Pentest com Kali Linux

Instrutor:Vitor Mazuco

http://facebook.com/vitormazuco

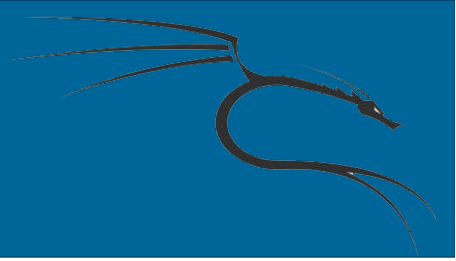
Email:vitor.mazuco@gmail.com

WebSite:http://vmzsolutions.com.br

Isto pode ser visto através do estabelecimento de uma ligação com a porta TCP 80 no sistema Metasploitable 2:

Banner com Python

```
root@KaliLinux:~# python Python 2.7.3 (default, Jan
2 2016, 16:53:07) [GCC 4.7.2] on linux2
Type "help", "copyright", "credits" or "license" for more
information.
>>> import socket
>>> bangrab = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
>>> bangrab.connect(("192.168.1.196", 80))
>>> bangrab.recv(4096)
```



Banner com Python

O serviço executado na porta 80 deste sistema é de aceitar conexões, mas não fornece uma faixa de serviços para que os clientes que se conectam. Se a função *recv* é usado, mas não há dados disponíveis para serem recebidos, a função irá travar. Para automatizar a prática da recolha de banners em Python, uma solução alternativa deve ser usada para identificar se qualquer banner está disponível para ver, antes de chamar esta função. A função *select* fornece uma solução conveniente para este problema:

Banner com Python

```
root@KaliLinux:~# python Python 2.7.3 (default, Jan
2 2013, 16:53:07) [GCC 4.7.2] on linux2 Type "help",
"copyright", "credits" or "license" for more information.
>>> import socket
>>> import select
>>> bangrab = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
>>> bangrab.connect(("192.168.1.196", 80))
>>> ready = select.select([bangrab],[],[],1)
```

Banner com Python

```
>>> if ready[0]:
… print bangrab.recv(4096)
... else:
… print "No Banner"
...
No Banner
```

O objeto *select* é criado e definido para o nome da variável pronto. Este objeto é passado quatro argumentos para incluir uma lista de leitura, uma lista de gravação, uma lista de exceções, e um valor inteiro que define o número de segundos até o tempo limite. Neste caso, só precisamos de identificar quando que o socket esteja pronto para ser lido, assim que o segundo e terceiro argumentos estão vazio.

Uma *array* é retornado com valores que correspondem a cada uma destas três listas. Só estamos interessados em saber se o socket de *bangrab* tem qualquer conteúdo para ler. Para determinar se este é o caso, pode-se testar o primeiro valor na *array*, e se existe um valor, que pode receber o conteúdo do socket. Todo esse processo pode ser automatizado em um script executavél em Python, vide no arquivo *banner_grab.py*

Banner com Python

No script fornecido aqui, três argumentos são aceitos como entrada. O primeiro argumento consiste em um endereço IP para testar a banners de serviços. O segundo argumento indica o primeiro número da porta em um intervalo de números de portas para ser verificado. O terceiro e último argumento indica o último número da porta em um intervalo de números de porta a ser escaneado.

Banner com Python

Quando executado, este *script* irá usar soquetes em Python para se conectar a todos os valores de porta em alcance do sistema remoto indicado, e irá coletar e imprimir todos os banners de serviços identificados. Este script pode ser executado através da modificação das permissões de arquivo e, em seguida, chamando-o diretamente a partir do diretório no qual estava escrito:

```
root@KaliLinux:~# chmod 777 banner_grab.py

root@KaliLinux:~# ./banner_grab.py 192.168.1.196 1 65535
```