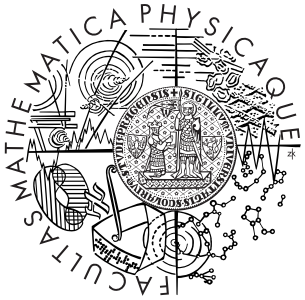
Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Anna Godušová

Důkazy bezpečnosti digitálních podpisů

Katedra algebry

Vedoucí bakalářske práce: Mgr. Štěpán Holub Ph.D.

Studijní program: Matematika

Studijní obor: Matematické metódy informačnej bezpečnosti

Praha 2011

Na tomto mieste by som sa rada poďakovala vedúcemu mojej bakalárskej práce Mgr. Štěpánovi Holubovi Ph.D. za veľké množstvo času, cenných rád a ochotu, s ktorou sa mi venoval pri písaní tejto práce, mojím rodičom a súrodencom za podporu počas celého doterajšieho štúdia, Marekovi Paškovi za veľa trpezlivosti a ochotu vyriešiť každý problém a v neposlednom rade Márii Špakovej za jazykovú korektúru.

Prehlasujem, že som túto bakalársku prácu vypracovala samostatne a výhradne s použitím citovaných prameňov, literatúry a ďalších odborných zdrojov.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona v platnom znení, najmä skutočnost, že Univerzita Karlova v Prahe má právo na uzavretie licenčnej zmluvy o použití tejto práce ako školského diela podle § 60 odst. 1 autorského zákona.

V Prahe dňa 9.12.2011

Názov práce: Důkazy bezpečnosti digitálních podpisů

Autor: Anna Godušová

Katedra (ústav): Katedra algebry

Vedúci bakalárskej práce: Mgr. Štěpán Holub Ph.D.

Abstrakt: Predložená práca sa zaoberá podpisovacími schémami, ktoré sa používajú na digitálne podpisovanie dokumentov. Presnejšie sa zaoberá obecnými podpisovacími schémami založenými na permutácii s padajúcimi dvierkami. Na úvod dokážeme bezpečnosť obecnej schémy, ďalej kvantifikujeme túto bezpečnosť a dokážeme bezpečnosť podpisovacej schémy na základe bezpečnosti permutácie s padajúcimi dvierkami. Na záver predstavíme upravenú schému s pravdepodobnostným hašovaním pre lepšie odhady v bezpečnosti.

Kľúčové slová: digitálny podpis, podpisovacia schéma, kvantifikovaná bezpečnosť, PSS

Title: Proofs of security for digital signatures

Author: Anna Godušová

Department: Department of Algebra

Supervisor: Mgr. Štěpán Holub Ph.D.

Abstract: This paper deals with signature schemes that are used for digital signing of documents. More specifically it deals with the general signature schemes based on trapdoor permutation. In the beginning we prove the security of general scheme, further we quantify this security and we prove the security of signature scheme based on security of trapdoor permutation. Finally, we introduce modified scheme with probability hashing for better estimation of the security.

Keywords: digital signature, signature scheme, exact security, PSS

Obsah

Úvod **1**

1 Značenie a definície **2**

2 Obecná schéma **5**

2.1 Dôkaz bezpečnosti obecnej schémy 5

3 Kvantifikácia **8**

3.1 Kvantifikácia bezpečnosti podpisovacej schémy 8

3.2 Kvantifikácia bezpečnosti permutácie s padajúcimi dvierkami . . . 9

3.3 Dôkaz kvantifikovanej bezpečnosti obecnej schémy 10

4 Schéma PSS **13**

4.1 Dôkaz bezpečnosti schémy $\mathcal{PSS}$ 15

5 Záver **20**

Literatúra **21**

Úvod

Na digitálne podpisovanie dokumentov sa používajú rôzne podpisovacie schémy. Niektoré z nich sa skladajú z hašovacej funkcie a z permutácie s padajúcimi dvierkami. Budeme sa zaoberať bezpečnosťou práve takýchto obecných podpisovacích schém, ktoré sa neviažu na žiadny konkrétny algoritmus - konkrétnu permutáciu.

V prvej kapitole si ozrejmime značenie a zadefinujeme pojmy používané v celej práci.

V druhej kapitole sa pozrieme na obecnú podpisovaciu schému a dokážeme bezpečnosť podpisovacej schémy na základe bezpečnosti permutácie s padajúcimi dvierkami.

Keďže nás zaujíma vyčíslená bezpečnosť, v tretej kapitole kvantifikujeme bezpečnosť podpisovacej schémy a permutácie s padajúcimi dvierkami a na základe tohto vyčíslenia ukážeme závislosť bezpečnosti podpisovacej schémy na bezpečnosti permutácie pomocou vyčíslených parametrov.

Na záver, v štvrtej kapitole, sa budeme zaoberať novou schémou s upraveným hašovaním. Vďaka tejto zmene dokážeme silnejšiu závislosť bezpečnosti podpisovacej schémy na permutácii.

Kapitola 1

Značenie a definície

V tejto kapitole si predstavíme značenie a základné definície použité v ďalšom texte.

Priestor konečných binárnych reťazcov budeme značiť $\{0,1\}^*$ a priestor binárnych reťazcov dĺžky k označíme $\{0,1\}^k$. Zápisom 1^k budeme rozumieť dĺžku vstupu k. Reťazec, ktorý vznikol spojením reťazca a a reťazca b, označíme $a \parallel b$.

Značením $x \xleftarrow{R} \{0,1\}^k$ budeme rozumieť priradenie náhodne vybraného binárneho reťazca dĺžky k premennej x.

Nech M je algoritmus a x je jeho vstup. Potom $[M(x)]$ značíme množinu všetkých možných výsledkov algoritmu M so vstupom x.

Značením $Pr[x \leftarrow S; y \leftarrow T; \ldots : p(x,\ y\ \ldots)]$ rozumieme pravdepodobnosť, že po vykonaní jednotlivých algoritmov $x \leftarrow S;\ y \leftarrow T$; atď, je výrok $p(x, y, \ldots)$ pravdivý.

Deterministickým algoritmom budeme rozumieť algoritmus, ktorého každý krok je jednoznačne definovaný.

Nedeterministický algoritmus je algoritmus, pre ktorý platí, že aspoň v jednom kroku si môže zvoliť z niekoľko možných ďalších krokov.

Polynomiálnym algoritmom rozumieme algoritmus, ktorého čas behu, tj. počet jeho krokov, je zhora obmedzený polynomiálnou funkciou veľkosti vstupu.

Pravdepodobnostný algoritmus je nedeterministický algoritmus, pre ktorý platí, že v každom kroku sú definované pravdepodobnosti možných následujúcich krokov a súčet každých takýchto pravdepodobností je rovný 1.

Pravdepodobnostný polynomiálny algoritmus budeme značiť PPT.

Funkciu $\epsilon(k) : \mathbb{N} \to \mathbb{R}$ nazveme *zanedbateľnou*, ak pre každé $c \in \mathbb{N}$ existuje k_c také, že $\epsilon(k) \leq \frac{1}{k^c}$, pre každé $k \geq k_c$. Funkciu nazveme *nezanedbateľnou*, ak nie je zanedbateľná.

Povieme, že funkciu f je *ľahké* vypočítať, ak pre každé x sa dá hodnota $f(x)$ vypočítať polynomiálnym algoritmom vo veľkosti x.

Naopak povieme, že je *ťažké* spočítať f, ak pre každý pravdepodobnostný algoritmus S so vstupom x dĺžky k je pravdepodobnosť výstupu $f(x)$ zanedbateľná funkcia. Teda $\epsilon(k) = Pr[S(x) = f(x)]$ je zanedbateľná funkcia. Ak označíme β_i sled krokov algoritmu S vedúci k spočítaniu $f(x)$, potom $\epsilon(k) = \sum_{\beta_i \in B_{f_x}} Pr[\beta_i]$, kde B_{f_x} je množina všetkých možných β_i.

Uniformnou distribúciou rozumieme rovnomerné rozdelenie, pre ktoré platí: $Pr[X = k] = \frac{1}{n}, k = 1, \ldots, n$.

Nech $n \in \mathbb{N}$, potom zobrazenie R_n z $\{0,1\}^*$ na $\{0,1\}^n$ nazveme *náhodné orákulum*, pre ktoré platí, že pre každé x je $R_n(x)$ vybrané uniformne a nezávisle. Pre náhodné orákulum platí, že pre rovnaký vstup dostaneme vždy rovnaký výstup. Náhodné orákulum budeme niekedy nazývať aj *náhodnou hašovaciou funkciou.*

Definícia 1. *Podpisovacou schémou* rozumieme trojicu algoritmov ($\mathcal{G}$, *Sign*, *Verify*) bežiacich v polynomiálnom čase, nazývajúcich sa $\mathcal{G}$ - generátor, *Sign* - podpisovací algoritmus a *Verify* - overovací algoritmus. Prvé dve sú pravdepodobnostné a posledné dve majú prístup k náhodnému orákulu. Na vstupe 1^k generátor vyprodukuje pár odpovedajúcich kľúčov (PK, SK) - verejný a súkromný kľúč. Spočítaním $\sigma \leftarrow Sign^R(SK, m)$ dostaneme podpis správy m. Na overenie (m, σ) sa spočíta $Verify^R(PK, m, \sigma) \in \{0,1\}$. Platí $Verify^R(PK, m, \sigma) = 1$ pre všetky $\sigma \in [Sign^R(SK, m)]$.

Definícia 2. *Útočník na podpisovaciu schému* F je neuniformný algoritmus bežiaci v polynomiálnom čase s prístupom k náhodnému orákulu R a k podpisovaciemu orákulu. To znamená, že útočník F môže urobiť požiadavok na zahašovanie a podpísanie správ a dostane odpoveď. Výstupom F je pár (m, σ), správa - podpis, kde m je nová správa, teda správa, na ktorú nebola vykonaná požiadavka na podpisovacie orákulum.

Povieme, že F je *úspešný*, ak jeho výstup (m, σ) spĺňa $Verify^R(PK, m, \sigma) = 1$.

Neuniformným algoritmom rozumieme nekonečnú spočetnú množinu algoritmov $\{F_n\}$, $n \in \mathbb{N}$, kde platí, že pre vstup dĺžky k sa vždy vyberie algoritmus F_k. Zápis každého algoritmu F_n je obmedzený polynomiálnou funkciou veľkosti vstupu.

Definícia 3. Podpisovacia schéma je *bezpečná*, ak pre každého útočníka F je funkcia $\epsilon(k)$, definovaná ako

$$Pr[R; (PK, SK) \leftarrow G(1^k);\ (m, \sigma) \leftarrow F^{R, Sign^R(SK,.)}(PK) : Verify^R(PK, m, \sigma) = 1],$$

zanedbateľná.

Definícia 4. *Generátor permutácie s padajúcimi dvierkami* je PPT algoritmus $\mathcal{G}$, ktorý na vstupe 1^k vráti trojicu algoritmov (f, f^{-1}, d). Prvé dve sú deterministické a posledný je pravdepodobnostný. Požadujeme, aby $[d(1^k)] \subset \{0,1\}^k$ a f a f^{-1} boli permutácie na $[d(1^k)]$ a inverzné jedna voči druhej. Ďalej požadujeme, aby sa f, f^{-1} a d dali vypočítať ľahko, teda v polynomiálnom čase a pre každý neuniformný algoritmus I bežiaci v polynomiálnom čase, je

$$\epsilon'(k) = Pr[(f, f^{-1}, d) \leftarrow \mathcal{G}(1^k);\ x \leftarrow d(1^k);\ y \leftarrow f(x) : I(f, d, y) = x]$$

zanedbateľná funkcia.

Permutáciu s padajúcimi dvierkami nazveme *uniformnou*, ak je d uniformná distribúcia na $\{0,1\}^k$, pre každé k a pre každé $(f, f^{-1}, d) \in [\mathcal{G}(1^k)]$.

Kapitola 2

Obecná schéma

V predchádzajúcej kapitole sme si ujasnili značenie a definície. V tejto kapitole sa pozrieme na prvú obecnú schému. Všetky podpisovacie schémy v tejto práci používajú permutáciu s padajúcimi dvierkami a na nej definujú svoju bezpečnosť. V tejto obecnej schéme ukážeme, ako sa presne postupuje pri podpisovaní správy a overovaní podpisu a dokážeme bezpečnosť podpisovacej schémy na základe bezpečnosti permutácie s padajúcimi dvierkami, kde v dôkaze vysvetlíme jednotlivé kroky útočníka.

Pozrime sa formálne na obecnú schému. Zafixujeme si generátor permutácie s padajúcimi dvierkami $\mathcal{G}$. Budeme predpokladať, že je uniformný. Nech $h : \{0,1\}^* \to \{0,1\}^k$ značí náhodnú hašovaciu funkciu. Ďalej nech $\mathcal{G}$ na vstupe 1^k spočíta $(f, f^{-1}, d) \leftarrow \mathcal{G}(1^k)$ a nastaví $PK = f$ a $SK = f^{-1}$. To, že $\mathcal{G}$ vygeneruje f a f^{-1} znamená, že ich definuje a vie spočítať, teda značením $PK = f$ a $SK = f^{-1}$ rozumieme, že ten, kto má PK, vie spočítať f a ten, kto má SK, vie spočítať f^{-1}. Podpisovacia schéma je $(\mathcal{G}, Sign^h, Verify^h)$, kde $Sign^h(f^{-1}, m) = f^{-1}(h(m))$ a $Verify^h(f, m, \sigma)$ sa rovná 1 práve vtedy, keď $f(\sigma) = h(m)$.

2.1 Dôkaz bezpečnosti obecnej schémy

Tvrdenie 2.1.1. *Majme permutáciu s padajúcimi dvierkami s generátorom $\mathcal{G}$. Potom podpisovacia schéma $(\mathcal{G}, Sign, Verify)$ je bezpečná, ak každý útočník na podpisovaciu schému F je úspešný s pravdepodobnosťou $\epsilon(k)$, ktorá je zanedbateľná.*

Dôkaz. Dokážeme nepriamo. Predpokladajme, že útočník na podpisovaciu schému F je úspešný s pravdepodobnosťou $\epsilon(k)$, ktorá nie je zanedbateľná. Skonštruujeme algoritmus $I(f, d, y)$ taký, že

$$\epsilon'(k) = Pr[(f, f^{-1}, d) \leftarrow \mathcal{G}(1^k);\ x \leftarrow d(1^k);\ y \leftarrow f(x) : I(f, d, y) = x]$$

nie je zanedbateľná. Teda generátor $\mathcal{G}$ nie je generátor permutácie s padajúcimi dvierkami.

Máme teda algoritmus I, ktorý dostane na vstupe y a chce spočítať $f^{-1}(y)$. Vie, že útočník F dokáže úspešne podpísať nejakú správu, preto sa ho pokúsi využiť na podpísanie svojej správy y. Bude postupovať takto. Spustí generátor $\mathcal{G}(1^k)$, ktorý vráti (f, f^{-1}, d), nastaví $PK = f$, tj. ten, kto má PK, vie spočítať f, a spustí útočníka F so vstupom PK. Predpokladáme, že útočník F spraví presne q_{hash} rôznych požiadaviek na zahašovanie správ a keď urobí požiadavku m na podpis, tak už predtým spravil požiadavku na zahašovanie správy m. Algoritmus I si náhodne vyberie $j \in \{1, \ldots, q_{hash}\}$ a na j-té miesto podstrčí svoje y. Ukážeme, ako I odpovedá na hašovacie požiadavky a na podpisovacie požiadavky.

1. Nech m_i značí i-tú požiadavku útočníka F na hašovanie. Ak $i = j$, algoritmus I odpovie svojím y, ináč si náhodne vyberie $x_i \leftarrow \{0,1\}^k$ a vráti $y_i = f(x_i)$.

2. Nech F urobí požiadavku na podpis m. Ak $m = m_j$, algoritmus I pripustí zlyhanie, pretože nevie podpísať správu y, ktorú na j-té miesto podstrčil. Ak $m \neq m_j$, teda $m = m_i$ pre $i \neq j$, algoritmus I vráti x_i, čo je právoplatný podpis, ktorý si vybral v hašovacej požiadavke.

Po týchto odpovediach útočník F odpovie (m, σ). Algoritmus I sa pozrie na m a zachová sa takto: ak $m \neq m_j$, I pripustí zlyhanie a skončí, pretože nezískal $f^{-1}(y)$. Ak $m = m_j$ a $f(\sigma) = h(m)$ potom I odpovie σ ako výsledok $f^{-1}(y)$, ktorý sa snažil získať a skončí. Ak $f(\sigma) \neq h(m)$, opäť pripustí zlyhanie. Uvedomme si, že ak výstup útočníka F je (m, σ) spĺňujúci $m = m_j$, potom podľa definície m_j nebola požiadavka na podpisovacie orákulum.

Pozrime sa na pravdepodobnosti $\epsilon(k)$ a $\epsilon'(k)$. Označme si skutočnosť, kedy je útočník F úspešný, ako S_k. Pravdepodobnosť $\epsilon'(k)$, teda pravdepodobnosť, že sa I podarí úspešne nájsť $f^{-1}(y)$, nastane vtedy, ak platí S_k a výstup F je (m, σ)

splňujúci $m = m_j$. Z toho máme $\epsilon'(k) = Pr[S_k \ \wedge \ (m \ = \ m_j)]$. Útočník F však môže uspieť, aj keď jeho výstupom je správa, ktorú si predtým nenechal zahašovať, tj. pre $m \notin \{m_1, \ldots, m_{q_{hash}}\}$. V tejto situácii F uspeje s pravdepodobnosťou len 2^{-k}, pretože máme náhodnú hašovaciu funkciu a počet rôznych podpisov je 2^k. Teda pravdepodobnosť, že je F úspešný, je

$$\epsilon(k) = 2^{-k} + \sum_{i=1}^{q_{hash}} Pr[S_k \wedge (m = m_i)] \tag{2.1}$$

čiže

$$\epsilon(k) - 2^{-k} = \sum_{i=1}^{q_{hash}} Pr[S_k \wedge (m = m_i)]. \tag{2.2}$$

Vzhľadom k tomu, že j volíme náhodne z $\{m_1, \ldots, m_{q_{hash}}\}$ nás zaujíma stredná hodnota pravdepodobnosti $Pr[S_k \wedge (m = m_j)]$. A keďže j je vybrané uniformne náhodne, stredná hodnota $Pr[S_k \wedge (m = m_j)]$ sa spočíta ako

$$\frac{\sum_{i=1}^{q_{hash}} Pr[S_k \wedge (m \ = \ m_i)]}{q_{hash}}.$$

Takže dostaneme

$$\epsilon(k) - 2^{-k} = q_{hash} \cdot Pr[S_k \wedge (m = m_j)]. \tag{2.3}$$

Využijeme rovnosť $Pr[S_k \wedge (m = m_j)] = \epsilon'(k)$ a dostaneme

$$\epsilon'(k) \geq (\epsilon(k) - 2^{-k})/q_{hash} \tag{2.4}$$

a to nie je zanedbateľná funkcia, ako sme predpokladali. □

Kapitola 3

Kvantifikácia

Dokázali sme, že obecná schéma je bezpečná. Nás ale zaujíma, nakoľko je bezpečná, ako presne súvisí bezpečnosť podpisovacej schémy s bezpečnosťou permutácie s padajúcimi dvierkami. Kvantifikujeme bezpečnosť podpisovacej schémy a permutácie. To znamená, že vyčíslime ich bezpečnosť na základe nejakých parametrov. Napríklad o podpisovacej schéme povieme, že je $(t, q_{sig}, q_{hash}, \epsilon)$-*bezpečná*, kde t, q_{sig}, q_{hash} sú parametre, na ktorých záleží s akou pravdepodobnosťou ϵ sa útočníkovi podarí nájsť platný podvrh podpisu. Podobne si definujeme, kedy je permutácia bezpečná. Ukážeme závislosť jednotlivých parametrov a dokážeme, že bezpečnosť podpisovacej schémy úzko súvisí s bezpečnosťou permutácie s padajúcimi dvierkami.

Nasledujú definície pojmov pre kvantifikáciu. Potom v tvrdeniu pristúpime k presnému vyčísleniu a následne toto tvrdenie dokážeme.

3.1 Kvantifikácia bezpečnosti podpisovacej schémy

Podpisovaciu schému sme definovali v definícii 1. Ďalej povieme, že podpisovacia schéma je $(t, q_{sig}, q_{hash}, \epsilon)$-*bezpečná*, ak útočník F, ktorý má k dispozícii verejný kľúč PK, je limitovaný časom $t(k)$, môže si vybrať až do $q_{sig}(k)$ správ, na ktoré vyvolá podpisovacie orákulum a môže vyvolať hašovacie orákulum $q_{hash}(k)$ krát, je úspešný vo falšovaní novej správy s pravdepodobnosťou maximálne $\epsilon(k)$. Novou správou sa myslí správa, na ktorú predtým nebola vyvolaná požiadavka o podpis.

Pre potreby dôkazov schém si presne definujeme útočníka F. Povieme, že útočník je (t, q_{sig}, q_{hash})-*útočník*, ak čas jeho behu plus veľkosť jeho zápisu je obmedzený $t(k)$, urobí maximálne $q_{sig}(k)$ požiadaviek na podpisovacie orákulum a maximálne $q_{hash}(k)$ požiadaviek na rôzne hašovacie orákula. Povieme, že takýto útočník F $(t, q_{sig}, q_{hash}, \epsilon)$-*prelomí* podpisovaciu schému, ak pre každé k, pravdepodobnosť, že výstupom F je platný podvrh je najmenej $\epsilon(k)$.

Útočníkovi musíme obmedziť veľkosť zápisu, pretože ináč by vo svojom zápise mohol obsahovať tabuľku predpočítaných hodnôt, čím by ušetril čas na ich výpočet.

Budeme predpokladať, že útočník nikdy nezopakuje tú istú požiadavku na hašovacie orákulum a čas $t(k)$ zahrňuje čas potrebný na odpoveď podpisovacieho orákula.

3.2 Kvantifikácia bezpečnosti permutácie s padajúcimi dvierkami

Definícia permutácie s padajúcimi dvierkami je popísaná v definícii 4. Ďalej povieme, že permutácie s padajúcimi dvierkami TP[1] je (t', ϵ')-*bezpečná*, ak pre náhodne vybrané $y \in \{0,1\}^k$ a v limitovanom čase $t'(k)$ útočník úspešne nájde $f^{-1}(y)$ s pravdepodobnosťou maximálne $\epsilon'(k)$.

Pre pochopenie dôkazov schém si definujeme invertovací algoritmus I a bezpečnosť permutácie s padajúcimi dvierkami v závislosti na I. Povieme, že *invertovací algoritmus* I permutácie s padajúcimi dvierkami je t'-invertor, kde $t' : \mathbb{N} \to \mathbb{N}$, ak čas jeho behu plus veľkosť jeho zápisu je obmedzený $t'(k)$. Povieme, že algoritmus I (t', ϵ')-*prelomí* permutáciu s padajúcimi dvierkami, kde $\epsilon' : \mathbb{N} \to [0,1]$, ak I je t'-invertor a pre každé k je pravdepodobnosť úspechu I najmenej $\epsilon'(k)$.

[1] z angl. trapdoor permutation

3.3 Dôkaz kvantifikovanej bezpečnosti obecnej schémy

Tvrdenie 3.3.1. *Predpokladajme, že TP je (t', ϵ')-bezpečná. Potom pre hocijaké q_{sig}, q_{hash} je podpisovacia schéma $(t, q_{sig}, q_{hash}, \epsilon)$-bezpečná, ak*

$$t(k) = t'(k) - [q_{sig}(k) + q_{hash}(k) + 1] \cdot \Theta(k^n) \quad (3.1)$$

$$\epsilon(k) = [q_{sig}(k) + q_{hash}(k) + 1] \cdot \epsilon'(k). \quad (3.2)$$

Dôkaz. Toto tvrdenie dokážeme obdobne, ako dôkaz obecnej schémy 2.1. Dokazujeme nepriamo. Predpokladajme, že F je útočník, ktorý $(t, q_{sig}, q_{hash}, \epsilon)$-prelomí podpisovaciu schému. Skonštruujeme invertovací algoritmus I, ktorý (t', ϵ')-prelomí TP. To znamená, že pre náhodne vybrané $y \in \{0,1\}^k$ algoritmus I úspešne nájde $f^{-1}(y)$ v limitovanom čase $t'(k)$ s pravdepodobnosťou minimálne $\epsilon'(k)$. Čiže

$$Pr[(f, f^{-1}, d) \leftarrow \mathcal{G}(1^k);\ x \leftarrow d(1^k);\ y \leftarrow f(x) : I(f, d, y) = x] > \epsilon'(k).$$

Máme teda algoritmus I, ktorý dostane na vstupe y a chce spočítať $f^{-1}(y)$. Spustí generátor permutácie $\mathcal{G}(1^k)$, ktorý vráti (f, f^{-1}, d), nastaví $PK = f$ a spustí útočníka F so vstupom PK. Teda útočník F bude vedieť spočítať f. Algoritmus I bude odpovedať na požiadavky útočníka F na hašovanie a podpísanie správ. Z definície útočníka F môže F spraviť až q_{hash} požiadaviek na zahašovanie správ a až q_{sig} požiadaviek na podpísanie. Predpokladajme, že útočník F využije všetky tieto požiadavky. Ďalej predpokladajme, že ak F urobí požiadavku m na podpis, tak už predtým spravil požiadavku na zahašovanie správy m. To môžme podľa autorov v článku [1] predpokladať preto, lebo keby prišiel útočník F a spravil požiadavku na podpis, aj keď predtým nespravil požiadavku na zahašovanie, algoritmus I by musel dodatočne sám urobiť požiadavku na zahašovanie danej správy. Ináč by totiž nemohol vrátiť právoplatný podpis danej správy. To znamená, že skutočný počet hašovacích požiadaviek je $q_{hash} + q_{sig}$.

Algoritmus I si ďalej náhodne vyberie $j \in \{1, \ldots, q_{hash} + q_{sig}\}$ a na j-té miesto podstrčí svoje y. Ukážeme ako I odpovedá na hašovacie požiadavky a na podpisovacie požiadavky.

1. Nech m_i značí i-tú požiadavku útočníka F na hašovanie. Ak $i = j$, algoritmus I odpovie svojím y, ináč si náhodne vyberie $x_i \leftarrow \{0,1\}^k$ a vráti $y_i = f(x_i)$.

2. Nech F urobí požiadavku na podpis m. Ak $m = m_j$, algoritmus I pripustí zlyhanie, pretože nevie podpísať správu y, ktorú na j-té miesto podstrčil. Ak $m \neq m_j$, teda $m = m_i$ pre $i \neq j$, algoritmus I vráti x_i, čo je právoplatný podpis, ktorý si vybral v hašovacej požiadavke.

Po tom, čo útočník F dostane odpovede na svoje požiadavky, odpovie (m, σ). Podobne ako v obecnej schéme, algoritmus I odpovie σ ako výsledok $f^{-1}(y)$ len ak $m = m_j$ a $f(\sigma) = h(m)$.

Ukážme si, z čoho plynú uvedené rovnosti v tvrdení. Pre rovnosť $t(k)$ si uvedomme, že algoritmus I spúšťa beh útočníka F. Z toho dostaneme, že čas behu I je čas behu F plus čas na vybratie hodnôt y_i. Časovo najnáročnejšou časťou v algoritme je jeden výpočet TP pre každé y_i, čo je polynomiálny čas veľkosti vstupu rovný k. Teda

$$t'(k) = t(k) + A \cdot \Theta(k^n),$$

kde A značí počet y_i. Ten je rovný práve $q_{hash}+q_{sig}+1$, čo je predpokladaný počet hašovacích požiadaviek plus jedno overenie výstupného podvrhu algoritmom I. Celkovo

$$t'(k) = t(k) + (q_{hash} + q_{sig} + 1) \cdot \Theta(k^n).$$

Pre druhú rovnosť si opäť označme skutočnosť, kedy je F úspešný, ako S_k. Uvedomme si, že algoritmus I je úspešný, ak je úspešný útočník F a zároveň platí $m = m_j$. Teda $\epsilon'(k) = Pr[S_k \ \wedge \ (m \ = \ m_j)]$. Útočník F však môže byť úspešný aj vtedy, keď jeho výstupom je správa, ktorú si predtým nenechal zahašovať, tj. ak $m \notin \{m_1, \ldots, m_{q_{hash}+q_{sig}}\}$. V tejto situácii F uspeje s pravdepodobnosťou len 2^{-k}, pretože máme náhodnú hašovaciu funkciu a počet rôznych podpisov je 2^k. Podobne ako v obecnej schéme dostávame

$$\epsilon(k) = 2^{-k} + \textstyle\sum_{i=1}^{q_{hash}+q_{sig}} Pr[S_k \wedge (m = m_i)]$$

a z rovnakého argumentu budeme počítať strednú hodnotu $Pr[S_k \ \wedge \ (m \ = \ m_j)]$. Teda máme

$$\epsilon'(k) = \frac{\epsilon(k) - 2^{-k}}{q_{sig}(k) + q_{hash}(k)}.$$

To znamená, že algoritmus I je úspešný s pravdepodobnosťou maximálne $\frac{\epsilon(k) - 2^{-k}}{q_{sig}(k) + q_{hash}(k)}$. Potom takisto platí

$$\epsilon'(k) = \frac{\epsilon(k)}{q_{sig}(k) + q_{hash}(k) + 1},$$

kde sme iba znížili hranicu maximálnej úspešnosti algoritmu I, pretože

$$\frac{\epsilon(k) - 2^{-k}}{q_{sig}(k) + q_{hash}(k)} > \frac{\epsilon(k)}{q_{sig}(k) + q_{hash}(k) + 1},$$

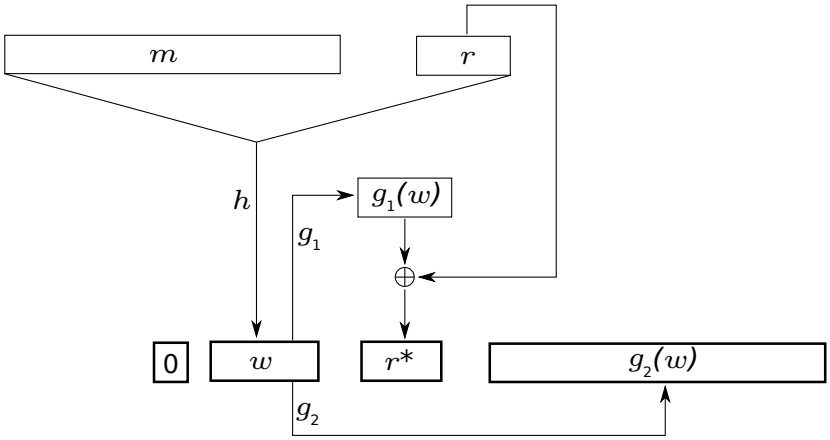
pre $\epsilon(k) > \frac{q_{sig}(k) + q_{hash}(k) + 1}{2^k}$.

□

Dokázali sme, že bezpečnosť podpisovacej schémy úzko súvisí s bezpečnosťou permutácie s padajúcimi dvierkami. Avšak z dokázaného tvrdenia plynie, že pravdepodobnosť ϵ môže byť výrazne väčšia než pravdepdobnosť ϵ'. To znamená, že aj keby bola bezpečnosť TP silná, môže byť bezpečnosť obecnej podpisovacej schémy slabá. Táto skutočnosť by sa dala vylepšiť zväčšením parametru k, no potom by sa výpočet permutácie stal výpočetne náročnejší. Ako autori v článku [1] poznamenajú, nie je známy ani iný dôkaz, kde by sa rovnosť pre t nezmenila, ale zlepšoval by sa dolný odhad pre ϵ. Predvedieme preto novú schému. Upravíme hašovanie na pravdepodobnostné, a tým dostaneme lepšiu bezpečnosť.

Kapitola 4

Schéma PSS



Obr. 4.1: Schéma $\mathcal{PSS}$

V tejto kapitole predvedieme novú schému $\mathcal{PSS}$[1]. Oproti predchádzajúcim schémam má upravené hašovanie na pravdepodobnostné, čím dostaneme lepšie odhady bezpečnosti podpisovacej schémy v závislosti na bezpečnosti permutácie s padajúcimi dvierkami. Odhad pre $t(k)$ sa nezmení, ale výrazne zlepšíme odhad pre $\epsilon(k)$. To docielime využitím pravdepodobnostného hašovania, vďaka čomu bude útočník na permutáciu úspešný vždy, keď bude úspešný útočník na podpisovaciu schému.

Jednotlivé zmeny v hašovaní popisuje obrázok 4.1. Pravdepodobnosť hašovania docielime tým, že do hašovacieho procesu nebude vstupovať samotná správa m, ale zreťazená s náhodne vygenerovanou inicializačnou hodnotou r. To znamená, že jedna správa môže mať viac rôznych podpisov - záleží na inicializač-

[1] z angl. The Probabilistic Signature Scheme

nej hodnote. Ďalšou zmenou v schéme $\mathcal{PSS}$ je, že na podpisovanie a overovanie sa nepoužíva jedna, ale dve hašovacie funkcie. *Kompresná* hašovacia funkcia $h : \{0,1\}^* \to \{0,1\}^{k_1}$ a hašovacia funkcia $g : \{0,1\}^{k_1} \to \{0,1\}^{k-k_1-1}$, tzv. *generátor*. Tieto zmeny spôsobujú, že do odhadovania závislosti medzi podpisovacou schémou a permutáciou s padajúcimi dvierkami vstupujú ďalšie parametre. V tomto prípade to budú parametre k_0 a k_1 hašovacích funkcí h a g, spĺňajúce $1 < k_0, k_1 < k$ a $k_0 + k_1 \leq k - 1$.

V tejto chvíli pridáme ešte jeden obmedzujúci predpoklad pre schému $\mathcal{PSS}$. Budeme predpokladať, že permutácia s padajúcimi dvierkami spĺňa vlastnosť multiplikativity, ktorú neskôr využijeme v dôkaze.

Formálne je $\mathcal{PSS}$ podpisovacia schéma ($Gen\mathcal{PSS}$, $Sign\mathcal{PSS}$, $Verify\mathcal{PSS}$). Zafixujeme si generátor permutácie s padajúcimi dvierkami $Gen\mathcal{PSS}$. Na vstupe 1^k spočíta (f, f^{-1}, d), to znamená, že ich definuje a vie spočítať, ďalej nastaví $PK = f$, $SK = f^{-1}$. Teda platí, že ten, kto má PK, vie spočítať f a ten, kto má SK, vie spočítať f^{-1}. Definujeme funkciu g_1, ktorá na vstupe $w \in \{0,1\}^{k_1}$ vráti prvých k_0 bitov z $g(w)$ a funkciu g_2, ktorá na vstupe $w \in \{0,1\}^{k_1}$ vráti zvyšných $k - k_0 - k_1 - 1$ bitov z $g(w)$. To, ako sa podpisuje a overuje je zhrnuté do následujúceho zápisu a nižšie uvádzame presný popis jednotlivých krokov.

$Sign\mathcal{PSS}(f^{-1}, m)$
$r \xleftarrow{R} \{0,1\}^{k_0}$; $w \leftarrow h(m \parallel r)$; $r^* \leftarrow g_1(w) \oplus r$
$u \leftarrow 0 \parallel w \parallel r^* \parallel g_2(w)$
return $f^{-1}(u)$

$Verify\mathcal{PSS}(f, m, \sigma)$
$u \leftarrow f(\sigma)$
Rozdeľ u na $b \parallel w \parallel r^* \parallel \gamma$, kde b je prvý bit z u, w dalších k_1 bitov, r^* dalších k_0 bitov a γ zvyšné bity.
$r \leftarrow r^* \oplus g_1(w)$
if $(h(m \parallel r) = w$ a $g_2(w) = \omega$ a $b = 0)$ **then return** 1
else return 0

V podpisovaní sa na začiatku náhodne vyberie inicializačná hodnota r dĺžky k_0. Túto hodnotu zreťazíme so správou m a pošleme ako vstup pre kompresnú hašovaciu funkciu h. Výstupom je k_1 bitový reťazec w, ktorý ďalej postúpime ako vstup pre druhú hašovaciu funkciu g. Výsledok $g(w)$ rozdelíme na dve časti. Prvých k_0 bitov tvorí hodnotu $g_1(w)$ a zvyšných $k-k_0-k_1-1$ bitov hodnotu $g_2(w)$. Hodnotu $g_1(w)$ použijeme na zamaskovanie inicializačnej hodnoty r spočítaním $r \oplus g_1(w) = r^*$. Na záver zreťazíme bit 0 s vypočítanými hodnotami w, r^* a $g_2(w)$. Na začiatku vyžadujeme bit 0, aby sme zaistili, že reťazec u bude skutočne patriť do definičného oboru permutácie f^{-1}. Ak máme danú permutácia s padajúcimi dvierkami, môže nastať prípad, kedy reťazec u patriť definičnému oboru nebude. Ako príklad si vezmime $f^{-1} : Z_{35}^* \to Z_{35}^*$. Potom $35_{10} = 100011_2$, avšak reťazec u by mohol nadobudnúť hodnoty až $11111_2 = 63_{10}$, kde časť hodnôt nebude patriť do Z_{35}^*. Keď však na začiatku budeme predpokladať bit 0, maximálna hodnota by mohla byť $011111_{10} = 31_{10}$, čo nám už problém robiť nebude. Takže zreťazením bitu 0 s w, r^* a $g_2(w)$ dostaneme $u = 0 \parallel w \parallel r^* \parallel g_2(w)$. A na záver spočítaním $f^{-1}(u) = \sigma$ dostaneme výsledný podpis.

Pre overovanie sa spočíta *Verify*(f, m, σ) tj. $f(\sigma) = u$ a z u sa obnoví r^*, w, r a overí sa, či bolo u správne skonštruované.

4.1 Dôkaz bezpečnosti schémy $\mathcal{PSS}$

Tvrdenie 4.1.1. *Prepodkládajme, že TP je (t', ϵ')-bezpečná. Potom pre hocijaké q_{sig}, q_{hash} je podpisovacia schéma $(t, q_{sig}, q_{hash}, \epsilon)$-bezpečná, ak*

$$t(k) = t'(k) - [(q_{sig}(k) + q_{hash}(k)) \cdot k_0 + 1] \cdot \Theta(k^n) \tag{4.1}$$

$$\epsilon(k) = \epsilon'(k) + [2 \cdot (q_{sig}(k) + q_{hash}(k))^2] \cdot (2^{-k_0} + 2^{-k_1}). \tag{4.2}$$

Dôkaz. Princíp dôkazu je opäť podobný dôkazu obecnej schémy 2.1. Taktiež dokazujeme nepriamo. Teda predpokladajme, že F je útočník, ktorý $(t, q_{sig}, q_{hash}, \epsilon)$-prelomí $\mathcal{PSS}$. Potom predvedieme algoritmus I, ktorý (t', ϵ')-prelomí *TP*. Teda, že pre náhodne vybrané $y \in \{0,1\}^k$ algoritmus I úspešne nájde $f^{-1}(y)$ v limitovanom čase $t'(k)$ s pravdepodobnosťou minimálne $\epsilon'(k)$.

Zopakujme predpoklady. Máme algoritmus I, ktorý na vstupe dostane y a chce spočítať $f^{-1}(y)$. Predpokladame útočníka F, ktorý dokáže úspešne podpísať

nejakú správu. Preto sa ho algoritmus I pokúsi využiť. Nepodstrčí však svoje y medzi hašovacie požiadavky ako v predchádzajúcich schémach, ale využije pravdepodobnostné hašovanie, aby z každého platného podvrhu získal $f^{-1}(y)$. To docieli špeciálnym tvarom zahašovanej správy $h(m \parallel r)$. Popíšeme, ako bude algoritmus I postupovať. Spustí generátor $Gen\mathcal{PSS}$, ktorý vráti (f, f^{-1}, d), nastaví $PK = f$ a spustí útočníka F so vstupom PK. Budeme predpokladať, že F spraví q_{hash} požiadaviek na hašovacie orákulum h alebo g a q_{sig} požiadaviek na podpisovacie orákulum a žiadne z požiadaviek na hašovacie orákulum sa nebude opakovať. Takisto predpokladáme, že ak F urobí požiadavku m na podpis, tak už predtým spravil požiadavku na zahašovanie správy m. Označme si postupnosť všetkých požiadaviek na orákula ako $q_1, \ldots, q_{q_{sig}+q_{hash}}$. Počas odpovedania na tieto požiadavky algoritmus I definuje hašovacie funkcie h a g.

Teraz už predvedieme, ako I odpovedá na jednotlivé požiadavky.

Nech $q_i = m_i$ je *požiadavka na podpis*. Teda algoritmus I má odpovedať podpisom správy m. Problém je, že I nevie spočítať f^{-1}. Preto si najprv náhodne vyberie podpis $x_i \leftarrow \{0,1\}^k$ a spočíta $u_i = f(x_i)$, pričom u_i má byť výsledok hašovacieho podpisu správy $m \parallel r$. Teda v tejto chvíli algoritmus I definuje hašovacie funkcie h a g. Najprv rozdelí u_i na $0 \parallel w_i \parallel r_i^* \parallel \gamma_i$ a definuje $h(m_i \parallel r_i) = w$ a $g(w) = r_i^* \oplus r_i \parallel \gamma_i$ pre nejaké náhodné r_i. Potom môže byť x_i vrátený ako právoplatný podpis správy $m_i \parallel r_i$.

Nasleduje teraz presný popis inštrukcií, ktoré algoritmus I vykoná po prijatí požiadavky na podpis:

(1) $i := i + 1$ a $m_i := q_i$.

(2) $r_i \overset{R}{\leftarrow} \{0,1\}^{k_0}$.

(3) **if** $\exists\, j : j < i \wedge r_j = r_i$ **then** zlyhá.

(4) **Repeat** $x_i \overset{R}{\leftarrow} \{0,1\}^k$; $u_i \leftarrow f(x_i)$ **until** prvý bit u_i je 0.

(5) Rozdeľ u_i na $0 \parallel w_i \parallel r_i^* \parallel \gamma_i$.

(6) Nastav $h(m_i \parallel r_i) = w_i$.

(7) **if** $\exists\, j : j < i \wedge w_j = w_i$ **then** zlyhá.

(8) Nastav $g_1(w_i) = r_i^* \oplus r_i$; $g_2(w_i) = \gamma_i$; $g(w_i) = g_1(w_i) \parallel g_2(w_i)$.

(9) **Return** útočníkovi F hodnotu x_i ako odpoveď na požiadavku o podpis $q_i = m_i$.

Nech q_i je *požiadavka na hašovacie orákulum* h, ktorá má dĺžku najmenej k_0. Ako už bolo spomenuté, chceme, aby algoritmus I, pre každý platný podvrh σ útočníka F na správu $m \parallel r$, získal $f^{-1}(y)$. Keďže nemáme samotnú správu m, ale $m \parallel r$, budeme navyše požadovať špeciálny tvar pre hodnotu u, čo je výstup hašovacieho procesu so vstupom $q_i = m \parallel r$. Teda sa budeme na u pozerať ako na tvar $y \cdot f(x_i)$, kde x_i je náhodné. Ak totiž príde útočník F so $\sigma = f^{-1}(y \cdot f(x_i)) = f^{-1}(y) \cdot x_i$, potom I z $f^{-1}(y) \cdot x_i$ získa $f^{-1}(y)$. Teda z každého platného podvrhu získa algoritmus I týmto spôsobom hľadané $f^{-1}(y)$. Nakoniec nezabudneme dodefinovať $h(m \parallel r) = w$ a $g(w) = r^* \oplus r \parallel \gamma$.

V tejto chvíli si uvedomme, že na to, aby platila rovnosť $f^{-1}(y \cdot f(x_i)) = f^{-1}(y) \cdot x_i$, potrebujeme, aby permutácia s padajúcimi dvierkami spĺňala vlastnosť multiplikativity, čo sme však predpokladali.

Opäť bude nasledovať presný popis inštrukcií, ktoré I vykoná po prijatí požiadavky q_i na hašovacie orákulum h:

(1) $i := i + 1$ a rozdeľ q_i na $m_i \parallel r_i$, kde r_i je posledných k_0 bitov z q_i.

(2) Povieme, že q_i je *staré*, ak $\exists\, j : j < i \ \wedge \ m_j \parallel r_j = m_i \parallel r_i$, ináč povieme, že je q_i *nové*. (Keďže požiadavky na hašovacie orákulum h sa neopakujú, q_i je staré, práve vtedy, keď m_i bola podpisovacia požiadavka m_j a v procese odpovedania sa vybralo $r_j = r_i$.)

If q_i je staré **then** nastav $(w_i, r_i^*, \gamma_i) = (w_j, r_j^*, \gamma_j)$ a **return** w_j $(= h(m_j \parallel r_j))$ **else** choď na (3).

(3) **Repeat** $x_i \overset{R}{\leftarrow} \{0,1\}^k$; $z_i \leftarrow f(x_i)$; $u_i \leftarrow y \cdot z_i$ **until** prvý bit u_i je 0.

(4) Rozdeľ u_i na $0 \parallel w_i \parallel r_i^* \parallel \gamma_i$.

(5) Nastav $h(m_i \parallel r_i) = w_i$.

(6) **if** $\exists\, j : j < i \wedge w_j = w_i$ **then** zlyhá.

(7) Nastav $g_1(w_i) = r_i^* \oplus r_i$; $g_2(w_i) = \gamma_i$; $g(w_i) = g_1(w_i) \parallel g_2(w_i)$.

(8) **Return** pre F hodnotu w_i ako odpoveď na požiadavku $q_i = m_i \parallel r_i$ o hašovacie orákulum h.

Nech q_i je *požiadavka na hašovacie orákulum* g, ktorá má dĺžku najmenej k_1.

Popis inštrukcií:

(1) $i := i + 1$ a $w_i := q_i$.

(2) **if** $\exists\ j\ :\ j < i\ \wedge\ w_j = w_i$ **then return** $g(w_j)$ **else** vyber náhodne $\alpha \xleftarrow{R} \{0,1\}^{k-k_1-1}$, nastav $g(w_i) := \alpha$ a **return** α.

Pozrime sa teraz na platnosť rovností v tvrdení. Pre rovnosť $t(k)$ si uvedomme, že algoritmus I spúšťa beh útočníka F. Teda čas behu algoritmu I je čas behu útočníka F plus čas na spočítanie permutácie f, čo je časovo najnáročnejšie z celého algoritmu. Čiže $t'(k) = t(k) + A \cdot \Theta(k^n)$, kde A značí počet výpočtov permutácie f. Tie sa počítajú v kroku (4) v odpovedaní na podpisovacie orákulum a v kroku (3) v odpovedaní na hašovacie orákulum h. Oba tieto kroky sa nedajú obmedziť, no očakávaný čas sú dva behy oboch smyčiek. Takže zastavme smyčku po k_0 krokoch. Z toho dostávame, že počet výpočtov permutácie f je $(q_{sig} + q_{hash}) \cdot k_0$. Pripočítame ešte jeden výpočet permutácie f z overovania výsledného podvrhu (m, σ) útočníka F. Teda $A = (q_{sig} + q_{hash}) \cdot k_0 + 1$. Celkovo dostaneme

$$t(k) = t'(k) - ((q_{sig} + q_{hash}) \cdot k_0 + 1) \cdot \Theta(k^n).$$

Podobne si rozoberme pravdepodobnosť $\epsilon(k)$. Pripomeňme, že algoritmus I je úspešný, ak sa mu podarí získať $f^{-1}(y)$. To je možné len v prípade, kedy nikdy nezlyhá pri odpovedaní na hašovacie požiadavky alebo na podpisovacie požiadavky. Taktiež vieme, že algoritmus I získa $f^{-1}(y)$ z výstupu útočníka F, ktorý je z predpokladu úspešný nezávisiac na tom, či algoritmus I zlyhá alebo nezlyhá. Teda pravdepodobnosť úspechu algoritmu I je rovná pravdepodobnosti úspechu útočníka F mínus pravdepodobnosť, že niekedy I zlyhá. Pre vypočítanie tejto pravdepodobnosti si označme *Distinct* skutočnosť, kedy nikdy I nezlyhá ani v jednom z krokov (3) a (7) v odpovedaní na hašovacie požiadavky h a v kroku (6) v odpovedaní na podpisovacie požiadavky. Budeme počítať $Pr[\neg Distinct]$, teda pravdepodobnosť, že algoritmus I niekedy zlyhá. Označme P_3 pravdepodobnosť, že zlyhá v kroku (3) v odpovedaní na podpisovaciu požiadavku, P_7 pravdepodobnosť, že nezlyhá v kroku (3) a zlyhá v kroku (7) v odpovedaní na podpisovaciu požiadavku a P_6 pravdepodobnosť, že zlyhá v kroku (6) v odpovedaní na hašovaciu požiadavku h. Spočítajme jednotlivé pravdepodobnosti.

$$P_6 = \sum_{i=1}^{q_{hash}-1} i \cdot \frac{1}{2^{k_1}} = \frac{1}{2^{k_1}} \cdot \sum_{i=1}^{q_{hash}-1} i = \frac{1}{2^{k_1}} \cdot (q_{hash} - 1) \cdot \frac{q_{hash}}{2}.$$

$$P_3 = \sum_{i=1}^{q_{sig}-1} i \cdot \frac{1}{2^{k_0}} = \frac{1}{2^{k_0}} \cdot \sum_{i=1}^{q_{sig}-1} i = \frac{1}{2^{k_0}} \cdot (q_{sig} - 1) \cdot \frac{q_{sig}}{2}.$$

$$P_7 = (1 - P_3) \cdot \sum_{i=1}^{q_{sig}-1} i \cdot \frac{1}{2^{k_1}} = (1 - P_3) \cdot \frac{1}{2^{k_1}} \cdot (q_{sig} - 1) \cdot \frac{q_{sig}}{2}.$$

Celkovo dostávame, že

$$Pr[\neg Distinct] = \frac{(q_{hash} - 1) \cdot q_{hash}}{2 \cdot 2^{k_1}} + \frac{(q_{sig} - 1) \cdot q_{sig}}{2 \cdot 2^{k_0}} +$$

$$+ \frac{(q_{sig} - 1) \cdot q_{sig}}{2 \cdot 2^{k_1}} - \frac{(q_{sig} - 1)^2 \cdot q_{sig}^2}{2 \cdot 2 \cdot 2^{k_0} \cdot 2^{k_1}}.$$

Odhadneme jednotlivé zlomky a využijeme predpokladu, že ak F urobí požiadavku m na podpis, tak už predtým spravil požiadavku na zahašovanie správy m. Teda $q_{sig} \leq q_{hash}$. Dostaneme

$$Pr[\neg Distinct] \leq \frac{(q_{hash} + q_{sig})^2}{2^{k_1}} + \frac{q_{sig}^2}{2^{k_0}} + \frac{q_{hash}^2}{2^{k_0}} \leq \frac{(q_{hash} + q_{sig})^2}{2^{k_1}} + \frac{(q_{hash} + q_{sig})^2}{2^{k_0}}$$

Potom

$$Pr[\neg Distinct] \leq 2 \cdot (q_{hash}(k) + q_{sig}(k))^2 \cdot (2^{-k_0} + 2^{-k_1}) = p.$$

Teda s pravdepodobnosťou $\epsilon - p$ platí, že algoritmus I nikdy nezlyhá a útočník F vráti platný podvrh. To je rovné pravdepodobnosti, že algoritmus I je úspešný. Čiže

$$\epsilon(k) = \epsilon'(k) + [2 \cdot (q_{hash}(k) + q_{sig}(k))^2] \cdot (2^{-k_0} + 2^{-k_1}).$$

□

Kapitola 5

Záver

Snahou tejto práce bolo predviesť nielen dokázateľnú bezpečnosť podpisovacích schém, ale kvantifikovať bezpečnosť a ukázať, ako veľmi súvisí bezpečnosť podpisovacej schémy s bezpečnosťou permutácie s padajúcimi dvierkami. Pomocou kvantifikácie bezpečnosti obecnej schémy a permutácie sme zistili, že síce bezpečnosť schémy závisí na bezpečnosti permutácie, ale dané odhady sme chceli ešte vylepšiť. Dokázali sme totiž, že aj napriek silnej bezpečnosti permutácie s padajúcimi dvierkami, môže byť bezpečnosť podpisovacej schémy veľmi slabá. Predstavili sme preto novú schému $\mathcal{PSS}$ s upraveným hašovaním na pravdepodobnostné, a tým sme dostali lepšie odhady v bezpečnosti.

Jednotlivé definície, tvrdenia, dôkazy a odhady sme čerpali najmä z článkov [1] a [2], pričom sme doplnili nemalé množstvo vysvetľujúcich komentárov, detailne vysvetlili jednotlivé kroky v dôkazoch, dokončili náznak dôkazu podpisovacej schémy $\mathcal{PSS}$, zjednotili značenie a článok [1] prepísali do obecnej podoby. Vďaka všetkým týmto a mnoho ďalším úpravám sú dôkazy zrozumiteľnejšie a ľahšie pochopiteľnejšie a čitateľ si nemusí sám odvádzať jednotlivé rovnosti či nerovnosti uvedené v tvrdeniach.

Literatúra

[1] M. Bellare, P. Rogaway: *The Exact Security of Digital Signatures: How to Sign with RSA and Rabin*, Advances in Cryptology, Eurocrypt '96, 1996

[2] M. Bellare, P. Rogaway: *Random Oracles are Practical: A Paradigm for Designing Efficient Protocols*, First ACM Conference on Computer and Communications Security 1993

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