SNMP Sweeping a11y.text SNMP Sweeping SNMP Auxiliary Module forÂ Metasploit a11y.text SNMP Auxiliary Module forÂ Metasploit Continuing with our information gathering, letâ€™s take a look at SNMP Sweeping . SNMP sweeps are often good at finding a ton of information about a specific system or actually compromising the remote device. If you can find a Cisco device running a private string for example, you can actually download the entire device configuration, modify it, and upload your own malicious config. OftenÂ the passwords themselves are level 7 encoded, which means they are trivial to decode and obtain the enable or login password for the specific device. Metasploit comes with a built in auxiliary module specifically for sweeping SNMP devices. There are a couple of things to understand before we perform our SNMP scan. First, â€˜ read only â€˜ and â€˜ read write â€˜ community strings play an important role in what type of information can be extracted or modified on the devices themselves. If you can â€œguessâ€• the read-only or read-write strings, you can obtain quite a bit of access you would not normally have. In addition, if Windows-based devices are configured with SNMP, often times with the RO/RW community strings, you can extract patch levels, services running, last reboot times, usernames on the system, routes, and various other amounts of information that are valuable to an attacker. Note : By default Metasploitableâ€™s SNMP service only listens on localhost. Many of the examples demonstrated here will require you to change these default settings. Open and edit /etc/default/snmpd , and change the following from: SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -I -smux -p /var/run/snmpd.pid 127.0.0.1' to SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -I -smux -p /var/run/snmpd.pid 0.0.0.0' A service restart will be needed in order for the changes to take effect. Once restarted, you will now be able to scan the service from your attacking machine. What is a MIB? a11y.text What is a MIB? When querying through SNMP, there is what is called an MIB API . The MIB stands for the Management Information Base . This interface allows you to query the device and extract information. Metasploit comes loaded with a list of default MIBs that it has in its database, it uses them to query the device for more information depending on what level of access is obtained. Letâ€™s take a peek at the auxiliary module. msf > search snmp

```
Matching Modules

================


   Name                                          Disclosure Date  Rank    Description

   ----                                     ---------------  ----    -----------

   auxiliary/scanner/misc/oki_scanner                                normal  OKI Printer Default Login

Credential Scanner

   auxiliary/scanner/snmp/aix_version                                normal  AIX SNMP Scanner Auxiliary

Module

   auxiliary/scanner/snmp/cisco_config_tftp                          normal  Cisco IOS SNMP Configuration

Grabber (TFTP)

   auxiliary/scanner/snmp/cisco_upload_file                          normal  Cisco IOS SNMP File Upload

(TFTP)

   auxiliary/scanner/snmp/snmp_enum                                  normal  SNMP Enumeration Module

   auxiliary/scanner/snmp/snmp_enumshares                               normal  SNMP Windows SMB Share

Enumeration

   auxiliary/scanner/snmp/snmp_enumusers                               normal  SNMP Windows Username

Enumeration

   auxiliary/scanner/snmp/snmp_login                                normal  SNMP Community Scanner

   auxiliary/scanner/snmp/snmp_set                                  normal  SNMP Set Module

   auxiliary/scanner/snmp/xerox_workcentre_enumusers                  normal  Xerox WorkCentre User

Enumeration (SNMP)

   exploit/windows/ftp/oracle9i_xdb_ftp_unlock        2003-08-18       great   Oracle 9i XDB FTP

UNLOCK Overflow (win32)

   exploit/windows/http/hp_nnm_ovwebsnmpsrv_main      2010-06-16       great   HP OpenView
```

```
Network Node Manager ovwebsnmpsrv.exe main Buffer Overflow
   exploit/windows/http/hp_nnm_ovwebsnmpsrv_ovutil    2010-06-16       great   HP OpenView
Network Node Manager ovwebsnmpsrv.exe ovutil Buffer Overflow
   exploit/windows/http/hp_nnm_ovwebsnmpsrv_uro       2010-06-08       great   HP OpenView
Network Node Manager ovwebsnmpsrv.exe Unrecognized Option Buffer Overflow
   exploit/windows/http/hp_nnm_snmp                   2009-12-09       great   HP OpenView Network
Node Manager Snmp.exe CGI Buffer Overflow
   exploit/windows/http/hp_nnm_snmpviewer_actapp      2010-05-11       great   HP OpenView
Network Node Manager snmpviewer.exe Buffer Overflow
   post/windows/gather/enum_snmp                                       normal  Windows Gather SNMP
Settings Enumeration (Registry)


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


Module options (auxiliary/scanner/snmp/snmp_login):


   Name              Current Setting                    Required  Description
   ----              ---------------                    --------  -----------
   BLANK_PASSWORDS   false                              no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                                  yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false                              no        Try each user/password couple stored in the
current database
   DB_ALL_PASS       false                              no        Add all passwords in the current database to
the list
   DB_ALL_USERS      false                              no        Add all users in the current database to the
```

```
list
   PASSWORD                                          no        The password to test
   PASS_FILE         /usr/share/wordlists/fasttrack.txt  no        File containing communities, one per
line
   RHOSTS                                        yes       The target address range or CIDR identifier
   RPORT             161                          yes       The target port
   STOP_ON_SUCCESS   false                         yes       Stop guessing when a credential works
for a host
   THREADS           1                            yes       The number of concurrent threads
   USER_AS_PASS      false                         no        Try the username as the password for all
users
   VERBOSE           true                          yes       Whether to print output for all attempts
   VERSION           1                            yes       The SNMP version to scan (Accepted: 1, 2c, all)

msf auxiliary(snmp_login) >  set RHOSTS 192.168.0.0-192.168.5.255
rhosts => 192.168.0.0-192.168.5.255
msf auxiliary(snmp_login) >  set THREADS 10
threads => 10
msf auxiliary(snmp_login) >  run
[*] >> progress (192.168.0.0-192.168.0.255) 0/30208...
[*] >> progress (192.168.1.0-192.168.1.255) 0/30208...
[*] >> progress (192.168.2.0-192.168.2.255) 0/30208...
[*] >> progress (192.168.3.0-192.168.3.255) 0/30208...
[*] >> progress (192.168.4.0-192.168.4.255) 0/30208...
[*] >> progress (-) 0/0...
[*] 192.168.1.50 'public' 'APC Web/SNMP Management Card (MB:v3.8.6 PF:v3.5.5
```

PN:apc_hw02_aos_355.bin AF1:v3.5.5 AN1:apc_hw02_sumx_355.bin MN:AP9619 HR:A10 SN: NA0827001465 MD:07/01/2008) (Embedded PowerNet SNMP Agent SW v2.2 compatible)'

[*] Auxiliary module execution completed As we can see here, we were able to find a community string of â€œpublicâ€•. This is most likely read-only and doesnâ€™t reveal a ton of information. We do learn that the device is an APC Web/SNMP device, and what versions itâ€™s running. SNMP Enum a11y.text SNMP Enum We can gather lots of information when using SNMP scanning modules such as open ports, services, hostname, processes, and uptime to name a few. Using our Metasploitable virtual machine as our target, weâ€™ll run the auxiliary/scanner/snmp/snmp_enum module and see what information it will provide us. First we load the module and set the â€˜RHOSTâ€™ option using the information stored in our workspace. Using hosts -R will set this options for us. msf auxiliary(snmp_enum) > run

```
[+] 172.16.194.172, Connected.


[*] System information:


Host IP                       : 172.16.194.172
Hostname                      : metasploitable
Description                   : Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
Contact                       : msfdev@metasploit.com
Location                      : Metasploit Lab
Uptime snmp                   : 02:35:38.71
Uptime system                 : 00:20:13.21
System date                   : 2012-7-9 18:11:11.0
```

```
[*] Network information:

IP forwarding enabled         : no
Default TTL                   : 64
TCP segments received         : 19
TCP segments sent             : 21
TCP segments retrans          : 0
Input datagrams               : 5055
Delivered datagrams           : 5050
Output datagrams              : 4527

...snip...

[*] Device information:

Id             Type            Status          Descr
768            Processor       unknown         GenuineIntel: Intel(R) Core(TM) i7-2860QM CPU
@ 2.50GHz
1025           Network         unknown         network interface lo
1026           Network         unknown         network interface eth0
1552           Disk Storage    unknown         SCSI disk (/dev/sda)
3072           Coprocessor     unknown         Guessing that there's a floating point
co-processor

[*] Processes:
```

```
Id          Status        Name          Path          Parameters

1           runnable       init          /sbin/init

2           runnable       kthreadd       kthreadd

3           runnable       migration/0    migration/0

4           runnable       ksoftirqd/0    ksoftirqd/0

5           runnable       watchdog/0      watchdog/0

6           runnable       events/0       events/0

7           runnable       khelper       khelper

41           runnable       kblockd/0      kblockd/0

68           runnable       kseriod       kseriod


...snip...


5696          runnable       su           su

5697          runnable       bash          bash

5747          running       snmpd          snmpd



[*] Scanned 1 of 1 hosts (100% complete)

[*] Auxiliary module execution completed
```

Reviewing our SNMP Scan a11y.text Reviewing our SNMP Scan The output provided above by our SNMP scanÂ provides us with a wealth of information on our target system. Although cropped for length, we can still see lots of relevant information about our target such as its processor type, process IDs, etc. Next Writing Your Own Scanner Prev Extending Psnuffle