FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Algoritmy pro sdílení tajemství
Student:	Hana Svobodová
Vedoucí:	Ing. Josef Kokeš
Studijní program:	Informatika
Studijní obor:	Bezpečnost a informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	Do konce letního semestru 2019/20

Pokyny pro vypracování

Proveďte rešerši známých algoritmů pro sdílení tajemství a popište jejich matematické principy.
Analyzujte faktory ovlivňující bezpečnost těchto algoritmů.
Naimplementujte aspoň dva algoritmy v jazyce C s využitím knihovny OpenSSL.
Otestujte vytvořené algoritmy a porovnejte jejich výsledky.
Naimplementujte aplikaci ovládanou z příkazové řádky. Jejím vstupem bude soubor a parametry pro rozdělení sdíleného tajemství (počet dílů, počet dílů nutných pro rekonstrukci). Aplikace zašifruje soubor bezpečnou symetrickou šifrou za použití náhodně vygenerovaného klíče, tento klíč pak bude algoritmy nastudovanými výše rozdělen na zadaný počet částí a spolu se zašifrovaným souborem předán uživateli. Druhá část aplikace pak umožní ze zašifrovaného souboru a zadané množiny dílů tajemství rekonstruovat originální soubor.

Seznam odborné literatury

Dodá vedoucí práce.

prof. Ing. Pavel Tvrdík, CSc.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 21. prosince 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Algoritmy pro sdílení tajemství

Hana Svobodová

Katedra počítačových systémů
Vedoucí práce: Ing. Josef Kokeš

7. května 2019

Poděkování

Ráda bych poděkovala Ing. Josefu Kokešovi za veškerou pomoc při zpracování této práce, především za cenné rady, podnětné připomínky a vstřícný přístup.

Prohlášení

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

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 7. května 2019 ……

Odkaz na tuto práci

Svobodová, Hana. *Algoritmy pro sdílení tajemství*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

Tato práce se zabývá algoritmy pro sdílení tajemství. Náplní rešeršní části práce je seznámení s matematickými principy algoritmů pro sdílení tajemství a analýza faktorů ovlivňujících jejich bezpečnost. V praktické části práce jsou implementovány Shamirův a Asmuthův–Bloomův algoritmus pro sdílení tajemství, v další části jsou porovnány jejich výsledky a implementována aplikace, která umožňuje zašifrovat soubor symetrickou šifrou a použitý šifrovací klíč rozdělit implementovanými algoritmy na zadaný počet dílů. Následně je možné z množiny dílů soubor dešifrovat. Aplikace je napsána v jazyce C s použitím knihovny OpenSSL.

Klíčová slova sdílení tajemství, sdílení šifrovacích klíčů, Shamirův algoritmus pro sdílení tajemství, Blakleyho algoritmus pro sdílení tajemství, Asmuthův–Bloomův algoritmus, aplikace pro sdílení šifrovacích klíčů, C, OpenSSL

Abstract

This thesis deals with secret sharing algorithms. The survey part is focused on the introduction of chosen secret sharing algorithms, their mathematical background and analysis of factors affecting their security. Shamir's and Asmuth-Bloom's algorithms are implemented in the second part. Then, their results are compared. The next part presents the implemented application which can encrypt given file with a symmetric cipher and share the used key using the implemented secret sharing algorithms. It is also possible to decrypt the file using shares of the secret and the encrypted file. The application is written in C and uses OpenSSL library.

Keywords secret sharing, sharing cryptographic keys, Shamir's secret sharing algorithm, Blakley's secret sharing algorithm, Asmuth-Bloom's algorithm, application for sharing encryption keys, C, OpenSSL

Obsah

Seznam obrázků

Seznam algoritmů a výpisů kódu

Úvod

S postupující digitalizací dat je stále naléhavější otázka, jak sdílet důležitá nebo citlivá data jako šifrovací klíče mezi více lidí. Pokud každý dostane kopii nebo část těchto dat, hrozí jejich zneužití. Je třeba, aby jednotlivci nemohli data zneužít, z čehož vyplývá, že je nesmí znát. Zároveň je nutné, aby určité množiny lidí mohly v případě potřeby data získat. Jedním z možných řešení problému je data rozdělit pomocí algoritmů pro sdílení tajemství.

Rešeršní část práce se zabývá vybranými algoritmy pro sdílení tajemství. Každý z nich vychází z jiných matematických principů, které jsou v této části popsány. V této části jsou rovněž popsány vlastnosti těchto algoritmů a provedena analýza jejich bezpečnosti.

Praktická část práce je zaměřena na implementaci dvou různých algoritmů popsaných v rešeršní části. Poté jsou tyto implementace otestovány a porovnány jejich výsledky. Následně je popsána implementace aplikace umožňující zašifrování souboru bezpečnou symetrickou šifrou, rozdělení šifrovacího klíče jedním z implementovaných algoritmů pro sdílení tajemství a opětovnou rekonstrukci klíče a zašifrovaného souboru pomocí zadané množiny dílů tajemství.

Kapitola 1

Cíl práce

Cílem rešeršní části práce je popis známých algoritmů pro sdílení tajemství, jejich matematických principů a analýza faktorů ovlivňujících jejich bezpečnost.

Cílem praktické části práce je implementace vybraných algoritmů v jazyce C s využitím knihovny OpenSSL, jejich otestování a porovnání výsledků. Dalším cílem je implementace aplikace, jejímž vstupem bude soubor a parametry pro rozdělení sdíleného tajemství (celkový počet dílů, počet dílů nutných pro rekonstrukci). Aplikace zašifruje soubor bezpečnou symetrickou šifrou za použití náhodně vygenerovaného klíče, který bude implementovanými algoritmy rozdělen na zadaný počet dílů a spolu se zašifrovaným souborem předán uživateli. Druhá část aplikace umožní ze zašifrovaného souboru a zadané množiny dílů tajemství zrekonstruovat originální soubor.

Kapitola 2

Algoritmy pro sdílení tajemství

Tato kapitola se zabývá popisem tří různých algoritmů pro sdílení tajemství. Nejprve jsou uvedeny definice a základní problémy, které při sdílení tajemství mohou nastat.

Definice 2.1 Systém, kdy je tajemství rozděleno na n dílů tak, že libovolných k z nich postačuje k rekonstrukci tajemství, se nazývá *(k,n)-prahové schéma*. [1]

Definice 2.2 Schéma pro sdílení tajemství je *dokonalé*, pokud každá podmnožina účastníků, která může ze svých dílů získat nějakou informaci o tajemství, je schopná tajemství zrekonstruovat. [2]

Jinými slovy, pokud není podmnožina účastníků oprávněna tajemství zrekonstruovat, nemůže o tajemství zjistit žádné informace navíc.

Jedním ze základních požadavků na schémata pro sdílení tajemství je co nejmenší velikost jednotlivých dílů. Schémata, u kterých velikost dílů nepřekročí velikost tajemství a zároveň jsou dokonalá, se nazývají *ideální*.

Může nastat situace, kdy je třeba rozdělit větší tajemství, pro které by nebylo praktické algoritmy aplikovat, např. kvůli časové náročnosti. V tom případě je možné tajemství rozdělit na menší části a každou z částí rozdělit na díly samostatně. Výsledný díl celého tajemství vznikne zřetězením těchto menších dílů. Rozdělování příliš dlouhých tajemství je možné eliminovat úplně, např. se tajemství nejprve zašifruje bezpečnou šifrou a následně se rozdělí pouze šifrový klíč, který bývá menší velikosti než dlouhé tajemství. Zašifrované tajemství je pak možné zveřejnit a v tajnosti zůstanou pouze jednotlivé díly.

Potenciálním problémem těchto algoritmů jsou nečestní účastníci. Pokud neexistuje možnost ověření správnosti jednotlivých dílů, tak i jenom

jeden chybný díl vede k vypočtení nesprávného tajemství, případně takové tajemství ani spočítat nelze. Může se stát, že jeden účastník může tímto způsobem ostatním znemožnit získání tajemství a zároveň není možné tohoto účastníka identifikovat. Pokud při společném pokusu o rekonstrukci zjistí díly ostatních, bude moci jako jediný zrekonstruovat správné tajemství. Nečestným účastníkem může být i samotný distributor dílů, který může vybraným účastníkům úmyslně rozdat podvržené díly. Ověřování dílů lze tedy využít nejenom k odhalení poškozených nebo úmyslně pozměněných dílů účastníků, ale také k ověření, že distributor záměrně nevydává některým účastníkům nesprávné díly. Pro použití při sdílení tajemství je důležité, aby bylo možné díly ověřit, aniž by se díly musely odhalit.

Díl tajemství se zneplatní, jen pokud se zničí všechny jeho kopie a tedy přestane existovat. V případě, že by se vedl seznam zneplatněných dílů a při každém pokusu o rekonstrukci by se kontrolovalo, zda poskytnuté díly na tomto seznamu nejsou, existovalo by riziko odcizení tohoto seznamu, s jehož pomocí by útočník mohl tajemství získat. Je vhodné, aby se pravidelně všechny díly společně zneplatnily a celé tajemství se rozdělilo znovu. Případný útočník tak nebude mít neomezeně času, aby získal alespoň k dílů, a jím získané neplatné díly budou bezcenné.

2.1 Shamirův algoritmus

Shamirův algoritmus pro (k, n)-prahové schéma, jehož princip je založen na polynomiální interpolaci, byl představen již v roce 1979. Tajemství je obsaženo v polynomu, jednotlivé díly tajemství jsou body, kterými polynom prochází. S dostatečným počtem bodů je možné jednoznačně určit polynom, který všemi body prochází. [1]

2.1.1 Matematické pojmy

V této části jsou zavedeny základní pojmy, z kterých Shamirovo schéma vychází.

Definice 2.3 Množina G se nazývá *grupa*, pokud je uzavřená na binární operaci, která má pro všechny prvky G následující vlastnosti:

(1) asociativita,

(2) existence neutrálního prvku,

(3) existence inverzních prvků.

Pokud je operace i komutativní, nazýváme grupu *komutativní grupou*. [3]

Definice 2.4 *Těleso* je množina T se dvěma operacemi „$+$“ a „$\cdot$“, které splňují:

(1) T s operací „$+$“ je komutativní grupa. Neutrální prvek je označen 0.

(2) T bez 0 s operací „$\cdot$“ je grupa. Je-li i komutativní, těleso se nazývá *komutativní těleso*.

(3) Obě operace jsou distributivní. [3]

Definice 2.5 *Polynom* nad tělesem T je vzorec

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n, \tag{2.1}$$

kde $a_0, \ldots, a_n \in T$ jsou *koeficienty polynomu* a x je proměnná. Hodnota

$$P(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n \tag{2.2}$$

se nazývá *hodnota polynomu P s koeficienty $a_0, \ldots, a_n$ v bodě x*. [3]

Definice 2.6 Polynom je *nulový polynom*, pokud pro všechny jeho koeficienty $a_0, \ldots, a_n$ platí $a_i = 0, i = 0, \ldots, n$. [3]

Definice 2.7 Nechť má polynom P koeficienty $a_0, \ldots, a_n$. *Stupeň polynomu* je největší index k takový, že $a_k \neq 0$. Pokud jsou všechny koeficienty nulové, stupeň je roven -1. [3]

Definice 2.8 *Kořen polynomu P* nad tělesem T je číslo $\alpha \in T$, pro které $P(\alpha) = 0$. [3]

Polynom P stupně n s kořenem α lze tedy zapsat jako

$$P(x) = (x - \alpha)Q(x), \tag{2.3}$$

kde $Q(x)$ je polynom stupně $n - 1$.

Věta 2.1 Každý polynom stupně n nad tělesem T má v T nejvýše n různých kořenů. [3]

Důkaz. Pro stupně $n = 0$ a $n = 1$ je tvrzení platné. Pak budiž předpoklad, že tvrzení platí pro stupeň polynomu $n-1$. Polynom P stupně n s kořenem α lze zapsat dle (2.3) jako $P(x) = (x-\alpha)Q(x)$, stupeň polynomu Q je $n-1$. Polynom $P(x) = 0$, pouze pokud $x = \alpha$ nebo $Q(x) = 0$. Protože $Q(x)$ má z indukce nejvýše $n - 1$ kořenů, polynom $P(x)$ může mít nejvýše n kořenů.

Definice 2.9 *Interpolace* je metoda aproximace funkce, při které hledáme aproximační funkci takovou, aby se její funkční hodnota a prvních r derivací shodovalo s funkční hodnotou a prvními r derivacemi aproximované funkce, a to v předem daných bodech $a_0, a_1, \ldots, a_n$. Tyto body se označují jako *uzly interpolace*. [4]

Polynomiální interpolace aproximuje funkci polynomem co nejmenšího stupně. [4]

Definice 2.10 V případě, že je potřeba nalézt polynom, který se s funkcí f shoduje pouze v uzlech interpolace, tedy není vyžadována shoda i v derivacích, pak je tento polynom L_n dán rovnicí

$$L_n(x) = \sum_{i=0}^{n} f(a_i) l_i(x), \tag{2.4}$$

kde

$$l_i(x) = \frac{(x-a_0)\ldots(x-a_{i-1})(x-a_{i+1})\ldots(x-a_n)}{(a_i-a_0)\ldots(a_i-a_{i-1})(a_i-a_{i+1})\ldots(a_i-a_n)}, \quad i = 0, 1, \ldots, n. \tag{2.5}$$

Polynom ve tvaru (2.4) se nazývá *Lagrangeův interpolační polynom*. [4]

Vzorec výše se dá zkráceně zapsat jako

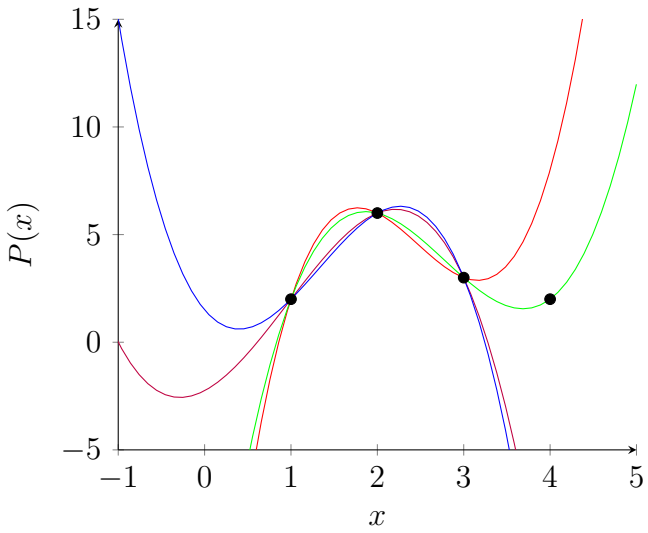
$$l_i(x) = \prod_{\substack{0 \le k \le n \\ k \ne i}} \frac{x - a_k}{a_i - a_k}. \tag{2.6}$$

Pro funkci l_i tedy platí

$$l_i(a_j) = \begin{cases} 1 & \text{pro } i = j \\ 0 & \text{pro } i \ne j \end{cases}. \tag{2.7}$$

Věta 2.2 Nechť je dáno $n+1$ bodů $[a_i, f(a_i)], i = 0, \ldots, n$ a $f(x)$ je funkce. Pak existuje právě jeden polynom $P_n(x)$ stupně nejvýše n, pro který $P_n(a_i) = f(a_i)$ pro všechna $i = 0, \ldots, n$.

Důkaz. Sporem. Nechť existují dva různé polynomy $P_n(x)$ a $Q_n(x)$ stupně nejvýše n, pro které $P_n(a_i) = f(a_i)$, resp. $Q_n(a_i) = f(a_i)$ pro všechny uzly interpolace $a_i, i = 0, \ldots, n$. Pak jejich rozdíl $R_n(x) = P_n(x) - Q_n(x)$ je polynom stupně nejvýše n. Navíc musí platit $R_n(a_i) = 0$ pro všechna $i = 0, \ldots, n$, poněvadž $P_n(a_i) = Q_n(a_i)$. Z toho vyplývá, že $R_n(x)$ má alespoň $n+1$ různých kořenů. To je možné pouze v případě, kdy je $R_n(x)$ nulový polynom, a tudíž $P_n(x)$ se rovná $Q_n(x)$.



Obrázek 2.1: Polynom stupně 3 je jednoznačně určen čtyřmi body. Stejnými třemi body prochází více polynomů stupně 3 a pouze jeden z nich prochází i čtvrtým bodem.

2.1.2 Popis algoritmu

Pro potřeby textu se předpokládá, že tajemství D lze vyjádřit jako číslo, které je prvkem komutativního tělesa $\mathbb{Z}_p$, p je prvočíslo větší než D. Následující popis vychází ze Shamirova článku [1]. Pro vytvoření (k, n)-prahového schématu je třeba vybrat polynom $Q(x)$ stupně $k-1$ nad tělesem $\mathbb{Z}_p$

$$Q(x) = a_0 + a_1 x + \ldots + a_{k-1} x^{k-1}, \tag{2.8}$$

kde $a_0 = D$ a $a_1, \ldots, a_{k-1}$ jsou náhodně zvolené koeficienty ze $\mathbb{Z}_p$. Následně se vypočtou díly $D_1, \ldots, D_n$:

$$D_1 = Q(1), D_2 = Q(2), \ldots, D_n = Q(n). \tag{2.9}$$

Pro výpočet dílů lze samozřejmě zvolit jiných n hodnot, pokud budou navzájem různé a také žádná nebude rovna nule, protože $D_0 = Q(0) = a_0 = D$. Poté každý z n účastníků obdrží tajný díl D_i.

Pomocí interpolace je možné z alespoň k bodů jednoznačně určit původní polynom, neboť je stupně nejvýše $k-1$. Se znalostí původního polynomu se pak snadno vypočítá tajemství $D = Q(0)$. [1]

2.1.3 Vlastnosti

Shamir [1] ve svém článku zmiňuje jako jednu z předních vlastností schématu, že velikost jednotlivých dílů nepřesáhne velikost sdíleného tajemství. Při výpočtu se používá $\mathbb{Z}_p$, tedy každé číslo je v rozsahu $0, \ldots, p-1$. Nechť jsou čísla reprezentována v bitech. Prvočíslo p se dá vyjádřit m bity. Pak všechna čísla z $\mathbb{Z}_p$ jsou nejvýše m-bitová, včetně tajemství D. Protože hodnota každého dílu D_i je také prvkem $\mathbb{Z}_p$, je každý díl nejvýše m-bitový a nepřekročí tak velikost tajemství D.

Mohlo by se zdát, že kromě samotného D_i je třeba uchovat v tajnosti i další informace, např. hodnoty použité k výpočtu dílu D_i. Ty ale mohou být veřejné, protože jejich výběr na hodnotě interpolačního polynomu nezávisí. Je ale potřeba umět tyto body přiřadit k odpovídajícím dílům D_i. Obdobně může být veřejná prahová hodnota k, která pouze udává počet dílů potřebných k rekonstrukci tajemství. Sice dává útočníkovi informaci, kolik dílů potřebuje, ale tu potřebují i účastníci. Kdyby byla součástí dílu, musela by být součástí všech dílů, a tudíž nemá smysl ji držet v tajnosti.

Schéma umožňuje dynamické přidávání a odebírání dílů [1]. Při zachování stejné prahové hodnoty k nic nebrání vypočtení dalších dílů, jediné omezení je dané velikostí $\mathbb{Z}_p$, protože v tomto tělese je p různých hodnot i, které je možné použít pro výpočet dílu D_i. Samotné tajemství je D_0, takže lze vytvořit až $p-1$ různých dílů. Kdyby někteří účastníci obdrželi stejné díly, mohlo by se stát, že by se při rekonstrukci sešlo více stejných dílů, počet různých dílů by nemusel být větší nebo roven k a tajemství by tak nebylo obnovitelné.

Někdy je nutné, aby některé díly účastníků měly větší váhu než jiné. Problém lze vyřešit tak, že se pro n účastníků vypočte více než n dílů a vybraní účastníci obdrží více různých dílů dle jejich důležitosti. Takové schéma ale očividně není velikostně ideální, protože díly důležitějších účastníků jsou několikanásobně větší než tajemství a navíc protože jsou jednotlivé díly samostatně použitelné, zcizení byť jen části dílů účastníka představuje velké riziko.

Schéma lze úspěšně použít při rozdělování dlouhých tajemství na části a následném rozdělení částí na díly a zřetězení dílů. Jednotlivé díly vzniklé rozdělením části tajemství nejsou větší než daná část; jejich zřetězení tedy nebude větší než původní tajemství a tento způsob zůstává z pohledu velikosti dílu ideální. Pro tento postup je klíčové použít pro každou část tajemství nový polynom. Pokud se použijí stejné koeficienty polynomu pro všechny části, tak části se stejnou hodnotou budou mít stejné hodnoty dílů a struktura tajemství bude vyzrazena.

2.1.4 Odolnost vůči útočníkům

Shamirovo schéma je *nepodmíněně bezpečné*, tzn. jeho bezpečnost nezávisí na technických nebo jiných možnostech útočníka; jeho bezpečnost vychází z matematických principů. Níže v textu bude dokázáno, že ani znalost $k-1$ dílů z k potřebných pro rekonstrukci o tajemství nedává žádné nové informace.

Dle vzorce (2.4) lze vyjádřit tajemství D následovně:

$$L_n(0) = D = \sum_{i=1}^{k} D_i l_i(0). \tag{2.10}$$

Tato rovnice se může dále rozepsat:

$$D = D_k l_k(0) + \sum_{i=1}^{k-1} D_i l_i(0). \tag{2.11}$$

Pokud útočník získá $k-1$ dílů $D_1, \ldots, D_{k-1}$, tak dle vzorce výše je pro něj hodnota $\sum_{i=1}^{k-1} D_i l_i(0)$ známá. Ze zápisu je patrné, že změnou tajemství D se změní i hodnota dílu D_k a naopak – pro každou hodnotu chybějícího dílu D_k z q možných vyjde jiná hodnota tajemství D. Proto ani znalost $k-1$ dílů nepřinese útočníkům žádné další informace o tajemství.

Útočník může získat dodatečné informace o tajemství i za předpokladu, že koeficienty $a_1, \ldots, a_n$ polynomu P nebyly vybrány náhodně a rovnoměrně. To znamená, že některé hodnoty koeficientů jsou pravděpodobnější než jiné, čehož může útočník využít při hádání polynomu P.

Při rekonstrukci tajemství toto schéma neposkytuje žádnou možnost, jak ověřit, jestli účastníci poskytli správné nebo nepoškozené díly. Stejně tak nelze ověřit, zda distributor poskytl správné díly. Návrhy, jak zaručit ověřitelnost, představili např. Feldman [5] nebo Pedersen [6]. Pedersenovo schéma je popsáno níže.

Pro schéma navržené Pedersenem je nejprve třeba vybrat prvočísla p, q a čísla $g, h \in \mathbb{Z}_q$ taková, že není známo $\log_g h$. Buď s sdílené tajemství a $t \in \mathbb{Z}_q$ náhodně zvolené číslo, pak buď $E(s,t) = g^s h^t$.

Rozdělení tajemství pro (k,n)-schéma se skládá z následujících kroků:

(1) spočtení a zveřejnění $E_0 = E(s,t)$;

(2) dle postupu popsaného v 2.1.2 zvolení polynomu F stupně $k-1$, pro který $F(0) = s$, značeno $F(x) = F_{k-1}x^{k-1}, F_{k-2}x^{k-2}, \ldots, F_1 x + s$, a vypočtení $s_i = F(i)$ pro všechna $i = 1, 2, \ldots, n$;

(3) zvolení náhodných koeficientů $G_1, G_2, \ldots, G_{k-1} \in \mathbb{Z}_q$, vypočtení a zveřejnění $E_i = E(F_i, G_i)$, $i = 1, 2, \ldots, k-1$ všem účastníkům;

(4) vytvoření polynomu $G(x) = G_{k-1}x^{k-1}, G_{k-2}x^{k-2}, \ldots, G_1x + t$, spočtení $t_i = G(i)$, $i = 1, 2, \ldots, n$ a distribuce tajných dílů (s_i, t_i) mezi účastníky.

Každý účastník ověří, zda platí

$$E(s_i, t_i) = \prod_{j=0}^{k-1} {E_j}^{i^j}, \tag{2.12}$$

protože

$$\prod_{j=0}^{k-1} {E_j}^{i^j} = \prod_{j=0}^{k-1} g^{F_j i^j} \prod_{j=0}^{k-1} h^{G_j i^j} = g^{s_i} h^{t_i} = E(s_i, t_i). \tag{2.13}$$

Tím je zaručeno, že distributor nemůže dát účastníkům neplatný díl. Ověření a rekonstrukce tajemství probíhá následovně:

(1) účastníci ověří díly ostatních dle rovnice (2.12) – mohou tak odhalit podvržený nebo poškozený díl jiného účastníka;

(2) tajemství je standardně obnoveno dle popisu v 2.1.2 a správnost může být ověřena rovností $E(s', t') = E_0$, kde s', t' jsou hodnoty vypočítané účastníky.

Věta 2.3 Pokud účastník nebo distributor neumí vypočítat $\log_g h$, nemůže vytvořit neplatný díl (s'_i, t'_i), pro který by platilo $E(s'_i, t'_i) = \prod_{j=0}^{k-1} {E_j}^{i^j}$.

Důkaz. Účastník nebo distributor spočítal podvržený díl (s'_i, t'_i), pro který platí $s'_i \neq s_i, t'_i \neq t_i$ a $E(s'_i, t'_i) = \prod_{j=0}^{k-1} E_j = E(s_i, t_i)$. Pak $g^{s'_i} h^{t'_i} = g^{s_i} h^{t_i}$ a lze snadno spočítat

$$\log_g h = \frac{s_i - s'_i}{t'_i - t_i} \pmod{q}. \tag{2.14}$$

Z předpokladu ale $\log_g h$ nelze spočítat, což je spor. [7]

Distributor i účastníci by mohli poskytnout podvržené díly, které by touto metodou nebyly odhaleny, pokud by dokázali spočítat $\log_g h$. Bezpečnost tohoto řešení tudíž není nepodmíněná, je dána obtížností výpočtu $\log_g h$. Tento problém je znám jako problém diskrétního logaritmu. V současné době neexistuje algoritmus, který by tento problém řešil na konvenčním počítači v polynomiálním čase. [8] Algoritmus, který pracuje v polynomiálním čase je ale možný s využitím kvantových počítačů. [9]

2.2 Blakleyho algoritmus

Blakleyho algoritmus pro sdílení tajemství je zástupcem algoritmů založených na geometrickém principu. Tajemství je vyjádřeno jednou souřadnicí bodu, který je průnikem nadrovin – dílů tajemství.

2.2.1 Matematické pojmy

V této části jsou zavedeny matematické pojmy potřebné pro Blakleyho algoritmus pro sdílení tajemství.

Definice 2.11 *Lineární prostor* nad tělesem $\mathbb{F}$ je neprázdná množina V uzavřená na operaci „$\oplus$“, definovanou mezi prvky V a na operaci „$\odot$“, definovanou mezi prvkem $\mathbb{F}$ a V. Prvky V jsou označovány jako *vektory*, operace tělesa $\mathbb{F}$ jsou značeny „$+$“ a „$\cdot$“. Lineární prostor musí splňovat následující vlastnosti:

(1) komutativita a asociativita operace „$\oplus$“, asociativita operace „$\odot$“;

(2) distributivita „$\odot$“ vzhledem k „$\oplus$“ i „$+$“;

(3) existence nulového vektoru;

(4) pro všechny vektory x platí $1 \odot x = x$, 1 je jednotkový prvek $\mathbb{F}$.

Definice 2.12 *Lineární podprostor* M je neprázdná podmnožina lineárního prostoru V s operacemi „$+$“ a „$\cdot$“ nad tělesem $\mathbb{F}$, pokud pro všechny $x, y \in V$ a $\alpha \in \mathbb{F}$ platí, že $x + y \in M$ a $\alpha \cdot x \in M$. [3]

Definice 2.13 Skupina vektorů $x_1, \ldots, x_n$ z lineárního prostoru V nad tělesem $\mathbb{F}$ je *lineárně závislá*, pokud existují koeficienty $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ takové, že vektor $\alpha_1 \cdot x_1 + \ldots + \alpha_n \cdot x_n$ je roven nulovému vektoru a zároveň nejsou všechny koeficienty rovny 0. Vektor ve tvaru $\alpha_1 \cdot x_1 + \ldots + \alpha_n \cdot x_n$ se nazývá lineární kombinací vektorů $x_1, \ldots, x_n$. Množina vektorů z V je lineárně závislá, pokud v této množině existuje konečný počet vektorů, které jsou lineárně závislé. Když skupina nebo množina není lineárně závislá, pak je *lineárně nezávislá*. [3]

Definice 2.14 *Báze* lineárního prostoru či podprostoru V je jeho podmnožina, která je lineárně nezávislá a zároveň množina všech lineárních kombinací jejích vektorů je rovna V.[3]

Věta 2.4 Všechny báze lineárního prostoru V mají stejný počet prvků.

Důkaz. Sporem. Existují dvě báze prostoru V $B_1 = x_1, \ldots, x_n$ a $B_2 = y_1, \ldots, y_m$, přičemž $n \neq m$. B_1 se skládá z n lineárně nezávislých vektorů. B_2 je báze s m lineárně nezávislými vektory. Z definice báze plyne, že každá množina s více než m prvky je lineárně závislá, tj. $n \leq m$. Obdobně B_1 je báze s n prvky a B_2 má m lineárně nezávislých vektorů, takže $m \leq n$. To je možné pouze pokud $n = m$, což je spor.

Definice 2.15 *Dimenze* lineárního prostoru či podprostoru V je počet prvků jeho báze. [3]

Z věty 2.4 vyplývá, že dimenze lineárního prostoru je dána jednoznačně. Navíc pro lineární prostory V nad tělesy $\mathbb{F}_p$ s konečným počtem prvků p platí, že počet prvků $V = p^m$, kde m je dimenze V. Každý prvek V se dá vyjádřit jako lineární kombinace prvků jeho báze, kterých je m. Koeficient každého z m prvků může nabývat p hodnot, celkem je p^m možných kombinací, tedy prvků V.

Definice 2.16 Jako *nadrovina* se označuje podprostor prostoru V, jehož dimenze je o jedna menší než dimenze V.

Věta 2.5 Nechť U a W jsou podprostory lineárního prostoru V. Pak jejich průnik $U \cap W$ je taktéž podprostorem V. [3]

Důkaz. Aby byl jejich průnik podprostorem, musí splňovat podmínky uvedené ve 2.12. Pokud jsou v průniku vektory x, y, musí být obsaženy v podprostoru U i W. Z toho plyne, že v U i W musí být i $x + y$, tedy je i v $U \cap W$. Obdobně, pokud je x obsaženo v průniku, v U i W se musí nacházet i $\alpha \cdot x, \alpha \in \mathbb{F}$, tudíž je opět součástí jejich průniku. Tím jsou podmínky podprostoru splněny.

Definice 2.17 *Spojení podprostorů* U, W lineárního prostoru V je množina obsahující všechny lineární kombinace všech vektorů z U i W a je značeno $U + W$. Z toho vyplývá, že je i podprostorem V. [3]

Věta 2.6 Nechť V je prostor konečné dimenze a U a W jsou jeho podprostory. Pak platí: [3]

$$dim(U) + dim(W) = dim(U + W) + dim(U \cap W). \tag{2.15}$$

Důkaz. Důkaz je pro svou délku vynechán. Uveden je např. v [3].

Z této věty vyplývá, že dimenze průniku dvou nadrovin dimenze d je rovna $d - 1$. $dim(U) + dim(W) = 2d$, $dim(U + W) \geq d + 1$, protože U a W jsou různé, báze $U + W$ musí obsahovat alespoň $d + 1$ prvků. Protože V je dimenze $d + 1$, $dim(U + W) = d + 1$ a tudíž $dim(U \cap W) = d - 1$.

Definice 2.18 *Afinní prostor* (X, V) je množina X s lineárním prostorem V nad tělesem $\mathbb{F}$ a operací „+“ mezi prvky X a V. Operace musí pro všechna $P, Q \in X$ a $v \in V$ splňovat $P + v \in X$, $P + o = P$ (o je nulový vektor), asociativitu vůči „$\oplus$“ z V a musí existovat právě jeden vektor v tak, že $P = Q + v$. Prvky X se nazývají *body*. [3]

Afinní podprostor B afinního prostoru (X, V) je množina $\{p + w : w \in W\}$, kde P je bod z X a W je lineární podprostor V. [3]

Dimenze afinního prostoru nebo podprostoru je dimenze jeho lineárního prostoru, resp. podprostoru.

Definice 2.19 *Souřadnice* vektoru $v \in V$ vzhledem k uspořádané bázi $(b_1, \ldots, b_n)$ lineárního prostoru V je uspořádaná n-tice koeficientů lineární kombinace vektorů báze, pro kterou $v = \alpha_1 \cdot b_1 + \ldots + \alpha_n \cdot b_n$. [3]

Souřadnicový systém afinního prostoru (X, V) je dvojice (O, B), $O \in X$, B je uspořádaná báze V. [3]

Souřadnice vektoru $v \in V$ vzhledem k (O, B) jsou souřadnice v vzhledem k B. [3]

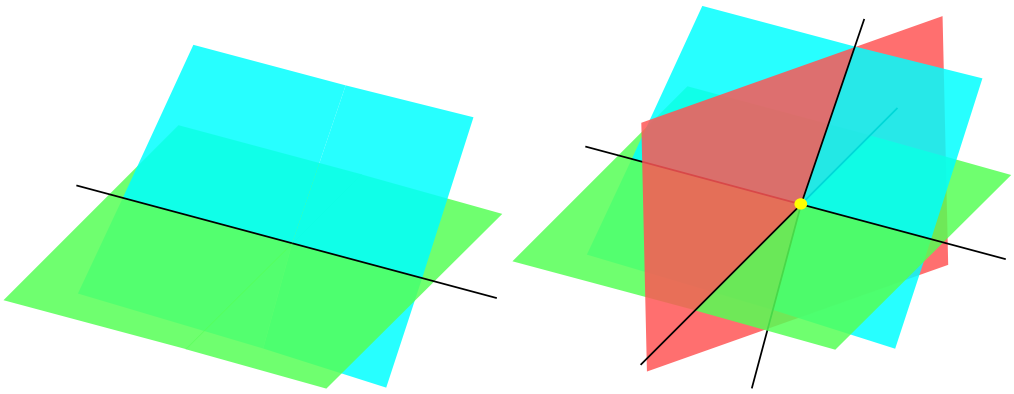
Souřadnice bodu $P \in X$ vzhledem k (O, B) jsou souřadnice vektoru $P - O$ vzhledem k B. [3]

2.2.2 Popis algoritmu

Blakley [10] popsal metodu, při níž se tajemství rekonstruuje z k nadrovin dimenze k, jejichž průnikem je jednodimenzionální podprostor – přímka. Tajemství je obsaženo ve vektoru, který tento podprostor obsahuje. Nevýhodou je, že toto řešení nedává jednoznačně tajemství; to je třeba identifikovat v až $(k + 1)^2$ možných hodnotách. Pokud je tajemstvím šifrový klíč, je možné testovat jeho správnost za pomoci vzorového šifrového textu a odpovídajícího otevřeného.

V následujícím textu je popsáno upravené (k, n)-prahové schéma, které ale vystihuje myšlenku a princip Blakleyho schématu – k rekonstrukci tajemství je třeba k afinních nadrovin dimenze $k - 1$, jejichž průnikem je podprostor dimenze 0, tedy bod, jehož jedna souřadnice je tajemství D. Výpočty jsou v afinním prostoru dimenze k, jeho lineární prostor je nad tělesem $\mathbb{Z}_p$, kde p je prvočíslo větší než D i n. Rozdělení tajemství se skládá z následujících kroků:

(1) Zvolení souřadnic bodu $d = d_1, d_2, \ldots, d_k$, kde d_1 je rovno tajemství D a hodnoty $d_2, \ldots, d_k$ jsou nenulové a náhodné;



Obrázek 2.2: V třírozměrném prostoru je průnikem dvou různoběžných rovin přímka. Průnikem tří rovin je bod, pokud je soubor jejich rovnic lineárně nezávislý.

(2) vytvoření n uspořádaných k-tic koeficientů $(a_{i1}, a_{i2}, \ldots, a_{ik})$ bez nulových prvků tak, aby každých k těchto k-tic koeficientů bylo lineárně nezávislých;

(3) spočtení n dílů $b_i = a_{i1}d_1 + a_{i2}d_2 + \ldots + a_{ik}d_k, i = 1, 2, \ldots, n$ a rozdělení mezi účastníky.

Nadrovina je vyjádřena rovnicí, jejíž množina řešení jsou body x se souřadnicemi $x_1, \ldots, x_n$, které splňují $b = a_1x_1 + a_2x_2 + \ldots + a_kx_k$. Dosazením bodu d a následným spočtením b se zajistí, že tato nadrovina bod d obsahuje. Tajemství D lze z k dílů získat vyřešením soustavy k rovnic o k neznámých

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1k}x_k &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2k}x_k &= b_2 \\ &\vdots \\ a_{k1}x_1 + a_{k2}x_2 + \ldots + a_{kk}x_k &= b_k. \end{aligned} \tag{2.16}$$

Protože jsou z předpokladu tyto rovnice lineárně nezávislé, je soustava řešitelná a má právě jedno řešení – bod d.

2.2.3 Vlastnosti

Dílem i-tého účastníka je dvojice $(b_i, a_{i1}d_1 + a_{i2}d_2 + \ldots + a_{ik}d_k)$. Z této dvojice stačí držet v tajnosti pouze b_i, druhá část je bez první bezcenná. Protože schéma je v tělese $\mathbb{F}_p$, platí, že $0 < b_i < p$. Pro tajemství D platí

totéž. Prvočíslo p se dá vyjádřit m bity, pak i D a b_i se dá vyjádřit nejvýše m bity. Velikost tajného dílu tak nepřesáhne velikost tajemství.

V tomto schématu je možné dynamicky přidávat díly tajemství. Pro přidání stačí nalézt další nadrovinu, která prochází bodem d. Těchto nadrovin je ale samozřejmě konečný počet a pro relativně velká k vůči p nemusí už žádná další existovat.

2.2.4 Bezpečnost

Vlastnosti a bezpečnost schématu ovlivňuje výběr nadrovin a tedy koeficientů $a_{ij}, 1 \leq i \leq n, 1 \leq j \leq k$. Lineární nezávislost libovolných k uspořádaných k-tic koeficientů zaručuje, že libovolná soustava k rovnic o k neznámých má právě jedno řešení. Pro soustavu méně než k takových rovnic existuje více řešení.

Pokud útočník získá m dílů, $m < k$, jejich průnikem dostane podprostor dimenze $k - m$, ve kterém bod s tajemstvím leží. Tento podprostor obsahuje p^{k-m} různých bodů. Tajemstvím ale není bod, ale pouze jedna jeho souřadnice, která může stále nabývat všech p hodnot.

Je-li průnikem podprostor, jehož všechny body mají shodnou hodnotu první souřadnice, tajemství může být zjistitelné i za pomoci méně než k dílů. Jako příklad poslouží $(3, n)$ schéma nad $\mathbb{Z}_5$, tajemstvím je 1 a bod $d = (1, 4, 2)$. Útočník získal 2 následující díly: $2x_1 + 3x_2 + x_3 = 1$ a $x_1 + 3x_2 + x_3 = 0$. Z těchto rovnic lze jednoduše spočíst $x_1 = 1$. x_2 ani x_3 není možné jednoznačně určit, ale klíčovou informaci, tj. tajemství, útočník získal. Aby bylo schéma dokonalé, je nutné, aby byly vybrány nadroviny, jejichž průnikem tyto podprostory nevznikají. V ostatních případech ani znalost $k - 1$ dílů nedává žádné nové informace, souřadnice s tajemstvím stále může být libovolná z p různých hodnot. V tomto případě je schéma dokonalé.

Praktický příklad výběru koeficientů nadrovin pro toto schéma popsali Hei et al. [11] Navrhují zvolit koeficienty dle rovnice $a_{pq} = a_{p-1q-1} + a_{p-1q}$, přičemž $1 < p \leq n, 1 < q \leq k, a_{1q} = 1$. Jako další možnost výběru koeficientů uvádí $a_{pq} = a_{pq-1} + a_{p-1q}$, přičemž $p > 1, q > 1, a_{1q} = 1, a_{q1} = 0$. Obě tyto varianty splňují podmínku, že libovolných k k-tic $(a_{i1}, a_{i2}, \ldots, a_{ik})$ je lineárně nezávislých.

Ověřitelnost tohoto schématu lze opět zajistit pomocí metody využívající problém diskrétního logaritmu. U schématu založeném na průniku nadrovin takovou metodu popsali např. Yang et al. [12]

2.3 Asmuthův–Bloomův algoritmus

Tento algoritmus pro (k, n)-schéma, založený na čínské větě o zbytcích (2.7), představili Asmuth s Bloomem [13] v roce 1983. O rok dříve zveřejnil Mignotte [14] podobné, jednodušší schéma. Obě schémata využívají speciální sekvence vzájemně nesoudělných čísel. Goldreich [15] naproti tomu uvažuje ve svém schématu, taktéž vycházejícím z čínské věty o zbytcích, pouze prvočísla.

2.3.1 Matematický základ

Cílem této sekce je zavést pojmy nutné k definování samotného Asmuthova–Bloomova algoritmu.

Definice 2.20 Buď $a, b, m \in \mathbb{Z}, m \neq 0$. a je *kongruentní s b modulo m*, pokud m dělí $b - a$. Tento vztah je značen $a \equiv b \pmod{m}$. [16]

Definice 2.21 Nechť $a, k, m \in \mathbb{Z}, m \neq 0$. *Zbytková třída modulo m* je množina všech čísel n, pro které $n \equiv a \pmod{m}$, neboli $n = a + km$. [16]

Věta 2.7 (Čínská věta o zbytcích) Buďte čísla $m_1, m_2, \ldots, m_t$ taková, že každá dvě jsou navzájem nesoudělná, $m = m_1 m_2 \ldots m_t$ a nechť $b_1, b_2, \ldots, b_t$ jsou celá čísla. Potom následující soustava kongruencí

$$\begin{aligned} x &\equiv b_1 \pmod{m_1}, \\ x &\equiv b_2 \pmod{m_2}, \\ &\vdots \\ x &\equiv b_t \pmod{m_t}. \end{aligned} \tag{2.17}$$

má vždy řešení, které je v modulo m jednoznačné. [16]

Důkaz. Existenci řešení se dokáže následovně: buď $n_i = m/m_i . m_i$ a n_i jsou nesoudělná. Pak existují čísla r_i a s_i taková, že $r_i m_i + s_i n_i = 1$. Je zřejmé, že $s_i n_i \equiv 1 \pmod{m_i}$ a $s_i n_i \equiv 0 \pmod{m_j, j \neq i}$. Pro přehlednost

$$x_0 = \sum_{i=1}^{t} b_i s_i n_i. \tag{2.18}$$

Z toho plyne, že $x_0 \equiv b_i s_i n_i \pmod{m_i}$, a tudíž i $x_0 \equiv b_i \pmod{m_i}$. Řešení soustavy existuje a je jím x_0. [16]

Důkaz jednoznačnosti řešení sporem: buď x_1 další řešení soustavy. Pak $m1, m2, \ldots, m_t$ dělí $x_1 - x_0$, m dělí $x_1 - x0$. x_1 a $x0$ se tedy liší přesně o m, tj. patří do stejné zbytkové třídy a řešení je v modulo m jednoznačné. [16]

2.3.2 Popis algoritmu

K rozdělení tajemství D a vytvoření (k, n)-prahového schématu se podle Asmutha a Blooma [13] vyberou následující čísla:

(1) n vzájemně nesoudělných čísel $m_1, m_2, \ldots, m_n$ takových, že $m_1 < m_2 < \ldots < m_n$;

(2) p, které je s nimi taktéž nesoudělné a zároveň $0 \leq D < p$;

(3) čísla výše musí vyhovovat nerovnosti $\prod_{i=1}^{k} m_i > p \prod_{i=1}^{k-1} m_{n-i+1}$;

(4) číslo A, $D + Ap < M$, $M = m_1 m_2 \ldots m_k$.

To postačuje k vytvoření jednotlivých dílů. Vypočtou se hodnoty $y_i \equiv D + Ap \pmod{m_i}$ pro všechna $i, 1 < i < n$. Dílem tajemství je dvojice (y_i, m_i). Tím se mezi účastníky rozdělí n rovnic

$$\begin{aligned} D + Ap &\equiv y_1 \pmod{m_1} \\ D + Ap &\equiv y_2 \pmod{m_2} \\ &\vdots \\ D + Ap &\equiv y_n \pmod{m_n}. \end{aligned} \tag{2.19}$$

Pokud je známo k dílů $y_{i_1}, y_{i_2}, \ldots, y_{i_k}$, tak z čínské věty o zbytcích (2.7) plyne, že v modulo $m_{i_1} m_{i_2} \ldots m_{i_k}$ je $D + Ap$ dáno jednoznačně. Podmínky výše zaručují, že $D + Ap$ je menší než součin k nejmenších m_i, tj. součin k libovolných m_i je větší než $D + Ap$, které se tak v modulu M rovná původnímu $D + Ap$. Tajemství D se posléze spočte jako $D = D + Ap \pmod{p}$.

2.3.3 Vlastnosti

Velikosti dílů se u tohoto schématu mohou lišit. Tajemství D je nejvýše rovno $p - 1$, p se dá vyjádřit m bity. Pro díl y_i platí $0 \leq y_i < m_i$. Největší z m_i je z předpokladu m_n, tudíž každé $y_i < m_n$. Číslo m_n lze vyjádřit $m + l$ bity, $0 \leq l$. Protože $p < m_n$, díl tajemství může být o l bitů větší než sdílené tajemství, proto nemůže být toto schéma ideální.

Stačí držet v tajnosti hodnotu y_i, m_i může být veřejné, protože nedává žádnou další informaci. Hodnota p může být taktéž veřejná, protože jinak by musela být součástí každého dílu, popř. uschována na jednom místě jiným způsobem; pak by ale její ztráta znemožnila tajemství zrekonstruovat.

Schéma není vhodné pro dynamické vkládání nových dílů – pro nový díl by bylo nutné najít takové m_{n+1}, které by s p a $m_1, m_2, \ldots, m_n$ splňovalo

podmínky popsané ve 2.3.2, což může být obtížné, ne-li nemožné. Pokud by nové m_{n+1} bylo menší než m_k, změnila by se hodnota M a nemusela by platit nerovnost $D + 1p < M$.

Asmuth s Bloomem [13] neuvedli žádný algoritmus, jak najít čísla, vyhovující podmínce ve 2.3.2. Quisquarter et al. [17] uvažují sekvenci po sobě jdoucích prvočísel, ale ani tato metoda nemusí vždy vytvořit sekvenci s požadovanými vlastnostmi, obzvlášť pro menší prvočísla.

2.3.4 Bezpečnost

Pokud útočník získá $k-1$ dílů $y_{i_1}, y_{i_2}, \ldots, y_{i_{k-1}}$, zná i $N = m_{i_1} m_{i_2} \ldots m_{i_{k-1}}$ a může určit $D + Ap \pmod N$. Podmínka $\prod_{i=1}^{k} m_i > p \prod_{i=1}^{k-1} m_{n-i+1}$ říká, že součin $k-1$ největších m_i není větší než součin k nejmenších m_i, a protože $D + Ap < M$, není $D + Ap$ v modulo N dáno jednoznačně. Navíc $M/N > p$ a N a p jsou dle předpokladu nesoudělné, proto všechna možná $n_i \leq M$ vyhovující $n_i \equiv D + Ap \pmod N$ pokryjí všechny zbytkové třídy modulo p tak, že v každé třídě bude nanejvýš o jeden prvek více nebo méně než v ostatních třídách. [13] S $k-1$ díly se tedy nezíská o tajemství žádná informace, ale rozložení četnosti výskytu možných hodnot tajemství není úplně rovnoměrné; proto se toto schéma nedá považovat za dokonalé.

Schéma není odolné vůči nečestným účastníkům. Asmuth s Bloomem [13] navrhli pozměněnou variantu schématu, která dokáže odhalit až $r - 1, r < n$ spolupracujících nečestných účastníků. Tato metoda využívá skutečnosti, že pokud m_i a m_j nebudou nesoudělné, jejich největší společný dělitel je značen q_{ij}, tak díly y_i a y_j budou splňovat $y_i \equiv y_j \pmod{q_{ij}}$. V případě chyby toto nebude platit. Pro odhalení až $r-1$ spolupracujících nečestných účastníků je potřeba $\binom{n}{r}$ hodnot q. Qiong et al. [7] označili tuto metodu jako nerealizovatelnou a navrhli jinou metodu ověření správnosti dílu. Tato metoda je opět dosti podobná Pedersenově [6] metodě, popsané u Shamirova algoritmu ve 2.1.4. Iftene [18] ve své práci popsal ověřitelné schéma založené na Mignottově [14] schématu, které se dá aplikovat i na Asmuthovo–Bloomovo schéma. I zde je jeho bezpečnost založena na problému diskrétního logaritmu.

Po úpravě lze dosáhnout schématu, kde mají jednotlivé díly účastníků různé váhy – např. lze účastníkům přiřadit různě velké moduly m_i dle jejich důležitosti. [19] Oproti Shamirově schématu tak má každý účastník pouze jeden díl.

Kapitola 3

Implementace

Tato kapitola nejdříve popisuje implementaci Shamirova a Asmuthova–Bloomova algoritmu. Následuje popis aplikace, která tyto algoritmy využívá.

Algoritmy i aplikace jsou implementované v jazyce C a využívají knihovnu OpenSSL. Aplikace byla napsána a testována na operačním systému Windows, ale funkčnost byla ověřena i na operačním systému Xubuntu.

Pro tajemství a části dílů je použit typ `BIGNUM` z této knihovny, který reprezentuje čísla neomezené délky. Knihovna taktéž obsahuje funkce implementující aritmetiku těchto čísel. Šifrování a dešifrování je realizováno pomocí funkcí `EVP` z OpenSSL, které implementují kryptografické funkce.

Jeden díl tajemství je reprezentován strukturou `share_t`, která obsahuje položky `x` a `y`. Struktura je definována v hlavičkovém souboru `sss.h`. Funkce pro ukládání dílů tajemství do souborů `sss_save_shares`, načítání dílů ze souborů `sss_load_shares` a funkce pro ukládání a načítání souboru s veřejně dostupnými informacemi `sss_save_pubfile` a `sss_load_pubfile` jsou definovány v souboru `sss.c`.

Struktura dílů ukládaných v `sss_save_shares` je následující:

(1) 4 bajty označující počet následujících bajtů položky `x` v `share_t`;

(2) bajty položky `x`;

(3) 4 bajty určující počet následujících bajtů položky `y` v `share_t`;

(4) bajty položky `y`.

Soubor s veřejnými informacemi se skládá z následujících částí:

(1) 1 bajt reprezentující typ algoritmu pro sdílení tajemství;

(2) 4 bajty určující k v (k, n)-prahovém schématu;

(3) 4 bajty udávající počet následujících bajtů inicializačního vektoru;

(4) bajty inicializačního vektoru.

Výpis kódu 1 Struktura reprezentující jeden díl tajemství

```
typedef struct share_t {  //representing one secret share
    BIGNUM * x;
    BIGNUM * y;
} share_t;
```

3.1 Implementace Shamirova algoritmu

Funkce implementující Shamirovo schéma se nachází v souboru `shamir.c`. Shamirův algoritmus pro (k, n)-schéma je implementován nad tělesem $\mathbb{Z}_p$, kde p je prvočíslo. K rozdělení tajemství slouží funkce `sh_share_secret` rozdělující tajemství na n dílů, jejím protějškem rekonstruujícím tajemství z dílů je funkce `sh_reconstruct_secret`.

Rozdělení tajemství na díly se skládá ze dvou kroků. Nejprve se pomocí funkce `sh_make_coef` vytvoří pole koeficientů $a_1, a_2, \ldots, a_{k-1}$ polynomu stupně $k - 1$. Koeficienty jsou pseudonáhodně generovány knihovní funkcí `BN_rand_range`. Poté se ve funkci `sh_make_shares` vyplní nově vytvořené pole dílů . Díly tajemství jsou body se souřadnicemi x_i, y_i. Hodnoty x_i jsou čísla $1, 2, \ldots, n$ a odpovídající y_i se získá pomocí funkce `sh_compute_point`, která počítá hodnotu polynomu v bodě x_i jako $y_i = \sum_{j=0}^{k-1} a_j x_i^j \pmod p$, její složitost je $O(n)$.

Při rekonstrukci tajemství funkcí `sh_reconstruct_secret` se pouze zavolá funkce `sh_interpolation_mod`, která implementuje výpočet $L_n(0)$ Lagrangeova interpolačního polynomu modulo p rovnicí (2.4). Implementace má složitost $O(n^2 \cdot \log(p)^2)$, protože knihovní funkce pro výpočet modulární multiplikativní inverze `BN_mod_inverse` používá rozšířený Euklidův algoritmus se složitostí $O(\log(p)^2)$. [20]

Algoritmus 2 Výpočet bodu polynomu z jedné souřadnice

procedure COMPUTE POINT($x, poly, k, mod$)
Vstup: souřadnice **x**; pole koeficientů polynomu **poly[0,...,k-1]**; počet koeficientů **k**; modulo **mod**.
Výstup: Souřadnice y bodu [x,y] v proměnné **res**.
$res \leftarrow 0$
for $i \leftarrow k-1$ downto 0 **do**
$res \leftarrow res \cdot x \pmod{mod}$
$res \leftarrow res + poly[i] \pmod{mod}$
end for
end procedure

Algoritmus 3 Lagrangeova interpolace v bodě 0 modulo mod

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

3.2 Implementace Asmuthova–Bloomova algoritmu

Funkce implementující Asmuthův–Bloomův algoritmus pro sdílení tajemství jsou definovány v souboru `asmuth-bloom.c`. Rozdělení tajemství na pole dílů pro (k, n)-schéma realizuje funkce `ab_share_secret`, následná rekonstrukce tajemství z dílů je ve funkci `ab_reconstruct_secret`.

Při rozdělování tajemství funkcí `ab_share_secret` se jako první volá funkce `ab_gen_primes` generující vzájemně nesoudělná čísla splňující pod-

mínku ve 2.3.2. Jako nesoudělná čísla je použita sekvence po sobě následujících předpokládaných prvočísel, která projdou testy prvočíselnosti funkce `BN_is_prime_ex`, která provádí Millerův–Rabinův test. Čísla se poté testují, zda splňují podmínku. Pokud ne, celý krok se znovu opakuje. Složitost jednoduché implementace Millerova–Rabinova testu je $O(c \cdot \log(n)^3)$, c je počet iterací algoritmu. [21] Celková složitost je $O(b \cdot (k^2 \cdot c \cdot log(n)^3 + 2k))$, b je počet opakování potřebných ke splnění podmínky algoritmu. Jednotlivé díly tajemství jsou vytvořeny ve funkci `ab_make_shares`, která dle postupu ve 2.3.2 pseudonáhodně zvolí A a následně spočte díly modulo m_i.

Algoritmus 4 Generování prvočísel pro Asmuthův–Bloomův algoritmus

procedure genPrimes(k, n, mod)
  Vstup: počet dílů nutných pro rekonstrukci tajemství **k**, počet dílů **n**, modulo **mod**.
  Výstup: pole **n+1** prvočísel **primes[0,...,n]**.
  $primes[0] \leftarrow mod$
  $prime \leftarrow mod \cdot 2 + 1$
  repeat
    for $i \leftarrow 1$ to n **do**
      $prime \leftarrow prime + 2$
      while IsPrime($prime$) = false **do**
        $prime \leftarrow prime + 2$
      end while
      $primes[i] \leftarrow prime$
    end for
    $left \leftarrow 1$
    $right \leftarrow primes[0]$
    for $i \leftarrow 1$ to k **do**
      $left \leftarrow left \cdot primes[i]$
    end for
    for $i \leftarrow n$ downto $n - k + 2$ **do**
      $right \leftarrow right \cdot primes[i]$
    end for
  until $left > right$
end procedure

Rekonstrukce tajemství z dílů ve funkci `ab_reconstruct_secret` probíhá následovně: nejprve se za pomoci čínské věty o zbytcích spočítá $s_2 = s + Am_0$, které je z předpokladu v modulu součinů modulů dílů dáno jednoznačně. Čínská věta o zbytcích je implementována ve funkci `ab_crt` postupem použitým v konstrukčním důkazu ve 2.3.1, který má i s výpočtem

multiplikativní inverze složitost $O(n \cdot \log(mod)^2)$. Tajemství se poté získá jako $s_2 \pmod{m_0}$.

Algoritmus 5 Čínská věta o zbytcích

procedure CRT(S, k)
Vstup: pole dvojic **S[0,...,k-1]**; každá dvojice se skládá z modula **S[i].y** a zbytku po dělení tímto modulem **S[i].x**; počet dvojic **k**.
Výstup: Hodnota **res** daná jednoznačně v součinu všech modulů **S[i].y**.
$sum \leftarrow 0$
$prod \leftarrow 1$
for $i \leftarrow 0$ to $k-1$ **do**
$prod \leftarrow prod \cdot S[i].y$
end for
for $i \leftarrow 0$ to $k-1$ **do**
$res \leftarrow \frac{prod}{S[i].y}$
$res \leftarrow res \cdot$ ModInv$(res, S[i].y) \cdot S[i].x$
$sum \leftarrow sum + res$
end for
$res \leftarrow sum \pmod{prod}$
end procedure

3.3 Implementace aplikace

Aplikace umožňuje zašifrovat zadaný soubor symetrickou šifrou a pomocí algoritmů implementovaných v předchozí části rozdělit šifrovací klíč na předem daný počet dílů. Šifrovací klíč je aplikací vygenerován, poté rozdělen a použit na šifrování souboru. Kromě dílů je vygenerován i soubor s veřejnými informacemi, ve kterém se nachází inicializační vektor a k z použitého (k, n)-schématu. Následně je možné pomocí aplikace z množiny dílů tajemství, veřejného souboru a zašifrovaného souboru klíč rekonstruovat a soubor dešifrovat.

Pro šifrování a dešifrování souboru byla vybrána symetrická šifra AES v módu CBC s 256-bitovým klíčem. Jako modulo je použito prvočíslo $2^{257} - 93$ [22], které bezpečně pokryje všechny možné hodnoty klíče. V aplikaci je také omezena nejvyšší možná hodnota k i n, a to $k = n = 100$. Tyto hodnoty jsou dány konstantami `MAX_K`, resp. `MAX_N` v souboru `sss.h`. Lze tedy dosáhnout nejvýše $(100, 100)$ schématu, což je pro většinu případů dostačující.

3.3.1 Šifrování a dešifrování souboru

Funkce pro šifrování a dešifrování souboru zvolenou symetrickou šifrou jsou implementovány v souboru `enc-dec.c`. Generování šifrovacích klíčů a inicializačního vektoru probíhá ve funkci `generate_key_iv`. `RAND_bytes` vygeneruje požadovaný počet pseudonáhodných bajtů, jejichž počet je určen `EVP_CIPHER_key_length`, resp. `EVP_CIPHER_iv_length`, které vrací počet bajtů klíče, resp. inicializačního vektoru zadané šifry.

Výpis kódu 6 Generování pseudonáhodného šifrovacího klíče a inicializačního vektoru

```
int generate_key_iv(unsigned char *key, unsigned char *iv,
                    const EVP_CIPHER *cipher){
/* key and iv are sufficent buffers for cipher key and iv
 */
//generating key
if(RAND_bytes((key), EVP_CIPHER_key_length(cipher))!=1)
  return ERROR_ENC_DEC;
//generating iv
if(RAND_bytes((iv), EVP_CIPHER_iv_length(cipher))!=1)
  return ERROR_ENC_DEC;
return SUCCESS;
}
```

K zašifrování zadaného souboru slouží funkce `encrypt_file_to_file`. Typ symetrické šifry, šifrovací klíč a inicializační vektor se nastaví funkcí `EVP_EncryptInit_ex`. Samotné šifrování souboru probíhá ve smyčce, kde se po částech načítají data ze vstupního souboru, která se šifrují funkcí `EVP_EncryptUpdate_ex` a ukládají do výstupního souboru. Poslední blok dat je zašifrován funkcí `EVP_EncryptFinal_ex`.

Dešifrování souboru je velmi podobné šifrování, pouze se místo funkcí typu `EVP_Encrypt` používají funkce pro dešifrování `EVP_DecryptInit_ex`, `EVP_DecryptUpdate_ex` a `EVP_DecryptFinal_ex`.

3.3.2 Konzolová aplikace

Funkce implementující konzolovou aplikaci `keyShare` se nachází v souboru `keyShare.c`.

Logika zpracování vstupních parametrů aplikace se nachází ve funkci `option_parser`. Syntax parametrů pro zašifrování souboru a rozdělení klíče je následující:

Výpis kódu 7 Zjednodušená ukázka smyčky pro zašifrování souboru

```
...
/* ctx is cipher context, cipher is used cipher, key and iv
 * are encryption key and initialization vector
 * srcFile is file to be encrypted, destFile is output file
 * inBuffer/outBuffer is buffer to store data to encrypt/
 * encrypted data
 */
//Initialization
//cipher context init
res = EVP_EncryptInit_ex(ctx, cipher, NULL, key, iv);
dataLen = fread(inBuffer, 1, sizeof(inBuffer), srcFile);
//Encryption
while(dataLen){  //while there are data to encrypt
  //encrypting data
  EVP_EncryptUpdate(ctx, outBuffer, &encLen, inBuffer,
                    dataLen);
  //writing encrypted data to file
  fwrite(outBuffer, 1, encLen, destFile)
  //reading new data
  dataLen = fread(inBuffer, 1, sizeof(inBuffer), srcFile);
}
//last block of data
EVP_EncryptFinal_ex(ctx, outBuffer, &encLen);
//writing last block of data
fwrite(outBuffer, 1, encLen, destFile)
...
```

```
keyShare.exe -e -sh|-ab k n file
```

Jako první je přepínač `-e` symbolizující zašifrování a rozdělení, poté `-sh` nebo `-ab` představující Shamirův nebo Asmuthův–Bloomův algoritmus. Následují čísla k a n pro (k, n)-prahové schéma a jako poslední je jméno souboru, který má být zašifrován. Zašifrování souboru `soubor` a rozdělení klíče Shamirovým $(5, 10)$-schématem se tedy provede příkazem:

```
keyShare.exe -e -sh 5 10 soubor
```

Díly tajemství budou zapsány do souborů se jménem `SHshare_x`, resp. `ABshare_x`, kde `x` je zaměněno za pořadové číslo dílu. Veřejné informace (typ použitého algoritmu, hodnota k a inicializační vektor) jsou uloženy do

souboru pojmenovaného `SHpubFile`, resp. `ABpubFile`. Zašifrovaný soubor má stejný název jako původní, pouze je na konci doplněný o řetězec `_enc`.

Pro rekonstrukci klíče a dešifrování souboru jsou aplikaci předány následující parametry:

```
keyShare.exe -d pubFile share1 share2 ... file
```

Parametr `-d` značí dešifrování a rekonstrukci, následuje jméno souboru s veřejnými informacemi, poté jména souborů s díly, které budou použity pro rekonstrukci a jako poslední je uveden název zašifrovaného souboru. Dešifrovaný soubor bude uložen do souboru s původním názvem doplněným na konci o řetězec `_dec`. Dešifrování souboru `soubor_enc` z dílů `share_01`, `share_05` a `share_03` a veřejného souboru `pubFile` se provede následujícím příkazem:

```
keyShare.exe -d pubFile share_01 share_05 share_03 soubor_enc
```

Funkce `enc_and_share` má na starosti zašifrování souboru a rozdělení tajemství. Nejprve vygeneruje klíč a inicializační vektor voláním funkce `generate_key_iv`, poté šifrovací klíč převede na typ `BIGNUM` a zavolá odpovídající funkci pro rozdělení tajemství – `sh_share_secret`, pro Shamirův algoritmus, resp. `ab_share_secret` pro Asmuthův–Bloomův, a uloží díly do souborů voláním funkce `sss_save_shares`. Nakonec soubor zašifruje funkcí `encrypt_file_to_file` a uloží veřejné informace do souboru funkcí `sss_save_pubfile`.

Rekonstrukci klíče a dešifrování souboru realizuje funkce `rec_and_dec`. Nejprve získá informace z veřejného souboru pomocí `sss_load_pubfile`, poté načte zadané díly funkcí `sss_load_shares` a pomocí všech načtených dílů rekonstruuje šifrovací klíč funkcí `sh_reconstruct_secret`, resp. `ab_reconstruct_secret`. Získaný klíč je převeden do binární podoby a voláním funkce `decrypt_file_to_file` je soubor dešifrován.

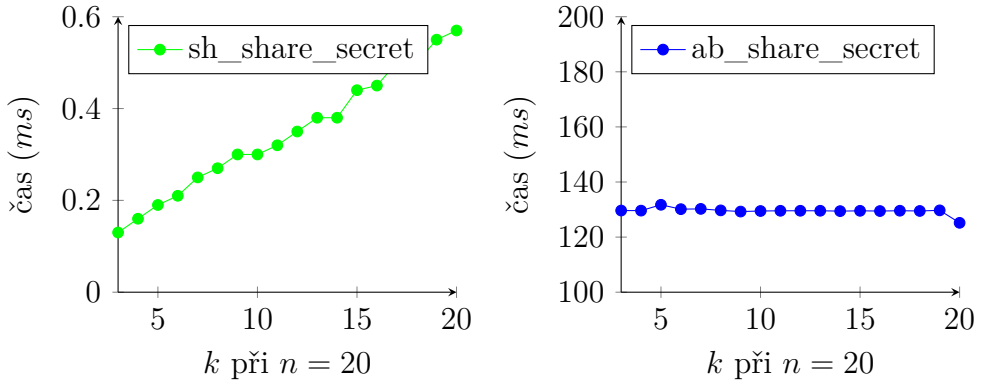
3.4 Testování rychlosti algoritmů

V této části je otestována a porovnána rychlost implementace Shamirova a Asmuthova–Bloomova algoritmu, konkrétně funkce `sh_share_secret`, `sh_reconstruct_secret`, `ab_share_secret` a `ab_reconstruct_secret`.

Algoritmy byly testovány na počítači s procesorem Intel Core i5 750 2.67 GHz a s operačním systémem Windows 10 Pro 64-bit, version 1803. Čas průběhu funkcí byl měřen knihovní funkcí `time`. Pro všechny testy bylo použito modulo $2^{257} - 93$, které je použito i v aplikaci `keyShare`, a délka testovaných tajemství byla vždy 256 bitů.

Testy funkcí `sh_share_secret` a `ab_share secret` pro (k, n)-schéma byly provedeny nejprve pro $k = 3, 4, \ldots, 20$ při $n = 20$ a poté pro $k = 5, 10, \ldots, 100$ při $n = 100$. Výsledné hodnoty v grafu jsou aritmetickým průměrem sta měření. Následně byly testovány měnící se hodnoty n při stejném k, konkrétně $n = 10, 15, \ldots, 100$ pro $k = 10$.

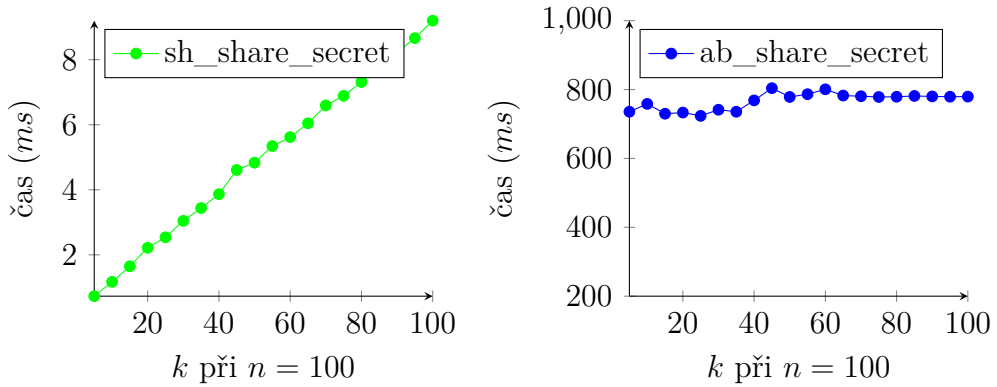
Čas nutný pro rozdělení tajemství funkcí `sh_share_secret` se úměrně zvyšoval s rostoucím k, protože bylo nutné generovat polynomy vyšších stupňů. Funkce `ab_share_secret` má vysokou režii při generování n prvočísel a čas nutný pro ověření podmínky, ve které se vyskytuje k, byl zanedbatelný. V porovnání s `sh_share_secret` byl pro měřené hodnoty řádově tisíckrát pomalejší.



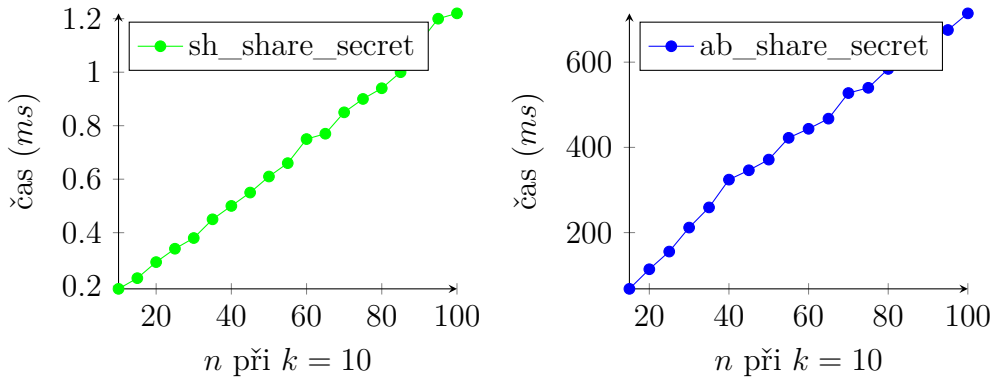
Obrázek 3.1: Časová náročnost jednoho průběhu funkcí rozdělujících tajemství sh_share_secret a ab_share_secret pro (k, n)-schéma a různá k při $n = 20$

Dalším testem byl test rozdělení tajemství pro $n = 10, 15, \ldots, 100$, přičemž $k = 10$. Při stejném k a rostoucím n se čas u obou funkcí úměrně zvyšoval s rostoucím n, neboť u funkce `ab_share_secret` bylo nutné generovat n prvočísel a funkce `ab_share_secret` sice stále vytvářela polynom stupně $k - 1$, ale bylo nutné dopočítat n bodů, tedy dílů.

Funkce `sh_reconstruct_secret` a `ab_reconstruct_secret` byly testovány pro $nS = k = 3, 4, \ldots, 20$ při $k = 20$ a pro $nS = k = 5, 10, \ldots, 100$ pro $n = 100$, přičemž nS značí počet zadaných dílů. Z naměřených hodnot druhého testu je patrná polynomiální složitost rekonstrukce tajemství obou funkcí. V tomto testu byla `ab_reconstruct_secret` pro měřené hodnoty řádově desetkrát rychlejší, což odpovídá předpokládané složitosti. Testy pro různé hodnoty nS při stejném k nebyly provedeny, protože aplikace použije k rekonstrukci všechny zadané díly a výsledky by se nelišily od předchozího testu.

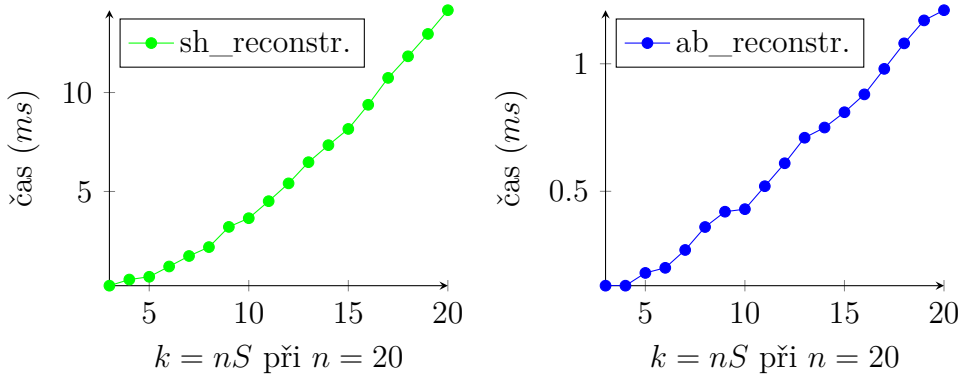


Obrázek 3.2: Časová náročnost jednoho průběhu funkcí rozdělujících tajemství sh_share_secret a ab_share_secret pro (k, n)-schéma při $n = 100$ a různých hodnotách k.

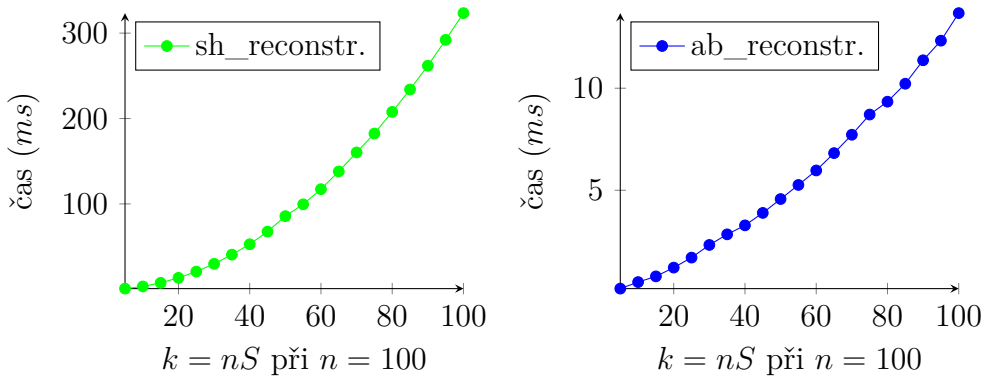


Obrázek 3.3: Čas průběhu funkcí rozdělujících tajemství sh_share_secret a ab_share_secret pro (k, n)-schéma s různými hodnotami n při $k = 10$

V žádném z testů nepřekročilo rozdělení ani rekonstrukce tajemství jednu sekundu. Pro rozdělení větších tajemství by už ale bylo vhodné tajemství rozdělit na více částí a postupně rozdělovat menší části. Další možností je optimalizace algoritmů, např. pomocí paralelizace. Rozdělení tajemství funkcí `ab_share_secret` by se dalo značně zrychlit použitím předpočítaných sekvencí nesoudělných čísel pro dané modulo. Pro praktické využití ale nejsou optimalizace nutné, protože je vždy možné dlouhé tajemství zašifrovat bezpečnou šifrou a rozdělit pouze použitý šifrovací klíč.



Obrázek 3.4: Časová náročnost průběhu funkcí rekonstruujících tajemství sh_reconstruct_secret a ab_reconstruct_secret pro (k, n)-schéma při $n = 20$ a různých hodnotách k, přičemž počet dílů použitých k rekonstrukci je roven k.



Obrázek 3.5: Časová náročnost jednoho průběhu funkcí rekonstruujících tajemství sh_reconstruct_secret a ab_reconstruct_secret pro (k, n)-schéma při $n = 100$ a různých hodnotách k, přičemž počet použitých dílů pro rekonstrukci je roven k.

Závěr

Práce se zabývala algoritmy pro sdílení tajemství. Z algoritmů pro sdílení tajemství byly popsány Shamirův, Blakleyho a Asmuthův–Bloomův algoritmus. U všech byly popsány jejich matematické principy, vlastnosti a byla provedena analýza jejich bezpečnosti. V další části byl implementován Shamirův a Asmuthův–Bloomův algoritmus a obě implementace byly otestovány a porovnány. V poslední části byla vytvořena aplikace pro sdílení symetrických šifrovacích klíčů implementující tyto algoritmy. Aplikace umožňuje zašifrovat zadaný soubor šifrou AES s 256-bitovým klíčem v módu CBC, šifrovací klíč je náhodně vygenerován a rozdělen na předem daný počet částí. Poté je možné ze zašifrovaného souboru, souboru s veřejnými informacemi a množiny částí zrekonstruovat šifrovací klíč a soubor dešifrovat. Program by bylo možné dále vylepšit např. sloučením veřejných informací a zašifrovaných dat do jednoho souboru nebo implementací ověřování správnosti dílů tajemství.

Bibliografie

1. SHAMIR, Adi. How to Share a Secret. *Communications of the ACM* [online]. 1979, roč. 22, č. 11, s. 612–613 [cit. 2019-03-30]. ISSN 0001-0782. Dostupné z DOI: 10.1145/359168.359176.
2. BRICKELL, Ernest F.; DAVENPORT, Daniel M. On the classification of ideal secret sharing schemes. *Journal of Cryptology* [online]. 1991, roč. 4, č. 2, s. 123–134 [cit. 2019-03-15]. ISSN 1432-1378. Dostupné z DOI: 10.1007/BF00196772.
3. OLŠÁK, Petr. *Lineární algebra* [online]. 2. vyd. 2010 [cit. 2019-03-30]. Dostupné z: http://petr.olsak.net/ftp/olsak/linal/linal2.pdf.
4. VITÁSEK, Emil. *Numerické metody*. Praha: SNTL, 1987.
5. FELDMAN, Paul. A practical scheme for non-interactive verifiable secret sharing. In: *28th Annual Symposium on Foundations of Computer Science* [online]. Washington, DC: IEEE, 1987, s. 427–438 [cit. 2019-03-20]. ISBN 0-8186-0807-2. Dostupné z DOI: 10.1109/SFCS.1987.4.
6. PEDERSEN, Torben Pryds. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: *Advances in Cryptology — CRYPTO '91* [online]. Berlín: Springer, 1992, s. 129–140 [cit. 2019-03-16]. ISBN 978-3-540-46766-3. Dostupné z DOI: 10.1007/3-540-46766-1_9.
7. QIONG, Li; ZHIFANG, Wang; XIAMU, Niu; SHENGHE, Sun. A Non-interactive Modular Verifiable Secret Sharing scheme. In: *Proceedings. 2005 International Conference on Communications, Circuits and Systems* [online]. Washington, DC: IEEE, 2005, sv. 1, s. 84–87 [cit. 2019-

03-20]. ISBN 0-7803-9015-6. Dostupné z DOI: 10.1109/ICCCAS.2005.1493367.

8. MCCURLEY, Kevin S. The Discrete Logarithm Problem. In: *Proceedings of Symposia in Applied Mathematics* [online]. Providence: AMS, 1990, sv. 42, s. 49–74 [cit. 2019-04-04]. ISBN 978-0-8218-0155-0. Dostupné z: http://www.mccurley.org/papers/dlog.pdf.

9. SHOR, Peter W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM J. Comput.* [online]. 1997, roč. 26, č. 5, s. 1484–1509 [cit. 2019-04-05]. ISSN 0097-5397. Dostupné z DOI: 10.1137/S0097539795293172.

10. BLAKLEY, G. R. Safeguarding cryptographic keys. In: *Proceedings of the AFIPS 1979 National Computer Conference* [online]. Montvale: AFIPS Press, 1979, sv. 48, s. 313–317 [cit. 2019-03-20]. Dostupné z DOI: 10.1109/AFIPS.1979.98.

11. HEI, Xiali; DU, Xiaojiang; SONG, Binheng. Two matrices for Blakley's secret sharing scheme. In: *2012 IEEE International Conference on Communications (ICC)* [online]. 2012, s. 810–814 [cit. 2019-04-15]. ISSN 1938-1883. Dostupné z DOI: 10.1109/ICC.2012.6364198.

12. YANG, X.; XIA, Z.; XIAO, M. Verifiable Secret Sharing and Distributed Key Generation Based on Hyperplane Geometry. In: *2015 2nd International Symposium on Dependable Computing and Internet of Things (DCIT)* [online]. 2015, s. 142–145 [cit. 2019-04-04]. Dostupné z DOI: 10.1109/DCIT.2015.10.

13. ASMUTH, Charles; BLOOM, John. A Modular Approach to Key Safeguarding. *IEEE Transactions on Information Theory* [online]. 1983, roč. 29, č. 2, s. 208–210 [cit. 2019-03-26]. ISSN 0018-9448. Dostupné z DOI: 10.1109/TIT.1983.1056651.

14. MIGNOTTE, Maurice. How to share a secret. In: *Proceedings of the Workshop on Cryptography* [online]. Berlín: Springer, 1983, s. 371–375 [cit. 2019-03-20]. ISBN 3-540-11993-0. Dostupné z DOI: 10.1007/3-540-39466-4_27.

15. GOLDREICH, Oded; RON, Dana; SUDAN, Madhu. Chinese remaindering with errors. *IEEE Transactions on Information Theory* [online]. 2000, roč. 46, č. 4, s. 1330–1338 [cit. 2019-03-20]. ISSN 0018-9448. Dostupné z DOI: 10.1109/18.850672.

16. IRELAND, Kenneth; ROSEN, Michael. *A Classical Introduction to Modern Number Theory* [online]. 2. vyd. New York: Springer, 1990 [cit. 2019-03-25]. ISBN 978-1-4757-2103-4. Dostupné z DOI: 10.1007/978-1-4757-2103-4.

17. QUISQUATER, Michaël; PRENEEL, Bart; VANDEWALLE, Joos. On the Security of the Threshold Scheme Based on the Chinese Remainder Theorem. In: *Public Key Cryptography* [online]. Berlín: Springer, 2002, s. 199–210 [cit. 2019-04-20]. ISBN 978-3-540-45664-3. Dostupné z DOI: 10.1007/3-540-45664-3_14.

18. IFTENE, Sorin. *General Secret Sharing Based on the Chinese Remainder Theorem.* [online]. 2006 [cit. 2019-03-30]. Dostupné z: https://eprint.iacr.org/2006/166.pdf.

19. HARN, Lein; MIAO, Fuyou. Weighted Secret Sharing Based on the Chinese Remainder Theorem. *I. J. Network Security*. 2014, roč. 16, č. 6, s. 420–425. ISSN 1816-3548.

20. CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. *Introduction to algorithms*. 3. vyd. Cambridge, Mass.: The MIT Press, 2009. ISBN 978-0-262-03384-8.

21. NIST. *Digital Signature Standard (FIPS 186-4)* [online]. Gaithersburg: NIST, 2013 [cit. 2019-04-20]. Dostupné z: https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.186-4.pdf.

22. Primes just less than a power of two, 201 to 300 bits. *The Prime Pages* [online] [cit. 2019-04-29]. Dostupné z: https://primes.utm.edu/lists/2small/200bit.html.

23. *Distribuovatelné součásti Visual C++ pro Visual Studio 2015* [online] [cit. 2019-04-30]. Dostupné z: https://www.microsoft.com/cs-CZ/download/details.aspx?id=48145.

24. *OpenSSL, Cryptography and SSL/TLS Toolkit* [online]. OpenSSL Software Foundation [cit. 2019-04-30]. Dostupné z: https://www.openssl.org.

Příloha A

Uživatelská příručka

A.1 Požadavky

Aplikace `KeyShare` je určena pro operační systém Windows 7 nebo vyšší. Pro spuštění aplikace je nutný balíček Microsoft Visual C++ 2015 Redistributable, návod na stažení a instalaci 32-bitové i 64-bitové verze se nachází na webových stránkách společnosti Microsoft [23]. Také je nutné mít knihovnu `libcrypto-1_1-x64.dll` pro 64-bitovou verzi aplikace, resp. `libcrypto-1_1.dll` pro 32-bitovou verzi aplikace. Tyto knihovny jsou součástí knihovny OpenSSL, použita byla knihovna OpenSSL verze 1.1.1b. Potřebné knihovny se nachází na přiloženém CD v adresáři `exe`, příp. je možné je získat instalací knihovny OpenSSL, návod je uveden na webových stránkách projektu OpenSSL [24].

A.2 Spuštění

Spustitelná aplikace `keyShare` se nachází na přiloženém CD v adresáři `exe`. Adresář obsahuje 32-bitovou verzi aplikace `keyShare.exe` a 64-bitovou verzi `keyShare_x64.exe`. Aplikace se spouští z příkazového řádku příkazem `keyShare.exe`, resp. `keyShare_x64.exe` doplněným o parametry potřebné k rozdělení nebo rekonstrukci klíče.

Pořadí a význam parametrů pro zašifrování souboru a rozdělení klíče:

-e šifrování souboru a rozdělení tajemství

-sh, -ab Shamirův, resp. Asmuthův–Bloomův algoritmus

k hodnota k v (k, n)-prahovém schématu

n hodnota n v (k, n)-prahovém schématu

file soubor k zašifrování

Jména souborů s díly klíče začínají řetězcem `SHshare`, resp. `ABshare`, soubor s veřejnými informacemi má název `SHpubFile`, resp. `ABpubFile`. Zašifrovaný soubor je uložen do souboru s názvem původního souboru doplněným o řetězec `_enc`.

Zašifrování souboru `soubor` a rozdělení klíče Shamirovým $(6, 12)$-schématem 64-bitovou verzí aplikace se tedy provede příkazem:

```
keyShare_x64.exe -e -sh 6 12 soubor
```

Pořadí a význam parametrů pro dešifrování souboru:

-d rekonstrukce tajemství a dešifrování souboru

pubFile soubor s veřejnými informacemi

share1 share2 ... soubory s díly klíče

file zašifrovaný soubor

Dešifrovaný soubor je uložen do souboru s názvem zašifrovaného souboru rozšířeným o řetězec `_dec`.

Dešifrování souboru `soubor_enc` z dílů `share_02`, `share_11` a `share_07` a veřejného souboru `pubFile` 32-bitovou verzí aplikace se provede následujícím příkazem:

```
keyShare.exe -d pubFile share_02 share_11 share_07 soubor_enc
```

Příloha B

Obsah přiloženého CD

- `BP-Svobodova-Hana-2019.pdf` ..... text práce ve formátu PDF
- `ctiMe.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou aplikací
- `src`
  - `impl` ..... zdrojové kódy aplikace
  - `openssl-1.1.1b` ..... zdrojové soubory knihovny OpenSSL
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `ZIP` ..... ZIP archívy přiložených souborů