

Od rootkitů k bootkitům

Rootkitům jsme se naposledy detailněji věnovali v čísle 6/2005, pojďme se nyní podívat co nového se v této oblasti událo za poslední rok.

Nejprve si krátce připomeňme, co je to rootkit. Tento pojem se objevil v unixovém světě a označoval programy, které umožnily útočnickovi utajit jím způsobené konání; dnes pronikl i do operačního prostředí Windows. Rootkit sice neslouží k iniciálnímu získání administrátorských oprávnění, jakmile však útočník taková práva získá, pomáhá mu rootkit ve skrývání jeho činnosti v systému. Rootkit jako takový jen skrývá určité aktivity (procesy, soubory, síťovou komunikaci) a sám neprovádí škodlivou činnost, může však vést k nestabilitě systému. Navíc to, co rootkit skrývá, typicky škodlivé bývá.

WINDOWS VISTA

Nejvýznamnější událostí ve windowsovém světě v poslední době je zřejmě uvedení operačního systému Microsoft Windows Vista na trh. Vista obsahuje celou řadu bezpečnostních vylepšení, nás však nyní zajímají především možnosti skrývání v operačním systému. Z důvodu ochrany před škodlivým softwarem a narušením DRM systému (digital rights management – digitální správa přístupových práv) musí být všechny ovladače v jádře digitálně podepsány. Zároveň se Vista snaží přesunout řadu ovladačů z jádra do uživatelského prostoru (user-mode drivers) a u některých procesorů chrání paměť jádra dodatečnými způsoby (kernel patch protection – KPP). Některé z těchto vlastností se objevily již v dřívějších verzích Windows, ale ve Vistě jde Microsoft dále a digitální podepisování ovladačů je povinné (pro plnou funkčnost jádra). Ladění jádra je samozřejmě povoleno, to však vyžaduje restart operačního systému a výběr takového režimu ve startovacím menu (následně je omezena funkčnost jádra).

ZNEUŽITÍ ODKLÁDACÍHO SOUBORU

Povinné podepisování ovladačů znamená, že pouhé získání administrátorských oprávnění pro kompromitaci jádra operačního systému nepostačuje. Ani administrátor systému nemůže zavádět do jádra nepodepsané ovladače. Útoky obcházející tuto ochranu však na sebe nenechaly dlouho čekat. J. Rutkowska publikovala záhy [10] způsob, jak obejít digitální podepisování ovladačů bez restartu počítače a bez nutnosti využít nějaké chyby v jádře. Idea je jednoduchá. Operační systém kontroluje digitální podpis pouze při zavádění ovladače, při běhu už žádné kontroly neprovádí. Paměť jádra sice není možné z uživatelského prostoru modifikovat, jakmile je však paměť jádra odložena na disk, nikdo nám nebrání ji na disku modifikovat.

Přímý přístup na disk pak nevyžaduje nic méně a nic více než administrátorská práva. V praxi je pro takový útok nutno vyřešit několik implementačních detailů, bylo však předvedeno, že popsany útok je v praxi proveditelný. Nejprve je nutné vybrat ovladač jádra, který bude změněn (pro demonstraci byl zvolen ovladač NULL.SYS) a přimět jádro kód tohoto ovladače přemístit do odkládacího prostoru na disku. To se dá provést násobnou alokací velkého množství paměti prováděnou tak dlouho, dokud operační systém nezačne odkládat i nepoužívané části jádra. Pak je samozřejmě nutné v odkládacím souboru nahradit kód ovladače, což je možné provést prostým hledáním určitého bitového vzorku.

Jakmile jsme kód ovladače na disku lokalizovali, můžeme přímým přístupem k disku tato data modifikovat a kód ovladače

nahradit vlastním kódem. Pro aktivaci našeho kódu potřebujeme aktivovat přepsaný ovladač. V případě ovladače NULL.SYS stačí požádat o otevření patřičného zařízení pomocí volání CreateFile. Z disku jsou zpět načtena modifikovaná data, je tudíž spuštěn náš kód v režimu jádra a ten může provést cokoliv potřebujeme. Příkladem může být deaktivace kontroly digitálního podpisu ovladačů.

Tato zranitelnost byla odstraněna v RC2 verzi Visty, kdy MS umožnil přímý zápis na disk až po získání zámku pro disk, což není možné, jsou-li na disku otevřeny soubory (např. odkládací soubor). Jinou možností by bylo zakázat odkládání paměti jádra nebo šifrovat odkládací soubor, to by však znamenalo znatelné snížení výkonu. Obě tyto volby je možné ve Vistě nastavit.

BOOTKIT

Ochrana jádra operačního systému Visty pomocí povinného podepisování ovladačů se stala terčem i dalšího útoku – tzv. bootkitu. Na rozdíl od předchozího útoku je v tomto případě potřebný restart systému, útok je však rovněž působivý.

Bootkit je druh rootkitu, který pro svou aktivaci nevyužívá služeb jádra, ale přímo mění zaváděcí proces. Ke slovu se dostává ještě dříve, než BIOS počítače začne zavádět operační systém. BIOS v takovém případě zavede do paměti místo operačního systému bootkit a ten pak modifikuje zaváděcí proces operačního systému tak, aby se přes celou sérii kroků zavádění operačního systému nepozorovaně dostal až do jádra operačního systému. Princip spočívá



v tom, že zavaděč operačního systému nemůže tušit, že počítač je již kompromitován bootkitem, a bootkit může následně zavaděč téměř libovolně modifikovat. Jako každý jiný rootkit může nakonec bootkit modifikovat jádro operačního systému tak, aby skrývalo určité činnosti nebo deaktivovat obranné mechanismy jádra (podobně jako v předchozím případě například digitální podepisování ovladačů nebo mechanismy DRM).

Aby se bootkit dostal v zaváděcím procesu před vlastní zavádění operačního systému, je potřeba buď přímý přístup na disk pro modifikaci zaváděcích oblastí disku nebo fyzický přístup k počítači (pro zavedení například z CD/DVD media, pokud je to povoleno). Zajímavá je možnost i síťového zavádění počítačů a podstrčení zaváděcího obrazu s bootkitem.

Bootkit není specifickou novinkou pro operační systém Windows Vista, bootkity existují i pro starší verze operačních systémů. Ostatně možná si ještě vzpomenete na dávno zapomenuté bootviry, které využívaly zaváděcího procesu ke svému šíření na jiná media. Síla boot-

kitu vyniká až v prostředí Visty, kdy již není možné použít „klasické“ způsoby modifikace jádra. Zaváděcí proces Visty není zcela triviální, aby bootkit dostal svůj kód až do jádra běžícího v chráněném režimu procesoru, musí modifikovat řadu kroků zavádění operačního systému [6]. Po úvodním načtení a spuštění zavaděče v MBR (Master Boot Record – hlavní zaváděcí záznam disku) a zaváděcího sektoru NT (boot sector) musí modifikovat zaváděcí manažer (boot manager, bootmgr.exe) a zavaděč Windows (Windows loader, Winload.exe), aby se konečně dostal do jádra Visty (přes NTOSKRNL.EXE).

Bootkit pro Windows Vistu (tzv. Vbootkit) vytvořili Nitin Kumar a Vipin Kumar [6]. Vbootkit zasahuje významným způsobem do zaváděcího procesu a je specifický pro konkrétní verzi Visty. Jako vzor byl vytvořen Vbootkit pro RC1 i RC2 verzi Visty (ne však pro finální verzi), kód je ale možné modifikovat i pro následující verze či jazykové varianty. Pro demonstraci možností Vbootkitu program provádí periodické zvýšení práv všech spuštěných procesů cmd.exe na práva spuštěného procesu SERVICES.EXE a automaticky spouští TELNET server. Z-

rojový kód ani binární program však nebyl zveřejněn z obavy před možným zneužitím; binární verze však byla poskytnuta některým antivirovým firmám. Bootkity od stejných autorů pro starší verze Windows jsou ke stažení včetně zdrojových kódů na [8].

Obrana vůči bootkitům není úplně snadná. Vůči současnému vzorovému a nepersistentnímu vbootkitu je možné se bránit zákazem zavádění operačního systému z externích médií a sítě. Principiálně však toto není dostatečné a pomůže až tzv. „bezpečné zavádění“ za pomoci TPM.

VIRTUALIZACE

Zajímavé možnosti skrývání přinesla virtualizace, a to nejprve softwarová a později i hardwarová. Začneme virtualizací softwarovou. Ta pracuje tak, že virtuální stroj běží v uživatelském režimu a jakmile je spuštěna privilegovaná instrukce, dojde k výjimce, kterou manažer virtuálního stroje zachytí a následně privilegovanou instrukci emuluje. Tímto způsobem se emuluje komunikace s hardwarem. Příkladem softwarové virtualizace jsou programy VMWare a VirtualPC.

Skupina výzkumníků z Microsoft Research a University of Michigan vyvinula zajímavý rootkit nazvaný Subvirt [5], využívající principy softwarové virtualizace. Subvirt po své aktivaci převádí spuštěný operační systém „za běhu“ do virtuálního stroje a manažer virtuálního stroje běží v minimální verzi operačního systému, která může dále spouštět jiný škodlivý software (zaznamenávání stisknutých kláves, phishing web server apod.). Je zřejmé, že prostředí virtuál-

ního stroje je od manažeru virtuálního stroje izolované a z virtuálního stroje není možné škodlivý software nijak detekovat. Zde je vidět sílu útoků pomocí virtualizace, neboť po přesunutí do virtuálního stroje má původní operační systém jen omezené možnosti detekce toho, co se děje mimo něj. V případě softwarové virtualizace je však přesun operačního systému do virtuálního stroje relativně snadno detekovatelný.

Rootkit SubVirt byl vyvinut ve verzi pro Windows XP a VirtualPC, kdy převádí běžící Windows XP do virtuálního stroje ve VirtualPC a na kořenové úrovni jej nahrazuje minimální verzí Windows XP (která i tak zabírá přes 250 MB) a ve verzi pro Linux a VMWare, kdy běžící RedHat Enterprise Linux 4 převádí do virtuálního stroje pod VMWare a na kořenové úrovni spouští Gentoo Linux v ořezané verzi (opět ale přes 200 MB). Princip virtualizace není omezen na konkrétní platformy, musíme však respektovat fakt, že sám manažer virtuálního stroje potřebuje určitý operační systém pro svůj běh.

Virtualizovaný operační systém sice nemůže přímo detekovat škodlivý software na vyšší úrovni, může se však pokusit o detekci toho, že byl virtualizován. A detekce softwarového virtuálního stroje není příliš náročná, je dokonce pravděpodobné, že si něčeho všimne uživatel sám. Virtuální stroj totiž emuluje jiný hardware než ten, který se v počítači nachází. To může být snadno rozpoznatelné zvláště u grafické karty, a to nejen kvůli nižšímu výkonu, ale třeba i změně chování nebo přímo nefunkčnosti v některých aplikacích (typicky herních). Implementace softwarových virtuálních strojů bývá kvalitní, takže celkový výkon systému nemusí být viditelně nižší, v každém případě ale start operačního systému trvá až několiknásobně déle (neboť je třeba zavést kořenový operační systém, v něm manažer virtuálního stroje a pak virtualizovaný operační systém).

Kromě výkonu potřebuje manažer virtuálního stroje i místo na disku a v paměti, což může také vést k odhalení. Existuje i celá řada exaktnějších metod

detekce běhu v softwarovém virtuálním stroji. Odhlédneme-li od toho, že běžně dostupné manažery virtuálních strojů (jako VirtualPC a VMWare) poskytují celou řadu specifických indikátorů běhu v jejich prostředí, existují i obecnější mechanismy. Například využití neprivilegovaných instrukcí (které tedy nevyvolají výjimku a nejsou ošetřeny manažerem virtuálního stroje), jež se chovají odlišně v nativním a virtuálním prostředí. Mezi takové instrukce patří například `sidt` (získání adresy tabulky deskriptorů přerušení procesoru), `sgdt` a `sldt` [4].

HARDWAROVÁ VIRTUALIZACE

Zatímco softwarově založená virtualizace musí privilegované instrukce spouštěné uvnitř virtuálního počítače emulovat, nové procesory (AMD-V, také známé pod SVM/Pacifica a Intel VT-x) umožňují virtualizaci skutečnou. Právě tato virtualizace může umožnit účinné skrývání škodlivého kódu, ladícího programu či nástroje pro obcházení DRM. Tímto způsobem lze obejít mechanismy všech operačních systémů; útok není specifický jen pro určité softwarové platformy.

Jak vlastně skutečná virtualizace funguje? Procesor zavádí nový příznak (flag) povolující či zakazující virtualizaci a sadu nových instrukcí. Pro vytvoření virtuálního stroje stačí povolit virtualizaci, připravit krátký kód, který bude virtuální stroj kontrolovat a řídit (tzv. hypervisor), a následně spustit instrukci pro vytvoření virtuálního stroje. Takto lze relativně snadno přemístit (a to okamžitě a bez restartu) nevirtualizovaný nativní operační systém do virtuálního stroje tak, že tento operační systém nemá žádnou možnost modifikovat nebo dokonce detekovat hypervisor. Hypervisor na druhou stranu může virtuální stroj monitorovat a modifikovat, může také nezávisle na něm vyvíjet aktivity (například komunikovat po síti).

Aby se zabránilo detekci hypervisoru ve virtualizovaném operačním systému, je třeba vyřešit řadu implementačních detailů. Předně je nutné implementovat možnost další vnořené virtualizace (jinak by bylo možné detekovat, že operační

systém už běží uvnitř virtuálního stroje). Dále je nutné se bránit detekci pomocí měření času. Operační systém by mohl spouštět instrukce, které pravděpodobně budou vyžadovat spuštění hypervisoru, a měřit čas potřebný pro provedení instrukce. Tento čas by byl v případě existence hypervisoru jiný než pokud by tento neexistoval. Problémem však zůstává, jak tento čas měřit. Nej přesnější měření času v cyklech procesoru (RDTSC) může hypervisor modifikovat, rovněž informaci o reálném čase v počítači (RTC) je možné monitorovat a v případě potřeby změnit.

Zbývá tedy možnost bezpečné komunikace s NTP serverem (od serveru získáme čas před provedením problematické instrukce a po ní). V takovém případě ale získáme od NTP serveru čas s přesností na sekundy a pro detekci hypervisoru bychom museli spustit inkriminovanou instrukci řádově milionkrát. A to už může hypervisor rozpoznat a dočasně se deaktivovat. Zajímavou novinkou při měření času je však využití paralelních jader procesorů (jednoho pro sledování běhu času a druhého pro vyvolání měřené emulované instrukce). Jiné způsoby detekce sledují využívání vyrovnávacích mezipamětí procesoru. Techničtější popis možností detekce lze například nalézt v [1], [2], [4], [7].

Aby hypervisor zůstal z virtualizovaného operačního systému zcela nedetekovatelný, musí hypervisor tento operační systém netriviálně monitorovat a v případě potřeby podstrčit upravenou realitu. Čím rozsáhlejší jsou však aktivity hypervisoru, tím více se musí hypervisor skrývat. Útoky pomocí virtualizace jsou zatím jen v plenkách. Řada principů detekce virtualizace a možné obrany na úrovni hypervisoru jsou zatím jen diskutovány v teoretické rovině. S přibývajícimi nápady na detekci hypervisoru narůstá komplikovanost „neviditelného“ hypervisoru a teprve budoucnost ukáže, jestli vůbec bude možné vyvinout naprosto nedetekovatelný hypervisor.

Jinou možností je spoléhat na to, že v implementaci procesoru bude nějaká chyba, díky které bude možné rozlišit, zda

LITERATURA:

- [1] Adams K. *Blue Pill Detection In Two Easy Steps Blog Post*. July 2007. URL: <http://x86vmm.blogspot.com/2007/07/bluepill-detection-in-two-easy-steps.html>.
- [2] Barбора E. *Detecting Blue Pill*. URL: <http://rapidshare.com/files/42452008/detection.rar.html>.
- [3] Embleton S. *Hooking CPUID – A Virtual Machine Monitor Rootkit Framework*. URL: <http://rootkit.com/newsread.php?newsid=758>.
- [4] Garfinkel T. et al.: *Compatibility is Not Transparency: VMM Detection Myths and Realities*. URL: http://www.cs.cmu.edu/~jfrankli/hotos07/vmm_detection_hotos07.pdf.
- [5] King S. T. et al. *SubVirt: Implementing malware with virtual machines*. URL: <http://www.eecs.umich.edu/~pmchen/papers/king06.pdf>.
- [6] Kumar N., Kumar V. *Vbootkit: Compromising Windows Vista Security*. Black Hat Europe 2007. URL: http://www.nvlab.in/files/vbootkit_nitin_vipin_whitepaper.pdf.
- [7] Myers M., Youndt S. *An Introduction to Hardware-Assisted Virtual Machine (HVM) Rootkits*. URL: <http://www.megasecurity.org/papers/hvmrootkits.pdf>.
- [8] NV labs. URL: <http://www.nvlab.in>.
- [9] *The Register: Owning Vista from the boot*. URL: http://www.theregister.co.uk/2007/04/26/vbootkit_authors_interview/
- [10] Rutkowska J. *Subverting Vista Kernel for Fun and Profit*, Black Hat Briefings, 2006. URL: <http://blackhat.com/presentations/bh-usa-06/BH-US-06-Rutkowska.pdf>.
- [11] Rutkowska J. *Virtualization – the other side of the coin*. NLUUG 2007. URL: <http://invisiblethings.org/papers/NLUUG-virtualization.ppt>.
- [12] Rutkowska J., Tereshkin A. *Blue Pill Project*. URL: <http://bluepillproject.org/>.
- [13] Rutkowska J., Tereshkin A.: *IsGameOver(), Anyone? Black Hat, USA 2007*. URL: <http://bluepillproject.org/stuff/IsGameOver.ppt>.
- [14] D. A. D. Zovi: *Hardware Virtualization Rootkits, Black Hat USA, 2006*. URL: <http://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Zovi.pdf>.

je instrukce spuštěna ve virtuálním stroji či nikoliv. Takové procesory a instrukce existují, detekce je ale specifická jen pro konkrétní model procesoru.

OBRANA PŘED ÚTOKY VIRTUALIZACÍ

Jak se takovým útokům bránit? Jako u každého jiného rootkitu je nejprve nutné získat práva administrátora, takže prevence může směřovat tímto směrem. U útoků virtualizací je potřebné spouštět kód v režimu jádra (ring 0), kde samotná administrátorská práva nestačí, spuštění kódu v tomto prostředí tedy může u některých systémů vyžadovat netriviální úsilí (viz ochrana jádra Visty popsána výše).

Pokud jde o konkrétní prevenci před útoky virtualizací, je samozřejmě možné virtualizaci deaktivovat nebo pořízovat pouze procesory, které skutečnou virtualizaci nepodporují, to však nemusí být vždy žádoucí řešení. Jinou zajímavou možností je instalovat hypervisor dříve než to udělá někdo jiný a nastavit jej tak, že zavedení dalších hypervisorů už nepovolí, případně povolí zavedení jen důvěryhodných hypervisorů (zde však nemusí být snadné zjistit, který hypervisor je důvěryhodný). Po zveřejnění možnosti virtualizačních útoků zareagoval výrobce procesorů AMD úpravou specifikace, kdy přidal instrukci, která umožňuje „zaheslovat“ virtualizaci až do resetu procesoru. Takové

heslo by mohl vygenerovat operační systém a „zamykal“ by procesor ihned při zavádění operačního systému. Nebude-li rootkit znát toto heslo, nebude moci virtualizaci využít. Bude-li však operační systém heslo někde ukládat, může jej rootkit (alespoň teoreticky) získat také.

BLUE PILL A VITRIOL

Možnosti skrývání pomocí hypervisoru za pomoci využití skutečné virtualizace u AMD SVM/Pacifica procesorů byly poprvé prezentovány J. Rutkowskou ([10], [11]), která tento útok nazvala Blue Pill a která také předvedla konkrétní implementaci útoku ve formě (z virtualizovaného operačního systému) nedetekovatelných síťových zadních vrátek. Nezávisle na J. Rutkowské prezentoval analogické možnosti skrývání na procesorech Intel VT-x Dino Dai Zovi [14]. V obou případech však autoři nedali k dispozici binární nebo zdrojové kódy předváděných rootkitů. To vyvolalo určité diskuze ohledně skutečné nedetekovatelnosti takových rootkitů. V nedávné době byly zveřejněny vzorové rootkity, založené na hardwarové virtualizaci pro obě platformy (AMD-V [12] i Intel VT-x [3]).

JE VIRTUALIZACE JEN ŠPATNÁ?

Ve všech případech virtualizace jde o „boj“, kdo bude kontrolovat nižší vrst-

vu. Pokud se to podaří škodlivému softwaru, má převahu nad detektory. Nižší vrstvy virtualizovaného systému však lze využít i v neprospěch škodlivého softwaru buď pro jeho detekci či například pro ladění škodlivého softwaru, který je vybaven účinnými, ladění bránci opatřeními. Virtualizace má samozřejmě celou řadu jiných legitimních aplikací, to je nakonec důvod, proč výrobci procesorů hardwarovou virtualizaci zavádějí. Proto je také potřeba si uvědomit, že detekce virtualizace se nerovná detekci škodlivého softwaru, to je podstatně náročnější úkol [13].

ZÁVĚR

Poslední rok přinesl řadu zajímavých novinek v oblasti skrývání se v operačních systémech. Ve většině případů šlo „jen“ o vzorové principy, skutečný škodlivý software zatím těchto principů nevyužil (nebo alespoň publikován). Řada principů však má potenciál brzy se ve škodlivém softwaru opravdu objevit. Ne vždy jsou však tyto útoky dávány do souvislosti jen se škodlivým softwarem. V případě útoků na jádro Visty se mluví i o „navrácení počítače jeho vlastníkov“ [9], neboť až tyto útoky umožní vlastníkům počítačů bez omezení spouštět kód v jádře Visty.

ZDENĚK ŘÍHA
zdenek.riha@jrc.it

**ING. MGR. ZDENĚK ŘÍHA, PH.D.**

Odborný asistent na Fakultě informatiky Masarykovy univerzity v Brně. V současné době je na dlouhodobé stáži ve Společném výzkumném centru (JRC) Evropské komise v italské Ispře, kde pracuje na projektech souvisejících s bezpečností hranic.

MANAGEMENT SUMMARY

Metody skrývání moderních rootkitů se neustále vylepšují, to však platí i pro jejich detektory. V poslední době se diskutují především možnosti skrývání za pomoci virtualizace (ať už softwarové nebo hardwarové), kdy má virtualizovaný operační systém jen malé možnosti zjistit co se děje v počítači mimo jeho virtuální stroj. Další zajímavou možností skrývání přináší bootkity, které připomínají dřívější bootviry.