

ÚTOK NA PRIVÁTNÍ PODPISOVÉ KLÍČE PGP

Czech attack

Titulková konference společnosti ICZ ze dne 20. března 2001 opravdu nezůstala bez povšimnutí. Dva čeští kryptologové (které dobře znáte z jejich pravidelných příspěvků v Chipu) na ní novinářům a odborné veřejnosti oficiálně oznámili svůj objev týkající se bezpečnostní chyby v PGP. Na rozdíl od většiny informací v neoborných médiích máte nyní v Chipu možnost dozvědět se o této záležitosti nezkrácená fakta, a to přímo z první ruky – od obou autorů.

Inu, tak se nám podařilo udělat pěkný rozruch ve světě i na domácí politické scéně. Některým novinářům se totiž zdálo, že v ohrožení je elektronický podpis jako takový, ve své obecnosti – a byl z toho dost velký poprask. Ve skutečnosti se podařilo něco jiného, ovšem velmi zajímavého, a jsme rádi, že vám to konečně můžeme podrobněji vysvětlit, protože „těch pravých vykladačů“ máme, mírně řečeno, už dost.

OČ VLASTNĚ ŠLO?

Objevili jsme útok na privátní podpisové klíče formátu OpenPGP a programu PGP. Protože na světě program PGP údajně používá asi 10 milionů lidí, nešlo jen tak o maličkost. (*PGP je registrovaná obchodní značka Network Associates, Inc., všechny ostatní registrované a neregistrované obchodní značky v tomto článku jsou majetkem odpovídajících vlastníků.*)

Formát **OpenPGP** byl definován v dokumentu RFC2440 z roku 1998 s cílem poskytnout progra-

mátorům všechny nezbytné informace, aby na jeho bázi mohly být vytvářeny další kompatibilní aplikace s programem **PGP**, který ho sám používá.

Poukázali jsme na závažnou chybu, která spočívá v nedostatečném zajištění integrity veřejných i privátních částí *podpisových* klíčů algoritmů DSA a RSA v tomto formátu, a ukázali jsme, že může být využita k **odhalení privátního podpisového klíče**.

Aby byl privátní klíč chráněn, je jeho hodnota uložena do privátní „klíčenky“ (souboru *secring.skr*) v **zašifrovaném** tvaru (ještě jednou zdůrazňujeme, že je tam zašifrován). K zašifrování je použita uživatelem zvolená silná symetrická šifra (AES, CAST5, IDEA) s dostatečně dlouhým klíčem. Ten je odvozen od tajného přístupového hesla, které zná jen uživatel (říkejme mu *password*, ale přesněji je to „passphrase“, klíčová přístupová věta – ostatně může to být skutečně velmi dlouhá věta). Ukázali jsme, že pokud útočník má přístup k souboru *secring.skr* (zdůrazněme, že přístup se požaduje pouze v době, kdy je tam privátní klíč uložen v **zašifrovaném** tvaru), může za určitých podmínek (jak uvidíme, nejsou nijak přehnané) získat privátní podpisový klíč uživatele, aniž by znal jeho přístupové heslo nebo proti němu útočil.

Útok spočívá ve speciální modifikaci (veřejných) parametrů podpisového algoritmu (u DSA) nebo šifrovaného obrazu privátního klíče (u RSA) a získání podpisu libovolného souboru (nebo e-mailové zprávy) takto modifikovaným podpisovým klíčem. Ukazujeme, že pak je útočník schopen vypočítat privátní podepisovací klíč. Protože soubor *secring.skr* může útočník po záměně uvést zpět do původního stavu, je útok velmi nebezpečný. Ve stejném ohrožení jsou i privátní klíče přenášené v zašifrovaném tvaru na disketách nebo po síti.

Útok jsme prakticky ověřili na nejnovější verzi programu PGP 7.0.3 s kombinací algoritmů AES a DH/DSS. Výsledkem bylo získání privátního podpisového klíče algoritmu DSA. V technické zprávě (najdete ji také v rubrice Chip Plus na Chip CD 5/01) jsme samozřejmě navrhli krátkodobá i dlouhodobá kryptografická opatření pro opravu formátu OpenPGP a změny v programu PGP. Podrobnou revizi ale musí projít všechny další aplikace, které jsou s formátem OpenPGP kompatibilní. Jedná se například o *GNU Privacy Guard* a další, uvedené na seznamu aplikací kompatibilních s OpenPGP, které jsou vyjmenovány na <http://www.pgpi.org/products>. K tomu také postupně dochází a informace o tom jsou průběžně aktualizovány na www.i.cz a dalších.

S ČÍM JSME ŠLI „NA ZTEČ“

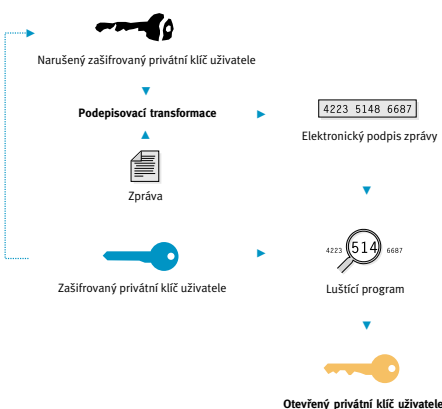
ICZ attack, Czech attack, Rosa and Klima attack – takové názvy už si náš počin vysloužil v kryptologickém světě. Naše technická zpráva popisuje celkem tři typy útoků (jeden na DSA, dva na RSA), s jejichž podstatou vás dále seznámíme. Než se na ně podíváme podrobněji, osvěžme si v paměti postup vytváření klíčového páru a podpisu pomocí 1024bitového algoritmu DSA a postup verifikace podpisu.

VYTVÁŘENÍ KLÍČOVÉHO PÁRU

Každý uživatel vygeneruje privátní a veřejný klíč následujícím postupem:

- 1) Zvol 160bitové prvočíslo q tak, že $2^{159} < q < 2^{160}$.
- 2) Zvol 1024bitové prvočíslo p tak, že q dělí $(p-1)$ a $2^{1023} < p < 2^{1024}$.
- 3) Vyber generátor g cyklické podgrupy řádu q v \mathbb{Z}_p^* (tj. zvolil se prvek $h \in \mathbb{Z}_p^*$ tak, že $g = h^{(p-1)/q} \bmod p$ a $g \neq 1$, jinak se volí jiné h).

Obecný postup útoku



Obr. 1. Obecný postup útoku

→ 4| Vyber náhodné číslo x tak, že $1 \leq x \leq q-1$.

5| Vypočti $y = g^x \bmod p$.

6| Veřejný klíč je y , veřejné parametry jsou (p, q, g) , privátní klíč je x .

VYTVÁŘENÍ DIGITÁLNÍHO PODPISU

Uživatel při vytváření podpisu zprávy m (resp. její hašovací hodnoty $h(m)$) používá svůj privátní klíč x a veřejné parametry takto:

1| Vyber náhodné tajné číslo k , $0 < k < q$.

2| Vypočti $r = (g^k \bmod p) \bmod q$.

3| Vypočti $kl_{nv} = k^{-1} \bmod q$.

4| Vypočti $s = [kl_{nv} * (h(m) + x^*r)] \bmod q$.

5| Digitální podpis zprávy m je dvojice (r, s) .

VERIFIKACE DIGITÁLNÍHO PODPISU

Při verifikaci digitálního podpisu zprávy m použijeme veřejný klíč spolu s veřejnými parametry signatáře (p, q, g, y) takto:

1| Ověř, že $0 < r, s < q$. V opačném případě je podpis neplatný.

2| Vypočti $sl_{nv} = s^{-1} \bmod q$ a hašovací hodnotu $h(m)$.

3| Vypočti $u_1 = sl_{nv} * h(m) \bmod q$,
 $u_2 = sl_{nv} * r \bmod q$.

4| Vypočti $v = (g^{u_1} * y^{u_2} \bmod p) \bmod q$.

5| Podpis je platný, právě když $v = r$.

ÚTOK NA PODPISOVÝ ALGORITMUS DSA

Po této malé teoretické průpravě si postup útoku na 1024bitový algoritmus DSA popíšeme konkrétně tak, jak jsme ho provedli s využitím programu PGP 7.0.3. V dalším textu budeme podvržené a na základě nich vypočítané hodnoty označovat vždy čárkou.

1. KROK

Zvolili jsme prvočíslo p' tak, aby mělo 159 bitů a byla tak určitě zaručena podmínka $p' < q$. Je to **první klíčová myšlenka** (využijeme ji ve čtvrtém kroku). Dále jsme volili p' ve tvaru $p' = t \cdot 2^s + 1$ tak, aby 2^s bylo co největší číslo a t bylo malé prvočíslo, konkrétně $s = 151$ a $t = 167$. Toto je **druhá klíčová myšlenka** útoku. **Pro takto speciálně volená prvočísla umíme rychle řešit problém diskretního logaritmu**, tj. z rovnice $r' = (g')^k \bmod p'$ umíme snadno vypočítat neznámé číslo k , zatímco pro původní sestavení veřejných parametrů by byla tato úloha výpočetně nevládnutelná. Celá podstata našeho útoku tak spočívá v modifikaci veřejných parametrů tak, že úloha vedoucí k nalezení privátního klíče se stává snadno řešitelnou.

Dále jsme zvolili číslo $g' = 0x31AC85291FF814E625E4B88C8C5047A7DB2F0E45$ tak, aby bylo generátorem multiplikační grupy $Z_{p'}^*$.

2. KROK

Nyní jsme získali soubor *secring.skr* a v jeho záznamu (*Secret Key Packet*) jsme vyměnili hodnoty (p, g) za hodnoty (p', g') . Upravili jsme také délky těchto čísel ve formátu MPI. Tento formát se skládá ze dvou délkových oktětů (bajtů) označujících počet platných bitů v následujících oktětích a z vlastní hodnoty „velkého“ čísla *Multi-Precision Integer*, která je ve formátu BIG ENDIAN a je zarovnaná na celé oktety. A nakonec jsme zkrátili celkovou délku Secret Key Packet (délkové oktety na počátku záznamu) tak, aby odpovídala kratším falešným hodnotám p' a g' . Situaci ilustrují obrázky: na obr. 2 jsou vyznačeny původní hodnoty, na obr. 3 hodnoty podvržené. Dále je vidět nedotčená oblast, kde je silnou šifrou ochráněn a námi zachován privátní klíč x .

3. KROK

S takto podvrženými hodnotami jsme vyčkali, až uživatel podepíše nějaký nám známý soubor (zprávu m), a získali jsme jeho podpis – hodnoty (r', s') . Tu neznámou náhodnou hodnotu, kterou uživatelův program zvolil při tomto podpisu, označme k (viz popis DSA výše).

4. KROK

V tomto kroku jsme vypočítali hodnotu privátního klíče x uživatele na základě hodnot p' , g' , m , r' a s' . Z definice hodnoty podpisu (r', s') totiž vyplývá, že

$$(1) \quad r' = (g')^k \bmod p' \bmod q, \text{ což vzhledem}$$

k volbě $p' < q$ dává

$$(1a) \quad r' = (g')^k \bmod p'$$

a

```
9501D7043A8D29DF110400F2E02A396A14E137085DA859B3569AF4027EA379682F467
80920B72127C88787DDC1BFF9FDB59E564B741FD5FC98856679F1C041CB71895CB697
5E7FE6E15A6D4B70514560E11A25637F3FBA35E89E5F1FA272A2707F4865EA106EE40
2973D4969A276DA4911005B968B81548621CEBB5771A35C5A785F7F480E47277D2BA
B500A0FF04303152BD2A9AD963E063A3FE34A8A5534F3F0400CD8580F20AA821A6D2F
F5255DFD02E4F4C8D8DA3731517476BEE096F7B104B01B6CE1C4DE586BAEA30D82B50
DCB3F0D20B0F0D07D8384C09F12CBF079887CEB696E822D753A48584F2BC573C84E84
90AB310FDBCC40EAEBBCD05973B3F2A1A479FFE0E4B63026E066B6E936F1B2B7F1C91C
65CBA0F27B4C0D22254B8C852DEDE10400AC756BB6EB82313A0FE91F47A36D1425D89
FB124CD0ACBA082E8B2C2B048BE92C5CE7A5FAA5AF317DCC086150B98AB504C0DA6BF
1D87FAB73C8F8D0FC821BD8902CA6927338CF0D682E7C9E3E8D89A3D00D53224C301E
6C932ADA7562FA15E9027E105F803043D4CBC08807A8FB71FEF9B27EE6A0722C4BF60
1D032CC59F6FE4FF09030213CF38106B7BCA3F603F59437C3860B98DA3A1A3F02A4D2
754075B494CAC156E38E1282705FB0BB68940A1653457E161AB00187B428566C6173
74696D696C204B6C696D612C20445353203C762E6B6C696D6140646563726F732E637
A3EB00303FFFF89005A04101102001A05023A8D29DF050901E13380040B0908070219
01051B0300000000A09109B89D5F084A0EAD7A35C009F5B643D5D2C37F4B2CFC9F39
9873B747CB3FBFE6800A0A ... zkráceno ... 0167
```

Obr. 2. Soubor *secring.skr* hexadecimálně; vyznačeny jsou původní hodnoty p, g , jejich délky a celková délka záznamu i zašifrovaná oblast.

```
9500FF043A8D29DF11009F53800000000000000000000000000000000000000000000100A0FF0
4303152BD2A9AD963E063A3FE34A8A5534F3F009E31AC85291FF814E625E4B88C8C50
47A7DB2F0E450400AC756BB6EB82313A0FE91F47A36D1425D89FB124CD0ACBA082E8B
2C2B048BE92C5CE7A5FAA5AF317DCC086150B98AB504C0DA6BF1D87FAB73C8F8D0FC8
21BD8902CA6927338CF0D682E7C9E3E8D89A3D00D53224C301E6C932ADA7562FA15E9
027E105F803043D4CBC08807A8FB71FEF9B27EE6A0722C4BF601D032CC59F6FE4FF09
030213CF38106B7BCA3F603F59437C3860B98DA3A1A3F02A4D2754075B494CAC156E3
8E1282705FB0BB68940A1653457E161AB00187B428566C617374696D696C204B6C69
6D612C20445353203C762E6B6C696D6140646563726F732E637A3EB00303FFFF89005
A04101102001A05023A8D29DF050901E13380040B090807021901051B0300000000A
09109B89D5F084A0EAD7A35C009F5B643D5D2C37F4B2CFC9F399873B747CB3FBFE680
0A0A ... zkráceno ... 0167
```

Obr. 3. Soubor *secring.skr* po narušení; vyznačeny jsou změněné hodnoty p', g' , jejich délky a celková délka záznamu i nezměněná zašifrovaná oblast.

→ (2) $s' = \{[k^{-1} \bmod q] * [h(m) + x * r']\} \bmod q$,
tedy

(2a) $x = \{[s' * k - h(m)] * [(r')^{-1} \bmod q]\} \bmod q$.

Klíčovým bodem je, že číslo k umíme díky volbě p' a g' vypočítat z rovnice (1a). Na základě toho pak z rovnice (2a) jen dopočítáme privátní klíč – hodnotu x . Jeho správnost ověříme proti hodnotě veřejného klíče: $y = g^x \bmod p$, kde y je veřejný klíč a g, p originální uživatelské parametry.

5. KROK

Uživateli jsme vrátili jeho původní soubor *secring.skr*.

ÚTOKY NA PODPISOVÝ ALGORITMUS RSA

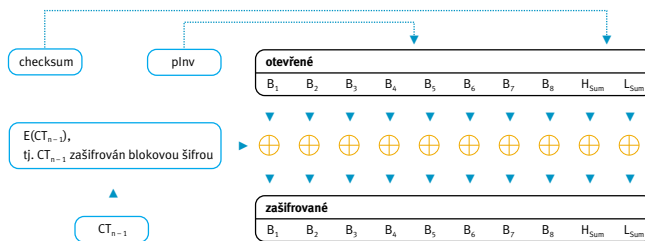
Zde vycházíme ze změny šifrovaného textu (nebo veřejného parametru udávajícího jeho délku) v souboru *secring.skr*, kde jsou uloženy hodnoty ($d, p, q, plnv$) ve struktuře Secret Key Packet. Připomeneme si, že modul RSA je součin dvou prvočísel $n = p * q$, veřejný exponent je e , d je privátní exponent a číslo $plnv = p^{-1} \bmod q$ slouží k rychlejšímu výpočtu podpisu. Veřejným klíčem označujeme dvojici (n, e), pro potřeby formátu OpenPGP se za privátní klíč považuje čtveřice ($d, p, q, plnv$). Podpis zprávy m je vytvořen jako $s = m^d \bmod n$, přičemž se předpokládá, že zpráva m už je určitým způsobem předem zformátována. Podpis je platný, pokud platí $m = s^e \bmod n$.

Předesíláme, že nyní útočíme na podpisový klíč algoritmu RSA ve formátu OpenPGP; programy PGP však mají navíc zabudovány další kontrolní mechanismy na integritu tohoto klíče před jeho použitím k podpisu, proto tam tento útok selhává (později bylo zjištěno, že v některých starších verzích PGP 2.6.x je ale útok funkční). I zde je privátní klíč uložen ve struktuře Secret Key Packet, přičemž v současné době se používají verze 3 a 4 tohoto formátu, které se liší ve způsobu šifrování privátních dat.

Klíčová myšlenka útoku **na verzi 3** formátu je založena na skutečnosti, že můžeme beztréstně zkrátit délku čísla $plnv$. Autoři formátu se domnívali, že délka $plnv$ není důležitá informace, takže ji ponechali veřejně přístupnou, zatímco hodnota $plnv$ je zašifrovaná! Jak využít změny $plnv$, si řekneme za okamžik.

Útok **na verzi 4** formátu je složitější. Čtveřice tajných čísel ve formátu MPI včetně jejich délkových oktětů je opatřena dvoubajtovým kontrolním součtem (*checksum*) a toto celé ($d, p, q, plnv, checksum$) je pak zašifrováno. Kontrolní součet je počítán jako $checksum = (d_1 + d_2 + \dots + d_n) \bmod 65536$, kde d_i jsou chráněná data, a k šifrování je použita uživatelem zvolená symetrická bloková šifra v tzv. specifickém (PGP) modu CFB, jak ukazuje obrázek 4.

Secret Key Packet



Obr. 4. Způsob šifrování posledního bloku dat privátního klíče v Secret Key Packet, obsahujícího privátní hodnotu *plnv*.

Útok na verzi 4 formátu Secret Key Packet jsme nazvali **podlezením šifry** (zde AES – Rijndael), protože se jedná o nevhodnou kombinaci modu šifrování CFB a výpočtu hodnoty checksum. Modifikujeme totiž poslední zašifrované bajty *plnv* a *checksum* tak, aby *checksum* po odšifrování souhlasila. Pokud použijeme blokovou šifru s délkou bloku 16 oktětů (tj. v případě AES) a modul RSA 1024 bitů, bude v posledním neúplném bloku šifrovaného textu osm posledních oktětů čísla *plnv* (označme je B_1, B_2, \dots, B_8) a dva oktety kontrolního součtu *checksum* (označme je H_{sum}, L_{sum}). Tato otevřená data budou zašifrována prostou operací XOR s určitým klíčovým materiálem, jak ukazuje obrázek 4. Pokud v posledním bloku šifrovaného textu provedeme změnu typu „XOR CONST“, projeví se to přesně jako „XOR CONST“ po odšifrování v otevřeném textu.

Přesný popis útoku lze nalézt ve zprávě, zde si uvedme alespoň malý příklad. V zašifrovaném souboru *secring.skr* zkusíme změnit například poslední bit B_8 a poslední bit L_{sum} . Po odšifrování bude pochopitelně hodnota *plnv* narušena (to chceme), ale jestliže byly v otevřeném tvaru oba bity stejné, po odšifrování se obě hodnoty buď o jedničku sníží, nebo o jedničku zvýší, takže *checksum* naše narušení *plnv* nepozná a bude v pořádku. Pokud to nevyjde, zkusíme to s dalšími bity (i jinak, viz zpráva). Nicméně v průměru dva pokusy by měly stačit, abychom byli úspěšní. Výsledkem této změny je narušení hodnoty *plnv*. Ve zprávě je pak ukázáno, jak lze z podpisu s narušenou hodnotou *plnv* vypočítat privátní klíč RSA!

DALŠÍ MOŽNOSTI ÚTOKŮ

Dále je nutné poznamenat, že kromě privátní klíčičenky *secring.skr* lze stejným způsobem útočit i na privátní klíč exportovaný do souboru typu *ASCII Key File* a přenášený potom prostřednictvím sítě nebo na disketě.

NAVRŽENÁ PŘECHODNÁ PROTIOPATŘENÍ

Odstranit hlavní příčinu prezentovaných útoků, kterou je nedostatečná kontrola integrity veřejných i privátních dat v souboru obsahujícím pri-

vátní klíč uživatele, nemusí být jednoduché. Proto jsme do doby, než dojde k úpravě formátu OpenPGP, navrhli přechodná opatření (bereme je jako „krizová“, nikoli tedy tak, že by hlavní příčinu vyřešila). Lze je přechodně použít v programech PGP a dalších, které implementují formát OpenPGP. K operaci vlastního podpisu přitom smí být použit jen takový klíč, jehož hodnoty projdou následujícím testem.

PŘECHODNÝ TEST PRO DSA

- 1| $p, q, g, x, y > 0$
- 2| p je liché, q je liché
- 3| $2^{159} < q < 2^{160}$
- 4| $1 < g < p$
- 5| $1 < y < p$
- 6| $x < q$
- 7| q dělí $(p-1)$
- 8| $g^q \bmod p = 1$
- 9| $g^x \bmod p = y$

PŘECHODNÝ TEST PRO RSA

- 1| $e * d \bmod (p - 1) = 1$
- 2| $e * d \bmod (q - 1) = 1$
- 3| $plnv * p \bmod q = 1$
- 4| n (ze záznamu veřejného klíče) = $p * q$
- 5| $e \in E$, kde E je množina možných hodnot plánovaných pro e , tj. pro PGP například {17, 65537, ...}

Poznamenejme, že v programu PGP jsou kontroly 1 až 4 a některé další implementovány. Ve formátu OpenPGP však tyto kontroly uvažovány nejsou.

DALŠÍ NÁMĚTY PRO FORMÁT OPENPGP

Zde uvádíme další náměty, které nás napadly po prvním seznámení s formátem OpenPGP (záznam →

INFOTIPY

Formát OpenPGP:
RFC 2440: OpenPGP Message Format, IETF, November 1998
Podrobnosti útoku (technické zprávy, prezentace, FAQ ap.):
► <http://www.i.cz>

- Secret Key Packet) a s programem PGP a které by rozhodně přispěly ke zvýšení bezpečnosti formátu i programu. Je však třeba je chápat spíše jako ideová doporučení a podrobit je hlubší analýze. Analýza celého formátu OpenPGP je však mnohem obtížnější a bohužel sám formát je tak nepřehledný, že by si zasloužil komplexní revizi. Námi navrhovaná opatření jsou:
- 1| Modus šifrování CFB nahradit modelem CBC.
 - 2| Kontrolní součet checksum (suma bajtů modulo 65536) nahradit HMAC založeným na SHA-1 nebo na jiné bezpečné hašovací funkci (například SHA-256, 384, 512 apod.).
 - 3| Nový kontrolní součet (HMAC):
 - a| ukládat v délce minimálně 160 bitů;
 - b| vypočítávat jej ze všech dat záznamu Secret Key Packet (nejen z privátních, ale i z veřejných);
 - c| klíč použitý v HMAC derivovat z passphrase jiným způsobem než klíč pro symetrickou šifru;
 - d| šifrovat výsledný HMAC společně s privátními daty symetrickou šifrou podobně, jako je to v případě checksum ve verzi 4 formátu Secret Key Packet.
 - 4| Pro podpisové schéma RSA používat formát typu EMSA-PSS. Toto pravděpodobnostní podpisové schéma účinně brání útokům založeným na chybové analýze, které jsme využili v naší práci.

DŮSLEDKY PRO PRAXI

Kdokoliv, kdo dokáže popsáním způsobem změnit soubor s privátním klíčem, je schopen na základě jediného chybného podpisu získat hodnotu privátního klíče u algoritmů DSA a RSA. K této změně zdaleka nemusí dojít jen na pracovní stanici napadeného uživatele. Citlivým místem jsou i soubory s exportovanými privátními klíči, které uživatel používá k přenosu mezi různými stanicemi. Dostane-li se k takové disketě při její přepravě útočník, je bezpečnost uživatelova privátního klíče vážně ohrožena.

Další scénář je možné použít, jestliže je soubor s privátním klíčem uložen na sdíleném zařízení. Zde může být útočníkem například správce serveru, který na určitou dobu uživateli podstrčí upravenou verzi souboru *secring.skr*, počká, až jej uživatel použije k podpisu (dobu lze poměrně přesně určit monitorováním síťové aktivity uživatelovy stanice), a poté vrátí zpět jeho původní obsah.

Námi popsany útok ukazuje, že v okamžiku, kdy uživatel příslušného programu zjistí, že byla vygenerována chybná hodnota podpisu, měl by se začít chovat velmi obezřetně. Může být právem na rozpacích, zda se jedná o důsledek záměrného útoku nebo „jen“ o technické selhání. Prakticky je ale zřejmé, že každý soubor s neplatným podpisem si zasluhuje stejnou pozornost, jako by se jednalo o soubor obsahující privátní klíč v otevřeném tvaru! To ovšem předpokládá odpovídající péči věnovanou jeho neobnovitelnému odstranění z příslušné stanice, nebo dokonce serveru, což může být někdy tvrdý oříšek.

ZÁVĚR

Poukázali jsme na závažné nedostatky formátu OpenPGP a popsali útoky, které vedou k získání nejcitlivějších informací systému. Upozorňujeme tím na důležitý aspekt ochrany privátních klíčů a veřejných parametrů asymetrických algoritmů v bezpečnostních systémech. Praktickým příkladem je program PGP 7.0.3., na němž jsme platnost hypotéz vyzkoušeli. Je zranitelný pomocí útoku na algoritmus DSA a z pozdějších reakcí vyplynulo, že některé jeho starší varianty nejsou odolné ani proti útoku na algoritmus RSA.

Vlastimil Klíma | v.klima@decros.cz

Tomáš Rosa | t.rosa@decros.cz