

Author: Inj3cti0n P4ck3t

Date: 11/10/10

Nome do Artigo: Criando Scanner para Dectar FTPs vulneráveis ao Metasploit

Contato: fer_henrick@hotmail.com

Linguagem de Programação: Perl (Practical Extraction and Report Language)

0x01 Introdução

O Metasploit Framework é uma ferramenta utilizada para explorar falhas em diversos tipos de softwares e serviços.

Qualquer usuário pode utilizar o Metasploit Framework para atacar um sistema operacional que executa um determinado serviço. Mas a diferença em usar o Metasploit, está na conhecimento e no sucesso de explorar falhas de segurança.

Uma técnica é atacar o servidor com todos os exploits do Metasploit.

Outra técnica é desenvolver um scanner para identificar serviços vulneráveis aos exploits do Metasploit Framework.

No artigo, escrevo como desenvolver um scanner na linguagem de programação PERL que identifica servidores FTPs vulneráveis ao software Metasploit Framework.

0x02 Versões de FTPs vulneráveis aos exploits do Metasploit Framework

O Metasploit possui muitos exploits para explorar falhas em serviços. Mas vamos escrever um scanner na linguagem de programação PERL que identifique o serviços de FTP vlneráveis para o sistema operacional Windows.

Os serviços de FTP identificados pelo scanner são vulneráveis aos exploits do Metasploit Framework 3.3.

A versões de FTPs vulneráveis aos exploits do Metasploit Framework e que serão dentificadas pelo scanner em PERL estão listadas:

- **Easy File Sharing FTP Server 2.0 PASS Overflow**

- **Exploit:** easyfilesharing_pass

- **NetTerm NetFTPD USER Buffer Overflow**

- **Exploit:** netterm_netftpd_user

- **KarjaSoft Sami FTP Server v2.02 USER Overflow**
- **Exploit:** sami_ftpd_user

- **3Com 3CDaemon 2.0 FTP Username Overflow**
- **Exploit:** 3cdaemon_ftp_user

- **HTTPDX tolog() Function Format String Vulnerability**
- **Exploit:** httpdx_tolog_format

- **EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow**
- **Exploit:** easyftp_cwd_fixret

- **Vermillion FTP Daemon PORT Command Memory Corruption**
- **Exploit:** vermillion_ftpd_port

- **Xlink FTP Client Buffer Overflow**
- **Exploit:** xlink_client

- **Cesar FTP 0.99g MKD Command Buffer Overflow**
- **Exploit:** cesarftp_mkd

- **Oracle 9i XDB FTP UNLOCK Overflow (win32)**
- **Exploit:** oracle9i_xdb_ftp_unlock

- **Serv-U FTPD MDTM Overflow**
- **Exploit:** Serv-U FTPD MDTM Overflow

- **freeFTPD 1.0 Username Overflow**
- **Exploit:** freeftpd_user

- **SlimFTPD LIST Concatenation Overflow**
- **Exploit:** slimftpd_list_concat

- **Ipswitch WS_FTP Server 5.05 XMD5 Overflow**
- **Exploit:** wsftp_server_505_xmd5

0x03 Softwares para testar o Scanner

Metasploit Framework 3.3

Download: <http://www.metasploit.com/releases/framework-3.4.1.exe>

Active Perl (Interpretador Perl para Windows)

Download:

<http://downloads.activestate.com/ActivePerl/releases/5.12.2.1202/ActivePerl-5.12.2.1202-MSWin32-x86-293621.msi>

EasyFTP Server 1.7.0.2 (FTP vulnerável para testar o Scanner)

Download: <http://easyftpsvr.googlecode.com/files/easyftpsvr-1.7.0.2.zip>

0x04 Desenvolvendo um Scanner para Dectar FTPs vulneráveis aos exploits do Metasploit

Se você usa sistema operacional Linux, o caminho dos scripts em Perl será `usr/bin/perl`, como apresento abaixo.

```
#!/usr/bin/perl
```

Se você fez download do Active Perl e instalou no sistema operacional Windows, o caminho para inserir o script que estamos desenvolvendo será `c:\perl\bin`.

```
#!C:\Perl\Bin
```

Os Módulos usados no desenvolvimento do scanner.

Vamos utilizar o módulo **IO::Socket** e **IO::Socket::INET** para fazer a conexão com o servidor FTP.

```
use IO::Socket;  
use IO::Socket::INET;
```

A linha de condição "IF" é executada.

Se o usuário não informou a lista com IPs ou sites para scanear as instruções contidas entre chaves ou no bloco são executadas.

A variável "\$sis" recebe "\$^O" para verificar o sistema operacional: Linux ou Windows.

Se o sistema operacional é Windows o comando de sistema "cls" será executado.

Se o sistema operacional é Linux o comando de sistema "clear" será executado.

O comando "print q { ... }" apresenta as linhas ou o banner de como usar o scanner.

A linha "exit();" finaliza o programa.

```
if (!$ARGV[0]) {  
  
    $sis="$^O";if ($sis eq windows){ $cmd="clear";} else { $cmd="cls"; }  
    system("$cmd");  
    print q {
```

Code desenvolvido por: Inj3cti0n P4ck3t

e-mail to contact: fer_henrick@hotmail.com

Modo de uso: perl ScanMetasploitFTP.pl listaDelps_ou_ListaDeSites.txt

```

    Nome: Fernando Henrique Mengali de Souza
};

exit();
}

```

A linha abaixo, abre o arquivo de sites ou IPS informado pelo usuário, caso não seja possível abrir o arquivo, a mensagem: "Nao foi possível abrir o arquivo:" é apresentada.

Geralmente, um arquivo não pode ser aberto pelo seguinte:

- O arquivo não está no diretório do script
- O nome do arquivo está errado
- O arquivo pode estar corrompido

```

open( SITE, "< $ARGV[0]" ) or die( "Nao foi possível abrir o arquivo: $" );
# Abri o arquivo .txt informado pelo usuário

```

Atribuímos ao array "**@array**" a lista de sites ou IPs para scanear.

A variável "**\$numero**" recebe o último vetor, pois será usado no laço de interação "**for**".

Após a variável **\$numero** receber o último array, na próxima linha temos um "**for**".

O "**for**" começará a ser executado:

```

our @array = <SITE>; # array recebe o sites contidos no arquivo

$numero = $#array; # $numero possui o ultimo array

for ($i = 0; $i <= $numero; $i++) {    # inicia-se o laço for

```

Vamos usar a variável "**\$Url**" para armazenar o endereço IP ou do site alvo.

Se o endereço alvo não possui o protocolo HTTP, usamos um "**IF**" como condição.

Se endereço não possui HTTP, o if inseri HTTP. Exemplo:

```

Não possui o protocolo HTTP
192.168.0.3

```

O "**IF**" verifica o endereço 192.168.0.3, não possui o protocolo HTTP. Então, inseri:
http://192.168.0.3

O endereço IP foi verificado pelo IF, o resultado foi inserir o protocolo HTTP:

```

$Url = "$array[$i]";

if($Url !~ /http:\V\/) { $Url = "http://$Url"; }

```

Agora, formatamos o endereço alvo usando o código abaixo:

```
$Stop = index($Url,":");
$Protocolo = substr($Url,0,$Stop);
$Start = index($Url,"/") + 2;
$Dominio = substr($Url,$Start);
$Stop = index($Dominio,"/");
$Dominio = substr($Dominio,0,$Stop);
$Start = rindex($Url,"/") + 1;
$NomeArq = substr($Url,$Start);
$Compr_Url = length($Url);
```

\$ponto = "\$Dominio"; #A variável \$ponto, recebe a \$Dominio ou URL formatada.

Criamos um simples array para armazenar a porta 21.

Depois criamos um “**foreach**”, atribuindo a variável “**\$porta**” o valor **21**.

```
our @portas = "21";
```

```
foreach $porta (@portas) {
```

Vamos criar a variável **\$sock**, que armazenará a conexão com o IP ou site armazenado na posição do array e a porta de conexão: 21.

Na próxima linha, usamos a variável \$sock como expressão para ser avaliada, ou seja, se a conexão for estabelecida com o IP e porta 21 o bloco será executado.

Caso o a expressão não retornar verdadeiro(TRUE), porque não conseguiu conectar ao alvo na porta 21, nenhuma informação é retornada no terminal.

```
$sock = IO::Socket::INET->new("$ponto:$porta");
```

```
if($sock) {
```

Se a conexão retornar TRUE, o bloco do "IF" é executado, portanto a variável \$remote terá uma conexão com o servidor alvo, usando o protocolo TCP e com Timeout 7.

A variável \$line agora possui resposta da conexão, ou seja, "o banner".

Sabemos que a variável \$line possui a resposta do servidor que conectamos, ou seja, temos o banner do servidor que conectamos.

```
$remote = IO::Socket::INET -> new (Proto => "tcp", PeerAddr => $ponto, PeerPort => $porta,
Timeout => "7");
```

```
$line = <$remote>;
```

Quando a expressão avaliada pelo comando "IF" resultar em verdadeiro, isto é, quando o operador "=" verificar se o conteúdo armazenado na variável corresponde ao texto ou frases entre aspas. o bloco será executado.

O bloco corresponde ao print do IP ou site com o servidor vulnerável e banner do servidor vulnerável.

Caso, não haja servidor vulnerável aos banners informados não será apresentado nenhuma mensagem.

```
if ($line =~ "Easy File Sharing FTP Server") {

# Se o conteúdo da variável $line é "Easy File Sharing FTP Server", o servidor é vulnerável
    print "$ponto -> Easy File Sharing FTP Server 2.0 PASS Overflow\n";
# O endereço alvo é apresentado na tela com a versão do FTP
}

if ($line =~ "NetTerm FTP server") {

# Se o conteúdo da variável $line é "Easy File Sharing FTP Server", o servidor é vulnerável
    print "$ponto -> NetTerm NetFTPD USER Buffer Overflow\n";
# O endereço alvo é apresentado na tela com a versão do FTP
}

if ($line =~ "Sami FTP Server 2.0.2") {

# Se o conteúdo da variável $line é igual a "ESami FTP Server 2.0.2", o servidor é vulnerável
    print "$ponto -> KarjaSoft Sami FTP Server v2.02 USER Overflow\n";
# O endereço alvo é apresentado na tela com a versão do FTP
}

if ($line =~ "3Com 3CDaemon FTP Server Version 2") {
# Se o conteúdo da variável $line é "3Com 3CDaemon FTP Server Version 2", o servidor é
#vulnerável

    print "$ponto -> 3Com 3CDaemon 2.0 FTP Username Overflow \n";
# O endereço alvo é apresentado na tela com a versão do FTP
}

if ($line =~ "httpdx") {
# Se o conteúdo da variável $line é igual a "httpdx", o servidor é vulnerável

    print "$ponto -> HTTPDX tolog() Function Format String Vulnerability\n";

# O endereço alvo é apresentado na tela com a versão do FTP
}

if ($line =~ "BigFoolCat") {
# Se o conteúdo da variável $line é igual a "BigFoolCat", o servidor é vulnerável

    print "$ponto -> EasyFTP Server 1.7.0.11 CWD Command Stack Buffer";
    print "Overflow\n";
# O endereço alvo é apresentado na tela com a versão do FTP
}
```

```

        if ($line =~ "vftpd") {
# Se o conteúdo da variável $line é igual a "vftpd", o servidor é vulnerável

print "$ponto -> Vermillion FTP Daemon PORT Command Memory Corruption\n";
# O endereço alvo é apresentado na tela com a versão do FTP
        }

        if ($line =~ "XLINK FTP Server") {
# Se o conteúdo da variável $line é igual a "XLINK FTP Server", o servidor é vulnerável
        print "$ponto -> Xlink FTP Client Buffer Overflow \n";
# O endereço alvo é apresentado na tela com a versão do FTP
        }

        if ($line =~ "CesarFTP 0.99g") {
# Se o conteúdo da variável $line é igual a "CesarFTP 0.99g", o servidor é vulnerável

        print "$ponto -> Cesar FTP 0.99g MKD Command Buffer Overflow \n";
# O endereço alvo é apresentado na tela com a versão do FTP
        }

        if ($line =~ "9.2.0.1.0") {

# Se o conteúdo da variável $line é igual a "9.2.0.1.0", o servidor é vulnerável

        print "$ponto -> Oracle 9i XDB FTP PASS Overflow (win32)\n";
# O endereço alvo é apresentado na tela com a versão do FTP
        }

        if ($line =~ "Serv-U FTP Server v4.0" || $line =~ "Serv-U FTP Server v4.1" ||
$line =~ "Serv-U FTP Server v5.0" || $line =~ "220 ProFTPD 1.2") {

# Se o conteúdo da variável $line é igual:
# Serv-U FTP Server v4.0
# Serv-U FTP Server v4.1
# Serv-U FTP Server v5.0, o servidor é vulnerável
        print "$ponto -> Serv-U FTPD MDTM Overflow \n";
# O endereço alvo é apresentado na tela com a versão do FTP
        }

        if ($line =~ "freeFTPd 1.0") {
# Se o conteúdo da variável $line é igual a "freeFTPd 1;0", o servidor é vulnerável

        print "$ponto -> freeFTPd 1.0 Username Overflow\n";
# O endereço alvo é apresentado na tela com a versão do FTP
        }

        if ($line =~ "SlimFTPd Server") {
# Se o conteúdo da variável $line é igual a "SlimFTPd Server", o servidor é vulnerável

```

```

        print "$ponto -> SlimFTPD LIST Concatenation Overflow\n";
# O endereço alvo é apresentado na tela com a versão do FTP
    }

    if ($line =~ "WS_FTP Server 5.0.5") {
# Se o conteúdo da variável $line é igual a "WS_FTP Server 5.0.5", o servidor é vulnerável

        print "$ponto -> Ipswitch WS_FTP Server 5.05 XMD5 Overflow\n";
# O endereço alvo é apresentado na tela com a versão do FTP
    }
}
}
}
}

```

0x05 Testando o scanMetasploitFTP.pl no Laboratório

Vamos executar o scanner informando um lista com 3 IPs locais para scannear.

1º 192.168.0.13

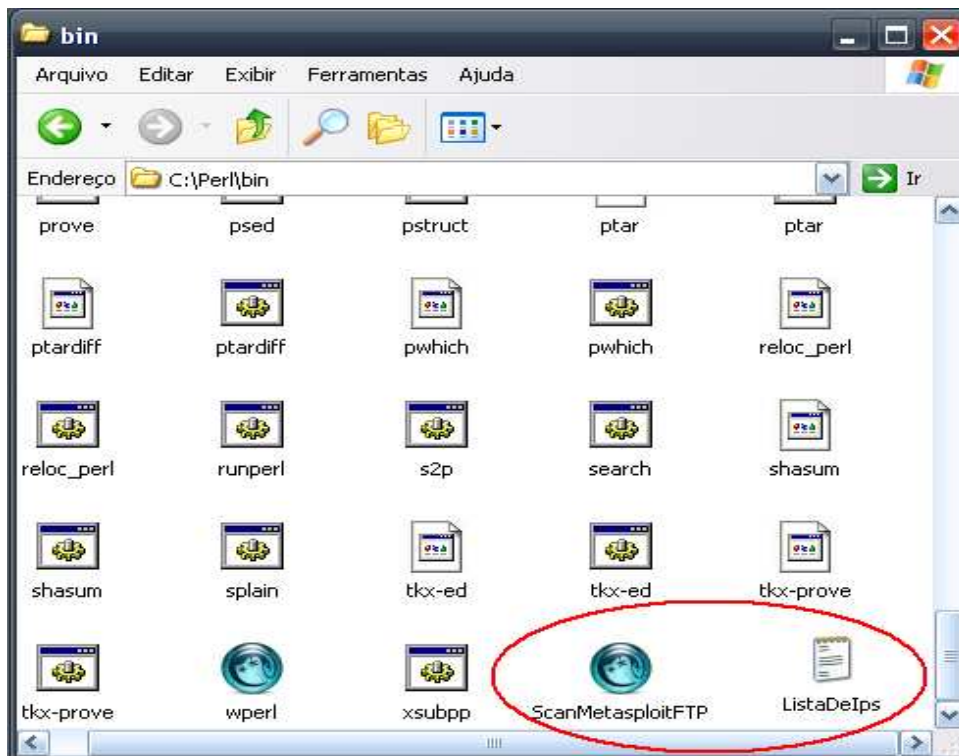
2º 127.0.0.1

3º 192.168.0.13

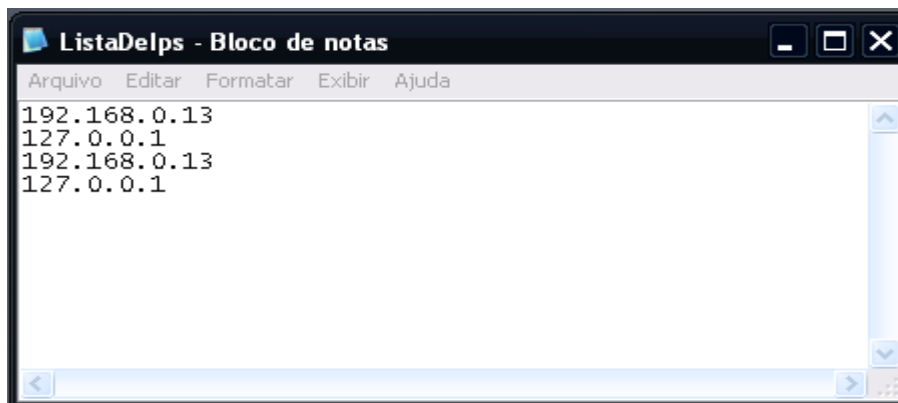
4º 127.0.0.1



1.0 Execute o software EasyFTP no sistema operacional Windows.



2.0 Insira o script ScanMetasploitFTP.pl e lista de IPs ou sites na pasta correta



2.2 Lista de IPs com 4 ips para scanear

Se o sistema operacional é Linux: `#!/usr/bin/perl`

Se o sistema operacional é Windows: `#!/c:\perl\bin`

```
C:\WINDOWS\system32\cmd.exe

C:\Perl\bin>perl ScanMetasploitFTP.pl ListaDeIps.txt
192.168.0.13
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
127.0.0.1
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
192.168.0.13
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
127.0.0.1
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
C:\Perl\bin>
```

3.0 Execute ScanMetasploitFTP.pl informando a lista de sites ou ips:

`perl ScanMetasploitFTP.pl listaDeSitesOuIPs.txt`

4.0 Quantos IPs tinham no arquivo... 4 IPS ??

```
C:\WINDOWS\system32\cmd.exe

C:\Perl\bin>perl ScanMetasploitFTP.pl ListaDeIps.txt
192.168.0.13
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
127.0.0.1
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
192.168.0.13
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
127.0.0.1
-> EasyFTP Server <= 1.7.0.11 CWD Command Stack Buffer Overflow
C:\Perl\bin>
```

ListaDelps - Bloco de notas

```
Arquivo  Editar  Formatar  Exibir  Ajuda
192.168.0.13
127.0.0.1
192.168.0.13
127.0.0.1
```

Observe a saída no terminal, 4 IPS com o servidor EasyFTP 1.7.0.11 vulnerável.

0x06 Código Completo do Scanner para Dectar FTPs vulneráveis aos exploits do Metasploit

```
use IO::Socket;
```

```
use IO::Socket::INET;
```

```
if (!$ARGV[0]) {
```

```
    $sis="$^O";if ($sis eq windows){ $cmd="clear";} else { $cmd="cls"; }
```

```
    system("$cmd");
```

```
    print q {
```

Code desenvolvido por: Inj3cti0n P4ck3t

e-mail to contact: fer_henrick@hotmail.com

Modo de uso: perl ScanMetasploitFTP.pl listaDelps_ou_ListaDeSites.txt

Nome: Fernando Henrique Mengali de Souza

```
};
exit;
}

open( SITE, "< $ARGV[0]" ) or die( "Nao foi possível abrir o arquivo: $" );

our @array = <SITE>;

$numero = $#array;

for ($i = 0; $i <= $numero; $i++) {

$Url = "$array[$i]";

if($Url !~ /http:\\\\/) { $Url = "http://$Url"; }

$Stop = index($Url,":");
$Protocolo = substr($Url,0,$Stop);
$Start = index($Url,"//") + 2;
$Dominio = substr($Url,$Start);
$Stop = index($Dominio,"/");
$Dominio = substr($Dominio,0,$Stop);
$Start = rindex($Url,"/") + 1;
$NomeArq = substr($Url,$Start);
$Compr_Url = length($Url);

$ponto = "$Dominio \n";

our @portas = "21";

foreach $porta (@portas) {

$sock = IO::Socket::INET->new("$ponto:$porta");

if($sock) {

$remote = IO::Socket::INET -> new (Proto => "tcp", PeerAddr => $ponto, PeerPort =>
$porta, Timeout => "7");

$line = <$remote>;

if ($line =~ "Easy File Sharing FTP Server") {
print "$ponto -> Easy File Sharing FTP Server 2.0 PASS Overflow\n";
}

if ($line =~ "NetTerm FTP server") {
```

```
print "$ponto -> NetTerm NetFTPD USER Buffer Overflow\n";
}

if ($line =~ "Sami FTP Server 2.0.2") {
print "$ponto -> KarjaSoft Sami FTP Server v2.02 USER Overflow\n";
}

if ($line =~ "3Com 3CDaemon FTP Server Version 2") {

print "$ponto -> 3Com 3CDaemon 2.0 FTP Username Overflow \n";
}

if ($line =~ "httpdx") {

print "$ponto -> HTTPDX tolog() Function Format String Vulnerability\n";
}

if ($line =~ "BigFoolCat") {

print "$ponto -> EasyFTP Server 1.7.0.11 CWD Command Stack Buffer";
print "Overflow\n";

}

if ($line =~ "vftpd") {

print "$ponto -> Vermillion FTP Daemon PORT Command Memory Corruption\n";
}

if ($line =~ "XLINK FTP Server") {

print "$ponto -> Xlink FTP Client Buffer Overflow \n";

}

if ($line =~ "CesarFTP 0.99g") {

print "$ponto -> Cesar FTP 0.99g MKD Command Buffer Overflow \n";

}

if ($line =~ "9.2.0.1.0") {

print "$ponto -> Oracle 9i XDB FTP PASS Overflow (win32)\n";

}

if ($line =~ "freeFTPd 1.0") {

print "$ponto -> freeFTPd 1.0 Username Overflow\n";

}
```

```

        if ($line =~ "SlimFTPD Server") {

print "$ponto -> SlimFTPD LIST Concatenation Overflow\n";

        }

        if ($line =~ "WS_FTP Server 5.0.5") {

print "$ponto -> Ipswitch WS_FTP Server 5.05 XMD5 Overflow\n";

        }

if ($line =~ "20 ProFTPD 1.2" || $line =~ "Serv-U FTP Server v4.0" || $line =~ "Serv-U FTP
Server v4.1" || $line =~ "Serv-U FTP Server v5.0") {

        print "$ponto -> Serv-U FTPD MDTM Overflow \n";

                }

        }

    }

}

```

Agradecimentos:

C00l3r - _MLK_ - s4r4d0 - DD3str0y3r - Sh0rtKiller - Z4i0n - M0nt3r - Th1nk3r
CODE RED - Forast - r0t3d - Arplhmd - Crackt0r - Chuck_NewBie - Col7er - w4n73d H4ck3r -
Colt7r - dr4k3 - Archit3ct - elemento_pcx - Obseving - D3UX - Believe - Lady Lara - b4rtb0y –
voidpointer - _Bl4ck9_f0x6

Groups: RitualistaS - Fatal Error - [#Elite Top Team] - [Collaps3 CREW] – #C00kies crew