší než výstupní, nutně bude docházet ke kolizím – pro více různých vstupních hodnot bude výsledkem stejná funkční hodnota. Odolnost vůči kolizím neznamená, že by kolize neexistovaly; tato vlastnost zaručuje, že nalézt byť jedinou takovou kolizi není v polynomiálním čase proveditelné.

Definice 3.3 *Univerzální rodina hashovacích funkcí* F je množina hashovacích funkcí f z množiny A do množiny B, pokud pro všechny dvojice různých prvků $x, y \in A$ platí, že pravděpodobnost, že $f(x) = f(y)$ pro náhodně vybranou funkci $f \in F$ je nejvýše $\frac{1}{|B|}$. [18]

Samotný postup zavázání se k tajné hodnotě se skládá z následujících kroků:

1. Druhá strana náhodně vybere hashovací funkci $h \in \{H\}_k$, $h : \{0,1\}^+ \mapsto \{0,1\}^{k+1}$ a pošle ji zavazujícímu.

2. Zavazující náhodně vybere hodnotu y délky $3(k+1)$ a univerzální hashovací funkci f, $f : \{0,1\}^{3(k+1)} \mapsto \{0,1\}^{k+1}$. Poté spočte závazek c:

$$c = h(f||h(y)||h(x) \oplus f(y)) \tag{3.7}$$

   a pošle jej druhé straně. Znak $\oplus$ značí bitovou operaci XOR a $||$ zřetězení bitů.

3. Pokud chce zavazující hodnotu x odhalit, pošle x, f a y protistraně. Druhá strana ověří platnost rovnice (3.7). [10]

Bezpečnost

Zavazující vlastnost tohoto schématu se odvíjí od odolnosti funkce h vůči kolizím, toto závazkové schéma tedy není dokonale zavazující. Zavazující strana nemůže hodnotu x podvrhnout, pokud nedokáže v polynomiálním čase najít kolizi.

Důkaz. Důkaz bude proveden sporem. Zavazující našel v polynomiálním čase hodnoty $x' \neq x$, y' a funkci f' takovou, že závazek s těmito hodnotami je rovný původnímu závazku c, tj. $c = h(f||h(y)||h(x) \oplus f(y)) = h(f'||h(y')||h(x') \oplus f'(y))$. Z bezkoliznosti hashovací funkce h plyne, že $f||h(y)||h(x) \oplus f(y) = f'||h(y')||h(x') \oplus f'(y')$. Dalším rozkladem vzniknou rovnosti $h(y) = h(y')$

a $h(x) = h(x')$. Z nich je patrné, že $y = y'$ a $x = x'$. Což je spor, protože pro podvržení je nutné, aby hodnoty x a x' byly různé, což bez nalezení kolize v polynomiálním čase není možné.

Za předpokladu, že hashovací funkce h z H jsou bezkolizní, je toto schéma statisticky skrývající, tj. neomezený útočník může podvádět jen se zanedbatelnou pravděpodobností. Důkaz je vynechán, uveden je v [10].

První krok tohoto schématu (náhodný výběr funkce h) může být proveden pouze jednou a použit i pro následující závazky, aniž by se snížila bezpečnost. [10] Náročnost výpočtu a ověření závazku se odvíjí od náročnosti hashovacích funkcí h a f. Výhodou je, že pro různě dlouhé hodnoty x a konstantní k zůstává stejná velikost závazku, a to $k + 1$ bitů.

Stejně jako v případě generátorů pseudonáhodných čísel, pokud se na hashovací funkci pohlíží jako na tabulku se vstupy a výstupy v podobě hashe, je k nalezení vzoru k danému hashi díky Groverovu algoritmu [15] třeba jen $\mathcal{O}(2^{\frac{n}{2}})$ pokusů.

3.2 Ověřitelné sdílení tajemství

Ověřitelné sdílení tajemství je takové schéma pro sdílení tajemství, které účastníkům umožňuje bezpečně rozdělit tajemství mezi účastníky a navíc ověřit, že je jejich díl poskytnutý distributorem platný, a také odhalit chybné nebo podvržené díly ve fázi rekonstrukce. Tento pojem zavedli v roce 1985 Chor, Goldwasserová, Micali a Awerbuch [19].

Definice 3.4 Množina n dílů tajemství se nazývá *k–konzistentní*, pokud každá její libovolná podmnožina k dílů definuje stejné tajemství. [5]

Základním požadavkem takových schémat je možnost účastníků ověřit, že díl patří k danému tajemství – ale tak, aby nebylo nutné znát příslušné sdílené tajemství. Díly by také měly jít ověřit samostatně, bez dílů ostatních účastníků. Navíc díly ostatních účastníků nemusí být vypovídající, pokud se ve fázi rekonstrukce podaří zrekonstruovat tajemství, nutně to neznamená, že jsou díly k–konzistentní; útočník mohl svůj díl vhodně podvrhnout. Ověřitelné sdílení tajemství tyto problémy řeší.

Schémata se dělí podle komunikace nutné k ověření dílu na *interaktivní* a *neinteraktivní*. U interaktivních schémat je k ověření nutná komunikace s jinými účastníky (např. se zavazujícím), zatímco u neinteraktivních schémat může účastník ověřit díl pouze se svým dílem a veřejnými informacemi, bez nutnosti spolupráce ostatních.

Ověřitelnost se dá rozšířit nejenom na vlastní díl, ale také na díly ostatních (samozřejmě bez toho, aby tyto díly zbylí účastníci znali). Což znamená, že ověřit platnost dílu účastníka může i někdo jiný než jiný účastník. Tato schémata, kde ověřit díly může kdokoliv, se nazývají *veřejně ověřitelná schémata pro sdílení tajemství*. [20]

3.2.1 Pedersenovo ověřitelné schéma

Pedersenovo ověřitelné schéma využívá Shamirova polynomiálního sdílení tajemství a kombinuje ho s Pedersenovým závazkovým schématem, popsaném v sekci 3.1.2. Toto schéma je prahové (k, n) schéma a jeho ověřovací fáze je neinteraktivní. [6]

Nechť g, h jsou generátory $\mathbb{Z}_q$. Postup tohoto schématu je následující:

1. Distributor vybere tajemství $s \in \mathbb{Z}_q$ a náhodně zvolí $t \in \mathbb{Z}_q$. K hodnotě s se zaváže pomocí t dle rovnice (3.2): $E_0 = E(s, t)$ a E_0 zveřejní.

2. Distributor zvolí polynom F stupně $k-1$ tvaru $F(x) = s + F_1 x + \ldots + F_{k-1} x^{k-1}$, kde F_i jsou náhodné koeficienty z $\mathbb{Z}_q$ a s je tajemství. Poté spočte $s_i = F(i)$ pro všechna $i \in \{1, \ldots, n\}$.

3. Poté vytvoří polynom G stupně $k-1$ tvaru tvaru $G(x) = t + G_1 x + \ldots + G_{k-1} x^{k-1}$ s náhodnými koeficienty $G_1 \ldots G_{k-1}$. Spočte $t_i = G(i)$ pro všechna $i \in \{1, \ldots, n\}$.

4. Následně použije G_i pro zavázání se ke koeficientům F_i – spočte a zveřejní $E_i = E(F_i, G_i), i \in \{1, \ldots, k-1\}$.

5. Distributor účastníkovi i tajně pošle díl (s_i, t_i), $i \in \{1, \ldots, n\}$.

Fáze rekonstrukce je obdobná, jako v případě Shamirova schématu pro sdílení tajemství, tajnou hodnotu lze spočítat z alespoň k různých tajných dílů pomocí Lagreangovy interpolace (popsané v sekci 2.2).

Každý účastník může nyní ověřit, zda dostal validní díl (či zda jsou validní díly ostatních ve fázi rekonstrukce), a to verifikací následující rovnosti:

$$E(s_i, t_i) = \prod_{j=0}^{k-1} E_j^{i^j} \tag{3.8}$$

Důkaz. Důkaz, že rovnost výše vede k ověření platnosti dílu:

$$\begin{aligned} \prod_{j=0}^{k-1} E_j^{i^j} &= g^{F_j i^j} h^{G_j i^j} = g^{F_0 + F_1 i + \ldots + F_{k-1} i^{k-1}} h^{G_0 + G_1 i + \ldots + G_{k-1} i^{k-1}} \\ &= g^{F(i)} h^{G(i)} = E(s_i, t_i) \end{aligned} \tag{3.9}$$

Tímto způsobem lze ve fázi rekonstrukce ověřit, zda jsou obdržené díly k-konzistentní. Obdobně jako v případě $E(s, t)$, znalost $E(s_i, t_i)$ neodhaluje nic o s_i a t_i.

Bezpečnost

Bezpečnost samotného sdílení a rekonstrukce tajemství je založena na Shamirově schématu, tj. pro (k, n) schéma ani $k-1$ účastníků nezjistí nic o sdíleném tajemství. Bezpečnost celého schématu vychází z bezpečnosti částí, ze kterých se skládá – kvůli povaze Pedersenova závazkového schématu, které staví na diskrétním logaritmu, není odolné vůči útokům pomocí kvantového počítače Shorovým algoritmem [16].

Ověřitelnost tohoto schématu není jenom jednostranná. Účastníci nejenom že mohou ověřit, zda jim distributor neposlal neplatný díl, ale ve fázi rekonstrukce můžou navíc ověřit, že díly poskytnuté ostatními nejsou podvržené. Tím se zamezí jednoduchému útoku, kdy při rekonstrukci tajemství pošle jeden účastník falešný díl a jako jediný získá správné tajemství.

Vlastnosti

Každý díl tajemství je jen dvojnásobný oproti velikosti samotného tajemství, protože se navíc sdílí i díl ke kontrolní hodnotě t. Oproti Shamirovu schématu se ale zvýšil i počet veřejných parametrů kvůli $E_0, E_1, \ldots, E_{k-1}$.

Toto schéma je aditivně homomorfní, stejně jako Pedersenovo závazkové schéma a Shamirovo schéma pro sdílení tajemství, z kterých vychází. Jsou-li mezi účastníky sdílena dvě tajemství s a s' (k, n) schématem, pak každý účastník i, $i \in \{1, \ldots, n\}$ má díly tajemství (s_i, t_i) a (s'_i, t'_i). Veřejné parametry jsou $(E_0, \ldots, E_{n-1})$, resp. $(E'_0, \ldots, E'_{n-1})$. Nové tajemství d, $s'' = s + s' \pmod q$ lze sdílet následovně: účastník i nový díl (s''_i, t''_i) získá součtem obou předchozích dílů

$$\begin{aligned} s''_i &= s_i + s'_i \pmod q \\ t''_i &= t_i + t'_i \pmod q \end{aligned} \tag{3.10}$$

Veřejné parametry $(E''_0, \ldots, E''_{k-1})$ vzniknou ze součinu původních parametrů

$$E''_i = E_i E'_i, i \in \{0, 1, \ldots, k-1\} \tag{3.11}$$

Tyto parametry jsou v souladu s novým tajemstvím, neboť dle (3.5) $E_i E'_i = E(F_i, G_i)E(F'_i, G'_i) = E(F_i + F'_i, G_i + G'_i)$.

Tímto způsobem lze sdílet anonymní tajemství, kdy každý účastník sdílí svoji tajnou hodnotu s ostatními, a všechny příchozí díly spolu se svým zkombinuje dle rovnic výše; všichni pak budou sdílet tajnou hodnotu, kterou ale nikdo nezná.

3.2.2 Schoenmakersovo veřejně ověřitelné schéma

Schoenmakersovo schéma je oproti předchozímu zmíněnému veřejně ověřitelné schéma pro sdílení tajemství. Obdobně jako předchozí popsané schéma

vychází ze Shamirova sdílení tajemství. Navíc využívá asymetrické šifrování a Chaumův-Pedersenův protokol. [21]

Definice 3.5 Chaumův-Pedersenův protokol, $DLEQ(g_1, h_1, g_2, h_2)$, je protokol mezi dokazujícím a ověřujícím určený k prokázání znalosti diskrétního logaritmu $h_1 = g_1^\alpha$ a $h_2 = g_2^\alpha$ bez nutnosti odhalit jeho hodnotu druhé straně – řadí se tak mezi tzv. *zero-knowledge* protokoly. Protokol se skládá z následujících částí: [22][21]

1. Dokazující pošle ověřujícímu $a_1 = g_1^w$ a $a_2 = g_2^w$, přičemž w je náhodné z $\mathbb{Z}_q$.
2. Ověřující pošle dokazujícímu náhodně vybranou výzvu $c \in \mathbb{Z}_q$.
3. Dokazující pošle protistraně $r = w - \alpha c \pmod q$
4. Ověřující ověří platnost rovnic $a_1 = g_1^r h_1^c$ a $a_2 = g_2^r h_2^c$.

Tento protokol dokazuje znalost α, neboť $a_1 = g_1^w = g_1^{r+\alpha c} = g_1^r g_1 \alpha c = g_1^w h_1^c$. Obdobně $a_2 = g_2^w = g_2^{r+\alpha c} = g_2^w h_2^c$. V protokolu je nutná komunikace mezi dokazujícím a ověřovatelem.

Nechť G_q je grupa řádu q a nechť g, G jsou její generátory takové, aby logaritmus $\log_g H$ nebyl známý. Jako s je značena náhodně vybraná hodnota z $\mathbb{Z}_q$, a $S = G^s$. Pak postup (k, n)–prahového schématu se skládá z následujících kroků:

1. Všichni účastníci P_i, $i \in \{0, 1, \ldots, n\}$ vygenerují náhodné soukromé klíče $x_i \in \mathbb{Z}_q^*$ a zveřejní příslušné veřejné klíče $y_i = G^{x_i}$.
2. Distributor vybere polynom $Q(x)$ nad $\mathbb{Z}_p$ stupně nejvýše $k - 1$:

$$Q(x) = a_0 + a_1 x + \ldots + a_{k-1} x^{k-1}, \tag{3.12}$$

   kde všechny koeficienty a_i, $i \in \{1, 2, \ldots, k-1\}$ jsou náhodně vybrané z $\mathbb{Z}_q$ a tajný exponent s je roven absolutnímu členu polynomu a_0.
3. Distributor zveřejní jednotlivé díly tajemství zašifrované veřejnými klíči:

$$Y_i = y_i^{Q(i)}, i \in \{1, 2, \ldots, n\} \tag{3.13}$$

   Spolu s díly zveřejní i závazky k polynomu $Q(x)$ v podobě hodnot $C_j = g^{\alpha_j}$, $j \in \{0, 1, \ldots, k\}$. Také zveřejní důkazy, že díly jsou zašifrované správnými veřejnými klíči pomocí neinteraktivního Chaumova-Pedersenova protokolu $DLEQ(g, X_i, y_i, Y_i)$ pro $i \in \{1, 2, \ldots, n\}$.
4. V dalším kroku může kdokoliv ověřit platnost zveřejněných dílů ověřením $DLEQ(g, X_i, y_i, Y_i)$ pro $i \in \{1, 2, \ldots, n\}$, neboť $X_i = \prod_{j=0}^{k-1} C_j^{ij}$. Všechny potřebné hodnoty – C_j, Y_i, g, y_i – jsou veřejné.

Chaumův-Pederesenův protokol je interaktivní, což nemusí být vždy žádoucí, proto je v tomto schématu upraven na neinteraktivní. Výzva c není náhodná, ale pevně daná – spočte se jako hash hodnot h_1, h_2, a_1 a a_2. Dokazující tedy v prvním kroku pošle druhé straně (nebo zveřejní) kromě hodnot a_1, a_2 rovnou i hodnotu r, neboť si výzvu c vypočte předem. Druhá strana taktéž spočte c a ověří platnost zaslaného a_1 a a_2. Žádná další komunikace mezi stranami tak není nutná. Toto řešení využívá Fiatovu-Shamirovu techniku [23] a je popsáno v [22].

Fáze rekonstrukce tajemství se skládá z následujících kroků:

1. Účastník i dešifruje tajný díl Y_i pomocí inverzní hodnoty svého tajného klíče x_i:
$$S_i = Y_i^{{x_i}^{-1}} = ((G^{x_i})^{Q(i)})^{{x_i}^{-1}} = G^{Q(i)} \tag{3.14}$$

2. Tajný exponent $s = Q(0)$ je možné nalézt Lagrangeovou interpolací. Protože ale dešifrované díly S_i nejsou rovny $Q(i)$, ale $G^{Q(i)}$, lze místo s spočítat tajemství G^s:
$$s = \sum_{i=1}^{k} Q(i)\lambda_i \tag{3.15}$$
$$G^s = G^{\sum_{i=1}^{k} Q(i)\lambda_i} = \prod_{i=1}^{k} (G^{Q(i)})^{\lambda_i} = \prod_{i=1}^{k} \tag{3.16}$$

   kde $\lambda_i = \prod_{\substack{j=0 \\ j\neq i}}^{k} \frac{j}{j-i}$

Při fázi rekonstrukce je každý účastník schopný dešifrovat svůj díl Y_i a získat tak $S_i = Y_i^{(x_i^{-1})}$. K důkazu, že S_i je opravdu jeho díl, lze využít protokol 3.5 $DLEQ(G, y_i, D_i, Y_i)$, a dokázat tak, že zná α, $y_i = G^\alpha$, $Y_i = S_i^\alpha$.

Konzistence dílů při distribuci je tak zaručena pomocí závazků C_i a příslušných důkazů. Při rekonstrukci účastníci dokazují, že poslaný díl je správně dešifrovaný díl z předchozího kroku, čímž je zajištěno, že díly ve fázi rekonstrukce budou k-konzistentní.

V tomto schématu je náhodně vybraný pouze exponent v tajemství, a nelze tak použít předem vybranou hodnotu pro celkové tajemství; tento nedostatek je v původním článku řešen a jsou popsány možnosti, jak schéma upravit i pro předem daná tajemství. [21]

Bezpečnost

Prolomit šifrování dílů by znamenalo umět v polynomiálním čase vyřešit tzv. *Diffieho-Hellmanův problém*, tj. spočítat $g^{\alpha\beta}$ při znalosti g^α a g^β.

Důkaz. Díl účastníka $i \in \{1, 2, \ldots, n\}$ se dá rozepsat následovně: $Y_i = y_i^{Q(i)} = G^{x_i Q(i)}$. Pro nějakou hodnotu $\delta \in G_q$ platí, že $G = g^\delta$, takže se veřejný klíč

dá vyjádřit jako $y_i = G^{x_i} = g^{(\delta x_i)}$, a pak se celý vztah dá přepsat na $Y_i = g^{(\delta x_i)Q(i)}$. Hodnoty $g^{\delta x_i} = y_i$, $Y_i = g^{(\delta x_i)Q(i)}$, $X_i = g^{Q(i)}$ jsou přitom veřejně známé. [21]

Stejně jako v případě Shamirova schématu, $k-1$ dílů nepostačuje ke spočítání tajemství. Tajemství lze s $k-1$ díly zrekonstruovat pouze pokud by účastníci vyřešili Diffieho-Hellmanův problém, neboť $G = g^{\delta}$ pro nějaké $\delta \in G_q$ a tajemství tak lze přepsat jako $G^s = g^{\delta s}$ a hodnoty $g^{\delta} = G$ a $g^s = C_0$ jsou známé.

Při použití Chaumova-Pedersenova algoritmu je důležité neopakovat w pro více běhů protokolu se stejným α, neboť se stejným w lze snadno spočítat dokazované α za pomocí dvou odpovědí r_1, r_2 na dvě různé výzvy c_1, c_2 pomocí vztahu $\frac{r_1 - r_2}{c_2 - c_1} = \frac{(w - \alpha c_1) - (w - \alpha c_2)}{c_2 - c_1} = \alpha$. Chaumův-Pedersenův algoritmus navíc v neinteraktivní verzi používá Fiatovu-Shamirovu techniku, která je bezpečná v modelu náhodného orákula. [21]

Protože šifrování dílů a dokazování znalosti staví na problému diskrétního logaritmu, je tento protokol prolomen Shorovým algoritmem.

Vlastnosti

Každý z n účastníků musí zveřejnit svůj veřejný klíč y_i a distributor musí zveřejnit n zašifrovaných dílů, k hodnot $C_i, i \in \{1, 2, \ldots, k\}$, n důkazů a výzvu (pro neinteraktivní verzi). Při rekonstrukci musí každý účastník kromě dešifrovaného dílu poslat navíc i důkaz o jeho správnosti. Schéma je $(\cdot, +)$ homomorfní ve vztahu k zašifrovaným dílům; vynásobením zašifrovaných dílů dvou různých tajemství vznikne díl reprezentující součet obou tajemství, neboť $X_i^1 X_i^2 = y_i^{Q^1(i)} y_i^{Q^2(i)} = y_i^{Q^1(i)+Q^2(i)}$. Této vlastnosti se kromě anonymního sdílení tajemství dá využít i v elektronických volbách, kdy celkové tajemství obsahuje výsledek tajného hlasování – jedná se tak o formu počítání více stran (popsané v následující kapitole), přesnější popis je uveden v [21].

Kapitola 4

Bezpečné počítání více stran

Ve sdílení tajemství je sdílena neznámá hodnota pomocí dílů, které samostatně o tajemství nic neprozrazují. Může ale nastat situace, kdy je vhodné spočítat tajnou hodnotu pomocí dílů, které mají i jinou roli, než jen sdílet část tajemství, tj. i samostatně představují nějakou hodnotu. V tomto případě může být nežádoucí, aby se ostatní účastníci dozvěděli díly ostatních, neboť mohou představovat jinou tajnou informaci. Vzniká tak potřeba zajistit bezpečnost dílů před ostatními účastníky, ale zároveň je využít k výpočtu celkového tajemství. Tento problém řeší bezpečné počítání více stran, kdy společnou hodnotu účastníci spočítají bez znalosti dílů všech ostatních.

Formálněji, problém počítání více stran (tzv. *multiparty computing*) se zabývá výpočtem funkce $f(x_1, x_2, \ldots, x_n)$ s tajnými vstupy $x_1, x_2, \ldots, x_n$, přičemž každý z n účastníků má svoji tajnou část vstupu x_i, $i \in \{1, \ldots n\}$, kterou nechce odhalit ostatním. Cílem je, aby bylo možné spočítat výslednou hodnotu, ale bez znalosti vstupů ostatních účastníků. Typickým problémem je tzv. problém dvou milionářů – dva milionáři chtějí zjistit, kdo z nich je bohatší, ale samozřejmě nechtějí druhému říci, jak jsou bohatí. Tento problém představil Andrew Yao ve svém článku, kde také navrhl řešení tohoto problému a dal tak základ pro další výzkum v této oblasti. [24]

Jako první se nabízí řešení v podobě důvěryhodné třetí strany, které účastníci tajně pošlou své hodnoty a dozví se až finální výsledek. Ne vždy je toto řešení využitelné, ať už z důvodu neexistence takové důvěryhodné strany nebo z obavy o bezpečnost dat – pokud je třetí strana kompromitována, všechny tajné hodnoty můžou být vyzrazeny. Bezpečné společné výpočty mají široké možnosti využití – s rostoucími požadavky na bezpečnost při práci s citlivými údaji tato technika nabízí nespornou výhodu, že se v průběhu výpočtu s údaji nepracuje přímo, a je tak vhodná pro elektronické hlasování, studie s využitím zdrojů z více databází [25], různé průzkumy nebo reklamní účely; obdobné využití je např. v elektronických aukcích, kde je třeba zachovat tajnost nabídek [26] (což byl případ pravděpodobně prvního významnějšího využití počítání více stran v praxi [27]).

4.1 Schéma založené na *garbled circuit*

Základ pro tento typ schémat položil v roce 1982 Yao [24]. Z jeho myšlenky se později vyvinula technika skrytého obvodu, tzv. *garbled circuit*, která ale samotná k bezpečnému výpočtu více stran nestačí. Proto se kombinuje s protokolem *oblivious transfer*.

Definice 4.1 Aritmetický obvod je acyklický graf nad konečným tělesem $\mathbb{F}$. Jeho vrcholy jsou hradla představující operaci sčítání, násobení a násobení konstantou. Hrany jsou spoje mezi výstupem z jednoho hradla a vstupy jiného; protože graf je acyklický, lze vytvořit posloupnost hradel takovou, že výstup z libovolného hradla je spojen se vstupem pouze po něm následujícího hradla. [28]

Logický obvod je acyklický graf, jehož vrcholy jsou hradla představující standardní bitové operace jako AND, XOR a NOT a hrany jsou opět spoje mezi výstupem jednoho a vstupem dalšího hradla. [14]

Logický obvod je vlastně aritmetický obvod nad $GF(2)$. Pro $x, y \in GF(2)$ operaci $AND(x, y)$ lze vyjádřit jako $x \cdot y$, operaci $XOR(x, y)$ jako $x + y$ a operaci $NOT(x)$ jako $1 + x$.

Na logický obvod lze převést libovolnou funkci, pokud je omezena velikost jejích vstupů [29], neboť v tomto případě je vždy možné vytvořit pravdivostní tabulku funkce a při konstrukci obvodu vycházet z ní.

4.1.1 Garbled circuit

Garbled circuit je logický obvod, jehož vnitřní hodnoty jsou skryty tak, že nelze nic usuzovat o vstupních nebo výstupních hodnotách jeho hradel. Cílem je bezpečně spočítat funkci, aniž by bylo nutné znát všechny její vstupy. Následující popis shrnuje základní princip fungování takového obvodu a vychází z [29], [30] a [31]:

Nechť C je logický obvod s d logickými hradly, přičemž každé hradlo má dva jednobitové vstupy w_1, w_2 a jeden jednobitový výstup w_3. Hradlo je vyjádřeno pomocí pravdivostní tabulky, která obsahuje vstupy a výstupy příslušící danému typu hradla. Jednotlivé bity zde ale nejsou reprezentovány pomocí hodnot 0 nebo 1, ale pro každý vstup, resp. výstup, jsou vygenerovány náhodné k-bitové řetězce $k^0_{w_i}$, $k^1_{w_i}$ – značky, reprezentující bity hodnoty 0, resp. 1. Protože jsou tyto značky náhodně zvolené, nevypovídají nic o bitu, který reprezentují. Značky v tabulce nahradí původní vstupní a výstupní bity. Aby bylo možné spojit dvojice vstupů s výstupem, aniž by se vstupní dvojice odhalily, a aby nebylo možné ze značky výstupu odvodit jeho hodnotu (v tabulce reprezentující operaci AND se $k^1_{w_3}$ objeví jen jednou, zatímco $k^0_{w_3}$ třikrát), je výstupní značka k_{w_3} pomocí vstupních značek zašifrována. V tomto popisu je k šifrování použit algoritmus se symetrickým klíčem $Enc(H, x)$, kde H je šifrovací klíč a x je hodnota, která má být zašifrována. Jsou-li dány vstupní

$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
$k^0_{w_1}$	$k^0_{w_2}$	$k^0_{w_3}$
$k^0_{w_1}$	$k^1_{w_2}$	$k^0_{w_3}$
$k^1_{w_1}$	$k^0_{w_2}$	$k^0_{w_3}$
$k^1_{w_1}$	$k^1_{w_2}$	$k^1_{w_3}$

$\rightarrow$

$\boldsymbol{w_3}$
$Enc(k^0_{w_1}, Enc(k^0_{w_2}, k^0_{w_3})$
$Enc(k^0_{w_1}, Enc(k^1_{w_2}, k^0_{w_3})$
$Enc(k^1_{w_1}, Enc(k^0_{w_2}, k^0_{w_3})$
$Enc(k^1_{w_1}, Enc(k^1_{w_2}, k^1_{w_3})$

$\rightarrow$

$\boldsymbol{w_3}$
$Enc(k^1_{w_1}, Enc(k^0_{w_2}, k^0_{w_3})$
$Enc(k^1_{w_1}, Enc(k^1_{w_2}, k^1_{w_3})$
$Enc(k^0_{w_1}, Enc(k^0_{w_2}, k^0_{w_3})$
$Enc(k^0_{w_1}, Enc(k^1_{w_2}, k^0_{w_3})$

Tabulka 4.1: Transformace pravdivostní tabulky hradla AND v protokolu garbled circuit. w_1, w_2 značí vstupy, w_3 výstup. Zleva doprava: zamaskování bitů odpovídajícími náhodnými hodnotami, jejich zašifrování a permutace řádků tabulky.

značky k_{w_1} a k_{w_2}, je výstupní značka k_{w_3} zašifrována dvojitě, a to následujícím způsobem:

$$k_{w_3} \rightarrow Enc(k_{w_1}, Enc(k_{w_2}, k_{w_3})) \tag{4.1}$$

V této podobě má tabulka zásadní nedostatek – je standardně seřazena. Proto jsou řádky v posledním kroku náhodně permutovány. To ale přináší nový problém – při dešifrování je nutné postupně zkoušet všechny řádky, a pokud se dešifrování povede, výsledkem je správná hodnota. Při dešifrování tedy nesmí dojít k úspěšnému dešifrování nesprávné hodnoty. Toho lze docílit např. přidáním dostatečně dlouhého kontrolního řetězce za šifrovanou hodnotu. Jiným řešením tohoto problému je přidání indexačních bitů k hodnotám k_{w_i} [30]. Dvojice hodnot $k^t_{w_1}$ reprezentuje bit t, pak je ke každé hodnotě přidán indexační bit p, $p = t \oplus r$, kde r je náhodný bit. Obdobně je k $k^u_{w_2}$ přidán indexační bit q. Dvojice bitů (p, q) pak unikátně určuje řádek, ve kterém se nachází výsledek pro tuto dvojici vstupů. Protože p ani q nevypovídají o bitech t a u, řádek nemá spojitost se standardním rozložením pravdivostní tabulky. Indexační bit je také připojen před šifrováním k výstupní hodnotě k_{w_3}, aby bylo možné použít indexaci i v následujících hradlech. Tato metoda má tu výhodu, že není nutné, aby bylo možné úspěšně dešifrovat pouze správnou hodnotu, neboť správná hodnota je vždy vybrána napoprvé. Navíc se tímto způsobem sníží výpočetní náročnost, předtím bylo v průměru třeba zkusit dešifrovat dva řádky, nyní pouze jeden.

Jako příklad transformace je použita transformace pravdivostní tabulky logického hradla AND, která je znázorněna v tabulce 4.1.

Výstup z jednoho hradla lze použít jako vstup do jiného; takto je možné hradla řetězit a sestavit z nich obvod. Na konci výpočtu je nutné zpětně přiřadit odpovídající bitovou hodnotu. To je možné realizovat buď nahrazením k_{w_3} bitem, který reprezentuje (a následně zašifrovat jako v příkladu výše) a nebo poskytnout tabulku, která k hodnotě k_{w_3} přiřadí odpovídající bitovou hodnotu.

Bezpečnost garbled circuit je dána použitým šifrovacím algoritmem. V popisu výše je použito dvojité šifrování, dva různé klíče jsou použity v různém pořadí pro zašifrování čtyř hodnot. Při znalosti obou klíčů může vyhodnocující strana původní hodnotu korektně dešifrovat. Je-li znám pouze první klíč, zbývá

prolomit jednoduché šifrování druhým klíčem. Pokud v tomto případě zašifrovaná hodnota představuje stejný bit jako korektně dešifrovaný, má nečestný účastník navíc k dispozici šifrový text a odpovídající otevřený text. Šifrovací systém by měl být vůči útokům tohoto typu odolný. Obdobná situace nastává při znalosti pouze druhého klíče. Znalost ani jednoho z klíčů nepřináší „dvojnásobnou“ bezpečnost, neboť proti dvojitému šifrování je možné vést např. meet-in-the-middle útok, který problém převede na složitost obdobnou jednoduchému šifrování [32]. Proto je vhodné použít šifru, která je proti těmto útokům považována za bezpečnou, mezi které patří např. šifra AES [33].

Bezpečnost garbled circuit vůči útokům pomocí kvantového počítače je taktéž redukována na bezpečnost použitého šifrovacího algoritmu; v případě šifry AES je možné použít Groverův algoritmus [15], který pro nalezení klíče délky n bitů potřebuje řádově jen $2^{\frac{n}{2}}$ pokusů. Šifra AES s 256bitovým klíčem by tak byla redukována na obdobnou bezpečnost jako varianta s 128bitovým klíčem.

Dvojité šifrování může být relativně pomalé, proto se využívají i jiné způsoby zašifrování značek, lze místo ní použít i jednoduché šifrování, jehož klíč je odvozen z obou vstupních značek [29], nebo jiné techniky [30].

Velikost obvodu je dána počtem jeho hradel. Je-li v obvodu d hradel, je pro ně potřeba prostor $\mathcal{O}(d)$, neboť každé hradlo je vyjádřeno tabulkou se čtyřmi záznamy pevné délky. Pro skrytí obvodu je pro každé hradlo potřeba 8 operací $Enc(H, x)$, pro jeho vyhodnocení v průměrném případě 4, nejhůře 8. Je-li využita technika indexačních bitů, budou potřeba vždy pouze 2 operace.

Velikost obvodu lze dále optimalizovat různými technikami – například je možné snížit počet řádků v tabulce hradla [26][34] nebo pozměnit hradlo XOR takovým způsobem, aby k jeho provedení nebyla nutná tabulka, a tedy ani šifrování [35].

4.1.2 Oblivious transfer

Protokol *oblivious transfer* řeší problém, kdy jedna strana – odesílatel – má k dispozici několik různých informací, a jednu nebo více chce poslat druhé straně – příjemci. Zároveň se ale odesílající nesmí dozvědět, jakou informaci si příjemce vybral, a příjemce se naopak nedozví jiné informace než ty, které si vybral. Toto lze samozřejmě realizovat pomocí důvěryhodné třetí strany, která od odesílatele získá všechny informace, a od příjemce obdrží pokyny, které informace mu má poslat, ale takové řešení není vždy praktické nebo možné.

Základní myšlenku pro řešení tohoto problému bez nutnosti třetí strany poprvé popsal Rabin [36], jehož protokol umožňuje poslat příjemci jednu hodnotu s pravděpodobností $\frac{1}{2}$, a odesílatel neví, jestli příjemce hodnotu získal, nebo ne. Později byl popsán obecnější případ, kdy příjemce získá jednu hodnotu ze dvou možných [37]. Protokoly typu *oblivious transfer* se dělí podle toho, kolik hodnot bude posláno příjemci; je-li posláno k hodnot z n možných,

je protokol označován jako k-n protokol. V této části bude popsán 1-2 protokol, který se hodí k přenosu hodnot reprezentujících jeden bit – toho se dá využít u počítání více stran založeném na garbled circuits.

V této části bude popsán 1-2 protokol, který staví na asymetrické kryptografii, a vychází z protokolu popsaného v [37]. Jako šifrovací systém je zde použit systém RSA [38].

Nechť e, d, N značí veřejný klíč, k němu příslušící soukromý klíč a modulo šifrovacího systému RSA. M_0 a M_1 značí dvě tajné hodnoty, které zná odesílatel. Množina všech možných tajných hodnot je $\mathcal{M}_x$. Protokol se skládá z následujících kroků:

1. Odesílatel zvolí e,d, N a dvě náhodné hodnoty m_0, $m_1 \in \mathcal{M}_x$. Čtveřici e, N, m_0 a m_1 pošle příjemci.

2. Příjemce zvolí náhodnou hodnotu $k \in \mathcal{M}_x$ a bit $r \in \{0, 1\}$. Poté spočte $q = (k)^e + m_r \pmod N$ a pošle odesílateli.

3. Odesílatel vypočte $k'_0 = (q - m_0)^d \pmod N$, $k'_1 = (q - m_1)^d \pmod N$. Následně pošle příjemci dvojici $m'_0 = M_0 + k'_0 \pmod N$ a $m'_1 = M_1 + k'_1 \pmod N$.

4. Příjemce je schopen dešifrovat M_r pomocí vztahu $M_r = m'_r - k \pmod N$, neboť $m'_r - k = M_r + ((k)^e + m_r - m_r)^d - k = M_r + k - k = M_r \pmod N$.

Příjemce sice kromě m'_r obdrží i $m'_{r\oplus 1}$, ale tato hodnota mu nedává žádnou informaci o $M_{r\oplus 1}$. Pokud by $M_{r\oplus 1}$ z $m_{r\oplus 1}$ odvodil, musel by znát soukromý klíč odesílatele d.

Důkaz. Ve vztahu $m_{r\oplus 1} = M_r + ((k)^e + m_r - m_{r\oplus 1})^d$ příjemce sice zná k, e, m_r i $m_{r\oplus 1}$, ale pro zjištění M_r potřebuje odečíst celý umocněný výraz, tj. znát i d.

Nevědomost opačným směrem je také zachována. Odesílatel z hodnoty q nezjistí hodnotu bitu r; může sice od q odečíst m_0 nebo m_1 a vzniklou hodnotu dešifrovat, ale protože k je náhodné, nemá jak rozlišit, jestli je dešifrovaná hodnota k, nebo ne.

Protokol je tak odolný vůči nečestným účastníkům, pokud ho dodržují. V případě, že odesílatel protokol nedodrží a jednu hodnotu m'_s, $s \in \{0, 1\}'$ podvrhne, pak příjemce tento podvod odhalí pouze pokud $r = s$. Předpokladem je, že příjemce dokáže určit, zda spočtená hodnota je platnou hodnotou M_r či nikoliv. V případě, že $r \neq s$, nemá jak podvrh zjistit. Námitky (nebo jejich absence) ze strany příjemce odesílateli odhalí bit r a tedy i zprávu M_r, kterou si zvolil.

Tento konkrétní příklad 1-2 protokolu oblivious transfer není odolný útoku kvantového počítače, neboť bezpečnost šifrovacího schématu RSA je postavena na problému faktorizace, který je kvantovým algoritmem řešitelný v po-

lynomiálním čase [16]. Existují ale i jiné varianty realizující oblivious transfer, které proti kvantovému počítači obstojí [39][40].

4.1.3 Počítání více stran pomocí garbled circuit

Jako základní stavební blok pro bezpečné počítání více stran je možné využít garbled circuit. Navíc je ale nutné, aby zůstala zachována tajnost vstupů jednotlivých účastníků. To je zajištěno pomocí protokolu oblivious transfer. Je-li vstup do počítané funkce n-bitový, pro každý z n bitů bude proveden jeden 1-2 protokol. Se znalostí všech vstupních dešifrovacích klíčů je nadále možné obvod vyhodnotit bez další komunikace. Následující popis je založen na variantě uvedené v [37]. Tento protokol je určen pro dva účastníky, jeden tvoří obvod, a druhý ho použije k výpočtu požadované funkce. Před zahájením protokolu se obě strany dohodnou na logickém obvodu reprezentujícím funkci $f(x, y)$, kde x a y je tajný vstup prvního, resp. druhého účastníka. Další kroky jsou následující:

1. První strana z dohodnutého obvodu vytvoří garbled circuit a pošle ho druhé straně, včetně dešifrovacích klíčů pro každý bit svého vstupu.

2. Druhá strana si vybere správné dešifrovací klíče pro každý bit svého tajného vstupu pomocí 1-2 varianty protokolu oblivious transfer.

3. Druhá strana soukromě spočte garbled circuit, dešifruje ho a výsledek pošle první straně.

Bezpečnost tohoto postupu se odvíjí od použitého způsobu skrytí značek v hradlech a typu protokolu oblivious transfer, který byl použit. Komunikace mezi stranami je nutná jen ve fázi poslání garbled circuit a vybrání správných klíčů pomocí oblivious transfer a poté po spočtení výsledku druhou stranou.

4.2 Schéma BGW (Ben-Or, Goldwasserová, Wigderson)

Dalším přístupem k řešení problému bezpečného počítání více stran je využití schémat pro sdílení tajemství. Toto schéma je založené na Shamirovu polynomiálním sdílení tajemství a využívá jeho homomorfismu. Principem je podobné anonymnímu sdílení tajemství – každý rozdělí svůj tajný vstup na díly, které pošle ostatním. Kromě operace sčítání dílů je ale zavedena i operace násobení dílů.

Následující popis obsahuje způsob realizace jednotlivých typů hradel použitých pro schéma BGW.

4.2.1 Aritmetické operace schématu BGW

Pro tuto část nechť $f(x)$ a $f'(x)$ jsou dva polynomy stupně t a jejich funkční hodnota v bodě nula nechť je rovna tajemstvím s, resp. s', tj. $f(0) = s$ a $f'(0) = s'$. Popis hradel vychází z [41].

Sčítání dvou dílů

Protože toto schéma je založeno na Shamirovu schématu pro sdílení tajemství, je $(+,+)$-homomorfní. Součtem výše zmíněných polynomů vznikne nový polynom $g(x) = f(x) + f'(x)$ stupně t. Účastník $P(i)$ tak pouze sečte své dva tajné díly $f(\alpha_i)$ a $f'(\alpha_i)$. Pokud byl alespoň jeden z původních polynomů náhodný, zůstává i $g(x)$ náhodným polynomem a součet dílů nedává žádné informace o součtu nebo původních hodnotách. Tato operace je neinteraktivní, účastníci nepotřebují žádné další informace nebo spolupráci.

Násobení dílu konstantou

Násobení polynomu konstantou c je analogické ke sčítání v tom smyslu, že pro jeho provedení stačí lokální výpočty. Polynom $g(x) = c \cdot f(x)$ má funkční hodnotu v bodě 0, tj. tajemství, rovnu $c \cdot s$. Účastník tak pouze vynásobí svůj díl $f(\alpha_i)$ konstantou c. Pokud byl $f(x)$ náhodný, $g(x)$ je též náhodný polynom a výsledek opět nedává žádné informace o původních hodnotách.

Pomocí operace sčítání a násobení konstantou lze vyjádřit lineární funkce. Pro ně tedy není nutná ve fázi emulace okruhu žádná komunikace mezi účastníky.

Násobení dvou dílů

Operace násobení dvou dílů je komplikovanější než sčítání, již nestačí pouze lokální výpočty. Vynásobením dvou polynomů $f(x)$ a $f'(x)$ sice vznikne požadovaný polynom s funkční hodnotou v bodě nula $g(0) = s \cdot s'$, ale tento polynom má stupeň $2t$. K rekonstrukci tajemství je tak nutné alespoň $2t + 1$ dílů. Pokud je účastníků více než $2t + 1$, je tajemství po prvním násobení rekonstruovatelné, ale každé další násobení zvýší stupeň polynomu, a po určitém počtu násobení nakonec nebude možné tajemství zrekonstruovat, stupeň polynomu bude moc velký. Navíc výsledný polynom není zcela náhodný – protože je výsledkem násobení dvou polynomů, je jistě redukovatelný. Řešením je polynom znovu udělat náhodným a zredukovat jeho stupeň, ale tak, aby funkční hodnota v bodě 0 zůstala zachována.

Proces znáhodnění polynomu proběhne ve 3 krocích:

1. Každý účastník zvolí polynom stupně $2t$ s náhodnými koeficienty tak, aby jeho funkční hodnota v bodě 0 byla nulová.

2. Každý účastník tento polynom rozdělí podle Shamirova $(2t+1, n)$ schématu a každému účastníkovi P_i pošle díl a_i.

3. Všichni účastníci lokálně sečtou nové díly se svým původním dílem.

Tím vznikne polynom s náhodnými koeficienty stupně, ale absolutní člen (tajemství) zůstává neměnný. Tato funkce je tvaru $g(x) = g_0 + g_1 x + \ldots + g_{2t} x^{2t}$. Jejím zkrácením vznikne funkce $h(x) = g_0 + g_1 x + \ldots + g_t x^t$. Původní díly účastníků samozřejmě nejsou platné pro tento zkrácený polynom, ale je možné transformovat hodnoty $g(\alpha_i)$ na $h(\alpha_i)$. Pro transformaci se předpokládá přístup k funkci $\mathbb{F}_{mult}$, která od účastníka P_i přijme vstup v podobě dílu znáhodněného polynomu $g(\alpha_i)$ a jako výstup pošle zredukovaný díl $h(\alpha_i)$ pro polynom $h(x)$. Redukce se tak skládá z jediného kroku:

1. Účastník P_i pošle svůj znáhodněný díl funkci $\mathbb{F}_{mult}$ a obdrží díl $h(\alpha_i)$ polynomu stupně t konzistentní s díly ostatních.

Taková funkce $\mathbb{F}_{mult}$ existuje a pro výpočet $h(\alpha_i)$ jako vstup postačuje pouze hodnota dílu $g(\alpha_i)$.

Důkaz. Nechť B je matice o rozměru $n \times n$ taková, že $b_{i,j} = \alpha_j^i$ pro $i, j \in \{0, 1, \ldots, n-1\}$ a nechť P je matice o rozměru $n \times n$ s prvky $p_{i,i} = 1$ pro $i \in \{1, 2, \ldots, t\}$, jinak $p_{i,j} = 0$. Nechť G a H jsou vektory délky n s prvky $(g_0, g_1, \ldots, g_{2t}, 0, \ldots, 0)$, resp. $(g_0, g_1, \ldots, g_t, 0, \ldots, 0)$. S a R budiž vektory délky n $(g(\alpha_0), g(\alpha_1), \ldots, g(\alpha_{n-1}))$, resp. $(h(\alpha_0), h(\alpha_1), \ldots, h(\alpha_{n-1}))$. Jsou to vektory obsahující jednotlivé body, tj. díly tajemství pro polynom $g(x)$, resp. $h(x)$. Cílem je transformovat vektor S na vektor R, obsahující díly pro zkrácený polynom.

Platí následující rovnosti: $G \cdot B = S$, $G \cdot P = H$ a $H \cdot B = R$. Dosazením lze odvodit, že $S \cdot (B^{-1} \cdot P \cdot B) = R$. Inverze matice B existuje, neboť hodnoty α_i jsou různé a matice tak není singulární.

Matice $(B^{-1} \cdot P \cdot B)$ je tak konstantní matice pro transformaci S na R. Prvky vektoru R r_i je možné spočítat pouze s využitím hodnoty $s_i = g(\alpha_i)$ a matice $(B^{-1} \cdot P \cdot B)$ jako vstupu. Každý účastník P_i tedy potřebuje k výpočtu $r_i = h(\alpha_i)$ znát pouze svoji hodnotu $s_i = g(\alpha_i)$ a nepotřebuje znát hodnoty ostatních.

Funkce $\mathbb{F}_{mult}$ provede tuto transformaci se vstupem $g(\alpha_i)$ a výsledkem je požadovaný díl $h(\alpha_i)$.

Zjednodušení násobení dílů dle Gennara, Rabina a Rabina

V této variantě násobení je provedeno znáhodnění a redukce polynomu v jediném kroku. [42]

1. Každý účastník P_i zvolí polynom $h_i(x)$ stupně t s náhodnými koeficienty tak, aby $h(0) = g(\alpha_i)$. Tento polynom rozdělí dle Shamirova algoritmu

mezi ostatní. Každý účastník tak bude mít díly n různých polynomů $h_i(x)$.

2. Účastník P_i může z těchto dílů spočítat funkční hodnotu v bodě α_i pro polynom s náhodnými koeficienty $H(x)$ stupně t pomocí Lagrangeovy interpolace:
$$H(x) = \sum_{i=1}^{2t+1} \lambda_i h_i(x) \tag{4.2}$$

3. Hodnota $H(\alpha_i)$ je tajný díl polynomu stupně t, který skrývá tajemství $f(0) \cdot f'(0)$.

Hodnoty λ_i jsou pomocné polynomy Lagrangeovy interpolace. Výsledný díl účastníka je tak lineární kombinace různých polynomů v bodě α_i. Protože všechny tyto polynomy měly jako konstantní člen požadovaný součin $f(0) \cdot f'(0)$, výsledkem interpolace je $H(0) = f(0) \cdot f'(0)$. Polynom $H(x)$ má skutečně v bodě 0 hodnotu rovnu požadovanému součinu. Výsledný polynom je navíc náhodný, protože jednotlivé polynomy $h_i(x)$ jsou náhodné.

4.2.2 Počítání více stran pomocí schématu BGW

Účastníci se nejprve domluví na funkci, kterou budou počítat, a její reprezentaci v aritmetickém obvodu pomocí výše zmíněných hradel. Samotný výpočet se skládá z několika fází:

1. Každý z účastníků rozdělí svůj tajný vstup dle Shamirova sdílení tajemství $(t + 1, n)$ schématem a díly pošle ostatním.

2. Všichni účastníci emulují okruh pomocí posloupnosti operací sčítání, násobení konstantou a násobení na tajných dílech. [41]

Fáze rekonstrukce je totožná s rekonstrukcí tajemství u Shamirova schématu:

1. Pro získání výsledné hodnoty si alespoň $t+1$ účastníků pošle mezi sebou svůj spočtený díl a z přijatých dílů pomocí Lagrangeovy interpolace zrekonstruují výsledek funkce. [41]

Bezpečnost

Shamirovo sdílení tajemství neposkytuje žádné ověření správnosti dílů nebo jejich konzistence – pokud útočník ve fázi rekonstrukce pošle neplatný díl, jako jediný bude schopný hodnotu funkce zrekonstruovat. Toto lze vyřešit nahrazením Shamirova schématu ověřitelným schématem, v původním článku [41] je uveden postup, který ověření dílů umožní.

Vlastnosti

Pro redukci polynomu stupně $2t$ je potřeba spolupráce alespoň $2t+1$ účastníků. V (k, n) schématu je použit polynom stupně $k-1$, takže je třeba, aby platilo, že $k \leq \frac{n+1}{2}$.

Oproti schématu založeném na garbled circuit se tohoto schématu může zúčastnit velké množství účastníků naráz. To je příhodné pro realizaci elektronických voleb – např. pro jednoduchou volbu ano/ne stačí mít hodnotu *ano* reprezentovanou číslem 1 a volbu *ne* reprezentovanou číslem 0. Výsledný hlas je pak dán prostým součtem hlasů. To lze realizovat tak, že každý hlasující rozdělí svůj hlas a pošle ho všem ostatním (nebo předem daným sčítacím komisím). Posléze jsou společně sečteny hlasy, aniž by bylo známo, kdo hlasoval jakým způsobem.

Kapitola 5

Kvantové sdílení tajemství

Základ pro kvantovou kryptografii a kvantové počítače byl položen již v 80. letech minulého století [43][44][45] a tato oblast je od té doby předmětem intenzivního výzkumu. Kvantové počítače totiž některé problémy dokáží řešit efektivněji než jejich klasické protějšky. Mezi oblasti, které kvantové počítače dokáží efektivně řešit, patří i výpočet diskrétního logaritmu [16] nebo faktorizace [16], což jsou problémy, jejichž obtížnost výpočtu se nyní v kryptografii hojně využívá (např. šifra RSA [46] nebo Diffieho-Hellmanova výměna klíčů [47]). Hledání praktických náhrad za dnes běžně používané kryptografické protokoly, které by byly odolné i vůči kvantovým počítačům, probíhá již několik let [48]. Mezi kandidáty na post-kvantovou kryptografii z asymetrické kryptografie patří např. systém založený na šifře McEliece [49][50]. U dnes používaných hashovacích funkcí nebo šifry AES je nutné pro stejnou míru bezpečnosti zdvojnásobit délku otisku, resp. klíče [51]. Není ale vyloučeno, že se v budoucnu najdou nové způsoby, jak efektivně řešit určité problémy pomocí kvantového počítače, a další protokoly budou shledány nedostatečnými pro bezpečné použití.

I když kvantové počítače v budoucnu umožní prolomení dnes běžně používaných systémů, neznamená to, že nemá cenu se takovými systémy již zabývat – není jisté, kdy technologie umožní stavbu dostatečně výkonných kvantových počítačů, aby představovaly hrozbu.

Kvantové počítače místo klasických bitů využívají jejich kvantovou obdobu, tzv. *qubity* (blíže jsou popsány v části 5.1.1). Dnešní kvantové počítače pracují i s několika desítkami nebo až s okolo stovkou qubitů [52][53][54] – k prolomení šifry RSA s modulem délky 2 048 bitů v řádech sekund by bylo třeba přes 4 000 dokonale stabilních qubitů [55]. To sice není nijak závratné číslo, ale v současné době při výpočtech dochází mj. k určité chybovosti, což výpočet komplikuje. Se započtením chybovosti by k prolomení stejného typu RSA v řádu hodin dle současných poznatků bylo potřeba kvantového počítače s přibližně 20 000 000 qubity [56]. Navíc ani nemusí nastat doba strojů schopných prolomit dnes běžné šifrovací algoritmy v řádu hodin – považuje-

li útočník tajemství za tak cenné, klidně stráví výpočtem dny nebo měsíce, na což by nebyl potřeba tak výkonný stroj; ale i taková doba k prolomení je obrovský krok kupředu oproti současným počítačům. Sdílení tajemství může obsahovat informace, které jsou velice dlouhou dobu neměnné a cenné, a tím je dán útočníkovi dostatek času k prolomení tajemství.

Kvantové počítače a technologie nepřináší jenom hrozbu pro bezpečnost klasických algoritmů – přináší také nové způsoby pro realizaci kryptografických algoritmů, které mohou mít oproti klasickým variantám rozdílné vlastnosti. Mezi příklady použití kvantové kryptografie patří kvantová distribuce klíče (protokol BB84 [44]), kvantové coin-flipping protokoly [44], kvantový protokol pro oblivious transfer [57] nebo kvantové sdílení tajemství [58](příklad je popsán v sekci 5.2).

5.1 Kvantové počítání

Oproti klasické informatice, ve které je základní jednotkou jeden bit, reprezentující buď stav 0, nebo stav 1, v kvantové informatice se používá kvantový bit, tzv. *qubit*. Qubit se kromě dvou základních stavů může nacházet v *superpozici*, neboli lineární kombinaci těchto dvou stavů. Navíc se kvantové počítání liší od klasického v základních principech, jako např. že měřením qubitu se ovlivní jeho stav nebo že není možné vytvořit přesnou kopii neznámého qubitu. [59]

5.1.1 Notace a základní vlastnosti kvantového počítání

Jak je uvedeno výše, základní jednotkou pro výpočet je qubit, jeho stav se dá vyjádřit vektorem. Vektory jsou značeny pomocí Diracovy notace [60] $|\cdot\rangle$, tj. skutečnost, že ψ je vektor, je značena $|\psi\rangle$. Následující definice vychází z [59]:

Definice 5.1 Nechť $\alpha, \beta \in \mathbb{C}$, pro které platí $|\alpha|^2 + |\beta|^2 = 1$, a nechť $\mathcal{H}$ je dvourozměrný Hilbertův prostor s bází $\{|0\rangle, |1\rangle\}$. *Qubit* je normalizovaný vektor v prostoru $\mathcal{H}$ tvaru

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{5.1}$$

V rovnici výše jsou α a β tzv. amplitudy a platí pro ně normalizační podmínka ($|\alpha|^2 + |\beta|^2 = 1$). Vektory $|0\rangle$ a $|1\rangle$ spolu tvoří bázi prostoru, tato báze se nazývá *výpočetní*. Další z často používaných bází je báze $\{|+\rangle, |-\rangle\}$ a její vektory jsou dány vztahem $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$, resp. $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$.

Jednou ze základních vlastností qubitů je, že změřením jejich stavu se nezjistí hodnoty α nebo β; získá se buď hodnota $|0\rangle$ s pravděpodobností $|\alpha|^2$, nebo hodnota $|1\rangle$ s pravděpodobností $|\beta|^2$. Původní stav se měřením pozmění, resp. zničí, tj. je-li měřen stav $|\psi\rangle$ v bázi $\{|0\rangle, |1\rangle\}$ s výsledkem $|0\rangle$, tak po změření bude $|\psi\rangle$ ve stavu $|0\rangle$ a informace o původním stavu je ztracena. [59]

Systém se samozřejmě může skládat z více než jednoho qubitu. Obecně, systém s n qubity bude popsán vektorem v prostoru vzniklém tenzorovým součinem n příslušných dvourozměrných prostorů (použitých pro systém s jedním qubitem). Báze takového prostoru vznikne tenzorovým součinem bazických vektorů jednotlivých systémů, značení těchto vektorů se obvykle zkracuje: [59]

$$|v_1\rangle \otimes |v_2\rangle \otimes \ldots \otimes |v_n\rangle = |v_1 v_2 \ldots v_n\rangle \tag{5.2}$$

Protože dimenze prostoru pro systémy s jedním qubitem je 2, je dimenze prostoru pro systémy s n qubity 2^n. Báze má tak 2^n prvků, a pro systémy se dvěma qubity může vypadat následovně (každý z jednobitových systémů měl bázi $\{|0\rangle, |1\rangle\}$):

$$\{|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle\} = \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\} \tag{5.3}$$

Tato báze samozřejmě není jediná možná; pro systémy se dvěma qubity se používá i např. Bellova báze popsaná níže v textu.

Systém dvou qubitů se dá obecně vyjádřit tvarem

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \tag{5.4}$$

přičemž $\alpha_{00}, \alpha_{01}, \alpha_{10}, \alpha_{11} \in \mathbb{C}$ a pro všechny amplitudy platí normalizační podmínka $|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1$. [59]

Qubity v systémech s více qubity mohou být na sobě vzájemně závislé a nelze k nim přistupovat samostatně. V takovém případě se jedná o *provázané* qubity, které hrají důležitou roli v kvantovém schématu ke sdílení tajemství, které je popsáno v následující části.

Definice 5.2 Stav systému s n qubity je provázaný, nejde-li tento stav zapsat jako tenzorový součin stavů jednotlivých n podsystémů. [59]

Pro schéma popsané níže jsou z provázaných systémů s více qubity důležité tzv. *Bellovy stavy* (patří mezi systémy skládající se ze dvou qubitů) a *GHZ stav* (skládající se ze tří qubitů). Nejprve jsou popsány Bellovy stavy, na kterých je demonstrována vlastnost provázaných systémů.

Definice 5.3 *Bellovy stavy* jsou systémy dvou qubitů v následujících podobách: [59]

$$\begin{aligned}
|\Phi^+\rangle &= |\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \\
|\Psi^+\rangle &= |\beta_{01}\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}} \\
|\Phi^-\rangle &= |\beta_{10}\rangle = \frac{|00\rangle - |11\rangle}{\sqrt{2}} \\
|\Psi^-\rangle &= |\beta_{11}\rangle = \frac{|01\rangle - |10\rangle}{\sqrt{2}}
\end{aligned} \tag{5.5}$$

Vlastnost provázanosti systému bude demonstrována na příkladu Bellova stavu $|\Phi^+\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$. Pak měření prvního qubitu určuje výsledek měření druhého qubitu; pokud byl první qubit roven $|0\rangle$, resp. $|1\rangle$, bude výsledkem měření druhého qubitu $|0\rangle$, resp. $|1\rangle$. V prvním měření mají možnosti navíc stejnou pravděpodobnost, neboť $\alpha = \beta = \frac{1}{\sqrt{2}}$ a tudíž $|\alpha|^2 = |\beta|^2 = \frac{1}{2}$. Tato skutečnost se označuje jako *maximálně provázaný* stav. Výsledek měření prvního qubitu je tedy sice náhodný, ale výsledek měření druhého qubitu je vždy stejný jako výsledek prvního měření. [59]

Pro systém se dvěma qubity se kromě výpočetní báze $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ používá taktéž Bellova báze, sestávající ze čtyř výše zmíněných vektorů.

Definice 5.4 Systém se třemi qubity v provázaném stavu tvaru

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \tag{5.6}$$

se nazývá *GHZ* stav. [61][58]

Tento systém je také provázaný, tj. jeho stav nelze zapsat jako tenzorový součin stavů jednotlivých třech podsystémů, ale navíc ho nelze rozepsat ani jako tenzorový součin dvou částí, přičemž jedna obsahuje stav jednoho podsystému a druhá část obsahuje stav zbylých dvou podsystémů.

Věta 5.1 Nechť je dán neznámý kvantový stav $|\psi\rangle$ a stav $|0\rangle$ nad prostorem $\mathcal{H}$. Pak neexistuje lineární operátor U, který by neznámý stav $|\psi\rangle$ zkopíroval, tj. splňoval by následující rovnici:

$$U(|\psi\rangle |0\rangle) = |\psi\rangle |\psi\rangle \tag{5.7}$$

Tato věta je známá jako *věta o nekopírovatelnosti*. [62]

Důkaz. Důkaz je proveden sporem a vyplývá z linearity operátoru. Neznámý stav $|\psi\rangle$ se dá zapsat jako lineární kombinace $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$. Rozepsáním stavu $|\psi\rangle$ z výsledku rovnice výše vznikne následující vztah:

$$\begin{aligned} U(|\psi\rangle |0\rangle) = |\psi\rangle |\psi\rangle &= (\alpha |0\rangle + \beta |1\rangle)(\alpha |0\rangle + \beta |1\rangle) \\ &= \alpha^2 |00\rangle + \alpha\beta |01\rangle + \beta\alpha |10\rangle + \beta^2 |11\rangle \end{aligned} \tag{5.8}$$

Z rozepsání stavu $|\psi\rangle$ a z linearity kopírujícího operátoru plyne tato rovnost

$$\begin{aligned} U(|\psi\rangle |0\rangle) = U((\alpha |0\rangle + \beta |1\rangle) |0\rangle) &= \alpha U(|0\rangle |0\rangle) + \beta U(|1\rangle |0\rangle) \\ &= \alpha |00\rangle + \beta |11\rangle \end{aligned} \tag{5.9}$$

Výsledky obou rovnic jsou ve sporu.

Tato vlastnost má zajímavý důsledek pro schémata pro sdílení tajemství – pokud by existovalo kvantové prahové (k, n) schéma takové, že $k \leq \frac{n}{2}$, pak by existovaly dvě různé množiny dílů s prázdným průnikem, z kterých by šlo tajemství rekonstruovat. Z toho vyplývá, že by bylo možné nezávisle na sobě alespoň dvakrát zrekonstruovat kvantovou informaci – to ale odporuje větě o nekopírovatelnosti, proto taková schémata nejsou možná. Každé kvantové (k, n) schéma tak musí splňovat podmínku $k > \frac{n}{2}$). O tuto skutečnost se opírá i nemožnost nepozorovaného odposlouchávání kvantové komunikace – protože změřením se stav qubitu zničí a původní neznámý qubit nelze přesně zkopírovat, odposlouchávání vnese do komunikace chyby.

Nemožnost zkopírování neznámého stavu ale neznamená, že žádné stavy nelze zkopírovat, z rovnic výše vyplývá, že na sebe ortogonální stavy zkopírovat lze [62]. I když nelze vytvořit přesnou kopii, je možné vytvořit aproximace původního qubitu [63].

5.1.2 Kvantové operace

Stejně jako v případě klasických obvodů, i kvantové obvody se skládají z hradel – operací na jednom nebo více qubitech. Tyto operace se dají vyjádřit pomocí maticových operací. Po provedení těchto operací musí být zachována normalizační podmínka. Skupina matic, po jejichž aplikaci je normalizační podmínka dodržena, se nazývá *unitární* matice. V následující části jsou představeny operace používané v kvantovém schématu pro sdílení tajemství, které je popsáno v následující části. Mezi základní operace patří operace realizované pomocí tzv. *Pauliho matic*; dvě z nich jsou popsány níže. Následující definice vychází z [59].

Pauliho matice σ_x

Je-li qubit ve stavu $\alpha |0\rangle + \beta |1\rangle$, jeho negace znamená navzájem zaměnit amplitudy α a β; výsledkem je $\beta |0\rangle + \alpha |1\rangle$. Operaci negace – obdobu klasického hradla NOT – realizuje následující matice, označovaná jako Pauliho σ_x matice, nebo písmenem X:

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{5.10}$$

Skutečně, tato matice provádí požadovanou operaci, neboť

$$X \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix} \tag{5.11}$$

Normalizační podmínka je zachována, došlo pouze k zaměnění amplitud.

Pauliho matice σ_z

Další ze základních operací je realizována Pauliho σ_z maticí, též značenou písmenem Z. Matice je definována následovně:

$$\begin{aligned} Z &\equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \\ Z \begin{bmatrix} \alpha \\ \beta \end{bmatrix} &= \begin{bmatrix} \alpha \\ -\beta \end{bmatrix} \end{aligned} \tag{5.12}$$

Tato operace opět zachovává normalizační podmínku.

Hadamardova matice

Dalším hradlem je tzv. Hadamardovo hradlo, představované Hadamardovou maticí, značenou H.

$$\begin{aligned} H &\equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ H \begin{bmatrix} \alpha \\ \beta \end{bmatrix} &= \frac{1}{\sqrt{2}} \begin{bmatrix} \alpha + \beta \\ \alpha - \beta \end{bmatrix} \end{aligned} \tag{5.13}$$

Aplikováním rovnice výše na stav $\alpha |0\rangle + \beta |1\rangle$ vznikne rovnice

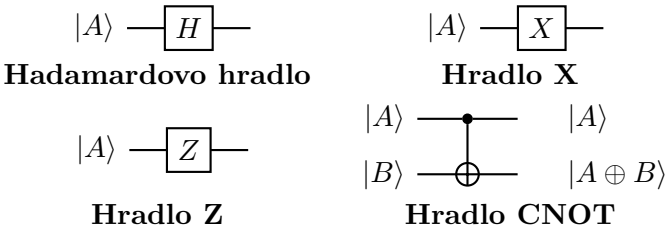
$$\frac{1}{\sqrt{2}}(\alpha + \beta) |0\rangle + \frac{1}{\sqrt{2}}(\alpha - \beta) |1\rangle = \alpha(\frac{|0\rangle + |1\rangle}{\sqrt{2}}) + \beta(\frac{|0\rangle - |1\rangle}{\sqrt{2}}) \tag{5.14}$$

Opět je zachována norma amplitud. Tato operace efektivně transformuje $|0\rangle \mapsto \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ a $|1\rangle \mapsto \frac{|0\rangle-|1\rangle}{\sqrt{2}}$. Tímto způsobem vznikne superpozice stavu. Díky této vlastnosti může být hradlo využito k vytvoření provázaných stavů, jako jsou Bellovy stavy nebo GHZ stav (viz sekce 5.1.2).

Kontrolovaná negace

Dalším často používaným hradlem je hradlo $CNOT$, neboli kontrolovaná negace. Toto hradlo má dva vstupy, první qubit funguje jako kontroler – jeho hodnota určuje, zda se provede negace druhého; je-li první qubit roven $|1\rangle$, pak je druhý qubit negován. Toto hradlo se dá považovat za obdobu klasického hradla XOR. Provádí tedy následující transformace:

$$|00\rangle \mapsto |00\rangle, |01\rangle \mapsto |01\rangle, |10\rangle \mapsto |11\rangle, |11\rangle \mapsto |10\rangle \tag{5.15}$$



Obrázek 5.1: Schematické značky kvantových hradel.

Tuto operaci lze provést maticovou transformací:

$$U_{CN} \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$
$$U_{CN} \begin{bmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{bmatrix} = \begin{bmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{11} \\ \alpha_{10} \end{bmatrix} \tag{5.16}$$

Schematické značení kvantových hradel

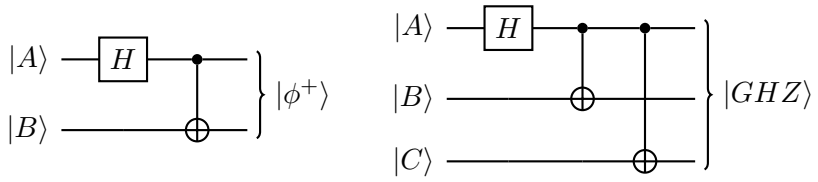
Stejně jako v případě klasických obvodů, i kvantová hradla mají svoje schematické značky. Obrázek 5.1 zobrazuje značení výše popsaných hradel.

S pomocí popsaných operací lze dvojici qubitů transformovat do Bellova stavu; je k tomu zapotřebí Hadamardova hradla a hradla CNOT. Podobným způsobem je možné vytvořit provázaný GHZ stav. Obě transformace jsou popsány v následující části.

Realizace Bellova a GHZ stavu pomocí kvantových hradel

Jak bylo zmíněno výše, pomocí popsaných hradel lze transformovat kvantový stav do superpozice nebo vytvořit provázaný stav. Jako první bude popsána transformace pro vytvoření Bellova stavu. Popis, který následuje, vychází z [59].

Nechť jsou dány stavy $|A\rangle, |B\rangle \in \{|0\rangle, |1\rangle\}$, cílem je vytvořit jejich provázáním Bellův stav. Toho lze dosáhnout aplikováním Hadamardova hradla a kontrolované negace dle obrázku 5.2 vlevo. Aplikováním Hadamardova hradla se vstupní stav $|A\rangle$ transformuje na stav $\frac{(|0\rangle+|1\rangle)}{\sqrt{2}}$, pokud byl roven $|0\rangle$, jinak se transformuje na stav $\frac{(|0\rangle-|1\rangle)}{\sqrt{2}}$. Tento superponovaný stav pak slouží jako kontrolní qubit pro hradlo CNOT. Celkový stav po aplikování Hadamardova



Tvorba Bellova stavu **Tvorba GHZ stavu**

Obrázek 5.2: Vytvoření Bellova a GHZ stavu pomocí hradel kontrolované negace a Hadamardova hradla.

hradla se tak dá vyjádřit jako

$$\begin{array}{c} \frac{(|0\rangle + |1\rangle)\,|B\rangle}{\sqrt{2}} \\ \frac{(|0\rangle - |1\rangle)\,|B\rangle}{\sqrt{2}} \end{array} \tag{5.17}$$

Rozepsáním možných hodnot $|B\rangle$ a aplikováním hradla kontrolované negace vzniknou požadované Bellovy stavy:

$$\begin{aligned} |A\rangle = |0\rangle, |B\rangle = |0\rangle &\rightarrow \frac{(|00\rangle + |11\rangle)}{\sqrt{2}} = |\Phi^+\rangle \\ |A\rangle = |0\rangle, |B\rangle = |1\rangle &\rightarrow \frac{(|01\rangle + |10\rangle)}{\sqrt{2}} = |\Psi^+\rangle \\ |A\rangle = |1\rangle, |B\rangle = |0\rangle &\rightarrow \frac{(|00\rangle - |11\rangle)}{\sqrt{2}} = |\Phi^+\rangle \\ |A\rangle = |1\rangle, |B\rangle = |1\rangle &\rightarrow \frac{(|01\rangle - |10\rangle)}{\sqrt{2}} = |\Psi^-\rangle \end{aligned} \tag{5.18}$$

Aplikováním hradla kontrolované negace se dosáhne požadovaného Bellova stavu a qubity byly transformovány správně. Vytvoření GHZ stavu je obdobné; je třeba jednoho Hadamardova hradla a dvou kontrolovaných negací. Schéma obvodu je znázorněno na obrázku 5.2 vpravo. Tento obvod obsahuje jako svoji část obvod pro vytvoření Bellova stavu; obvod je rozšířen pouze o jednu kontrolovanou negaci. Pro vytvoření GHZ stavu ve tvaru (5.6) jsou vstupní hodnoty $|A\rangle$ a $|B\rangle$ rovny $|0\rangle$. Před aplikací poslední kontrolované negace je stav systému následující:

$$\frac{(|00\rangle + |11\rangle)\,|C\rangle}{\sqrt{2}} \tag{5.19}$$

Což je jeden z Bellových stavů. Ať je poslední vstupní hodnota $|C\rangle$ rovna $|0\rangle$ nebo $|1\rangle$, v obou případech je po aplikaci poslední kontrolované negace stav systému roven

$$|A\rangle = |0\rangle, |B\rangle = |0\rangle, |C\rangle \in \{|0\rangle, |1\rangle\} \rightarrow \frac{(|000\rangle + |111\rangle)}{\sqrt{2}} \tag{5.20}$$

To je stav ve tvaru požadovaného GHZ stavu a obvod skutečně transformuje qubity do GHZ stavu.

5.2 Schéma Hillery, Bužek, Berthiaume

Toto schéma bylo první kvantové schéma pro sdílení tajemství, které bylo popsáno; umožňuje sdílet jeden qubit mezi dvěma účastníky tak, že k rekonstrukci qubitu je třeba jejich vzájemná spolupráce, je to tedy schéma typu $(2, 2)$ a neporušuje větu o nekopírovatelnosti. Ke sdílení qubitu se využívá systém tří provázaných qubitů GHZ stavu. [58]

Nechť distributor a dva účastníci sdílí společný GHZ stav skládající se ze tří qubitů, přičemž každý z nich má tajně jeden qubit GHZ stavu. Sdílení jednoho tajného qubitu se pak skládá z tohoto jediného kroku:

1. Distributor změří v Bellově bázi dvojici tajného qubitu a svého qubitu sdíleného GHZ stavu. [58]

Pro rekonstrukci tajného qubitu je nutná spolupráce obou účastníků, a navíc je nutná i komunikace s distributorem. Tento protokol se skládá z následujících kroků:

1. Distributor náhodně vybere účastníka, který změří svůj qubit v bázi $|+\rangle, |-\rangle$.

2. Vybraný účastník a distributor pošlou zbylému účastníkovi výsledky svých měření.

3. Zbylý účastník na základě zaslaných informací transformuje svůj qubit na tajný qubit. [58]

Transformace použité k získání tajného stavu jsou popsány v tabulce 5.1 a nezohledňují výsledné znaménko. V základní variantě jsou k realizaci schématu potřeba pouze 4 qubity; tři qubity GHZ stavu a jeden tajný qubit. Zrekonstruovaný qubit nemůžou získat obě strany naráz, protože by to bylo v rozporu s větou o nekopírovatelnosti. Druhá strana může změřit zrekonstruovaný qubit a hodnotu poslat první straně, ale schéma neřeší situaci, kdy tak neučiní a první strana se tak nedozví tajnou hodnotu. Ve schématu distributor náhodně vybere stranu, která má svůj qubit změřit; pokud by se jednomu z účastníků podařilo odcizit tajný qubit druhého účastníka, je poloviční šance, že by zůstal neodhalen. Aby hned nebyl odhalen, musel by na místě odcizeného qubitu zanechat podvržený, a protože odcizený qubit nelze zkopírovat, a bez informací o měření distributora ho nemůže padělat, nutně by byl podvržený qubit jiný než odcizený a jeho chybnost by se dala detekovat. Distributor by s oběma stranami mohl sdílet více GHZ stavů, přičemž některé by se náhodně použily ke kontrole, zda žádná strana takto nepodváděla, a zbylé ke

Výsledky měření *D* a *A*		**Transformace**
$\|\Psi_+\rangle_D$	$\|+x\rangle_A$	I
$\|\Psi_+\rangle_D$	$\|-x\rangle_A$	Z
$\|\Psi_-\rangle_D$	$\|+x\rangle_A$	Z
$\|\Psi_-\rangle_D$	$\|-x\rangle_A$	I

Výsledky měření *D* a *A*		**Transformace**
$\|\Phi_+\rangle_D$	$\|+x\rangle_A$	X
$\|\Phi_+\rangle_D$	$\|-x\rangle_A$	XZ
$\|\Phi_-\rangle_D$	$\|+x\rangle_A$	XZ
$\|\Phi_-\rangle_D$	$\|-x\rangle_A$	X

Tabulka 5.1: Přehled transformací, které na základě výsledků měření distributora (D) a první strany (A) provede druhá strana, aby transformovala svůj qubit na tajný qubit. X a Y značí příslušné Pauliho matice a I je jednotková matice. [58]

sdílení tajemství. Více GHZ stavů lze také použít ke sdílení dalších qubitů mezi účastníky. [58]

Útočník by také mohl se sdíleným GHZ stavem provázat svůj vlastní qubit ve snaze získat tak víc informací; ale i v tomto případě by tak GHZ stav pozměnil a tuto změnu by bylo možné detekovat [58].

Kapitola 6

Realizace

Tato kapitola popisuje implementaci Shamirova, Pedersenova, Schoenmakersova a kvantového sdílení tajemství a implementaci BGW schématu pro bezpečné počítání více stran. Implementace kvantového sdílení tajemství je realizována pomocí skriptu `hbb.py`. Implementace zbylých algoritmů je zahrnuta v knihovně `vsssLib` a práce s touto knihovnou je popsána v aplikaci `demo`, která knihovnu `vsssLib` využívá.

Knihovna `vsssLib` a demo aplikace jsou napsány v jazyce C s využitím knihovny OpenSSL, skript kvantového sdílení tajemství je napsán v jazyce Python s využitím vývojové sady Qiskit.

6.1 Knihovna `vsssLib`

Zdrojové kódy ke knihovně `vsssLib` se nachází na přiloženém CD ve složce `src` v adresáři `vsssLib`. Zkompilovaná statická verze knihovny `vsssLib.lib` se nachází ve složce `bin` v podadresáři `vsssLib`. Tento podadresář dále obsahuje složku s dokumentací knihovny `doc` vygenerovanou pomocí aplikace Doxygen a složku s hlavičkovými soubory knihovny `include`. Pro reprezentaci dlouhých čísel je v knihovně použit datový typ `BIGNUM` z knihovny OpenSSL. K sestavení knihovny byla použita knihovna OpenSSL verze 3.0.2.

6.1.1 Shamirův algoritmus

Implementace rozdělení a rekonstrukce tajemství Shamirova schématu vychází ze zdrojových kódů bakalářské práce Algoritmy pro sdílení tajemství [64]. Zdrojové kódy pro funkce realizující Shamirovo sdílení tajemství se nachází v podadresáři `sss` v souboru `shamir.c`. Struktura reprezentující díl tajemství `share_t` se nachází v souboru `sss.h`. Struktura dílu je znázorněna ve výpisu 1. Ke struktuře existují funkce pro inicializaci, resp. uvolnění prostředků vnitřních proměnných `share_t_init`, resp. `share_t_clear`. Funkce `sh_share_secret` rozdělí tajemství na díly. Pomocná funkce `sh_make_coef`

Výpis 1 Struktura reprezentující jeden díl v Shamirově sdílení tajemství. Díl je bod polynomu se souřadnicemi (x,y). [64]

```
...
typedef struct share_t {
  BIGNUM* x;      ///<X coordinate of a point.
  BIGNUM* y;      ///<Y coordinate of a point.
} share_t;
...
```

Algoritmus 2 Lagrangeova interpolace v bodě *point* modulo *mod* [64]

procedure Interpolace modulo($S, point, k, mod$)
Vstup: pole bodů **S[0,. . .,k-1]**; bod pro výpočet interpolace **point**; počet bodů **k**; modulo **mod**.
Výstup: Hodnota Lagrangeova interpolačního polynomu pro bod **point** v proměnné **res**.
$sum \leftarrow 0$
for $i \leftarrow 0$ to k **do**
$prod \leftarrow 1$
for $j \leftarrow 0, j < k, j \leftarrow j + 1$ **do**
if $i = j$ **then** continue
end if
$prod \leftarrow prod \cdot ((point - S[j].x) \cdot \text{ModInv}(S[i].x - S[j].x, mod))$
end for
$res \leftarrow res + S[i].y \cdot prod \pmod{mod}$
end for
end procedure

nejdříve vygeneruje pseudonáhodné koeficienty pro polynom, který je následně použit pro rozdělení tajemství. Výpočet bodu na polynomu (tajného dílu) funkcí `sh_compute_point` má složitost $\mathcal{O}(n)$. Rekonstrukce tajemství znamená Lagrangeovu interpolaci v bodě 0, pseudokód interpolace je zobrazen v algoritmu 2. Složitost této implementace je $\mathcal{O}(k^2 \log(mod)^2)$, protože výpočet multiplikativní inverze je realizován rozšířeným Euklidovým algoritmem se složitostí $\mathcal{O}(\log(mod)^2)$.

Nově byly přidány funkce pro sčítání dílů `sh_share_add`, násobení dílu konstantou `sh_share_mul_const` a násobení dvou dílů `sh_share_mul`. Tyto funkce provedou patřičnou operaci se souřadnicemi y daných bodů. Protože je schéma homomorfní, výsledkem jsou opět platné díly. V případě násobení dvou dílů je ale stupeň polynomu použitého pro rozdělení tajemství zvýšen na $2(k-1)$ z $k-1$.

V knihovně je také zahrnuta funkce pro podvržení dílu `sh_forge_share`. Z poskytnutého dílu a $k-1$ souřadnic x dalších dílů vytvoří falešný díl. Pokud

Výpis 3 Struktura reprezentující jeden díl v Pedersenově ověřitelném sdílení tajemství. Díl se skládá ze dvou dílů Shamirova sdílení tajemství.

```
...
typedef struct ped_share_t {
  share_t share_f;    ///<Share of secret polynomial.
  share_t share_g;    ///<Share of commitment polynomial.
} ped_share_t;
...
```

je falešný díl použit při rekonstrukci s díly, jejichž souřadnice x jsou shodné se souřadnicemi poskytnutými při vytváření falešného dílu, bude rekonstrukce úspěšná, ale místo tajemství s bude zrekonstruovaná hodnota rovna $s - 1$. Ze zadaných souřadnic x jsou vytvořeny díly ve tvaru $(x, 0)$. K nim je přidán díl (bod) reprezentující tajemství $(0, -1)$. Z těchto k bodů reprezentujících uměle vytvořený polynom je interpolací v bodě x dílu, který má být podvržen, vypočítána hodnota uměle vytvořeného polynomu v tomto bodě. Součtem této hodnoty a původního dílu vznikne díky homomorfismu schématu platný díl, a protože funkční hodnota uměle vytvořeného polynomu v ostatních bodech byla nulová, díly ostatních jsou platné a výsledkem rekonstrukce je nesprávné tajemství. Složitost této funkce je daná složitostí Lagrangeovy interpolace.

6.1.2 Pedersenovo ověřitelné schéma

Funkce implementující Pedersenovo ověřitelné schéma se nachází v podadresáři `vsss` v souboru `pedersen.c`.

Společné struktury jsou zahrnuty v podadresáři `sss` v souboru `sss.h`. To zahrnuje jeden díl Pedersenova sdílení tajemství, který se skládá ze dvou dílů Shamirova schématu a je reprezentován strukturou `ped_share_t`. Veřejné parametry zahrnující prvočísla p a q a generátory g a h představuje struktura `vsss_comm_coef_t` znázorněná ve výpisu 4. K oběma strukturám existují funkce na inicializaci příslušných prostředků vnitřních proměnných struktury `ped_share_t_init`, resp. `ped_comm_coef_t_init`, a také funkce pro uvolnění těchto prostředků `ped_share_t_clear`, resp. `ped_comm_coef_t_clear`. Generování veřejných parametrů ze zadaného bezpečnostního parametru (délka prvočísla q) realizuje funkce `ped_commit_coef`, samotné generování prvočísel zadaného typu provádí funkce knihovny OpenSSL `BN_generate_prime_ex`. Pro vysoké hodnoty bezpečnostního parametru může být generování pomalé.

Vytvoření závazku je ve funkci `ped_commit_create` a realizuje závazek dle rovnice (3.2). Ověření závazku ve funkci `ped_commit_verify` probíhá novým výpočtem závazku a jeho porovnáním s poskytnutou hodnotou.

Ověření dílu tajemství obstarává funkce `ped_share_verify`. Díl v (k, n) schématu lze ověřit pomocí k porovnání nově vypočteného závazku k dílu a závazku vzniklého výpočtem ze závazků ke koeficientům polynomu dle rov-

Výpis 4 Struktura reprezentující veřejné parametry Pedersenova sdílení tajemství.

```
...
typedef struct vsss_comm_coef_t {
  BIGNUM* p;      ///<prime p.
  BIGNUM* q;      ///<prime q, q=(p-1)/2.
  BIGNUM* g;      ///<generator g.
  BIGNUM* h;      ///<generator h.
} vsss_comm_coef_t;
...
```

nice (3.8). Algoritmus výpočtu druhého závazku je znázorněn v algoritmu 5. Složitost tohoto řešení je $\mathcal{O}(k)$.

Samotné sdílení tajemství funkcí `ped_share_secret` je obdobné jako v případě Shamirova sdílení tajemství, pouze se navíc generují i závazky. Rekonstrukce tajemství funkcí `ped_reconstruct_secret` je taktéž obdobná, před samotnou rekonstrukcí ale proběhne ověření poskytnutých dílů. Složitost funkcí je tak oproti Shamirově sdílení tajemství navýšena o generování, resp. ověřování dílů.

Algoritmus 5 Výpočet závazku k dílu ze závazků ke koeficientům polynomu v Pedersenově sdílení tajemství.

procedure VERIFY SHARE($share, commitments, coef, k, mod$)
Vstup: Díl k ověření **share**; pole závazků ke koeficientům polynomu **commitments[0,. . .,k-1]**; veřejné parametry závazků **coef**; počet závazků **k**.
Výstup: Závazek k dílu **res** spočtený ze závazků **commitments**.
$res \leftarrow 1$, $exp \leftarrow 1$
for $i \leftarrow 0$ upto k **do**
if $i \neq 0$ **then** $exp \leftarrow exp^{share->share\_f.x} \pmod{coef \rightarrow p}$
end if
$res \leftarrow res \cdot exp^{commitments[j]} \pmod{coef \rightarrow p}$
end for
end procedure

6.1.3 Schoenmakersovo veřejně ověřitelné schéma

Zdrojové kódy funkcí implementujících Schoenmakersovo veřejně ověřitelné schéma se nachází v podadresáři `vsss` v souboru `schoenmakers.c`.

Toto veřejně ověřitelné schéma využívá pro reprezentaci dílu tajemství díl Shamirova schématu `share_t`. Pro reprezentaci veřejných parametrů je použita struktura `vsss_comm_coef_t`. Ve schématu jsou použity páry veřejných

a soukromých klíčů, takový pár lze vygenerovat funkcí `sch_keypair_create`, která na základě bezpečnostního parametru (délka prvočísla q z veřejných parametrů) vytvoří náhodný soukromý klíč a odpovídající veřejný klíč vhodný k použití v Schoenmakersově schématu. Generování parametrů je stejné jako v případě Pedersenova ověřitelného schématu.

Výpis 6 Struktura reprezentující hodnoty neinteraktivní verze Chaumova-Pedersenova protokolu.

```
...
typedef struct chaum_ped_proof_t {
  BIGNUM *r;   ///<r=w-c*S, c is challenge, S secret, w random.
  BIGNUM *g1;  ///<generator g1
  BIGNUM *h1;  ///<generator h1
  BIGNUM *g2;  ///<generator g2
  BIGNUM *h2;  ///<generator h2
  BIGNUM *a1;  ///<a1 = g1^w, w is random
  BIGNUM *a2;  ///<a2 = g2^w, w is random
} chaum_ped_proof_t;
...
```

Rozdělení náhodného tajemství je realizováno ve funkci `sch_share_secret`, která také vygeneruje potřebné závazky a důkazy k ověřování dílů. K ověření jsou v tomto schématu použity závazky pomocí diskrétního logaritmu a důkazy neinteraktivní verzí Chaumova-Pedersenova protokolu, jehož parametry jsou obsaženy ve struktuře `chaum_ped_proof_t`, která představuje důkaz a je definovaná v podadresáři `sss` v souboru `sss.c`. Ke struktuře také existuje funkce pro inicializaci vnitřních proměnných `chaum_ped_proof_t_init` a funkce k uvolnění vnitřních proměnných `chaum_ped_proof_t_clear`. Struktura je znázorněna ve výpisu 6. Struktura obsahuje hodnoty na základě výzvy c, která je vypočítána pomocí Fiatovy-Shamirovy techniky jako hash zřetězených parametrů. Výpočet výzvy obstarává funkce `sch_hash_generate`, která vrací hash zřetězených hodnot h_1, h_2, a_1 a a_1. Jako hashovací funkce je použita funkce SHA-256 a samotné hashování je provedeno posloupností volání funkcí knihovny OpenSSL, a to `EVP_DigestInit_ex`, `EVP_DigestUpdate` a `EVP_DigestFinal_ex`. Ověření důkazu zahrnuje nové spočtení hodnot důkazu a jejich porovnání s poskytnutým důkazem. Tato funkcionalita je implementována ve funkci `sch_proof_verify`.

Rozeslané díly lze veřejně ověřit pomocí veřejného klíče, důkazů a závazků pomocí funkce `sch_share_verify`, princip výpočtu hodnoty ze závazků je analogický k postupu v Pedersenově ověřitelném schématu v předchozí sekci.

Před samotnou rekonstrukcí tajemství je nutné díl dešifrovat soukromým klíčem a vytvořit důkaz o správnosti dešifrování, tuto funkcionalitu obstarává funkce `sch_share_decrypt_proof`. Funkce `sch_reconstruct_secret` ověří dešifrované díly a obdobou Lagrangeovy interpolace zrekonstruuje tajemství.

6.1.4 BGW schéma

Zdrojové kódy bezpečného počítání více stran se nachází v podadresáři `mpc` v souboru `bgw.c`. Tento soubor obsahuje implementaci hradel sčítání, násobení konstantou a násobení. Funkce sčítání a násobení konstantou `bgw_add`, resp. `bgw_mul_const` jsou přímočaré, protože pro jejich výpočet není nutná komunikace s ostatními účastníky. Využívají tak pouze funkce sčítání, resp. násobení konstantou pro díly Shamirova sdílení tajemství.

Hradlo reprezentující násobení není možné spočítat lokálně a je nutná komunikace s ostatními, proto je rozděleno do dvou funkcí, `bgw_mul_randred` a `bgw_mul_final`. Znáhodnění a redukce polynomu vychází z metody Gennara, Rabina a Rabina, která je popsaná v sekci 4.2.1 a je realizována funkcí `bgw_mul_randred`. Složitost této funkce se odvíjí od složitosti rozdělení tajemství Shamirovým schématem. Funkce `bgw_mul_final` z přijatých dílů ostatních zrekonstruuje konečnou hodnotu dílu po násobení, její složitost je dána složitostí rekonstrukce tajemství v Shamirově schématu. Tyto funkce jsou znázorněny ve výpisu 7.

6.2 Demo aplikace

Zdrojové kódy aplikace `demo.exe` se nachází na přiloženém CD ve složce `bin` v podadresáři `demo`. Tato aplikace demonstruje funkčnost algoritmů implementovaných v knihovně `vsssLib`.

6.2.1 Práce se soubory

Funkce použité pro ukládání dílů nebo závazků do souboru a jejich opětovné načtení ze souborů se nachází v souboru `in-out.c` v podadresáři `demo`.

Ukládá-li se do souboru typ `BIGNUM`, jsou do souboru nejprve uloženy 4B reprezentující počet bajtů ukládaného čísla, následují bajty ukládaného čísla. Ukládání a načítání typu `BIGNUM` realizují funkce `io_write_bn` a `io_read_bn`. Veřejné parametry Pedersenova nebo Schoenmakersova schématu jsou ukládány jako posloupnost čísel `BIGNUM`. V případě důkazu Chaumova-Pedersenova protokolu jsou do souboru postupně uloženy všechny `BIGNUM` hodnoty. To obstarávají funkce `io_write_chp_proof`, resp. `io_read_chp_proof`. Do souboru s klíčem Schoenmakersova schématu se nejprve ukládají koeficienty použité k jeho generování, následuje samotný klíč jako hodnota `BIGNUM`, ukládání a načítání klíčů realizují funkce `io_save_keys` a `io_load_keys`.

Funkce ukládající díly `io_save_shares` implementuje ukládání Shamirova, Pedersenova i Schoenmakersova schématu, typ použitého schématu je jedním z parametrů funkce. Soubory s díly Shamirova schématu obsahují nejprve 4B čísla k z (k, n) schématu následovaná dvěma čísly `BIGNUM` reprezentujícími souřadnice x a y. Díl Pedersenova schématu navíc ukládá druhou sadu čísel `BIGNUM` pro druhý polynom. Díly Schoenmakersova schématu obsahují díl

Výpis 7 První a druhá fáze hradla násobení v protokolu BGW, v první fázi je pro redukci stupně polynomu součin dílů rozdělen (k, n) schématem, ve druhé fázi jsou přijaté díly z první fáze použity pro rekonstrukci konečného dílu.

```
...
//Shares a and b are multiplied
//Result is used as secret and splitted in (k,n) scheme
int bgw_mul_randred(share_t** resShares, share_t* a, share_t* b,
                    int32_t k, int32_t n, BIGNUM* mod) {
  share_t mul;                      //multiplication of a and b
  int ret = ERROR_COMPUTING;        //initial return value
  do {  //error checking tone time while
    share_t_init(&mul);                    //initializing
    ret = sh_share_mul(&mul, a, b, mod); //multiplying shares
    if (ret != SUCCESS) break;
    *resShares = sh_share_secret(mul.y, k, n, mod); //splitting
    if (*resShares == NULL) {
      ret = ERROR_S_SHARING;
      break;
  } while (0);
  share_t_clear(&mul); //cleaning
  return ret;
}

//Shares from previous phase are sent to other parties and
//received values are used to reconstruct final share
int bgw_mul_final(share_t *res, share_t *shares, BIGNUM *x,
                  int32_t k, BIGNUM *mod) {
  int ret = ERROR_COMPUTING;
  if(BN_copy(res->x, x) == NULL) return ret; //x coordinate
  //reconstructing y coordinate of final share
  ret = sh_reconstruct_secret(res->y, shares, 2*(k-1)+1, mod);
  return ret;
}
...
```

Shamirova schématu následovaný důkazem Chaumova-Pedersenova protokolu. Jejím protějškem je funkce `io_load_shares`, která díly načítá.

Pro ukládání souborů jsou implementované ještě další funkce, např. funkce `io_append_number` k názvům souborů přiřadí pořadové číslo a je využita při ukládání více dílů najednou.

6.2.2 Použití knihovny `vsssLib`

Hlavní částí programu `demo.exe` je ukázka funkčnosti algoritmů popsaných v této práci a implementovaných v knihovně vsssLib.

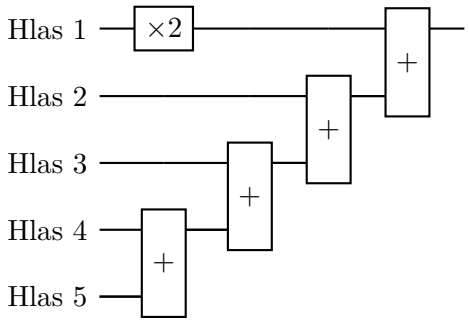
Princip čtení parametrů příkazové řádky a vykonávání funkcí na jejich základě je přejatý z [64]. Funkcionality aplikace se nachází v souboru `options.c` v podadresáři `demo`. Soubor `demo.c` obsahuje funkci `main` programu.

Každá z implementovaných funkcionalit aplikace je rozdělena do samostatné funkce, které využívají funkce knihovny `vsssLib`. Aplikace umožňuje rozdělit zadané hexadecimální tajemství Shamirovým (k, n) schématem a díly uložit do souborů (funkce `opt_sh_split`), tajemství ze zadaných dílů zrekonstruovat (funkce `opt_sh_rec`) nebo zadaný díl podvrhnout (`opt_sh_forge`). Pro Shamirovo sdílení tajemství je pevně nastavené modulo, v kterém jsou prováděny výpočty, a to na prvočíslo $2^{257} - 93$. Demo aplikace se tak hodí např. k rozdělování 256bitových šifrovacích klíčů. Hodnoty k a n (k, n) schémat jsou omezeny maximální hodnotou 100, která pokrývá většinu praktických využití aplikace.

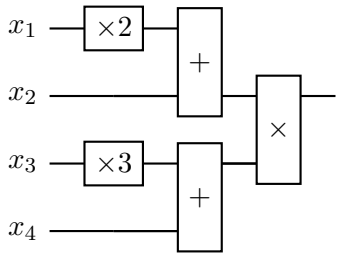
Dále umožňuje zadané tajemství rozdělit Pedersenovým (k, n) schématem, uložit díly a soubor se závazky (funkce `opt_ped_split`), poté tajemství ze zadaných dílů a závazků zrekonstruovat (funkce `opt_ped_rec`). Je také možné vygenerovat a uložit zadaný počet párů veřejných a soukromých klíčů pro Schoenmakersovo veřejně ověřitelné schéma(funkce `opt_sc_keys`). Jako další je možné rozdělit náhodné tajemství Schoenmakersovým veřejně ověřitelným schématem pomocí zadaných veřejných klíčů, díly s důkazy a závazky jsou uloženy do souborů (`opt_sc_split`). Na tuto funkcionalitu navazuje funkce `opt_sc_dec`, která načte uložené díly, závazky a páry klíčů, ověří platnost dílů, dešifruje je a dešifrované díly s důkazy o dešifrování uloží do souborů. Dešifrované díly je možné použít pro rekonstrukci tajemství ve funkci `opt_sc_rec`.

Bezpečné počítání více stran je zastoupeno BGW protokolem. V aplikaci je implementován BGW protokol simulací dvou různých počítaných funkcí. Ve funkci `opt_bgw_sim1` je sekvenčně simulováno hlasování. Funkce hlasování s pěti voliči x_1 až x_5, přičemž hlas prvního z nich má dvojnásobnou váhu, lze vyjádřit funkcí $f(x_1, x_2, x_3, x_4, x_5) = 2x_1 + x_2 + x_3 + x_4 + x_5$. Schéma aritmetického obvodu ekvivalentního této funkci je zobrazeno na obrázku 6.1. Obvod obsahuje pouze hradla sčítání a násobení konstantou, takže celou fázi emulace obvodu lze vypočítat bez komunikace s ostatními, což je v implementaci zdůrazněno postupnou emulací celého obvodu pro každého voliče ve smyčce.

Funkce `opt_bgw_sim2` simuluje bezpečné počítání funkce se čtyřmi proměnnými $f(x_1, x_2, x_3, x_4) = (2x_1 + x_2)(3x_3 + x_4)$. Schéma 6.2 vzniklo převedením této funkce na aritmetický obvod. V obvodu jsou zastoupena všechna tři hradla včetně násobení, ke kterému je nutná komunikace mezi ostatními. V implementaci je tato skutečnost zdůrazněna sekvenční emulací obvodu ve dvou samostatných smyčkách, kdy mezi smyčkami probíhá komunikace mezi účastníky a smyčky samotné jsou prováděny lokálně.



Obrázek 6.1: Aritmetický obvod realizující funkci hlasování s pěti voliči, kdy hlas prvního z nich má dvojnásobnou váhu.



Obrázek 6.2: Aritmetický obvod realizující funkci f se všemi třemi typy hradel, $f(x_1, x_2, x_3, x_4) = (2x_1 + x_2)(3x_3 + x_4)$.

6.2.3 Testování rychlosti

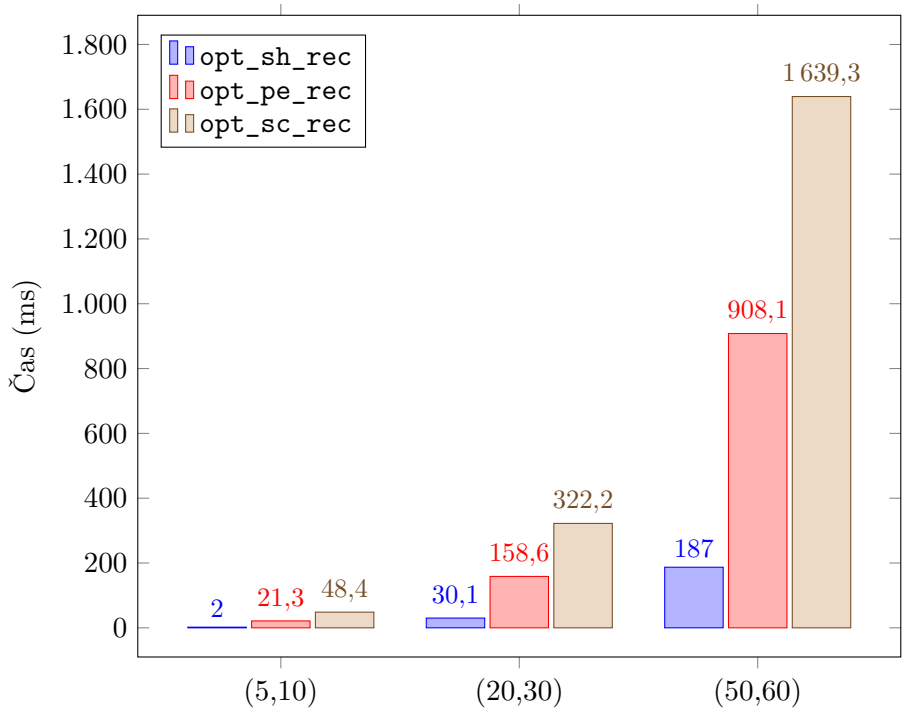
V této části je otestována a porovnána rychlost rekonstrukce tajemství u Shamirova, Pedersenova a Schoenmakersova schématu.

Pro testování byla vybrána rekonstrukce tajemství, a to z několika důvodů. Hlavní důvod je ten, že pokud jsou citlivá data – jako např. šifrovací klíče – rozdělena schématem pro sdílení tajemství, je pravděpodobné, že se jejich rekonstrukce neočekává často, ale pouze v krizových situacích, kdy čas může být důležitý faktor. Mezi další důvody patří fakt, že při rozdělování tajemství můžou být koeficienty u ověřitelných schémat dané předem, stejně jako šifrovací klíče, což implementace nezohledňuje.

Testována byla rychlost funkcí `opt_sh_rec`, `opt_pe_rec` a `opt_sc_rec`. Čas byl měřen knihovní funkcí `time`. Testovací počítač obsahoval procesor Intel Xeon E-2124G @3,40 GHz a operační systém Windows 10 Pro N 64-bit, version 21H2.

Testovány byly tři případy (k, n) schématu, a to pro hodnoty $(5, 10)$, $(20, 30)$ a $(50, 60)$. Pro každé (k, n) schéma byly funkce testovány desetkrát. U Pedersenova a Schoenmakersova schématu byl bezpečnostní parametr nastaven na 1 024 a u Shamirova schématu bylo jako modulo použito prvočíslo délky 1 024 bitů. Hodnoty byly nastaveny tak, aby všechna schémata umož-

ňovala sdílet tajemství stejné délky. Pro rekonstrukci tajemství ani ověřování dílů není směrodatná hodnota n, protože při rekonstrukci se pracuje s k díly a závazky se vztahují k polynomu, tudíž k hodnotě k. Graf 6.3 zobrazuje průměrné hodnoty časů funkcí přes všechna měření. Dle očekávání byla nej-



Obrázek 6.3: Čas průběhu funkcí rekonstruujících tajemství `opt_sh_rec`, `opt_pe_rec` a `opt_sc_rec` pro (k, n) schémata s různými hodnotami k a n.

rychlejší funkce `opt_sh_rec`, neboť díly neověřuje. Nejpomalejší byla funkce realizující rekonstrukci tajemství ve Schoenmakersově schématu, což je zapříčiněno jak nutností ověřovat důkazy platnosti dílů, tak skutečností, že funkce načítá $3k + 1$ souborů (soubor se závazky, k klíčů, k původních a k dešifrovaných dílů), zatímco funkce `opt_pe_rec` $k + 1$ a `opt_sh_rec` pouze k. Ale i v případě $(50, 60)$ schématu celá rekonstrukce trvala méně než dvě sekundy, ve zbylých případech desítky až stovky milisekund, takže je vhodná i pro použití, kde je čas rekonstrukce důležitým faktorem.

6.3 Obvod realizující kvantové schéma pro sdílení tajemství

Kvůli své povaze je kvantové schéma pro sdílení tajemství demonstrováno samostatně. Implementováno je schéma Hillery, Bužek a Berthiaume představené v předchozí kapitole. Fáze sdílení i rekonstrukce je implementována

najednou v jediném kvantovém obvodu. Implementace obvodu se nachází ve složce `src` v podadresáři `qsss` souboru `hbb.py`. K popsané implementaci není nutné mít přístup ke kvantovému počítači, kvantové obvody lze modelovat pomocí simulátoru.

K implementaci byla vybrána softwarová vývojová sada *Qiskit* [65], která umožňuje vytváření kvantových obvodů a jejich nasazení jak na simulátorech kvantových počítačů, tak i na skutečných kvantových počítačích společnosti IBM. Qiskit je projekt s otevřeným kódem využívající programovací jazyk Python. Pro vytvoření implementace obvodu byl použit Qiskit verze 0.34.1 a Python verze 3.9.1.

Implementovaný obvod potřebuje 4 qubity – 3 qubity sdíleného GHZ stavu a jeden tajný qubit. V obvodu je pevně zvolený účastník, který je vybrán pro změření svého qubitu, je to účastník vlastnící qubit $|A\rangle$. Qubit $|D\rangle$ má distributor, stejně jako tajný qubit $|S\rangle$. Účastník, který tajný qubit zrekonstruuje, má qubit $|B\rangle$. Qubity $|D\rangle, |A\rangle$ a $|B\rangle$ tvoří provázaný GHZ stav.

Qiskit umožňuje provádět měření qubitů ve výpočetní bázi $\{|0\rangle, |1\rangle\}$, ale účastník s qubitem $|A\rangle$ má provést měření v bázi $\{|+\rangle, |-\rangle\}$. Měření v bázi $\{|+\rangle, |-\rangle\}$ za pomoci výpočetní báze lze realizovat pomocí jednoho Hadamardova hradla před změřením qubitu (vyplývá z definice cílové báze a z rovnice (5.14)). Výsledek měření lze interpretovat následovně: $|0\rangle \rightarrow |+\rangle$ a $|1\rangle \rightarrow |-\rangle$.

Účastník s qubitem $|B\rangle$ má na základě měření ostatních provést se svým qubitem transformace dle tabulky 5.1. V implementaci je využita vlastnost Pauliho matic, že jsou samy k sobě inverzní, tj. pro ně platí:

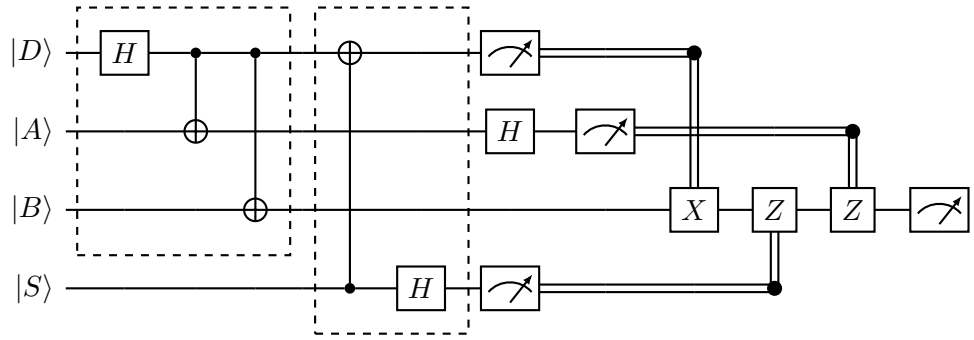
$$XX = I \qquad ZZ = I \tag{6.1}$$

Tato vlastnost jasně vyplývá z definic v sekci 5.1.2.

V implementaci pro transformaci qubitu stačí trojice hradel v pořadí X, Z a Z. Hradla jsou použita, resp. nepoužita dle výsledku měření qubitů $|D\rangle$, $|S\rangle$ a $|A\rangle$, které v tomto pořadí slouží jako kontrolní hodnota k výše zmíněným hradlům. Protože výsledkem transformací byla vždy opačná hodnota, než původní sdílená, je oproti tabulce 5.1 opačné použití hradla X. Výsledné transformace z tabulky 5.1 zachycuje tabulka 6.1. Celý obvod je schematicky znázorněn na obrázku 6.4. Kromě 4 qubitů obvod obsahuje také jeden klasický bit, který reprezentuje výsledky měření. Qubity jsou v Qiskitu ve výchozím nastavení inicializovány na hodnotu $|0\rangle$. Pro inicializaci qubitu na hodnotu $|1\rangle$ je možné na něj aplikovat hradlo X. Qiskit navíc umožňuje inicializovat qubity pomocí funkce `initialize`. V implementovaném obvodu je tato funkce využita k inicializaci tajného qubitu na hodnotu zadanou parametrem příkazové řádky, standardně je hodnota nastavena na $|1\rangle$. Obvod se z předem vytvořených kvantových a klasických bitů vytvoří pomocí funkce `QuantumCircuit`, jak je ukázáno ve výpisu 8. Na takto vytvořený obvod lze následně přidávat hradla a provádět měření. Ve výpisu 9 je uveden příklad vytvoření GHZ stavu. V obvodu se provádí transformace pomocí hradel X a Z na základě výsledků

Hodnoty $\|D\rangle$, $\|S\rangle$ a $\|A\rangle$		Transformace
$\|01\rangle_{SD}$	$\|0\rangle_A$	XZZ
$\|01\rangle_{SD}$	$\|1\rangle_A$	XZ
$\|11\rangle_{SD}$	$\|0\rangle_A$	XZ
$\|11\rangle_{SD}$	$\|1\rangle_A$	X
$\|00\rangle_{SD}$	$\|0\rangle_A$	ZZ
$\|00\rangle_{SD}$	$\|1\rangle_A$	Z
$\|10\rangle_{SD}$	$\|0\rangle_A$	Z
$\|10\rangle_{SD}$	$\|1\rangle_A$	

Tabulka 6.1: Použité transformace k získání tajného qubitu. Hodnota $|D\rangle$ určuje, zda bude aplikováno hradlo X, hodnota $|S\rangle$, resp. $|A\rangle$ určuje, zda bude použito první, resp. druhé hradlo Z.



Obrázek 6.4: Schéma obvodu realizujícího rozdělení a rekonstrukci tajného qubitu $|S\rangle$. Qubity $|D\rangle$, $|A\rangle$ a $|B\rangle$ tvoří provázaný GHZ stav (přerušovaně vlevo), příprava k měření $|D\rangle$ a $|S\rangle$ v Bellově bázi je naznačena přerušovaně vpravo. Výsledky měření ostatních qubitů rozhodují o transformacích qubitu $|B\rangle$ na tajný qubit.

měření, což je možné pomocí funkce `c_if`, která aplikuje nebo neaplikuje hradlo na základě hodnoty klasického bitu, ukázka je ve výpisu 10. K simulaci obvodu byl vybrán výchozí simulátor `AerSimulator`. Pro demonstraci správnosti je obvod simulován vícekrát za sebou a do konzole jsou vypsány počty, kolikrát byl rekonstruovaný qubit změřen s hodnotou $|0\rangle$ a $|1\rangle$. Qiskit umožňuje simulovat také šum a chyby měření, v obvodu ale pro jednoduchost nejsou simulovány. Kromě výsledků je do konzole vykresleno i schéma implementovaného obvodu. Vzorový výstup skriptu s tajným qubitem hodnoty $|1\rangle$ a s počtem opakování obvodu nastaveném na 100 je znázorněn v obrázku 6.5.

Výpis 8 Vytvoření prázdného obvodu se čtyřmi qubity a jedním klasickým bitem.

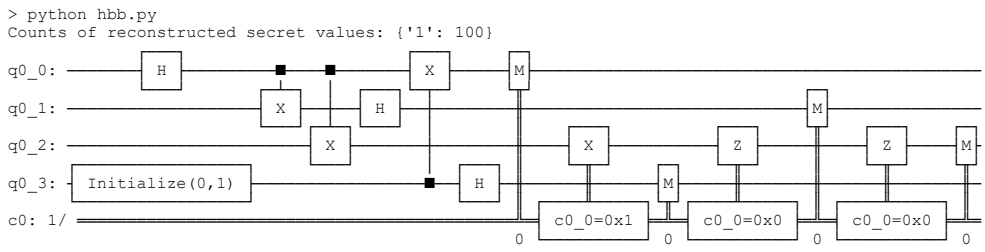
```
...
#4 qubits and 1 classical bit
qr = QuantumRegister(4)
cr = ClassicalRegister(1)
#creating empty circuit
circuit = QuantumCircuit(qr,cr)
...
```

Výpis 9 Vytvoření provázaného GHZ stavu tří qubitů.

```
...
#adding gates to put qubits 0,1,2 (D,A,B) in GHZ state
circuit.h(qr[0])            #Hadamard gate on qubit 0
circuit.cx(qr[0], qr[1])    #CNOT gate, control qubit 0, target 1
circuit.cx(qr[0], qr[2])    #CNOT gate, control qubit 0, target 2
...
```

Výpis 10 Příklad měření qubitu a podmíněné aplikace hradla X na základě výsledku měření.

```
...
#measuring qubit 0 (D), result is in classical register cr[0]
circuit.measure(qr[0],cr[0])
#appyling transformation to qubit 2 (B) depending on result
circuit.x(qr[2]).c_if(cr[0], 1)
...
```



Obrázek 6.5: Příklad výstupu spuštění skriptu `hbb.py` s výchozími parametry. Nejprve jsou vypsány výsledky všech 100 běhů simulace obvodu, následně je zobrazeno schéma implementovaného obvodu.

Závěr

Práce se zabývala schématy pro sdílení tajemství, ověřitelnými a kvantovými schématy a bezpečným počítáním více stran. Bylo popsáno Shamirovo základní schéma pro sdílení tajemství a popsány jeho nedostatky. Dále byly popsány možnosti odstranění těchto nedostatků pomocí ověřitelných algoritmů pro sdílení tajemství a bezpečného počítání více stran, u nichž byla provedena bezpečnostní analýza, i ve vztahu ke kvantovým počítačům. Z algoritmů byl implementován Shamirův algoritmus pro sdílení tajemství, Pedersenův ověřitelný a Schoenmakersův veřejně ověřitelný algoritmus pro sdílení tajemství. Bezpečné počítání více stran bylo demonstrováno protokolem BGW. Tyto algoritmy byly implementovány v knihovně `vsssLib`, demonstrovány aplikací `demo.exe` a byla diskutována rychlost rekonstrukce tajemství. Z kvantových schémat bylo popsáno schéma, které popsali Hillery, Bužek a Berthiaume a toto schéma bylo implementováno pomocí simulátoru kvantového počítače. Knihovnu `vsssLib` by bylo možné do budoucna rozšířit o další algoritmy a demonstrační aplikaci doplnit o další funkcionality.

Bibliografie

1. SHAMIR, Adi. How to Share a Secret. *Communications of the ACM* [online]. 1979, roč. 22, č. 11, s. 612–613 [cit. 2021-12-04]. ISSN 0001-0782. Dostupné z DOI: 10.1145/359168.359176.
2. BLAKLEY, G. R. Safeguarding cryptographic keys. In: *Proceedings of the AFIPS 1979 National Computer Conference* [online]. Montvale: AFIPS Press, 1979, sv. 48, s. 313–317 [cit. 2021-12-04]. Dostupné z DOI: 10.1109/AFIPS.1979.98.
3. ASMUTH, Charles; BLOOM, John. A Modular Approach to Key Safeguarding. *IEEE Transactions on Information Theory* [online]. 1983, roč. 29, č. 2, s. 208–210 [cit. 2021-12-04]. ISSN 0018-9448. Dostupné z DOI: 10.1109/TIT.1983.1056651.
4. NAOR, Moni; SHAMIR, Adi. Visual cryptography. In: DE SANTIS, Alfredo (ed.). *Advances in Cryptology — EUROCRYPT'94* [online]. Springer Berlin Heidelberg, 1995, s. 1–12 [cit. 2021-12-04]. ISBN 978-3-540-44717-7.
5. BENALOH, Josh Cohen. Secret Sharing Homomorphisms: Keeping Shares of a Secret Secret (Extended Abstract). In: ODLYZKO, Andrew M. (ed.). *Advances in Cryptology — CRYPTO' 86* [online]. Springer Berlin Heidelberg, 1987, s. 251–260 [cit. 2021-12-04].
6. PEDERSEN, Torben Pryds. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: *Advances in Cryptology — CRYPTO '91* [online]. Berlín: Springer, 1992, s. 129–140 [cit. 2021-12-04]. ISBN 978-3-540-46766-3. Dostupné z DOI: 10.1007/3-540-46766-1_9.
7. VITÁSEK, Emil. *Numerické metody*. Praha: SNTL, 1987.
8. TOMPA, Martin; WOLL, Heather. How to Share a Secret with Cheaters. *J. Cryptology* [online]. 1988, roč. 1, s. 133–138 [cit. 2021-12-05]. Dostupné z DOI: 10.1007/BF02252871.

9. HERZBERG, Amir; JARECKI, Stanislaw; KRAWCZYK, Hugo; YUNG, Moti. Proactive Secret Sharing Or: How to Cope With Perpetual Leakage. In: [online]. 1995-01, s. 339–352 [cit. 2021-12-05].

10. DAMGÅRD, Ivan; PEDERSEN, Torben; PFITZMANN, Birgit. Statistical Secrecy and Multibit Commitments. *IEEE Transactions on Information Theory* [online]. 1998-05, roč. 44, s. 1143–1151 [cit. 2022-01-15]. Dostupné z DOI: 10.1109/18.669255.

11. NAOR, Moni. Bit Commitment Using Pseudo-Randomness. In: *Advances in Cryptology - CRYPTO '89* [online]. Springer, 1989, s. 128–136 [cit. 2022-01-15]. Dostupné z DOI: 10.1007/0-387-34805-0_13.

12. GOLDREICH, Oded; MICALI, Silvio; WIGDERSON, Avi. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In: *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*. 1986, s. 174–187. Dostupné z DOI: 10.1109/SFCS.1986.47.

13. BLUM, Manuel. Coin Flipping by Telephone a Protocol for Solving Impossible Problems. *SIGACT News* [online]. 1983-01, roč. 15, č. 1, s. 23–27 [cit. 2022-01-19]. ISSN 0163-5700. Dostupné z DOI: 10.1145/1008908.1008911.

14. GOLDREICH, Oded. *Foundations of Cryptography* [online]. Cambridge University Press, 2001 [cit. 2021-12-04]. ISBN 0-521-79172-3. Dostupné z DOI: 10.1017/CBO9780511546891.

15. GROVER, Lov K. A Fast Quantum Mechanical Algorithm for Database Search. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing* [online]. Association for Computing Machinery, 1996, s. 212–219 [cit. 2022-01-15]. ISBN 0897917855. Dostupné z DOI: 10.1145/237814.237866.

16. SHOR, Peter W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM J. Comput.* [Online]. 1997, roč. 26, č. 5, s. 1484–1509 [cit. 2021-12-04]. ISSN 0097-5397. Dostupné z DOI: 10.1137/S0097539795293172.

17. HALEVI, Shai; MICALI, Silvio. Practical and Provably-Secure Commitment Schemes from Collision-Free Hashing. In: *Advances in Cryptology — CRYPTO '96* [online]. 1996, s. 201–215 [cit. 2022-03-05]. Dostupné z DOI: 10.1007/3-540-68697-5_16.

18. CARTER, J.Lawrence; WEGMAN, Mark N. Universal classes of hash functions. *Journal of Computer and System Sciences* [online]. 1979, roč. 18, č. 2, s. 143–154 [cit. 2022-01-15]. ISSN 0022-0000. Dostupné z DOI: https://doi.org/10.1016/0022-0000(79)90044-8.

19. CHOR, Benny; GOLDWASSER, Shafi; MICALI, Silvio; AWERBUCH, Baruch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In: *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*. 1985, s. 383–395. Dostupné z DOI: 10.1109/SFCS.1985.64.

20. STADLER, Markus. Publicly Verifiable Secret Sharing. In: MAURER, Ueli (ed.). *Advances in Cryptology — EUROCRYPT '96*. Springer Berlin Heidelberg, 1996, s. 190–199. ISBN 978-3-540-68339-1.

21. SCHOENMAKERS, Berry. A Simple Publicly Verifiable Secret Sharing Scheme and Its Application to Electronic Voting. In: *CRYPTO* [online]. 1999 [cit. 2022-01-16].

22. CHAUM, David; PEDERSEN, Torben P. Wallet Databases with Observers. In: *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology* [online]. Springer-Verlag, 1992, s. 89–105 [cit. 2022-02-22]. CRYPTO '92. ISBN 3540573402.

23. FIAT, Amos; SHAMIR, Adi. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In: *Advances in Cryptology — CRYPTO' 86* [online]. Springer Berlin Heidelberg, 1987, s. 186–194 [cit. 2022-02-26]. Dostupné z DOI: 10.1007/3-540-47721-7_12.

24. YAO, Andrew C. Protocols for secure computations. In: *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)* [online]. 1982, s. 160–164 [cit. 2022-02-07]. Dostupné z DOI: 10.1109/SFCS.1982.38.

25. BOGDANOV, Dan et al. Students and Taxes: a Privacy-Preserving Study Using Secure Computation. *Proceedings on Privacy Enhancing Technologies* [online]. 2016-07, roč. 2016 [cit. 2022-03-15]. Dostupné z DOI: 10.1515/popets-2016-0019.

26. NAOR, Moni; PINKAS, Benny; SUMNER, Reuban. Privacy Preserving Auctions and Mechanism Design. In: *Proceedings of the 1st ACM Conference on Electronic Commerce* [online]. New York, NY, USA: Association for Computing Machinery, 1999, s. 129–139 [cit. 2022-02-18]. EC '99. ISBN 1581131763. Dostupné z DOI: 10.1145/336992.337028.

27. BOGETOFT, Peter et al. Secure Multiparty Computation Goes Live. In: [online]. 2009-02, sv. 5628, s. 325–343 [cit. 2022-02-18]. ISBN 978-3-642-03548-7. Dostupné z DOI: 10.1007/978-3-642-03549-4_20.

28. GOLDREICH, Oded. *Foundations of Cryptography* [online]. Cambridge University Press, 2004 [cit. 2021-12-04]. ISBN 978-0-521-11991-7. Dostupné z DOI: 10.1017/CBO9780511721656.

29. GOLDREICH, O.; MICALI, S.; WIGDERSON, A. How to Play ANY Mental Game. In: *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing* [online]. Association for Computing Machinery, 1987, s. 218–229 [cit. 2022-02-15]. STOC '87. ISBN 0897912217. Dostupné z DOI: 10.1145/28395.28420.

30. BEAVER, D.; MICALI, S.; ROGAWAY, P. The Round Complexity of Secure Protocols. In: *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing* [online]. Association for Computing Machinery, 1990, s. 503–513 [cit. 2022-02-15]. STOC '90. ISBN 0897913612. Dostupné z DOI: 10.1145/100216.100287.

31. LINDELL, Yehuda; PINKAS, Benny. A Proof of Security of Yao's Protocol for Two-Party Computation. *Journal of Cryptology*. 2008, roč. 22, s. 161–188.

32. DIFFIE, W.; HELLMAN, M.E. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *Computer* [online]. 1977, roč. 10, č. 6, s. 74–84 [cit. 2022-02-18]. Dostupné z DOI: 10.1109/C-M.1977.217750.

33. *Specification for the Advanced Encryption Standard (AES)* [Federal Information Processing Standards Publication 197]. 2001 [cit. 2022-03-12]. Dostupné z: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

34. ZAHUR, Samee; ROSULEK, Mike; EVANS, David. Two Halves Make a Whole: Reducing Data Transfer in Garbled Circuits using Half Gates. In: [online]. 2015-04, sv. 9057, s. 220–250 [cit. 2022-02-18]. ISBN 978-3-662-46802-9. Dostupné z DOI: 10.1007/978-3-662-46803-6_8.

35. KOLESNIKOV, Vladimir; SCHNEIDER, Thomas. Improved Garbled Circuit: Free XOR Gates and Applications. In: [online]. 2008-07, sv. 7, s. 486–498 [cit. 2022-02-18]. ISBN 978-3-540-70582-6. Dostupné z DOI: 10.1007/978-3-540-70583-3_40.

36. RABIN, Michael. How To Exchange Secrets with Oblivious Transfer. *IACR Cryptology ePrint Archive* [online]. 2005-01, roč. 2005, s. 187 [cit. 2022-02-10].

37. EVEN, Shimon; GOLDREICH, Oded; LEMPEL, Abraham. A Randomized Protocol for Signing Contracts. *Communications of the ACM* [online]. 1985-06, roč. 28, s. 637–647 [cit. 2022-02-15]. Dostupné z DOI: 10.1145/3812.3818.

38. RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM* [online]. 1978-02, roč. 21, č. 2, s. 120–126 [cit. 2022-03-10]. ISSN 0001-0782. Dostupné z DOI: 10.1145/359340.359342.

39. BRAKERSKI, Zvika; DÖTTLING, Nico. Two-Message Statistically Sender-Private OT from LWE. In: *Theory of Cryptography* [online]. Springer International Publishing, 2018, s. 370–390 [cit. 2022-03-20]. ISBN 978-3-030-03810-6. Dostupné z DOI: 10.1007/978-3-030-03810-6_14.

40. DOWSLEY, Rafael; GRAAF, Jeroen van de; MÜLLER-QUADE, Jörn; NASCIMENTO, Anderson C. A. Oblivious Transfer Based on the McEliece Assumptions. In: *Information Theoretic Security* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, s. 107–117 [cit. 2022-03-15]. Dostupné z DOI: 10.1007/978-3-540-85093-9_11.

41. BEN-OR, Michael; GOLDWASSER, Shafi; WIGDERSON, Avi. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing* [online]. Association for Computing Machinery, 1988, s. 1–10 [cit. 2022-01-29]. ISBN 0897912640. Dostupné z DOI: 10.1145/62212.62213.

42. GENNARO, Rosario; RABIN, Michael O.; RABIN, Tal. Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography. In: *Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing* [online]. Association for Computing Machinery, 1998, s. 101–111 [cit. 2022-02-11]. PODC '98. ISBN 0897919777. Dostupné z DOI: 10.1145/277697.277716.

43. WIESNER, Stephen. Conjugate Coding. *SIGACT News* [online]. 1983-01, roč. 15, č. 1, s. 78–88 [cit. 2022-03-12]. ISSN 0163-5700. Dostupné z DOI: 10.1145/1008908.1008920.

44. BENNETT, Charles; BRASSARD, Gilles. Quantum cryptography: Public key distribution and coin tossing. In: [online]. 1984, sv. 560, s. 175–179 [cit. 2022-03-01]. Dostupné z DOI: 10.1016/j.tcs.2011.08.039.

45. BENIOFF, Paul. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics* [online]. 1980-05, roč. 22, s. 563–591 [cit. 2022-03-12]. Dostupné z DOI: 10.1007/BF01011339.

46. RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM* [online]. 1978, roč. 21, č. 2, s. 120–126 [cit. 2022-03-13]. ISSN 0001-0782. Dostupné z DOI: 10.1145/359340.359342.

47. DIFFIE, W.; HELLMAN, M. New directions in cryptography. *IEEE Transactions on Information Theory* [online]. 1976, roč. 22, č. 6, s. 644–654 [cit. 2022-03-13]. Dostupné z DOI: 10.1109/TIT.1976.1055638.

48. STANDARDS, National Institute of; TECHNOLOGY. *Post-Quantum Cryptography* [online]. 2017 [cit. 2022-03-12]. Dostupné z: https://csrc.nist.gov/projects/post-quantum-cryptography.

49. MCELIECE, R. J. A Public-Key Cryptosystem Based On Algebraic Coding Theory. *Deep Space Network Progress Report* [online]. 1978-01, roč. 44, s. 114–116 [cit. 2022-03-14]. Dostupné z: https://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF.

50. ALAGI, Gorjan et al. Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Proces [online]. 2020-07 [cit. 2022-03-12]. Dostupné z DOI: 10.6028/NIST.IR.8309.

51. MOODY, Dustin et al. NIST Report on Post-Quantum Cryptography [online]. 2016-04 [cit. 2022-03-14]. Dostupné z DOI: 10.6028/NIST.IR.8105.

52. IBM. *IBM Unveils Breakthrough 127-Qubit Quantum Processor* [online]. 2021-11 [cit. 2022-03-17]. Dostupné z: https://newsroom.ibm.com/2021-11-16-IBM-Unveils-Breakthrough-127-Qubit-Quantum-Processor.

53. ARUTE, Frank et al. Quantum supremacy using a programmable superconducting processor. *Nature* [online]. 2019-10, roč. 574, s. 505–510 [cit. 2022-03-17]. Dostupné z DOI: 10.1038/s41586-019-1666-5.

54. ZHONG, Han-Sen et al. Quantum computational advantage using photons. *Science* [online]. 2020, roč. 370, č. 6523, s. 1460–1463 [cit. 2022-03-17]. Dostupné z DOI: 10.1126/science.abe8770.

55. HÄNER, Thomas; ROETTELER, Martin; SVORE, Krysta. Factoring using 2n+2 qubits with Toffoli based modular multiplication. *Quantum Information and Computation* [online]. 2016-11, roč. 17 [cit. 2022-03-14]. Dostupné z DOI: 10.26421/QIC17.7-8-7.

56. GIDNEY, Craig; EKERÅ, Martin. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum* [online]. 2021-04, roč. 5 [cit. 2022-03-14]. ISSN 2521-327X. Dostupné z DOI: 10.22331/q-2021-04-15-433.

57. BENNETT, Charles; BRASSARD, Gilles; CRÉPEAU, Claude; SKUBISZEWSKA, Marie-Hélène. Practical Quantum Oblivious Transfer. In: [online]. 1991-01, sv. 576, s. 351–366 [cit. 2022-03-13].

58. HILLERY, Mark; BUŽEK, Vladimír; BERTHIAUME, André. Quantum secret sharing. *Physical Review A* [online]. 1999, roč. 59, č. 3 [cit. 2022-03-11]. ISSN 1094-1622. Dostupné z DOI: 10.1103/physreva.59.1829.

59. NIELSEN, Michael A.; CHUANG, Isaac L. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. 10th. USA: Cambridge University Press, 2011. ISBN 1107002176.

60. DIRAC, P. A. M. A new notation for quantum mechanics. *Mathematical Proceedings of the Cambridge Philosophical Society* [online]. 1939, roč. 35, č. 3, s. 416–418 [cit. 2022-03-10]. Dostupné z DOI: 10.1017/S0305004100021162.

61. GREENBERGER, Daniel M.; HORNE, Michael A.; ZEILINGER, Anton. Going Beyond Bell's Theorem. In: *Bell's Theorem, Quantum Theory and Conceptions of the Universe* [online]. 1989, s. 69–72 [cit. 2022-03-10]. ISBN 978-94-017-0849-4. Dostupné z DOI: 10.1007/978-94-017-0849-4_10.

62. WOOTTERS, William K.; ZUREK, Wojciech. A single quantum cannot be cloned. *Nature*. 1982, roč. 299, s. 802–803.

63. BUŽEK, V.; HILLERY, M. Quantum copying: Beyond the no-cloning theorem. *Physical Review A* [online]. 1996-09, roč. 54, č. 3 [cit. 2022-03-12]. ISSN 1094-1622. Dostupné z DOI: 10.1103/physreva.54.1844.

64. SVOBODOVÁ, Hana. *Algoritmy pro sdílení tajemství* [online]. České vysoké učení technické v Praze, Fakulta informačních technologií, 2019 [cit. 2022-08-04]. Dostupné z: http://hdl.handle.net/10467/83204. Bakalářská práce.

65. *Open-Source Quantum Development* [online]. Qiskit [cit. 2022-03-30]. Dostupné z: https://qiskit.org.

Příloha A

Uživatelská příručka

A.1 Demo aplikace

A.1.1 Požadavky

Demo aplikace `demo.exe` je určena pro 64-bitový operační systém Windows 7 nebo vyšší. Pro spuštění aplikace je nutná knihovna `libcrypto-3-x64.dll`. Tato knihovna je součástí OpenSSL a nachází se i na přiloženém CD v adresáři spolu s demo aplikací. Zdrojový kód použité knihovny OpenSSL se taktéž nachází na přiloženém CD, a to ve složce `src` v podadresáři `openssl-3.0.2`. Dále je vyžadován balíček Microsoft Visual C++ 2015 Re-distributable. Návod na stažení a instalaci se nachází na adrese `https://www.microsoft.com/cs-CZ/download/details.aspx?id=48145`.

A.1.2 Spuštění

Spustitelná aplikace `demo.exe` se nachází na přiloženém CD ve složce `bin` v podadresáři `demo`. Aplikace se spouští z příkazové řádky a všechny její možnosti a parametry jsou vypsány níže.

--list
: Zobrazí nápovědu se stručným popisem jednotlivých funkcí.

-sh --split <secret> <k> <n> [outFilename]
: Rozdělí hexadecimální tajemství `<secret>` dle Shamirova (`<k>`,`<n>`) schématu. Nepovinný parametr `<outFilename>` určuje název pro generování názvů souboru pro uložení dílů, výchozí hodnota je `sh_share`.

-sh --reconstruct <share_1> ... <share_k>
: Ze zadaných dílů `<share_1>` až `<share_k>` rekonstruuje tajemství rozdělené Shamirovým sdílením tajemství a vypíše ho do konzole.

-sh --forge <share> <x_1> ... <x_k-1>
Podvrhne díl Shamirova schématu `<share>` pomocí zadaných x souřadnic bodů ostatních dílů `x_1` až `x_k-1`. Podvržený díl je platný jen při rekonstrukci tajemství spolu s díly, jejichž souřadnice x jsou shodné se zadanými.

-pe --split <secret> <k> <n> <sec> [outFilename]
Rozdělí hexadecimální tajemství `<secret>` dle Pedersenova (`<k>`,`<n>`) ověřitelného schématu. Bezpečnostní parametr `<sec>` určuje počet bitů prvočísla použitého pro generování závazků. Parametr `<outFilename>` je nepovinný a určuje název pro generování názvů souboru pro uložení dílů, výchozí hodnota je `ped_share`. Závazky jsou uloženy do souboru `ped_share_comm`.

-pe --reconstruct <commitment> <share_1> ... <share_k>
Ze zadaného souboru se závazky `<commitments>` a dílů `<share_1>` až `<share_k>` ověří platnost dílů a rekonstruuje tajemství rozdělené Pedersenovým ověřitelným schématem a vypíše ho do konzole.

-sc --keys <sec> <numKeys> <keyFilename>
Vygeneruje `<numKeys>` párů soukromý–veřejný klíč použitých v Schoenmakersově veřejně ověřitelném schématu. Parametr `<sec>` určuje počet bitů prvočísla použitého pro generování klíčů a klíče jsou uloženy do souborů s názvem odvozeným od `keyFilename`.

-sc --split <k> <n> <pubkey_1> ... <pubkey_n>
Rozdělí náhodné tajemství dle Schoenmakersova veřejně ověřitelného (`<k>`,`<n>`) schématu s pomocí veřejných klíčů `<pubkey_1>` až `<pubkey_n>`. Díly spolu s důkazy o korektnosti jsou uloženy do souborů s předponou `sc_share`, závazky jsou uloženy do souboru `sc_comm`.

-sc --decrypt <commitment> <share_1> ...<share_k> <pubkey_1> ... <pubkey_k> <privkey_1> ... <privkey_k>
Ověří platnost dílů `share_1` až `share_k` Schoenmakersova veřejně ověřitelného sdílení tajemství za pomoci závazků `commitment` a veřejných klíčů `pubkey_1` až `pubkey_k` a díly dešifruje soukromými klíči `privkey_1` až `privkey_k`. Dešifrované díly spolu s důkazy o korektnosti dešifrování jsou uloženy do souborů s předponou `dec_`.

-sc --reconstruct <commitment> <decShare_1> ... <decShare_k> <pubkey_1> ... <pubkey_k>
Ověří platnost dešifrovaných dílů `decShare_1` až `decShare_k` pomocí závazků `commitment` a odpovídajících původních dílů bez předpony `dec_`. Z dešifrovaných dílů je rekonstruováno tajemství rozdělené Schoenmakersovým schématem a vypsáno do konzole.

-bgw --sim1 <vote_1> ... <vote_5>
Simuluje bezpečné počítání více stran protokolem BGW a realizuje hlasování pěti voličů, přičemž hlas prvního z nich má dvojnásobnou váhu. Počítaná funkce je tedy $f(v_1, v_2, v_3, v_4, v_5) = 2v_1 + v_2 + v_3 + v_4 + v_5$. Parametry `vote_1` až `vote_5` reprezentují hlasy voličů a výsledek hlasování je vypsán do konzole.

-bgw --sim2 <input_1> ... <input_4>
Simuluje bezpečné počítání více stran protokolem BGW a počítá funkci $f(i_1, i_2, i_3, i_4) = (2i_1 + i_2)(3i_3 + i_4)$. Funkce obsahuje násobení, které vyžaduje komunikaci mezi účastníky. Parametry `input_1` až `input_4` reprezentují hexadecimální tajné vstupy a výsledek je vypsán do konzole.

A.2 Skript `hbb.py`

A.2.1 Požadavky

Skript `hbb.py` obsahuje implementaci kvantového schématu pro sdílení tajemství, které popsali Hillery, Bužek a Berthiaume. Nutnou podmínkou pro jeho spuštění je nainstalovaná vývojová sada Qiskit. Pro úspěšnou instalaci vývojové sady je nutný Python verze 3.6 nebo vyšší a lze ji instalovat pomocí správce balíčků `pip` příkazem

```
pip install qiskit
```

Více informací o instalaci vývojové sady se nachází na webové stránce projektu Qiskit `https://qiskit.org`.

A.2.2 Spuštění

Skript `hbb.py` se nachází na přiloženém CD ve složce `src` v podadresáři `qsss`. Soubor se spouští z příkazové řádky a má tři volitelné parametry. Základní spuštění souboru se provede příkazem:

```
python hbb.py
```

V této podobě proběhne simulace s výchozími parametry, které je možné změnit pomocí následujících volitelných přepínačů:

-s, --secret Hodnota tajného qubitu, povolené hodnoty jsou 0, resp. 1 a reprezentují stav $|0\rangle$, resp. $|1\rangle$. Výchozí hodnota je 1.

-r, --runs Počet opakování simulace obvodu. Výchozí hodnota je 100.

--export_png Přepínač pro vygenerování samostatného grafického schématu obvodu. Schéma je uloženo do souboru `hbb_circuit.png`

Příloha **B**

Obsah přiloženého CD

- `DP-Svobodova-Hana-2022.pdf` ..... text práce ve fromátu PDF
- `ctiMe.txt` ..... stručný popis obsahu CD
- `bin` ..... zkompilované verze implementace
  - `vsssLib` ..... zkompilovaná knihovna vsssLib
    - `doc` ..... dokumentace knihovny vsssLib
    - `include` ..... hlavičkové soubory knihovny vsssLib
  - `demo` ..... spustitelná demo aplikace
- `src` ..... zdrojové kódy implementace
  - `vsssLib` ..... zdrojové kódy knihovny vsssLib
  - `demo` ..... zdrojové kódy demo aplikace
  - `qsss` ..... zdrojové kódy kvantového sdílení tajemství
  - `openssl-3.0.2` ..... zdrojové kódy knihovny OpenSSL
- `text` ..... zdrojová forma práce ve formátu LaTeX