FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Možnosti aplikace důkazu s nulovými znalostmi v praxi
Student:	Kryštof Nevšímal
Vedoucí:	Ing. Miroslav Pospíšek, CSc.
Studijní program:	Informatika
Obor / specializace:	Informační bezpečnost 2021
Katedra:	Katedra informační bezpečnosti
Platnost zadání:	do konce letního semestru 2024/2025

Pokyny pro vypracování

V kryptografii je důkaz s nulovými znalostmi (Zero knowledge proof/protocol) metoda, pomocí které může jedna strana (dokazující) prokázat druhé straně (ověřovateli), že dané tvrzení je pravdivé, a přitom se vyhnout tomu, aby ověřovateli předala jakékoli informace nad rámec faktu o pravdivosti výroku, tedy ověřovatel vlastní data neuvidí.
I když je tato metoda známá již několik desítek let, současné IT technologie (a hrozby) ukazují na nové možnosti jejího použití.
V poslední době se tomuto tématu věnují renomované technologické společnosti i nově vznikající firmy (tzv. startupy).

1. Vysvětlete základní principy zero-knowledge proof/protocol.
2. Na základě dostupných informací vypracujte přehled o současném stavu.
3. Diskutujte možnosti praktické aplikace a kombinace s jinými kryptografickými metodami.
4. S využitím open-source nástrojů sestavte ukázkovou aplikaci využívající ZKP v oblasti autentizace či autorizace.
5. Zhodnoťte dosažené výsledky.

Elektronicky schválil/a prof. Ing. Róbert Lórencz, CSc. dne 30. ledna 2024 v Praze.

Bakalářská práce

MOŽNOSTI APLIKACE DŮKAZU S NULOVÝMI ZNALOSTMI V PRAXI

Kryštof Nevšímal

Fakulta informačních technologií
Katedra informační bezpečnosti
Vedoucí: Ing. Miroslav Pospíšek, CSc.
16. května 2024

Odkaz na tuto práci: Nevšímal Kryštof. *Možnosti aplikace důkazu s nulovými znalostmi v praxi.* Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií, 2024.

Obsah

Seznam obrázků

Seznam tabulek

Chtěl bych poděkovat především svému vedoucímu Ing. Miroslavu Pospíškovi, CSc. za ochotu, odborné vedení a konzultace.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl 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, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

V Praze dne 16. května 2024

Abstrakt

Tato bakalářské práce se zabývá problematikou důkazu s nulovými znalostmi. Jedná se o metodu, která umožňuje dokázat pravdivost tvrzení bez zveřejňování konkrétní informace v něm obsažené. Práce vysvětluje základní principy a vytváří přehled o současném stavu. Z konkrétních protokolů jsou blíže popsány schémata SNARK a STARK, u kterých je provedeno vzájemné porovnání. Na zmíněné protokoly navazuje diskuze k možným oblastem využití. Mezi oblasti patří autentizace, na kterou navazují dvě praktické aplikace. První demonstruje použití protokolu STARK za využití open-source knihovny Winterfell. Druhá uvádí protokol SRP, který je určený konkrétně k autentizaci. S ohledem na praktické aplikace je provedena analýza, která se zabývá podstatnými vlastnostmi protokolů.

Klíčová slova důkaz s nulovými znalostmi, autentizace, SNARK, STARK, Winterfell, SRP

Abstract

This bachelor thesis focuses on a zero-knowledge proof. It is a method that allows to prove the truth of a statement without disclosing the specific information contained in it. The thesis explains the basic principles and provides an overview of the current state. Among the concrete protocols, the SNARK and STARK schemes are described in more detail and a comparison is made between them. These protocols are followed by a discussion of possible areas of application. The areas include authentication, which is followed by two practical applications. The first demonstrates the use of the STARK protocol using the open-source Winterfell library. The second introduces the SRP protocol, which is designed specifically for authentication. Based on the practical applications, an analysis is performed that focuses on the essential properties of the protocols.

Keywords zero knowledge proof, authentication, SNARK, STARK, Winterfell, SRP

Seznam zkratek

FRI	Fast Reed-Solomon Interactive Oracle Proofs of Proximit
LDE	Low Degree Extension
SNARK	Succinct Non-interactive Argument of Knowledge
SRP	Secure Remote Password
STARK	Scalable Transparent Argument of Knowledge
ZKP	Zero Knowledge Proof
ZKPP	Zero Knowledge Password Proof
zkSNARK	Zero Knowledge Succinct Non-interactive Argument of Knowledge
zkSTARK	Zero Knowledge Scalable Transparent Argument of Knowledge

Úvod

V posledních letech je čím dál tím větší důraz kladen na kybernetickou bezpečnost. Kybernetická bezpečnost chrání digitální informace, zařízení a aktiva. To zahrnuje osobní údaje, účty, soubory, fotky i peníze. Vzhledem k růstu digitalizace v dnešní společnosti je kybernetická bezpečnost prakticky nepostradatelná. Existuje mnoho způsobů a metod, pomocí kterých této bezpečnosti můžeme pomoci. Jednou z možností je právě důkaz s nulovými znalostmi (ZKP - zero knowledge proof).

Tato metoda omezuje množství zveřejněných informací, a tím zmenšuje obsah dat, které může útočník použít v dalších fázích útoku. Této vlastnosti se dá využít třeba v autentizaci, kde již existují různé protokoly postaveny na tomto principu, například Secure Remote Password protokol z konce 20. století.

Důkaz s nulovými znalostmi zajišťuje větší soukromí, kdy se citlivé informace (tajemství) nevystavují riziku. Je proto vhodné použít důkaz s nulovými znalostmi v případech zveřejňování citlivých informací. Jedná se například o oblasti elektronický volební systém, prokázání výše příjmu při zažádání o hypotéku či držení průkazu dávající nám určité oprávnění.

Téma jsem si vybral, protože o důkazu s nulovými znalostmi sice existují práce, ale jsou zejména teoretické a zaměřené na konkrétnější oblast. Rád bych proto přinesl práci, která přinese ucelený a přehledný celek vysvětlující problematiku důkazu s nulovými znalostmi a důvody proč lze ZKP použít.

Struktura bakalářské práce je rozdělena na dvě části. Za prvé na teoretickou část, která se zabývá hlavními rysy důkazu s nulovými znalostmi. Uvedený teoretický popis je doplněný o ukázkové příklady, kde jsou vyzdviženy hlavní vlastnosti. Dále je popsán aktuální stav problematiky, kde je uvedeno jaké různé typy protokolů existují a v čem jsou odlišné. Na vybraných skupinách je dále ukázán princip, na kterým fungují. Od konkrétních protokolů se práce přesouvá k popisu různých oblastí, kde lze důkaz s nulovými znalostmi aplikovat.

Za druhé na praktickou část, kde se nachází dvě odlišné demonstrace využití ZKP v autentizaci. První praktická aplikace ukazuje použití všestranného schématu za pomoci open-source knihovny Winterfell a druhá aplikace předvádí protokol určený konkrétně k autentizaci. Na implementacích je založena analýza, která je zaměřena na hlavní vlastnosti protokolů.

Cíl práce

Hlavním cílem bakalářské práce je přinést ucelený a přehledný celek, který popíše koncept důkazu s nulovými znalostmi, zhodnotí jeho výhody a nevýhody a ilustruje jeho možné využití v reálném prostředí.

Teoretická část si klade tři cíle. Prvním cílem je popsat důkaz s nulovými znalostmi a jeho vlastnosti pro zorientování v tématu a pochopení hlavní myšlenky této kryptografické metody.

Druhým cílem je na základě dostupných informací poskytnout přehled o aktuálním stavu a informovat, že existují různé typy protokolů s různými vlastnostmi.

Třetím cílem je uvést různé oblasti, kde je vhodné tuto metodu použít, pro pochopení, v jak širokém spektru situací lze ZKP aplikovat.

Praktická část si klade za cíl demonstrovat, jak rámcově vypadá implementace důkazu s nulovými znalostmi v praxi, a analyzovat důležité vlastnosti pro praktické používání, jako jsou čas a paměťová náročnost.

Kapitola 1

Důkaz s nulovými znalostmi

V této kapitole popíšu základní rozdělení ZKP. Uvedu příklady, na kterých vyzdvihnu, jak ZKP funguje a jaké má vlastnosti. Poté se zaměřím na širší spektrum různých protokolů a odkryji, jaké principy se za tím schovávají.

1.1 Základní rozdělení

V kryptografii je důkaz s nulovými znalostmi metoda, pomocí které může jedna strana (dokazující) prokázat druhé straně (ověřovateli), že dané tvrzení je pravdivé, a přitom se vyhnout tomu, aby ověřovateli předala jakékoliv informace nad rámec faktu o pravdivosti výroku, tedy ověřovatel vlastní data neuvidí.

S první zmínkou o ZKP se setkáváme v práci od Shafi Goldwasser, Silvio Micali a Charles Rackoff[1], kteří představili ZKP jako interaktivní protokol. Interaktivní protokoly jsou známé svými interakcemi mezi dvěma stranami. Většinou se jedná o dokazujícího, který přesvědčuje, a ověřovatele, který ověřuje. Klasická výměna mezi stranami se skládá ze tří kroků:

- dokazující pošle hodnotu,
- ověřovatel odešle nazpět výzvu,
- dokazující odpoví na výzvu.

Při každé interakci je ověřen dokazující s určitou pravděpodobností, která se každým opakováním zvyšuje. Důkaz pokračuje dalšími interakcemi, dokud není pravděpodobnost správnosti úsudku ověřovatele dostatečná. Hlavní princip interaktivního schématu je předveden na příkladu testování barvosleposti.

Nevýhodou interaktivního schématu je, že musí proběhnout znovu pro každého dalšího ověřovatele. To je jeden z důvodů, proč se v reálném světě pro interaktivní typ nenašlo mnoho efektivních aplikací. Do praxe potřebujeme protokol, který nemusí probíhat pro každého ověřovatele znovu a který nepotřebuje více interakcí. Proto později zavedli v práci Manuel Blum, Paul Feldman a Silvio Micali[2] podskupinu ZKP, a to neinteraktivní ZKP.

Zde dokazující vytvoří bez pomoci ověřovatele důkaz o pravdivosti svého tvrzení, který může ověřit kdokoliv. Nyní se oproti interaktivní verzi vymění pouze jediná zpráva mezi dokazujícím a ověřovatelem, a dokonce proces ověřování lze posunout i do budoucna. Tento princip je uveden na příkladu hry „Kde je Valda?“.

1.2 Vlastnosti ZKP

ZKP má tři základní vlastnosti[1], které musí splňovat:

- úplnost (completeness),
- spolehlivost (soundness),
- nulovou znalost (zero knowledge).

První vlastnost říká, že pokud dokazující zná tajemství, tak dojde bez pochyby k přesvědčení ověřovatele o jeho znalosti. Nemůže se stát, že by dokazující znal tajemství, a navzdory tomu se mu nepovedla prokázat jeho znalosti.

Druhá popisuje situaci, kdy se dokazující, který nedisponuje znalostí tajemství, snaží přesvědčit ověřovatele. Pravděpodobnost, že ověřovatel přijme důkaz, se musí blížit nule.

Třetí přidává nulovou znalost. To znamená, že proběhlá interakce mezi dokazujícím a ověřovatelem neumožňuje získat více informací než samotnou pravdivost tvrzení.

1.3 Ukázkové příklady

V této části přiblížím myšlenku zero knowledge proof na jednoduchých příkladech. Nejdříve vezmu příklad „Barvoslepého kamaráda“ a popíšu interaktivní ZKP. Poté na příkladu hry „Kde je Valda?“ poukážu na rysy neinteraktivního ZKP. Příklady jsou převzaty od autorky Nicole Zhu[3].

1.3.1 Barvoslepý kamarád

Mějme Alici a Boba, kde Bob je barvoslepý. K tomu máme dva míčky lišící se pouze barvou: jeden červený a druhý zelený. Bobovi se míčky zdají stejné a nevěří Alici, že se dají rozlišit. Alice mu chce dokázat, že míčky jsou různobarevné, ale nic jiného.

Bob si vezme oba míčky a umístí si je za záda. Poté jeden míček vezme a ukáže Alici. Vrátí si ho za záda a se stejnou pravděpodobností rozhodne, který míček ze dvou vezme. Znovu ukáže Alici a zeptá se: „Vyměnil jsem míček?“.

Buď byly míčky různobarevné a pravděpodobnost, že Alice odpoví správně, je 100 %, nebo měly stejnou barvu a pravděpodobnost, že Alice zvolí náhodně ten správný, je 50 %. Taková pravděpodobnost Bobovi nestačí, a proto provede tyto kroky vícekrát. Například při páté správné odpovědi je šance, že jsou míčky stejné barvy, rovna 2^{-5} (3,125 %). Tato interakce probíhá do té doby, dokud není Bob spokojený s pravděpodobností, která se každým krokem zvyšuje.

Na příkladu lze ukázat splnění vlastností ZKP. Mějme situaci, kdy byla interakce mezi Alicí a Bobem provedena vícekrát, například stokrát. Kdyby Alice pouze hádala, jestli byl míček prohozen, pravděpodobnost úspěchu by byla rovna 2^{-100}. To se považuje za nesmírně nepravděpodobné, a tudíž se Alici, která neví, jestli byl míček prohozen, nepodaří přesvědčit Boba o různobarevnosti míčků (spolehlivost).

Na druhou stranu, pokud Alice stokrát odpoví správně, nakonec se jí podaří přesvědčit Boba na základě pravděpodobnosti, že mezi míčky je rozdíl.

Výše uvedený důkaz také disponuje nulovou znalostí, protože Bob se nikdy nedozví, který míček je zelený a který červený. Ve skutečnosti nezíská žádné znalosti o tom, jak míčky rozlišit. Pouze si ověří, že Alice má určitou znalost.

1.3.2 Kde je Valda?

Mějme Alici a Boba soutěžící, kdo najde Valdu jako první. Dostaneme se do situace, kdy jeden z dvojice najde Valdu. Chceme prokázat, že jsme ho našli, ale na druhou stranu neukázat jeho přesnou lokaci. Na to se nabízí zero knowledge proof.

Například Bob našel Valdu a aby mu Alice uvěřila, zkusí jí to dokázat. V této situaci se nabízejí dvě možné řešení. Buď může Bob Valdu vystřihnout z obrázku a ukázat Alici výstřižek, nebo může vystřihnout obrys Valdy do středu jiného většího papíru a následně Alici ukázat původní obrázek překrytý papírem. V obou případech Alice ověří, zdali Bob našel Valdu.

I zde můžeme najít všechny vlastnosti ZKP. Pokud Bob Valdu našel může to jednoduše dokázat (úplnost).

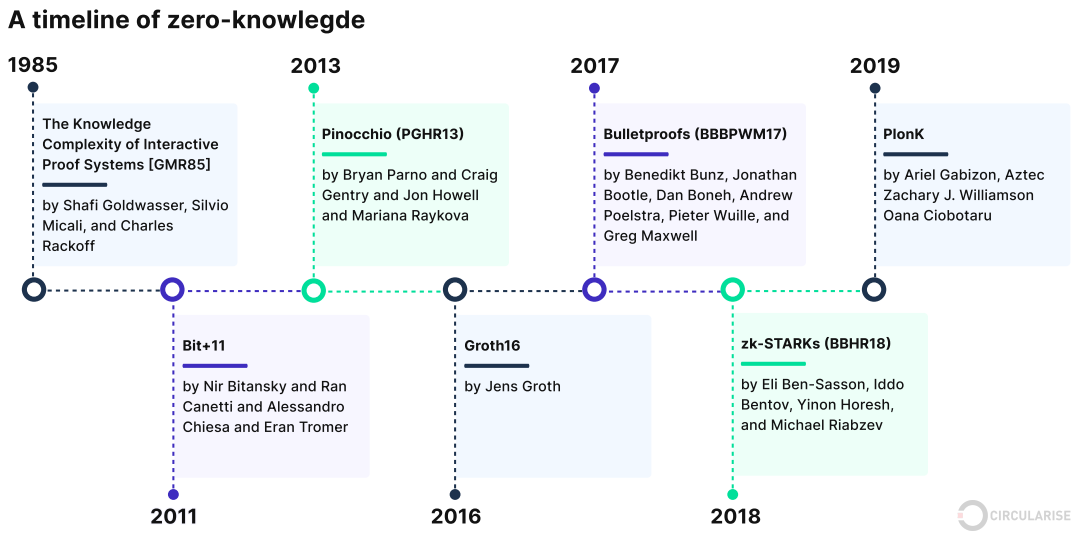
Pokud Valdu nenašel, Alice z náhodného výstřižku pozná, že Bobův výrok, že našel Valdu, je nepravdivý. Zároveň šance, že Bob udělá náhodný výstřižek a trefí se, je extrémně nepravděpodobná (spolehlivost).

Alice se z důkazu pouze dozví, jestli měl Bob pravdu. Nedostane žádné informace navíc, například informaci o výskytu postavy (zero knowledge).

1.4 Historický vývoj

Na začátku 80. let minulého století se začaly objevovat první zmínky o ZKP. Vyvrcholilo to roku 1985 zveřejněním práce *The knowledge complexity of interactive proof systems* od tří autorů Shafi Goldwasser, Silvio Micali a Charles Rackoff[1]. V této práci byl definován pojem ZKP v kontextu interaktivních protokolů.

O pár let později v roce 1988 definovali autoři Manuel Blum, Paul Feldman a Silvio Micali neinteraktivní ZKP. Po tomto příspěvku do teorie ZKP nastala delší odmlka, kdy nepřišly zásadní průlomy. ZKP se do středu zájmu dostal až po vzniku stručného neinteraktivního argumentu znalosti (SNARK - Succinct Non-interactive ARgument of Knowledge) roku 2011. Termín SNARK definovala čtveřice Nir Bitansky, Ran Canetti, Alessandro Chiesa a Eran Tromer v práci *From Extractable Collision Resistance to Succinct Non-Interactive Arguments of Knowledge, and Back Again*[4]. SNARK se do popředí dostal, díky malé velikosti důkazu a krátkému času jeho ověření.



Obrázek 1.1 Historický vývoj[5]

Od té doby se tvořilo velké množství různých implementací inspirovaných termínem SNARK. V roce 2013 publikovali Bryan Parno, Jon Howell, Craig Gentry a Mariana Raykova práci *Pinocchio: Nearly Practical Verifiable Computation*[6], v níž předvedli protokol Pinocchio. Jak název napovídá, jedná se o první téměř praktickou implementaci protokolu SNARK.[7] I když má Pinocchio v názvu, že je téměř praktický, byl aplikován společností Zcash. Jedná se o oblast kryptoměn, kdy je prvně ZKP použito pro ochranu soukromí. Přesněji zde slouží jako nástroj pro ověření zůstatku na účtu pomocí historie transakcí, ale bez odhalení zůstatku i transakcí.[8]

Mezi další důležité SNARK implementace patří Groth16 z roku 2016, kdy Jens Groth publikoval článek *On the size of pairing-based non-interactive arguments*[9]. Groth16 výrazně vylepšil výkonnost protokolu Pinocchio, a tím ho nahradil.[10]

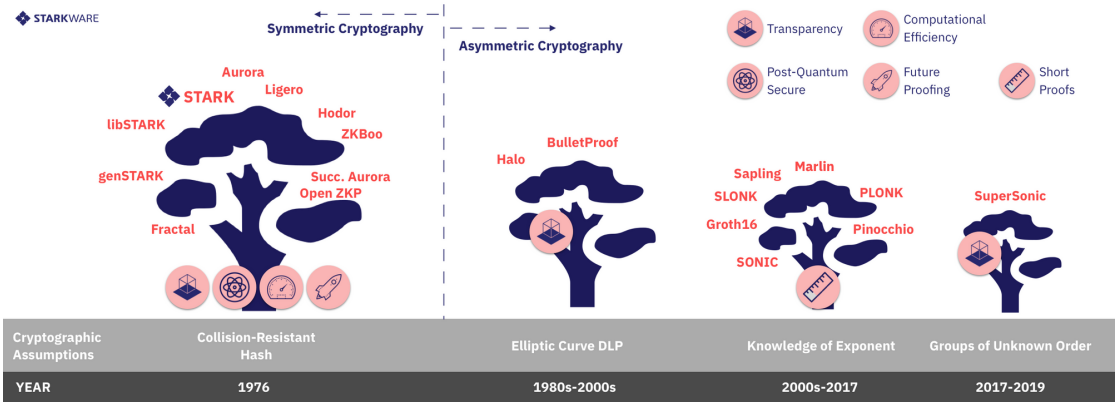
O tři roky později přichází v článku *PlonK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge*[11] protokol PlonK. Také se jedná o SNARK schéma, ale na rozdíl od Groth16 používá univerzální nastavení. Univerzální nastavení je vysvětleno v sekci 1.7.2.

Postupem času začali vznikat také alternativy ke schématu SNARK. Například v roce 2017 vzniklo schéma Bulletproofs *Bulletproofs: Short Proofs for Confidential Transactions and More*[12], které bylo prezentováno na konferenci o rok později, nebo v roce 2018 bylo zveřejněno schéma Scalable Transparent Argument of Knowledge (STARK) v práci *Scalable, transparent, and post-quantum secure computational integrity*[13]. Tyto schémata disponují transparentním nastavením, které je popsáno v sekci 1.7.2. Na druhou stranu mají oproti SNARK větší paměťové či časové nároky v jednotlivých krocích protokolu.

1.5 Typy protokolů

Vycházím z informací dostupných z článku *A Cambrian Explosion of Crypto Proofs* od autora Eli Ben-Sasson[14]. Vývoj ZKP protokolů trvá již přes 30 let, a proto existuje více směrů, kterými se protokoly vydaly. Největším rozdílným aspektem je, zdali bezpečnost protokolu stojí na symetrické či asymetrické kryptografii, viz obrázek 1.2.

Mezi typické symetrické patří protokoly postaveny na hashovacích funkcích. Předpokládáme, že jsou odolné vůči kolizím, pseudonáhodné a chovají se jako náhodné orákulum. Mezi tyto protokoly můžeme zařadit například známý STARK. Mezi ty asymetrické se řadí protokoly postaveny na problému diskrétního logaritmu v eliptických křivkách (BulletProof), znalosti exponentu (Groth16) či skupinách neznámého řádu (SuperSonic).



Obrázek 1.2 Bezpečnostní předpoklady[14]

Rozdílnost zmíněných kryptografických předpokladů má několik dopadů:

- výpočetní náročnost (Computational Efficiency),
- postkvantovou odolnost (Post-Quantum Secure),
- velikost argumentu/důkazu (Short Proofs).

Výpočetní náročnost se naklání k symetrickým systémům, neboť pracují oproti asymetrickým s mnohem menšími prvky. Co se týče postkvantové odolnosti, Eli Ben-Sasson uvádí, že současné asymetrické systémy použité v této oblasti budou efektivně prolomeny pomocí kvantových počítačů s dostatečně velkým počtem qubitů. Zatímco symetrické jsou s případnou úpravou bezpečnostních parametrů pravděpodobně postkvantově odolné. Ohledně velikosti argumentu je výhodnější použít asymetrické protokoly jako je Groth16, který má důkaz za stejných bezpečnostních podmínek o velikosti v řádu stovek bytů, oproti symetrickým, které dosahují velikosti desítek kilobytů.

Na obrázku 1.2 jsou dále znázorněny symboly „Future Proofing“ a „Transparency“. Future proofing značí Lindyho efekt, který říká, že čím starší je technologie, tím větší je pravděpodobnost, že se bude nadále užívat. Transparentnost je vysvětlena v sekci 1.7.2.

Ve své práci se dále budu zabývat schématy SNARK a STARK, které již nalezly významné uplatnění v praktických aplikacích.

1.6 Kryptografické předpoklady

V této sekci uvedu podstatu dvou hlavních kryptografických předpokladů, na kterých jsou postaveny konkrétní protokoly ZKP.

1.6.1 Diskrétní logaritmus

Problém diskrétního logaritmu je jeden ze základních matematických problémů v kryptografii. Diskrétní logaritmus je logaritmus operující v konečné grupě. Ještě lépe v cyklické grupě, to znamená, že existuje generátor, který pomocí opakování operace dokáže vygenerovat všechny prvky z grupy. Tímto je výpočetní složitost problému výrazně zvýší.

Mějme tedy konečnou grupu G, prvek $a \in G$ a $b \in P(a)$, kde $P(a)$ značí všechny možné prvky generované bodem a. Poté problém diskrétního logaritmu je nalezení nejmenšího kladného x řešící rovnici $a^x = b$.[15]

1.6.2 Odolnost vůči kolizi u hashovacích funkcí

Dalším častým bezpečnostním předpokladem je odolnost hashovacích funkcí vůči kolizi. Bezkoliznost je 2 řádů. Bezkoliznost 1. řádu nám říká, že hashovací funkce je odolná vůči kolizi 1. řádu, pokud je výpočetně nemožné nalezení dvou různých zpráv x a y tak, že $h(x) = h(y)$.

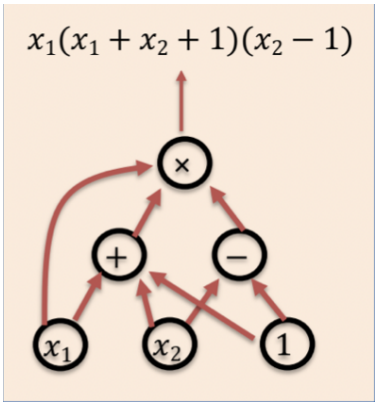
Odolná vůči kolizi 2. řádu je hashovací funkce, jestliže pro vzor x je výpočetně nemožné najít vzor y takový, že $x \neq y$ a $h(x) = h(y)$. Při splnění těchto vlastností je hashovací funkce navíc používaná jako náhodné orákulum.[16]

1.7 Charakteristiky zero knowledge protokolů

Dále abychom mohli protokoly popisovat a mezi sebou porovnávat, je potřeba vysvětlit pojmy aritmetický obvod, přípravná fáze, časová a paměťová náročnost a stručnost.

1.7.1 Aritmetický obvod

Aritmetickým obvodem je v mé práci míněn model pro výpočet polynomů. Podporuje operace sčítání a násobení na rozdíl od známějšího Boolean obvodu, který manipuluje s logickými hodnotami (true/false). Aritmetické obvody se skládají z dvou prvků: brány (gates) a spoje (wires). Brány, do kterých nevede spoj, jsou nazývány jako vstup a zbylé značí operaci sčítání či násobení. Tyto brány jsou spojeny pomocí spojů, a tím tvoří aritmetické funkce. Tudíž aritmetický obvod má vstupy. Z každého vstupu vedou spoje, které určují jaké operace budou se vstupním prvkem provedeny. Tímto způsobem se provede n operací, až se dojde k výsledné hodnotě.[17]



Obrázek 1.3 Aritmetický obvod[17]

1.7.2 Přípravná fáze

Přípravná fáze zahrnuje nutnou práci provedenou před začátkem samotné interakce mezi dokazujícím a ověřovatelem. Většinou se vytvoří aritmetický obvod popisující problém, z kterého se spočítají veřejné parametry pro dokazujícího a ověřovatele. V této části vycházím z informací od Dana Boneha[18]. Obecně máme tři typy:

- důvěryhodné nastavení,
- důvěryhodné ale univerzální nastavení,
- transparentní nastavení.

V důvěryhodném nastavení se pro konkrétní obvod spočítají veřejné parametry za pomoci náhodných bitů. Tyto náhodné bity musí být po celou dobu zachovány v tajnosti, neboť při úniku by mohl dokazující generovat falešné důkazy. Formálněji to lze popsat jako funkci S, která pomocí obvodu C a náhodných bitů r vytvoří veřejné parametry S_p a S_v.

$$S(C, r) \rightarrow PublicParameters(Sv, Sp)$$

V univerzálním nastavení náhodné bity nejsou závislé na obvodu. Použijí se jednou pro inicializaci, kdy se vytvoří veřejné parametry pp. Pomocí těchto veřejných parametrů se tvoří deterministicky další parametry již pro konkrétní obvod. Neboť se jedná o deterministický postup, není zde nutné nic držet v tajnosti a každý si je může spočítat.

$$S_{init}(\lambda, r) \to pp, S_{index}(C, pp) \to (S_v, S_p)$$

V transparentním nastavení se pro tvorbu veřejných parametrů používá pouze konkrétní obvod a není zde nutnost skrývat jakékoliv tajemství.

$$S(C) \to pp$$

Nejbezpečnější je používat transparentní nastavení, v kterém se nemusí nic skrývat. V univerzálním se musí vytvořit náhodné bity jednou a ty zachovat v tajnosti. Nejméně bezpečným je důvěryhodné nastavení, kde je více skupin náhodných bitů, které se musí zachovat skryté. Cena za transparentní nastavení se ale promítne na výpočetních a paměťových náročností algoritmu.

1.7.3 Časová a paměťová náročnost

Mezi další důležité vlastnosti patří časová a paměťová složitost jednotlivých implementací. Nejčastějším nástrojem pro zaznamenání složitostí se používá asymptotická horní mez, protože nám poskytuje vhled do toho, jak se protokol chová ve vztahu k růstu jeho vstupů. Samotné složitosti můžeme popsat následovně.

Pro časovou složitost o vstupu velikosti n uvažujme, kolik maximálně instrukcí protokol vykoná pro všechny možné vstupy o velikosti n.

Pro paměťovou složitost o vstupu velikosti n uvažujme, kolik maximálně paměťových buněk protokol využije přes všechny možné vstupy o velikosti n.

Asymptotickou horní mez pro tyto složitosti definujme dle definice z [19]:

▶ **Definice 1.1.** *Asymptotická horní mez O: Mějme dvě funkce f a g a bod $a \in \mathbb{R}$ takový, že a je hromadným bodem množiny $D_f \cap D_g$ (Definiční obory funkcí) a existuje okolí V_a splňující $V_a \cap D_f = V_a \cap D_g$. Řekneme, že funkce f je asymptoticky shora omezená funkcí g pro x jdoucí k a, symbolicky*

$$f(x) = O(g(x)) \text{ pro } x \to a,$$

právě když existuje kladná konstanta $c \in \mathbb{R}$ a okolí U_a bodu a tak, že pro všechna $x \in (U_a \cap D_f \cap D_g) \setminus \{a\}$ platí

$$|f(x)| \leq c \cdot |g(x)|.$$

1.7.4 Stručnost

Stručnost v kontextu ZKP se váže k velikosti důkazu a času jeho zpracování. Aby byl důkaz stručný musí být výrazně menší než velikost samotného vstupního problému. Například zdroj [18] se zmiňuje o logaritmické velikosti vůči vstupu.

Díky této vlastnosti lze efektivně ověřit důkaz v krátkém čase a s nízkými nároky na výpočetní výkon. U některých SNARK protokolů se dosahuje i konstantní složitosti, což patří mezi jejich silné stránky.

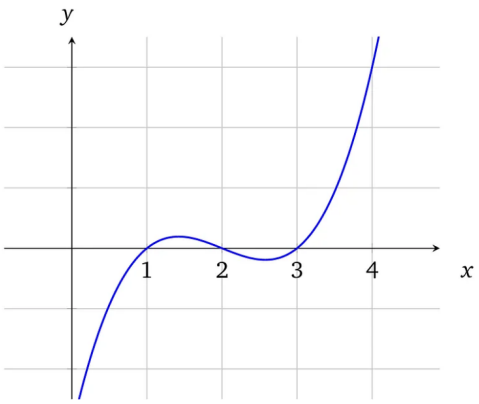
Kapitola 2
Schémata ZKP

V této kapitole popíšu, jak jednotlivé skupiny schémat fungují. Začnu obecným schématem a dále se budu zabývat protokoly SNARK a STARK.

2.1 Obecné schéma

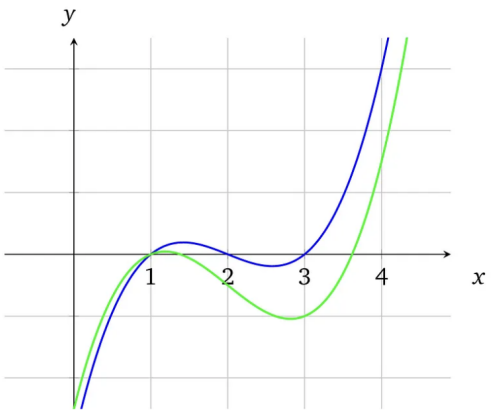
Protokol s nulovými znalostmi lze sestavit konkrétně pro autentizaci nebo pro věkové ověření. Tyto protokoly však nelze aplikovat na všechny možné problémy, neboť fungují různě v konkrétních případech. Proto existují obecná schémata. Tato schémata pracují většinou nad aritmetickým obvodem, protože je práce s polynomiálními funkcemi výhodná. V následujících odstavcích vysvětlím proč.

Mějme polynom $f(x) = x^3 - 6x^2 + 11x - 6$, viz obrázek 2.1.



Obrázek 2.1 Polynom $f(x) = x^3 - 6x^2 + 11x - 6$[20]

Polynomy disponují vlastností, že pokud máme dva různé polynomy řádu nejvýše d, mohou se protínat maximálně v d bodech. Řád polynomu určuje nejvyšší exponent x, který je v tomto případě 3. Vezměme tedy druhý lehce odlišný polynom $q(x) = x^3 - 6x^2 + 10x - 5$ a zobrazme do stejného grafu zeleně, viz obrázek 2.2.



Obrázek 2.2 Polynomy $f(x)$ a $q(x)$[20]

Je vidět, že i při malých změnách se graf velmi liší. Tyto polynomy mohou mít společné pouze tři body. To se dá lehce zjistit přes výpočet rovnice sestavené pro hledání průsečíků.

$$x^3 - 6x^2 + 11x - 6 = x^3 - 6x^2 + 10x - 5$$

Pomocí úprav dostaneme výsledek.

$$x = 1$$

Tím jsme ověřili, jak je na grafu vidět, že mají pouze jeden společný bod, a to $x = 1$. Této vlastnosti využívá právě dokazující a ověřovatel. Zjednodušeně mohu uvést následovně. Dokazující se zaváže k polynomu. Ověřovatel pošle výzvu (hodnotu x). Dokazující spočte polynom v bodě x a vrátí výsledek. Nakonec ověřovatel zkontroluje, že výsledek odpovídá.

Pokud bychom měli x z rozsahu 1 až 10^{50}, tak počet lišících se bodů dvou polynomů řádu d je $10^{50} - d$. Tudíž pravděpodobnost, že se x trefí do jednoho z d společných bodů, je $\frac{d}{10^{50}}$.

Díky této vlastnosti jsou polynomy použity jako základ mnoha ZKP protokolů.[20]

Na aritmetickém obvodu lze pak postavit obecné schéma mezi dokazujícím a ověřujícím. Jedná se o trojici S (set), P, (prove) a V (verify), neboli:

- přípravná fáze,
- vytvoření důkazu,
- ověření důkazu.

V prvním kroku se pomocí aritmetického obvodu C připraví veřejné parametry pro dokazujícího S_p a ověřujícího S_v.

$$S(C) \rightarrow S_p, S_v$$

V druhém kroku dokazující vytvoří důkaz π. K vytvoření důkazu použije veřejný parametr S_p, dále veřejné vstupy x a také tajné w (witness).

$$P(S_p, x, w) \rightarrow \pi$$

Ve třetím kroku ověřovatel vezme důkaz π. Ten pomocí svého veřejného parametru a veřejných vstupů ověří.

$$V(S_v, x, \pi) \rightarrow true/false$$

Na těchto krocích se již protokoly liší, neboť je mnoho variant, které lze použít.[18]

2.2 Zk-SNARK

Succinct Non-interactive Argument of Knowledge je jedním ze schémat, které je stručné (succinct) a neinteraktivní (non-interactive). Samo o sobě ale není zatím zero knowledge. Upravením některých operací SNARK můžeme vlastnost zero knowledge přidat, a tím získáme zero knowledge SNARK (zkSNARK). Popis protokolu SNARK vychází ze zdrojů [18][21] od autora Dana Boneha.

2.2.1 Úvod do teorie

Před popisováním jednotlivých kroků protokolu přiblížím teorii, která je zásadní pro pochopení. Pro protokol SNARK se jedná konkrétně o závazky (commitments)[22], homomorfní šifrování[23], Fiat-Shamirovu transformaci[24] a bilineární párování[25].

2.2.1.1 Commitments

Jedná se o kryptografický protokol, který umožňuje odesílateli (sender) se zavázat k určité hodnotě či tvrzení. Tomu se říká závazek. Tento závazek zůstane skrytý vůči příjemci (receiver), až do doby jeho odhalení v další fázi protokolu. Obecně se uvádí dvě fáze:

- zavázání,
- odhalení.

Ve fázi zavázání posílající má zprávu m, náhodně si zvolí klíč k a zašifruje zprávu m pomocí klíče k a náhodnosti r. Výsledek nazveme commitment (závazek). Ten je následně poslán příjemci.

Ve fázi odhalení odesílatel pošle příjemci klíč k. Příjemce pomocí klíče závazek otevře a ověří pravost zavázání.

2.2.1.2 Homomorfní šifrování

Cílem homomorfní šifrování je umožnit matematické operace nad zašifrovanými daty $E(x)$ a získat výsledek, který odpovídá operacím provedeným nad nezašifrovanými daty (x). Operace, které je možné provádět, se liší na základě použité šifry. Existují takové, které podporují pouze určité operace, například sčítání

$$E(x) + E(y) = E(x + y)$$

nebo násobení

$$E(x) * E(y) = E(x * y).$$

Tyto šifry se označují částečně homomorfní. Dále jsou plně homomorfní šifry, které podporují všechny základní matematické operace.

2.2.1.3 Fiat-Shamirova transformace

Pomocí Fiat-Shamirovy transformace lze udělat z interaktivního protokolu neinteraktivní. Dosáhne se toho použitím simulovaných náhodných hodnot místo těch od ověřujícího.

Tyto simulované hodnoty vypočítá dokazující. Neboť nelze nechat dokazujícího si hodnoty vybrat, použije se náhodné orákulum. Náhodné orákulum je funkce, která má nepředvídatelný výstup, ale pro stejný vstup vydá stejný výsledek. Dokazující této funkce využije tak, že hodnoty, které by poslal ověřujícímu, použije jako vstup do orákula. Tím pádem lze později ověřit, že náhodné orákulum bylo správně použito a vybrané hodnoty byly skutečně náhodné.

2.2.1.4 Bilineární párování

Bilineární párování je funkce, která mapuje dva prvky z grup G_1, G_2 do třetí grupy G_T. Formálně lze definovat následovně:

▶ **Definice 2.1.** *Mějme tři cyklické grupy G_1, G_2 a G_T stejného řádu. Pak lze definovat bilineární párování jako funkci*

$$e : G_1 \times G_2 \to G_T$$

takovou, že pro všechny $u \in G_1$, $v \in G_2$, $a, b \in \mathbb{Z}$ platí

$$e(u^a, v^b) = e(u, v)^{ab}.$$

2.2.2 SNARK

K ukázce protokolu SNARK jsem si vybral závazkové schéma KZG[26], které používá bilineární párování. Toto schéma patří mezi ty jednodušší a zároveň vystihuje podstatu SNARK protokolů.

2.2.2.1 Přípravná fáze

Pro KZG je zprvu nutné určit grupu $G = 0, g, 2g, 3g, \ldots, (p-1)g$ řádu p, kde g je generátor a p prvočíslo. Dále se určí polynom $f \in F_p^{\leq d}$, kde $F_p(X)$ značí polynomy s koeficienty z konečného tělesa F_p (těleso s p prvky) a d je maximální řád polynomů, pro který chceme později dokázat, že známe $v = f(u)$ pro veřejné $u, v \in F_p$.

Nejdříve proběhne výpočet veřejných parametrů pp. Vybere se náhodné $\alpha \in F_p$ a spočítají se

$$pp = (H_0 = g, H_1 = \alpha g, H_2 = \alpha^2 g, \ldots, H_d = \alpha^d g).$$

Dále se musí α smazat, neboť se jedná o důvěryhodné nastavení prostředí a jeho kompromitace by vedla k možnosti tvorby falešných důkazů.

2.2.2.2 Vytvoření důkazu

Dalším důležitým krokem je tvorba závazku k funkci. K funkci f se vytvoří závazek $com_f = f(\alpha)g \in G$. Velikost com_f je pouze jeden element v grupě G. Jelikož parametr α není dostupný, použije se

$$com_f = f_0 \cdot H_0 + \cdots + f_d \cdot H_d.$$

Pro vysvětlení to lze rozepsat jako

$$com_f = f_0 \cdot g + f_1 \cdot \alpha g + f_2 \cdot \alpha^2 g + \cdots,$$

kde vytknutím g získáme $f(\alpha)$. Závazek com_f se zašle ověřovateli.

Cílem schématu KZG je dokázat, že existuje x, pro které platí, že $f(x) = v$. To lze rozepsat jako

$$f(u) = v \Leftrightarrow u \text{ je kořenem } l = f - v$$
$$\Leftrightarrow (X - u) \text{ dělí } l \Leftrightarrow \text{ existuje } q \in F_p, \text{ takové že } q(X) \cdot (X - u) = f(X) - v.$$

Dokazující pošle k funkci $q(x)$ závazek com_q ověřovateli.

2.2.2.3 Ověření důkazu

Ověřovatel si ověří pravost důkazu pomocí vztahu

$$(\alpha - u) \cdot com_q = com_f - v \cdot g.$$

Pro tento výpočet se použije bilineární párování a vlastnost KZG homomorfního sčítání. Ověřovatel musí znát pouze g a $H_1 = \alpha \cdot g$ z veřejných parametrů pp. Označme $g_a = \alpha \cdot g, g_u = u \cdot g$ a $g_v = v \cdot g$. Potom můžeme ekvivalenci přepsat na

$$e(com_q, g_a) - e(com_q, g_u) = e(com_f, g_v).$$

2.2.3 Poslední kroky k zk-SNARK

Uvedeným postupem jsme dostali částečné řešení. Ze schématu KZG lze ještě zjistit něco o polynomu $f(x)$, tudíž protokol je zatím SNARK a ne zkSNARK. Tuto vlastnost lze opravit jednoduchým trikem. Trik se zakomponuje do části, kde se dokazovalo

$$q(X) \cdot (X - u) = f(X) - v.$$

Do levé strany rovnice se přičte náhodný polynom $r(X)$, který se jako $q(X)$ vynásobí $(X - u)$

$$q(X) \cdot (X - u) + r(X) \cdot (X - u) = f(X) - v.$$

Tímto se zachovají kořeny polynomu a zároveň se získá vlastnost zero knowledge.[27]

Také v uvedeném provedení byla potřebná interakce s ověřovatelem. To lze vyřešit imitací ověřovatele pomocí Fiat-Shamirovy transformace vysvětlené výše.

2.3 Zk-STARK

V roce 2018 byl představen protokol Scalable Transparent Argument of Knowledge, neboli škálovatelný transparentní argument znalosti. Jedná se o další významné schéma a mezi jeho hlavní pozitiva patří transparentnost a škálovatelnost. Transparentnost již byla vysvětlena, a tak stačí přiblížit pojem škálovatelnost. Tím se myslí, že lze pomocí nastavení parametrů dynamicky měnit vlastnosti důkazu jako je doba generování a ověřování, velikost a úroveň bezpečnosti. Při popisu protokolu STARK vycházím z informací od tvůrců protokolu zkSTARKs StarkWare.[28]

2.3.1 Úvod do teorie

Před popisováním jednotlivých kroků přiblížím teorii, která je zásadní pro pochopení protokolu. Pro protokol STARK se jedná konkrétně o Low degree extension (LDE)[29], Merkle tree[30] a Fast Reed-Solomon Interactive Oracle Proofs of Proximity (FRI)[28].

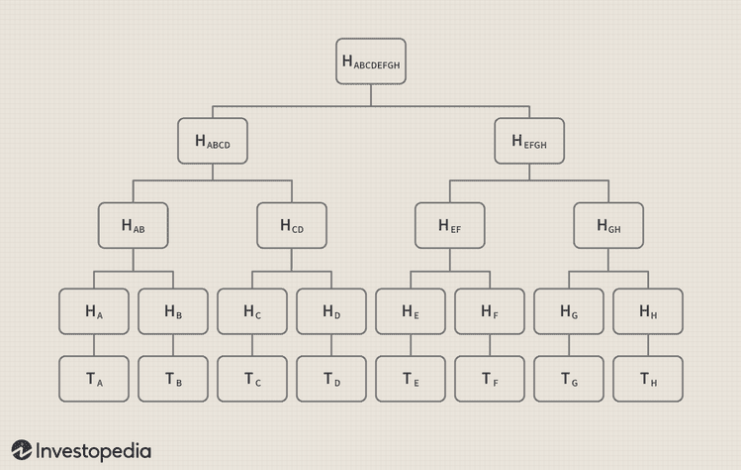
2.3.1.1 Low Degree Extension

Jedná se o matematický trik, kdy rozšíříme množinu bodů, na které je polynom definován. Pomocí tohoto triku zmenšíme pravděpodobnost kolize dvou polynomů. Formálně lze definovat následovně:

▶ **Tvrzení 2.2** (Rozšíření nízkého stupně). *Nechť F je libovolné těleso a $m \in \mathbb{N}$. Nechť $H \subseteq F$ je množina velikosti h. Pak každá funkce $f : H^m \to F$ může být jednoznačně rozšířena na funkci $\bar{f} : F^m \to F$, kde $\bar{f}$ je m-variantní polynom s jednotlivým stupněm nejvýše $h-1$ v každé proměnné. $\bar{f}$ nazýváme rozšířením nízkého stupně funkce f.*

2.3.1.2 Merkle Tree

Merkle tree je datová struktura ve tvaru binárního stromu. Nejnižší úroveň stromu se skládá z hashů prvků a každá další úroveň je vždy tvořena hashem dvou prvků z nižší úrovně. Viz obrázek 2.3.



Obrázek 2.3 Fáze zavázání a odhalení[30]

Jedná se o strukturu, pomocí které se zavážeme k množině prvků, v našem případě $T_A, ..., T_H$. Hash v kořenu struktury nazveme kořenový hash. Ten v sobě obsahuje informaci o každém prvku použitém pro stavbu Merkle tree.[30]

Ověření, že vybraný prvek byl v zavázané množině, proběhne pomocí autentizační cesty. Obecně dokazující pošle vždy sousední hash potřebný k stavbě následujících na cestě mezi vybraným prvkem a kořenem stromu.

2.3.1.3 Fast Reed-Solomon Interactive Oracle Proofs of Proximity

Protokol FRI se používá pro ověření, že daná racionální funkce je polynom. Aby byl výpočet efektivní, stačí, že je racionální funkce blízko k polynomu malého stupně. To znamená, že dva polynomy mají malý počet lišících se bodů. Protokol se skládá z opakujících se tří kroků:

- dokazující obdrží náhodné číslo β,
- použije se FRI operátor, který sníží stupeň polynomu,
- odešle se závazek.

Myšlenka FRI protokolu je, že postupem operací se sníží velikost stupně polynomu až na stupeň 0, kdy každý jeden cyklus sníží stupeň na polovinu.

Mějme polynom

$$p_o(x) = 5x^5 + 3x^4 + 7x^3 + 2x^2 + x + 3$$

a obdržené náhodné číslo β. Polynom si rozdělíme na dvě části podle sudých a lichých mocnin. Ze sudých získáme

$$g(x^2) = 3x^4 + 2x^2 + 3$$

a z lichých

$$xh(x^2) = 5x^5 + 7x^3 + x.$$

Vzniklé polynomy dáme dohromady pomocí vztahu

$$p_1(x) = g(y) + \beta h(y).$$

Nový polynom se znovu převede do struktury Merkle tree a kořenový hash se odešle ověřovateli.

2.3.2 STARK

K protokolu STARK je i ukázán postup, jakým lze vytvořit polynom, na který následně protokol naváže. Jedná se konkrétně o postup k příkladu, kde máme posloupnost $a_{n+2} = a_{n+1}^2 + a_n^2$.

2.3.2.1 Převedení na polynom

Nejdříve se převede vstup, nad kterým se chce tvrzení dokázat, na polynom. Tento proces převedení můžeme rozdělit na tři kroky:

- generování vstupu,
- interpolace,
- rozšíření.

V prvním kroku se vygeneruje n čísel y_0 až y_n. Ke každému y_k se vybere x_k, kde $k \in 0, ..., n$, viz tabulka 2.1. V ukázkových příkladech předpokládejme, že $x_0, x_1, x_2, ...$ jsme vybrali jako $1, g, g^2, g^3, ...$

x	y
x_0	y_0
x_1	y_1
x_2	y_2
x_3	y_3
x_4	y_4

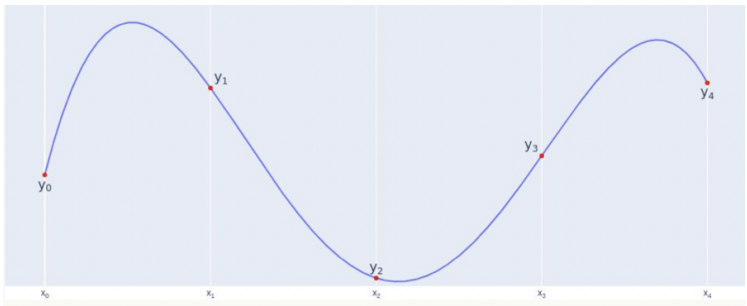
Tabulka 2.1 Znázorněné body x,y.

Tyto body vyznačíme do grafu, viz obrázek 2.4.

Obrázek 2.4 Zakreslení bodů x,y do grafu[28]

Pomocí vyznačených bodů provedeme interpolaci a nalezneme odpovídající polynom, viz obrázek 2.5, tak, že platí

$$f(x_i) = y_i \text{ pro všechna } i \in 0, ..., n.$$



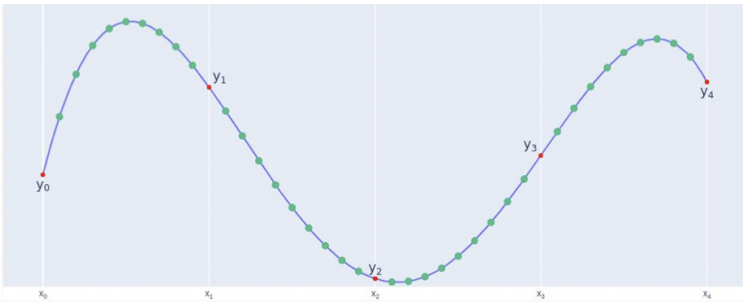
Obrázek 2.5 Interpolace[28]

Nakonec, když máme polynom, rozšíříme ho o počet bodů z množiny X (provedeme LDE). Tedy pro každý bod z množiny najdeme odpovídající y_i pomocí vztahu $y_i = f(j_i)$, kde $j \in X$. Znázorněno na Obrázku č. 2.6.

Pomocí LDE se šance, že při náhodném výběru bodu x z tělesa vyjde dvěma polynomům stejný bod y ($f(x) = y = g(x)$), zmenší, neboť rozšířením zvětšíme počet bodů a zachováme stupeň polynomu.

2.3.2.2 Přípravná fáze

Finální polynom f se zapíše do struktury Merkle tree a výsledný kořenový hash se pošle ověřovateli ve formě závazku.



Obrázek 2.6 Rozšíření[28]

2.3.2.3 Vytvoření důkazu

Momentálně je dokazující zavázán k funkci f. Nyní je ještě potřeba se omezit k tomu, co chce ověřovatel dokázat. Například pro naši posloupnost $a_{n+2} = a_{n+1}^2 + a_n^2$ zná x takové, že $a_0 = 1, a_1 = x, a_{1000} = 10290390$. Toho dosáhneme ve třech krocích redukce na jiné tvrzení, které implikuje pravdivost původních omezení.

Při prvním kroku převedeme omezení ze vstupu na naší funkci f. V našem případě

$$a_0 = 1 \rightarrow f(x) = 1, \text{ pro } x = x_0,$$
$$a_{1000} = 10290390 \rightarrow f(x) = 10290390, \text{ pro } x = x_{1000},$$
$$a_{n+2} = a_{n+1}^2 + a_n^2 \rightarrow f(g^2x) = f(gx)^2 + f(x)^2.$$

Kroky budu provádět na příkladu s a_0. Poté se každé omezení polynomu převede na kořen polynomů způsobem

$$f(x) - 1 = 0, \text{ pro } x = g^0.$$

Nyní pomocí kořenů vytvoříme z polynomů racionální funkci, tedy

$$\frac{a(x)}{b(x)}.$$

Zde se využije vlastnosti, že pokud je z kořenem $a(x)$, pak $(x - z)$ dělí $a(x)$. V našem případě jednu ze tří racionálních funkcí tvoří polynomy $f(x) - 1$ a $x - g^0$. Racionální funkce bude vypadat jako

$$p(x) = \frac{f(x) - 1}{x - g^0}.$$

Pokud je p polynom, pak je opravdu g^0 kořenem polynomu $f(x)$. Nakonec se výsledné racionální funkce $p_0, p_1, \ldots, p_n$ zkombinují způsobem náhodné lineární kombinace, kde se náhodně vybere $\alpha_0, \alpha_1, ..., \alpha_n$

$$CP = \alpha_0 p_0(x) + \alpha_1 p_1(x) + ... + \alpha_n p_n(x).$$

Pokud všechny funkce $p_0, ..., p_n$ byly polynomiální, pak s velmi velkou šancí je i CP polynomiální, a tedy původní tvrzení bylo pravdivé.

Tímto nám končí fáze redukce. Funkce CP se zapíše jako Merkle tree a odešle ověřovateli.

2.3.2.4 Ověření důkazu

V další fázi chce dokazující přesvědčit ověřovatele, že je CP polynomem. V protokolu STARK se místo tohoto prokazuje, že je CP blízko k polynomu malého stupně z důvodu efektivnosti výpočtů.

To se prokazuje pomocí protokolu FRI. Kdy při každém cyklu ověřovatel pošle náhodné β a dokazující nazpět pošle kořenový hash Merkle tree z nově vzniklého polynomu. Když protokol FRI je u konce, dokazující pošle výsledek protokolu.

Ověření důkazu proběhne tak, že ověřovatel pošle q náhodných elementů a dokazující mu pro každý poskytne data k ověření. Data, která pomůžou ověřit každý krok provedených přeměn.

Pro element q_0 poskytne dokazující tři hodnoty, v tomto případě $f(q_0)$, $f(q_0 g)$ a $f(q_0 g^2)$. Pomocí těchto hodnot si ověřovatel spočte odpovídající hodnotu $cp_0(q_0)$ v polynomu CP_0. Spočítat si jej lze, neboť náhodné parametry α a uvedené omezení jsou veřejné.

Pro $cp_0(q_0)$ dokazující také odešle $cp_0(-q_0)$, pomocí kterých si ověřovatel spočte $cp_1(q_0^2)$, protože parametr β je také veřejný a platí

$$CP_i(x) = g(x^2) + xh(x^2),$$
$$CP_i(-x) = g(x^2) - xh(x^2).$$

Vyjádříme $g(x^2)$, $h(x^2)$ a dostaneme

$$g(x^2) = \frac{CP_i(x) - CP_i(-x)}{2},$$
$$h(x^2) = \frac{CP_i(x) - CP_i(-x)}{2x}.$$

Tudíž lze spočítat další člen $CP_{i+1}(x^2)$ pouze za pomoci $CP_i(x)$ a $CP_i(-x)$.

Tímto postupem se dostaneme až k ověření výsledku, který dokazující dříve zaslal. S každým zaslaným bodem dokazující také zasílá autentizační cestu, pomocí které lze ověřit přítomnost prvku v závazku.

2.4 Shrnutí

Nyní, když už jsou popsány protokoly SNARK a STARK, lze uvést jejich rozdíly a navzájem je porovnat. Zaměřím se na důležité charakteristiky: přípravná fáze, časová složitost tvorby důkazu, paměťová velikost důkazu a časová složitost ověření důkazu.

Jak bylo dříve uvedeno, jsou tři typy přípravné fáze, a to důvěryhodné, univerzální a transparentní nastavení. SNARK je postaven na důvěryhodném nastavení. V uvedeném popisu používá náhodně vygenerované bity α. Tyto bity musí být uchovány v tajnosti, neboť při jejich úniku je možné generovat falešné důkazy pro obvod, od kterého tyto bity unikly. Naopak, STARK začíná s transparentním nastavením. V uvedeném popisu nebyly generovány žádné bity navíc, takže nebylo potřeba uchovávat další informace v tajnosti. Z hlediska bezpečnosti je transparentní nastavení lepší než důvěryhodné, proto se začaly objevovat i SNARK schémata s transparentním nastavení[31]. Má to ale veliký dopad na časovou složitost, která se výrazně zvětší, aby se zachovala stručnost důkazu.

Podle zdrojů [32][33] je sestavena tabulka složitostí 2.2. Písmeno n značí počet bran v aritmetickém obvodu a k je konstanta.

Z tabulky 2.2 lze vyčíst, jak roste složitost s narůstající velikostí vstupního problému. U protokolu SNARK je proces tvorby důkazu asymptoticky shora omezen funkcí $nlog(n)$. Jedná se o nejsložitější fázi v tomto protokolu. Na uvedeném popisu se konkrétně jedná o část výpočtu, které se v angličtině říká multiscalar multiplication. Tato operace zabere 80 % výpočtu a používá se při získání $com_f = f_0 \cdot H_0 + ... + f_d \cdot H_d$.[27]

protokol	čas dokazování	velikost důkazu	čas ověřování
SNARK	$O(nlog(n))$	$O(1)$	$O(1)$
STARK	$O(nlog(n)^k)$	$O(log(n)^k)$	$O(log(n)^k)$

Tabulka 2.2 Složitosti SNARK a STARK

Velikost důkazu SNARK je $O(1)$, tedy konstantní. To znamená, že je nezávislý na velikosti vstupu a jeho velikost je stále stejná. V uvedeném popisu se jedná o element grupy $com_f = f(\alpha)g \in G$. Tudíž se jeho velikost pohybuje v bytech.[18]

Konstantní ($O(1)$) se uvádí i čas ověření důkazu, protože spočívá v homomorfním sčítání a bilineárním párování. Tyto operace jsou však proveditelné v konstantním čase.

U protokolu STARK proces dokazování odpovídá časové složitosti $O(nlog(n)^k)$. Podle [13] je za tím provedená polynomiální interpolace. Velikost důkazu je $O(log(n)^k)$ a čas jeho ověření je také $O(log(n)^k)$. Souvisí to s použitím Merkle tree, kde je jeho hloubka logaritmická vzhledem k n. Při ověřování určitého elementu se posílá i autentizační cesta o velikosti ($O(log(n))$). To se provádí pro k elementů, proto $O(log(n)^k)$.[28]

Dalším důležitým bezpečnostním aspektem v dnešní době je kvantová odolnost. Protokol SNARK s jeho kryptografickým předpokladem diskrétního logaritmu není odolný vůči rozšifrování pomocí kvantových počítačů. Zatímco některé zdroje považují STARK za odolný, neboť je postaven na odolnosti vůči kolizi u hashovacích funkcí.[13]

Výhodami SNARK je stručný důkaz a jeho rychlé ověření. Na druhou stranu strádá používáním důvěryhodného nastavení a kvantové neodolnosti. Výhodami STARK je transparentní nastavení a kvantová odolnost, ale za cenu logaritmických velikostí důkazu a času jeho ověření. Obě schémata mají v porovnání své silné a své slabé stránky. Jejich použití závisí na konkrétních situacích a jednotlivých implementacích.

Kapitola 3

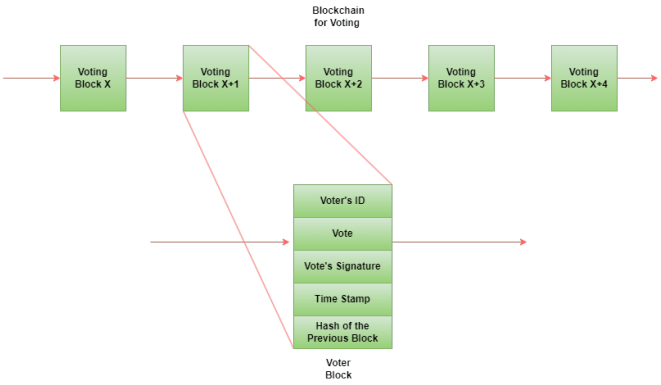
Aplikace důkazu s nulovými znalostmi

Tato kapitole je zaměřena na možné využití důkazu s nulovými znalostmi v reálném světě. Každým rokem přibývá počet oblastí, kde je ZKP přínosem. ZKP najde uplatnění v nejrůznějších směrech od elektronických voleb, autentizace až po ověření věku či anonymní transakce. V této kapitole se budu zabývat elektronickými volbami, autentizací, ověřením nukleárních hlavic a anonymními transakcemi.

3.1 Elektronické volby

V dnešním světe se stále více činností přesouvá do online prostředí. Jednou z těchto činností jsou elektronické volby. V některých zemích je již tato funkcionalita použita, například v Estonsku od roku 2005[34]. Představa elektronických voleb je lákavá, ale jak na to bezpečně? Jedna z nabízených možností je použitím blockchain sítě. Na konkrétní řešení již existuje několik prací, například *Blockchain Based Anonymous Voting System Using zkSNARKs* [35] či práce *Anonymous Voting using Zero-knowledge Proofs*[36].

Každý hlas je obsažen v bloku spolu s dalšími informacemi, které mohou být následující: ID voliče, hlas, podpis, časová známka a hash předchozího bloku, viz obrázek 3.1. K procesu je potřeba třetí důvěryhodná strana, která voličům přiřadí jednotlivé parametry potřebné k hlasování. Tyto parametry musí být podepsány, aby s hlasy nešlo manipulovat.



Obrázek 3.1 Volební blockchain[37]

Použití ZKP zvyšuje bezpečnost tím, že umožňuje voličům prokázat pravost jejich hlasů bez odhalení citlivých informací. ZKP také snižuje šanci podvodů a manipulace s výsledky a hlavně pomocí důkazu s nulovými znalostmi není možné zjistit, kdo jak hlasoval. Na závěr, spojení technologií blockchain a ZKP vytváří efektivní a bezpečný způsob elektronických voleb.[37]

3.2 Autentizace

V digitálním světě je autentizace klíčovým procesem, který zajišťuje, že uživatelé jsou skutečně těmi, za které se vydávají. Autentizovat uživatele lze pomocí tří faktorů:

- znalosti (heslo),
- vlastnictví (token),
- biometriky (otisk prstu).

V dnešní době je nejvíce rozšířena autentizace pomocí znalosti, neboť je jednoduchá a nemá vysoké nároky na použití. Na druhou stranu při prováděné autentizaci může být heslo odcizeno, neboť se posílají data na heslu závislé. Síla ZKP spočívá v odolnosti vůči řadě útoků. Ze své podstaty jsou imunní proti pasivnímu odposlechu, replay útokům a kryptoanalýze databáze na straně serveru. Jedním z autentizačních ZKP protokolů je Secure Remote Password protokol (SRP)[38], který je založen na problému diskrétního logaritmu v konečných tělesech.

Protokol SRP se díky svým vlastnostem začlenil i do reálných aplikací. Do autentizačního procesu ho zapojila celosvětová firma Apple v iCloudu[39]. Společnost 1Password, která je zaměřena na správu hesel, využívá funkcionalit SRP k autentizaci, jak je uvedeno v *1Password Security Design*[40]. Protokol se objevil i jako jedna z možností ustanovení TLS spojení v OpenSSL[41].

Výhody, které SRP přináší oproti běžné autentizaci, jsou následující:

- server nikdy nemusí znát heslo,
- nelze provádět replay útoky, neboť je protokol počítán s privátními náhodnými hodnotami,[1]
- je odolný vůči phishing útokům,
- je odolný vůči slovníkovým útokům.

Nevýhodou je větší množství interakcí mezi dvěma stranami a náročnější složitost operací. Proto existují také protokoly, jako je Schnorrovo schéma, které omezují počet interakcí za cenu snížení bezpečnosti. Návrh SRP protokolu je popsán podrobněji v praktické části.

3.3 Ověření nukleárních hlavic

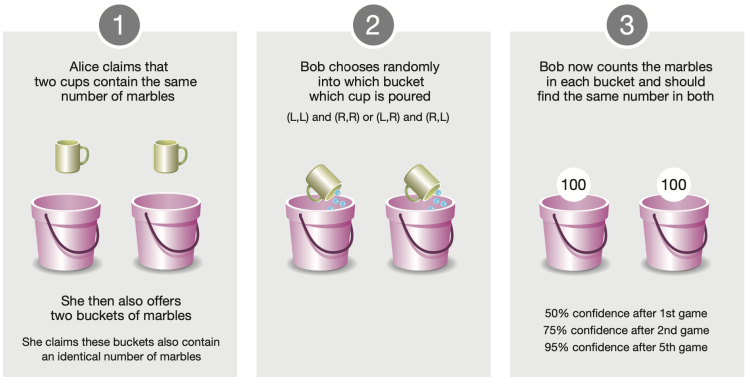
V posledních desetiletích se nabídla ZKP utilizace také v oblasti nukleárních hlavic. Z politického hlediska je důležité, aby existovala možnost, jak ověřit, že se jedná o nukleární hlavici. Bez pomoci ZKP by se ověření stalo odhalením nukleární hlavice a mezinárodní inspektoři by se dozvěděli veškeré detaily. Protokol obohacený ZKP, ale pouze potvrdí, jestli se jedná o nukleární hlavici. Článek *A zero-knowledge protocol for nuclear warhead verification*[42] přichází s novým přístupem, kde citlivé informace nejsou implicitně měřeny, a proto nemusí být skryty. Na tomto článku je také postaven zbytek sekce.

Ověření nukleárních hlavic je postaveno na následujícím principu: Dokazující má dva kelímky, pro které chce dokázat, že v každém je x kuliček, kde x je z rozsahu 1 až 100. Dále ale dokazující nechce prozradit x. K tomu mu slouží dva kbelíky, o kterých také tvrdí, že každý obsahuje $100 - x$

[1] S tím je úzce spojena i forward secrecy.

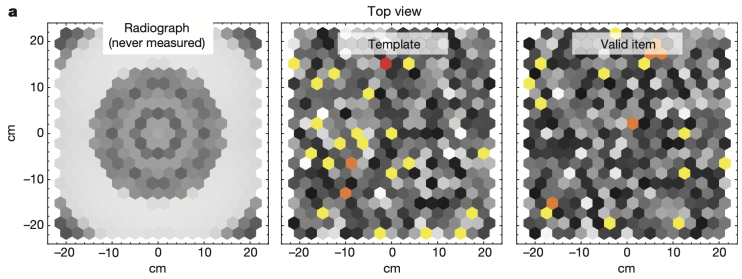
kuliček. Ověřovatel dále náhodně vybere, jaký kelímek vysype do jakého kbelíku. Nyní se ověří, že je v obou kbelících 100 kuliček, viz obrázek 3.2.

Tento způsob neprozradí x, neboť při pravdivém výroku je počet kuliček na konci vždy 100. Pokud by se nevycházelo z pravdivého tvrzení a dokazující neměl stejný počet, pak bez ohledu na to, jak si připraví kbelíky, je pravděpodobnost odhalení 50 %. Při n výměně je pravděpodobnost vypočítána podle vzorce $1 - 2^n$, kde u 5. iterakce se jedná o šanci odhalení > 95 %.



Obrázek 3.2 ZKP pro ověření stejného počtu kuliček ve 2 kelímcích[42]

Spojitost výše uvedeného protokolu a identifikace hlavice spočívá v tom, že se chce prokázat, že dvě a více hlavic mají identické počty neutronových transmisí a emisí při ozařování vysokoenergetickými neutrony. Takové měření záření vytváří jedinečný a komplexní otisk. K posouzení hlavice tedy musí existovat vzor, který se s naměřenými hodnotami porovná. Naměřené hodnoty vzoru a vyšetřované hlavice vypadají například jako na obrázku 3.3.



Obrázek 3.3 Měření záření nukleární hlavice[42]

Na základě těchto měření a porovnání s referenčním vzorem se nakonec usoudí, jestli se jedná o identický případ, tedy nukleární hlavici.

Tím, že je ZKP u ověření nukleárních hlavic založen na fyzikálních vlastnostech objektů, otvírá nové možnosti použití, které jsou dosud neprozkoumány.

3.4 Anonymní transakce

Při běžné transakci jsou zveřejněny obvykle tyto údaje:

- odesílatel,
- příjemce,
- suma.

Tyto informace lze skrýt použitím některých zero knowledge protokolů. Jedním z takových je Zerocash[43]. Zerocash rozšiřuje protokol Bitcoinu o nový typ transakce, který skrývá zmíněnou trojici. Vytvoří oddělenou anonymní měnu (zerocoin) k neanonymní základní měně (basecoin). Uživatel poté může převádět basecoin na zerocoin a provádět anonymní typ transakce. Po provedení může uživatel převést zerocoin zpět na basecoin.[44]

Realizace vlastností Zerocash probíhá pomocí dvou typů transakcí: mint transakce a pour transakce. Mint transakce umožňuje z basecoinu vytvořit zerocoin. Vytvoření nových zerocoinů probíhá pomocí zkSNARK. Tímto způsobem se skryje hodnota, adresa vlastníka a unikátní sériové číslo. Pomocí pour transakce lze dále provádět anonymní platby. Každá transakce spotřebuje nějaké množství zerocoinu a další zerocoiny vytvoří (případně basecoin). ZKP se používá například pro dva vstupní a dva výstupní zerocoiny k ověření, že:

- uživatel vlastní vstupní zerocoin,
- vstupní zerocoin se objevil buď v mint transakci, nebo jiné pour transakci,
- výsledná hodnota součtu vstupních zerocoinů se rovná součtu výsledných.

S dalšími anonymními transakcemi se můžeme setkat u kryptoměny Monero, která používá schéma Bulletproof.[44]

Kapitola 4

ZKP v oblasti autentizace

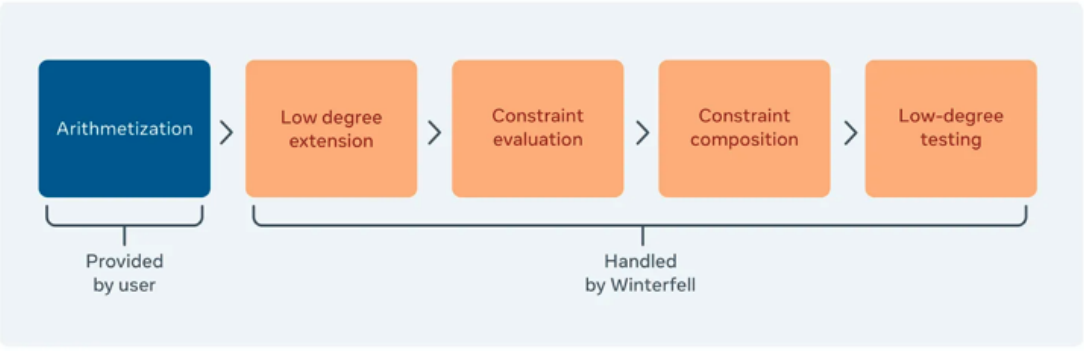
V kapitole zero knowledge proof v oblasti autentizace uvádím dva rozdílné přístupy. První demonstruje na implementaci chování protokolu STARK a používá k tomu open source knihovnu Winterfell. Druhý se zabývá implementací Zero Knowledge Password Protocol (ZKPP)[45], kde je pro ukázku vytvořen proof of concept protokolu SRP. Součástí kapitoly je i různé testování, které je provedeno na zařízení MacBook Air M1 s 8 GB RAM.

4.1 STARK autentizace

Je mnoho způsobů, na kterých lze postavit autentizaci. Pro demonstraci je vybrán příklad použití Lamportova podpisu, kde klient prokazuje, že zná odpovídající tajný klíč.

4.1.1 Knihovna Winterfell

Winterfell[46] je open source knihovna vyvinutá společností Meta, která poskytuje nástroje pro ověřování a dokazování. Jedná se o implementaci protokolu STARK napsanou v jazyce Rust. Knihovna přináší řadu výhod. Za prvé je navržena tak, aby umožnila vývojářům napsání a ověření důkazu bez hlubší znalosti používaných matematických operací kromě aritmetizace[1], viz obrázek 4.1.



Obrázek 4.1 Proces generování STARK důkazu[47]

[1] Aritmetizace je technika, která redukuje složité výpočetní problémy na algebraické pomocí polynomů s nízkým stupněm nad konečným tělesem.

Za druhé nabízí řadu konfigurovatelných parametrů, což umožňuje měnit chování protokolu, jako jsou velikost konečného tělesa nebo typ hashovací funkce. Za třetí je pravidelně aktualizována, například open source knihovna libSTARK[48], která také nabízí nástroje pro protokol STARK, je již 6 let stará.

4.1.2 Lamportův podpis

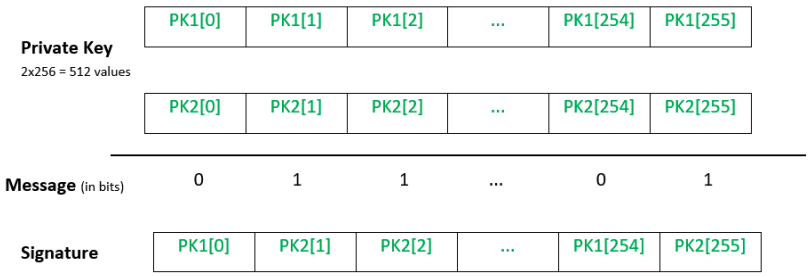
Lamportův podpis byl představen v článku *Constructing Digital Signatures from a One Way Function*[49], z kterého v této sekci vycházím, od Leslieho Lamporta již v roce 1979. Lamportův podpis je jedním ze schémat sloužícím k digitálním podpisům. Na rozdíl od podpisů, které jsou založeny na matematických výpočtech[2], je Lamportův podpis založen na hashovacích operacích. Díky tomu je potencionálně při použití velkých[3] hashovacích funkcí postkvantově odolný.

Lamportův podpis je také nazýván jako Lamportovo jednorázové podpisové schéma, neboť každý privátní klíč musí být použit pouze jednou pro konkrétní podepsání a pak smazán. Aby šlo podepsat více zpráv, je možné vygenerovat dopředu určité množství dvojic privátních a veřejných klíčů a uložit hash veřejných do struktury Merkle tree (Merkle signature scheme). Díky tomu lze následně zveřejnit všechny veřejné klíče v jedné struktuře na jednom místě.

Na vytvoření podpisu je nejdříve potřeba vytvořit pár privátních klíčů. Tyto klíče se skládají z náhodných čísel, kde se jejich počet odvíjí od délky výstupu použité hashovací funkce. Pro funkci SHA256 je každý klíč tedy složen z 256 náhodných čísel. Tyto čísla se generují pomocí náhodného generátoru a mohou nabývat velikosti například 256 bitů. V součtu privátní pár nabývá velikosti $2 \cdot 256 \cdot 256$ b $= 16$ kB.

K privátnímu páru se vytvoří veřejný klíč tím, že se pro každé náhodné číslo vytvoří odpovídající hash. Po vytvoření všech hashů veřejný klíč obsahuje $2 \cdot 256$ hashů, kde každý je 256 bitů velký. Veřejný klíč je tedy také 16 kB velký.

Nyní už jsou všechny potřebné proměnné dostupné a lze sestavit Lamportův podpis. Schéma funguje tak, že se nejdříve pro zprávu m vytvoří hash pomocí bezpečné jednosměrné hashovací funkce. Bitový zápis hashe je složen z 0 a 1. Pro každý bit na pozici i se vybere buď první, nebo druhý privátní klíč na základě hodnoty bitu[4] a z něj náhodná hodnota na odpovídající pozici i, viz obrázek 4.2.



Obrázek 4.2 Lamportův podpis[50]

K ověření digitálního podpisu jsou zveřejněny následující hodnoty:

- zpráva m,
- veřejný klíč,
- Lamportův podpis.

[2]Například digitální podpis pomocí RSA s problémem diskrétního logaritmu.
[3]Hashovací funkce s vyšším počtem výstupních bitů.
[4]Pokud je bit 0 vybere se první z dvojice, jinak druhý.

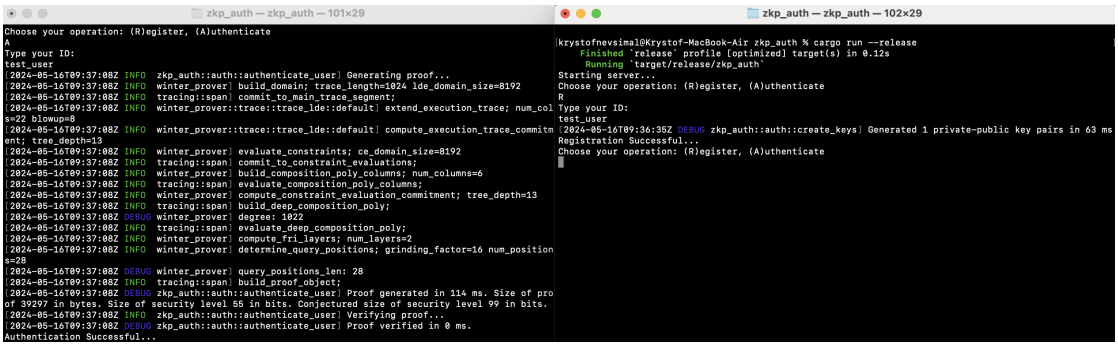
Pomocí těchto hodnot lze jednoduše podpis ověřit. Zprvu se vytvoří hash zprávy m a na základě hodnoty bitu se vezme hodnota z veřejného klíče. Nakonec se pro každé náhodné číslo ze zveřejněného podpisu vytvoří hash a ověří, zdali odpovídá vybrané hodnotě z veřejného klíče. Pokud všechny hashe odpovídají, digitální podpis je ověřen.

4.1.3 Popis implementace

Ukázková implementace protokolu STARK v oblasti autentizace je psána v jazyce Rust, aby byla kompatibilní s knihovnou Winterfell. V knihovně je připravená potřebná aritmizace výpočtů Lamportova podpisu, který je zvolen pro demonstraci. Také obsahuje podpůrné funkce pro tvorbu klíčů a podepisování zpráv.

Implementace se skládá z dvou částí:

- příprava veřejných parametrů (registrace),
- tvorba a ověření důkazu (autentizace).



Obrázek 4.3 Ukázkový běh aplikace využívající STARK

4.1.3.1 Registrace

K získání veřejných parametrů je nutné pro Lamportův podpis vytvořit zprávu m, pár privátních klíčů a odpovídající pár veřejných klíčů. Z těchto hodnot se jako veřejný vstup pro protokol STARK použije dvojice zpráva m a pár veřejných klíčů.

V registrační fázi je ještě potřeba vytvořit podpis zprávy m pomocí privátních klíčů, jehož znalost je později ověřována.

4.1.3.2 Autentizace

Autentizaci lze rozdělit na vytvoření důkazu (klient) a ověření důkazu (server). K vytvoření důkazu jsou potřeba tři struktury:

- veřejný vstup,
- konfigurace volitelných parametrů,
- podpis zprávy m.

Kroky k získání veřejného vstupu i podpisu zprávy m již proběhly v registrační fázi. Zbývá nastavit volitelné parametry[5], které jsou zabudovány v programu. Po nastavení parametrů se vytvoří důkaz, který se předá serveru k ověření.

[5]Nastavení a možnosti parametrů jsou diskutovány v následující sekci.

K ověření důkazu se použijí veřejné vstupy pro daného klienta. Zároveň je možné ověřit i bezpečnostní úroveň[6] důkazu (security level) a při nízké úrovni neschválit autentizaci. Při úspěšném ověření je klient autentizován.

4.1.4 Analýza

Knihovna Winterfell poskytuje velkou svobodu, co se týče volby parametrů. Mezi hlavní patří již volba hashovací funkce, která výrazně ovlivní čas tvorby důkazu a v řádu bytů i jeho velikost. V základu jsou podporované následující hashovací funkce:

- Blake3_192,
- Blake3_256,
- Sha3_256.

V tabulce 4.1 je uvedeno, jak se mění vlastnosti čas tvorby a ověření důkazu, velikost důkazu a úroveň bezpečnosti při použití uvedených hashovacích funkcí.

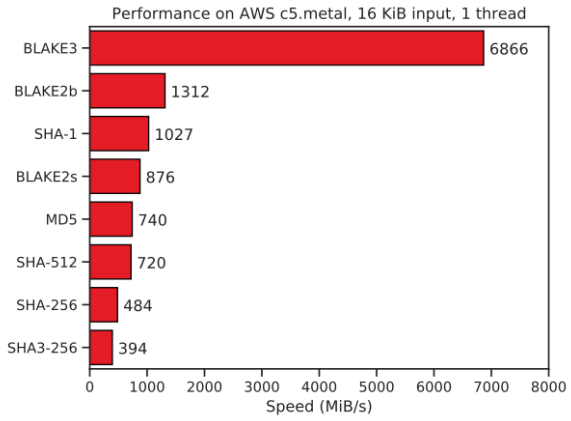
hashovací funkce	dokazování (ms)	velikost (B)	security level (bit)	ověřování (ms)
Blake3_192	134	34 592	55	0,6
Blake3_256	127	39 297	55	0,6
Sha3_256	204	39 871	55	0,7

Tabulka 4.1 Vlastnosti programu při použití různých hashovacích funkcích

Úroveň bezpečnosti je shora omezena kolizní odolností vybrané hashovací funkce. Testované funkce mají kolizní odolnost 96 a 128 bitů[7], a proto mají v tabulce stejnou hodnotu.

Větší rozdíl ve velikosti důkazu mezi Blake3_192 a Blake3_256 společně s Sha3_256 je dán velikostí výstupního hashe funkcí. Zatímco výstupem Blake3_192 je hash o velikosti 192 bitů, funkce Blake3_256 a Sha3_256 mají 256 bitů dlouhý výstupní hash.

Další důležitou položkou, kterou hashovací funkce ovlivňují, je celková doba trvání dokazování a ověřování důkazu. Tento čas se odvíjí od výkonnosti funkcí, která je například mezi Blake3_256 a Sha3_256 velmi odlišná, viz obrázek 4.4.



Obrázek 4.4 Porovnání výkonností hashovacích funkcí[51]

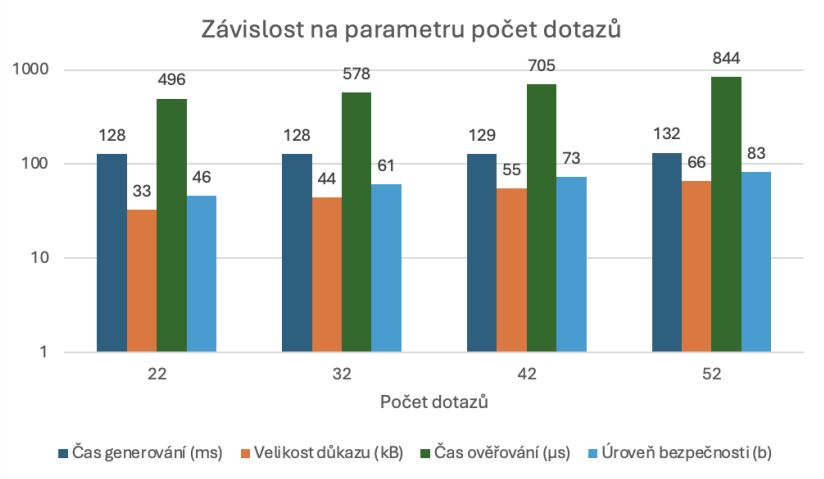
[6] Číslo spojené s množstvím práce, které je potřeba k prolomení kryptografického algoritmu.
[7] To znamená, že je potřeba 2^96 (2^128) provedených hashovacích operacích k prolomení odolnosti vůči kolizi.

Po hashovacích funkcích lze upravit následující parametry:

- počet dotazů (queries),
- blowup faktor,
- grinding faktor,
- použití rozšíření tělesa,
- faktory FRI.

Tyto parametry mají přímý dopad na spolehlivost[8], čas generování důkazu a velikost důkazu. Přičemž každý z nich ovlivní protokol jiným způsobem.

Počet dotazů q značí počet náhodných elementů (simulovaných výzev ověřovatele), pro které jsou předem připraveny data k ověření, že CP je polynom[9]. Poskytnuté data k ověření musí být vyhodnoceny dopředu pro domluvených q elementů, neboť se jedná o neinteraktivní protokol. Čím více elementů je ověřeno, tím vyšší je spolehlivost (úroveň bezpečnosti), ale i velikost důkazu, která je obohacena o data k ověření q elementů, viz graf 4.5.



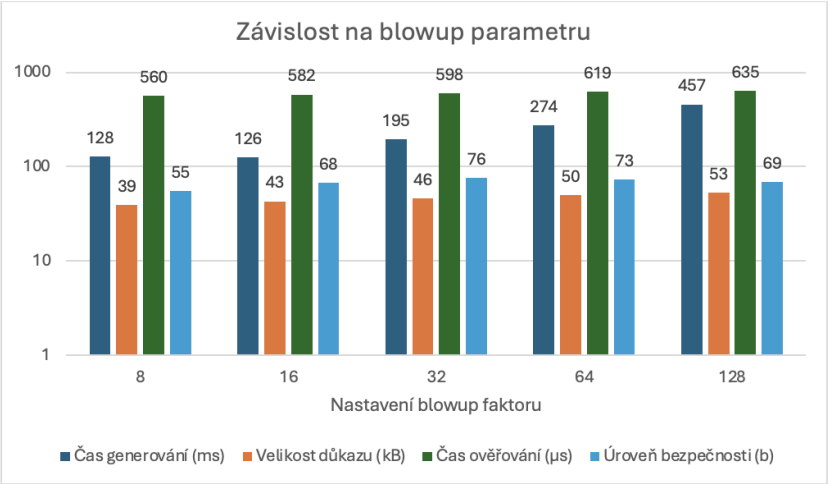
Obrázek 4.5 Závislost protokolu STARK na počtu dotazů

Podle naměřených hodnot na grafu 4.5 s růstem velikosti počtu dotazů roste lineárně velikost důkazu, úroveň bezpečnosti a čas ověřování. Čas generování zůstává s malými výkyvy konstantní.

Parametr blowup faktor k se používá při provádění LDE. Tento faktor značí kolikrát rozšíříme (nafoukneme) množinu bodů, na které je původní polynom definován. Když se zvětší blowup faktor, pravděpodobnost kolize dvou polynomů se sníží, protože stupeň polynomů zůstane stejný, ale množina bodů se zvětší $k\times$. Tedy poměr společných bodů oproti počtu všech bodů se zmenší. Zvýšením blowup faktoru se zvýší spolehlivost, ale společně s časem generování a velikostí důkazu. Chování protokolu se zvyšujícím se faktorem je vyobrazeno na grafu 4.6.

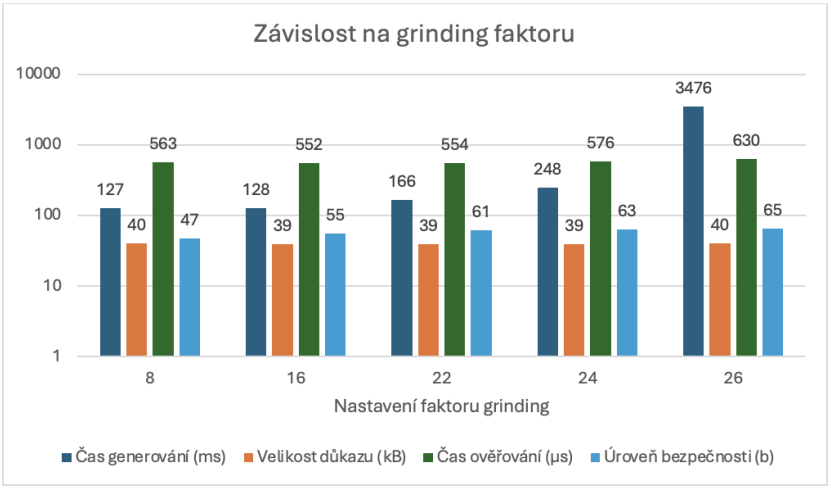
U faktoru blowup se podle naměřených hodnot 4.6 dostalo neočekávaného chování oproti popisu v knihovně Winterfell. Při překročení hranice 32 začala úroveň bezpečnosti klesat. Z toho plyne, že zvyšováním velikosti důkazu a délky doby generování není zaručeně zvedání celkové úrovně bezpečnosti.

[8] Jedna ze tří hlavních vlastností důkazů s nulovými znalostmi.
[9] Odkazuji se na dříve uvedenou teorii.



Obrázek 4.6 Závislost protokolu STARK na faktoru blowup

Grinding faktor g souvisí se zvýšením složitosti tvorby falešného důkazu. Jedná se o určitou práci, která je nutná před samotným posláním důkazu. Z pravidla se vytvoří hash, který souvisí se všemi závazky (hash řetězec) a g bitů dlouhá šablona. Dokazující musí najít k hashi náhodné číslo n, které je v g bodech šablony stejné. Tím pádem při jakékoliv změně libovolného závazku musí být náhodná hodnota znovu nalezena. Grinding faktor zvyšuje spolehlivost, ale na druhou stranu zvýší čas generování důkazu. Od určité velikosti g se čas přidané práce výrazně projeví na celém procesu generování a s jeho dalším zvětšováním roste čas exponenciálně, viz graf 4.7.



Obrázek 4.7 Závislost protokolu STARK na grinding faktoru

Dokud je grinding faktor nízký, je hledání hashe náhodné hodnoty zapadajícího do šablony výpočetně snadné. Ovšem od grinding faktoru 25 je už 2^{25} možností a hledání začíná být náročnější. Podle naměřených hodnot přidáváním bitů do šablony lineárně roste úroveň bezpečnosti. Výhodou je, že velikost důkazu zůstává přibližně stejná a s tím i doba ověřování.

Spolehlivost dále závisí na velikosti konečného tělesa používaného v protokolu STARK pro finální polynom CP. Když se pracuje s menšími tělesy, úroveň bezpečnosti je omezena jejich velikostí, a proto pro zvýšení spolehlivosti je nutné rozšířit konečné těleso. Knihovna Winterfell podporuje k výběru tři možnosti: žádné, kvadratické a kubické. Například pro těleso 2^{64} je úroveň

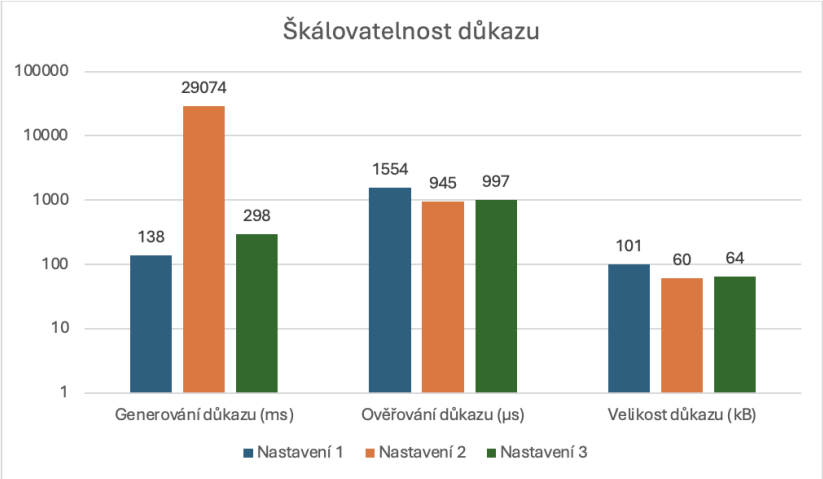
bezpečnosti omezena na 64 bitů. Použitím kvadratického rozšíření se dosáhne tělesa o velikosti 2^{128} a úroveň bezpečnosti stoupne na 128 bitů. Pokud je nutná ještě vyšší maximální úroveň bezpečnosti, lze jí kubickým rozšířením dostat až na 192 bitů.

Nakonec se dají nastavit parametry ovlivňující chování protokolu FRI. FRI folding faktor, který určuje, jak rychle se snižuje stupeň polynomu při každém dalším průchodu procesem FRI, a faktor FRI remainder max degree, který značí maximální povolený stupeň polynomu zbytku. Pomocí těchto parametrů je ovlivněn čas generování a velikost důkazu, například zvýšením FRI folding faktoru se zvýší čas a zmenší velikost.

4.1.5 Shrnutí

Dle provedených měření a analýzy chování při změně parametrů je podtržena hlavní výhoda použití protokolu STARK, a to škálovatelnost. Umožňuje dynamicky měnit všechny podstatné atributy důkazu a přitom zachovat velmi nízký čas ověřování. Tato vlastnost umožňuje klást důraz na jednotlivé vlastnosti na úkor druhých. Další možností je najít optimální střední cestu.

Například pro dosažení úrovně bezpečnosti 100 bitů existuje více způsobů jak navolit parametry, a tedy jaké výsledné charakteristiky bude mít důkaz, viz graf 4.8.



Obrázek 4.8 Škálovatelnost protokolu STARK

Dále je ukázkou demonstrováno, že se jedná o všestranné schéma, které lze aplikovat v mnoha oblastech. STARK také nabízí robustní schéma, které je bezpečné i do budoucna, neboť je stále považováno za odolné vůči kvantovým počítačům.

Na druhou stranu jsou ukázkou vyzdviženy i nepraktické vlastnosti protokolu. Protokol má stále příliš vysoké nároky na čas a paměť. Důkazy o velikost 100 kB a časy generování v řádu stovek milisekund jsou i v dnešní době oproti jiným protokolům více jak nadprůměrné.

4.2 SRP protokol

Secure Remote Password protokol je jedním z Zero Knowledge Password Protokolů. S popisem první verze přišel Thomas Wu v práci *The Secure Remote Password Protocol*[38] z roku 1997. Z informací od Thomase Wu budu nadále vycházet. Práce představuje nový typ autentizace pomocí hesla a zároveň protokol, který může být použit pro výměnu klíčů v nedůvěryhodným prostředí. Cílem bylo vytvořit protokol, který:

- je odolný vůči slovníkovým útokům,
- splňuje forward secrecy (dopředná bezpečnost),
- neumožní po kompromitaci serveru útočníkovi vydávat se za jednotlivé uživatele.

Kryptografický pojem forward secrecy značí vlastnost, která říká, že pokud v budoucnu nastane případná kompromitace hesla, dosavadní komunikace zůstane nedešifrovatelná. Těchto vlastností je dosaženo pomocí kombinace důkazu s nulovými znalostmi a asymetrické výměny klíčů.

4.2.1 Návrh

Pro popis protokolu je použita matematická notace z tabulky 4.2.

n	Veliké prvočíslo, v kterém jsou prováděny matematické operace
g	Generátor modula n
s	Náhodný řetězec použit jako sůl
I	Username uživatele
P	Heslo uživatele
x	Privátní hash odvozen od parametrů uživatele
V	Ověřovatel hesla (password verifier)
u	Veřejný náhodný parametr
a, b	Privátní náhodné parametry
A, B	Veřejné klíče odvozeny od a, b
S	Privátní společný klíč
$H()$	Jednosměrná hashovací funkce
M_1, M_2	Ověřovací zprávy
K	Privátní společný klíč relace (session key)
k	Multiplikativní konstanta

Tabulka 4.2 Matematická notace pro SRP

Protokol SRP lze rozdělit do dvou celků: registrační a autentizační. V registrační části se mezi uživatelem a serverem ustanoví veřejné prvočíslo n a generátor g konečného tělesa s velikostí n. Údaje n a g jsou zpravidla zveřejněny na serveru. Uživatel si vygeneruje sůl s a zvolí heslo P. Pomocí hashovací funkce H vytvoří x. Je více způsobů, například se uvádí

$$x = H(s,\ H(I \mid ':' \mid P)).$$

Pomocí x a generátoru g vypočte ověřovatel hesla

$$V = g^x.$$

Uživatel pošle parametry s, V a I na server, který si je pro uživatele zachová v databázi. Tímto se připraví potřebné proměnné pro autentizaci.

	Carol		Steve
1.		C -->	(lookup s, v)
2.	$x = H(s, P)$	<-- s	
3.	$A = g^a$	A -->	
4.		<-- B, u	$B = v + g^b$
5.	$S = (B - g^x)^{(a + ux)}$		$S = (A \cdot v^u)^b$
6.	$K = H(S)$		$K = H(S)$
7.	$M[1] = H(A, B, K)$	$M[1]$ -->	(verify $M[1]$)
8.	(verify $M[2]$)	<-- $M[2]$	$M[2] = H(A, M[1], K)$

Obrázek 4.9 Proces autentizace[38]

U autentizace je důležité dodržet přesné pořadí operací, viz obrázek 4.9. Uživatel nejdříve zašle na server své uživatelské jméno a nazpět obdrží sůl, která je v databázi k uživatelovi přiřazena. Pomocí soli si opět vypočte

$$x = H(s,\ H(I \mid ':' \mid P)).$$

Zvolí si náhodný element a z n a dopočítá

$$A = g^a.$$

Veřejný klíč A odešle na server. Na serveru se v mezičase zvolí veřejný element u a privátní b. Pomocí b a uloženého V se vypočte

$$B = V + g^b.$$

Server odešle parametry B, u až po obdržení A. Nyní si mohou obě strany spočítat společný privátní klíč S. Uživatel provede operace

$$S = (B - g^x)^{a+u \cdot x}$$

a server

$$S = (A \cdot V^u)^b.$$

Privátní klíč S vyjde stranám stejný, neboť řadou úprav na straně uživatele

$$S = (B - g^x)^{a+u \cdot x},$$

$$S = (V + g^b - g^x)^{a+u \cdot x},$$

$$S = (g^x + g^b - g^x)^{a+u \cdot x},$$

$$S = (g^b)^{a+u \cdot x},$$

se získá

$$S = g^{ab+bux}.$$

Obdobně vyjde na straně serveru

$$S = (A \cdot V^u)^b,$$

$$S = (g^a \cdot (g^x)^u)^b,$$

$$S = (g^{(a+xu)})^b,$$

$$S = g^{ab+bux}.$$

Poté si strany odvodí relační klíč

$$K = H(S).$$

V této fázi stranám zbývá pouze ověřit, zdali mají stejný relační klíč K. K tomu slouží zprávy M_1 a M_2. Nejdříve je nutno poslat zprávu

$$M_1 = H(A, B, K).$$

M_1 posílá uživatel na server, kde ji server porovná se svou zprávou M_1. Pokud se zprávy liší autentizace nebyla úspěšná a končí, jinak se vypočte zpráva

$$M_2 = H(A, M_1, K).$$

M_2 posílá server uživateli, který ověří totožnost. Jestli jsou zprávy stejné, autentizace proběhla úspěšně a byl vytvořen společný klíč K, který může být použit pro následující šifrování komunikace.

Dále je nutno přidat tři kontroly posílaných parametrů, pokud se v komunikaci vyskytne $A = 0$, $B = 0$ nebo $u = 0$ spojení bude ukončeno a autentizace se stane neúspěšnou. Například kdyby se A rovnalo 0, vznikl by i nulový klíč, a to by vedlo k falešné úspěšné autentizaci uživatele.

Kromě kontrol v novějších verzích[52] došlo k dalším jednotlivým úpravám, aby se zvýšila odolnost SRP protokolu. Přidala se multiplikativní konstanta k, která se nyní používá u výpočtu

$$B = k \cdot V + g^b$$

a na straně uživatele

$$S = (B - k \cdot g^x)^{a+u \cdot x}.$$

Konstanta k je veřejně známá a uvádí se

$$k = 3 \text{ nebo } k = H(N, g).$$

Veřejný klíč u se nově neposílá přes síť, ale je tvořen na obou stranách jako

$$u = H(A, B).$$

4.2.2 Popis implementace

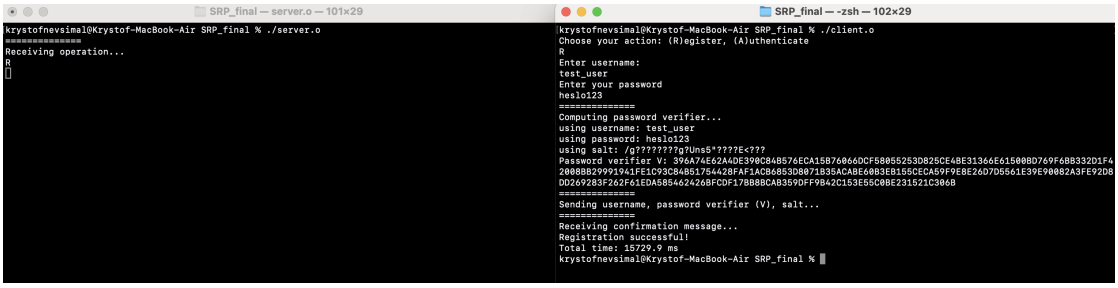
Cílem implementace je demonstrovat bezpečné autentizační schéma SRP. Aplikace je psána v jazyce C++ a využívá externí nástroje OpenSSL a SQLite3. OpenSSL je použita pro hashovací operace a práci s velkými čísly. Práce s SQLite3 databází je určena k ukládání perzistentních informací. V implementaci je vytvořen klient a server pro simulaci reálného prostředí. Server naslouchá na definovaném portu a přijímá paralelně jednotlivé požadavky klientů, které jsou posílány protokolem TCP. Podporovány jsou dvě operace:

- registrace,
- autentizace.

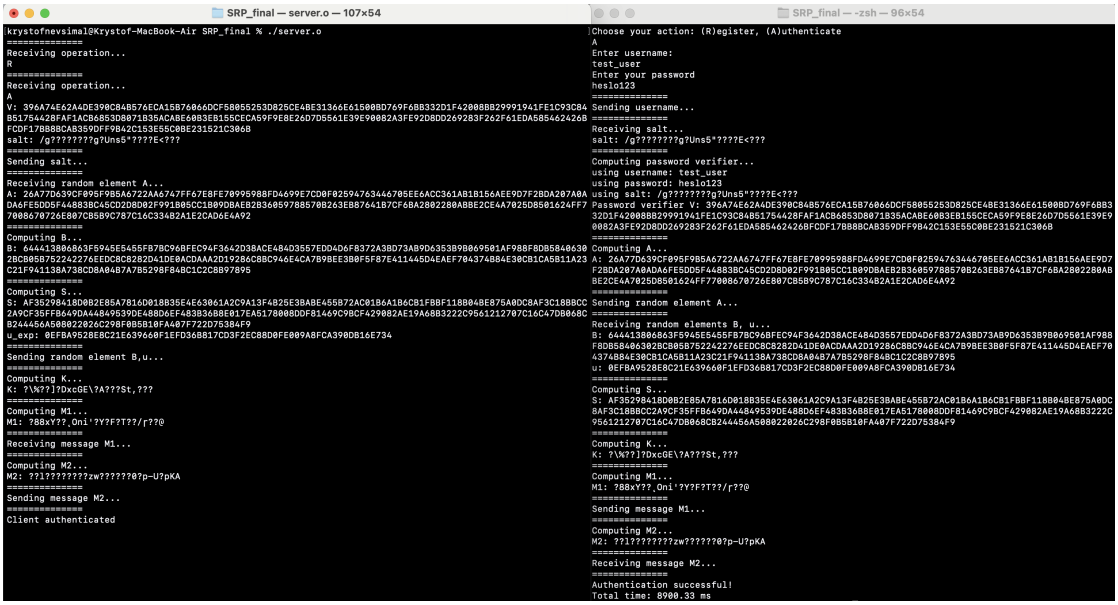
Při registraci uživatelů si server ukládá informace potřebné k budoucí autentizaci do databáze, pokud již není nějaký jiný uživatel zaregistrován pod stejným uživatelským jménem. Po úspěšném zapsání je uživateli poslána potvrzovací zpráva.

Při autentizaci uživatel zašle své uživatelské jméno a na jeho základě jsou přečteny informace z databáze. Pokud uživatel neexistuje je komunikace ukončena, jinak započnou kroky protokolu SRP.

Na straně klienta si program vyžádá uživatelské jméno, heslo a operaci, kterou chce klient provést. Podle rozhodnutí naváže spojení se serverem.



Obrázek 4.10 Ukázkový běh registrace protokolu SRP



Obrázek 4.11 Ukázkový běh autentizace protokolu SRP

V implementaci jsou pro protokol SRP zvoleny následující parametry:

- modulo n délky 1024 bitů převzaté z dodatku A článku RFC 5054[53],
- generátor $g = 2$ pro modulo n,
- náhodné hodnoty s, a, b a u[10] délky 256 bitů,
- multiplikativní konstanta $k = 3$,
- hashovací funkce SHA256,
- sdílený klíč K je hashován pomocí SHA256[11].

[10] Tedy u je generováno náhodně na straně serveru

[11] V demonstrovaném případě se jedná pouze o autentizace, a tedy je tento přístup dostačující.

4.2.3 Časová analýza

V analýze měřím časovou složitost implementace protokolu SRP a jak se mění v závislosti na nastavených parametrech. Parametry, které lze nastavovat jsou:

- modulo n,
- generátor g,
- privátní náhodné elementy a, b a u.

Nejvíce je časová i paměťová složitost ovlivněná velikostí modula n, v kterém probíhají veškeré operace. Testovány jsou modula n o velikosti 1024, 2048, 4096 a 8192 bitů a pro ně různě veliké hodnoty a, b a u. Generátory jsou ke zmíněným n následující:

- $g = 2$ pro 1024 a 2048 bitů veliké n,
- $g = 5$ pro 4096 bitů veliké n,
- $g = 19$ pro 8192 bitů veliké n.

K měření je použit náhodně generovaný seznam tisíce uživatelů, kde každý uživatel má své jméno o deseti znacích a heslo o dvanácti. Uživatelům je nejdříve stopován čas registrace a následně autentizace. V potaz jsou brány pouze úspěšné operace, neboť neúspěšné nemají vypovídající hodnotu.

4.2.3.1 Registrace

Při registraci probíhá několik operací:

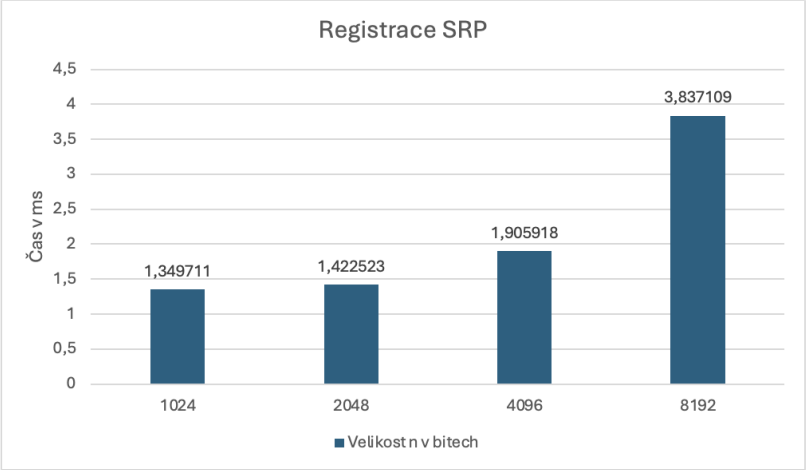
- matematické,
- síťové,
- databázové.

Čas síťových a databázových operací nesouvisí s protokolem SRP, a proto nejsou dále rozebírány. Z matematických operací dobu trvání protokolu SRP ovlivňuje výpočet ověřovatele hesla $V = g^x$, kde $x = H(s,\ H(I \mid ':' \mid P))$. Jelikož je používána hashovací funkce SHA256 a její výstup je vždy stejně velký, x je 256 bitů dlouhý, a to pro všechny délky[12] soli, uživatelského jména a hesla.

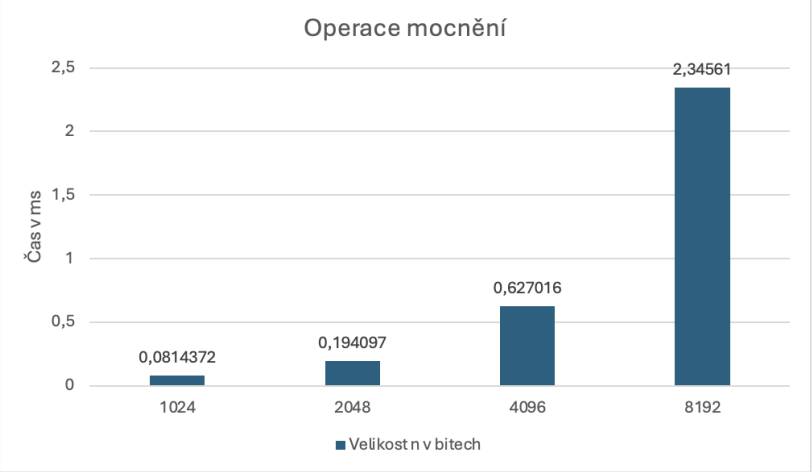
Doba výpočtu g^x lze tím pádem ovlivnit pouze změnou generátoru g nebo modula n, pokud počítáme s konzistentním časem hashovací funkce pro výpočet x. Otestováno je tisíc uživatelů s výše uvedenými moduly n a jejich generátory g. Ze všech časů pro konkrétní n je spočítán průměr. Tyto průměrné hodnoty jsou uvedené v grafu 4.12.

Při pohledu na čas trvání matematické operace mocnění g^x na grafu 4.13 je vidět, že pro malá modula s nízkými generátory má čas matematických výpočtů malý podíl na celkové době trvání implementace protokolu SRP. Na druhou stranu mocnění ve velkém modulu se již významně projeví ve výsledném čase. Na základě výsledků lze odhadnout, že síťová a databázová část trvá přibližně $1,3167752$ ms.

[12] Dokud se nepřekročí maximální vstupní velikost hashovací funkce.



Obrázek 4.12 Čas registrace SRP v závislosti na velikosti n



Obrázek 4.13 Čas mocnění při registraci SRP v závislosti na velikosti n

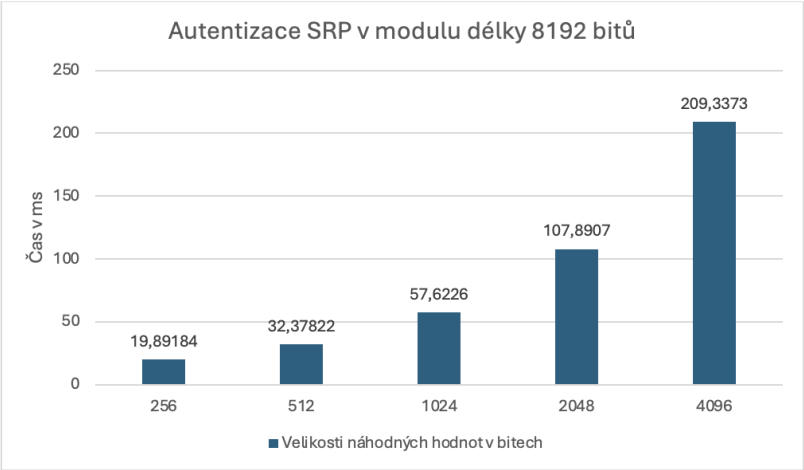
4.2.3.2 Autentizace

V implementaci autentizace je již použito více matematických operací a navíc jsou k výpočtům použity náhodné hodnoty a, b a u. Nastavení velikosti těchto hodnot přímo ovlivňuje následující prvky. Tedy velikost náhodných hodnot řetězově ovlivní celý autentizační proces, a tím i jeho dobu trvání. Například pro n délky 8192 bitů je čas pro různé velikosti náhodných hodnot znázorněn na grafu 4.14.

Čas programu se dle grafu 4.14 pro náhodné hodnoty délky 256, 512, 1024, 2048 a 4096 bitů lineárně zvyšuje. Pro vyznačené hodnoty lze přibližně určit multiplikativní konstantu $l = 2$, která určuje jak moc se zvýší čas při zdvojnásobení velikosti náhodných prvků a, b a u.

Při testování v menších modulech nemusí konstanta l odpovídat, neboť čas autentizace není z větší části ovlivněn matematickými operacemi, nýbrž čtením databáze či příjmem a odesíláním zpráv, viz kompletní seznam naměřených průměrných hodnot z testů v tabulce 4.3.

Z naměřených hodnot lze vyčíst, že velikost modula ovlivňuje čas mnohem výrazněji, než velikost náhodných hodnot. To je důsledek toho, že náhodné hodnoty se průběhem protokolu přiblíží maximální velikosti modula a nadále už nemohou stěžovat matematické operace.



Obrázek 4.14 Čas autentizace SRP v závislosti na velikosti náhodných hodnot v modulu délky 8192

velikost n	256	512	1024	2048	4096
1024	1,443495	1,707263	N/A	N/A	N/A
2048	2,293087	3,202423	4,935023	N/A	N/A
4096	5,816762	9,116614	16,84381	32,68276	N/A
8192	19,89184	32,37822	57,6226	107,8907	209,3373

Tabulka 4.3 Čas autentizace (ms) v modulu n v závislosti na velikosti náhodných čísel (v bitech)

4.2.4 Shrnutí

Secure remote password protokol nabízí spolehlivé autentizační schéma, které sebou přináší řadu pozitivních vlastností:

- heslo není posíláno ani uloženo na serveru,
- při komunikaci neuniká žádná informace o heslu,
- autentizuje se nejen uživatel, ale i server,
- autentizací rovnou vznikne společný klíč.

I když se takto může zdát SRP bezchybný, má i své negativní vlastnosti. První z nich přichází s kvantovými počítači, kdy se za jejich pomoci problém diskrétního logaritmu pokládá za řešitelný. Tím by protokol SRP nebyl nadále použitelný.

Za druhé je požadováno větší množství interakcí oproti běžné autentizaci, i když je možné uvedený návrh optimalizovat na čtyři kroky[13]. Pro řešení tohoto problému lze například experimentálně nasadit zmíněnou transformaci Fiat-Shamir, která je spojována se zero knowledge proof.

Ačkoli je SRP protokol vhodným kandidátem pro autentizaci, v dnešní době se dává přednost použití druhého faktoru a ověřování pomocí třetích stran. Tento způsob totiž přináší neprolomitelnou autentizaci i při úniku hesla, neboť je potřeba i druhý faktor vlastnictví, například mobilní zařízení.

[13]Lze poslat na začátku autentizace s uživatelským jménem rovnou i element A a obdržet s společně s prvky B a u.

Kapitola 5
Závěr

Cílem práce bylo vytvořit studii, která shrne a ucelí kryptografickou metodu zvanou důkaz s nulovými znalostmi, a aplikaci demonstrující toto téma v oblasti autentizace. Tyto cíle byly realizovány ve dvou částech, a to teoretické a praktické.

Teoretický cíl zaměřený na shrnutí a ucelení důkazu s nulovými znalostmi byl rozdělen na tři podcíle. Prvním bylo popsat důkaz s nulovými znalostmi a jeho vlastnosti. Druhým bylo vypracovat přehled o současném stavu na základě dostupných informací. Posledním třetím podcílem bylo vyzdvihnout různé oblasti, v kterých je možné tuto kryptografickou metodu aplikovat.

Pro splnění podcílů práce nejdříve popisovala důkaz s nulovými znalostmi a zabývala se základními vlastnostmi, které musí být splněny. Podstata a vlastnosti důkazu byly znázorněny na jednoduchých příkladech pro rychlé pochopení jinak složité kryptografické metody. Pro vypracování přehledu o současném stavu byl nejdříve uveden historický vývoj, v kterém byly zmíněny významné protokoly, které již našly uplatnění v reálných aplikacích. Tyto protokoly využívají ve své podstatě odlišné mechanismy, a proto byly podle rozdílných vlastností rozděleny do skupin. Ze zmíněných protokolů byly pro detailnější popis vybrány schémata SNARK a STARK. Nejdříve bylo uvedeno obecné schéma, kterého se většina univerzálních protokolů drží. Poté následoval popis jednotlivých kroků těchto protokolů a vysvětlení související teorie, bez které by nešlo důkazy sestrojit.

Pro splnění třetího teoretického podcíle byla teoretická část zakončena uvedením aplikací důkazu s nulovými znalostmi v reálném světě. Mezi vybranými příklady byly elektronické volby, autentizace, anonymní transakce a ověření nukleárních hlavic. Avšak tyto ukázky demonstrují pouze malý zlomek všech oblastí, v kterých lze důkaz uplatnit. V průběhu práce byly také průběžně zmiňovány i kryptografické metody, s kterými lze důkaz s nulovými vlastnostmi kombinovat. Od homomorfního šifrování, přes schéma jednorázového digitálního podpisu, až po autentizační mechanismy.

Splnění cíle demonstrovat důkaz s nulovými znalostmi v oblasti autentizace bylo plněno pomocí dvou praktických aplikací. K první aplikaci byla použita open-source knihovna Winterfell, která je implementací protokolu STARK. V autentizaci byla ověřována znalost Lamportových privátních klíčů. Na základě ukázky byla provedena analýza, která se zabývala podstatnými vlastnostmi důkazu. Druhá aplikace se věnovala protokolu SRP, který je určený konkrétně k autentizaci a je považován za ZKPP. Na této aplikaci byla analyzována časová složitost v závislosti na nastavených parametrech.

Tato bakalářská práce přináší přehledný vstup do problematiky důkazu s nulovými znalostmi a uvádí možné nástroje pro tvorbu vlastního důkazu. Nulová znalost se v dnešní době stále vyvíjí a již zaznamenává uplatnění v reálném světě. I když nulová znalost nabízí kvalitní zvýšení kybernetické bezpečnosti, stále je potřeba vyřešit vyšší nároky na čas a paměť.

Bibliografie

1. GOLDWASSER, S; MICALI, S; RACKOFF, C. The knowledge complexity of interactive proof-systems. In: *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*. Providence, Rhode Island, USA: Association for Computing Machinery, 1985, s. 291–304. ISBN 0897911512. Dostupné z DOI: 10.1145/22145.22178.
2. BLUM, Manuel; FELDMAN, Paul; MICALI, Silvio. Non-interactive zero-knowledge and its applications. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*. Chicago, Illinois, USA: Association for Computing Machinery, 1988, s. 103–112. ISBN 0897912640. Dostupné z DOI: 10.1145/62212.62222.
3. ZHU, Nicole. *Understanding Zero-knowledge proofs through illustrated examples* [online]. 2019. Dostupné také z: https://blog.goodaudience.com/understanding-zero-knowledge-proofs-through-simple-examples-df673f796d99. [Accessed 25-04-2024].
4. BITANSKY, Nir; CANETTI, Ran; CHIESA, Alessandro; TROMER, Eran. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. Cambridge, Massachusetts: Association for Computing Machinery, 2012, s. 326–349. ISBN 9781450311151. Dostupné z DOI: 10.1145/2090236.2090263.
5. DAPHNE, Tian. *Zero-knowledge proofs explained in 3 examples* [online]. 2022. Dostupné také z: https://www.circularise.com/blogs/zero-knowledge-proofs-explained-in-3-examples. [Accessed 22-04-2024].
6. PARNO, Bryan; HOWELL, Jon; GENTRY, Craig; RAYKOVA, Mariana. Pinocchio: Nearly Practical Verifiable Computation. In: *2013 IEEE Symposium on Security and Privacy*. 2013, s. 238–252. Dostupné z DOI: 10.1109/SP.2013.47.
7. BINELLO, Maurizio. *The Pinocchio protocol* [online]. 2019. Dostupné také z: http://www.zeroknowledgeblog.com/index.php/the-pinocchio-protocol. [Accessed 25-04-2024].
8. ZCASH. *Learn Zcash, What are zero knowledge proofs?* [Online]. [B.r.]. Dostupné také z: https://z.cash/learn/what-are-zero-knowledge-proofs/#:~:text=Zcash%20was%20the%20first%20real,21%20years%20old%20without%20exposing. Accessed: 31.3.2024.
9. GROTH, Jens. On the Size of Pairing-Based Non-interactive Arguments. 2016, s. 305–326. ISBN 978-3-662-49895-8. Dostupné z DOI: 10.1007/978-3-662-49896-5_11.
10. BINELLO, Maurizio. *Groth16* [online]. 2019. Dostupné také z: https://www.zeroknowledgeblog.com/index.php/groth16. [Accessed 25-04-2024].
11. GABIZON, Ariel; WILLIAMSON, Zachary J.; CIOBOTARU, Oana. *PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge* [Cryptology ePrint Archive, Paper 2019/953]. 2019. Dostupné také z: https://eprint.iacr.org/2019/953.

12. BÜNZ, Benedikt; BOOTLE, Jonathan; BONEH, Dan; POELSTRA, Andrew; WUILLE, Pieter; MAXWELL, Greg. Bulletproofs: Short Proofs for Confidential Transactions and More. 2018, s. 315–334. Dostupné z DOI: 10.1109/SP.2018.00020.
13. BEN-SASSON, Eli; BENTOV, Iddo; HORESH, Yinon; RIABZEV, Michael. *Scalable, transparent, and post-quantum secure computational integrity* [Cryptology ePrint Archive, Paper 2018/046]. 2018. Dostupné také z: https://eprint.iacr.org/2018/046.
14. BEN-SASSON, Eli. *A Cambrian Explosion of Crypto Proofs* [online]. 2020. Dostupné také z: https://nakamoto.com/cambrian-explosion-of-crypto-proofs/. [Accessed 25-04-2024].
15. *9 the discrete logarithm problem* [online]. 2022. Dostupné také z: https://math.mit.edu/classes/18.783/2022/LectureNotes9.pdf. [Accessed 13-03-2024].
16. LÓRENCZ, Róbert. *Hašovací funkce, SHA-x, HMAC* [online]. [B.r.]. Dostupné také z: https://courses.fit.cvut.cz/BI-KAB/media/lectures/kab5.pdf. [Accessed 17-04-2024].
17. ZEILBERGER, Hadas. *Simple Explanations of Arithmetic Circuits and Zero-Knowledge Proofs* [online]. 2019. Dostupné také z: https://medium.com/web3studio/simple-explanations-of-arithmetic-circuits-and-zero-knowledge-proofs-806e59a79785. [Accessed 15-04-2024].
18. BONEH, Dan. *What is a SNARK?* [Online]. 2023. Dostupné také z: https://zkhack.dev/whiteboard/module-one/. [Accessed 15-03-2024].
19. PAVEL HRABÁK Tomáš Kalvoda, Ivo Petr. *Matematická analýza 1* [online]. 2024. Dostupné také z: https://courses.fit.cvut.cz/BI-MA1/textbook/bi-ma1-textbook.pdf. [Accessed 1-04-2024].
20. PETKOV, Maksym. *Why and How zk-SNARK Works 1: Introduction & the Medium of a Proof* [online]. 2019. Dostupné také z: https://medium.com/@imolfar/why-and-how-zk-snark-works-1-introduction-the-medium-of-a-proof-d946e931160. [Accessed 16-04-2024].
21. BONEH, Dan. *Building a SNARK (Part I)* [online]. 2023. Dostupné také z: https://zkhack.dev/whiteboard/module-two/. [Accessed 15-03-2024].
22. YIWEN, Song. *Commitment schemes* [online]. 2019. Dostupné také z: https://cseweb.ucsd.edu/classes/fa19/cse206A-a/LecCommit.pdf. [Accessed 05-03-2024].
23. ARAMPATZIS, Anastasios. *Homomorphic Encryption: What Is It and How Is It Used* [online]. 2023. Dostupné také z: https://venafi.com/blog/homomorphic-encryption-what-it-and-how-it-used/. [Accessed 17-04-2024].
24. VERIDISE. *ZK Fundamentals: The Fiat-Shamir Transform* [online]. 2023. Dostupné také z: https://medium.com/veridise/zk-fundamentals-the-fiat-shamir-transform-bfa69e2fd32e. [Accessed 17-04-2024].
25. BETHENCOURT, John. *Intro to Bilinear Maps* [online]. 2015. Dostupné také z: https://people.csail.mit.edu/alinush/6.857-spring-2015/papers/bilinear-maps.pdf. [Accessed 17-04-2024].
26. KATE ANIKET Zaverucha Gregory, Goldberg Ian. Constant-Size Commitments to Polynomials and Their Applications. 2010, s. 177–194. ISBN 978-3-642-17372-1. Dostupné z DOI: 10.1007/978-3-642-17373-8_11.
27. MARCIN, Vladimír. *Informatické večery: Rozluštění tajemství Zero-Knowledge kryptografie* [online]. 2023. Dostupné také z: https://fit.cvut.cz/cs/zivot-na-fit/aktualne/udalosti/20176-informaticke-vecery-rozlusteni-tajemstvi-zero-knowledge-kryptografie. [Accessed 07-04-2024].
28. STARKWARE. *STARK 101* [online]. 2019. Dostupné také z: https://starkware.co/stark-101/. [Accessed 06-04-2024].

29. MICHAEL FORBES, Dana Moshkovitz. *Lecture 3: The Sum Check Protocol* [online]. 2010. Dostupné také z: https://people.csail.mit.edu/dmoshkov/courses/pcp-mit/3-sum-check.pdf. [Accessed 06-04-2024].
30. *Merkle Tree in Blockchain: What it is and How it Works* [online]. 2021. Dostupné také z: https://www.investopedia.com/terms/m/merkle-tree.asp. [Accessed 06-04-2024].
31. BÜNZ, Benedikt; FISCH, Ben; SZEPIENIEC, Alan. Transparent SNARKs from DARK Compilers. In: *Advances in Cryptology – EUROCRYPT 2020*. Cham: Springer International Publishing, 2020, s. 677–706. ISBN 978-3-030-45721-1.
32. MALLER, Mary; BOWE, Sean; KOHLWEISS, Markulf; MEIKLEJOHN, Sarah. Sonic: Zero-Knowledge SNARKs from Linear-Size Universal and Updatable Structured Reference Strings. 2019, s. 2111–2128. ISBN 978-1-4503-6747-9. Dostupné z DOI: 10.1145/3319535.3339817.
33. ZHOU, Linfeng. *Comparison of Different zk-SNARKs* [online]. 2020. Dostupné také z: https://medium.com/@daniel.linfeng.zhou/comparison-of-different-zk-snarks-3f3ac7dd8a4a. [Accessed 19-04-2024].
34. *e-Democracy & open data* [online]. [B.r.]. Dostupné také z: https://e-estonia.com/solutions/e-governance/e-democracy/?ref=hackernoon.com. [Accessed 20-04-2024].
35. MURTAZA, Malik Hamza; ALIZAI, Zahoor Ahmed; IQBAL, Zubair. Blockchain Based Anonymous Voting System Using zkSNARKs. In: *2019 International Conference on Applied and Engineering Mathematics (ICAEM)*. 2019, s. 209–214. Dostupné z DOI: 10.1109/ICAEM.2019.8853836.
36. CHINMAY, Sonar; RAKSHITH Gopalakrishna nad Radha, Kumaran. *Anonymous Voting using Zero-knowledge Proofs* [online]. [B.r.]. Dostupné také z: https://chinmaysonar.github.io/Projects/bc-zk-report.pdf. [Accessed 20-04-2024].
37. *Zero-Knowledge Proofs in Blockchain Voting* [online]. 2023. Dostupné také z: https://hackernoon.com/zero-knowledge-proofs-in-blockchain-voting. [Accessed 20-04-2024].
38. WU, Thomas. *The Secure Remote Password Protocol* [online]. 1997. Dostupné také z: http://srp.stanford.edu/ndss.html. [Accessed 20-04-2024].
39. *Escrow security for iCloud Keychain* [online]. 2021. Dostupné také z: https://support.apple.com/en-my/guide/security/sec3e341e75d/web. [Accessed 05-05-2024].
40. 1PASSWORD. *1Password Security Design* [online]. 2023. Dostupné také z: https://1passwordstatic.com/files/security/1password-white-paper.pdf. [Accessed 5-5-2024].
41. OPENSSL. *OpenSSL Cryptography and SSL/TLS Toolkit* [online]. [B.r.]. Dostupné také z: https://www.openssl.org/docs/manmaster/man3/SSL_CTX_set_srp_password.html. [Accessed 05-05-2024].
42. ALEXANDER, Glaser; BOAZ, Barak; ROBERT J., Goldston. A zero-knowledge protocol for nuclear warhead verification [Nature]. 2014. Dostupné z DOI: 10.1038/nature13457.
43. BEN-SASSON, Eli; CHIESA, Alessandro; GARMAN, Christina; GREEN, Matthew; MIERS, Ian; TROMER, Eran; VIRZA, Madars. Zerocash: Decentralized Anonymous Payments from Bitcoin. 2014, s. 459–474. Dostupné z DOI: 10.1109/SP.2014.36.
44. *How Zerocash works* [online]. [B.r.]. Dostupné také z: http://zerocash-project.org/how_zerocash_works. [Accessed 22-04-2024].
45. *Zero-Knowledge Password Protocol* [online]. [B.r.]. Dostupné také z: https://csrc.nist.gov/glossary/term/zero_knowledge_password_protocol. [Accessed 04-05-2024].
46. FACEBOOK. *winterfell* [online]. [B.r.]. Dostupné také z: https://github.com/facebook/winterfell. [Accessed 04-05-2024].

47. IRAKLIY, Khaburzaniya; KOSTAS, Chalkias; LEWI, Kevin; HARJASLEEN, Malvai. *Open sourcing Winterfell: A STARK prover and verifier* [online]. 2021. Dostupné také z: `https://engineering.fb.com/2021/08/04/open-source/winterfell/`. [Accessed 01-05-2024].

48. BEN-SASSON, Eli. *libSTARK* [online]. [B.r.]. Dostupné také z: `https://github.com/elibensasson/libSTARK`. [Accessed 01-05-2024].

49. LAMPORT, Leslie. *Constructing Digital Signatures from a One Way Function* [SRI International]. 1979. Dostupné také z: `https://www.microsoft.com/en-us/research/uploads/prod/2016/12/Constructing-Digital-Signatures-from-a-One-Way-Function.pdf`. [Accessed 01-05-2024].

50. *Lamport One Time Signature Scheme* [online]. 2021. Dostupné také z: `https://www.geeksforgeeks.org/lamport-one-time-signature-scheme/`. [Accessed 01-05-2024].

51. *BLAKE3* [online]. [B.r.]. Dostupné také z: `https://github.com/BLAKE3-team/BLAKE3`. [Accessed 02-05-2024].

52. WU, Tom. *SRP Protocol Design* [online]. [B.r.]. Dostupné také z: `http://srp.stanford.edu/design.html`. [Accessed 05-05-2024].

53. TAYLOR, David; PERRIN, Trevor; WU, Thomas; MAVROGIANNOPOULOS, Nikos. *Using the Secure Remote Password (SRP) Protocol for TLS Authentication* [RFC 5054]. RFC Editor, 2007. Request for Comments, č. 5054. Dostupné z DOI: `10.17487/RFC5054`.

Obsah příloh