ANÁLISE DE *EXPLOITS* PARA APLICAÇÕES WINDOWS

Welington Gomes da Silva[1]; Fabrício Sérgio de Paula[2]

[1]Estudante do Curso de Ciência da Computação da UEMS; Unidade Universitária de Dourados; E-mail: 017389@comp.uems.br; Bolsista UEMS

[2]Professor do Curso de Ciência da Computação da UEMS; Unidade Universitária de Dourados; E-mail: fabricio.paula@gmail.com

Redes de Computadores e Segurança Computacional

Resumo

Este projeto propõe analisar *exploits* para aplicações que executem sob o sistema operacional Windows. Serão analisadas as ações realizadas por *exploits* durante a fase inicial de execução, de forma a identificar os eventos mais frequentes durante ataques. De posse dessas informações, é possível, em um trabalho futuro, realizar o projeto de um IDS adequado para detectar ataques a aplicações nesse sistema.

Palavras-chave: *Buffer overflow*. Máquina virtual. Vulnerabilidade.

Introdução

Diversas tecnologias têm sido empregadas visando garantir os quesitos de segurança computacional, incluindo *firewalls*, mecanismos criptográficos e sistemas de detecção de intrusão (Nakamura & de Geus, 2003). Um sistema de detecção de intrusão (IDS) é especialmente interessante porque permite identificar a ocorrência de ataques em um computador ou rede de computadores. Devido às suas características, o IDS tornou-se uma tecnologia essencial para administrar a segurança de corporações.

Um IDS atuando no nível de pacotes é bastante útil para identificar ataques. Entretanto, sua capacidade pode ser limitada, especialmente ao lidar com tráfego cifrado. Apesar de, nesse nível, haver essa limitação, um ataque com sucesso irá produzir comportamento inadequado em processos do sistema operacional atacado (de Paula, 2004). Por essa razão, a identificação de ataques no nível de aplicação pode ser mais precisa, embora em alguns casos possa ser mais tardia.

A definição de *exploits* encontrada em (Northcutt *et al*., 2001), se trata de ataques projetados para explorar fraquezas em aplicações de modo a comprometer o sistema atacado.

A análise de *exploits* para aplicações Microsoft Windows[1] possibilitará entender qual o comportamento inicial de processos atacados nesse tipo de sistema.

Este trabalho analisou *exploits* para aplicações que executem sob os sistemas operacionais da família Windows. Foram analisadas as ações realizadas por *exploits* durante a fase inicial de execução, de forma a identificar os eventos mais frequentes durante ataques.

De posse dessas informações, é possível, em um trabalho futuro, realizar o projeto de um IDS adequado para detectar ataques a aplicações nesse sistema. Todo o trabalho foi desenvolvido em ambiente seguro, através do uso de máquinas virtuais (Silberschatz, *et al*., 2004), sem qualquer prejuízo a outros sistemas na rede local ou mesmo na Internet.

Material e Métodos

Como parte das atividades propostas neste projeto, foram realizados estudos e experiências práticas envolvendo a execução de aplicações em máquinas virtuais e o monitoramento da execução de aplicações.

As fontes de referências bibliográficas incluem artigos disponíveis na Internet, e o acesso às bibliotecas da UEMS e UFGD. Também foram utilizados tutoriais de acesso gratuito, como os disponíveis em (Microsoft, 2003-2010).

As atividades práticas foram realizadas através do uso de máquinas virtuais e ferramentas de monitoramento gratuitas como Process Monitor[2] para o Windows XP. O computador utilizado para monitoração tinha as seguintes configurações: processador QuadCore Intel Core i7 920 2.66GHz, 6GB de memória RAM 1600MHz e um disco rígido com 120GB SATA II 7200rpm.

Resultados e Discussão

A ideia básica de uma máquina virtual é a abstração do hardware de um único computador em vários ambientes de execução (Silberschatz, *et al.*, 2004). Os níveis de abstração permitem que detalhes de implementação em níveis inferiores fiquem protegidos dos níveis superiores (Smith, 2005).

Utilizando técnicas de escalonamento de CPU (Unidade Central de Processamento) e de memória virtual, os programas de máquinas virtuais simulam outro computador com sua própria CPU e memória principal.

---

[1] Detalhes do sistema operacional Windows podem ser encontrados em http://windows.microsoft.com/pt-br/windows/products.

[2] Mais informações sobre a ferramenta pode ser obtida em http://technet.microsoft.com/en-us/bb896645.aspx.

Desta maneira, é criado um processo virtual que geralmente é um novo sistema operacional, tornando assim possível a instalação de vários sistemas operacionais em um único computador (Craig, 2005).

Com a utilização da máquina virtual é possível estabelecer restrições, que garantem a segurança do sistema operacional hospedeiro (Castro, 2006). Sendo assim é possível fazer testes com *exploits* em aplicações na máquina virtual, não afetando o sistema hospedeiro, outras máquinas virtuais e o *hardware*.

Alguns programas podem ser utilizados nas máquinas virtuais para monitorar o funcionamento de processos, por exemplo, o Process Monitor e o Process Explorer, realizam monitoramento de processos. Essas ferramentas também podem ser utilizadas para analisar o comportamento de programas maliciosos como os vírus e *exploits*.

Exploits são scripts e programas que tem como função detectar e explorar falhas em sistemas operacionais e aplicações. Para administradores de redes os *exploits* podem ser utilizados para realizar buscas por falhas, por outro lado pessoas mal intencionadas usam os *exploits* para obter informações ou causar danos a computadores desprotegidos (Ulbrich, 2009).

Diariamente são descobertas novas vulnerabilidades em vários aplicativos. Os mais visados são os aplicativos para plataforma Windows. Mais de 89% dos PCs do mundo rodam alguma versão do Windows, distribuída pela Microsoft.

Um sistema operacional deve ter mecanismos de proteção e segurança. Uma aplicação com mau comportamento deve ser detectada pelo sistema. Nenhuma aplicação está livre de falhas, as mais conhecidas e exploradas são segundo Silberschatz: a quebra de confidencialidade envolve leitura ou roubo de informações não autorizadas de dados; quebra de integridade envolve modificações não autorizadas de dados ou códigos; na quebra de disponibilidade, invasores mal intencionados alteram dados apenas por diversão ou reconhecimento.

Outras falhas como roubo de serviço é o uso não autorizado de recursos de uma máquina invadida. Na recusa de serviço o *hacker* promove o impedimento do uso de serviço da máquina, deixando a máquina inalienável (Ulbrich, 2009).

Exemplos de ataques são muitos, o mais recente e famoso se refere ao ataque em massa à rede de videogames PSN (*PlayStation Network*) da Sony, onde *hackers* invadiram servidores da empresa e roubaram informações pessoais de milhares de clientes, tais como nome, número de contas bancárias entre outros. Outro ataque comum e sem intuito de roubo de informações é a modificação ou desfiguração de *websites*.

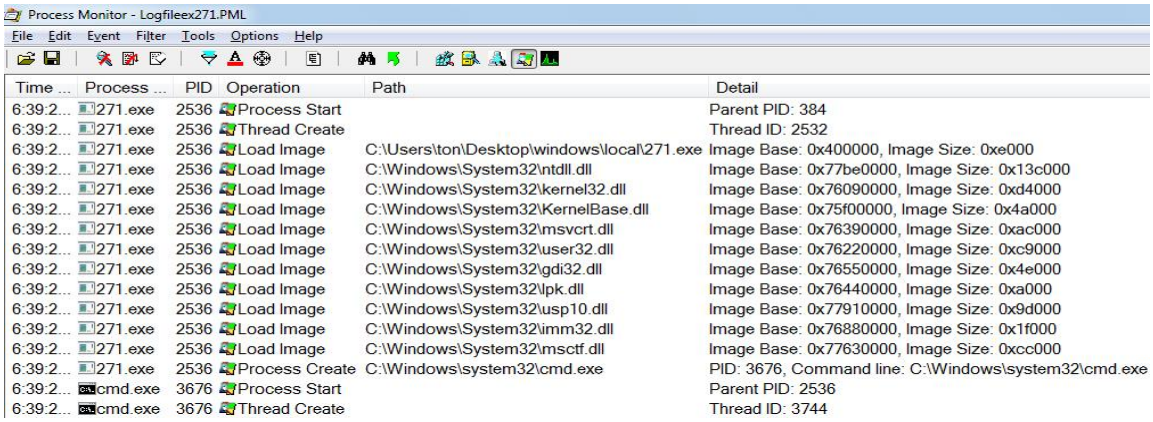
Outro *software* alvo dos criminosos é o Adobe Reader. Em 2009 a própria Adobe, fabricante do aplicativo, alertou seus usuários sobre uma falha crítica. A falha explorada fazia com que o aplicativo travasse e o invasor tomasse conta do sistema. A técnica usada era o estouro de pilha (*buffer overflow*).

Técnicas de *buffer overflow* são amplamente usadas para realizar ataques de vulnerabilidades. Trata-se de um estouro de pilha onde uma variável não trata corretamente sua capacidade de memória, onde *exploits* sobrescrevem códigos que alteram parâmetros de retorno, sendo possível apontar para códigos maliciosos.

A coleta de *exploits* foi realizada no repositório online *Exploit Database*[3], onde são disponibilizados *exploits* para várias plataformas, inclusive Windows, que são criados por usuários e disponibilizados no site.

Dentre outros *exploits* analisados, o *exploit* 271 traz de forma mais clara sua execução. O comando no *buffer* do *exploit* faz a criação de um novo processo *shell* do Windows, o cmd.exe, por onde é possível a execução de programas pela linha de comando.

A execução do *exploit* 271 pode ser vista na Figura 1 a seguir. Ele carrega *dlls* do diretório *system32* e cria um novo processo com privilégios de usuário, tomando conta do sistema.



Time ...	Process ...	PID	Operation	Path	Detail
6:39:2...	271.exe	2536	Process Start		Parent PID: 384
6:39:2...	271.exe	2536	Thread Create		Thread ID: 2532
6:39:2...	271.exe	2536	Load Image	C:\Users\ton\Desktop\windows\local\271.exe	Image Base: 0x400000, Image Size: 0xe000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\ntdll.dll	Image Base: 0x77be0000, Image Size: 0x13c000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\kernel32.dll	Image Base: 0x76090000, Image Size: 0xd4000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\KernelBase.dll	Image Base: 0x75f00000, Image Size: 0x4a000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\msvcrt.dll	Image Base: 0x76390000, Image Size: 0xac000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\user32.dll	Image Base: 0x76220000, Image Size: 0xc9000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\gdi32.dll	Image Base: 0x76550000, Image Size: 0x4e000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\lpk.dll	Image Base: 0x76440000, Image Size: 0xa000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\usp10.dll	Image Base: 0x77910000, Image Size: 0x9d000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\imm32.dll	Image Base: 0x76880000, Image Size: 0x1f000
6:39:2...	271.exe	2536	Load Image	C:\Windows\System32\msctf.dll	Image Base: 0x77630000, Image Size: 0xcc000
6:39:2...	271.exe	2536	Process Create	C:\Windows\system32\cmd.exe	PID: 3676, Command line: C:\Windows\system32\cmd.exe
6:39:2...	cmd.exe	3676	Process Start		Parent PID: 2536
6:39:2...	cmd.exe	3676	Thread Create		Thread ID: 3744

Figura 1 – Execução *exploit* 271.

Mais de 20 *exploits* foram analisados, em sua maioria se comportam como o *exploit* da Figura 1, fazendo acesso as *dlls* do diretório *system32* e criando novos processos.

Conclusões

O desenvolvimento do projeto atingiu o objetivo proposto que foi analisar o comportamento de *exploits* durante sua execução. A maioria deles trata de criar processos com o mesmo nome dos processos do próprio Windows e acessar *dlls* do sistema.

[3] Disponível em: http://www.exploit-db.com/

O estudo sobre máquinas virtuais faz-se de grande importância para realizar o projeto, já que muitas das vezes os arquivos de sistemas eram corrompidos após a execução de alguns *exploits*, no caso de negação de serviços. Com a ferramenta de *snapshot* do programa de máquina virtual foi possível restaurar o sistema sempre que necessário.

Uma dificuldade trata-se do Windows ser um programa proprietário e sem código aberto, dificultando a criação de códigos para monitorar as chamadas de sistema, mas com o uso do Process Monitor é possível monitorar grande parte da execução dos processos e dos *exploits*.

Agradecimentos

Agradeço pelo apoio familiar e apoio financeiro concedido pela Universidade Estadual de Mato Grosso do Sul e também por todos os acadêmicos, coordenadores e professores que, diretamente ou indiretamente, contribuíram para a realização desse trabalho.

Referências

Castro, A. Bispo de., *Máquinas virtuais em ambientes seguros*. Tese de mestrado, Instituto de Computação, Universidade Estadual de Campinas, 2006.

Craig, Iain D. *Virtual machines* – Springer, 2005.

de Paula, F. *Uma arquitetura de segurança computacional inspirada no sistema imunológico*. Tese de doutorado, Instituto de Computação, Universidade Estadual de Campinas, 2004.

Microsoft. *Microsoft Windows administration tutorials*. Disponível em maio de 2010 na URL http://www.tutorialized.com/tutorials/Windows/Administration/1, 2003-2010.

Nakamura, E., de Geus, P. *Segurança de redes em ambientes cooperativos*. Editora Futura, São Paulo, 3ª edição, 2003.

Northcutt, S., Cooper, M., Fearnow, M., Frederick, K. *Intrusion signatures and analysis*. New Riders Publishing, 1ª edição, 2001.

Silberschatz, A., Galvin, P., Gagne, G. *Fundamentos de sistemas operacionais*. LTC, 6ª edição, 2004.

Smith, James E., Nair, R. *Virtual machines* – Elsevier, 2005.

Ulbrich, Henrique César. *Universidade hacker* – São Paulo: Digerati Books, 2009.