Kryptologie pro praxi – jak pašák chránil hesla

Příběh, o který se s vámi zde podělíme, se skutečně stal. Je to již pár let a bezpečnostní díra, která bude popsána dále, byla neprodleně zalepena, takže nebezpečí pro jednu anonymní aplikaci jednoho anonymního informačního systému jedné anonymní úspěšné mezinárodní firmy již nehrozí. Naopak skutečnost, že kdosi byl bez uzardění a následné rituální sebevraždy schopen spáchat to, co za okamžik uvidíme, je velmi dobrým příkladem pro všechny bezpečnostní architekty. Dodejme, že příkladem odstrašujícím.

Pašákova funkce

Hrdinovi našeho příběhu z doby, kdy jsme působili jako konzultanti, bychom mohli říkat třeba programátor všeuměl. Nechceme však přiživovat zkreslený názor, podle kterého je každý programátor nutně hlupák, takže použijeme profesně neutrální výraz pašák. Pokud je nám známo, tak dotyčný pašák stejně pověsil programování na hřebík a věnuje se vyššímu manažerskému poslání. Chválabohu. Nicméně v době aktivního působení potřeboval tento bodrý chlapík nějak vykoumat způsob ověřování uživatelova hesla v jisté aplikaci. Celkem bystře nahlédl, že ukládat kdesi v aplikaci přímo dvojice (*jméno*, *heslo*) a pak pro zadané (*jméno*', *heslo*') porovnat, zda v příslušném záznamu platí *heslo*=*heslo*', nebude to pravé. Došlo mu, že například administrátor systému může tyto záznamy snadno získat a následně zneužít pro přístup k identitě libovolného uživatele. Rozhodl se tedy ukládat záznamy ve tvaru (*jméno*, *h*(*heslo*)), kde *h* je nějaká jednocestná hašovací funkce, takže administrátor ani nikdo jiný nebude pro kvalitní heslo schopen jeho hodnotu z hašového kódu získat. Možnost ověření hesla podle vztahu $h(heslo)=^{?}h(heslo')$ přitom zůstane zachována.

Na obecné úrovni nelze proti takovému přístupu, na kterém například vyrostla základní autentizace v systémech UNIX, nic zásadního namítat. Při konkrétní realizaci je ovšem nutné ohlídat ještě pár detailů. Kromě rozumné kvality hesel, aby administrátor nemohl snadno uspět se slovníkovým útokem, je tu i zásadní důraz na jednocestnost funkce *h*. I tohle ještě pašák věděl, ale bohužel už tak nějak po svém. Místo aby použil libovolnou z plejády v té době používaných hašovacích funkcí (MD5, SHA-1, RIPEMD-160, atp.), rozhodl se, že vytvoří nějakou vlastní, ušitou takříkajíc na míru aplikaci. A tak spatřila světlo světa rodina funkcí h_β, jejíž popis je v *tabulce 1*. Indexový parametr β určující konkrétní funkci měl asi původně sloužit k odlišení různých účelů použití pašákovy funkce. Kdoví. Každopádně to pro nás není podstatné, neboť útočník hodnotu β zná. Vidíme, že funkce zpracovává prakticky libovolně dlouhé heslo na dekadický řetězec pevné délky deseti znaků. To je však bohužel asi jediný neutrální výrok, který lze na její adresu pronést.

Tabulka 1 Pašákova jednocestná funkce h_β

Vstup: řetězec hesla délky L bajtů $P=(P_1, \ldots, P_L)$, $P_i \in \langle 0,255\rangle$.
Výstup: hašový kód $H=(H_1, \ldots, H_{10})$, $H_i \in \langle 0,9\rangle$.
Výpočet:
*) definujme $f_{\beta,L}(i)=1+((49*i+13*\beta) \bmod L)$
1) vypočtěme $s=P_1+P_2+\ldots+P_L$
2) pro $i=1, \ldots, 10$ vypočtěme $H_i=(P_j+\lvert\beta-57*i\rvert+i*s) \bmod 10$, kde $j=f_{\beta,L}(i)$
3) vraťme výsledek $H=(H_1, \ldots, H_{10})$

Takže zaprvé

První, co kryptoanalytikovi lehce naznačí, že pašák to s kvalitou své funkce myslel opravdu upřímně, je použití multiplikativního faktoru 49 v předpisu funkce

$f_{\beta,L}$ použité pro transpozici znaků zpracovávaného hesla. Domníváme se, že pašák chtěl původně trefit 47, což je prvočíslo, ale bohužel minul. Důsledkem je z pohledu útočníka fascinující chování pro sedmiznaková hesla, která v praxi nejsou až tak neobvyklá. Jelikož $7^2=49$, platí $49 \bmod 7=0$, v důsledku čehož potom

$f_{\beta,7}(i)=1+(13*\beta \bmod 7)$

Pátráte po nějaké závislosti na vstupním parametru *i* ? Pátráte marně. Při výpočtu otisku hesla o délce sedmi znaků tak bude uvedená funkce vybírat vždy stejnou souřadnici heslového znaku. Předpokládejme pro ilustraci $\beta=5$, takže $f_{\beta,7}(i)=3$. Předpis pro znaky řetězce hašového kódu se nám tím zjednodušil na

$H_i=(P_3+s*i+\lvert 5-57*i\rvert) \bmod 10$, $1\le i\le 10$.

Vidíme, že celý otisk je kompletně určen dvojicí proměnných ($P_3 \bmod 10$, $s \bmod 10$), což dává nejvýše 100 možností pro jeho hodnotu. Uvažujme nyní útočníka, který sedí před pašákovou aplikací a chce se pomocí hrubé síly přihlásit na účet uživatele, o kterém ví, že si nerozvážně zvolil sedmiznakové heslo. Útočník si nejprve sestaví masku hesla ve tvaru $1\lvert\lvert P_2\rvert\lvert P_3\rvert\lvert 4567$, načež bude postupně zkoušet všechny konkrétní řetězce pro $(P_2,P_3)=(0,0), \ldots, (9,9)$. Lze snadno ukázat, že tyto řetězce může zadávat přímo z klávesnice (jejich překódování do ASCII postup nepokazí) a že tímto způsobem projde všechny možné hodnoty neznámého otisku. Jakmile se strefí, bude přihlášen. Ve střední hodnotě bude za předpokladu rovnoměrného rozdělení neznámého otisku potřebovat asi padesát pokusů. Svého času jsme takový postup skutečně prakticky předvedli a výsledný umělecký dojem byl opravdu hluboký.

Finální trapas

Zahřáti předchozí ukázkou se nyní podívejme, jak je na tom útočník, který by chtěl ze znalosti hašového obrazu odvodit příslušné přihlašovací heslo. Uvidíme, že v tomto směru náš pašák asi už definitivně pozbyl veškerou duchapřítomnost. Mějme tedy dáno $H=(H_1, \ldots, H_{10})$, β a hledejme řetězec hesla $P=(P_1, \ldots, P_L)$ splňující $H=h_\beta(P)$. Poznamenejme, že nemusíme a ani nechceme najít nutně přesně tu hodnotu hesla, kterou napadený uživatel používá. Stačí najít libovolné heslo, dávající stejnou hodnotu otisku *H*. Označme délku hesla *L*. Nelze si nevšimnout, že pro pevnou hodnotu *L* vede náš problém na soustavu 10 lineárních (!) kongruencí o *L* neznámých. Konkrétně:

$P_j+i*(P_1+P_2+\ldots+P_L)\equiv H_i-\lvert\beta-57*i\rvert \pmod{10}$, $j=f_{\beta,L}(i)$, $1\le i\le 10$

Transpozice $f_{\beta,L}$ tak byla jediným pokusem o zavedení nějaké větší složitosti do problému výpočtu heslového vzoru k zadanému hašovému obrazu. Dodejme, že pokusem značně chabým, neboť útočník může postupně zkoušet řešit soustavy indukované pro $L=1, 2, \ldots, 11$, dokud nenajde soustavu, která je řešitelná. Její řešení pak bude hledaným heslem, které útočníka dove-

de k identitě napadeného uživatele. Lze velmi snadno ukázat, že pro L=11 už soustava řešení mít musí, takže nějaké jedenáctiznakové heslo, které dává stejný otisk jako sebekvalitnější a sebedelší skutečné heslo, bude nalezeno velmi záhy. Vlastní soustavu kongruencí můžeme řešit jako dvě soustavy lineárních rovnic, jednu nad tělesem $GF(2)$ a druhou nad $GF(5)$. Zde přitom použijeme stejný přístup, na jaký jsme například zvyklí z řešení „obyčejných" soustav nad tělesem racionálních čísel Q. Jen místo operací (+, *) z Q použijeme jejich analogy z příslušného tělesa. Jak jsme uvedli výše, budeme postupně zkoušet soustavy vznikající pro L=1, 2, …, přičemž zajímat nás bude nejmenší hodnota L, pro kterou budou mít řešení obě soustavy (nad $GF(2)$ i $GF(5)$). Z obou řešení potom vhodným algoritmem vycházejícím z dobře známé Čínské věty o zbytku najdeme řešení platné pro výchozí soustavu kongruencí. Snad už jen pro úplnost uveďme, že to celé na běžném kancelářském PC trvá s přehledem méně než 1 sekundu.

Závěr

Co říci závěrem? Na první pohled by se zdálo, že pašákův pokus o pokoutné oslabení bezpečnosti byl včas odhalen, zasažená aplikace opravena a konečně i sám pašák pochopil, že strmým kariérním postupem prokáže celému lidstvu ohromnou službu, a tak dobrovolně včas vyklidil bitevní pole. Ne náhodou jsme ale v celém textu použili pašáka s malým p. On totiž nebyl, není a nikdy nebude ve své svaté válce prosazující sebestřednou hloupost a pomatenou zbrklost sám. Naopak, někdy to vypadá, že se nám ti pašáci technikou buněčného dělení množí geometrickou řadou. A tak nezbývá, než si na ně dávat dobrý pozor, a sem tam jejich úspěchy náležitě ocenit. Třeba i podobným článkem. Snad trochu silná káva, řeknete si. Možná, ale my bychom byli opravdu velmi neradi svědky toho, jak bezpečnost našeho světa závisí na představě pašáků, že útočníci stejně jako oni neumí řešit lineární rovnice.

Vlastimil Klíma, Tomáš Rosa,
v.klima@volny.cz, trosa@ebanka.cz

LITERATURA

[1] E-archivy http://cryptography.hyperlink.cz, http://crypto.hyperlink.cz

Bude rok 2006 ve znamení HDTV?

Televizní vysílání ve vysoké kvalitě rozlišení HDTV (High Definition TV), přes všechny problémy vyplývající z podmínky distribuce značného objemu dat v krátkém časovém úseku i přes vysokou cenu zobrazovacích jednotek, nalézá mezi televizními diváky, zejména v USA a v Japonsku, stále více příznivců. Metoda komprimace obrazových dat podle standardu MPEG-2 je pro HDTV nepoužitelná, aniž by došlo ke znatelné ztrátě obrazové kvality. Datový tok MPEG-2 lze částečně snížit zařazením vhodné filtrace a předzpracováním signálu před kódováním, tzn. odstraněním videošumu, zkreslení, impulzního šumu a redukcí nadbytečných vysokofrekvenčních složek signálu. Může se však snadno stát, že dojde zároveň i ke zkreslení zdrojového signálu a pak se kvalitou dostaneme na úroveň klasického analogového přenosového kanálu.

Proto byly speciálně pro HDTV vyvinuty nové standardy, z nichž nejperspektivnější jsou kódovací systémy MPEG-4, ve verzi H.264 a Windows Media 9. Jejich mnohem sofistikovanější algoritmy však kladou daleko větší nároky na výkon používaných procesorů. Nicméně už byly předvedeny první funkční vzorky univerzálních Set Top Boxů, které rozpoznají způsob komprimace (MPEG-2 nebo MPEG-4) a podle toho generují výstupní signál SDTV (standardní rozlišení) nebo HDTV.

Stručné shrnutí základních vlastností používaných algoritmů pro kódování videosignálů je uvedeno v *tabulce 1*. MPEG-4 H.264/AVC (Advanced Video Coding) a Windows Media 9 odstraňují redundanci v obrazových datech se srovnatelnými výsledky. Motion JPEG 2000 byl původně vyvíjen speciálně pro digitální kino. Přehled kompresních možností pro zvukový kanál shrnuje *tabulka 2*.

Tabulka 1 Přehled algoritmů pro kódování videa

Standard	tok SDTV	tok HDTV
MPEG-2	3,5 Mb/s	16 Mb/s
MPEG-4 H.264/AVC	1–2,5 Mb/s	8 Mb/s
Windows Media 9	méně než 1 Mb/s	5–7 Mb/s
Motion JPEG 2000	méně než 1 Mb/s	5–7 Mb/s

Tabulka 2 Přehled algoritmů pro kódování zvuku

- MPEG (Layer 2) je považován za standard s dobrými výsledky při přenosových rychlostech 192 kb/s a vyšších, pro prostorový zvuk 5.1 je třeba počítat s objemem dat kolem 500 kb/s,
- MP3 (MPEG Layer 3) je formát určený spíše pro mobilní audiopřehrávače, v broadcastingu se však zatím příliš neprosazuje,
- MP3 Surround 5.1 je novinka, která používá mimořádně zdařilý algoritmus (pro prostorové kódování zvuku vystačí s tokem 128–196 kb/s),
- MPEG-2 AAC (Advanced Audio Coding) zakóduje stereofonní signál do streamu 64 kb/s a při toku kolem 256 kb/s je kvalita přenášeného zvuku srovnatelná se záznamem na CD,
- MPEG-4 AAC patří v současné době k nejpokročilejšímu způsobu kódování zvuku, který využívá systém QuickTime6 (při toku kolem 48 kb/s přenáší poměrně věrný prostorový zvuk).

Při kombinaci komprimovaného videa podle MPEG-4/HE-AAC (High Efficiency AAC) a zvuku podle QuickTime6 lze sledovat v přijatelné kvalitě filmový záznam s prostorovým zvukem zhruba od 500 kb/s. Podobnou variantu kódování zvuku používá i technologie VoIP.

Do konce letošního roku se celosvětově očekává růst ze současných 24 milionů na 30 milionů prodaných přijímačů HDTV a do konce roku 2010 by mohl počet provozovaných přijímačů HDTV překročit magických 100 milionů kusů. Největší poptávka po přijímačích HDTV se očekává v Japonsku a také v USA a v Austrálii, v Evropě se dostává do popředí v rostoucím zájmu o televizní přijímače s vysokým rozlišením především Německo. Takové trendy jsou podporovány i klesajícími cenami zobrazovacích jednotek plazmových i LCD (zhruba o 30 % ročně) obrazovek. Kromě toho se na trhu objevily čipové sady pro dekódování toku podle standardu MPEG-4 H.264 AAC (vyrábí např. firma Sigma Designs). Aby se poskytovatelům a zprostředkovatelům HDTV začaly vracet nemalé už proinvestované prostředky, bude nezbytné datové toky kódovat a programy poskytovat za úplatu. HDTV má naději na úspěch pouze v modelu placené televize. Zlomovým okamžikem v zavádění HDTV v Evropě se může stát zejména pro německý trh Mistrovství světa v kopané 2006. Každý zápas bude zřejmě k dispozici ve formátu SD i HD, a tak budou záznamy i distribuovány k divákům. O HDTV projevují stále větší zájem také výrobci PC. Přibudou čtecí a vypalovací moduly HD-DVD. V této souvislosti se stále více mluví o perspektivách bezdrátového rozhraní HDMI (High Definition Multimedia Interface) s přenosovými rychlostmi 1,5–3 Gb/s.

Jik