O klíčových kolizích ve schématech (EC)DSA

Kdopak se to vlastně podepsal?!

Nepopiratelnost. To slovo létá vzduchem snad na každé přednášce o elektronickém podepisování. Nejčastěji zde symbolizuje víru uživatelů, že nikdo nemůže později jednoduše tvrdit, že jím podepsaný dokument ve skutečnosti nikdy nepodepsal a podepsat ani nechtěl. Opravdu nemůže? O jedné, sice teoretické, leč pro praxi nikterak slibně vyhlížející možnosti bude pojednáno v následujícím článku.

Na úvod si nejprve uveďme volnou, ale dostatečně názornou definici pojmu nepopiratelnost: O systému řekneme, že poskytuje službu nepopiratelnosti, pokud pro nezávislou třetí stranu existuje způsob, jak se jednoznačně (se zanedbatelnou pravděpodobností omylu) přesvědčit o tom, že daná akce v systému nastala, respektive nenastala. Pod pojmem akce si můžeme představit například právě podepsání dokumentu či autentizaci subjektu do systému.

Z kryptologického hlediska chápeme nepopiratelnost jako jednu ze základních služeb (vedle ostatních, jako jsou důvěrnost, integrita a autentizace), které se snažíme vhodnou konstrukcí kryptografických mechanismů zajistit. Situace je přitom ve většině ohledů podobná zmíněným ostatním službám. Vezměme si například starou známou službu zajištění důvěrnosti. Víme, že základem všeho je kvalitní šifrovací algoritmus, ale že ten sám o sobě je pouze podmínkou nutnou (a nikoliv postačující). Víme, že je třeba ještě vybudovat kvalitní klíčové hospodářství, přičemž zvláštní pozornost musí být věnována způsobu uložení klíčového materiálu, a tak dále. V případě nepopiratelnosti je situace v zásadě obdobná. Nejprve musíme mít kryptografické schéma schopné tuto službu zajistit a poté je nutné věnovat pozornost návaznosti na ostatní procesy informačního systému tak, aby třetí strana (soud) byla v případě nějakého sporu schopna nezávisle rozhodnout ve smyslu výše zavedené definice.

V následujícím textu se budeme věnovat problematice (ne)popiratelnosti digitálních podpisů, i když zde představená technika útoku je aplikovatelná i v ostatních případech, kdy je dané podpisové schéma využito k zajištění mírně odlišné služby - například právě k autentizaci uživatelů.

Rovněž poznamenejme, že útok, který bude dále popsán, se zaměřuje na prolomení nepopiratelnosti přímo u základních podpisových schémat, tedy nikoliv ve vyšších vrstvách systému. Pro lepší porozumění technikám popírání podpisu je vhodné si uvést, jakým způsobem se vlastně proti uživateli zříkajícímu se svého podpisu vede důkaz na kryptologické úrovni. Budeme zde přitom předpokládat znalosti na úrovni článků [7]. Víme, že u schémat digitálního podpisu se mimo jiné opíráme o základní vlastnost, která říká, že: Pro libovolnou zprávu je výpočetně nemožné při pouhé znalosti veřejného klíče a parametrů nalézt její platný podpis (obecněji viz [7]). Platí-li tato hypotéza a předpoklad, že pouze daný uživatel zná svůj privátní klíč a jakékoliv jeho použití musí být tímto uživatelem vědomě schváleno (toto je nutno explicitně zajistit organizačně-technickými prostředky), potom každý platný podpis usvědčuje daného uživatele z toho, že daný dokument opravdu podepsal. Nyní se vžijme do situace, kdy bude uživatel chtít svůj podpis popřít. Aby tak učinil, musí zpochybnit platnost některého z tvrzení (či spíše z předpokladů), které jsou v důkazním procesu použity. Na výběr má zejména napadení:

a) jednoznačnosti obsahu zprávy (viz [7]);
b) platnosti základní vlastnosti;
c) výhradní kontroly nad privátním klíčem.

Cílem přitom vždy je přinést zmíněné důvěryhodné třetí straně alternativní vysvětlení toho, proč jen se pod daným dokumentem nachází platný podpis, když jej náš uživatel ve skutečnosti vlastně vůbec nepodepsal. Je velmi důležité si zde uvědomit, že veškeré "dokazování" je zde založeno na tom, že uživatel takové alternativní vysvětlení, které by vypadalo alespoň trochu věrohodně, předložit nedokáže. Jakmile by takové vysvětlení přinesl, musí se jím třetí strana začít vážně zabývat, a pokud zase ona nepřijde s rozumným důvodem, proč je to celé nesmysl, útočník vyhrál. Alespoň tedy ve světě teoretické kryptologie.

Praxe může být nakonec jiná a na základě fenoménu zvaného zdravý lidský rozum přece jen útočníka zarazit. Nicméně nemusí to tak být vždy, a proto bychom nad chybami v teorii rozhodně neměli zavírat oči. Právě proto se kryptologové snaží ze všech sil, vymýšlí a kombinují, aby si včas sami povšimli byť jen trochu rozumně vypadající cesty vedoucí k uvedenému alternativnímu vysvětlení. Když uvážíme, že ve světle ryze matematického pojetí je řada současných kryptografických mechanismů poněkud na vodě (toto téma by s přehledem vydalo na samostatnou knihu), není to rozhodně lehká a už vůbec ne rutinní práce, která by šla nějak metodicky podchytit (v tom je ovšem také její kouzlo).

Pojem k-kolize

Víme, že slovo kolize ve spojení s digitálním podpisem rozhodně nevěští nic dobrého. Většinou se zde mluví o kolizi použité hašovací funkce (h), což znamená, že útočník disponuje dvěma různými zprávami (m1,2) se stejným hašovým kódem (čili h(m1) = h(m2)). Vzhledem k tomu, že na úrovni běžných podpisových schémat se hašové kódy považují za dostatečně věrné reprezentanty podepisovaných zpráv, je útok na nepopiratelnost doslova nabíledni. Uživatel se bude při popírání svého podpisu snažit stranu provádějící důkaz přesvědčit o tom, že místo m1 ve skutečnosti podepsal m2 či obráceně. V drtivé většině případů tak dosáhne alespoň podstatného prodloužení celého sporu. Kryptologové o tomto problému samozřejmě vědí a usilovně pracují na tom, aby používané hašovací funkce hledání kolizí neumožňovaly.

Problémem, na který bylo poprvé poukázáno v pracích [6, 5], však je, že zprávy nejsou tím jediným, co může v daném schématu kolidovat! Kolidovat mohou totiž také veřejné klíče. O tom, jak taková situace vypadá, hovoří následující definice: Dvojici různých veřejných klíčů (PubA,B) nazveme klíčovou kolizí (dále k-kolizí, též k-kolizí prvního druhu), pokud existuje zpráva m a její podpis S tak, že S je platný podpis zprávy m vzhledem k oběma veřejným klíčům PubA i PubB.

Stejně jako je v případě hašovacích funkcí nežádoucí, aby byl útočník schopen nalézt kolizi podepisovaných zpráv, je zde nežádoucí to, aby byl schopen zkonstruovat k-kolizi. V opačném případě totiž může k třetí nezávislé straně přijít snadno s alternativním vysvětlením, kde bude tvrdit, že daný podpis by sice mohl být jeho, ale že ve skutečnosti jeho není! Jako důkaz pak předloží druhý z kolidujících veřejných klíčů a osobu s ním spojenou, která naopak bude tvrdit, že danou zprávu skutečně podepsala.

Jistě si umíme představit praktickou situaci, kdy by nám pomohlo, pokud bychom mohli později svůj podpis "hodit" na někoho jiného (někdy to může být i vzájemně výhodné). Celá situace je ilustrována na obrázku 1. Pro lepší názornost zde místo samostatných veřejných klíčů a parametrů schématu vystupují jejich autentizované nosiče - certifikáty. S ohledem na další zaměření tohoto příspěvku zde také uvádíme podpis implicitně ve tvaru dvojice hodnot (r, s).

Ještě horších překvapení se můžeme dočkat v případě, kdy je znám nekooperativní postup vedoucí ke zvládnutelnému hledání k-kolizí. To znamená, že útočník je schopen nalézt kolidující veřejný klíč (nejlépe včetně privátního), aniž by od majitele prvního klíče potřeboval jakékoliv informace, které nejsou veřejně k dispozici. V takovém případě se může útočník namísto popírání svého podpisu naopak aktivně hlásit k podpisu cizímu.

Na první pohled to nevypadá jako vážný nedostatek, neboť je-li dokument veřejný, může jej kdokoliv a kdykoliv bez obav podepsat. Určitě bychom však našli řadu případů, kdy není možné, aby dokument podepsal kdokoliv a kdykoliv, a pak lze logicky usuzovat na to, že útočník získá nekooperativním výpočtem k-kolize jistou nezanedbatelnou výhodu. Jako příklad si můžeme uvést situaci, kdy je podepsaný dokument opatřen nějakým razítkem (časovým, notářským podpisem apod.). Toto razítko již nelze později "podlézt", takže přidání nového či náhrada starého podpisu nejsou jednoduše možné. Pokud však původní razítko nepokrývá také detailní informace o veřejném klíči, který se má k ověření podpisu použít (což se bohužel v praxi stává), může útočník takto orazítkovaný podpis směle vydávat za svůj. Tato situace je ilustrována v levé části obrázku 1.

Výpočet k-kolizí u (EC)DSA

Existenci a způsoby hledání k-kolizí můžeme studovat u každého schématu digitálního podpisu, pochopitelně ovšem s různým výsledkem. Zatím byla tato problematika otevřena v souvislosti se schématy RSA ([6]) a (EC)DSA ([6, 5]). V obou případech existují výpočetně triviálně schůdné postupy pro nekooperativní hledání k-kolizí pro prakticky libovolné hodnoty vstupní zprávy, jejího podpisu a veřejného klíče (včetně veřejných parametrů). Nejjednodušší je pak situace jednoznačně v případě (EC)DSA, kdy je navíc velmi obtížně prokazatelné, který z dvojice kolidujících veřejných klíčů byl vytvořen útočníkem. To výrazně ztěžuje potenciální spory o autorství podpisu v případě, že se k němu aktivně hlásí oba majitelé kolidujících klíčů (viz níže). Dále se proto budeme věnovat právě schématu (EC)DSA ([2, 3, 4]).

Z důvodu rozsahu článku musíme udělat ještě jeden zjednodušující krok. Vlastní popis útoku provedeme pouze na DSA s tím, že hned zde zkonstatujeme, že uvedený postup je triviálně rozšiřitelný i na (EC)DSA, což bylo ukázáno v [6]. Sama jednoduchost tohoto přechodu je zde pozoruhodná a plyne z toho, že útok využívá velmi obecných algebraických vlastností DSA, které jsou s ECDSA společné.

Hlavní partie schématu DSA jsou připomenuty na obrázku 2, vlastní postup výpočtu kolidující instance je uveden na obrázku 3. Podrobnější rozbor tohoto algoritmu je podán v [6, 5], zde se zaměříme pouze na jeho hlavní body. Připomeňme, že cílem výpočtu je pro známou hodnotu podpisu (r, s) zprávy m a veřejný klíč yA s parametry (pA, qA, gA) najít veřejný klíč yB s parametry (pB, qB, gB) tak, aby (yA, yB) vytvářely na podpisu (r, s) zprávy m k-kolizi. V prvním kroku algoritmu začínáme tím, že pokládáme pB =

pA a qB = qA. Tento krok je zcela v souladu s definicí DSA [2], kde se předpokládá, že více uživatelů může sdílet společné veřejné parametry. Pokud se použijí mechanismy pro svědectví o správném vygenerování těchto hodnot ([2]), tak je k tomu dokonce i dobrý důvod. V případě ECDSA tento krok odpovídá tomu, že útočník použije stejnou eliptickou křivku nad stejným tělesem jako majitel instance A. Takový postup je zcela běžný a na první pohled rovněž nebudí žádné podezření (zvláště když A použil některou ze standardizovaných křivek [2]).

V krocích (ii) až (v) je nalezen generátor gB spolu s hodnotou dočasného klíče zprávy kB pro podpis od uživatele B. Zde je patrně hlavní moment celého postupu, kdy díky (velmi obecným) algebraickým vlastnostem DSA umíme najít hodnoty gB, kB tak, aby platilo r = (gBkB mod pB) mod qB, aniž bychom byli nuceni řešit složitý problém diskrétního logaritmu. V jistém, zcela nepravděpodobném případě se může stát, že jsme přímo našli kB = kA. Potom jednoduše podle rovnice pro sA (viz obrázek 2, krok (iii)) najdeme přímo privátní klíč xA uživatele A. Zde si nelze odpustit jistý komentář - totiž stojí za poznamenání, že pokud útoky tohoto typu z nějakého důvodu během výpočtu vybočí z "normálních" kolejí, potom většinou ku prospěchu útočníka, který tak získá nakonec víc, než původně chtěl!

Dále algoritmus pokračuje nalezením hodnot privátního a veřejného klíče uživatele B. V poznámkách je pak uveden vztah mezi těmito hodnotami a klíči na straně uživatele A. Z těchto poznámek je vidět, že ve velmi speciálním případě se může stát, že výsledkem algoritmu bude yB = yA. Taková situace však nastává se zanedbatelnou pravděpodobností, a proto ji zde nebudeme uvažovat.

Za zmínku dále stojí hodnota b, která obě instance v podstatě odděluje a která je závislá na náhodných číslech kA a z, přičemž každé z nich je voleno jiným uživatelem. K tomu, aby jeden z nich dokázal obě instance propojit, musí znát obě tyto hodnoty (nebo být schopen rozbít DSA zcela, což nepředpokládáme). Odtud plyne, že oba majitelé kolidujících instancí vystupují před třetí důvěryhodnou stranou ve zcela symetrických pozicích a na základě pouhé matematiky stojící za DSA nelze rozhodnout, který z nich vlastní "padělanou" instanci. Obě instance lze samozřejmě bez problémů (ať už s funkčností či s bezpečností) používat k "normálnímu" podepisování. Více o vlastnostech vypočtených instancí viz [6, 5].

Samozřejmě i zde přichází záchrana v podobě mechanismů vyšší úrovně. Například druhý uživatel bude mít problém získat důvěryhodný certifikát datující vznik jeho instance zpět do minulosti, takže pomocí zdravého lidského rozumu snad nakonec bude rozhodnuto správně. Na druhé straně je to jasné selhání DSA, které by do vyšších vrstev rozhodně nemělo pouštět takové "špeky".

Protiopatření a k-kolize druhého druhu

Za hlavní opatření proti uvedenému útoku lze považovat rozšíření kontrol korektního vygenerování parametrů p, q ([2]) i na generátor g. Tím by útočník ztratil možnost libovolné volby tohoto parametru, na čemž je celý jeho postup založen. Základní princip zmíněných kontrol spočívá v tom, že sledované parametry jsou vytvářeny jednosměrnými funkcemi (fp, fq), přičemž systém kromě (p, q) uchovává také SEEDp a SEEDq, kde p = fp(SEEDp), q = fq(SEEDq). Tento vztah lze kdykoliv ověřit a díky jednosměrnosti použitých funkcí nelze testovací hodnoty SEEDp a SEEDq padělat (v současné definici [2] se používá jedna společná hodnota SEED). Analogický mechanismus existuje i pro ECDSA.

Je celkem s podivem, že tato technika nebyla rozšířena i na generátor g (toto už v minulosti způsobovalo problémy - viz [8], kde je diskutována problematika "klasických" kolizí zpráv). Tento krok by proto nyní měl logicky následovat. Nelze ovšem dopředu říci, zda a kdy se k němu americká autorita NIST odhodlá (o útoku byla informována), takže zatím nezbývá než obdobný mechanismus v citlivých systémech zavést po svém. Prakticky by pak vyhodnocení správnosti podpisu probíhalo podle obrázku 4. Poznamenejme, že operátor AND na konci postupu v podstatě existuje už dnes a vstupují sem podmínky, jako je platnost certifikátu apod.

V jistých případech může pomoci ještě jeden druh protiopatření a tím je přidání detailních informací o veřejném klíči (nejlépe celého certifikátu) do podepisovaných dat. Podle osobních zkušeností s návrháři informačních systémů mohu konstatovat, že tento postup vypadá z jejich pohledu elegantně a jednoduše. Důrazně zde ovšem podotýkám, že to rozhodně není univerzální opatření a že může stejně elegantně a jednoduše selhat. Abychom viděli proč, musíme nejprve zavést k-kolize druhého druhu: Dvojici různých veřejných klíčů (PubA,B) nazveme k-kolizí druhého druhu, pokud existují dvě zprávy m1,2 a jeden podpis S tak, že S je platný podpis zprávy m1 vzhledem ke klíči PubA a zároveň S je platný podpis zprávy m2 vzhledem ke klíči PubB. Na rozdíl od k-kolize prvního druhu nestanovujeme u druhého druhu to, že se podpis musí vztahovat ke stejné hodnotě zprávy. Taková kolize je užitečná právě v případě útočení na zmíněné elegantní a jednoduché protiopatření.

Celá situace je naznačena na obrázku 5. Vidíme, že pro uživatele A a B se liší konkrétní hodnota zprávy, kterou nakonec oba podepisují. Tato odlišnost je způsobena právě přidáním informací o veřejném klíči. Pokud je však tato informace přidána tak, že ji lze kdykoliv vyměnit (tj. není pevně svázána s vlastní

zprávou nějakou formou razítka - pozor, i zde se mohou uplatnit k-kolize), může útočník se schopností hledat k-kolize druhého druhu pro libovolné dvojice zpráv snadno zvítězit.

Nyní již zbývá jen ukázat, že u (EC)DSA lze snadno nekooperativně hledat i k-kolize druhého druhu, a to pro prakticky libovolnou dvojici m1,2. Použijeme k tomu přímo algoritmus z obrázku 3, kde v kroku (ii) použijeme jako m zprávu m1 a v kroku (vi) jako m dosadíme m2. Poznámky na konci algoritmu pak budou pochopitelně vypadat poněkud jinak. S ohledem na rozsah článku však jejich odvození musíme ponechat na "domácí cvičení".

Závěr

Problematika klíčových kolizí v podpisových schématech představuje hrozbu pro systémy spoléhající se na nepopiratelnost elektronic-kého podepisování založeného na bázi schémat digitálního podpisu. V příspěvku jsme si v hrubých rysech představili tento zcela původní způsob útoku na podpisová schémata, přičemž jako konkrétní příklad bylo předvedeno napadení schématu (EC)DSA, jež spolu s RSA (u kterého lze také najít tento druh útoku - viz [6]) patří celosvětově k nejrozšířenějším.

S ohledem na praktické systémy je hrozba plynoucí z rozebíraného útoku zatím spíše teoretická. Tento stav však není ani tak způsoben "zázračnou" odolností praktických systémů, jako skutečností, že dosud neexistuje mnoho takových aplikací, ve kterých by se zisk z prolomení nepopiratelnosti vyrovnal vynaloženému úsilí. V okamžiku, kdy takové systémy vzniknou, stane se uvedená hrozba mnohem reálnější. Proto je jistě vhodné využít tento "oddechový" čas k implementaci příslušných protiopatření.

Tomáš Rosa, autor@chip.cz

Literatura: [1] Archiv českých článků o kryptologii, www.decros.cz/bezpecnost/_kryptografie.html. [2] FIPS PUB 186-2: Digital Signature Standard (DSS), National Institute of Standards and Technology, January 27, 2000, http://csrc.nist.gov/publications/fips/fips1862/fips186-2.pdf, update: October 5, 2001. [3] Klíma, V.: Eliptické křivky a šifrování (1 - 2), CHIP, září říjen 2002, dostupné v [1]. [4] Klíma, V.: Počítačová bezpečnost (DSS): Podpis bez pera i papíru, Chip, květen 1999, dostupné v [1]. [5] Rosa, T.: On Key-collisions in (EC)DSA, CRYPTO 2002 Rump Session, Santa Barbara, USA, August 2002, http://eprint.iacr.org/2002/129/. [6] Rosa, T.: O klíčových kolizích v podpisových schématech, Sborník přednášek semináře Velikonoční kryptologie, str. 14 - 26, Brno, 2002, dostupné v [1]. [7] Rosa, T.: Podpis pro pokročilé (1 - 2), Chip, listopad prosinec 2000, dostupné v [1]. [8] Vaudenay, S.: Hidden Collisions on DSS, in Proc. of CRYPTO '96, pp. 83-88, Springer-Verlag, 1996.

Obr. 2. Přehled schématu DSA

Instance DSA je tvořena:

Veřejnými parametry (p, q, g), kde:
p, q jsou prvočísla, q|(p-1), 21023 < p < 21024, 2159 < q < 2160,
g je generátor cyklické podgrupy grupy Zp* řádu q.
Privátním klíčem x, 0 < x < q.
Veřejným klíčem y, y = gx mod p.

Výpočet podpisu zprávy m:

i) Zvolme náhodný klíč zprávy k, 0 < k < q.
ii) Vypočtěme r = (gk mod p) mod q.
iii) Vypočtěme s = (h(m) + xr)*k-1 mod q, kde kk-1 ľ 1 (mod q) a h je hašovací funkce - ve standardu DSS h =def SHA-1.
iv) Podpisem budiž dvojice (r, s).

Ověření podpisu (r, s) zprávy m:

i) Vypočtěme w: ws ľ 1 (mod q).
ii) Vypočtěme v = (gwh(m)ywr mod p) mod q.
iii) Podpis je platný iff v = r.

Obr. 3. Postup výpočtu k-kolize pro DSA

Vstup: Podpis (r, s) zprávy m, veřejný klíč yA a parametry (pA, qA, gA), hašovací funkce h použitá při podpisu m.

Postup:
i) Položme pB = pA = p, qB = qA = q.
ii) Vypočtěme a = gAwh(m)yAwr mod p, kde w*s ľ 1 (mod q).
iii) Zvolme náhodné číslo kB, 0 < kB < q.
iv) Vypočtěme z: zkB ľ 1 (mod q).
v) Vypočtěme gB = az mod q.
vi) Vypočtěme privátní klíč xB = t(kBs - h(m)) mod q, kde rt ľ 1 (mod q).
vii) Vypočtěme veřejný klíč yB = gBxB mod p.
viii) Výstupem budiž instance DSA (pB, qB, gB, xB, yB).

Poznámky:
i) xB = (b-1 - 1)h(m)t + b-1xA mod q, kde b ľ kAz (mod q), bb-1 ľ 1 (mod q)
ii) yB = gA(1-b)h(m)tyA mod p