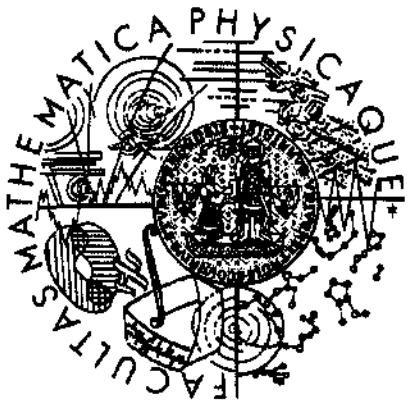
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Tomáš Kadlček

Kryptografická schémata používající diskrétní logaritmus

Katedra algebry
Vedoucí diplomové práce: Mgr. Štěpán Holub, Ph.D.

Studijní program: Matematika
Studijní obor: Matematické metody informační bezpečnosti

2008

Chtěl bych poděkovat panu Ryuichi Sakaiovi, který mně poskytnul kopii článku, jehož byl autorem. Tento článek je jinak na webu dostupný pouze přes členství v organizaci IEICE, což by pro mně byl problém.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním. V

Praze dne 07.08.2008 Tomáš Kadlček

Obsah

1 Úvod **6**

2 Problémy **8**

2.1 Výpočetní Diffie-Hellmanův problém 9

2.2 Rozhodovací Diffie-Hellmanův problém 9

2.3 Diffie-Hellmanův mezerový problém 10

2.4 Rozhodovací lineární Diffie-Hellmanův problém 10

2.5 ℓ-slabý Diffie-Hellmanův problém 10

2.6 ℓ-silný Diffie-Hellmanův problém 11

2.7 Problém koalice ℓ traitorů 11

2.8 Bilineární Diffie-Hellmanův problém 12

2.9 ℓ-bilineární problém D-H inverze 12

2.10 ℓ-bilineární problém D-H exponentu 12

2.11 Srovnání variant D-H problému 13

3 Eliptické křivky a párování **18**

3.1 Definice eliptických křivek 18

3.2 Výpočet Weilova párování 20

3.3 Parametry týkající se eliptických křivek 21

4 Schémata a jejich bezpečnost **23**

4.1 Úvodní bezpečnostní definice 23

4.1.1 Základní definice 23

4.1.2 Podpisová schémata a CMA 25

4.1.3 IBE schéma a IND-ID-CCA bezpečnost 26
4.1.4 Canetti, Halevi, Katz: konstrukce CPA2→CCA2 30
4.1.5 Schéma broadcastového šifrování 32
4.1.6 Model náhodných orákul - Bellare, Rogaway 34
4.2 Schémata 35
4.2.1 Šifrovací schéma ElGamal 35
4.2.2 Šifrovací schéma - Cramer, Shoup 36
4.2.3 Podpisové schéma - Boneh, Boyen 37
4.2.4 Podpisové schéma v.2 - Boneh, Boyen 38
4.2.5 Podpisové schéma - Waters 39
4.2.6 Schéma skupinového podpisu - Boneh, Boyen, Shacham . . . 40
4.2.7 Traitor tracing - Mitsunari, Sakai, Kasahara 42
4.2.8 Traitor tracing - Tô, Safavi-Naini, Zhang 43
4.2.9 IBE schéma - Boneh, Franklin 44
4.2.10 HIBE schéma - Gentry, Silverberg 45
4.2.11 IBE schéma - Boneh, Boyen 47
4.2.12 HIBE schéma - Boneh, Boyen 48
4.2.13 IBE schéma - Waters 49
4.2.14 HIBE schéma - Boneh, Boyen, Goh 51
4.2.15 Schéma broadcastového šifrování - Boneh, Gentry, Waters . 53
4.3 Třístranná Diffie-Hellmanova dohoda na klíči 54

5 Útoky **56**
5.1 Útoky proti tajemství α 56
5.1.1 Model generických grup 56
5.1.2 Generické útoky 57
5.1.3 J. H. Cheon - analýza k-SDH 60
5.2 Útoky proti konkrétním schématům 60
5.2.1 útok na MSK02 traitor tracing schéma 60
5.2.2 Slabiny ElGamal schématu 61

6 Shrnutí **63**

Literatura **65**

Název práce: Kryptografická schémata používající diskrétní logaritmus
Autor: Tomáš Kadlček
Katedra (ústav): Katedra algebry
Vedoucí bakalářské práce: Mgr. Štěpán Holub, Ph.D.
e-mail vedoucího: Stepan.Holub@mff.cuni.cz
Abstrakt: V práci se snažíme podat ucelený přehled o problematice diskrétního logaritmu, zejména nových variant vyskytujících se v literatuře od roku 2001, založených na práci s eliptickými křivkami a Weilovým nebo Tateovým párováním. Podáváme přehled těchto nových problémů včetně redukcí mezi nimi. Uvádíme také vybraná schémata založená na těchto problémech, která jsou něčím vyjímečná – ať už tím, že v nich byl daný problém představen, nebo tím, že mají velmi praktické parametry, nebo tím, že měli jako první formálně dokázanou bezpečnost. V práci také podáváme přesné definice týkajících se pojmů, které jsou v literatuře opomíjeny a počítá se s tím, že si čtenář hodně souvislostí domyslí sám.

Klíčová slova: Problém diskrétního logaritmu, IBE a HIBE schéma, bilineární grupy, CCA2 bezpečnost, párování na eliptických křivkách

Title: Kryptographics schemates using discrete logarithm
Author: Tomáš Kadlček
Department: Department of Algebra
Supervisor: Mgr. Štěpán Holub, Ph.D.
Supervisor's e-mail address: Stepan.Holub@mff.cuni.cz
Abstract: In the paper we try to give a view of the discrete logarithm problem, especially of related problems that appear in literature since 2001. These problems are based on a computation of Weil and Tate pairing on eliptic curves. We give a view of these problems including some reductions. We mention some chosen schemes based on these problems that are iteresting because of their practical parameters, primaci of security proofs or because these schemes introduced the new problem. We try to cover precisely the most important definitions in this sector of cryptography because these definition are omitted in the literature and it is often left up to reader to presume details by himself.

Keywords: Discrete logarithm problem, Identity-based encryption, bilinear groups, CCA2 security, paring on eliptic curves

Kapitola 1

Úvod

V práci se zabýváme schématy založenými na variantách Diffie-Hellmanova problému (pro zkrácení budeme občas psát D-H), jako jsou např. výpočetní D-H problém, bilineární D-H problém, ℓ-bilineární D-H problém inverze a další. Tyto nové problémy se objevují v literatuře od roku 2001, kdy Boneh a Franklin přišli s návrhem IBE schématu a o rok později Mitsunari, Sakai, Kasahara s návrhem traitor tracing schématu, oběma založenýma právě na některé z těchto nových variant Diffie-Hellmanova problému.

V kapitole 2 uvádíme formální definice všech dále rozebíraných variant Diffie-Hellmanova problému a mezi některými ukazujeme známé redukce. Odtud je patrné, že tyto zdánlivě odlišné problémy jsou mezi sebou provázány tak, že z řešení jednoho problému je možné dopočítat řešení druhého problému.

V kapitole 3 uvádíme základní definice týkající se eliptických křivek, Weilova a Tateova párování. Také rozebíráme možná rozšíření základního tělesa, do kterého se zobrazují hodnoty párování. Vliv rozšíření na rychlosti výpočtu a vůbec možnou existenci některých rozšíření pro supersingulární křivky.

V kapitole 4 uvádíme základní definice bezpečnosti pro různé typy schémat (podpis, IBE, HIBE, ...), praktické požadavky na tato schémata a tabulku s přehledem typů schémat a použitých problémů, na nichž jsou postaveny.

Dále uvádíme některá schémata založená na variantách Diffie-Hellmanova problému chronologicky tak, jak se objevila v literatuře. Neuvádíme všechna známá schémata, pouze výber těch, která jsou nějak vyjímečná. Ať už tím, že jako první měla důkaz bezpečnosti, nebo kvůli své efektivitě. Proto vždy zmiňujeme také parametry schémat – náročnost šifrovacích a dešifrovacích procedur, velikost soukromého klíče, veřejného klíče a šifrového textu. Zmiňujeme se o současných znalostech o jejich bezpečnosti, ať už v modelu náhodných orákul nebo ve standarním modelu.

Jde o podpisová schémata, schémata šifrování s veřejným klíčem, IBE a HIBE schémata, traitor tracing schémata, schéma skupinového podpisu a schéma broadcastového šifrování. Pro úplnost uvádíme také návrhy praktického využití těchto schémat, tak jak je jednotliví autoři zmiňují ve svých článcích.

V kapitole 5 uvádíme seznam známých dolních odhadů na složitost různých variant Diffie-Hellmanova problému v modelu generických grup (viz [32]). Dále uvádíme známé generické útoky na problém diskrétního logaritmu. Připojujeme známé útoky proti některým konkrétním schématům a bezpečnostní redukci aplikovatelnou na některé varianty Diffie-Hellmanova problému podle J. H. Cheona.

Kapitola 2

Problémy

V celém textu budeme označovat G_1, G_2, G_T abelovy grupy prvočíselného řádu p. Pokud je $G_1 = G_2$, budeme kvůli přehlednosti psát jenom G. Jak známo, všechny tyto grupy jsou automorfní grupě $(\mathbb{Z}_p, +)$. My však bumeme takto označovat různé grupy s různými representacemi. Dále budeme značit prvky $P \in G$, $P_1 \in G_1$. $P_2 \in G_2$ generátory příslušných grup.

Budeme pracovat s α v tělese $(\mathbb{Z}_p, +, \cdot)$, které bude vždy označovat neznámou hodnotu vyskytující se v definicích jednotlivých variant Diffie-Hellmanova problému. Automorfismus $G_2 \to G_1$ označíme ψ. Při výpočtech budeme požadovat, aby zobrazení ψ bylo efektivně vypočitatelné (jeho náročnost závisí na representacích grup G_1, G_2). O efektivnosti výpočtu ψ v opačném směru nepředpokládáme nic. Pod pojmem „efektivní vypočitatelnost“ máme přirozeně na mysli polynomiální výpočetní složitost vzhledem k délce vstupu.

V definicích problémů a schémat nebudeme explicitně psát, že vylučujeme krajní případy, např. když je možné nevhodnou volbou hodnot proměnných vytvořit zlomek se jmenovatelem rovným nule. Samozřejmě však takové případy neuvažujeme.

Dále budeme značit $e : G_1 \times G_2 \to G_T$ (nedegenerované) bilineární zobrazení, tedy takové, že pro všechna $Q_1 \in G_1$, $Q_2 \in G_2$, $a, b \in (\mathbb{Z}_p, +, \cdot)$: $e(aQ_1, bQ_2) = e(Q_1, Q_2)^{ab}$ a navíc $e(P_1, P_2) \neq 1$. Všimněme si na tomto místě, že prvky tělesa $(\mathbb{Z}_p, +, \cdot)$ používáme pro zkrácený zápis iterování aditivní operace v grupách G, G_1, G_2, zatímco v grupě G_T takto zkracujeme zápis iterování multiplikativní operace. Pro $Q \in G$ značíme $aQ = \underbrace{Q + \cdots + Q}_{a}$, pro $Q \in G_T$ značíme $Q^a = \underbrace{Q \cdots Q}_{a}$.

Kryptograficky použitelná bilineární zobrazení jsou známa jenom dvě, Weilovo a Tateovo párování na eliptických křivkách. Ostatní bilineární zobrazení mají pro kryptografické účely nevhodné vlastnosti. Pro Weilovo a Tateovo párování např.

není známo, jak je efektivně invertovat, máme-li pevně dánu jednu ze složek zobrazení. Proto budeme v textu běžně používat aditivní i multiplikativní značení v závislosti na předpokladu, že G, G_1, G_2 jsou podgrupy bodů na eliptické křivce a G_T je multiplikativní grupa. Pro prvky grup G, G_1, G_2 budeme používat velká písmena, jak je tomu zvykem pro body na eliptických křivkách, prvky G_T budeme značit malými písmeny.

Definice: Dvojici (G_1, G_2) nazýváme *bilineární grupy*, pokud obě jsou abelovy, prvočíselného řádu p, a existují efektivně vyčíslitelná zobrazení ψ a e taková, že ψ je automofismum $G_2 \to G_1$, e je (nedegenerované) bilineární zobrazení $G_1 \times G_2 \to G_T$, kde G_T je nějaká abelova grupa prvočíselného řádu p, a navíc grupové operace v G_1, G_2 a G_T jsou efektivně vyčíslitelné.

Problém diskrétního logaritmu (DLP) spočívá v úloze nalézt α, známe-li pouze prvky $P, P^\alpha \in G$. V některých grupách, např. v $(\mathbb{Z}_p, +)$, je řešení DLP jednoduché, pomocí euklidova algoritmu. V jiných grupách, např. v podgrupách $(\mathbb{Z}_p^*, \cdot)$ prvočíselné velikosti, je DLP těžký problém. Kryptografická schémata jsou často založena na některé z variant tohoto problému. Všechny tyto varianty jsou však z bezpečnostního hlediska slabší než původní DLP.

Pro kryptografické účely se také využívají zobecněné verze uváděných problémů, kdy zobecnění tkví v řádu použité grupy. Můžeme totiž pracovat i v grupách s neprvočíselným řádem, např. v grupě $(\mathbb{Z}_p^*, \cdot)$ řádu $p-1$. Pak se mohou objevit problémy pocházející z vlastnosti rozkladu čísla $p-1$ na součin prvočísel. Je intuitivně zřejmé, že čím menší prvočísla se v rozkladu vyskytují, tím snazší bude pro útočníka vyřešit danou instanci problému, na níž je založeno konkrétní kryptografické schéma.

2.1 Výpočetní Diffie-Hellmanův problém

„Computation Diffie-Hellman Problem“ (CDH)

Definice: Nechť $a, b \in \mathbb{Z}_p$ jsou neznámé hodnoty a $P \in G$. Ze zadaných hodnot (P, aP, bP) máme vypočítat $abP \in G$.

2.2 Rozhodovací Diffie-Hellmanův problém

„Decision Diffie-Hellman Problem“ (DDH)

Definice: Nechť $a, b, c \in \mathbb{Z}_p$ jsou neznámé hodnoty a $P \in G$. Pro zadané hodnoty (P, aP, bP, cP) máme rozhodnout, zda $c = ab$ v $\mathbb{Z}_p$.

V bilineárních grupách lze snadno vyřešit DDH problém. Pro vstupní hodnoty (P, aP, bP, cP) stačí testovat, zda $e(aP, bP) = e(P, cP)$, neboť rovnost nastává, právě když $ab = c$.

2.3 Diffie-Hellmanův mezerový problém

„Gap Diffie-Hellman problem" (GDH)

Tento problém byl představen v článku [29] a dále rozebrán v článku [24].

Problém je založen na existenci tzv. „Gap Diffie-Hellman groups", což jsou bilineární grupy, ve kterých je snadné řešit DDH problém, ale CDH problém je v nich stále těžký. Název problému nepřekládáme do češtiny, protože označení Gap D-H groups už je zažité. Jediné takové známé grupy jsou bilineární grupy. Párování slouží jako orákulum pro řešení DDH problému.

Definice: Nechť $a, b \in \mathbb{Z}_p$ jsou neznámé hodnoty a $P \in G$. Pro zadané hodnoty (P, aP, bP) máme vypočítat $abP \in G$ (CDH problém) s využitím orákula (pro DDH problém), které rozhoduje, zda pro libovolnou čtveřici $(P, a'P, b'P, c'P)$ platí, že $c' = a'b'$ v $\mathbb{Z}_p$.

2.4 Rozhodovací lineární Diffie-Hellmanův problém

„Decision Linear Diffie-Hellman Problem" (DLDH)

Tento problém byl představen v článku [8] o schématu skupinového podpisu.

Definice: Nechť $a, b, c \in \mathbb{Z}_p$ jsou neznámé hodnoty a $Q, R, S \in G$ jsou libovolné prvky G. Pro zadané hodnoty (Q, R, S, aQ, bR, cS) máme rozhodnout, zda $c = a+b$.

2.5 ℓ-slabý Diffie-Hellmanův problém

„ℓ-Weak Diffie-Hellman Problem" (ℓ-wDH)

Tento problém byl představen v článku o traitor tracing schématu v [26].

Definice: Ze zadaných hodnot $(P_1, \alpha^i P_2, 0 \leq i \leq \ell, \mid P_1 = \psi(P_2))$, máme spočítat $\frac{1}{\alpha} P_1 \in G_1$.

Tento ℓ-wDH problém je ekvivalentní následujícímu problému: Ze zadaných hodnot $(P_1, \alpha^i P_2, 0 \leq i \leq \ell, \mid P_1 = \psi(P_2))$ spočítat $\alpha^{\ell+1} P_2 \in G$.

Polynomiální redukce funguje takto:

Na vstupu máme $(P_1, P_2, \alpha P_2 \ldots, \alpha^\ell P_2)$, tž. $P_1 = \psi(P_2)$. Označme si $Q = \alpha^\ell P_2$ a $y = \alpha^{-1} \in \mathbb{Z}_p$. Pak lze vstup přepsat jako $(P_1, Q = \alpha^\ell P_2, yQ = \alpha^{\ell-1} P_2 \ldots, y^\ell Q = P_2)$. Naším úkolem je spočítat $y^{\ell+1} Q = \frac{1}{\alpha} P_2$.

Tedy umíme-li řešit jeden z problémů (označme takový algoritmus $\mathcal{A}$), pak dokážeme snadno vyřešit i druhý problém, a to jednoduše tak, že pozměníme pořadí vstupních hodnot a pustíme znovu $\mathcal{A}$.

2.6 ℓ-silný Diffie-Hellmanův problém

„ℓ-Strong Diffie-Hellman Problem“ (ℓ-SDH)

Tento problém byl představen v článku o podpisovém schématu v [4].

Definice: Pro vstupní hodnoty $(P_1, \alpha^i P_2,\ 0 \leq i \leq \ell, \mid P_1 = \psi(P_2))$ máme najít libovolnou dvojici $\left(\frac{1}{\alpha+c} P_1, c\right) \in G_1 \times \mathbb{Z}_p^*$.

Tento problém je podobný předchozímu, ovšem je třeba si všimnout, že existuje velké množství řešení ℓ-SDH, zatímco ℓ-wDH má pouze jedno řešení. Pro fixní c jsou si oba problémy ekvivalentní. Pokud je ale c volitelné, má útočník více možností jak vyřešit problém ℓ-SDH než má u problému ℓ-wDH.

Na tomto místě se hodí poznamenat, že ℓ-silný Diffie-Hellmanův problém je vlastně „slabší“ než ℓ-slabý Diffie-Hellmanův problém. Přívlastek „silný“ pochází ze síly našeho předpokladu o tomto problému. Pokud předpokládáme, že ℓ-SDH problém je těžké vyřešit, tak ve skutečnosti předpokládáme víc, než kdybychom předpokládali, že ℓ-wDH problém je těžké vyřešit.

2.7 Problém koalice ℓ traitorů

„Collusion attack with ℓ traitors“ (ℓ-CAA)

Tento problém byl (stejně jako ℓ-wDH problém) představen v článku [26] o traitor tracing schématu.

Definice: Nechť jsou dána po dvou různá $u_0, u_1, \ldots, u_\ell \in \mathbb{Z}_p$. Pro zadané hodnoty $\left(\frac{1}{\alpha+u_i} P,\ 1 \leq i \leq \ell\right)$ máme spočítat $\frac{1}{\alpha+u_0} P$.

Tento problém byl představen v článku o traitor tracing schématu v [26]. Zde u_i představují jednotlivé uživatele a $\frac{1}{\alpha+u_i} P$ jsou jejich tajemství. Takže vyřešit ℓ-CAA odpovídá útoku ℓ spolupracujících traitorů na tajemství uživatele u_0. Ale v traitor tracing schématech požadujeme, aby bylo obtížné vytvořit pirátský dekoder

nedopátratelný k někomu z traitorů, a to odpovídá vyřešení ℓ-CAA problému pro libovolné, předem fixně nestanovené, u. To je zřejmě jednodušší problém.

2.8 Bilineární Diffie-Hellmanův problém

„Bilinear Diffie-Hellman Problem" (BDH)

Tento problém byl představen v článcích [10] o IBE schématu a v původní verzi článku [25] o jednokolové D-H výměně klíče mezi třemi stranami.

Definice: Nechť $a, b, c \in \mathbb{Z}_p$ jsou neznámé hodnoty a $P \in G$. Ze zadaných hodnot (P, aP, bP, cP) máme vypočítat $e(P, P)^{abc} \in G_T$.

Boneh a Franklin v [10] zobecnili BDH problém, a toto zobecnění navrhli jako úpravu jejich IBE schématu. Definovali co-BDH problém následovně:
Definice: Nechť $a, b, c \in \mathbb{Z}_p$ jsou neznámé hodnoty, $P_1 \in G_1$, $P_2 \in G_2$ jsou dány. Na vstupu $(P_1, aP_1, bP_1, P_2, aP_2, cP_2)$ máme vypočítat $e(P_1, P_2)^{abc} \in G_T$.

BDH je přirozeným zobecněním CDH pro bilineární grupy. Stejně jako DDH se proto definuje i rozhodovací bilineární Diffie-Hellmanův problém (DBDH):
Definice: Nechť $a, b, c, d \in \mathbb{Z}_p$ jsou neznámé hodnoty a $P \in G$. Pro zadané hodnoty (P, aP, bP, cP) a $e(P, P)^d$ máme rozhodnout, zda $abc = d$.

2.9 ℓ-bilineární problém D-H inverze

„ℓ-Bilinear Diffie-Hellman Inversion Problem" (ℓ-BDHI)

Definice: Pro zadané hodnoty $(P, \alpha^i P,\ 1 \leq i \leq \ell)$, máme spočítat $e(P, P)^{1/\alpha} \in G_T$.

V literatuře se definuje i rozhodovací varianta ℓ-BDHI problému. Označme ji ℓ-DBDHI.

Definice: Nechť $t \in \mathbb{Z}_p$ je neznámé. Pro zadané hodnoty $(P, e(P, P)^t, \alpha^i P,\ 1 \leq i \leq \ell)$, máme rozhodnout, zda $t = \frac{1}{\alpha} \in \mathbb{Z}_p$.

2.10 ℓ-bilineární problém D-H exponentu

„ℓ-Bilinear Diffie-Hellman Exponent Problem" (ℓ-BDHE)

Tento problém byl představen v článku [7] o HIBE schématu. Použit je také pro konstrukci schématu broadcastového šifrování, viz strana 53.

Definice: Pro vstupní hodnoty $(P_1, \alpha^i P_2, 0 \leq i \leq 2\ell, i \neq \ell. \mid P_1 = \psi(P_2))$ máme spočítat $e(P_1, P_2)^{\alpha^\ell} \in G_T$.

2.11 Srovnání variant D-H problému

Pro dva problémy A a B budeme značit A $\Rightarrow$ B, pokud („slabší") problém B lze vyřešit v polynomiálním čase s použitím polynomiálního počtu dotazů na orákulum řešící („silnější") problém A.

DL $\Rightarrow$ CDH $\Rightarrow$ DDH:

Triviální. Pomocí algoritmu na řešení DL najdeme z aP, bP exponenty a, b, vypočteme součin ab a nakonec i abP. Umíme-li řešit výpočetní variantu problému, umíme zřejmě rozhodovat i odpovídající rozhodovací problém.

Implikace v opačném směru nejsou dosud známy.

1-BDHI $\Leftrightarrow$ BDH:

„$\Rightarrow$" :

Podle předpokladu máme algoritmus $\mathcal{A}$ takový, že $\mathcal{A}(P, \alpha P) = e(P, P)^{1/\alpha}$. Navíc známe $P, aP, bP, cP \in G$ a chceme vypočítat $e(P, P)^{abc}$ pro nějaké neznámé hodnoty $a, b, c \in \mathbb{Z}_p$.
Označme si $Q = aP + bP + cP$.
Potom máme

$$\mathcal{A}(Q, P) = \mathcal{A}(Q, \frac{1}{a+b+c}Q) = e(Q, Q)^{a+b+c} = e(P, P)^{(a+b+c)^3} = T_1.$$

Podobně pokračujeme, abychom dostali:

$$\begin{aligned} T_2 &= e(P, P)^{a^3}, \quad T_3 = e(P, P)^{b^3}, \quad T_4 = e(P, P)^{c^3}, \quad T_5 = e(P, P)^{(a+b)^3}, \\ T_6 &= e(P, P)^{(a+c)^3}, \quad T_7 = e(P, P)^{(b+c)^3}. \end{aligned}$$

Nakonec vypočteme $e(P, P)^{abc} = \left(\frac{T_1 \cdot T_2 \cdot T_3 \cdot T_4}{T_5 \cdot T_6 \cdot T_7}\right)^{\frac{1}{6}}$.

„$\Leftarrow$" :

Předpokládejme, že máme algoritmus $\mathcal{A}$, který pro libovolné $a, b, c \in \mathbb{Z}_p$ vypočte $\mathcal{A}(P, aP, bP, cP) = e(g, g)^{abc}$. Navíc známe $P, xP \in G$ a chceme vypočítat $e(P, P)^{1/x}$ pro neznámé $x \in \mathbb{Z}_p$.

Nakonec spočteme:

$$\mathcal{A}(xP, P, P, P) = \mathcal{A}(xP, \frac{1}{x}xP, \frac{1}{x}xP, \frac{1}{x}xP) = e(xP, xP)^{\frac{1}{x}\cdot\frac{1}{x}\cdot\frac{1}{x}} = e(P, P)^{\frac{1}{x}}.$$

Jak je to s ℓ-BDHI $\Rightarrow$ BDH se neví. ℓ-BDHI, pro $\ell \geq 2$, je jednodušší problém než 1-BDHI, a proto i kdybychom měli pro ℓ-BDHI algoritmus, tak tento nám nemusí vůbec pomoct při řešení BDH problému.

1-wDH $\Leftrightarrow$ CDH:

„$\Rightarrow$“ :

Podle předpokladu máme algoritmus $\mathcal{A}$, který pro libovolné $x \in \mathbb{Z}_p$ dává $\mathcal{A}(P, xP) = \frac{1}{x}P$. Navíc známe $P, aP, bP \in G$ a chceme vypočítat abP pro neznámé $a, b \in \mathbb{Z}_p$.

Vypočteme postupně:
$\mathcal{A}(aP, P) = \mathcal{A}(aP, \frac{1}{a}aP) = \left(\frac{1}{a}\right)^{-1}(aP) = a^2P$,
$\mathcal{A}(aP, P) = \mathcal{A}(aP, \frac{1}{a}aP) = \left(\frac{1}{a}\right)^{-1}(aP) = a^2P$,
$\mathcal{A}(bP, P) = b^2P$,
$\mathcal{A}((a+b)P, P) = (a+b)^2P$.
Nyní už snadno dopočítáme $\frac{1}{2}\left((a+b)^2 - a^2 - b^2\right)P = abP$.

„$\Leftarrow$“ :

Podle předpokladu máme algoritmus $\mathcal{A}$ takový, že pro libovolné $a, b \in \mathbb{Z}_p$ $\mathcal{A}(P, aP, bP) = abP$. Navíc známe $P, xP \in G$ a chceme vypočítat $\frac{1}{x}P$ pro neznámé $x \in \mathbb{Z}_p$.

Pustíme algoritmus na vstupu (xP, P, P):

$$\mathcal{A}(xP, P, P) = \mathcal{A}(xP, \frac{1}{x}xP, \frac{1}{x}xP) = \frac{1}{x^2}xP = \frac{1}{x}P.$$

$(\ell - 1)$-wDH $\Leftrightarrow$ ℓ-CAA:

„$\Rightarrow$“ :

Podle předpokladu máme algoritmus $\mathcal{A}$ takový, že $\mathcal{A}(P, xP, \ldots, x^{\ell-1}P) = \frac{1}{x}P$. Navíc známe po dvou různá $u_0, \ldots, u_\ell \in \mathbb{Z}_p$ a $\frac{1}{x+u_1}P, \ldots, \frac{1}{x+u_\ell}P \in G$. Chceme vypočítat $\frac{1}{x+u_0}P$ pro neznámé $x \in \mathbb{Z}_p$.

Označme $y = x + u_0$ a $Q = \frac{1}{(x+u_1)\ldots(x+u_\ell)}P$. Pořád však neznáme x ani y.

Pro $j = 0, \ldots, \ell - 1$ máme $y^j Q = \frac{(x+u_0)^j}{(x+u_1)\ldots(x+u_\ell)} P$. Protože je stupeň čitatele pro všechna uvažovaná j menší než stupeň jmenovatele, můžeme $y^j Q$ rozvinout v součet parciálních zlomků:

$$y^j Q = \sum_{i=1}^{\ell} \frac{c_{ij}}{x + u_i} P,$$

kde $c_{ij} \in \mathbb{Z}_p$ lze spočítat z $u_1, \ldots, u_\ell$.

Tedy umíme se dopočítat ke $Q, yQ, \ldots, y^{\ell-1}Q$ a následně pomocí algoritmu $\mathcal{A}$ spočteme i $\frac{1}{y}Q$. Opět rozvineme v součet parciálních zlomků:

$$\frac{1}{y} Q = \frac{1}{(y + u_0)(y + u_1)\ldots(x + u_\ell)} P = \sum_{i=0}^{\ell} \frac{c_i'}{x + u_i} P,$$

pro $c_i' \in \mathbb{Z}_p$.

Jelikož je $c_0' \neq 0$, můžeme dokončit výpočet $\frac{1}{x+u_0} P = {c_0'}^{-1} \left(\frac{-1}{x} P + \sum_{i=1}^{\ell} \frac{c_i'}{x+u_i} P \right)$.

„$\Leftarrow$“ :

Podle předpokladu algoritmus $\mathcal{A}$ pro po dvou různá $u_0, u_1, \ldots, u_\ell \in \mathbb{Z}_p$ vypočte $\mathcal{A}\left(u_0, \ldots, u_\ell, \frac{1}{x+u_1} P, \ldots, \frac{1}{x+u_\ell} P\right) = \frac{1}{x+u_0} P$. Navíc známe hodnoty $P, xP, \ldots, x^{\ell-1}P$ a chceme vypočítat $\frac{1}{x} P$ pro neznámé $x \in \mathbb{Z}_p$.

Zvolme si po dvou různá $u_0, u_1, \ldots, u_\ell \in \mathbb{Z}_p$. Označme $y = x - u_0$ a $Q = (y + u_1)\ldots(y + u_\ell)P$. Stále však neznáme x ani y.

Nyní uvažme zlomek tvaru $\frac{1}{y+u_i} Q = (y + u_1)\ldots\underbrace{(y + u_i)}_{\text{vynechat}}\ldots(y + u_\ell)P$. Po rozepsání získáme polynom stupně $\ell - 1$ jako koeficient u bodu P. Vzhledem k tomu, že známe $P, xP, \ldots, x^{\ell-1}P$, jsme schopni spočítat $\frac{1}{y+u_i} Q$ pro všechna $i = 1, \ldots, \ell$.

Pustíme algoritmus $\mathcal{A}$ a získáme $\frac{1}{y+u_0} Q$. Dál můžeme počítat:

$$\frac{1}{y + u_0} Q = \frac{1}{x} Q = \frac{1}{x}(x - u_0 + u_1)\ldots(x - u_0 + u_\ell)P = \sum_{i=-1}^{\ell-1} d_i x^i P.$$

Protože u_i jsou po dvou různé, je $d_{-1} \neq 0$ a tedy můžeme dokončit výpočet následovně:

$$\frac{1}{x} P = (d_{-1})^{-1} \left(\frac{1}{y + u_0} Q - \sum_{i=0}^{\ell-1} d_i x^i P \right).$$

DDH v grupě $G_T \Rightarrow$ DBDH:

Máme $\mathcal{A}$, který na vstupu (P, xP, yP, zP) rozhodne, zda $xy = z$, pro libovolné $x, y, z \in \mathbb{Z}_p$. Chceme s jeho pomocí pro $\left(P, aP, bP, cP, e(P,P)^d\right)$ rozhodnout, jestli $abc = d$. Samozřejmě neznáme hodnoty $a, b, c, d \in \mathbb{Z}_p$.

Stačí se zeptat $\mathcal{A}$, zda $\left(e(P,P), e(aP,bP), e(cP,P), e(P,P)^d\right)$ je DDH instance v G_T. $\mathcal{A}$ odpoví „ANO“ tehdy a jen tehdy, když $abc = d$.

Implikace v opačném směru není dosud známa.

CDH v grupě G (případně v G_T) $\Rightarrow$ BDH:

Máme $\mathcal{A}$, který na vstupu (P, xP, yP) vypočte xyP, pro libovolné $x, y \in \mathbb{Z}_p$. Chceme s jeho pomocí pro (P, aP, bP, cP) vypočítat $e(P,P)^{abc}$. Samozřejmě neznáme hodnoty $a, b, c \in \mathbb{Z}_p$.

Postupně vypočteme $\mathcal{A}(P, aP, bP) = abP$, $\mathcal{A}(P, abP, cP) = abcP$ a vydáme hodnotu $e(P, abcP) = e(P,P)^{abc}$.

Obdobně pro CDH v G_T:

Označme si $X = e(P,P)$. Spočteme:

$$\mathcal{A}\left(e(P,P), e(P,aP), e(P,bP)\right) = \mathcal{A}\left(X, X^a, X^b\right) = X^{ab} = e(P,abP)$$

a vydáme:

$$\mathcal{A}\left(e(P,P), e(P,abP), e(P,cP)\right) = e(P,P)^{abc}.$$

Implikace v opačném směru není dosud známa (v G ani v G_T).

ℓ-wDH $\Leftrightarrow$ ℓ-SDH s předem zadanou konstantou c:

V článku [4] definují Boneh a Boyen ℓ-SDH s konstantou $c \in \mathbb{Z}_p^*$ (viz definice na straně 11), která není pevně předem dána. Zmiňují se (bez důkazu), že pokud je c fixně určeno, pak ℓ-SDH je ekvivalentní ℓ-wDH problému, kde je (de fakto) $c = 0$.

Podobně Cheon v [22] píše, že pomocí algoritu pro ℓ-wDH dokážeme vyřešit problém ℓ-SDH. Tedy pokud uníme na daném vstupu pro ℓ-wDH problém vypočítat $g_1^{1/\alpha}$, stačí nám to k výpočtu $g_1^{1/(\alpha+c)}$ na tomtéž vstupu, kde $c \in \mathbb{Z}_p^*$ je pevně dáno dopředu.

Diskuse:
$\mathbb{Z}_p^*$ je obvykle značení pro multiplikativní grupu, tedy $\alpha + c$ nedává smysl. Autoři se nepřesně vyjádřili při popisu problému, což může zmást čtenáře. Navíc neuvedli redukci, pouze se zmínili, že je triviální.

Uveďme zde – pro pořádek – tuto redukci:
(bereme c jako prvek tělesa $(\mathbb{Z}_p, +, \cdot)$, protože s konstantou c potřebujeme sčítat i dělit v tomto tělese)

„$\Rightarrow$“ :

Podle předpokladu máme algoritmus $\mathcal{A}$, který na vstupu $(P_1, P_2, \alpha P_2, \ldots, \alpha^\ell P_2)$ vypočte hodnotu $\frac{1}{\alpha} P_1$. Chceme s jeho pomocí na tomtéž vstupu, pro předem zadané $c \in \mathbb{Z}_p \setminus \{0\}$, vypočítat $\frac{1}{\alpha+c} P_1$.

Pro $1 \leq i \leq \ell$ rozepíšeme $(\alpha + c)^i P_2$ do binomického rozvoje:

$$(\alpha + c)^i P_2 = \alpha^i P_2 + \cdots + \binom{i}{j} \alpha^{i-j} c^j P_2 + \cdots + c^i P_2.$$

Jednotlivé sčítance v rozvoji jsem schopni spočítat, protože všechna potřebná α^i známe. Postupně takto získáme nový vstup $(P_1, P_2, (\alpha + c)P_2, \ldots, (\alpha + c)^\ell P_2)$ pro algoritmus $\mathcal{A}$, který vrátí hledanou hodnotu $\frac{1}{\alpha+c} P_1$.

„$\Leftarrow$“ :

V opačném směru budeme s algoritmem $\mathcal{A}$ řešícím ℓ-SDH problém postupovat podobně. Přepočítáme vstup na $(P_1, P_2, (\alpha - c)P_2, \ldots, (\alpha - c)^{ell} P_2)$ a spustíme na tomto vstupu $\mathcal{A}$. ten vydá požadovanou hodnotu:

$$\frac{1}{(\alpha - c) + c} P_1 = \frac{1}{\alpha} P_1.$$

Kapitola 3

Eliptické křivky a párování

V této kapitole připomeneme potřebný základ pro práci s eliptickými křivkami, definice párování a důležité parametry křivek pro použití v kryptografii.

3.1 Definice eliptických křivek

Pracujeme s křivkou $E/\mathbb{F}_p$. O je bod v nekonečnu. Předem uvedeme odkaz na knihu [33], ve které jsou podrobně popsány všechny potřebné základy k eliptickým křivkám a algebraické geometrii. Uváděné definice slouží jako přehled potřebných pojmů a zavedení zanačení. Přesné definování pojmů týkajících se eliptických křivek by zabralo více prostoru a navíc to ani není smyslem této práce. Nám stačí pouze dobrat se k definici Weilova a Tateova párování, které pak budeme hojně používat při výpočtech různých procedur v uvedených schématech. Jak už bylo zmíněno dříve, tyto dvě párování jsou jediná známá bilineární zobrazení, která mají vhodné vlastnosti pro použití v kryptografii.

Definice: *Funkční těleso* $K(E)$ eliptické křivky E je množina tříd reálných racionálních funkcí ve dvou proměnných x, y, počítáno modulo rovnice křivky E : $y^2 = x^3 + ax + b$.

Definice: *Divisor* D je prvek volné grupy generované body na křivce E, tj. lze jej zapsat jako formální (konečnou) sumu $D = \sum_i a_i(P_i)$, kde P_i jsou body E, (P_i) divisor odpovídající bodu P_i, a_i jsou celá čísla. *Stupeň divisoru* D definujeme jako $\deg(D) = \sum_i a_i$.

Definice: Pro funkci $f \in K(E)$ definujeme divisor (stupně 0) jako $\mathrm{div}(f) = \sum_i a_i(P_i)$, kde P_i jsou nuly nebo póly funkce f, a_i jsou odpovídající násobnosti nul a pólů. Nuly v definici $\mathrm{div}(f)$ přičítáme zatímco póly odečítáme. Jakýkoliv divizor $D = \mathrm{div}(f)$ nazýváme *hlavní divisor*. Grupu divisorů nulového stupně můžeme

faktorizovat podle podgrupy hlavních divisorů, čímž získáme důležitou *třídovou grupu divisorů*, ve které rozdíl dvou divisorů ze stejné třídy dává hlavní divisor.

Př. Ať $f : ax+by+c = 0$ je přímka procházející křivkou E v bodech $P_1 \neq \pm P_2$. Pak f protíná E ještě ve třetím bodě $P_3 \neq O$. Funkce f má tedy tři nuly P_1, P_2, P_3, všechny stupně 1, a jeden pól stupně 3 v nekonečnu O. $\mathrm{div}(f) = (P_1)+(P_2)+(P_3)-3(O)$.

Test, zda divisor $D = \sum_i a_i(P_i)$ stupně nula je hlavní divisor lze provést tak, že spočteme $\sum_i a_i(P_i)$ pro všechny body P_i, s výjimkou bodu O v nekonečnu. Výsledek tohoto výpočtu je bod O tehdy a jen tehdy, pokud je D hlavní divisor.

Definice: Pro $D = \sum_i a_i(P_i)$ definujeme $\mathrm{supp}(D) = \{P_i | a_i \neq 0\}$ („Nosná množina divisoru D“).

Definice: Je-li $P \notin \mathrm{supp}(\mathrm{div}(f))$, pak $f(P)$ vypočteme tak, že souřadnice x, y bodu P dosadíme do racionální funkce representující f. Už můžeme také definovat funkční hodnotu f pro divisor $D = \sum_i a_i(P_i)$. Jsou-li $\mathrm{supp}(\mathrm{div}(f))$ a $\mathrm{supp}(D)$ disjunktní, pak definujeme $f(D) = \prod_i f(P_i)^{a_i}$.

Nyní už můžeme definovat Weilovo párování pro dva body $P, Q \in E$.

Definice: *Weilovo párování* $e_n(\cdot,\cdot)$ je bilineární zobrazení z torzní grupy $E[n] = \{P \in E | nP = O\}$ do multiplikativní grupy G_n n-tých odmocnin z jednotky v nějakém rozšíření $\mathbb{F}_{p^k}$. Pro dva body $P, Q \in E[n]$ najdeme funkce f_P, f_Q, tž. $\mathrm{div}(f_P) = n(P) - n(O)$, $\mathrm{div}(f_Q) = n(Q) - n(O)$, a divisory D_P, D_Q, tž. D_P je ze stejné třídy divisorů jako $(P) - (O)$ a obdobně D_Q je ze stejné třídy divisorů jako $(Q) - (O)$. Weilovo párování definujeme takto:

$$e_n(P, Q) = \frac{f_P(D_Q)}{f_Q(D_P)}.$$

Menezes, Okamoto, Vanstone navrhli v práci [27] redukci pro řešení diskrétního logaritmu v nějakém malém rozšíření konečného tělesa namísto v grupě bodů na eliptické křivce. Redukce funguje následovně:
Pro X lineárně nezávislé na P, Q převedeme problém nalézt α z rovnice $Q = \alpha P$ na eliptické křivce na problém nalézt α z rovnice $e_n(Q, X) = e_n(P, X)^\alpha$ v tělese $\mathbb{F}_{p^k}$.

Definice: *Tateovo párování* $t_n(\cdot,\cdot)$ je bilineární zobrazení pracující s tzv. „n-fold“ divisory, (D je „n-fold“ divisor, pokud nD je hlavní divisor). Pro takové divisory D_1, D_2 definované nad nějakým rozšířením $\mathbb{F}_{p^k}$, které obsahuje n-té odmocniny z jednotky, najdeme funkci f_{D_1}, tž. $\mathrm{div}(f_{D_1}) = nD_1$. Tateovo párování definujeme takto:

$$t_n(D_1, D_2) = f_{D_1}(D_2)^{(p^k-1)/n}.$$

Tateovo párování je na první pohled složitější než Weilovo párování, protože pracuje s divisory namísto s body křivky. K výpočtu je však potřeba spočítat hodnotu jen jedné funkce na eliptické křivce, zatímto Weilovo párování potřebuje vypočítat dvě takové funkce.

3.2 Výpočet Weilova párování

Máme dány dva body $P, Q \in E[n]$ n-torzní podgrupy bodů na křivce E. Předpokládejme, že $P \neq Q$. Chceme spočítat párování $e_n(P, Q) \in G_T$. Vezněme dva náhodné doby $R_1, R_2 \in E[n]$ a uvažme divisory $\mathcal{A}_P = (P + R_1) - (R_1)$ a $\mathcal{A}_Q = (QP + R_2) - (R_2)$. Tyto divisory jsou ve stejných třídách jako divisory $(P) - (O)$, resp. $(Q) - (O)$. Weilovo párování vypočteme pomocí vzorce:

$$e_n(P, Q) = \frac{f_P(\mathcal{A}_Q)}{f_Q(\mathcal{A}_P)} = \frac{f_P(Q + R_2) f_Q(R_1)}{f_P(R_2) f_Q(P + R_1)}.$$

Pokud by nastal málo pravděpodobný případ, že výraz není dobře definovaný, stačí zvolit jiné divisory R_1, R_2.

Teď potřebujeme jen vypočítat f_P, f_Q pro zadané divisory. Definujme pro kladné číslo b divisor:

$$\mathcal{A}_b = b(P + R_1) - b(R_1) - (bP) + (O).$$

Tento je jistě hlavní divisor a proto existuje funkce f_b taková, že $(f_b) = \mathcal{A}_b$. Je patrné, že $(f_P) = (f_n)$, proto $f_P(\mathcal{A}_Q) = f_n(\mathcal{A}_Q)$. Obdobně pro f_Q.

Teď už stačí jen vypočítat $f_n(\mathcal{A}_Q)$.

Ať $a_1 x + b_1 y + c_1 = 0$ je přímka vedoucí body bP, cP, případně pro $b = c$ to bude tečna ke křivce E. Definujme funkci $g_1(x, y) = a_1 x + b_1 y + c_1$.

Dále ať $x + c_2 = 0$ je přímka procházející bodem $(b + c)P$. Definujme funkci $g_2(x, y) = x + c_2$.

Divisory těchto dvou funkcí jsou:

$$\begin{aligned} (g_1) &= (bP) + (cP) + (-(b + c)P) - 3(O), \\ (g_2) &= ((b + c)P) + (-(b + c)P) - 2(O). \end{aligned}$$

Máme tedy:

$$\begin{aligned} \mathcal{A}_b &= b(P + R_1) - b(R_1) - (bP) + (O), \\ \mathcal{A}_c &= c(P + R_1) - c(R_1) - (cP) + (O), \\ \mathcal{A}_{b+c} &= (b + c)P + R_1) - (b + c)(R_1) - ((b + c)P) + (O). \end{aligned}$$

Odtud je vidět, že $\mathcal{A}_{b+c} = \mathcal{A}_b + \mathcal{A}_c + (g_1) + (g_2)$ a proto:

$$f_{b+c}(\mathcal{A}_Q) = f_b(\mathcal{A}_Q) \cdot f_c(\mathcal{A}_Q) \cdot \frac{g_1(\mathcal{A}_Q)}{g_2(\mathcal{A}_Q)}.$$

Celkem jsme tedy dokázali ze vstupních hodnot ($f_b(\mathcal{A}_Q)$, $f_c(\mathcal{A}_Q)$, bP, cP, $(b+c)P$) vypočítat hodnotu $f_{b+c}(\mathcal{A}_Q)$, vše s použitím několika aritmetických operací. Označme si tento algoritmus např. $\mathcal{M}$.

Algoritmus pro výpočet $f_n(\mathcal{A}_Q)$ funguje následovně:
1 - Nastavíme $Z = O, V = f_0(\mathcal{A}_Q) = 1, k = 1$, a ať $n = b_m \dots b_1 b_0$ je binární zápis čísla n.
2 - Iterujeme pro $i = m$ až $i = 0$:
2a - Pokud $b_i = 1$, pak nastavíme $V = \mathcal{M}(V, f_1(\mathcal{A}_Q), Z, P, Z+P),\ Z = Z+P,\ k = k+1$.
2b - Pro $i > 0$ nastavíme $V = \mathcal{M}(V, V, Z, Z, 2Z),\ Z = 2Z,\ k = 2k$.
3 - Po poslední iteraci, kdy $k = n$, vydáme $V = f_n(\mathcal{A}_Q)$.

Poznamenejme ještě, že při výpočtu potřebujeme vypočítat $f_1(\mathcal{A}_Q)$. To je ovšem snadné, protože funkce f_1 má divisor $(f_1) = (P+R_1) - (R_1) - (P) + (O)$ a tedy pokud si označíme přímku $a_1x + b_1y + c_1 = 0$ vedoucí body P, R_1, a dále přímku $x + c_2 = 0$ procházející bodem $(P+R_1)$, a definujeme funkce $h_1(x,y) = a_1x + b_1y + c_1$ a $h_2(x,y) = x + c_2$, pak $f_1(x,y) = h_2(x,y)/h_1(x,y)$, a to už dokážeme v G_T snadno vyčíslit.

Výpočet Tateova párování používá podobný postup.

3.3 Parametry týkající se eliptických křivek

Párování dává hodnoty v nějakém rozšíření tělesa $\mathbb{F}_{q^k}$. Parametr k je důležitý z hlediska výpočetní náročnosti párování. Pokud je k moc velké, výpočet je téměř nemožný. Typické hodnoty k v používaných implementacích jsou 1,2,3,4 a 6. Možné hodnoty k pro supersingulární křivky závisí na charakteristice p základního tělesa a také na paritě exponentu r a velikosti rozšíření $q = p^r$.

Pro $p = 2$ máme jednu supersingulární křivku danou rovnicí $y^2 + y = x^3$ (až na izomorfismy v algebraickém uzávěru $\mathbb{F}_2$). Pokud se ale omezíme na izomorfismy v $\mathbb{F}_q$, nabídne se nám hned několik křivek. Pro r liché má křivka $E : y^2 + y = x^3$ parametr $k = 2$ a $q+1$ bodů. Pro r sudé je parametr $k = 1$ a křivka má $(p^{r/2}+1)^2$ bodů. Křivky dané rovnicemi $y^2 + y = x^3 + x$ nebo $y^2 + y = x^3 + x + 1$ mohou pro r liché mít k rovno až 4.

Pro $p = 3$ máme jednu supersingulární křivku danou rovnicí $y^2 = x^3 + 2x + 1$ (až na izomorfismy v algebraickém uzávěru $\mathbb{F}_3$). Pokud se ale omezíme na izomorfismy

v $\mathbb{F}_q$, pak stejně jako v předchozím případě máme možnost hned několika křivek. Např. křivka $E : y^2 = x^3 + 2x + 1$ má:

r (mod 6)	0	1	2	3	4	5
k	1	6	3	2	3	6

Konečně pro prvočíslo $p > 4$, je-li r liché, je $k = 2$, je-li r sudé, je $k = 1$ nebo $k = 3$. Pro ne-supersingulární křivky lze zkonstrovat i větší k. Ovšem s rostoucím k těchto křivek jsou výpočty na nich téměř nereálné.

Prací zabývajících se konstrukcí eliptických křivek, jejich implementací a bezpečností je hodně. Není ale cílem této práce uvádět přehled všech výsledků, zmiňujeme pouze některé relevantní.

Kapitola 4

Schémata a jejich bezpečnost

V této kapitole rozebereme bezpečnostní otázky protokolů a uvedeme si příklady schémat založených na problémech, které jsme představili v předchozí kapitole.

4.1 Úvodní bezpečnostní definice

4.1.1 Základní definice

Definice:
non-malleability (NM) - (viz [18]) znamená, že útočník není schopen z původního šifrového textu $C = \mathsf{encrypt}(M)$ vytvořit nový šifrový text C^*, který by se následně dešifroval na otevřený text $M^* = \mathsf{decrypt}(C^*)$, a tento text M^* by byl svázán s původním M nějakou předem známou relací. Přitom se útočník vůbec nemusí o obsahu textu M dozvědět nic bližšího.

Definice:
indistinquishability (IND) - „nerozlišitelnost" znamená, že útočník není schopný rozlišit mezi dvěma šifrovými texty vzniklými ze dvou, útočníkovi známých, otevřených textů. Navíc povolujeme útočníkovi, aby si tyto dva otevřené texty zvolit. Podmínkou je, aby byli stejně dlouhé.

Definice:
CCA, CCA1, CCA2, CPA, CPA2, CMA, CMA2 - zkratka CCA znamená „chosen ciphertext attack", tedy útok vedený útočníkem, který má přístup k dešifrovacímu orákulu a může mu posílat dotazy na dešifrování libovolného řetězce znaků, jenž vydává za platný šifrový text. Tento útočník má samozřejmě přístup i k ostatním procedurám, které dané schéma nabízí, jako je šifrování (šifrovat může dokonce sám, protože pracujeme s šifrovacími schématy s veřejným klíčem) nebo procedura

extract v IBE schématech, pomocí níž získá soukromé a veřejné klíče libovolné identity, kterou si vybere.

Útočníka vždy modelujeme jako polynomiální pravděpodobnostní algoritmus. Útočník vystupuje ve schématech jako právoplatný uživatel schématu, který může mít přístup k soukromým klíčům několika jiných uživatelů. Útočníkovi rovněž přiřazujeme vlastní zdroj náhodných bitů, podle kterých se při výpočtu rozhoduje.

V případě CCA1 máme na mysli neadaptivní útok, kdy útočník odešle pouze jednu sadu dotazů $(C_1, \ldots, C_n)$ na dešifrovací orákulum. Poté, co dostane příslušnou sadu odpovědí $\mathsf{decrypt}(C_1), \ldots, \mathsf{decrypt}(C_n)$, se už nemůže dál dotazovat na dešifrování. CCA1 se také označuje jako „Lunch Time Attack".

Oproti tomu v případě CCA2 (nebo adaptivní CCA, případně „Midnight Attack"), má útočník možnost adaptivně odesílat dotazy, kdy reaguje na příchozí odpovědi od dešifrovacího orákula a na základě nově získané informace (ať už je jakákoliv) vytváří další dotazy.

Slabší CPA („chosen plaintext attack") bezpečnost definujeme pro útočníka, který nemá možnost dotazovat se dešifrovacího orákula. Jinak však může dělat všechny povolené operace, stejně jako při CCA útocích. Značíme CPA1 a CPA2 v závislosti na tom, jestli jde o adaptivního nebo neadaptivního útočníka. CPA bezpečnost se označuje také jako sémantická bezpečnost.

CCA, CCA1, CCA2, CPA, CPA2 vztahujeme na šifrovací, IBE a HIBE schémata. CMA naopak vztahujeme na podpisová schémata, kde útočník má možnost nechat si podepsat jím zvolené zprávy podpisovacím orákulem. V podpisových schématech není šifrovací ani dešifrovací procedura. Více v následující sekci o podpisových schématech.

Často budeme označení např. „CCA1" používat pro samotný útok prováděný ne-adaptibilním útočníkem. Budeme však toto značení používat také jako přídavné jméno označující schopnosti útočníka, tedy výrazem „CCA1 útočník" myslíme takového útočníka, který provádí ne-adaptibilní útok CCA1. Nebo také pro označení bezpečnosti schématu, tedy např. „IND-CCA1 bezpečné schéma" je takové schéma, které si zachovává vlastnost nerozlišitelnosti (IND) proti CCA1 útočníkovi. Podobně pro ostatní zkratky útoků. Speciálně pro podpisová schémata budeme používat značení „CMA bezpečnost", protože v podpisových schématech definujeme bezpečnost jako existenční nepadělatelnost při CMA a vlastnost non-malleability (NM) ani nerozlišitelnost (IND) zde nemá smysl.

Poznámka - NM a IND popisují „nežádoucí" operace ve schématu. CCA, CPA, CMA a další popisují schopnosti útočníka. Tyto dvě roviny bezpečnostních definic můžeme kombinovat a získáme tak finální definice bezpečnosti, kdy např. NM-CCA2 znamená, že CCA2 útočník má pouze zanedbatelnou šanci provést operaci narušující non-malleabilitu schématu, tedy modifikovat šifrový text nějakým, jemu

známým, způsobem. IND-CCA2 zase znamená, že CPA2 útočník není schopný s vyšší než zanedbatelnou pravděpodobností rozpoznat mezi dvěma šifrovými texty, které vznikly z jemu známých dvou otevřených textů.

V rozšíření verzi [18] je ukázáno, že vlastnost schématu „být NM-CCA2" je ekvivalentní vlastnosti „být odolný proti IND-CCA2". Existují ještě další vztahy mezi NM a IND v případě útoků CPA a CCA2. Více k tomuto tématu nalezneme v [18] a také na webu [37] – heslo „indistinquishability".

4.1.2 Podpisová schémata a CMA

Podpisová schémata se skládají ze tří algoritmů, keygen, sign a verify, jejichž funkce je patrná přímo z názvu. Mohou být pravděpodobnostní.

Definujeme bezpečnost podpisového schématu pomocí výhody útočníka, kterou může získat při maximální možné volnosti aktivit během reálném útoku. Přesněji: Jde o situaci, kdy útočník je schopný vytvořit novou zprávu a k ní platný podpis s využitím předchozí komunikace s orákulem, jenž mu poskytovalo podpisy pro libovolné, jím zvolené, zprávy. V literatuře se taková bezpečnost nazývá „existential unforgeability under a chosen message attack", neboli existenční nepadělatelnost při CMA (útok s volenými zprávami k podpisu).

Budeme značit CMA1 pro neadaptivního útočníka, tj. útočníka, který odešle sadu dotazů k podpisovacímu orákulu a poté, co obdrží zpět příslušné podpisy, už žádné další dotazy orákulu neposílá. Naopak CMA2 budeme používat pro adaptivního útočníka, který tvoří nové a nové dotazy na podpisovací orákulum na základě informací, které získá z předešlé komunikace s orákulem.

.....

Definice: Existenční nepadělatelnost při CMA2 je definována následující hrou mezi útočníkem $\mathcal{A}$ a vyzyvatelem:

Setup: vyzyvatel pustí algoritmus keygen, ponechá si soukromý klíč SK a veřejný klíč PK odešle útočníkovi $\mathcal{A}$.

Dotazy: Útočník $\mathcal{A}$ se adaptivně dožaduje u vyzyvatele na podpisy k nejvýše q_s zprávám $M_1, \ldots, M_{q_s} \in \{0,1\}^*$, které si $\mathcal{A}$ sám průběžně volí. Podpisy samozřejmě vztahuje k veřejnému klíči PK. Vyzyvatel odpovídá příslušnými podpisy $\sigma_i = \mathsf{sign}(M_i)$.

Výstup: Útočník $\mathcal{A}$ vydá dvojici (M, σ) a vyhrává hru pokud platí obě následující podmínky:

(1) M není mezi podepsanými zprávami $M_1, \ldots, M_{q_s}$

(2) σ je platný podpis zprávy M, tedy: $\mathsf{verify}(\mathrm{PK}, M, \sigma) = \mathrm{true}$.

Všimněme si, že pokud je útočník schopný vytvořit platný nový podpis k již podepsané zprávě, tak stejně nevyhrává hru. To odpovídá představě z reality, kdy nový podpis k již podepsané zprávě nemá pro útočníka praktickou hodnotu. Tato definice pochází z dílny Goldwasser, Micali a Rivest z roku 1988 (viz [20]). Boneh a Boyen v [4] použili definici silnější bezpečnosti (viz [1]), kdy zakazují útočníkovi vytvořit nový podpis k již podepsané zprávě – jejich podpisové schéma totiž dovoluje vytvořit více různých platných podpisů k jedné zprávě.

Stačí v předchozí hře namísto podminky (1) vzít podmínku, aby se (M, σ) nevyskytovalo mezi $(M_1, \sigma_1), \ldots, (M_{q_s}, \sigma_{q_s})$, čímž zajistíme, že útočník $\mathcal{A}$ vyhrává hru, i pokud se mu podaří vytvořit nový podpis k již podepsané zprávě. Takto zvyšujeme útočníkovu šanci $\mathrm{AdvSig}_{\mathcal{A}}$ na výhru předchozí hry, kterou však chceme omezit hodnotou ϵ (viz následující definice).

Je otázkou, jak může útočník napadnout některou z požadovaných užitečných vlastností podpisového schématu, pokud by dokázal nově podepisovat již podepsané zprávy.

Definice: Nechť $\mathrm{AdvSig}_{\mathcal{A}}$ je pravděpodobnost, že útočník $\mathcal{A}$ vyhraje hru z předchozí definice. Pravděpodobnost bereme přes všechny náhodné volby útočníka $\mathcal{A}$ i vyzyvatele.

Definice: Říkáme, že útočník $\mathcal{A}$ (t, q_s, ϵ)-*prolomil podpisové schéma*, pokud $\mathcal{A}$ pracuje nejvýše v čase t, $\mathcal{A}$ se dotáže vyzyvatele na nejvýše q_s podpisů a $\mathrm{AdvSig}_{\mathcal{A}}$ je alespoň ϵ.
Říkáme, že *podpisové schéma je* (t, q_s, ϵ)-*existenčně nepadělatelné při CMA2*, pokud neexistuje žádný CMA2 útočník $\mathcal{A}$, který by schéma (t, q_s, ϵ)-prolomil.

4.1.3 IBE schéma a IND-ID-CCA bezpečnost

Dalším, a zřejmě nejdůležitějším, využitím nových variant Diffie-Hellmanova problému je jejich aplikace při tvorbě IBE schémat. IBE = Identity Based Encryption. Tento koncept poprvé představil v roce 1984 Adi Shamir v [31]. Jde o šifrovací schémata s veřejným klíčem, kdy jako veřejný klíč můžeme zvolit libovolný znakový řetězec, např. emailovou adresu, IP adresu, a další. IBE se skládá ze 4 algoritmů:

setup generuje základní veřejné parametry schématu **params** a tajný **master-key**, který vlastní pouze jedna identita v celém schématu. Tato se označuje jako PKG (=private key generator). V ℓ-HIBE schématech s maximální možnou hloubkou identit ℓ bereme PKG jako identitu v nulové hloubce. Soukromý klíč PKG označujeme také jako $d_{\mathsf{ID}|0} = \mathsf{master\text{-}key}$.

extract s využitím **master-key** generuje soukromé klíče d_{ID} příslušející znakovým řetězcům $\mathsf{ID} \in \{0,1\}^*$.
encrypt šifruje zprávu M s použitím veřejného klíče ID
decrypt dešifruje zprávu C s použitím soukromého klíče d_{ID}.

Od procedury **encrypt** požadujeme, aby byla prováděna pravděpodobnostním algoritmem, tedy aby proces šifrování závisel na náhodné hodnotě. Pak každý otevřený text lze různě zašifrovat pomocí jednoho veřejného klíče, a přitom vznikne vždy jiný šifrový text. To se nám hodí např. v situaci, kdy několik odesílatelů chce nezávisle na sobě zašifrovat a poslat stejnou zprávu (např. vyplněný anonymní dotazník). Pak ze zachycených šifrových textů nejsme schopni poznat, že odpovídající otevřené texty jsou shodné.

Za standardní definici bezpečnosti šifrovacích schémat s veřejným klíčem se běžně bere ne-adaptivní (případně adaptivní) „chosen ciphertext security" (viz [30]), značíme IND-CCA (případně IND-CCA2). Je přirozené mít podobnou definici bepečnosti i v IBE schématech. V IBE může útočník $\mathcal{A}$ mít přístup k soukromým klíčům několika různých identit $\mathsf{ID}_1, \ldots, \mathsf{ID}_n$, proto je třeba v definici bezpečnosti IBE dát útočníkovi možnost dotazovat se vyzyvatele na soukromé klíče k libovolným identitám ID_i. Vzhledem k tomu, že chceme definovat schéma odolné vůči co nejsilnějšímu útočníkovi, dáme mu možnost dotazovat se adaptivně a navíc i možnost zvolit si identitu ID, na kterou bude vyzyvatelem testován. Samozřejmě tato identita ID musí být různá od všech identit ID_i, na které vyzyvatel vydal útočníkovi soukromé klíče v předchozí fázi. Boneh a Franklin v [10] označují takovou bezpečnost jako IND-ID-CCA, ale myslí tím adaptivního útočníka. My budeme pro rozlišení ne-adaptivního a adaptivního útočníka pracovat se značením IND-ID-CCA a IND-ID-CCA2.

.....

Definice: IND-ID-CCA2 bezpečnost IBE schématu je definována následující hrou mezi útočníkem $\mathcal{A}$ a vyzyvatelem:

Setup: Vyzyvatel ze vstupu k (=bezpečnostní parametr) pomocí algoritmu **setup** vygeneruje veřejné parametry schématu **params**, které pošle útočníkovi $\mathcal{A}$, a navíc vygeneruje **master-key**, který si ponechá.

1. fáze: Útočník $\mathcal{A}$ odesílá adaptivně dotazy $q_1, \ldots, q_m$ vyzyvateli. Dotazy jsou následujícího typu:
-dotaz na **extract**(ID_i). Vyzyvatel vygeneruje soukromý klíč d_i příslušný identitě ID_i pomocí algoritmu **extract**. Klíč d_i pak odešle útočníkovi $\mathcal{A}$.
-dotaz na **decrypt**(ID_i, C_i). Vyzyvatel pomocí algoritmu **extract** vygeneruje klíč d_i příslušný k ID_i a pak tento společně s algoritmem **decrypt** použije k dešifrování šifrového textu C_i. Výledný text odešle útočníkovi.

Výzva: Poté, co se útočník $\mathcal{A}$ rozhodne ukončit 1. fázi útoku, vytvoří $\mathcal{A}$ dva

stejně dlouhé texty M_1, M_2, zvolí si identitu ID^*, na kterou chce být testován vyzyvatelem, a odešle $(M_1, M_2, \mathsf{ID}^*)$ vyzyvateli. Samozřejmě tato identita ID^* nesmí být použita v 1. fázi mezi dotazy na **extract**. Vyzyvatel pak náhodně vybere bit $b \in \{0,1\}$ a spočte šifrový text $C^* = \mathsf{encrypt}(\mathsf{params}, \mathsf{ID}^*, M_b)$, který odešle útočníkovi.

2. **fáze**: Útočník $\mathcal{A}$ odesílá adaptivně další dotazy $q_{m+1}, \ldots, q_n$ vyzyvateli. Ty jsou následujícího typu:
-dotaz na $\mathsf{extract}(\mathsf{ID}_i)$, pro $\mathsf{ID}_i \neq \mathsf{ID}^*$. Vyzyvatel odpovídá stejně jako v 1. fázi.
-dotaz na $\mathsf{decrypt}(\mathsf{ID}_i, C_i)$, pro $(\mathsf{ID}_i, C_i) \neq (\mathsf{ID}^*, C^*)$. Vyzyvatel odpovídá stejně jako v 1. fázi.

Odhad: Jakmile je útočník $\mathcal{A}$ u konce s 2. fází útoku, vydá svůj odhad $b^* \in \{0,1\}$. Utočník $\mathcal{A}$ vyhrává hru, pokud byl jeho odhad správný, tj. pokud $b^* = b$.

V případě HIBE schémat, musíme útočníka omezit v dotazech týkajících se prefixů identity ID^*, kterou hodlá napadnout. V případě **extract** dotazů je třeba zakázat útočníkovi ptát se na soukromé klíče identit ID, které jsou prefixy ID^*, protože s jejich znalostí by si útočník mohl dopočítat soukromé klíče $d_{(\mathsf{ID}||\ldots)}$ identit ve větší hloubce hierarchie, tedy i klíč pro ID^*. V případě **decrypt** dotazů je třeba zakázat útočníkovi ptát se na dešifrování podle identit, které jsou prefixy ID^*, protože jimi dešifrovaná zpráva může být v nějaké relaci s původní zprávou, uvědomíme-li si, že veřejný klíč takovéto nadřazené identity je prefixem identity ID^*, na kterou $\mathcal{A}$ útočí.

Definice: Říkáme, že *IBE (příp. HIBE) schéma je* $(t, q_{\mathsf{ID}}, q_C, \epsilon)$*-bezpečné při CCA2*, pokud neexistuje CCA2 útočník $\mathcal{A}$ pracující nejvýše v čase t, který se dotáže vyzyvatele na nejvýše q_{ID} dotazů na **extract** a nejvýše q_C dotazů na **decrypt** a přitom má pravděpodobnost výhry předcházející hry alespoň ϵ (bráno přes všechny náhodné volby útočníka i vyzyvatele).

Definice IND-CCA (případně IND-CCA2) bezpečnosti je podobná předchozímu, až na to, že útočník $\mathcal{A}$ nemá možnost posílat dotazy na **extract**, protože v šifrovacích schématech (vyjma IBE) algoritmus **extract** vůbec není. Navíc je $\mathcal{A}$ testován pro náhodný veřejný klíč zvolený vyzyvatelem a ne pro klíč, který by si sám zvolil. Ne-adaptivní útočník navíc vynechává 2. fázi útoku.

V literatuře se běžně zavádí i sémantická bezpečnost schémat pro šifrování s veřejným klíčem, značíme IND-CPA (indistinquishability under chosen plaintext attack). Vystihuje obecnou představu bezpečnosti, kdy útočník není schopen se ze získaného šifrového textu dozvědět jakékoliv informace o odpovídajícím otevřeném textu. Formální definice sémantické bezpečnosti IBE schématu (IND-ID-CPA, IND-ID-CPA2) je velmi podobná předchozí definici, jenom v průběhu hry není útočníkovi povoleno dotazovat se na **decrypt**. Jinak je mu umožněno vytvářet dotazy

na extract i zvolit si veřejný klíč ID^*, na který chce být vyzyvatelem testován. Tedy jde o slabší útok, než je předchozí (adaptivní) CCA2.

Upozorněme ještě na vlastnost IBE a HIBE schémat, která se v anglické literatuře označuje jako „key-escrow". Znamená to, že ve schématu existuje zúčastněná strana, která vlastní master-key, ze kterého je schopna vypočítat tajemství všech uživatelů, vystupovat za ně a číst jim určené šifrové texty. Někteří autoři IBE a HIBE schémat proto navrhují treshold variantu PKG, tedy situaci, kdy práci PKG může vykonat pouze koalice alespoň t distribuovaných autorit z celkového počtu n oprávněných autorit, přitom žádná se nedozví společně vypočítané tajemství. Toto tajemství si z přijatých t zpráv zrekonstruuje až sama koncová identita.

Aplikace IBE a HIBE v praxi:

Zneplatnění veřejných klíčů - certifikáty veřejných klíčů obsahují datum zneplatnění. Při použití IBE schémat můžeme však připojovat datum zneplatnění přímo k veřenému klíči ID. Např. Alice bude šifrovat odesílanou zprávu veřejným klíčem PK =„ID_{Bob}||aktuální rok". Příjemce Bob si musí vždy na konci roku zažádat u PKG o nový soukromý klíč příslušející ke klíči PK =„ID_{Bob}||následující rok". Samozřejmě můžeme docílit libovolných časových period pro zneplatňování soukromých klíčů. Výhodou pro odesílatele je, že nemusí u autority online zjišťovat aktuální veřejný klíč příjemce, sám si ho zvládne odvodit. Také má odesílatel možnost zašifrovat zprávu do budoucnosti, protože příjemce bude schopný takovou zprávu dešifrovat až ve zvolený den v budoucnu.

Delegování soukromých klíčů - nyní uvažujme, že příjemce (Bob) figuruje ve schématu jako PKG, tedy vlastní tajný master-key. Autorita vydá Bobovi certifikát, že params jsou jeho veřejným klíčem schématu. Bob může generovat soukromé klíče určené jenom pro dané období, tj. pokud se takový soukromý klíč ztratí společně s notebookem, nebude kompromitován master-key, ale jenom klíč pro dané krátké období. Bob může také delegovat práva a povinnosti uživatelů. Pomocí master-key může např. vytvořit soukromé klíče pro čtení emailů v závislosti na předmětu zprávy. Pak zaměstnanci z technické podpory můžou číst pouze emaily pro ně určené, ale už nemůžou číst emaily pro obchodní oddělení, reklamace, atd. Zde odesílatel Alice vlastní pouze params a přitom může odesílat zašifrované emaily různým příjemcům v Bobově firmě. Bob má absolutní právo číst libovolný email.

Dopředu-bezpečné šifrování - úkolem je zajistit, aby zprávy zašifrované před kompromitací soukromého klíče zůstali utajené. Tedy je potřeba veřejný i soukromý klíč v průběhu času měnit. Dopředu-bezpečná podpisová schémata, schémata dohody na klíči i symetrická šifrovací schémata byla známa už dávno před rokem 2000, ale první dopředu-bezpečné šifrovací schéma s veřejným klíčem bylo představeno až v roce 2003 v práci [13]. Pomocí HIBE schématu můžeme šifrovat zprávy v 2^t časových úsecích vždy s jiným veřejným klíčem a odpovídajícím soukromým klíčem.

Pokud dojde ke kompromitaci jednoho ze soukromých klíčů (ne však **master-key**), bezpečnost ostatních klíčů není ohrožena a proto zprávy zašifrované jinými klíči (v jiných časových úsecích) zůstávají utajené.

Broadcast šifrování s veřejným klíčem - viz strana 32. HIBE se dá použít pro modifikaci existujících návrhů broadcastového šifrování ze symetrického nastavení na veřejné.

Šifrování do budoucna - IBE schéma se dá využít pro šifrování do budoucna s využitím důvěryhodné autority. Ta vlastní **master-key** a v každém časovém úseku zveřejňuje odpovídající soukromý klíč. Takže zprávy zašifrované pro daný časový úsek můžou být dešifrovány teprve až v tomto úseku. Problém tohoto postupu je v lineární náročnosti na veřejnou „nástěnku" s minulými soukromými klíči. Toho se můžeme zbavit „obrácením" schématu dopředu-bezpečného šifrování.
Mějme $T \leq 2^t$ časových úseků a dopředu-bezpečné šifrovací schéma založené na HIBE. Pak při zašifrování zprávy pro úsek $n \leq T$ použijeme dopředu-bezpečné šifrovací schéma pro časový interval $T - n$. Podobně autorita v úseku n zveřejní soukromý klíč pro interval $T - n$. Tento jediný soukromý klíč stačí k vygenerování soukromých klíčů pro následující intervaly $T - n, \ldots, T$ dopředu-bezpečného šifrovacího schématu. Takže každý může dešifrovat zprávy z časových úseků $1, \ldots, n$.

Podpisové schéma - Soukromým klíčem podpisového schématu je **master-key** IBE schématu. Veřejný klíč podpisového schématu odpovídá veřejným parametrům IBE schématu. Podpisem zprávy M je soukromý klíč d_M IBE schématu odpovídající identitě $\mathsf{ID} = M$. K ověření podpisu je třeba zvolit náhodné $r \in \mathbb{Z}_p$, zašifrovat r veřejným klíčem M (jako v IBE schématu), a pak dešifrovat výsledek s použitím soukromého klíče d_M, který je však v podpisovém schématu veřejně známý jako podpis M. Pokud ověření projde, podpis přijmeme. Pokud ne, podpis odmítneme. Jestliže je IBE schéma IND-ID-CCA2 bezpečné, pak je podpisové schéma existenčně nepadělatelné při CMA2.

4.1.4 Canetti, Halevi, Katz: konstrukce CPA2→CCA2

V článku [13] z roku 2003 představili tito autoři obecnou metodu, jak z IND-sID-CPA2 bezpečného IBE schématu vytvořit IND-sID-CCA2 bezpečné schéma, ale jejich konstrukce se opírá o schéma neinteraktívního důkazu s nulovou znalostí, což je dosud velmi neefektivní kryptografický primitiv. Proto v současnosti máme IND-sID-CCA2 bezpečná IBE schémata založená na jejich druhé konstrukci uvedené v článku [14] z roku 2004. Jde o metodu, jak z IND-sID-CPA2 bezpečného ℓ-HIBE schématu vytvořit IND-sID-CCA2 bezpečné $(\ell - 1)$-HIBE schéma. K tomu potřebují navíc existenčně nepadělatelné podpisové schéma proti CMA2

útočníkovi. Vše se obejde bez náhodných orákul, tedy výsledné schéma má prokazatelnou bezpečnost ve standardním modelu.

Nechť Π = (setup, extract_Π, encrypt, decrypt) je IND-sID-CPA2 bezpečné 2-HIBE schéma a Σ = (keygen_Σ, sign, verify) je CMA2 bezpečné podpisové schéma. Vytvoříme nové IBE schéma Π' následovně:

.....

$\text{setup}'(p)$: PKG spustí $\text{setup}(p)$ a získá parametry schématu - grupu G, veřejný klíč PK a tajný master-key.

$\text{extract}'(\text{ID}, d_{\text{ID}|j-1})$: Pro $j = 1$ je aktivní PKG, který generuje soukromý klíč identity ID. Postupuje stejně jako ve schématu Π, tedy spustí extract a získá: $\text{extract}_\Pi(\text{ID}, \text{master-key}) = d_{\text{ID}}$.
Pro $j = 2$ je aktivní identita ID první hladiny, která s využitím svého soukromého klíče generuje soukromý klíč podřazené identitě $\text{ID}||w$ ve druhé hladině. Identita ID postupuje stejně jako ve schématu Π, tedy spustí extract a získá: $\text{extract}_\Pi(\text{ID}||w, d_{\text{ID}}) = d_{\text{ID}||w}$.

$\text{encrypt}'(\text{ID}, \text{params}, M)$: Identita ID nejdříve spustí $\text{extract}_\Sigma(p)$ odkud získá klíče V_{sig} a K_{sig}. Pak vypočte $C_0 = \text{encrypt}(\text{ID}||V_{\text{sig}}, \text{params}, M)$. $\sigma = \text{sign}(C_0, K_{\text{sig}})$ a vydá šifrový text:

$$C = (V_{\text{sig}}, C_0, \sigma).$$

$\text{decrypt}'(\text{ID}, d_{\text{ID}}, C)$: Označme si $C = (V_{\text{sig}}, C_0, \sigma)$. Nejdříve ID ověří, že σ je platný podpis zprávy C_0 pomocí procedury $\text{verify}(C_0, \sigma, V_{\text{sig}})$. Pokud podpis nesouhlasí, decrypt' skončí. Pokud souhlasí, pak ID spustí $\text{extract}'(\text{ID}||V_{\text{sig}}, d_{\text{ID}})$ a získá soukromý klíč $d_{\text{ID}||V_{\text{sig}}}$. Pomocí decrypt vypočte:

$$\text{decrypt}(\text{ID}||V_{\text{sig}}, d_{\text{ID}||V_{\text{sig}}}, C_0) = M.$$

.....

Rozeberme nyní, proč je nové IBE schéma Π' bezpečné proti CCA2. Označme si $C^* = (V^*, C_0^*, \sigma^*)$ šifrový text vyzyvatele, kterým testuje útočníka. Protože Π je CPA2 bezpečné, nelze z C^* usuzovat prozatím nic o bitu b ukrytém v C^*, a další dotazy na extract' útočníkovi také nepomůžou. Navíc V^* je nezávislé na b a σ^* je podpis C^*, takže z něj útočník b nedostane. Chceme tedy, aby dotazy na decrypt' v další fázi útoku nijak nepomohly útočníkovi určit bit b. Útočník pošle dotaz $C = (V, C_0, \sigma)$ různý od C^*, ale $V = V^*$, pak decrypt' skončí, protože útočník není schopen vytvořit nový platný podpis σ zprávy C_0. Pokud je $V \neq V^*$, pak útočník sice vlastní podpisový klíč K a může vytvářet platné podpisy, ale decrypt' vydává odpovědi vztažené k identitě $\text{ID}||V \neq \text{ID}||V^*$, z nichž vůbec není jasné, jak spočítat bit b, protože o složení šifrovací a neodpovídající dešifrovací funkce předpokládáme, že se chová jako náhodné zobrazení mezi otevřenými texty.

Formální důkaz uvedené úvahy je uveden v článku [14].

Konstrukce od Canetti, Halevi, Katz byla dále vylepšena v článku [12], kde pomocí této nové metoda prodloužíme šifrový text jenom o MAC (message authentication code) a závazek (commitment). Metoda redukuje ℓ-HIBE schéma na $(\ell - 1)$-HIBE schéma.

4.1.5 Schéma broadcastového šifrování

Od schématu pro broadcastové šifrování požadujeme, aby každý uživatel byl schopen zašifrovat zprávu pro libovolnou cílovou podmnožinu S uživatelů. Tito zamýšlení příjemci jsou schopni s využitím svého soukromého klíče dešifrovat přijatý šifrový text. Pokud navíc ani spolupráce všech uživatelů mimo S nevede k získání jakékoliv informace o obsahu odeslané zprávy, říkáme, že schéma je odolné vůči koalicím (collusion resistant). Zajímá nás velikost soukromého klíče každého uživatele, velikost veřejného klíče a velikost šifrového textu, vše v závislosti na počtu uživatelů. Závislost na velikosti povolené koalice, vůči níž je schéma pořád ještě bezpečné, byla dalším parametrem dřívějších návrhů broadcastových šifrovacích schémat, ale ve schématu odolnému vůči koalicím nemá smysl.

Definice: Schéma broadcastového šifrování se skládá ze tří algoritmů:

setup(N): Vygeneruje veřejný klíč PK a soukromé klíče $d_1, \ldots, d_N$ pro všech N uživatelů.

encrypt(S, PK): Pro veřejný klíč PK a množinu $S \subseteq \{1, \ldots, N\}$ vydá pár (Hdr, K), kde Hdr nazýváme hlavička a K je klíč pro (symetrické) dešifrování zpráv.

Pro zprávu M označme C_K její (symetrické) zašifrování klíčem K. Šifrový text odesílaný množině uživatelů S sestává z (S, Hdr, C_M). Pár (S, Hdr) se nazývá úplná hlavička.

decrypt(S, i, d_i, Hdr, PK): Pokud $i \in S$, pak decrypt vrátí z hlavičky Hdr klíč K, a ten se použije k dešifrováví C_K na zprávu M. PK je potřebný pro výpočet klíče z hlavičky.

Definice: CCA2 bezpečnost schématu broadcastového šifrování je definována následující hrou mezi útočníkem $\mathcal{A}$ a vyzyvatelem:

Inicializace: $\mathcal{A}$ si vybere skupinu příjemců $S^* \in \{1, \ldots, N\}$, na kterou chce zaútočit a odešle S vyzyvateli.

Setup: Vyzyvatel spustí setup(N), získá PK a $d_1, \ldots, d_N$, a pošle útočníkovi PK a všechny d_i pro $i \notin S^*$.

Fáze 1: Útočník $\mathcal{A}$ se adaptivně ptá na dešifrovací dotazy $q_1, \ldots, q_m$, kde $q_i = (u, S, \mathrm{Hdr}), S \subseteq S^*, u \in S$. Vyzyvatel spustí $\mathsf{decrypt}(S, u, d_u, \mathrm{Hdr}, \mathrm{PK})$ a odešle výsledek útočníkovi.

Výzva: Vyzyvatel spustí $\mathsf{encrypt}(S^*, \mathrm{PK})$ a obdrží (Hdr^*, K^*). Vyzyvatel zvolí náhodně bit $b \in \{0, 1\}$, a nastaví $K_b = K^*$. K_{1-b} = (náhodný klíč). Odešle útočníkovi $(\mathrm{Hdr}^*, K_0, K_1)$.

Fáze 1: Útočník $\mathcal{A}$ se dále adaptivně ptá na dešifrovací dotazy $q_{m+1}, \ldots, q_k$, kde $q_i = (u, S, \mathrm{Hdr}), S \subseteq S^*, u \in S, \mathrm{Hdr} \neq \mathrm{Hdr}^*$. Vyzyvatel dešifruje pomocí $\mathsf{decrypt}(S, u, d_u, \mathrm{Hdr}, \mathrm{PK})$ a odešle výsledek útočníkovi.

Odhad: $\mathcal{A}$ vydá svůj tip $b^* \in \{0, 1\}$ na hodnotu bitu b a vyhrává hru, pokud $b^* = b$.

.....

Definice: Říkáme, že útočník $\mathcal{A}$ (t, q_k, N, ϵ)-prolomil schéma broadcastového šifrování pro N uživatelů, pokud $\mathcal{A}$ pracuje nejvýše v čase t, dotáže se vyzyvatele na nejvýše q_k decrypt dotazů a pravděpodobnost výhry předcházející hry je alespoň ϵ (bráno přes všechny náhodné volby útočníka i vyzyvatele).
Říkáme, že schéma broadcastové šifrování je (t, q_k, N, ϵ)-CCA2 bezpečné, pokud neexistuje žádný útočník $\mathcal{A}$, který by schéma (t, q_k, N, ϵ)-prolomil.

Aplikace broadcastového šifrování v praxi:

Sdílení souborů v šifrovaných file systémech: Broadcastové šifrování můžeme použít při kontrole přístupu uživatelů k zašifrovaným souborům na sdíleném disku. Data souboru šifrujeme klíčem K_F a k nim do hlavičky přidáme zašifrování klíče K_F, který si oprávněný uživatel může pomocí svého soukromého klíče dešifrovat a mít tak přístup k obsahu souboru. Běžně se do hlavičky přidává jedna položka za každého oprávněného uživatele, což je však málo efektivní. Díky broadcastovému šifrování můžeme mít v hlavičce každého chráněného souboru pouze 2 položky - popis oprávněné skupiny uživatelů S a šifrový text konstantní velikosti ukrývající klíč K_F. Potřebujeme mít navíc někde na veřejně přístupném místě uložený soukromý klíč, který má velikost lineární vzhledem k počtu všech uživatelů systému. Navíc (při použití schématu popsaného na straně 53) není problém pro vlastníka souboru přidávat a odebírat uživatele z jednotlivých skupin, případně přidávat nové uživatele celého systému.

Šifrovaný email podle seznamů kontaktů: Uvažme, že máme velkou skupinu uživatelů S, kteří dostanou od autority veřejný klíč (lineární v $|S|$) a chtějí posílat šifrovanou poštu různým skupinám kontaktů G. Autorita vygeneruje dodatečný jednoprvkový veřejný klíč příslušející každé skupině kontaktů G a soukromé klíče členům této skupiny. Uživatelé tedy budou mít seznamy skupin kontaktů G, kterých jsou členy, jednoprvkové veřejné klíče těchto skupin a příslušné soukromé klíče pro přístup k emalům. Lineárně dlouhý (v $|S|$) veřejný klíč stačí jen jeden pro celý

systém. Není problém updatovat seznamy kontaktů, stačí upozornit všechny členy seznamu G, že byla zvětšena nebo zmenšena skupina o uživatele z. Členové skupiny si už pak sami updatují své hodnoty.

Ochrana obsahu DVD: DVD přehrávače mají každý svůj tajný klíč k dešifrování obsahu DVD. Pokud však dojde ke kompromitaci tohoto klíče, mohou útočníci vytvořit pirátský přehravač. Broadcastové šifrování lze využít k zneplatnění jejich klíče a tedy i přístupu k obsahu DVD. Protože však potřebujeme veřejný klíč každého distributora DVD k dešifrování jeho DVD disků, musí být tento veřejný klíč uložen v hlavičce každého DVD. To dává základní fixní paměťovou náročnost na DVD. Dál roste hlavička pomalým tempem s přibývajícími zneplatněnými klíči. Jiné postupy ochrany obsahu DVD pracují tak, že sice nemají základní fixní paměťovou náročnost a také přidávají data o zneplatněných přehrávačích do hlavičky DVD (lineárně) s každým kompromitovaným přehrávačem, ale každá tato položka je několikrát větší než u broadcastového šifrování. Broadcastové šifrování se proto vyplatí použít až případě, kdy je kompromitováno větší množství přehrávačů.

4.1.6 Model náhodných orákul – Bellare, Rogaway

Model náhodných orákul (random oracle model) byl představen v práci [3] od Bellare a Rogawaye z roku 1993. Ukázali, že použití snadno implementovatelných pseudonáhodných funkcí (modifikace DES, MD5, SHA1, atd.) při konstrukci kryptografických schémat namísto náhodných orákul nevede ke ztrátě na bezpečnosti, kterou jsme schopni dokázat pro schémata postavená na náhodných orákulech. Jejich návrh, jak konstruovat schémata je následující:

1 - Najdeme formální definici kryptografického problému Π, kde všechny zúčastněné strany (včetně případného útočníka) mají přístup k náhodnému orákulu R.
2 - Navrhneme schéma P pro tento problém Π.
3 - Dokážeme všechny požadované vlastnosti schématu P, které po něm v definici problému požadujeme.
4 - Nahradíme náhodné orákulum v R výpočtem praktické pseudonáhodné funkce.

Bezpečnost schémat dokázaná v modelu náhodných orákul je založena na heuristické úvaze. Z takto dokázané bezpečnosti však neplyne, že schéma je bezpečné v praxi. Schéma totiž může být prolomeno díky budoucím poznatkům o implementované pseudonáhodné funkci, i když při použití jiné pseudonáhodné funkce by prolomeno nebylo.

4.2 Schémata

Uvedeme si postupně schémata pro šifrování s veřejným klíčem, podpisová schémata, traitor tracing schémata, IBE, HIBE schémata a schéma broadcastového šifrování. U jednotlivých návrhů schémat nás budou zajímat zejména parametry jako jsou náročnost jednotlivých procedur, kterými jsou schémata definována, velikost soukromého a veřejného klíče, velikost šifrového textu a otázka prokazatelné bezpečnosti ve standardním modelu nebo v modelu náhodných orákul.

Násobení v grupě je méně časově náročná položka než umocňování, proto jeho vliv na časovou náročnost celého výpočtu neuvažujeme. Zajímají nás operace umocňování, dělení a párování.

4.2.1 Šifrovací schéma ElGamal

Šifrovací schéma zvané ElGamal, navržené v roce 1985 v [19] je relativně jednoduché schéma pro šifrování s veřejným klíčem, které však nesplňuje mnoho požadavků z dnešního pohledu na bezpečnost kryptografických schémat.

Schéma funguje následovně:

Nechť p je velké prvočíslo a g je s ním nesoudělné přirozené číslo. Počítáme v grupě $G = (\mathbb{Z}_p^*, \cdot)$ velikosti $p-1$ generované prvkem g. Odesílatel chce zašifrovat a odeslat zprávu $M \in \mathbb{Z}_p^*$.

keygen(p): Vygenerujeme náhodné $x \in \mathbb{Z}_p^*$ a vypočteme $y = g^x$. Veřejný klíč je y, soukromý klíč je x.

encrypt(M, y): Zvolíme náhodné $k \in \mathbb{Z}_p^*$ a vypočteme šifrový text jako dvojici $C = (g^k, y^k M)$.

decrypt(C, x): Dešifrujeme přijatý šifrový text $C = (A, B)$ následovně:

$$\frac{B}{A^x} = \frac{y^k M}{(g^k)^x} = \frac{g^{xk} M}{g^{kx}} = M.$$

Výkonnost: Šifrování zabere výpočet dvou umocňování, dešifrování zabere jedno umocňování a jedno dělení (všechno v $\mathbb{Z}_p^*$). Jsou to tedy velmi rychlé procedury. Šifrový text má dvě položky, tedy řádově $2\log(p)$ bitů. Sémantická bezpečnost ElGamalu je založená na obtížnosti řešit DDH problém. Ale proti CCA2 není ElGamal vůbec bezpečný (viz strana 61).

4.2.2 Šifrovací schéma – Cramer, Shoup

Šifrovací schéma s veřejným klíčem z roku 1998 podle Cramer, Shoup (viz [16]) bylo v té době první efektivní šifrovací schéma s prokazatelnou bezpečné proti adaptivnímu útoku s volenými šifrovými texty, tedy IND-CCA2 bezpečné. Je založeno na obtížnosti řešit DDH problém.

Schéma je následující:

.....

keygen(p): Podle bezpečnostního parametru p zvolíme grupu G. Vybereme náhodně generátory $g_1, g_2 \in G$ a náhodné $x_1, x_2, y_1, y_2, z \in \mathbb{Z}_p^*$. Vypočteme:

$$c = g_1^{x_1} g_2^{x_2}, \quad d = g_1^{y_1} g_2^{y_2}, \quad h = g_1^z.$$

Navíc vybereme kryptograficky bezpečnou hashovací funkci $\mathrm{H} : \{0,1\}^* \to \mathbb{Z}_p$.

Veřejný klíč $\mathrm{PK} = (g_1, g_2, c, d, h, \mathrm{H})$, soukromý klíč $\mathrm{SK} = (x_1, x_2, y_1, y_2, z)$.

encrypt(PK, M): Předpokládejme, že $M \in G$. Zvolíme náhodně $r \in \mathbb{Z}_p^*$ a spočteme:

$$u_1 = g_1^r, \quad u_2 = g_2^r, \quad w = h^r M, \quad \alpha = \mathrm{H}(u_1, u_2, w), \quad v = c^r d^{\alpha r}.$$

Šifrový text je čtveřice $C = (u_1, u_2, w, v)$.

decrypt(SK, C): Označme si šifrový text $C = (A_1, A_2, A_3, A_4)$. Vypočteme $\alpha = \mathrm{H}(A_1, A_2, A_3)$ a ověříme, zda:

$$u_1^{x_1 + y_1 \alpha} \cdot u_2^{x_2 + y_2 \alpha} = v.$$

Pokud ověření projde, pak dešifrovací algoritmus vydá hodnotu $w / u_1^z = M$.

.....

Ověřme funkčnost prováděného testu (tedy, že korektní šifrování a dešifrování vede k platnému ověření):

$$u_1^{x_1 + y_1 \alpha} \cdot u_2^{x_2 + y_2 \alpha} = g_1^{r(x_1 + y_1 \alpha)} \cdot g_2^{r(x_2 + y_2 \alpha)} = (g_1^{x_1} g_2^{x_2})^r \cdot (g_1^{y_1} g_2^{y_2})^{r\alpha} = c^r d^{r\alpha} = v,$$

a že výstupem dešifrovací procedury je původní otevřený text:

$$\frac{w}{u_1^z} = \frac{h^r M}{g_1^{rz}} = \frac{g_1^{zr} M}{g_1^{rz}} = M.$$

Výkonnost: Schéma pracuje s hashovací funkcí $\mathrm{H} : \{0,1\}^* \to \mathbb{Z}_p$. Veřejný klíč obsahuje pět prvků grupy G a definici hashovací funkce H, soukromý klíč obsahuje pět prvků $\mathbb{Z}_p^*$. Šifrový text má velikost čtyř prvků grupy G, kdy k výpočtu šifrového

textu je potřeba pěti umocňování v G a jedno hashování. Dešifrovací procedura provádí jedno hashování při ověřovacím testu a jedno umocňování a dělení pro výpočet otevřeného textu M.

IND-CCA2 bezpečnost schématu je dokázána v modelu náhodných orákul, což je idealizovaný model, s lepšími bezpečnostními výsledky než standardní model. Tedy otázka existence efektivního IND-CCA2 schématu prokazatelně bezpečného ve standardním modelu nebyla schématem od autorů Cramer, Shoup vyřešena. Ale i přesto bylo v té době jejich schéma velkým přínosem pro kryptografii.

Jelikož je NM-CCA2 ekvivalentní IND-CCA2, je toto šifrovací schéma příkladem non-malleabilního schématu při CCA2. NM je oproti (malleabilnímu) šifrovacímu schématu ElGamal zaručena použitím kryptograficky bezpečné hashovací funkce H. Pokud by chtěl útočník pozměnit zprávu M ukrytou v šifrovém textu $C = (A_1, A_2, A_3, A_4)$, musel by změnit $A_3 = w = h^r M$, čímž by ovlivnil i výstup hashovací funkce $\mathrm{H}(A_1, A_2, A_3)$, který se počítá v ověřovací fázi dešifrovací procedury.

Pokud bychom se spokojili jenom s CPA bezpečností, pak můžeme použít jednodušší schéma, kde úplně vynecháme výpočet d, y_1, y_2 a hashovací funkci H. Budeme počítat $v = c^r$ a při dešifrování otestujeme, zda $v = u_1^{r_1} u_2^{r_2}$. Jinak zůstává toto CPA bezpečné schéma stejné jako předchozí schéma.

4.2.3 Podpisové schéma – Boneh, Boyen

Podpisové schéma z roku 2004 tak jak jej navrhli Boneh a Boyen, viz [4], je založeno na předpokladu, že vyřešit 1-SDH problém je těžké. V tomto článku byl poprvé představen ℓ-SDH problém.

Schéma je následující:

.....

keygen(p): Na základě bezpečnostního parametru p zvolíme bilineární grupy (G_1, G_2). Vybereme náhodný generátor $P_2 \in G_2$ a k němu dopočítame $P_1 = \psi(P_2)$. Vybereme náhodně $x, y \in \mathbb{Z}_p^*$ a vypočteme $U = xP_2$, $V = yP_2$, $z = e(P_1, P_2)$.

Veřejný klíč $\mathrm{PK} = (P_1, P_2, U, V, z)$, soukromý klíč $\mathrm{SK} = (x, y)$.

sign(SK, M): Předpokládejme, že zpráva $M \in \mathbb{Z}_p^*$. Vybereme náhodné $r \in \mathbb{Z}_p^*$ a vypočteme:

$$\alpha = x + M + ry, \quad \sigma = \frac{1}{\alpha} P_1.$$

Podpisem zprávy je dvojice (σ, r).

verify(PK, (σ^*, r^*), M): S veřejným klíčem PK $= (P_1, P_2, U, V, z)$, zprávou $M \in \mathbb{Z}_p^*$ a podpisem (σ^*, r^*) provedeme ověření, zda-li:

$$e(\sigma^*, U + P_2 M + r^* V) = z.$$

Pokud vztah platí, podpis přijmeme. Jinak odmítneme.

.....

Ověřme funkčnost prováděného testu (tedy, že korektně vytvořený podpis vede k platnému ověření):

$$e(\sigma, U + P_2 M + rV) = e(\frac{1}{\alpha} P_1, xP_2 + P_2 M + P_2 yr) = e(\frac{1}{\alpha} P_1, \alpha P_2) = z.$$

Předpokládejme, že útočník chce vytvořit podpis (σ^*, r^*) k jím zvolené zprávě M^*. Označme si $\alpha^* = x + r^* y$, pro $M = 0$. Útočník toto α^* nezná.
K vytvoření σ^* musí ze znalosti $(P_1, P_2, \alpha^* P_2 = U + r^* V)$ vypočítat

$$\sigma^* = \frac{1}{\alpha^* + M^*} P_1,$$

což odpovídá problému 1-SDH pro konstantu $c = M^*$ (viz definice ℓ-SDH na straně 11).

Výkonnost: Podpis má pouze dvě složky, jeden prvek G_1 a jeden prvek $\mathbb{Z}_p^*$, tedy řádově $2\log(p)$ bitů. Vytvoření podpisu zabere jedno dělení v $\mathbb{Z}_p^*$ a jedno umocňování v G_1. Při ověřování je potřeba spočítat jedno párování a dvě umocňování. To je však řádově rychlejší než výpočet párování. Všimněme si také, že P_1, z lze z veřejného klíče dopočítat. V [4] je uveden důkaz existenční nepadělatelnosti schématu při CMA2, navíc ve standardním modelu.

4.2.4 Podpisové schéma v.2 – Boneh, Boyen

Uvádíme i zjednodušenou verzi předchozího podpisového schématu, která je však bezpečná pouze proti neadaptivnímu útoku CMA1. Při důkazu plné bezpečnosti předchozího schématu (viz [4]) se využívá vlastností tohoto jednoduššího schématu. Navíc je toto schéma zajímavé tím, že podpis má velikost pouze jednoho prvku grupy G.

Další důvod, proč uvádíme toto jednodušší schéma, je konstrukce uvedená v [4]. Ta z libovolného podpisového schématu bezpečného proti CMA1 vytvoří nové schéma, které je prokazatelně bezpečné proti (adaptivnímu) CMA2.
Konstrukce využívá dvě hashovací funkce. Důkaz bezpečnosti takto vzniklého schématu je předveden v modelu náhodných orákul. Zajímavá na nově vzniklém schématu je velikost podpisu. Ta se zvětší jen o jeden bit vůči podpisu v originálním CMA1 bezpečném schématu.

Zjednodušená verze podpisového schématu od Boneh, Boyen je následující:

---

keygen(p): Na základě bezpečnostního parametru p zvolíme bilineární grupy (G_1, G_2). Vybereme náhodný generátor $P_2 \in G_2$ a k němu dopočítame $P_1 = \psi(P_2)$. Vybereme náhodně $x \in \mathbb{Z}_p^*$ a vypočteme $V = yP_2$, $z = e(P_1, P_2)$.

Veřejný klíč PK $= (P_1, P_2, V, z)$, soukromý klíč SK $= x$.

sign(SK, M): Předpokládejme, že zpráva $M \in \mathbb{Z}_p^*$. Vypočteme: $\alpha = x + M$, $\sigma = 1/\alpha P_1$. Podpisem zprávy je σ.

verify(PK, σ^*, M): S veřejným klíčem PK $= (P_1, P_2, V, z)$, zprávou $M \in \mathbb{Z}_p^*$ a podpisem σ provedeme ověření, zda-li:

$$e(\sigma^*, P_2 M + V) = z.$$

Pokud vztah platí, podpis přijmeme. Jinak odmítmene.

Ověření funkčnosti testu i schématu je podobné jako v případě předchozího CMA2 bezpečného schématu od Boneh, Boyen.

Výkonnost: Schéma vyžaduje jedno dělení a umocnění při výpočtu podpisu, a jedno párování a umocnění pro ověření podpisu. Podpis zprávy je tvoří jen jeden prvek v G_1. Schéma je bezpečné proti CMA1.
Aplikujeme-li konstrukci uvedenou v [4], získáme schéma využívající dvě hashovací funkce, které je CMA2 bezpečné v modelu náhodných orákul a navíc má velmi krátký podpis. Pouze o 1 bit delší než v původním schématu bez hashovacích funkcí.

4.2.5 Podpisové schéma – Waters

Waters v roce 2005 navrhl efektivní IBE schéma s plnou bezpečností ve standardním modelu, jako reakci na první (avšak neefektivní) IBE schéma s IND-ID-CCA2 bezpečností od Boneh, Boyen (viz [6]). Watersovo IBE schéma uvádíme na straně 49.

Toto IBE schéma (postavené na problému DBDH) dále modifikoval na podpisové schéma tak, že jeho bezpečnost závisí na slabším předpokladu, tj. na obtížnějším problému CDH. Přitom nepoužil generickou konstrukci (viz [10] z roku 2003) jak z IBE schématu udělat podpisové schéma.

Podpisové schéma je následující:

---

setup(p): Na základě bezpečnostního parametru p zvolíme grupu G a vygenerujeme parametry schématu stejně v IBE schématu: Zvolíme náhodně $\alpha \in \mathbb{Z}_p^*$, generátor $P \in G$, nastavíme $P_1 = \alpha P$, zvolíme náhodné $P_2 \in G$ a vypočteme tajné αP_2. Dále zvolíme náhodné $U' \in G$ a vektor náhodných hodnot $V = (U_1, \ldots, U_n) \in G^n$.

Veřejný klíč $\mathrm{PK} = (P, P_1, P_2, U', V)$, soukromý klíč $\mathrm{SK} = \alpha P_2$.

$\mathsf{sign}(\mathrm{SK}, M)$: Ať $M = (M_1, \ldots, M_n)$ je n-bitový řetězec representující danou zprávu. Zvolíme náhodně $r \in \mathbb{Z}_p^*$ a vypočteme:

$$\sigma = \left(\alpha P_2 + r(U' + \sum_{M_i=1} U_i),\ rP \right).$$

$\mathsf{verify}(\mathrm{PK}, \sigma, M)$: Chceme ověřit, že $\sigma = (\sigma_1, \sigma_2)$ je podpis zprávy M. Ověřme tedy, že:

$$\frac{e(\sigma_1, P)}{e\left(\sigma_2,\ U' + \sum_{M_i=1} U_i\right)} = \frac{e(\alpha P_2,\ P) + e\left((U' + \sum_{M_i=1} U_i),\ P\right)^r}{e\left(rP,\ U' + \sum_{M_i=1} U_i\right)} = e(P_2, P_1).$$

.....

Výkonnost: Pro podpis potřebujeme provést v průměru $n/2$ grupových operací v G a dále tři umocňování. Pro ověření podpisu potřebujeme spočítat dvě párování a jedno dělení. Veřejný klíč má $(n+4)$ prvky grupy, a podpis má velikost dvou prvků grupy. Schéma je existenčně nepadělatelné při CMA2 s důkazem ve standardním modelu.

Ukažme si, že schéma je založené na CDH problému:
Útočník zná $(P,\ P_1 = \alpha P,\ P_2)$ a chce αP_2, aby mohl vytvářet podpisy. Nechť $P_2 = \beta P$ pro nějaké neznámé $\beta \in \mathbb{Z}_p$. Pak útočník potřebuje ze znalosti $\beta P, \alpha P$ spočítat $\alpha\beta P = \alpha P_2$, což je CDH problém.

4.2.6 Schéma skupinového podpisu – Boneh, Boyen, Shacham

Skupinové podpisové schéma tak jak jej navrhli pánové Boneh, Boyen a Shacham v článku [8].

Od schématu skupinového podpisu vyžadujeme:
1 - správně vytvořený podpis musí být ověřitelný a případně vystopovatelný k podpisovateli.
2 - anonymitu podpisů, tj. aby bez účasti skupinového managera nebylo možné odhalit identitu podpisovatele na základě znalosti libovolného počtu podpisů od kohokoliv ze skupiny.
3 - aby z podvodných podpisů (včetně účasti managera na podvodu) byla zpětně vystopovatelná identita aspoň jednoho podvádějícího uživatele.

Formální definice anonymity a vystopovatelnosti je obsáhlejší. Je uvedena např. v [2].

Anonymita podpisů je založena na obtížnosti řešit DLDH problém (viz definice na straně 10), zatímco trasovací algoritmus managera skupiny stojí na problému ℓ-SDH (viz definice na straně 11).

Schéma skupinového podpisu se skládá z těchto čtyř algoritmů:

.....

keygen(N, p): Chceme schéma pro skupinu N uživatelů. Na základě bezpečnostního parametru p zvolíme bilineární grupy (G_1, G_2). Zvolíme P_2 náhodný generátor G_2, $P_1 = \psi(P_2)$. Vybereme náhodné $Q \in G_1 \setminus \{1\}$, náhodné $\xi_1, \xi_2 \in \mathbb{Z}_p^*$ a $U, V \in G_1$ tak, že $\xi_1 U = \xi_2 V = Q$. Ještě vybereme náhodné $\gamma \in \mathbb{Z}_p^*$ a nastavíme $W = \gamma P_2$.

S využitím γ generujeme pro každého uživatele i, $1 \leq i \leq N$, dvojici (A_i, x_i), kde $x_i \in \mathbb{Z}_p^*$ vybereme nahodné a $A_i = 1/(\gamma + x_i)P_1 \in G_1$.

Skupinový veřejný klíč je gpk $= (P_1, P_2, Q, U, V, W)$. Soukromý klíč managera skupiny, který je schopný vystopovat podpis, je gmsk $= (\xi_1, \xi_2)$. Jednotlivé soukromé klíče uživatelů jsou páry gsk$[i] = (A_i, x_i)$.

Nikdo ze skupiny nezná γ, pouze důvěryhodná autorita, která na začátku vygenerovala W $(= \gamma P_2)$.

Ještě podotkněme, že Q zvolíme jako $Q = rP_1$ pro nějaké náhodné $r \in \mathbb{Z}_p^*$. $\xi_1, \xi_2 \in \mathbb{Z}_p^*$ zvolíme také náhodně a U, V vypočteme jako $U = r/\xi_1 P_1$, $V = r/\xi_2 P_1$.

sign(gpk, gsk$[i]$, M): S veřejným klíčem skupiny gpk $= (P_1, P_2, Q, U, V, W)$, soukromým klíčem uživatele gsk$[i] = (A_i, x_i)$ a zprávou $M \in \{0,1\}^*$ vytvoříme podpis následovně:

1. Zvolíme náhodně $\alpha, \beta, r_\alpha, r_\beta, r_x, r_{\delta_1}, r_{\delta_2} \in \mathbb{Z}_p^*$. Nastavíme $\delta_1 = x_i\alpha$, $\delta_2 = x_i\beta$ a vypočteme hodnoty $T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5$:

$$\begin{aligned} T_1 &= \alpha U, \quad T_2 = \beta V, \quad T_3 = A_i + (\alpha + \beta)Q, \quad R_1 = r_\alpha U, \quad R_2 = r_\beta V, \\ R_3 &= e(T_3, P_2)^{r_x} \cdot e(Q, P_2)^{-r_{\delta_1} - r_{\delta_2}}, \quad R_4 = r_x T_1 - r_{\delta_1} U, \quad R_5 = r_x T_2 - r_{\delta_2} V. \end{aligned}$$

2. Pomocí kryptograficky bezpečné hashovací funkce spočteme výzvu $c \in \mathbb{Z}_p^*$:

$$c = \text{hash}(M, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5).$$

3. S použitím c vypočteme hodnoty $s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2}$:

$$s_\alpha = r_\alpha + c\alpha, \quad s_\beta = r_\beta + c\beta, \quad s_x = r_x + cx, \quad s_{\delta_1} = r_{\delta_1} + c\delta_1, \quad s_{\delta_2} = r_{\delta_2} + c\delta_2.$$

4. Podpisem zprávy M je:

$$\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2}).$$

verify(gpk, M, σ): S veřejným klíčem skupiny gpk = (P_1, P_2, Q, U, V, W), zprávou M a podpisem $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$ ověříme:

1. Vypočítáme hodnoty $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4, \tilde{R}_5$:

$$\begin{aligned}\tilde{R}_1 &= s_\alpha U - cT_1, \quad \tilde{R}_2 = s_\beta V - cT_2, \quad \tilde{R}_4 = s_x T_1 - s_{\delta_1} U, \quad \tilde{R}_5 = s_x T_2 - s_{\delta_2} V,\\ \tilde{R}_3 &= e(T_3, P_2)^{s_x} \cdot e(Q, W)^{-s_\alpha - s_\beta} \cdot e(Q, P_2)^{-s_{\delta_1} - s_{\delta_2}} \cdot (e(T_3, W)/e(P_1, P_2))^c.\end{aligned}$$

2. Zkontroluje, zda: $c \stackrel{?}{=} \text{hash}(M, T_1, T_2, T_3, \tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4, \tilde{R}_5)$. Pokud ano, podpis přijmeme. Ne, odmítneme.

trasování(gmsk, gpk, M, σ): Na vstupu máme soukromý klíč managera skupiny gmsk = (ξ_1, ξ_2), veřejný klíč skupiny gpk = (P_1, P_2, Q, U, V, W), zprávu M a její podpis $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_{\delta_1}, s_{\delta_2})$. Manager skupiny postupuje následovně:

1. Ověří, že σ je platný podpis zprávy M.

2. Vypočte identitu uživatele, který vytvořil daný podpis, jako:

$$A = T_3 - (\xi_1 T_1 + \xi_2 T_2).$$

Má-li tedy manager skupiny seznam $\{A_i\}$ uživatelů, může v něm podpisovatele dopátrat.

Výkonnost: Uživatel i si může spočítat $e(P_1, P_2)$, $e(Q, P_2)$, $e(Q, W)$, $e(A_i, P_2)$ předem, nebude pak při podpisu potřebovat žádný výpočet párování, protože hodnotu $e(Q, P_2)$ má předpočítanou a $e(T_3, P_2)$ lze dopočítat z uloženého $e(A_i, P_2)$. Je však třeba provést deset umocňování. Pro ověření podpisu je třeba spočítat jedno párování a dvanáct umocňování.

Podpis má tři elementy G_1 a šest prvků $\mathbb{Z}_p^*$, tedy velikost podpisu je nezávislá na počtu členů skupiny N. V praxi lze použít např. 170-bitové prvočíslo p a grupu G_1 s elementy o 171 bitech. Pak bude mít podpis délku 1533 bitů, tedy 192 bytů, a srovnatelnou bezpečnost jako skupinově schéma založené na strong-RSA předpokladu s velikostí modulu 1024 bitů. Definici strong-RSA problému najdeme např. v [17].

4.2.7 Traitor tracing – Mitsunari, Sakai, Kasahara

Traitor tracing schéma z roku 2002 podle Mitsunari, Sakai, Kasahara (viz [26]) umožňuje vysílat pouze centru. Nejde tedy o veřejné schéma, kde by každý uživatel mohl vysílat ostatním.

Je založeno na problému k-CAA (případně ekvivalentním $(k-1)$-wDH, viz redukce na straně 14). Ve zmiňovaném článku představili oba problémy včetně

redukce mezi nimi. Neukázali však žádný důkaz bezpečnosti pro svůj návrh traitor tracing schématu.

Později ukázali Tô, Safavi-Naini, Zhang v článku [34], že návrh traitor tracing schématu od MSK není bezpečný. Dokázali totiž pomocí lineární kombinace soukromých klíčů uživatelů vytvořit další klíč, použitelný v pirátském dekoderu, ze kterého není dohledatelný nikdo z původní koalice traitorů. Tento útok je uveden v kapitole 5 na straně 60.

MSK traitor tracing schéma funguje následovně:

---

setup(p): Na základě bezpečnostního parametru p zvolí poskytoval grupu G. Poskytoval dále náhodně volí $a \in \mathbb{Z}_p$, $P \in G$. Pak distribuuje hodnoty $K_u = \frac{1}{u+a} P$, pro $u + a \neq 0$, registrovaným uživatelům $u \in \mathbb{Z}_p$. K_u je soukromý klíč uživatele u.

encrypt(M, a, P): Poskytoval volí náhodně $s \in \mathbb{Z}_p^*$ a $Q \in G$ různé od P. Zašifruje obsah vysílání M pomocí klíče relace s a tento klíč odesílá skrytý v hlavičce:

$$\mathrm{Hdr} = (s \cdot e(P, Q), Q, aQ).$$

decrypt(Hdr, K_u): Uživatel u z hlavičky $\mathrm{Hdr} = (A, B, C)$ vypočte:

$$e(K_u, uB + C) = e\left(\frac{1}{u+a} P, (u+a)Q\right) = e(P, Q)$$

a odtud už snadno dopočítá session key:

$$s = \frac{A}{e(P, Q)} = \frac{s \cdot e(P, Q)}{e(P, Q)},$$

který dále použije jako symetrický klíč k dešifrování vysílání.

Výkonnost: Hlavička obsahuje 3 elementy z G_T (rádově $3\log(p)$ bitů), tedy její velikost je nezávislá na počtu uživatelů. Dešifrování klíče relace s z hlavičky vyžaduje výpočet jednoho párování a jednoho dělení. Schéma však není bezpečné, jak ukázali Tô, Safavi-Naini, Zhang v článku [34].

4.2.8 Traitor tracing – Tô, Safavi-Naini, Zhang

Tô, Safavi-Naini, Zhang navrhli v [34] modifikaci MSK traitor tracing schématu, která je odolná vůči lineárnímu útoku, jímž rozbili původní MSK schéma. Tato nová modifikace dává veřejné schéma, takže šifrovat a odesílat data může

jakýkoliv uživatel. TSNZ také ukázali IND-CPA bezpečnost proti pasivnímu útočníkovi (pouze odposlouchává komunikaci, nezasahuje do ní) založenou na obtížnosi řešit DBDH problém (viz definice na straně 12). V závislosti na bezpečnostním parametru k je ze zabaveného pirátského dekoderu dopátratelná libovolná množina nejvýše k traitorů spolupracujících na vytvoření pirátského dekoderu.

Schéma funguje takto:

---

setup(p, k): Na základě bezpečnostního parametru p zvolí poskytoval grupu G. Dále náhodně volí polynom $f(x) = a_0 + a_1 x + \cdots + a_{2k-2} x^{2k-2} + x^{2k-1}$, a $P, Q \in G$. Nastaví $Q_i = a_i$, pro $0 \leq i \leq 2k-2$, a dále $g = e(P, Q) \in G_T$. Uživateli $u_i \in \mathbb{Z}_p^*$ pošle tajemství $K_u = \frac{1}{f(u)} P$.

Soukromý klíč poskytovalete je $(P, f(x))$ (slouží ke generování soukromých klíčů uživatelům), veřejný klíč je $\mathrm{PK} = (g, Q, Q_1, \ldots, Q_{2k-2})$.

Soukromý klíč uživatele $u \in \mathbb{Z}_p^*$ je $K^{(u)} = (K_u, uK_u, \ldots, u^{2k-1} K_u)$. Tedy uživatelům stačí si zapamatovat pouze hodnotu K_u, zbytek jejich soukromého klíče lze z K_u dopočitat.

encrypt(M, PK): Odesílatel volí náhodně $r \in \mathbb{Z}_p^*$, $s \in G_T$, zašifruje obsah vysílání M pomocí klíče relace s a tento klíč odesílá skrytý v hlavičce:

$$\mathrm{Hdr} = (sg^r, rQ, rQ_0, rQ_1, \ldots, rQ_{2k-2}) .$$

decrypt$(\mathrm{Hdr}, K^{(u)})$: Uživatel u ze svého soukromého klíče $K^{(u)}$ a z hlavičky $\mathrm{Hdr} = (A, B, C_0, \ldots, C_{2k+1})$ vypočte s následovně:

$$\frac{A}{e\left(K_u, \left(\sum_{i=0}^{2k-2} u^i C_i\right) + u^{2k-1} B\right)} = s.$$

který dále použije jako symetrický klíč k dešifrování vysílaných dat.

Výkonnost: Dešifrování klíče relace s z hlavičky zabere jedno párování a jedno dělení. V porovnání s traitor tracing schématem od Boneh, Franklin z roku 1999 (viz [9]) je toto nové schéma ve velikostech veřejného klíče i šifrového textu (=hlavičky) úspornější. Velikost hlavičky je $2k + 1$ elementů, tedy závislá na počtu uživatelů.

Trasovací algoritmus (uveden v [34]) je exponenciální v počtu traitorů k.

4.2.9 IBE schéma – Boneh, Franklin

Od představení IBE myšlenky v roce 1984 Adi Shamirem podali teprve až v roce 2001 Boneh a Franklin první praktické IBE schéma s důkazem bezpečnosti, založené

na BDH problému (viz [10]). Společně s nimi navrhl Cocks v článku [15] IBE schéma založené na práci s kvadratickými zbytky modulo velký modulus N. Obě schémata mají důkazy provedené v modelu náhodných orákul. Boneh a Franklin použili Fujisaki-Okamotovu transformaci, aby dosáhli IND-ID-CCA2 bezpečnosti.

IBE schéma se skládá z následujících 4 procedur:

---

setup(p): Na základě bezpečnostního parametru p PKG zvolí grupy G, G_T, a bilineární zobrazení $e : G \times G \to G_T$. Vezme náhodný generátor $P \in G$ a náhodné $x \in \mathbb{Z}_p^*$. Nastaví $Q = xP$. Zvolí ještě hashovací funkce $H_1 : \{0,1\}^* \to G$, $H_2 : G_T \to \{0,1\}^n$, $H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_p$, $H_4 : \{0,1\}^n \to \{0,1\}^n$, pro nějaké přirozené n. Předpokládejme, že budeme šifrovat zprávu M délky n.

Veřejné parametry params $= (P, Q, H_1, H_2, H_3, H_4)$, tajný master-key $= x$.

extract(ID): Pro identitu ID $\in \{0,1\}^*$ PKG procedurou extract vytvoří soukromý klíč d_{ID} příslušný identitě ID. Vypočte:

$$d_{\mathsf{ID}} = xH_1(\mathsf{ID}).$$

encrypt(params, ID, M): Zašifrujeme zprávu $M \in \{0,1\}^n$ pomocí veřejného klíče ID s využitím veřejných parametrů schématu. Vybereme náhodné $s \in \{0,1\}^n$, vypočteme $r = H_3(s, M)$ a nakonec vydáme šifrový text ($\oplus$ je binární sčítání):

$$C = (rP,\ s \oplus H_2(e(H_1(\mathsf{ID}), Q)^r),\ M \oplus H_4(s)).$$

decrypt(d_{ID}, C): Dešifrujeme šifrový text $C = (U, V, W)$ s využitím soukromého klíče d_{ID} takto:
1 - Vypočteme $V \oplus H_2(e(d_{\mathsf{ID}}, U)) = s$,
2 - vypočteme $W \oplus H_4(s) = M$,
3 - nastavíme $r = H_3(s, M)$ a ověříme, zda $U = rP$ (pokud ne, decrypt skončí),
4 - vydáme výsledný text M.

Výkonnost: Šifrování vyžaduje dvě umocňování, žádné párování, protože hodnotu $(e(H_1(\mathsf{ID}), Q)$ lze předpočítat, a výpočet všech čtyř hashovacích funkcí H_1, H_2, H_3, H_4. Dešifrování vyžaduje jedno párování a jednou výpočet hashovacích funkcí H_2, H_3, H_4. V ověřovacím kroku potřebujeme provést další jedno umocňování. Šifrový text obsahuje 3 prvky, řádově délky $(\log(p) + 2n)$ bitů.

Schéma je prokazatelně IND-ID-CCA2 bezpečné v modelu náhodných orákul.

4.2.10 HIBE schéma – Gentry, Silverberg

V roce 2002 navrhli Gentry a Silverberg první ℓ-HIBE schéma, které je bezpečné pro libovolnou hloubku ℓ identit v modelu náhodných orákul, je plně bezpečné proti koalicím útočníků a navíc je i efektivní. Jeho bezpečnost je založená

na BDH problému, stejně jako IBE schéma od Boneh a Franklina z roku 2001, ze kterého autoři uváděného HIBE schématu vycházeli. Pro zajištění IND-ID-CCA2 bezpečnosti použili Fujisaki-Okamotovu metodu využívající (kryptograficky bezpečné) hashovací funkce.

HIBE schéma se skládá z následujících 4 procedur:

setup(p): Na základě bezpečnostního parametru p PKG zvolí grupy G, G_T, a bilineární zobrazení $e : G \times G \to G_T$. Vezme náhodný generátor $P \in G_1$ a náhodné $x \in \mathbb{Z}_p^*$. Nastaví $Q = xP$. Zvolí ještě hashovací funkce $H_1 : \{0,1\}^* \to G$, $H_2 : G_T \to \{0,1\}^n$, $H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_p$, $H_4 : \{0,1\}^n \to \{0,1\}^n$, pro nějaké n. Předpokládejme, že budeme šifrovat zprávu M délky n. Pro $\leq i \leq t$ označme identitu v hloubce i jako $\mathsf{ID}|i = (\mathsf{ID}_1, \ldots, \mathsf{ID}_i)$.

Veřejné parametry jsou: $\mathsf{params} = (P, Q, H_1, H_2, H_3, H_4)$, tajný je $\mathsf{master\text{-}key} = x$.

Navíc si každá identita ID zvolí náhodně svůj tajný parametr $r_{\mathsf{ID}} \in \mathbb{Z}_p^*$ a uchová si ho.

extract(ID): Rodič identity ID, tj. identita o jeden stupeň výše v hierarchii, procedurou extract vytvoří soukromý klíč d_{ID} příslušný identitě $\mathsf{ID}|j = (\mathsf{ID}_1, \ldots, \mathsf{ID}_j)$. Vypočte postupně:
$A_j = H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_j) \in G_1$,
$B_{j-1} = r_{\mathsf{ID}|j-1} P$, a předá hodnoty $B_1, \ldots, B_{j-2}$, které zná, společně s novou hodnotou B_{j-1} identitě ID,
a vypočte soukromý klíč ($g_0 = 1$ v G):

$$d_{\mathsf{ID}} = d_{\mathsf{ID}|j-1} + r_{\mathsf{ID}|j-1} A_j = \sum_{i=1}^{j} r_{\mathsf{ID}|i-1} A_i .$$

encrypt(params, ID, M): Zašifrujeme zprávu M pomocí veřejného klíče ID s využitím veřejných parametrů schématu. Vybereme náhodné $s \in \{0,1\}^n$ a pro $1 \leq i \leq j$ vypočteme $A_i = H_1(\mathsf{ID}_1, \ldots, \mathsf{ID}_i)$ a ještě $r = H_3(s, M)$. Nakonec vydáme šifrový text:

$$C = (rP, rA_2, \ldots, rA_j,\ s \oplus H_2(e(Q, A_1)^r),\ M \oplus H_4(s)) .$$

decrypt(d_{ID}, C): Dešifrujeme šifrový text $C = (U_0, U_2, \ldots, U_j, V, W)$ s využitím soukromého klíče d_{ID} takto:
1 - Vypočteme $V \oplus H_2\left(\frac{e(y, d_{\mathsf{ID}})}{\prod_{i=2}^{j} e(b_{i-1}, U_i)}\right) = s$,
2 - vypočteme $W \oplus H_4(s) = M$,
3 - nastavíme $r = H_3(s, M)$ a ověříme, zda-li opravdu při zašifrování M pomocí r a $\mathsf{ID}|j$ obdržíme $(U_0, U_2, \ldots, U_j, V, \cdot)$ (pokud ne, decrypt skončí),
4 - vydáme výsledný text M.

Výkonnost: Šifrování vyžaduje $(j+1)$ umocňování, žádné párování, protože $e(Q, A_1)$ lze předpočítat, a výpočet všech čtyř hashovacích funkcí H_1, H_2, H_3, H_4, z toho H_1 budeme počítat j-krát. Hodnoty A_i však můžeme mít předpočítány, čímž ušetříme výpočty funkce H_1. Dešifrování vyžaduje j párování, jedno dělení a jednou výpočet hashovacích funkcí H_2, H_3, H_4. V ověřovacím kroku potřebujeme provést dalších $(j+1)$ umocňování. Obě procedury jsou tedy lineární vzhledem k hloubce identity $\mathsf{ID}|j$. Šifrový text obsahuje $(j+2)$ prvků, řádově délky $(j \cdot \log(p) + 2n)$ bitů.

Schéma je prokazatelně IND-ID-CCA2 bezpečné v modelu náhodných orákul.

4.2.11 IBE schéma – Boneh, Boyen

IBE schéma od Boneh, Boyen z roku 2004, (článek ([5])), je založeno na nové variantě Diffie-Hellmanova problému, ℓ-BDHI. Důkaz bezpečnosti schématu je postaven na rozhodovací variantě tohoto problému, tedy na ℓ-DBDHI (viz definice na straně 12).

IBE schéma se skládá z následujících 4 procedur:

$\mathsf{setup}(p)$: Na základě bezpečnostního parametru p PKG zvolí grupu G. Vezme náhodný generátor $P \in G$ a prvky $x, y \in \mathbb{Z}_p^*$. Nastaví hodnoty $X = xP$ a $Y = yP$. Veřejné parametry jsou $\mathsf{params} = (P, X, Y)$, tajný je $\mathsf{master\text{-}key} = (x, y)$.

$\mathsf{extract}(\mathsf{master\text{-}key}, \mathsf{ID})$: PKG pomocí extract vytvoří soukromý klíč d_{ID} příslušný identitě ID. Zvolí náhodné $r \in \mathbb{Z}_p$ a spočte:

$$d_{\mathsf{ID}} = \frac{1}{ID + x + ry} P \in G.$$

Soukromý klíč identity ID je dvojice (r, d_{ID}).

$\mathsf{encrypt}(\mathsf{params}, \mathsf{ID}, M)$: Zašifrujeme zprávu $M \in G_T$ pomocí veřejného klíče ID s využitím veřejných parametrů schématu. Vybereme náhodné $s \in \mathbb{Z}_p^*$ a vydáme šifrový text:

$$C = (s\mathsf{ID}P, sY, M \cdot e(P, P)^s).$$

$\mathsf{decrypt}(r, d_{ID}, C)$: Dešifrujeme šifrový text $C = (A_1, A_2, A_3)$ s využitím soukromého klíče (r, d_{ID}) takto:

$$\frac{A_3}{e(A_1 \cdot A_2^r,\ K)} = \frac{M \cdot e(P, P)^s}{e\left(s\mathsf{ID}P + s(xP) + r(sY),\ \frac{1}{ID+x+ry}\right) P} = M.$$

Výkonnost: Šifrování zprávy zabere pouze dvě umocňování, neboť hodnotu párování $e(P, P)$ lze předpočítat. Dešifrování zabere výpočet jednoho párování, umocnění a dělení. Šifrový text se skládá ze dvou prvků G a jednoho prvku G_T, tedy rádově $3\log(p)$ bitů. Tři prvky grupy má také veřejný klíč. Schéma je prokazatelně IND-sID-CPA2 bezpečné ve standardním modelu. Pomocí konstrukce uvedené v článku [13] lze získat IBE schéma, které je prokazatelně IND-sID-CCA2 bezpečné ve standardním modelu, avšak konstrukce se opírá o schéma neinteraktivního důkazu s nulovou znalostí, což je dosud velmi neefektivní kryptografický primitiv.

Zajímavou vlastností schématu je randomizace generování soukromých klíčů. Tedy jedna identita může vlastnit více soukromých klíčů, a každý jednotlivý klíč postačuje k dešifrování zpráv. Je otázkou, jak toho využít v praxi.

4.2.12 HIBE schéma – Boneh, Boyen

HIBE schéma z roku 2004 podle Boneh a Boyen (viz [5]) je prokazatelně IND-sID-CPA2 bezpečné ve standardním modelu. Bezpečnost je založena na DBDH problému. Pomocí konstrukce uvedené v článku [14] lze z ℓ-HIBE získat $(\ell - 1)$-HIBE schéma (o jeden stupeň „chudší"), které je prokazatelně IND-sID-CCA2 bezpečné ve standardním modelu, za cenu prodloužení každého šifrového textu o jednorázový podpis a odpovídající ověřovací klíč a několika extra výpočtů spojených s vytvořením a ověřením podpisu.

Předpokládejme, že veřejné klíče ID hloubky ℓ jsou prvky $\mathbb{Z}_p^\ell$. Označíme ID$=(I_1, \ldots, I_\ell) \in \mathbb{Z}_p^\ell$. Prvních j komponent odpovídá identitě v hloubce j.

Čtyři algoritmy definující HIBE schéma jsou následující:

setup(p, ℓ): Ze vstupu vygeneruje PKG grupu G a parametry pro HIBE schéma s maximální možnou hloubkou ℓ. Náhodně zvolí generátor $P \in G$, $\alpha \in \mathbb{Z}_p^*$ a nastaví $P_1 = \alpha P$. Dále vybere náhodné $Q_1, \ldots, Q_\ell \in G$ a náhodné $P_2 \in G$. Veřejné parametry params a tajný master-key jsou:

$$\mathsf{params} = (P, P_1, P_2, Q_1, \ldots, Q_\ell), \quad \mathsf{master\text{-}key} = \alpha P_2.$$

extract(ID): PKG vygeneruje soukromý klíč d_{ID} pro identitu ID $= (I_1, \ldots, I_j)$ s využitím master-key. Zvolí náhodně $r_1, \ldots, r_j \in \mathbb{Z}_p^*$ a vypočte $(j+1)$-tici:

$$d_{\mathsf{ID}} = \left(\alpha P_2 + \sum_{k=1}^{j} r_k (I_k P_1 + Q_k),\ r_1 P, \ldots,\ r_j P \right).$$

Je možné také vypočítat d_{ID_j} jen ze znalosti $d_{\mathsf{ID}_{j-1}}$, jak je požadováno v definici HIBE. Stačí zvolit náhodné $r_j \in \mathbb{Z}_p^*$ a pro $d_{\mathsf{ID}_{j-1}} = (d_0, \ldots, d_{j-1})$ vypočítat:

$$d_{\mathsf{ID}} = (d_0 + r_j(I_j P_1 + Q_j),\ d_1, \ldots, d_{j-1},\ r_j P)\,.$$

encrypt(params, ID, M): Zašifrujeme zprávu $M \in G_T$ pomocí veřejného klíče $\mathsf{ID} = (I_1, \ldots, I_j)$. Vybereme náhodně $s \in \mathbb{Z}_p^*$ a vypočteme:

$$C = (e(P_1, P_2)^s \cdot M,\ sP,\ s(I_1 P_1 + Q_1), \ldots, s(I_j P_1 + Q_j))\,.$$

decrypt(d_{ID}, C): Dešifrujeme zprávu $C = (A, B, C_1, \ldots, C_j)$ pro identitu $\mathsf{ID} = (I_1, \ldots, I_j)$ s využitím $d_{\mathsf{ID}} = (d_0, \ldots, d_j)$ následujícím výpočem:

$$A \cdot \frac{\prod_{k=1}^{j} e(C_j, d_j)}{e(B, d_0)} = \frac{e(P_1, P_2)^s \cdot M \cdot \prod_{k=1}^{j} e\,(I_k P_1 + Q_k,\ P)^{sr_k}}{e(P, P_2)^{s\alpha} \cdot \prod_{k=1}^{j} e\,(P,\ I_k P_1 + Q_k)^{sr_k}} = M.$$

.....

Výkonnost: Šifrování zprávy zabere $(j+2)$ umocňování. Všimněmě si, že $e(P_1, P_2)$ můžeme předpočítat, takže encrypt nebude k výpočtu vyžadovat žádné párování. Navíc $e(P_1, P_2)$ můžeme dát mezi parametry schématu params namísto hodnoty P_2, kterou už pak při výpočtech nebudeme potřebovat. Dešifrování zabere $(j+1)$ párování. Tedy obě procedury mají časovou složitost řádově $O(\ell)$.
Veřejný klíč se skládá z $(\ell+3)$ prvků G. Šifrový text z $(j+1)$ prvků G a 1 prvku G_T. Tedy paměťová náročnost schématu je lineární v hloubce hierarchie HIBE, $O(\ell)$.

V [5] je v sekci 4.2 podán důkaz, že předchozí (ℓ)-HIBE schéma je IND-sID-CPA2 bezpečné, pro libovolnou hloubku ℓ, pokud je v grupě G DBDH problém těžký.
V [14] je popsán postup, jak efektivně z IND-sID-CPA2 bezpečného ℓ-HIBE vytvořit IND-sID-CCA2 bezpečné $(\ell - 1)$-HIBE schéma. Tato konstrukce nám tedy dává HIBE schémata s libovolnou hloubku identit ℓ, navíc bez použití náhodných orákul, vše za cenu prodloužení každého šifrového textu o jednorázový podpis a odpovídající ověřovací klíč a několika extra výpočtů spojených s vytvořením a ověřením podpisu.

4.2.13 IBE schéma – Waters

Boneh a Boyen v [6] navrhli IBE schéma, které je jako první IBE prokazatelně IND-ID-CCA bezpečné ve standardním modelu. Tím vyřešili otevřený problém navržený pány Boneh a Franklin v [10]. Bohužel nové schéma je velmi nepraktické v porovnání s ostatními efektivními schématy. Je náročné na výpočet šifrovací a zejména dešifrovací procedury, navíc s dlouhým veřejným i soukromým klíčem.

V roce 2005 Waters přišel se svým návrhem, kdy modifikoval výše uvedené l-HIBE schéma od Boneh, Boyen s IND-sID-CPA2 bezpečností (představené v [5]) a získal efektivní IBE schéma s prokazatelnou IND-ID-CPA2 bezpečností. Bezpečnost je založena na DBDH problému, bez použití náhodných orákul.

Dále navrhl hybridní 2-HIBE schéma, kde jako první stupeň použil svůj návrh IBE schématu a jako druhý stupeň použil konstrukci HIBE schématu od Boneh a Boyen (viz strana 48). S použitím konstrukce v [14] takto získal efektivní plně bezpečné (IND-ID-CCA2) IBE schéma bez použití náhodných orákul.

Čtyři IBE procedury podle Waterse jsou následující.

.....

setup(p, n): Na základě bezpečnostního parametru p PKG zvolí grupu G a vygeneruje parametry schématu. Zvolí náhodně $\alpha \in \mathbb{Z}_p^*$, generátor $P \in G$, nastaví $P_1 = \alpha P$, zvolí náhodné $P_2 \in G$ a vypočte tajné αP_2. Dále zvolí náhodné $U' \in G$ a vektor náhodných hodnot $V = (U_1, \ldots, U_n) \in G^n$. Veřejné parametry params a tajný master-key jsou:

$$\mathsf{params} = (P, P_1, P_2, U', V), \quad \mathsf{master\text{-}key} = \alpha P_2.$$

extract(ID, master-key): PKG vygeneruje soukromý klíč pro indentitu ID, kde ID je maximálně délky n bitů. Označme $\mathsf{ID} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_n)$ n-bitový řetězec reprezentující danou identitu. PKG zvolí náhodně $r \in \mathbb{Z}_p^*$ a vypočte soukromý klíč příslušný identitě ID:

$$d_{\mathsf{ID}} = \left(\alpha P_2 + r(U' + \sum_{\mathsf{ID}_i = 1} U_i), rP \right).$$

encrypt(ID, params, M): Pro zprávu $M \in G_T$ a identitu $\mathsf{ID} = (\mathsf{ID}_1, \ldots, \mathsf{ID}_n)$ zvolíme náhodné $t \in \mathbb{Z}_p^*$ a vypočteme šifrový text:

$$C = \left(e(P_1, P_2)^t \cdot M,\ tP,\ t(U' + \sum_{\mathsf{ID}_i = 1} U_i) \right).$$

decrypt(d_{ID}, C): Pro šifrový text $C = (C_1, C_2, C_3)$ a identitu ID se soukromým klíčem $d_{\mathsf{ID}} = (d_1, d_2)$ dešifruje následovně:

$$\begin{aligned} C_1 \frac{e(d_2, C_3)}{e(d_1, C_2)} &= e(P_1, P_2)^t \cdot M \cdot \frac{e\left(rP,\ t(U' + \sum_{\mathsf{ID}_i=1} U_i)\right)}{e\left(\alpha P_2 + r(U' + \sum_{\mathsf{ID}_i=1} U_i),\ tP\right)} = \\ &= e(P, P_2)^{\alpha t} \cdot M \cdot \frac{e\left(P,\ U' + \sum_{\mathsf{ID}_i=1} U_i\right)^{rt}}{e(P_2, P)^{\alpha t} \cdot e\left(U' + \sum_{\mathsf{ID}_i=1} U_i,\ P\right)^{rt}} = M. \end{aligned}$$

.....

Výkonnost: Hodnota $e(P_1, P_2)$ může být předpočítána, a tudíž pro šifrovaní potřebujeme provést v průměru $n/2$ grupových operací v G, a dále tři umocňování. Pro dešifrování potřebujeme spočítat dvě párování a jedno (rychlejší) dělení. Veřejný klíč má $(n + 4)$ prvky grupy. Šifrový text má tři prvky grupy. Schéma je IND-ID-CPA2 bezpečné ve standardním modelu, a po modifikaci navržené Watersem z něj lze vytvořit prokazatelně IND-ID-CCA2 bezpečné, efektivní IBE schéma. Opět za cenu prodloužení každého šifrového textu o jednorázový podpis a odpovídající ověřovací klíč a několika extra výpočtů spojených s vytvořením a ověřením podpisu.

Waters ukazuje i na možnost HIBE schématu postaveného na základě svého návrhu IBE schématu. Ovšem v tomto HIBE schématu dochází k nárustu veřejných parametrů lineárně s hloubkou ℓ a také k nepříznivé (exponenciální v ℓ) redukci na bezpečnosti. Pořád však jde o lepší HIBE schéma než je návrh od Boneh a Boyen v [6]. Přesto zůstává otevřeným problémem zkonstruovat efektivní HIBE schéma plně bezpečné ve standardním modelu.

4.2.14 HIBE schéma – Boneh, Boyen, Goh

Boneh, Boyen a Goh v roce 2005 navrhli v článku [7] nové HIBE schéma, které má velmi praktické parametry. Šifrový text má velikost tří prvků grupy, nezávisle na hloubce identity, která šifrový text vytvoří. K dešifrování pak potřebujeme vypočítat jenom dvě párování, protože šifrový text má konstantní velikost. V ostatních parametrech je lineární vzhledem k hloubce hierarchie, stejně jako starší schémata (jako např. HIBE v [5]).

Schéma je prokazatelně IND-sID-CPA2 bezpečné ve standardním modelu a tedy pomocí konstrukce uvedené v [14] můžeme z ℓ-HIBE schématu získat $(\ell - 1)$-HIBE schéma, které je IND-sID-CCA2 bezpečné.

Předpokládejme, že veřejné klíče ID v hloubce ℓ jsou prvky $\mathbb{Z}_p^\ell$. Označíme $\mathsf{ID} = (I_1, \ldots, I_\ell)$. Prvních j komponent odpovídá identitě v hloubce j. Předpokládejme navíc, že zprávy, které chceme zašifrovat, jsou prvky G_T.

Schéma je následující:

.....

$\mathsf{setup}(\ell, p)$: Na základě bezpečnostního parametru p PKG zvolí grupu G a vygeneruje parametry pro HIBE schéma s maximální možnou hloubkou ℓ. Náhodně zvolí generátor $P \in G, \alpha \in \mathbb{Z}_p^\star$ a nastaví $P_1 = \alpha P$. Dále vybere náhodně $P_2, P_3, Q_1, \ldots, Q_\ell \in G$. Veřejné parametry params a tajný $\mathsf{master\text{-}key}$ jsou dány takto:

$$\mathsf{params} = (P, P_1, P_2, P_3, Q_1, \ldots, Q_\ell), \quad \mathsf{master\text{-}key} = \alpha P_2$$

extract(ID): PKG vygeneruje soukromý klíč d_{ID} pro identitu $\mathsf{ID} = (I_1, \ldots, I_j)$ s využitím master-key. Zvolí náhodně $r \in \mathbb{Z}_p^*$ a vypočte $(\ell - j + 2)$-tici:

$$d_{\mathsf{ID}} = (\alpha P_2 + r(I_1 Q_1 + \cdots + I_j Q_j + g_3),\ rP,\ rQ_{j+1}, \ldots, rQ_\ell)\,.$$

Všimněme si, že d_{ID} se zkracuje s rostoucí hloubkou identity ID.

Je také možné spočítat d_{ID} jen ze znalosti $d_{\mathsf{ID}|j-1}$, jak bylo požadováno v definici HIBE. Označme si $d_{\mathsf{ID}|j-1} = (A_0, A_1, B_j, \ldots, B_\ell)$. Stačí zvolit náhodné $t \in \mathbb{Z}_p^*$ a vypočítat:

$$d_{\mathsf{ID}} = \left(A_0 + I_j B_j + t(I_1 Q_1 + \cdots + I_j Q_j + P_3)',\ A_1 + tP,\ B_{j+1} + tQ_{j+1}, \ldots, B_\ell + tQ_\ell\right).$$

encrypt(params, ID, M): Zašifruje zprávu $M \in G_T$ pomocí veřejného klíče $\mathsf{ID} = (I_1, \ldots, I_j)$. Vybere náhodně $s \in \mathbb{Z}_p^*$ a vypočte:

$$C = (e(P_1, P_2)^s \cdot M,\ sP,\ s(I_1 Q_1 + \cdots + I_j Q_j + P_3))\,.$$

decrypt(d_{ID}, C): Dešifruje zprávu $C = (A, B, C)$ pro identitu $\mathsf{ID} = (I_1, \ldots, I_j)$ s užitím odpovídajícího klíče $d_{\mathsf{ID}} = (A_0, A_1, B_{j+1}, \ldots, B_\ell)$ následujícím výpočtem:

$$A \cdot \frac{e(A_1, C)}{e(B, A_0)} = M.$$

Ověření:

$$\begin{aligned}\frac{A \cdot e(A_1, C)}{e(B, A_0)} &= \frac{e(P_1, P_2)^s M \cdot e\left(rP,\ s(I_1 Q_1 + \cdots + I_j Q_j + P_3)^s\right)}{e\left(sP,\ \alpha P_2 + r(I_1 Q_1 + \cdots + I_j Q_j + P_3)\right)} \\ &= \frac{e(P_1, P_2)^s M}{e(P, P_2)^{s\alpha}} = M.\end{aligned}$$

Výkonnost: Šifrování nepotřebuje žádný výpočet párování, protože hodnotu $e(P_1, P_2)$ lze předpočítat. Navíc $e(P_1, P_2)$ můžeme dát mezi parametry params namísto hodnoty P_2, kterou už dále při výpočtech nebudeme potřebovat. Dešifrování potřebuje provést dvě párování, tedy je z hlediska časové náročnosti je nezávislé na hloubce hierarchie. Soukromý klíč identity v hloubce j je roven $(\ell - j + 2)$-tici prvků G, z nichž pouze první dva prvky jsou potřeba k dešifrování, zbytek slouží ke generování soukromých klíčů identitám ve větší hloubce. Veřejný klíč má velikost $(\ell + 4)$ prvků G, tedy je lineární v hloubce celého schématu. Šifrový text je trojice prvků, má tedy konstantní velikost.

Schéma je IND-sID-CPA2 bezpečné ve standardním modelu. Bezpečnost je založena na obtížnosti řešit ℓ-BDHI problém. Pomocí konstrukce uvedené v [14] dostaneme IND-sID-CCA2 bezpečné schéma bez použití náhodných orákul, za cenu

prodloužení každého šifrového textu o jednorázový podpis a odpovídající ověřovací klíč a několika extra výpočtů spojených s vytvořením a ověřením podpisu.

Uvedené HIBE schéma bylo v článku [7] postaveno původně na problému ℓ-BDHE (viz definice na straně 12), ale pak se ukázalo, že lze založit upravené schéma na obtížnějším problému ℓ-BDHI. Proto autoři Boneh, Boyen a Goh schéma předělali a publikovali updatovaný článek. Problém ℓ-BDHE však nebyl zapomenut a použili ho Boneh, Boyen a Gentry téhož roku (2005) při konstrukci schématu broadcastového šifrování.

4.2.15 Schéma broadcastového šifrování – Boneh, Gentry, Waters

Boneh, Gentry, Waters navrhli v roce 2005 v článku [11] schéma broadcastového šifrování, které je odolné vůči koalicím („collusion resistant"), viz definice na straně 32. Navíc je IND-CCA2 bezpečné proti útočníkovi, který si hned na začátku pevně zvolí množinu S uživatelů, kterou hodlá napadnout. Navrhli sémanticky bezpečné schéma, které dále modifikovali, aby dostali IND-CCA2 bezpečné schéma s konstatní velikostí odesílaného šifrového textu a jednotlivých soukromých klíčů a s veřejným klíčem lineárně dlouhým v závislosti na počtu uživatelů N. Schéma je postaveno na $(N+2)$-BDHE problému a využívá CMA2 bezpečné podpisové schéma s funkcemi (keygen, sign, verify).

Schéma je následující:

setup(p, N): Na základě bezpečnostního parametru p důvěryhodná autorita zvolí grupu G a nastaví veřejné parametry schématu tak, aby schéma mohlo používat až N uživatelů. Vybere náhodně generátor $P \in G$ a náhodné $\alpha, \gamma \in \mathbb{Z}_p^*$. Vypočte $P_i = \alpha^i P$, pro $i = 1, \ldots, 2(N+1)$ a nastaví $V = \gamma P$. Veřejný klíč PK a soukromý klíč d_i uživatele $i \in \{1, \ldots, N\}$ jsou:

$$\mathrm{PK} = (P, P_1, \ldots, P_{N+1}, P_{N+3}, \ldots, P_{2(N+2)}, V), \quad d_i = \gamma P_i = (\alpha^i)V.$$

encrypt(S, PK): Spustíme keygen, abychom dostali klíče K_{sig}, V_{sig}. Předpokládejme, že $V_{\mathrm{sig}} \in \mathbb{Z}_p^*$, jinak bychom museli nejdřív použít nějakou hashovací funkci. Vybereme náhodné $t \in \mathbb{Z}_p^*$ a spočteme $K = e(P_{N+2}, P)^t = e(P_N, P_2)^t \in G_T$. Vypočteme:

$$C = \left(tP,\ t(V + V_{\mathrm{sig}} P_1 + \sum_{j \in S} P_{N+2-j}) \right), \quad \mathrm{Hdr} = (C, \mathsf{sign}(C, K_{\mathrm{sig}}), V_{\mathrm{sig}})$$

a vydáme dvojici (Hdr, K).

decrypt$(S, i, d_i, \mathrm{Hdr}, \mathrm{PK})$: Pro $\mathrm{Hdr} = (\tilde{C}_0, \tilde{C}_1, \tilde{\sigma}, \tilde{V}_{\mathrm{sig}})$ provedeme postupně:

1 - pomocí verify a klíče $\tilde{V}_{\mathrm{sig}}$ ověříme, zda $\tilde{\sigma}$ je platný podpis zprávy $(\tilde{C}_0, \tilde{C}_1)$ podepsané klíčem K_{sig}.

2 - zvolíme náhodné $w \in \mathbb{Z}_p^*$ a vypočteme:

$$\tilde{d}_0 = \left(d_i + \tilde{V}_{\mathrm{sig}} P_{i+1} + \sum_{\substack{j \in S \\ j \neq i}} P_{N+2-j+i} \right) + w \left(V + \tilde{V}_{\mathrm{sig}} P_1 + \sum_{j \in S} P_{N+2-j} \right) \quad \tilde{d}_1 = P_1 + wP.$$

3 - Vydáme:

$$K = \frac{e\left(\tilde{d}_1, C_1\right)}{e\left(\tilde{d}_0, C_0\right)}.$$

Ověřit, že dešifrování funguje správně, je jednoduchá manipulace s výrazem. Ukáže se, že takto získáme $K = e(P_{N+1}, P)^t$.

.....

Výkonnost: Soukromý klíč je pouze jeden prvek G, šifrový text Hdr je dvojice prvků G společně s podpisem a ověřovacím klíčem ze $\mathbb{Z}_p^*$, veřejný klíč má velikost $(2N + 1)$ prvků G, tedy lineární ve velikosti skupiny uživatelů N. Šifrování nepotřebuje žádný výpočet párování, protože hodnotu $e(P_{N+2}, P)$ lze předpočítat, dešifrování potřebuje spočítat dvě párování.

Navíc lze provést optimalizaci pro práci s relativně velkými množinami S, kdy $|S| = N - r$, pro $r \ll N$. Stačí, aby si každý uživatel ke svému soukromému klíči d_i předpočítal a zapamatoval také hodnotu

$$\prod_{j=1,\, j \neq i}^{N} P_{N+2-j+i},$$

což mu umožní dešifrovat s výpočtem jenom r grupových operací.
Podobná optimalizace lze aplikovat i pro proceduru encrypt.

Ve stejném článku navrhli autoři i modifikaci, kdy máme možnost zvolit si trade-off mezi velikostí šifrového textu a veřejného klíče. Takto lze získat schéma s konstatní velikostí soukromého klíče a s velikostmi veřejného klíče a šifrového textu v řádu $\mathrm{O}(\sqrt{N})$.

4.3 Třístranná Diffie-Hellmanova dohoda na klíči

V článku [25] uvádí Joux zobecnění klasické D-H dohody na klíči pro bilineární grupy. Takto dosáhneme dohody na klíči mezi třemi stranami vyžadující pouze

jedno kolo komunikace. Schéma je založeno na obtížnosti BDH problému.

Chceme, aby Alice, Bob a Carl pouze jednou odeslali hodnoty přes veřejný vysílací kanál a aby každý z nich dokázal z přijatých zpráv od zbylých dvou spočítat stejné (sdílené) tajemství. Přitom není potřeba, aby byli všichni ve stejnou dobu online. Jde nám o neinteraktivní schéma.

Schéma je následující:

---

setup: A,B,C zvolí pracovní grupu G a dohodnou se na veřejných hodnotách $P, Q \in G$, tž. $e(P,Q) \neq 1$.

keygen: A (resp. B,C) zvolí náhodně své tajemství a (resp. b, c), vypočte veřejnou hodnotu $(P_A, Q_A) = (aP, aQ)$ (resp. $(P_B, Q_B) = (bP, bQ)$, $(P_C, Q_C) = (cP, cQ)$) a odešle tuto dvojici ostatním dvěma účastníkům.

A,B,C pak vypočtou společné tajemství $K = e(P,Q)^{abc}$:

$$\begin{aligned} \text{A} &: e(P_B, Q_C)^a = e(P_C, Q_B)^a = e(P,Q)^{abc}, \\ \text{B} &: e(P_A, Q_C)^a = e(P_C, Q_A)^a = e(P,Q)^{abc}, \\ \text{C} &: e(P_A, Q_B)^a = e(P_B, Q_A)^a = e(P,Q)^{abc}. \end{aligned}$$

Tento kryptografický primitiv, stejně jako originální schéma D-H dohody na klíči, není odolné vůči man-in-the-middle útoku. Útočník, který má přístup ke komunikačnímu kanálu, může založit tři různé komunikace s jednotlivými účastníky A,B,C a v každé z těchto komunikací se vydávat za zbylé dva účastníky. Takto se dohodne se všemi třemi na třech různých tajemstvých a může odposlouchávat i měnit následující šifrovanou komunikaci, aniž by kdokoliv z A,B,C tušil, že je podváděn.

Kapitola 5

Útoky

5.1 Útoky proti tajemství α

Označme N velikost pracovní grupy. N není nutně prvočíslo. Budeme probírat algoritmy a útoky na tajemství α, např: v DLP, kde ze znalosti g, g^α chceme vypočítat α. V této kapitole budeme používat multiplikativní notaci, která je více zažitá pro práci s algoritmy pro řešení DLP. Budeme značit $g \in G$ generátor multiplikativni grupy G.

5.1.1 Model generických grup

V článku [32] představil Shoup model generických grup, pomocí něhož je možno heuristicky dokázat obtížnosti různých variant Diffie-Hellmanova problému. Model pracuje s výpočetní představou, že na začátku výpočtu nemáme představu o struktuře pracovních grup a všechny grupové operace, izomorfismus i párování za nás provádí pět orákul.

Předpokládejme, že máme kódování ξ pro každou pracovní grupu, které pro generátor g dané grupy G a číslo $a \in \mathbb{Z}_p$ kóduje prvek $g^a \in G$ binárním řetízkem, tedy $\xi(g^a) \in \{0,1\}^*$. Při výpočtech komunikujeme s orákuly prostřednictvím těchto representací prvků grup. Stejně tak je omezen i případný útočník. Algoritmus $\mathcal{A}$ pracující za těchto podmínek nazveme generický. Podotkněme, že náročnost $\mathcal{A}$ závisí na velikosti grup a na „úspornosti“ kódování.

Např. Pohlig-Hellmanův algoritmus je generický. Stejně tak Pollard-ρ algoritmus pro řešení DLP. Naopak algoritmus Index-calculus generický není.

Shoup v [32] dokázal:

1 - Pokud řešíme DLP v grupě řádu N, kde p je největší prvočíselný dělitel N, generickým algoritmem $\mathcal{A}$, který provede nejvýše m dotazů na orákulum, pak pravděpodobnost, že výstupem $\mathcal{A}(g, g^a)$ bude tajemství a, je řádově $O\left(\frac{m^2}{p}\right)$.

2 - Pokud řešíme CDH v grupě řádu N, kde p je největší prvočíselný dělitel N, generickým algoritmem $\mathcal{A}$, který provede nejvýše m dotazů na orákulum, pak pravděpodobnost, že výstupem $\mathcal{A}(g, g^a, g^b)$ bude hodnota g^{ab}, je řádově $O\left(\frac{m^2}{p}\right)$.

3 - Pokud řešíme DDH v grupě řádu N, kde p je největší prvočíselný dělitel N, generickým algoritmem $\mathcal{A}$, který provede nejvýše m dotazů na orákulum, pak pravděpodobnost, že $\mathcal{A}$ rozpozná mezi vstupem (g, g^a, g^b, g^c), kde $ab = c$, a náhodným vstupem $(g^{r_1}, g^{r_2}, g^{r_3}, g^{r_4})$, pro $r_i \in \mathbb{Z}_p$ náhodná, je řádově $O\left(\frac{1}{2} + \frac{m^2}{p}\right)$.

Boneh a Boyen v [4] dokázali:

4 - Pokud řešíme ℓ-SDH v grupách G_1, G_2, G_T řádu p, generickým algoritmem $\mathcal{A}$, který provede nejvýše m dotazů na orákula, pak pravděpodobnost, že výstupem $\mathcal{A}(g_1, g_2, g_2^{\alpha}, \ldots, g_2^{\alpha^{\ell}})$ bude dvojice $(c, g_1^{\frac{1}{\alpha+c}})$, je řádově $O\left(\frac{m^2\ell+\ell^3}{p}\right)$.

Důsledek: Pokud řešíme ℓ-SDH v bilineárních grupách (G_1, G_2) řádu p, generickým algoritmem $\mathcal{A}$, s nějakou konstantní nenulovou výhodou $\epsilon > 0$, a navíc $\ell < o(\sqrt[3]{p})$, pak musíme provést $\Omega\left(\sqrt{\epsilon p/\ell}\right)$ kroků výpočtu, včetně dotazů na orákula.

Boneh, Boyen a Goh v [7] dokázali:

5 - Pokud řešíme BDH v grupě G řádu p, generickým algoritmem $\mathcal{A}$, s nějakou konstantní nenulovou výhodou $\epsilon > 0$, musíme provést $\Omega\left(\sqrt{\epsilon p/3}\right)$ kroků výpočtu, včetně dotazů na orákula.

6 - Pokud řešíme ℓ-BDHI v grupě G řádu p, generickým algoritmem $\mathcal{A}$, s nějakou konstantní nenulovou výhodou $\epsilon > 0$, musíme provést $\Omega\left(\sqrt{\epsilon p/2\ell}\right)$ kroků výpočtu, včetně dotazů na orákula.

7 - Pokud řešíme ℓ-BDHE v bilineárních grupách G_1, G_2 řádu p, generickým algoritmem $\mathcal{A}$, s nějakou konstantní nenulovou výhodou $\epsilon > 0$, musíme provést $\Omega\left(\sqrt{\epsilon p/4\ell}\right)$ kroků výpočtu, včetně dotazů na orákula.

5.1.2 Generické útoky

Pro více informaci o generických útocích odkazujeme na knihu o aplikované kryptografii [28]. Algoritmus index calculus pochází ze 20-tých let 19. století. Od té doby se jím zabývalo hodně matematiků. Index calculus má subexponenciální

časovou složitost na multiplikatvních grupách konečných těles, avšak na eliptických křivkách běží pomalu a dosud není známá nejaká rychle fungující varianta.

.....

Hrubá síla: Při použití hrubé síly k získání neznámého exponentu (jenom z hodnot g, g^α) je třeba generovat posloupnost prvků $g^0, g^1, g^2, g^3, \ldots$ dokud nenarazíme na zadaný prvek g^α. Tento jednoduchý algoritmus má však časovou složitost $O(N)$, paměťovou $O(1)$, a tudíž není prakticky v kryptoanzlýze použitelný.

.....

Baby-step giant-step: Jde o time-memory trade-off úpravu algoritmu hrubé síly.

Vezněme číslo $j = \sqrt{N}$. Využijeme následujícího pozorování: Ať $g^\alpha = h$ a $\left\lceil k = \sqrt{N} \right\rceil$. Pak existují čísla $0 \geq i, j \geq k$, tak že $\alpha = im + j$. Potom máme $h = g^\alpha = g^{im+j}$ a odtud $hg^{-im} = g^j$.

Z pozorovámí plyne algoritmus: Nejprve spočítáme a uložíme si tabulku dvojic (j, g^j) pro $0 \geq j \geq k$ (baby-step), a tuto si uspořádáme podle druhé hodnoty. Následně počítáme hodnoty hg^{-im} pro $0 \geq i \geq k$ (giant-step) a v každém kroku se podíváme de naší tabulky, jestli se v ní aktuální hodnota hg^{-im} nevyskytuje. Pozorování nás ujišťuje o tom, že alespoň jednou shodu v tabulce najdeme.

V tomto případě, kdy $hg^{-im} = g^j$ neboli $h = g^\alpha = g^{j+im}$, dopočítáme snadno tajný exponent $\alpha = j + im$.

Algoritmus má paměťové nároky řádu $O(\sqrt{N})$, k vytvoření tabulky navíc potřebuje $O(\sqrt{N})$ násobení v grupě a $O(\sqrt{N} \cdot \lg N)$ porovnání při uspořádávání tabulky. Pak procházíme řádově $O(\sqrt{N})$ „velkých-kroků" a vždy vyhledáváme v tabulce. Uvážíme-li ale, že násobení v grupě je řádově složitější než vyhledávání hodnot v tabulce (jedno vyhledání odpovídá $\lg \sqrt{N}$ porovnáním), můžeme uzavřít odhad časové složitosti algoritmu jako $O(\sqrt{N})$.

.....

Pollard-ρ: Idea algoritmu stojí na pozorování, že když pomocí dostatečně náhodných funkcí vybereme posloupnosti prvků z G, kde $|G| = p$, pro p prvočíslo, pak smyčky v těchto posloupnosti mají průměrnou délku $O(\sqrt{p})$. Stačí nám tedy chytře zvolit funkci a pak najít dva prvky $x_i = x_j$, $i \neq j$, vygenerované posloupnosti.

Zvolme $x_0 = 1$, a dále $x_{i+1} \stackrel{def}{=} f(x_i) = \begin{cases} h \cdot x_i \\ x_i^2 \\ g \cdot x_i \end{cases}$, náhodně vybraná možnost.

Pak $x_i = g^{a_i} \cdot h^{b_i}$, pro nějaká přirozená a_i, b_i. Pokud najdeme $x_i = x_j$, $i \neq j$, pak máme $g^{a_i} \cdot h^{b_i} = g^{a_j} \cdot h^{b_j}$. Odtud zlogaritmováním při základě g obdržíme

$(b_i - b_j) \cdot \log_g(h) = a_j - a_i$, odkud snadno dopočítáme hledaný diskrétní logaritmus $\alpha = \log_g(h)$.

Algoritmus sice běží v průměru v čase $O(\sqrt{p})$, ale zabírá také $O(\sqrt{p})$ paměti. Proto lze využít Floydovo vylepšení a hledat v posloupnosti dvojici prvků (x_i, x_{2i}), tž. $x_i = x_{2i}$. Tímto vylepšením získáme algoritmus, který má stejnou časovou náročnost, ale navíc má pouze konstantní paměťovou složitost $O(1)$.

.....

Pohlig-Hellman: Tento algoritmus využívá znalosti rozkladu řádu grupy na součin prvočísel. Nechť tedy známe rozklad $N = p_1^{c_1} p_2^{c_2} \cdots p_k^{c_k}$, $e_i \geq 1$, pak se snažíme spočítat jednotlivá $\alpha_i = \log_g(h) \bmod p_i^{e_i}$. Odtud užitím Gausova algoritmu (pro řešení sady kongruencí podle čínské věty o zbytcích) zrekonstruujeme původní α, tak aby $\alpha_i = \alpha \bmod p_i^{e_i}$.

Všechna α_i jsou rovna nejaké kombinaci $l_0 + l_1 p_i + \cdots + l_{e_i-1} p_i^{e_i-1}$, kde $0 \leq l_j \leq p_i - 1$. K výpočtu této kombinace můžeme použít např. Floydovu úpravu Pollard-ρ algoritmu.

(Výpočet α_i)
1. $\gamma = 1$, $l_{-1} = 0$, $\bar{g} = g^{n/p_i}$, $\bar{g}$ je řádu p_i.
2. (Výpočet l_j) Pro $0 \leq j \leq e_i - 1$:
   $\gamma = \gamma\, g^{l_{j-1}\, p_i^{j-1}}$ a $\bar{h} = (h/\gamma)^{n/p_i^{j+1}}$,
   $l_j = \log_{\bar{g}} \bar{h}$, s použitím Floydova algoritmu.
3. $\alpha_i = l_0 + l_1 p_i + \cdots + l_{e_i-1}\, p_i^{e_i-1}$.

Časová složitost algoritmu (při známém rozkladu N) je $O(\sum_{i=1}^{k} e_i(\log N + \sqrt{p_i}))$ grupových operací.

Pro N pročíslo, pak Pohlig-Hellman algoritmus nepřináší žádné zlepšení oproti Pollard-ρ nebo Baby-step Giant-step algoritmům.

.....

Index calculus: Je nejsložitější známý algoritmus pro řešení DLP, nefunguje dobře ve všech používaných grupách, ale pokud už funguje, pak je subexponenciální v časové náročnosti. Nebudem zde popisovat přeně všechny kroky, pouze ideu.

1. Je potřeba vybrat relativně malou podmnožinu $S = \{p_1, \cdots, p_t\} \subset G$, tak aby podstatná část prvků v G šla vyjádřit jako součin prvků z S.
2. Vybereme náhodně přirozené k, $0 \leq k \leq N - 1$ a vypočteme g^k.
3. Pokusime se rozepsat $g^k = \prod_{i=1}^{t} p_i^{c_i}$, $c_i \geq 0$. Pokud uspějeme, získáme lineární kongruenci $k \equiv \sum_{i=1}^{t} c_i \log_g(p_i) \pmod N$.
Toto opakujeme, dokud nemáme dostatek rovnic. Tj. alespoň $t + c$, pro nějaké malé c, např. 10. To kvůli potřebě existence jednoznačného řešení sady $t + c$ rovnic.
4. Z rovnic získáme neznámé dílčí logaritmy $\log_g(p_i)$.

5. Opět vybereme náhodné k a spočítáme $h \cdot g^k$. Pokusíme se rozepsat $h \cdot g^k = \prod_{i=1}^{t} p_i^{d_i}$, $d_i \geq 0$, a pokud uspějeme, získáme výsledek:

$\alpha = \log_g(h) = \left(\sum_{i=1}^{t} d_i \log_g(p_i) - k\right) \bmod N.$

5.1.3 J. H. Cheon - analýza k-SDH

J. H. Cheon v článku [22] poukázal na principiální slabost problémů k-SDH, k-wDH, ... projevující se pro některá prvočísla p, která hrají v k-SDH problému roli velikosti pracovních grup. Ukázal redukci složitosti, jak ze vstupu pro k-SDH vypočítat vlastní tajemství α. Tato redukce je závislá na výběru prvočsla p, tedy při opatrné volbě p nelze jeho výsledky aplikovat při útocích na uvedená schémata.

Tvrzení: Ať g je generátor p-prvkové abelovy grugy G. Předpokládejme, že d dělí $p-1$. Ať $g, g_1 = g^{\alpha}, g_d = g^{\alpha^d}$ pro nějaké $\alpha \in \mathbb{Z}_p^*$ jsou dány. Pak α lze spočítat v čase $O\left(\log p \cdot (\sqrt{(p-1)/d} + \sqrt{d})\right)$ s využitím $O\left(\max\{\sqrt{(p-1)/d}, \sqrt{d}\}\right)$ paměti.

Důsledek: Ať g je generátor p-prvkové abelovy grugy G. Předpokládejme, že $p - 1 = d_1 \ldots d_t$. Ať g a $g_{(p-1)/d_i} = g^{\alpha^{(p-1)/d_i}}$ jsou dány. Pak α lze spočítat v čase $O\left(\log p \cdot \sum_{i=1}^{t} \sqrt{d_i}\right)$ s využitím $O\left(\max_{1 \leq i \leq t} \sqrt{d_i}\right)$ paměti.

Tvrzení: Ať g je generátor p-prvkové abelovy grugy G. Předpokládejme, že d dělí $p+1$. Ať g, g^{α^i}, $1 \leq i \leq 2d$, jsou dány. Pak α lze spočítat v čase řádově $O\left(\log p \cdot (\sqrt{(p+1)/d} + d)\right)$ s využitím $O\left(\max\{\sqrt{(p+1)/d}, \sqrt{d}\}\right)$ paměti.

Ve zmiňovaném článku rozebírá autor možnost praktického použití této bezpečnostní redukce. Rozebírá také některé eliptické křivky doporučené úřadem NIST USA.

5.2 Útoky proti konkrétním schématům

5.2.1 útok na MSK02 traitor tracing schéma

V [34] je představen jednoduchý útok, který pomocí lineární kombinace soukromých klíčů spolupracujících traitorů vytvoří pirátský klíč, který není dohledatelný ke komukoliv ze spolupracující skupiny uživatelů.

Ať $u_1, \ldots, u_k$ jsou spolupracující uživatelé, $K_{u_1}, \ldots, K_{u_k}$ jsou jejich soukromé klíče. Vyberme náhodně $\mu_1, \ldots, \mu_k \in \mathbb{Z}_p^*$ tak, aby $\mu_1 + \cdots + \mu_k = 1$. Vypočteme $K_0^{pir} = \mu_1 K_{u_1} + \cdots + \mu_k K_{u_k}$ a $K_1^{pir} = \mu_1 u_1 K_{u_1} + \cdots + \mu_k u_k K_{u_k}$. Pirátský klíč je dvojice (K_0^{pir}, K_1^{pir}).

V původním schématu pro $1 \leq i \leq k$ platí, že $e(K_{u_i}, u_i Q + aQ) = e(P, Q)$. My chceme pomocí pirátského klíče vypočítat $e(P, Q)$, což pak použijeme k získání session klíče s z hlavičky vysílaných dat.

Spočteme:

$$\begin{aligned} e(K_0^{pir}, aQ) \cdot e(K_1^{pir}, Q) &= e\left(\sum_{i=1}^{k} \mu_i K_{u_i}, aQ\right) \cdot e\left(\sum_{i=1}^{k} \mu_i u_i K_{u_i}, Q\right) \\ &= \prod_{i=1}^{k} e(K_{u_i}, u_i Q + aQ)^{\mu_i} = \prod_{i=1}^{k} e(P, Q)^{\mu_i} = e(P, Q). \end{aligned}$$

Máme-li dvě disjunktní skupiny uživatelů $\{u_1, \ldots, u_k\}$ a $\{u'_1, \ldots, u'_\ell\}$, pak dokážeme snadno najít čísla $\mu_1, \ldots, \mu_k$ a $\mu'_1, \ldots, \mu'_\ell$ tak, aby $\sum_{i=1}^{k} \mu_i = \sum_{i=1}^{\ell} \mu'_i = 1$ a navíc

$$K_0^{pir1} = \sum_{i=1}^{k} \mu_i K_{u_i} = \sum_{i=1}^{\ell} \mu'_i K u'_i = K_0^{pir2}$$

a

$$K_1^{pir1} = \sum_{i=1}^{k} \mu_i u_i K_{u_i} = \sum_{i=1}^{\ell} \mu'_i u'_i K u'_i = K_1^{pir2}.$$

Takže z pirátského klíče (K_0^{pir}, K_1^{pir}) nemůže být dohledatelný kdokoliv ze skupiny traitorů, protože (K_0^{pir}, K_1^{pir}) může vytvořit jakákoliv skupina spolupracujících uživatelů.

5.2.2 Slabiny ElGamal schématu

Indistinquishability (**IND**):

Útočník zvolí dvě zprávy M_0, M_1, odešle je šifrovacímu orákulu, orákulum si zvolí bit $b \in \{0, 1\}$ a pošle útočníkovi zpět šifrový text $C = (g^r, y^r M_b) = (A, B)$. Pokud by útočník uměl efektivně rozhodovat DDH problém, pak by mu stačilo se zeptat, která z trojic $(A, y, B/M_0)$ nebo $(A, y, B/M_1)$ tvoří Diffie-Hellmanovu trojici, tj. trojici tvaru (g^a, g^b, g^{ab}). Je-li totiž např. $b = 1$, pak

$$(A, y, B/M_1) = \left(g^r, g^x, \frac{(g^x)^r M_1}{M_1}\right)$$

a útočník pozná jak šifrovací orákulum zvolilo bit b.

Zranitelnost vůči CCA2:

Navíc ElGamal není vůbec bezpečný proti CCA2 útočníkovi. Uvažme, že šifrovací orákulum zvolí M a odešle útočníkovi zašifrovanou zprávu $C = (A, B) = (g^k, y^k M)$. Útočníkovi pak stačí odeslat požadavek $(A, zB) \neq (A, B)$ dešifrovacímu orákulu, které mu vrátí $\frac{B}{z.A^x} = \frac{zy^k M}{(g^k)^x} = \frac{zg^{xk} M}{g^{kx}} = zM$. Odtud útočník získá snadno M, protože z zná (sám si ho zvolil).

Zde nejenom, že útočním prolomit vlastnost IND šifrovacího schématu ElGamal, ale je dokonce schopný získat celý otevřený text za předpokladu, že má přístup k dešifrovacímu orákulu.

Kapitola 6

Shrnutí

V práci uvádíme přehled relativně nových problémů na bázi diskrétního logaritmu. Připojujeme také známé redukce mezi problémy, a sepsali jsme redukci mezi ℓ-wDH a ℓ-SDH, která je v literatuře pouze zmíněna, ale není explicitně uvedena. Je to vcelku jednoduchá redukce, ale na tomto případě ilustrujeme, že v literatuře se často nechává hodně věcí na čtenáři samotném, zejména ať si domyslí detaily nebo dokončí některé zkrácené kroky v důkazech. Uvádíme přehled schémat založených na těchto nových problémech - vybraná schémata jsou něčím vyjímečná, např. tím, že jako první měla dokázanou bezpečnost, nebo jsou postavena na novém problému, a nebo mají velmi praktické parametry.

Připojujeme krátký přehled teorie potřebné k základnímu náhledu na práci s eliptickými křivkami. Zejména nás zajímá Weilovo a Tateovo párování, jejichž výpočet se snažíme nahlédnout z praktického hlediska, spíše než je popisovat, pro neznalého čtenáře jistě velmi obtížnou, mašinerií algebraické geometrie.

V sekcích, kde uvádíme definice týkající se bezpečnosti schémat s veřejným klíčem, se snažíme přesně a srozumitelně formulovat zaváděné pojmy. Právě toto bývá v literatuře často opomíjeno. Našim cílem je stav, kdy nezasvěcený čtenář pochopí správně význam zaváděného pojmu, aniž by si sám musel domýšlet přidružené detaily, které by definice neobsáhla. Také se v celé práci snažíme používat jednotné značení, protože v literatuře si každý autor zvolí jedno z možných značení, a toho se drží. Pro zaujatého čtenáře pak může být problém pročítat více textů týkajících se podobného tématu, neboť v těchto textech se značení mezi sebou často mírně lišit. Když si pak čtenář zafixuje jednu definici a pochopí její přesný obsah, může v jiném textu, kde je definice pozměněná, narazit.

Některé definice se od sebe liší odůvodněně, např. definice hry pomocí níž definujeme bezpečnost IBE a HIBE schémat. V literatuře se už však nevysvětluje, proč se musí definice lišit. V tomto textu se u této otázky pozastavujeme a osvětlujeme

potřebu pozměnit definici pro HIBE (vůči definici pro IBE).

Nakonec připojujeme přehled známých útoků proti problému diskrétního logaritmu a také několik útoků na konkrétní schémata.

Celý text je protkán odkazy na literaturu týkající se řešených problémů a otázek. Zvídavý čtenář si tak snadno může doplnit detailnější informace, např. jednotlivé důkazy, k tématům, která jej blíže zajímají.

Literatura

[1] J. H. An, Y. Dodis, T. Rabin: *On the Security of Joint Signature and Encryption,*
Eurocrypt 2002, LNCS 2332, Springer-Verlag, pp. 83–107, 2002

[2] M. Bellare, D. Micciancio, B. Warinschi: *Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions,*
Eurocrypt 2003, LNCS 2656, Springer-Verlag, pp. 614–629, 2003

[3] M. Bellare, P. Rogaway: *Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols,*
In ACM conference on Computers and Communication Security, pp. 62–73, 1993

[4] D. Boneh, X. Boyen: *Short Signatures Without Random Oracles,*
Eurocrypt 2004, LNCS 3027, Springer-Verlag, pp. 56–73, 2004

[5] D. Boneh, X. Boyen: *Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles,*
Eurocrypt 2004, LNCS 3027, Springer-Verlag, pp. 223–238, 2004

[6] D. Boneh, X. Boyen: *Secure Identity Based Encryption Without Random Oracles,*
Crypto 2004, LNCS 3152, Springer-Verlag, pp. 443–459, 2004

[7] D. Boneh, X. Boyen, E. Goh: *Hierarchical Identity Based Encryption with Constant Size Ciphertext,*
Eurocrypt 2005, LNCS 3494, Springer-Verlag, pp. 440–456, 2005

[8] D. Boneh, X. Boyen, H. Shacham: *Short Group Signatures,*
Crypto 2004, LNCS 3152, Springer-Verlag, pp. 41–55, 2004

[9] D. Boneh, M. Franklin: *An Efficient Public Key Traitor Tracing Scheme,*
Crypto'99, LNCS 1666, Springer-Verlag, pp. 338–353, 1999

[10] D. Boneh, M. Franklin: *Identity Based Encryption from the Weil Pairing,*
Crypto 2001, LNCS 2139, Springer-Verlag, pp. 213–229, 2001

[11] D. Boneh, C. Gentry, B. Waters: *Collusion Resistant Broadcast Encryption With Short Ciphertexts and Private Keys* ,
Crypto 2005, LNCS 3621, Springer-Verlag, pp. 258–275, 2005

[12] D. Boneh, J. Katz: *Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption,*
Topics in Cryptology – CT-RSA 2005, LNCS 3376, Springer-Verlag, pp. 87–103, 2005

[13] R, Canetti, S. Halevi, J. Katz: *A Forward-Secure Public-Key Encryption Scheme,*
Eurocrypt 2003, LNCS 2656, Springer-Verlag, pp. 646, 2003

[14] R, Canetti, S. Halevi, J. Katz: *Chosen-Ciphertext Security from Identity-Based Encryption,*
Eurocrypt 2004, LNCS 3027, Springer-Verlag, pp. 207–222, 2004

[15] C. Cocks: *An Identity Based Encryption Scheme Based on Quadratic Residues,*
Cryptography and Coding, LNCS 2260, Springer-Verlag, pp. 360–363, 2001

[16] R. Cramer, V. Shoup: *A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack,*
Crypto'98, LNCS 1462, Springer-Verlag, pp. 13–25, 1998

[17] R. Cramer, V. Shoup: *Signature schemes based on the strong RSA assumption,*
ACM TISSEC, Volume 3, Issue 3, pp. 161–185, 2000

[18] D. Dolev, C. Dwork, M. Naor: *Non-Malleable Cryptography,*
`http://www.cs.huji.ac.il/~dolev/dd-publications.html`, updatovaná verze (2003) původního článku z roku 1991

[19] T. El Gamal: *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms* ,
Advances in Cryptology, LNCS 196, Springer-Verlag, pp. 10–18, 1985

[20] S. Goldwasser, S. Micali, R. Rivest: *A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attack,*
SIAM Journal on Computing, No. 17, 281–308, 1988

[21] L. Chen, Z. Cheng: *Security Proof of Sakai-Kasahara's Identity-Based Encryption Scheme,*
Cryptography and Coding, LNCS 3796, Springer-Verlag, pp. 442–459, 2005

[22] J. H. Cheon: *Security Analysis of the Strong Diffie-Hellman Problem,*
Eurocrypt 2006, LNCS 4004, Springer-Verlag, pp. 1–11, 2006

[23] A. Joux: *The Weil and Tate Pairings as Building Blocks for Public Key Cryptosystems,*
Algorithmic Number Theory, LNCS 2369, Springer-Verlag, pp. 11–18, 2002

[24] A. Joux: *Separating Decision Diffie Hellman from Computational Diffie–Hellman in Cryptographic Groups,*
Journal of Cryptology, Springer-Verlag, pp. 239–247, 2003

[25] A. Joux: *A One Round Protocol for Tripartite Diffie–Hellman,*
Journal of Cryptology, Volume 17, Number 4, Springer-Verlag, pp. 263–276, 2004

[26] S. Mitsunari, R. Sakai, M. Kasahara: *A New Traitor Tracing,*
IEICE Trans. Fundamentals, Vol.E85-A, No.2, pp.481–484, 2002

[27] A. Menezes, T. Okamoto, S. Vanstone: *Reducing elliptic curve logarithms to logarithms in a finite field,*
IEEE Trans. Inform. Theory, vol. 39, pp. 1639—1646, 1993

[28] A. Menezes, P. Oorschot, S. Vanstone : *Handbook of Applied Cryptography,*
CRC Press, 1996

[29] T. Okamoto, D. Pointcheval.: *The Gap Problems: A New Cass of Problems for the Security of Cryptographic Primitives,*
In Public Key Cryptography, PKC 2001, LNCS 1992, Springer-Verlag, pp. 104-–118., 2001

[30] C. Rackoff, D. R. Simon: *Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack ,*
Crypto'91, LNCS 576, Springer-Verlag, pp. 433–444, 1991

[31] A. Shamir: *Identity-Based Cryptosystems and Signature Schemes,*
Crypto 1984, LNCS 196, Springer-Verlag, pp. 47–53, 1984

[32] V. Shoup: *Lower Bounds for Discrete Logarithms and Related Problems,*
Eurocrypt'97, LNCS 1233, Springer-Verlag, pp. 256–266, 1997

[33] J. H. Silverman: *The Arithmetic of Elliptic Curves,*
Number 106 in Graduate Texts in Mathematics. Springer-Verlag, New York, 1992

[34] V. D. Tô, R. Safavi-Naini, F. Zhang: *New Traitor Tracing Schemes Using Bilinear Map*,
Digital Rights Management Workshop, pp. 67–76, 2003

[35] B. Waters: *Efficient Identity-Based Encryption Without Random Oracles*,
Eurocrypt 2005, LNCS 3494, Springer-Verlag, pp. 114–127, 2005

[36] V. Wei: *Tight Reductions among Strong Diffie-Hellman Assumptions*,
Cryptology ePrint Archive, Report 2005/057

[37] `www.wikipedia.org` - hesla: Chosen-ciphertext attack, Ciphertext indistinguishability, Malleability, Weil pairing