O klíčových kolizích v podpisových schématech

Tomáš Rosa

tomas.rosa@i.cz

ICZ a.s., V Olšinách 75, 100 97 Praha 10, tel: (+420 2) 8100 2277

Katedra počítačů, Elektrotechnická fakulta, České vysoké učení technické v Praze
Karlovo náměstí 13, 121 35 Praha 2

Abstrakt

Označme jako (m, S) zprávu m a její podpis S, který je v daném podpisovém schématu považován za platný vzhledem k veřejnému klíči Pub_A. Dvojici různých veřejných klíčů (Pub_A, Pub_B) nazveme *klíčovou kolizí* (dále *k*-kolizí) pokud existuje (m, S) takové, že S je platný podpis zprávy m vzhledem k oběma veřejným klíčům Pub_A i Pub_B. Možnost efektivní konstrukce *k*-kolizí oslabuje zejména kvalitu služeb nepopiratelnosti.

V příspěvku je původním způsobem zavedena a diskutována problematika *k*-kolizí pro podpisová schémata RSA ([15]) dle PKCS#1 ([12]), DSA a ECDSA ([6]) a jsou ukázány způsoby hledání *k*-kolizí. Je uveden základní koncept protiopatření a zmíněny praktické důsledky této problematiky.

Klíčová slova: *k*-kolize, nepopiratelnost, RSA, DSA, ECDSA, TRKG, diskrétní logaritmus

1 Úvod

Zavedená definice říká, že *k*-kolize nastává v případě, kdy je daná zpráva s danou hodnotou podpisu ověřitelná jako správně podepsaná vzhledem k alespoň dvěma různým veřejným klíčům. Existence a možnost cíleného vyvolání *k*-kolizí negativně ovlivňuje zejména kvalitu služby nepopiratelnosti u systémů PKI. Předpokládáme zde obecně uznávanou definici nepopiratelnosti, která volně řečeno praví, že třetí nezávislou stranu lze jednoznačně přesvědčit o tom, zda konkrétní událost v daném systému nastala, respektive nenastala ([9, 10]). S využitím *k*-kolize lze například realizovat následující druh podvodu: Dvě strany se dohodnou, že společně (tj. shodnou hodnotou podpisu) podepíší nějakou smlouvu. Jejího plnění se nakonec ujme ta strana, pro kterou bude smlouva výhodnější. Jsou jistě situace, kdy právě diskutovaná *k*-kolize přinese oběma stranám neoprávněnou (nelegální) výhodu.

Velmi zajímavé jsou i ty situace, kdy lze vyvolat *k*-kolizi bez přímé spolupráce obou signatářů. V takovém případě lze hovořit o *ukradení podpisu* jednoho z nich tím druhým, zatímco v předchozím případě mluvíme o *kooperativním podvodu*. Možnost kradení podpisu může mít negativní důsledky například pro systémy, kde je schéma digitálního podpisu určitým způsobem využito k zajištění autorských práv (pochopitelně s odpovídající nadstavbou). V tomto případě si může uživatel B vygenerovat klíčový pár vykazující *k*-kolizi s veřejným klíčem uživatele A vzhledem k danému podpisu S_A, který se podílí na průkazu jeho autorství (uživatele A). Předpokládáme, že tento podpis je spolu s datovým obrazem předmětu autorství dále podepsán a že právě tento (druhý „stvrzující“) podpis nějakým způsobem autorství dokládá. Uživatel B tak nemůže změnit to, co je již (druhým podpisem) dáno, avšak může stvrzený podpis S_A uživatele A prohlásit díky *k*-kolizi za svůj. Všechny zde prezentované útoky mohou sloužit jak ke kooperativním podvrhům, tak i ke kradení podpisu.

Dále provedeme rozbor *k*-kolizí v teoretické rovině s tím, že jejich dopad na systémy PKI (*Public Key Infrastructure*) je rovněž považován za praktickou certifikační slabinu. Ačkoliv příspěvek přímo nenapadá žádnou konkrétní realizaci PKI, měly by být jeho důsledky zváženy při hodnocení existujících systémů a při projektování systémů nových.

2 Hledání *k*-kolizí ve schématech RSA

V této části se budeme věnovat problematice konstrukce *k*-kolizí v podpisových schématech na bázi algoritmu RSA ([15]). Tento algoritmus je dnes široce používán, zejména v systémech založených na standardech PKCS (viz [13]). Základní techniky pro výpočet podpisu zprávy a pro jeho ověření jsou uvedeny ve standardu PKCS#1 ([12]).

Je všeobecně známo, že nedílnou součástí podpisového schématu na bázi algoritmu RSA je také správně navržený způsob formátování podepisované zprávy, který má za cíl zabránit zejména cílenému vyvolání kolizí

různých zpráv pro jeden veřejný klíč (a jednu hodnotu podpisu) a podvrhům podpisů (přehled viz [10]). Způsob formátování se dále liší v závislosti na tom, zda se RSA používá ve schématech s dodatkem či ve schématech s obnovou zprávy ([10]). V následujícím budeme předpokládat, že je použito schéma s dodatkem, které se v prostředí běžných aplikací vyskytuje častěji. Poznamenejme, že standardy PKCS, stejně jako například zákon [18] a související vyhláška [17], jiný druh schématu ani nespecifikují. Další nároky na vlastnosti použitého formátování klást nebudeme. To znamená, že dále popsaná technika konstrukce *k*-kolizí je použitelná ve většině podpisových schémat (RSA) s doplňkem bez ohledu na konkrétní způsob formátování podepisované zprávy (například podle PKCS#1 přicházejí v úvahu dva druhy formátování). Ve schématech, která i v procesu formátování nějakým způsobem využívají číselných vlastností použitého klíče, se mohou vyskytnout s použitím uvedeného postupu problémy. V případě převládajícího standardu PKCS#1 však takový případ nenastává. Poznamenejme dále, že použitá metoda je velmi snadno rozšiřitelná i pro podpisová schémata s obnovou zprávy. Toto rozšíření plyne přímo z textu a explicitně jej zde neuvádíme z důvodu přehlednosti.

Definice (Cíl útoku na RSA). *Mějme zprávu m a její platný podpis S. Podpis S nechť je vytvořen v instanci RSA s veřejným klíčem (n_A, e_A) uživatele A, kde n_A je modul RSA a e_A veřejný exponent. Cílem je najít odlišný veřejný klíč (n_B, e_B) uživatele B a jemu odpovídající privátní exponent d_B tak, aby platilo $S^{eA} \bmod n_A = S^{eB} \bmod n_B = r$, kde r je obecně nějaké celé číslo, $0 < r < n_A$. Navíc požadujeme, aby platilo $\lceil |n_B|/8 \rceil = \lceil |n_A| /8 \rceil$ a $n_A < n_B$.*

V definici uvedená podmínka na shodné bajtové délky obou modulů zaručí, že útočníkem vytvořená kolize nebude vyloučena na základě porušení pravidel použitého způsobu formátování, kterým je hodnota r dále zpracovávána. Silnější podmínkou by bylo vyžadovat, aby oba moduly měly shodné bitové délky, avšak s ohledem na PKCS#1 toto není nutné. V případě potřeby pro atypický druh formátování lze ovšem tuto podmínku uplatnit bez podstatného zhoršení efektivity celého postupu. Další podmínkou tohoto druhu je, aby platilo $n_A < n_B$. Splnění této podmínky zaručuje, že $S < n_B$ a $r < n_B$.

Postup naplnění definovaného cíle útoku, uvedený dále, navazuje na techniky publikované v práci [3], zejména pak na výsledky dosažené v oblasti výpočtu diskrétních logaritmů v „grupách RSA“ (multiplikativní grupy $\boldsymbol{Z}_n^*$, kde n je příslušný modul RSA). Hledání *k*-kolize rozdělíme do čtyř kroků.

2.1 První krok: Generování modulu n_B

Prvním krokem postupu je vygenerování modulu n_B, pro který platí $n_B = pq$, kde p a q jsou prvočísla speciálních vlastností. Konkrétně požadujeme, aby platilo $p = 2v^{\alpha}+1$ a $q = 2w^{\beta}+1$, kde v a w jsou prvočísla. Takto volená prvočísla indukují faktorové grupy $\boldsymbol{Z}_p^*$ a $\boldsymbol{Z}_q^*$, ve kterých je složitost výpočtu diskrétního logaritmu určena zejména velikostí prvočísel v a w. Předpokládáme, že tato prvočísla budeme volit s ohledem na použití Pohligova-Hellmanova algoritmu pro výpočet diskrétního logaritmu (viz [14]). Konkrétně použijeme variantu popsanou v [10], která má v tomto případě výpočetní složitost $O[\alpha(\alpha^* log_2(v)+log_2(v)+v^{1/2}+1)+2^{1/2}+1]$, respektive $O[\beta(\beta^* log_2(w)+log_2(w)+w^{1/2}+1)+2^{1/2}+1]$. Za dominantního člena ve výrazu pro složitost můžeme považovat druhou odmocninu z prvočísla v, respektive w. S ohledem na obecně uznávanou hranici pro prakticky triviálně schůdnou složitost jako $O(2^{40})$ lze odhadnout, že maximální délka použitých prvočísel by neměla přesáhnout 80 bitů (je třeba přihlédnout také k paměťovým nárokům). Konkrétní bitová délka samozřejmě závisí na konkrétní platformě, která bude pro výpočet použita. Zde uvedenou hodnotu je třeba chápat jako odhad horní meze. Obecně lze říci, že čím větší délku prvočísel zvolíme (nemusí být stejná pro v a w), tím složitější bude výpočet potřebných logaritmů, avšak tím vyšší bude zároveň šance, že se hledaná *k*-kolize najde. Tento závěr vychází z algebraických vlastností řešeného problému a jeho vliv je dobře patrný z pravděpodobností úspěchu, které vypočítáme dále. Poznamenejme, že byl použit Pohligův-Hellmanův algoritmus i přes to, že existuje obecně preferovanější metoda NFS (případně GNFS), která rovněž vykazuje prakticky zvládnutelnou složitost pro speciální druhy prvočísel (viz [7]). Důvody naší volby spočívají jednak ve větší průhlednosti celého výpočtu, jednak také v tom, že zde narozdíl od metody NFS nevzniká podstatné omezení na délku modulu. U NFS byla v roce 1992 přijímána maximální délka prvočísel upravených pro efektivní řešení problému diskrétního logaritmu řádově 600 bitů (viz [4]). I přes jistý teoretický a technologický pokrok v této oblasti by se mohl tento limit již prakticky projevit při snaze o hledání *k*-kolizí pro větší délky modulů (například 2048 bitů). V našem případě počítáme se schůdností výpočtu pro běžné moduly v délce řádově tisíců bitů.

Vlastní postup generování modulu n_B sestává z opakovaného generování prvočísla v, respektive w a testování, zda hodnota $2v^{\alpha}+1$, respektive $2w^{\beta}+1$, je také prvočíslo. Prvočísla v a w musí být generována různá, abychom mohli ve 3. kroku využít Čínskou větu o zbytku (CRT). Mocninu α, respektive β, volíme tak, aby prvočísla p a q měly požadovanou délku. Tento krok končí nalezením modulu n_B splňujícího výše stanovené podmínky.

2.2 Druhý krok: Výpočet exponentů e_p, e_q

Cílem tohoto kroku je vyřešit následující kongruence:

$r \equiv S^{ep} \pmod{p}$

$r \equiv S^{eq} \pmod{q}$

Exponenty e_p a e_q jsou pomocné hodnoty, které využijeme v dalším kroku k výpočtu hledaného veřejného exponentu e_B. Řešení každé z uvedených kongruencí pro příslušné exponenty lze převést na výpočet diskrétního logaritmu v určité cyklické podgrupě faktorové grupy $\boldsymbol{Z}_p^*$, respektive $\boldsymbol{Z}_q^*$. Nejprve určíme, jaký řád bude příslušná podgrupa mít. Zabývejme se nejprve grupou $\boldsymbol{Z}_p^*$. Připomeňme, že pro řád grupy $\boldsymbol{Z}_p^*$ platí $ord(\boldsymbol{Z}_p^*) = p-1$. Vzhledem k tomu, že známe rozklad tohoto řádu, $p-1 = 2v^{\alpha}$, můžeme triviálně určit $ord_p(S)$, což je řád prvku S v grupě $\boldsymbol{Z}_p^*$, neboť $ord_p(S) | (p-1)$. Prvek S můžeme považovat za generátor cyklické podgrupy $\boldsymbol{A}$ řádu $ord_p(S)$, kde $\boldsymbol{A} =^{def} \{ [S^x]_p : 0 \leq x < ord_p(S) \}$, $\boldsymbol{A} \subseteq \boldsymbol{Z}_p^*$, přičemž $[y]_p$ označuje třídu prvku y vzhledem k ekvivalenci kongruence mod p (někdy se symbol $[\ ]_p$ vynechává a pokládá se za implicitně zřejmý). Z teorie grup víme, že podgrupa o řádu $ord_p(S)$ je v grupě $\boldsymbol{Z}_p^*$ právě jedna. Bez újmy na obecnosti ji tak můžeme definovat s použitím prvku S jako generátoru. V případě grupy $\boldsymbol{Z}_q^*$ lze postupovat zcela analogicky s tím, že se dopracujeme k řádu $ord_q(S)$ a podgrupě $\boldsymbol{B}$, $\boldsymbol{B} \subseteq \boldsymbol{Z}_q^*$.

Dále výpočet pokračuje nasazením Pohligova-Hellmanova algoritmu (dle [10]) k výpočtu e_p, respektive e_q, jako diskrétního logaritmu hodnoty r o základu S v podgrupě $\boldsymbol{A}$, respektive $\boldsymbol{B}$. K tomu, aby se nám oba logaritmy podařilo najít, musí současně platit, že $[r]_p \in \boldsymbol{A}$ a $[r]_q \in \boldsymbol{B}$. Pokud jeden nebo oba vztahy neplatí, musíme se vrátit ke kroku 1 a provést generování jednoho nebo obou faktorů znovu.

Postup v tomto kroku končí nalezením hodnot e_p a e_q.

2.3 Třetí krok: Výpočet exponentu e_B

Exponent e_B hledáme jako řešení následující soustavy kongruencí:

$S^{eB} \equiv S^{ep} \pmod{p}$

$S^{eB} \equiv S^{eq} \pmod{q}$

S využitím řádů $ord_p(S)$ a $ord_q(S)$ vypočtených v předchozím kroku můžeme tuto soustavu přepsat jako:

$e_B \equiv e_p \pmod{ord_p(S)}$

$e_B \equiv e_q \pmod{ord_q(S)}$

Naším cílem je zde použít Čínskou větu o zbytku, konkrétně Gaussův algoritmus (viz [10]), pro nalezení řešení e_B modulo $lcm(ord_p(S), ord_q(S))$. Víme, že pro řád prvku S v grupě $\boldsymbol{Z}_{nB}^*$ platí $ord_{\boldsymbol{Z}_{nB}^*}(S) = lcm(ord_p(S), ord_q(S))$. Odtud plyne, že získané řešení uvedené soustavy je hledaným veřejným exponentem. Nutnou a postačující podmínkou pro existenci řešení je vztah $e_p \equiv e_q \pmod{b}$, kde $b = gcd(ord_p(S), ord_q(S))$. V našem případě platí $b \leq 2$, konkrétní varianty budou rozebrány zvlášť.

2.3.1 Případ b = 1

V tomto případě lze přímo použít Gaussův algoritmus a určit hodnotu e_B.

2.3.2 Případ b = 2

V tomto případě musí platit $e_p \equiv e_q \pmod{2}$. Díky způsobu, jakým jsme generovali prvočísla p a q, bude tato podmínka zhruba s pravděpodobností 1/2 splněna. V případě, že podmínka splněna nebude, vrací se výpočet do kroku 1, kde bude vygenerován nový modul n_B.

Pokud podmínka splněna je, lze již rovněž snadno aplikovat Gaussův algoritmus. Jeho vstupem je soustava kongruencí vzhledem ke vzájemně nesoudělným modulům. Proto naší soustavu upravíme na tvar

$e_B \equiv e_p \pmod{ord_p(S)/2}$

$e_B \equiv e_q \pmod{ord_q(S)/2}$

$e_B \equiv e_p \pmod{2}$

a jejím řešením obdržíme hledaný veřejný exponent e_B.

2.4 Čtvrtý krok: Určení privátního exponentu d_B

V tomto kroku dopočítáme privátní exponent d_B. Poznamenejme, že v našem případě platí $lcm(p-1, q-1) = 2v^{\alpha}w^{\beta}$, proto můžeme psát:

$$e_B * d_B \equiv 1 \pmod{2v^{\alpha}w^{\beta}}$$

Nutnou a postačující podmínkou k existenci privátního exponentu je $gcd(e_B, 2v^{\alpha}w^{\beta}) = 1$. Pokud tato podmínka není splněna, vrací se výpočet do kroku 1. V případě splnění této podmínky je rozšířeným Euklidovým algoritmem ([10]) dopočítán privátní exponent d_B a celý postup je tímto ukončen.

2.5 Odhady složitosti a úspěšnosti

Z hlediska složitosti celého postupu zde dominuje výpočet diskrétních logaritmů v druhém kroku. Jak jsme uvedli výše, je tato složitost určena v zásadě hodnotou $v^{1/2}$, respektive $w^{1/2}$. Podle toho, jaký výkon použitá výpočetní platforma poskytuje, se předpokládá volba konkrétní délky generovaných prvočísel. Délka 80 bitů by měla odpovídat současné hranici praktické schůdnosti.

V průběhu hledání kolidujícího klíčového páru se vyskytují celkem tři podmínky (v krocích 2, 3 a 4), jejichž nesplnění vrací výpočet zpět do kroku 1. S ohledem na efektivitu celého postupu zde učiníme základní odhady pravděpodobností, že jednotlivé podmínky budou splněny (respektive nebudou). Z toho potom odvodíme odhad celkové pravděpodobnosti (P), že výpočet proběhne bez jediného návratu do kroku 1. Odtud pak lze usuzovat na skutečný počet návratů do kroku 1, než dojde k naplnění cíle útoku.

Označme P_1 pravděpodobnost, že v kroku 2 platí $[r]_p \in \boldsymbol{A}$ a P_2 pravděpodobnost, že v kroku 2 platí $[r]_q \in \boldsymbol{B}$. Při odhadu těchto pravděpodobností vyjdeme z přístupu použitého v [3]. Výpočet provedeme nejprve pro P_1. Nejdříve stanovíme pravděpodobnost, že podgrupa generovaná pevně zvoleným prvkem x, $x \in \boldsymbol{Z}_p^*$, obsahuje náhodně zvolený nenulový prvek y, $y \in \boldsymbol{Z}_p^*$. S využitím řádu prvku x můžeme psát:

$$P(cover_x) = ord_p(x) / (p-1)$$

Hledanou pravděpodobnost P_1 pak můžeme odhadnout jako průměrnou hodnotu pravděpodobnosti $P(cover_x)$ počítanou přes všechny nenulové prvky grupy $\boldsymbol{Z}_p^*$, což odráží průměrování přes možné hodnoty S.

$$P_1 \approx [\sum_{i=1}^{p-1} P(cover_i)]/(p-1) = [\sum_{i=1}^{p-1} ord_p(i)]/(p-1)^2 = [\sum_{(d|(p-1))} d*\phi(d)]/(p-1)^2$$

Při úpravě tohoto výrazu jsme využili toho, že počet prvků řádu d je roven hodnotě Eulerovy funkce $\phi(d)$. S využitím definovaného tvaru prvočísla $p = 2v^{\alpha}+1$ lze výraz dále upravit na:

$$P_1 \approx 3(3v+v^{-2\alpha}) / 4(1+v)$$

Pro dostatečně velká v (řádově desítky bitů a více) se P_1 limitně blíží k hodnotě 3/4. Analogicky odhadneme P_2, takže můžeme hledání odhadu uzavřít jako:

$$P_1 \approx 3/4,\ P_2 \approx 3/4$$

Dále označme P_3 pravděpodobnost, že v kroku 3 platí $b = 2$ a není splněna podmínka $e_p \equiv e_q \pmod 2$ pro řešitelnost uvedené soustavy kongruencí. Tuto pravděpodobnost vyjádříme jako $P_3 = P_{3,1} * P_{3,2}$, kde $P_{3,1}$ je pravděpodobnost, že $b = 2$ a $P_{3,2}$ je pravděpodobnost, že podmínka řešitelnosti není splněna. Odhad $P_{3,2}$ lze provést jako $P_{3,2} \approx 1/2$. K odhadu $P_{3,1}$ si stačí uvědomit, jaké řády prvků přicházejí v obou grupách v úvahu: Pro $\boldsymbol{Z}_p^*$ jsou to řády 1, 2, v^i a $2v^i$, pro $0 < i \leq \alpha$. Pro $\boldsymbol{Z}_q^*$ jsou to analogicky řády 1, 2, w^i a $2w^i$, pro $0 < i \leq \beta$. Celkem tak v $\boldsymbol{Z}_p^*$, respektive v $\boldsymbol{Z}_q^*$ existuje $1+\sum_{i=1}^{\alpha}\phi(v^i)$, respektive $1+\sum_{i=1}^{\beta}\phi(w^i)$ prvků lichého řádu a $1+\sum_{i=1}^{\alpha}\phi(2v^i) = 1+\sum_{i=1}^{\alpha}\phi(v^i)$, respektive $1+\sum_{i=1}^{\beta}\phi(2w^i) = 1+\sum_{i=1}^{\beta}\phi(w^i)$ prvků sudého řádu. Odtud lze odvodit, že náhodně zvolený prvek bude mít v $\boldsymbol{Z}_p^*$, respektive v $\boldsymbol{Z}_q^*$ sudý řád s pravděpodobností 1/2. Pravděpodobnost $P_{3,1}$ je pravděpodobnost toho, že S bude mít sudý řád v obou grupách, proto lze (bez uvažování závislostí) odhadnout $P_{3,1} \approx 1/4$. Odtud pak $P_3 \approx 1/8$.

Zbývá nám ještě odhadnout pravděpodobnost P_4, že k vypočtenému veřejnému exponentu e_B nebude existovat privátní exponent d_B. Nutnou a postačující podmínkou neexistence d_B je zde platnost jednoho ze vztahů: $2|e_B$, $v|e_B$ nebo $w|e_B$. Odtud můžeme P_4 s ohledem na typ úlohy odhadnout jako $P_4 \approx 1/2+v^{-1}+w^{-1}$. Pro běžné hodnoty v a w pak $P_4 \approx 1/2$.

Pro výslednou pravděpodobnost P dostáváme pro naše potřeby dostatečně přesný odhad $P \approx P_1*P_2*(1-P_3)*(1-P_4) \approx 3/4*3/4*7/8*1/2 = 63/256 = 0.246094$. S pravděpodobností zhruba 25 procent tak projde celý výpočet bez návratu do kroku 1. Odtud lze usuzovat, že v praxi dojde před splněním cíle zhruba k jednotkám návratů do kroku 1. Celkově lze tedy takto sestavený postup považovat v běžných podmínkách za triviálně schůdný. Rozbor

schůdnosti pro použitý tvar prvočísel p a q je zároveň rozšířením teoretických závěrů v práci [3]. Ukázali jsme, že odhadnutá pravděpodobnost pro nutnost návratu do kroku 1 je stejná jako pravděpodobnost odhadnutá v [3] pro takzvaná *prvočísla Germainové* (odpovídá použitému tvaru p, q pro $\alpha = \beta = 1$). Námi použitá prvočísla (jejichž vhodnost je v [3] zmíněna, avšak ne prokázána) navíc narozdíl od prvočísel Germainové vedou i při velkých délkách (stovky až tisíce bitů) k výpočetně stále zvládnutelným instancím problému diskrétního logaritmu, což je pro uvedený postup podstatné.

2.6 Poznámky

První poznámka se týká rozsahu hodnot exponentu e_B. Z postupu je patrné, že tento exponent může nabývat hodnot z intervalu $<0, lcm(ord_p(S), ord_q(S))>$. Ačkoliv tento rozsah není v rozporu se standardem PKCS#1, některé implementace RSA zavádějí vlastní omezení pro délku veřejných exponentů. Například subsystém CryptoAPI na platformě operačních systémů Microsoft Windows povoluje nevýše 32bitové délky veřejného exponentu (viz [11]). V takovýchto podmínkách útok s vysokou pravděpodobností neuspěje. Hledání výpočetně schůdného postupu vedoucího k nalezení *k*-kolizí pro veřejné exponenty s takto silně omezenou délkou je zatím otevřeným problémem, jehož řešení již nelze postavit pouze na myšlence schůdných instancí problému diskrétního logaritmu. Do doby, než bude v tomto směru učiněn nějaký pokrok, lze tak omezení délky veřejného exponentu považovat zároveň za opatření proti popsanému útoku.

Dále je důležité si uvědomit, že námi navržený postup hledání *k*-kolizí nevyžaduje žádnou speciální spolupráci s majitelem kolidujícího klíče (n_A, e_A), takže může sloužit jak ke kooperativním podvodům, tak i ke kradení podpisu uživatele A uživatelem B.

3 Hledání *k*-kolizí pro DSA

V této části představíme výpočetně a algebraicky velmi jednoduchý, avšak o to více efektní postup k nalezení klíčové kolize ve schématu DSA ([6]). Naším cílem je najít takovou instanci DSA, která bude na daném podpisu kolidovat s jinou (danou) instancí DSA, bude standardně použitelná a její původ bude možné věrohodně podložit. Začneme proto následující definicí.

Definice (Korektní instance DSA). *Instanci DSA, tvořenou veřejnými parametry (p, q, g), veřejným klíčem y a privátním klíčem x, budeme považovat za korektní, pokud budou splněny následující podmínky:*

- p *a* q *jsou prvočísla*
- $2^{L-1} < p < 2^L$, *pro nějaké* L, $L = 512 + 64j$, *kde* j *je celé číslo splňující* $0 \leq j \leq 8$
- $2^{159} < q < 2^{160}$, $q \mid (p-1)$
- $g^q \bmod p = 1$
- $g^x \bmod p = y$

Předpokládáme, že tyto podmínky jsou ověřovány při žádosti o registraci veřejného klíče y (poslední podmínku lze ovšem ověřit jen nepřímo). Podmínky byly stanoveny s ohledem na aplikaci *principu robustnosti* zavedeného v [1].

Definice (Cíl útoku na DSA). *Mějme dánu dvojici (m, S) představující zprávu m a podpis S = (r, s), který je platný v korektní instanci DSA* (p_A, q_A, g_A, y_A, x_A) *uživatele A. Cílem útoku je nalézt korektní instanci DSA* (p_B, q_B, g_B, y_B, x_B) *uživatele B splňující podmínku* $y_B \neq y_A$, *ve které je S rovněž platným podpisem zprávy m.*

3.1 Postup útoku

Položme $p_B = p_A = p$, $q_B = q_A = q$. Tímto je zaručena korektnost parametrů p_B a q_B, aniž by tento krok byl v rozporu se zásadami doporučenými standardem [6]. Naopak, schéma DSA je navrženo tak, aby tyto parametry mohly být sdíleny několika uživateli. Útočník tak může argumentovat tím, že chtěl použít již jednou kvalitně vygenerovaná prvočísla. Proti tomuto argumentu nelze v zásadě nic namítat, zejména pokud jsou používány certifikáty pro původ těchto prvočísel tak, jak je definuje standard [6]. Je zde poněkud paradoxní, že tyto certifikáty mají právě zaručit, že daná prvočísla nebude možné využít k přípravě útoků na daný podpisový systém.

Dále označme $h = SHA\text{-}1(m)$, kde SHA-1 je hašovací funkce definovaná standardem [5], která je implicitně použita ve schématu DSA. Z definice tohoto schématu víme, že pro zadanou dvojici (m, S), kde $S = (r, s)$ platí:

$r = (g_A^{kA} \bmod p) \bmod q$, kde k_A je náhodné číslo, $0 < k_A < q$

$s = (h + rx_A)k_A^{-1} \bmod q$, kde $k_A k_A^{-1} \equiv 1 \pmod q$

Na základě znalosti (m, S), veřejných parametrů a veřejného klíče uživatele A nyní vypočteme hodnotu:

$\alpha = g_A^{wh} y_A^{wr} \bmod p$, kde $w*s \equiv 1 \pmod q$

Nyní zvolíme náhodně číslo k_B, $1 < k_B < q$, a vypočteme jeho inverzi z jako $z*k_B \equiv 1 \pmod q$. Na základě získané hodnoty položíme:

$g_B = \alpha^z \bmod p$

Víme, že $gcd(z, q) = 1$, takže $ord(g_B) = ord(g_A) = q$, a proto hodnota g_B je korektní veřejný parametr DSA. Povšimněme si, že $\alpha = g_A^{kA} \bmod p$, takže platí:

$$(g_B^{kB} \bmod p) \bmod q = (\alpha^{zkB} \bmod p) \bmod q = (g_A^{zkBkA} \bmod p) \bmod q = (g_A^{kA} \bmod p) \bmod q = r$$

Bylo tedy dosaženo kolize pro popisový parametr r a navíc známe příslušný exponent (k_B) produkující tuto hodnotu. Nyní již zbývá jen určit privátní a veřejný klíč, což provedeme podle následujících vztahů:

$x_B = t(k_B s - h) \bmod q$, kde $r*t \equiv 1 \pmod q$

$y_B = g_B^{xB} \bmod p$

Takto vytvořená instance DSA je korektní a dvojice $S = (r, s)$ je v ní platným podpisem zprávy m. Platnost podpisu snadno ověříme: Z definice schématu DSA víme, že podpis $S = (r, s)$ bude uznán jako platný podpis zprávy m tehdy, když platí:

$(g_B^{wh} y_B^{wr} \bmod p) \bmod q = r$, kde $w*s \equiv 1 \pmod q$, viz [6]

V našem případě snadno ověříme, že:

$$(g_B^{wh} y_B^{wr} \bmod p) \bmod q = (g_B^{w(h+rxB)} \bmod p) \bmod q = (g_B^{wskB} \bmod p) \bmod q = (g_B^{kB} \bmod p) \bmod q = r$$

Zbývá ještě ukázat, že nalezená instance (p_B, q_B, g_B, y_B, x_B) s velkou pravděpodobností splňuje stanovenou podmínku na odlišnost. Nejprve vyjádříme vztah mezi privátními klíči x_A a x_B. Položme $\beta \equiv k_A k_B^{-1} \pmod q$ a poznamenejme, že potom $g_B \equiv g_A^{\beta} \pmod p$. S využitím výše odvozených vztahů můžeme psát:

$x_B \equiv ht(\beta^{-1} - 1) + \beta^{-1} x_A \pmod q$

Z tohoto vztahu je mimo jiné vidět, že ani jeden z uživatelů nemůže odvodit hodnotu privátního klíče toho druhého, neboť ani jeden z nich nezná přímo hodnotu „separátoru“ β. Pokud platí, že DSA nelze prolomit, potom jsou vyloučeny vzájemné útoky na privátní klíče mezi uživateli A a B. Pro vztah mezi veřejnými klíči y_A a y_B lze zase odvodit:

$y_B \equiv g_A^{ht(1-\beta)} y_A \pmod p$, kde $r*t \equiv 1 \pmod q$

Víme, že $gcd(q, t) = 1$ a q je prvočíslo. Pokud by mělo platit, že $y_B \equiv y_A \pmod p$, potom by buď muselo být $\beta \equiv 1 \pmod q$, nebo $q|h$. Odtud plyne, že uvedený postup vede téměř jistě k nalezení instance DSA s různou hodnotou veřejného klíče, čímž splňuje stanovený cíl.

3.2 Schůdnost a vlastnosti útoku

Ve výše uvedeném postupu jsme použili pouze takové algebraické operace, které se při praktickém používání DSA běžně provádějí. Náročnost provedení útoku tak můžeme považovat za zanedbatelnou a celý postup označit za triviálně schůdný na běžném kancelářském PC. Popsaný útok rovněž nevyžaduje žádnou zvláštní spolupráci s napadeným uživatelem A, takže může sloužit jak ke kooperativním podvrhům, tak i ke kradení podpisu.

4 Hledání *k*-kolizí pro ECDSA

Předchozí útok využívá natolik obecné algebraické vlastnosti schématu DSA, že jej lze snadno rozšířit i na systém ECDSA, který z DSA s ohledem na tyto vlastnosti přímo vychází. Z důvodu přehlednosti zde nebudeme znovu provádět podrobný rozbor postupu, pouze si poukážeme na jeho základní kroky. V použité notaci budeme vycházet z dokumentu [8].

Definice (Cíl útoku na ECDSA). *Mějme dánu dvojici (m, S) představující zprávu m a její platný podpis S = (r, s) v korektní instanci ECDSA ($E(\mathbf{F}_q)$, G_A, Q_A, d_A) uživatele A. Cílem útoku je nalézt korektní instanci ECDSA ($E(\mathbf{F}_q)$, G_B, Q_B, d_B) uživatele B splňující podmínku $Q_B \neq Q_A$.*

Základní způsoby ověření korektnosti příslušné instance ECDSA jsou popsány v [8]. V našem případě jsme použili zkrácený zápis instance, kde $E(\boldsymbol{F}_q)$ představuje množinu bodů eliptické křivky E nad konečným tělesem $\boldsymbol{F}_q$ společně s dodefinovaným bodem v nekonečnu $\boldsymbol{O}$. Množina $E(\boldsymbol{F}_q)$ tvoří aditivní Abelovu grupu ($\boldsymbol{O}$ zde vystupuje jako nulový prvek) řádu $\#E(\boldsymbol{F}_q)$, přičemž algoritmus ECDSA využívá její cyklickou podgrupu o vysokém prvočíselném řádu. Poznamenejme, že v případě ECDSA se ještě častěji nežli v případě DSA daná křivka používá současně ve více systémech (viz seznam doporučených křivek v [6]). Proto jsme cíl útoku již přímo definovali s tím, že uživatel B v roli útočníka použije stejnou křivku. Dále jsme v zápisu instance ECDSA použili označení G pro generátor cyklické podgrupy o velkém prvočíselném řádu n, $n|\#E(\boldsymbol{F}_q)$, symbol Q pro veřejný klíč a symbol d jako označení privátního klíče. V korektní instanci ECDSA platí $Q = [d]G$, $Q \in E(\boldsymbol{F}_q)$, $G \in E(\boldsymbol{F}_q)$, $0 < d < n$. Operace $[d]G$ značí d-krát iterovaný součet bodu G. Konkrétní počet provedených součtů přitom s ohledem na řád generátoru G odpovídá libovolnému reprezentantu třídy ekvivalence $[d]_n$ vzhledem ke kongruenci modulo n (při praktickém výpočtu samozřejmě použijeme reprezentanta d, $d < n$). Argumentace bezpečnosti ECDSA se opírá o složitost úlohy ECDLP – obdoba problému diskrétního logaritmu, zde pro eliptické křivky, která zde spočívá v nalezení celého čísla d splňujícího $Q = [d]G$. Pro správně generovanou grupu $E(\boldsymbol{F}_q)$ se tento problém považuje za výpočetně neschůdný.

4.1 Postup útoku

Konkrétní postup začíná opět nalezením vhodného generátoru G_B. K tomu účelu zvolíme náhodně číslo k_B, $1 < k_B < n$, a vypočteme jeho inverzi k_B^{-1} jako $k_B k_B^{-1} \equiv 1 \pmod n$. Na základě získané hodnoty položíme:

$$G_B = [k_B^{-1}]X, \text{ kde } X = [hs^{-1}]G_A + [rs^{-1}]Q_A, \text{ pro } ss^{-1} \equiv 1 \pmod n \text{ a } h = SHA\text{-}1(m)$$

Víme ([8]), že pro hodnoty $(r, s) = S$ ve schématu ECDSA platí:

$$r = x_1 \bmod n, \text{ pro } X = (x_1, y_1), X = [k_A]G_A, \text{ kde } k_A \text{ je náhodné číslo, } 0 < k_A < n$$

$$s = (h + rd_A)k_A^{-1} \bmod n, \text{ kde } k_A k_A^{-1} \equiv 1 \pmod n$$

Proto $G_B = [\beta]G_A$ pro $\beta \equiv k_A k_B^{-1} \pmod n$, obdobně, jako jsme v případě DSA měli vztah $g_B \equiv g_A^{\beta} \pmod p$ pro $\beta \equiv k_A k_B^{-1} \pmod q$. Analogicky opět platí, že $ord_{E(\boldsymbol{F}q)}(G_B) = n$, takže G_B je korektní generátor stejné cyklické podgrupy, jaká je generována generátorem G_A.

S využitím vztahu pro s odvodíme privátní klíč d_B a veřejný klíč Q_B jako:

$$d_B = r^{-1}(k_B s - h) \bmod n, \text{ kde } r^{-1}r \equiv 1 \pmod n$$

$$Q_B = [d_B]G_B$$

Obdobně jako v případě DSA lze ukázat, že v takto získané instanci ECDSA představuje dvojice (m, S), S = (r, s) korektně podepsanou zprávu (pro přehlednost zde tento jednoduchý krok vynecháme). Nyní se podívejme na vztahy mezi instancemi ECDSA uživatelů A a B. Vzhledem k nezávislosti na struktuře použité podgrupy je vztah mezi privátními klíči uživatelů A a B až na použité značení stejný jako v případě DSA:

$$d_B \equiv hr^{-1}(\beta^{-1} - 1) + \beta^{-1}d_A \pmod n, \text{ kde } r^{-1}r \equiv 1 \pmod n \text{ a } \beta \equiv k_A k_B^{-1} \pmod n$$

Odtud pak pro veřejný klíč Q_B platí:

$$Q_B = [hr^{-1}(1\text{-}\beta)]G_A + Q_A$$

K tomu, aby platilo $Q_B = Q_A$ musí být $[hr^{-1}(1\text{-}\beta)]G_A = \boldsymbol{O}$. To zde nastává v případě, kdy $\beta \equiv 1 \pmod n$ nebo $n|h$. Proto s vysokou pravděpodobností platí, že $Q_B \neq Q_A$ a zvolený cíl útoku je splněn. Separátor β navíc obě instance díky neschůdnosti problému ECDLP dostatečně odděluje.

4.2 Schůdnost a vlastnosti útoku

Stejně jako v případě DSA se jedná o útok, který je triviálně schůdný na běžném kancelářském PC a může sloužit jak ke kooperativním útokům, tak i ke kradení podpisu. Poznamenejme, že přímočarost, s jakou bylo možné rozšířit původní útok na DSA, ukazuje, že ECDSA nemusí být za všech okolností univerzálně lepší alternativou k DSA (což se někdy nesprávně naznačuje). Naopak je logické očekávat, že útoky založené na velmi obecných algebraických vlastnostech DSA budou s velkou pravděpodobností rozšiřitelné i na ECDSA.

5 Protiopatření (koncept *TRKG*)

Základním předpokladem pro úspěšnost předložených útoků je, aby uživatel B (v roli útočníka) mohl libovolně generovat položky tvořící instanci jeho podpisového schématu a to způsobem, který mu poskytuje značnou

volnost, a který zároveň třetí nezávislá strana nemůže jednoznačně zpochybnit (předpokládáme, že nezávislá strana může pracovat pouze s veřejnými informacemi – to je plně v souladu s praktickým chápáním tohoto subjektu, viz [9, 10]). Zamezením této možnosti hrozba předložených útoků zaniká. Podíváme-li se na celou věc z pohledu požadavku na zajištění služby nepopiratelnosti, tak zjistíme, že se v podstatě jedná o prosté tranzitivní rozšíření z nepopiratelnosti podpisu na nepopiratelnost důvěryhodného generování klíčů. Jedná se o přímou analogii tranzitivity známé například z oblasti symetrického šifrování, kde se požadavek na utajení zprávy přenáší na požadavek utajení klíče.

Univerzálním východiskem je zavedení a prosazení důsledného používání důvěryhodného generování klíčů (kam zařadíme i veřejné parametry, pokud jsou tyto spolu s klíči generovány), které jsme nazvali *tamper resistant key generation* (*TRKG*). Symbolicky můžeme za hlavní součásti systému *TRKG* označit procedury *GenKey* a *VerifyKey*, jejichž činnost lze vyjádřit jako:

GenKey: (SEED) $\rightarrow$ *(PublicKey, PrivateKey, PublicParameters,* ***Witness****)*

VerifyKey: (PublicParameters, PublicKey, ***Witness****)* $\rightarrow$ *("OK"/"FALSE")*

Názvem *SEED* v argumentu funkce *GenKey* označujeme náhodný počáteční stav, který do procesu generování vnáší požadovanou entropii. Důležitým novým prvkem je zde položka *Witness*, která má později umožnit třetí nezávislé straně pomocí procedury *VerifyKey* ověřit, že daná instance byla skutečně vygenerována procesem *TRKG*. Jedná se tedy o průkaz nepopiratelnosti použití mechanizmu *TRKG*. Hlavní požadavky na tento proces jsou přitom následující:

1. Generované instance jsou kryptograficky silné s ohledem na všechny známé útoky.
2. Procedura *GenKey* je vzhledem ke vstupním parametrům jednosměrná, uživatel nemůže změnou vstupů dosáhnout zvolené změny výstupu.
3. Jakákoliv změna procedury *GenKey*, která by vedla k porušení bodu (1) nebo (2), je detekovatelná procedurou *VerifyKey*.

Způsoby konstrukce robustních *TRKG* mechanizmů jsou tématem na samostatný výzkum. Cílem tohoto příspěvku je zejména podpořit význam a nutnost studia této problematiky. Proto se zde omezíme na nástin dalších možných postupů v této oblasti.

5.1 *TRKG* pro DSA a ECDSA

V případě těchto algoritmů již bylo jisté úsilí standardizační institucí NIST vyvinuto, avšak z pohledu předvedených útoků jej lze považovat za nedostatečné. Ve standardu [6] (a jím odkázaných dokumentech) je pozornost soustředěna zejména na prokazatelnost důvěryhodného generování prvočísel p, q v případě DSA, či eliptické křivky *E* v případě ECDSA. Zcela mimo pozornost však v tomto směru v obou případech stojí generátor použité podgrupy. V případě DSA na tento problém bylo již jednou upozorněno ([16]), tento příspěvek na něj důrazně upozorňuje znovu. Navíc ukazujeme, že tento problém se týká i schématu ECDSA.

Logickým východiskem se zdá být rozšíření mechanizmu důvěryhodného generování veřejných parametrů DSA či ECDSA i na zmíněný generátor. Tímto krokem by byly oba prezentované útoky znemožněny. Navíc v systémech, kde se mechanizmus založený na jednosměrném generování veřejných parametrů (viz [6, 8]) již používá, by neměl být problém toto rozšíření provést. Bohužel však využívání tohoto způsobu není v praxi příliš podporováno, proto tento příspěvek nepředstavuje pouze motivaci pro rozšíření, ale i pro skutečné reálné praktické nasazení.

5.2 *TRKG* pro RSA

V případě RSA jsme postaveni před poměrně komplikovaný problém: Na jedné straně požadujeme, aby byla dostatečně chráněna informace o faktorizaci modulu n (abychom poctivé uživatele zbytečně neohrožovali), na druhé straně však chceme, aby byla informace o použitých prvočíslech ověřitelná na základě veřejné informace. Otázka, jak tyto požadavky naplnit, je zatím otevřeným problémem. Otevřeným problémem také zůstává, zda by v případě RSA postačovalo použít *TRKG* metodu pouze na generování veřejného exponentu. Zde by totiž bylo možné použít stejné principy, jaké se používají v případě (EC)DSA: Používat generátor pseudonáhodných čísel a jako svědka (*Witness*) použít startovní hodnotu tohoto generátoru (ta může být generována s pomocí fyzikálního RNG). Poznamenejme také, že s ohledem na prezentovaný útok se zatím jako dostatečné protiopatření jeví omezení délky veřejného exponentu (například na 32 bitů). Zcela mimo nebezpečí jsou pak zatím systémy, které hodnotu veřejného exponentu definují pevně (například často používaná hodnota *F4 = 65537*).

6 Závěr

V příspěvku byla otevřena problematika klíčových kolizí (*k*-kolizí) v podpisových schématech RSA, DSA a ECDSA. Jedná se o teoretickou možnost útoku na vlastnost nepopiratelnosti. Z praktického hlediska se v současnosti jedná o certifikační slabinu, která by však mohla časem vyústit v reálnou praktickou hrozbu. Jedná se zde přitom zejména o důvěru běžných uživatelů v bezpečnost elektronického podepisování. S ohledem na tato rizika je jistě vhodné mít problematiku *k*-kolizí při návrhu systémů elektronického podpisu na zřeteli. Za hlavní protiopatření na úrovni elementárních kryptografických mechanizmů považujeme použití vhodného *TRKG* mechanizmu, jehož základní koncept zde byl formulován. Další přechodná opatření je možné učinit v protokolech vyšších vrstev. Jedná se zejména o systémy využívající princip „podpis-podpisu“, jako jsou například některé druhy služeb časových razítek či notářské služby. Společným znakem zde je, že dochází k podpisu již podepsaného dokumentu a to včetně jeho původního podpisu. V takovém případě je s ohledem na riziko pozdějšího útoku prostřednictvím *k*-kolizí vhodné zahrnout do podepisovaných dat také certifikát původního signatáře. V praxi existuje také možnost přenést zodpovědnost za důvěryhodnost vygenerovaných klíčů do prostředí certifikační autority a požadovat, aby uživatelé využívali v tomto směru výhradně jejích služeb. Tento přístup sice nelze z čistě teoretického hlediska považovat za úplné vyřešení problému, nicméně v praktických situacích, kdy se víceméně stejně vyžaduje vysoká míra důvěry uživatelů v danou certifikační autoritu, jej můžeme považovat za akceptovatelný.

Z obecného kryptologického hlediska lze problematiku *k*-kolizí chápat jako upozornění na často přehlíženou nuanci, která odlišuje útoky na nepopiratelnost od jiných útoků. Zde je totiž často útočníkem sám majitel privátního klíče, tedy subjekt, který obvykle hraje v ostatních modelech roli oběti. Většina protiopatření se zaměřuje zejména na to, aby tento subjekt nebyl svým okolím nějak poškozen ([2]), přičemž jeho vlastnímu počínání je již věnována velmi malá pozornost. Tento příspěvek poukazuje na hrozby, které z tohoto přístupu plynou. Ukazujeme, že ani sám uživatel by neměl mít přílišnou volnost ve volbě svých vlastních klíčů, pokud jde o stanovení jejich konkrétní hodnoty. Základní apel celého příspěvku pak zní: Věnovat otázkám nepopiratelnosti více pozornosti a nepovažovat ji jen za samozřejmou vlastnost podpisových schémat. Ukázali jsme, že nejčastěji používané kryptografické mechanizmy totiž samy o sobě nepopiratelnost nezaručují. Lze proto předpokládat, že bez odpovídající pozornosti při návrhu jejich praktického nasazení rovněž nebude nepopiratelnosti přinejmenším v teoretické rovině dosaženo.

7 Poděkování

Autor děkuje RNDr. Vlastimilu Klímovi za cenné připomínky k textu příspěvku.

8 Literatura

[1] Anderson, R. and Needham, R.: Robustness Principles for Public Key Protocols, *in Proc. of CRYPTO '95*, pp. 236-247, Springer-Verlag, 1995.

[2] Anderson, R. and Vaudenay, S.: Minding your p's and g's, *in Proc. of ASIACRYPT '96*, pp. 26-35, Springer-Verlag, 1996.

[3] Chen, M. and Hughes, E.: Protocol Failures Related to Order of Encryption and Signature: Computation of Discrete Logarithms in RSA Groups, in *Proc. of ACISP '98*, pp. 238-249, Springer-Verlag, 1998.

[4] Desmedt, Y., Landrock, P., Lenstra, A., McCurley, K., Odlyzko, A., Rueppel, R. and Smid, M.: The Eurocrypt '92 Controversial Issue - Trapdoor Primes and Moduli, in *Proc. of EUROCRYPT '92*, pp. 194-199, Springer-Verlag, 1992.

[5] FIPS PUB 180-1: *Secure Hash Standard (DSS)*, National Institute of Standards and Technology, January 2001, http://www.itl.nist.gov/fipspubs/fip180-1.htm.

[6] FIPS PUB 186-2: *Digital Signature Standard (DSS)*, National Institute of Standards and Technology, January 2001, http://csrc.nist.gov/publications/fips/fips186-2/fips186-2.pdf.

[7] Gordon, D.-M.: Designing and Detecting Trapdoors for Discrete Log Cryptosystems, in *Proc. of CRYPTO '92*, pp. 66-75, Springer-Verlag, 1992.

[8] Johnson, D., Menezes, A. and Vanstone, S.: The Elliptic Curve Digital Signature Algorithm (ECDSA), *International Journal of Information Security*, Vol 1, Issue 1, pp. 36-63, Springer-Verlag, 2001.

[9] Landwehr, C.-E.: Computer Security, *International Journal of Information Security*, Vol 1, Issue 1, pp. 3-13, Springer-Verlag, 2001.

[10] Menezes, A.-J., van Oorschot, P.-C. and Vanstone, S.-A.: *Handbook of Applied Cryptography*, CRC Press, 1996, online at http://www.cacr.math.uwaterloo.ca/hac/.

[11] Microsoft: CryptoAPI Version 2.0, *MSDN - Platform SDK*, 2000.

[12] *PKCS#1 v2.1: RSA Cryptography Standard*, RSA Laboratories, DRAFT2, January 5 2001.

[13] *Public-Key Cryptography Standards (PKCS)*, RSA Security, available at http://www.rsasecurity.com/rsalabs/pkcs/index.html.

[14] Pohlig S.-C., Hellman M.-E.: An improved algorithm for computing logarithms over GF(p) and its cryptographic significance, *IEEE Transactions on Information Theory*, 24 (1978), 106-110.

[15] Rivest, R.-L., Shamir, A. and Adleman L.: A method for obtaining digital signatures and public-key cryptosystems, *Communications of the ACM*, pp. 120-126, 1978.

[16] Vaudenay, S.: Hidden Collisions on DSS, *in Proc. of CRYPTO '96*, pp. 83-88, Springer-Verlag, 1996.

[17] *Vyhláška č. 366/2001 Sb.*, ÚOOÚ 2001.

[18] Zákon o elektronickém podpisu, *zákon č. 227/2000 Sb.*, 2000.