Orange
Cyberdefense

WMI Attacks

From Theory To Practice

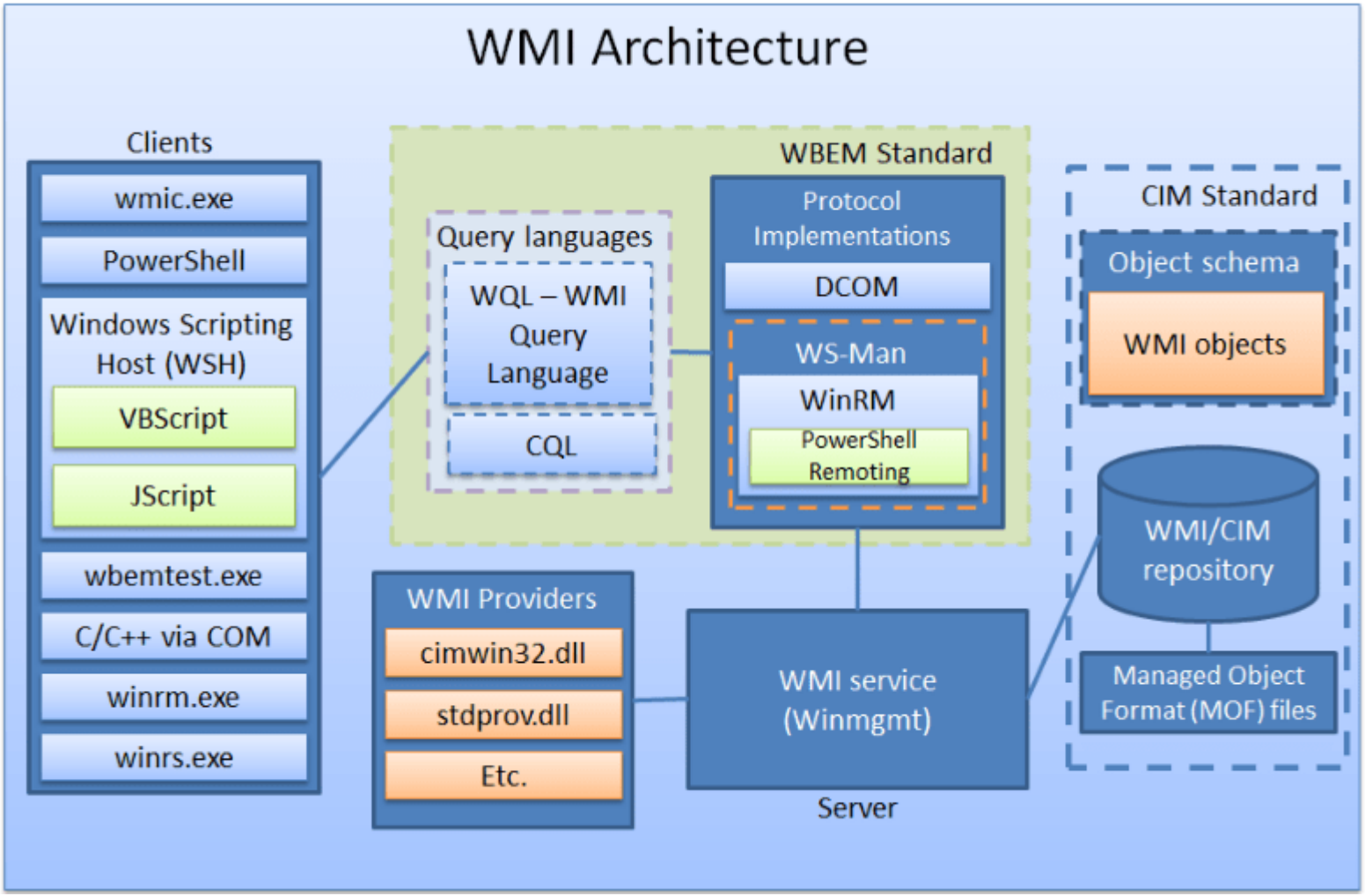
Andrei Dumitrescu
@_dracu_

orange

WMI

- Windows Management Instrumentation (since 1996!)
- Architecture: database (repository) of objects and data providers:
  - Namespace -> Class -> Object -> Property -> Value
- WQL read-only access (no INSERT or UPDATE statements)
- Execute / kill processes, server shutdown, modify registry etc.
- **No output** for process execution
- Remote access: DCOM protocol (TCP/135), *WinRM (TCP/5985)*
- Tools : wmic.exe, Get-WmiObject, Invoke-WmiMethod , *Get-CimInstance*, *Invoke-CimMethod* etc.

WMI



https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent%20Asynchronous-And-Fileless-Backdoor-wp.pdf

orange

WMI

- Example in PowerShell
  - *Get-WmiObject -Query "Select * Win32_Process"*
- Example command line:
  - *wmic /node:10.0.0.1 /user:MYUSR /password:MYPWD process list*

WMI Persistence techniques

MOF Files

- For example:

```
#pragma namespace("\\\\.\\root\\cimv2")
class MyClass@CLASS@
{
        [key] string Name;
};
class ActiveScriptEventConsumer : __EventConsumer
{
        [key] string Name;
        [not_null] string ScriptingEngine;
        string ScriptFileName;
        [template] string ScriptText;
  uint32 KillTimeout;
};
instance of __Win32Provider as $P
{
    Name  = "ActiveScriptEventConsumer";
    CLSID = "{266c72e7-62e8-11d1-ad89-00c04fd8fdff}";
    PerUserInitialization = TRUE;
};
```

https://github.com/rapid7/metasploit-framework/blob/master/lib/msf/core/exploit/wbemexec.rb

MOF Files

- Stuxnet (2010):

1. Exploit MS10-061 and upload an .EXE and .MOF:
   - .EXE file in *C:\Windows\system32*
   - .MOF file in *C:\Windows\system32\wbem\MOF*

2. ??? Magic!

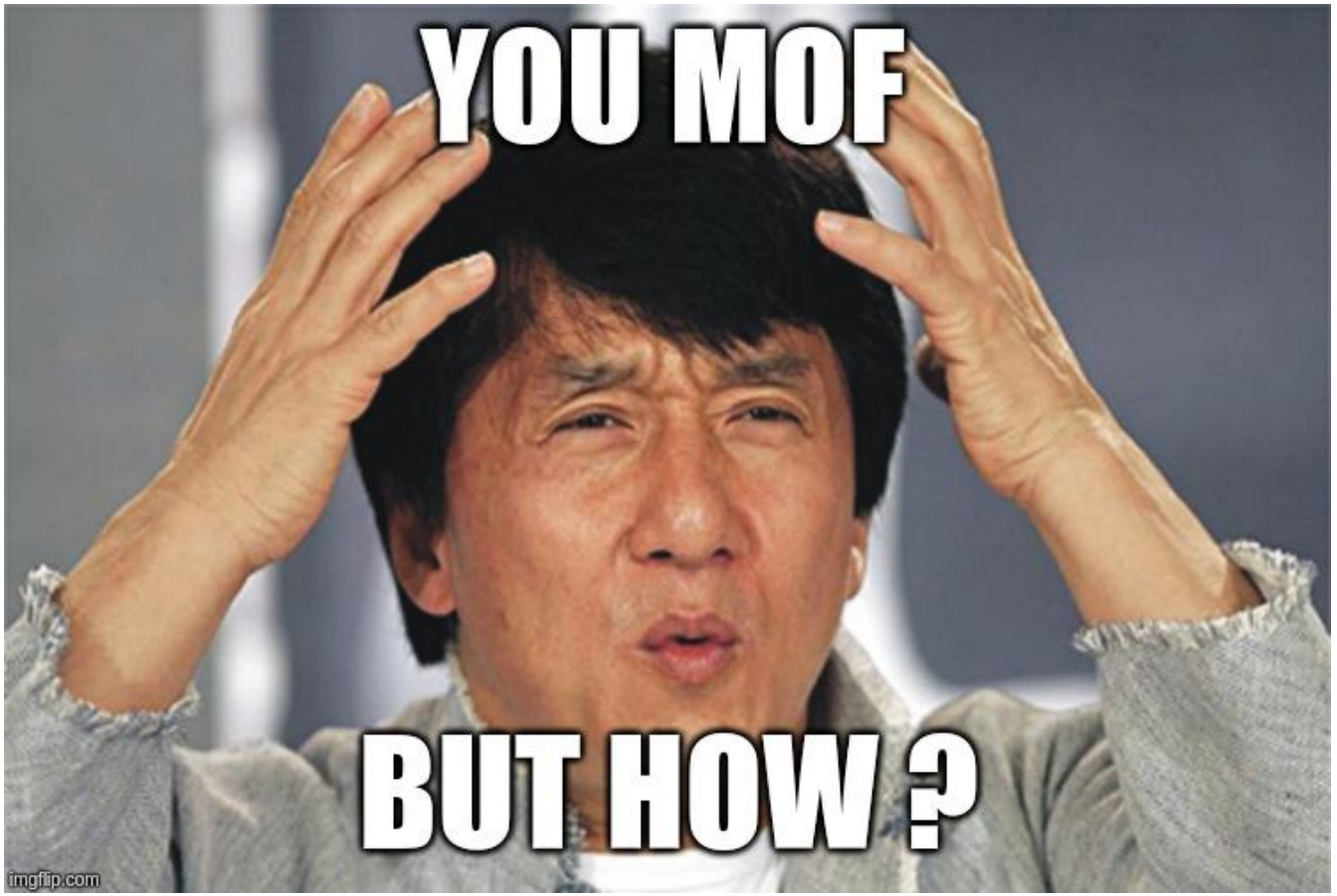
- Today (Windows Vista+) we have #PRAGMA AUTORECOVER

- The list of MOF files for autorecovery is stored here :
  - HKLM\SOFTWARE\Microsoft\WBEM\CIMOM\Autorecover MOFs
  - C:\Windows\system32\wbem\AutoRecover

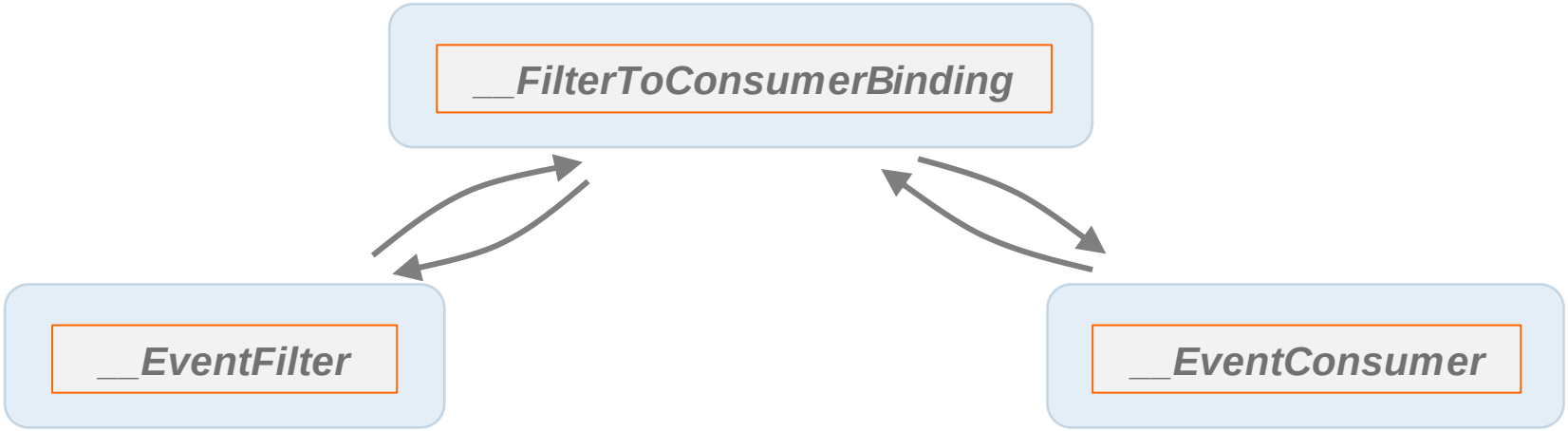
MOF Files

- Moar MOF:
  - https://poppopret.blogspot.com/2011/09/playing-with-mof-files-on-windows-for.html
  - https://khrox40sh.wordpress.com/2014/06/10/moftastic_powershell/
  - https://www.secureworks.com/blog/wmi-persistence
  - https://in.security/an-intro-into-abusing-and-identifying-wmi-event-subscriptions-for-persistence/

MOF Files



WMI Event Subscriptions



Intrinsic events:

- __ClassModificationEvent
- __ClassCreationEvent
- __InstanceCreationEvent
- __MethodInvocationEvent
- __InstanceModificationEvent
- …

Extrinsic events:

- ROOT\DEFAULT:RegistryKeyChangeEvent
- ROOT\DEFAULT:RegistryValueChangeEvent
- ROOT\CIMV2:Win32_ModuleLoadTrace
- ROOT\CIMV2:Win32_VolumeChangeEvent
- …

Event Consumers:

- LogFileEventConsumer
- **ActiveScriptEventConsumer**
- NTEventLogEventConsumer
- SMTPEventConsumer
- **CommandLineEventConsumer**

WMI Event Subscriptions

- Turla / Snake malware WMI persistence example:
  - Event Filter

```
$IT825cd = "SELECT * FROM __instanceModificationEvent WHERE TargetInstance ISA 'Win32_LocalTi
me' AND TargetInstance.Hour=15 AND TargetInstance.Minute=30 AND TargetInstance.Second=40";
$VQI79dcf = Set-WmiInstance -Class __EventFilter -Namespace root\subscription -Arguments @{na
me='Log Adapter Filter';EventNameSpace='root\CimV2';QueryLanguage='WQL';Query=$IT825cd};
```

  - Event Consumer

```
$NLP35gh = Set-WmiInstance -Namespace "root\subscription" -Class 'CommandLineEventConsumer' -
Arguments @{name='Syslog Consumer';CommandLineTemplate="$($Env:SystemRoot)\System32\WindowsPo
werShell\v1.0\powershell.exe -enc $HL39fjh";RunInteractively='false'};
```

  - Filter To Consumer Binding

```
Set-WmiInstance -Namespace root\subscription -Class __FilterToConsumerBinding -Arguments @{Fi
lter=$VQI79dcf;Consumer=$NLP35gh};
```

https://www.welivesecurity.com/2019/05/29/turla-powershell-usage/

WMI Event Subscriptions

- Malware:
  - https://blog.trendmicro.com/trendlabs-security-intelligence/cryptocurrency-miner-uses-wmi-eternalblue-spread-filelessly/
  - https://www.welivesecurity.com/2019/05/29/turla-powershell-usage/
  - https://secrary.com/ReversingMalware/WMIGhost/
  - https://twitter.com/HuntressLabs/status/1134827127751270401

- List of malware using WMI Event Subscriptions:
  - https://attack.mitre.org/techniques/T1084/

- Pentest tools :
  - https://github.com/Sw4mpfox/PowerLurk/
  - https://github.com/GhostPack/SharpWMI/
  - https://www.rapid7.com/db/modules/exploit/windows/local/wmi_persistence
  - https://github.com/EmpireProject/Empire/blob/master/data/module_source/persistence/Persistence.psm1

WMI Providers

- WMI Providers are COM objects, the backend to WMI
- A WMI Provider is made of .DLL and a .MOF (optional). Example:
  - *C:\Windows\System32\Wbem\CIMWin32.dll*
  - *C:\Windows\System32\Wbem\CIMWin32.mof*
- Registering a WMI Provider:
  - *InstallUtil.exe provider.dll*

  OR
  - From WMI itself
- Advantages:
  - Remote loading
  - Run as SYSTEM

WMI Providers

- PoCs (that work!):
  - https://github.com/re4lity/subTee-gits-backups/blob/master/EvilWMIProvider.cs

```
Invoke-WmiMethod -Namespace root\cimv2- -Class Win32_Evil -Name ExecShellCode
 -ArgumentList @(0x90,0x90,0x90),  $null
```

  - https://github.com/jaredcatkinson/EvilNetConnectionWMIProvider

- Derbycon'17 : Building Better Backdoors with WMI, Alexander Leary
  - https://www.slideshare.net/AlexanderLeary/building-better-backdoors-with-wmi-derbycon-2017

  - https://github.com/oxbadjuju/PowerProvider

Invoke-WMIDuplicateClass, Install-WMIProviderExtention…

  - https://github.com/oxbadjuju/WheresMyImplant

WMI

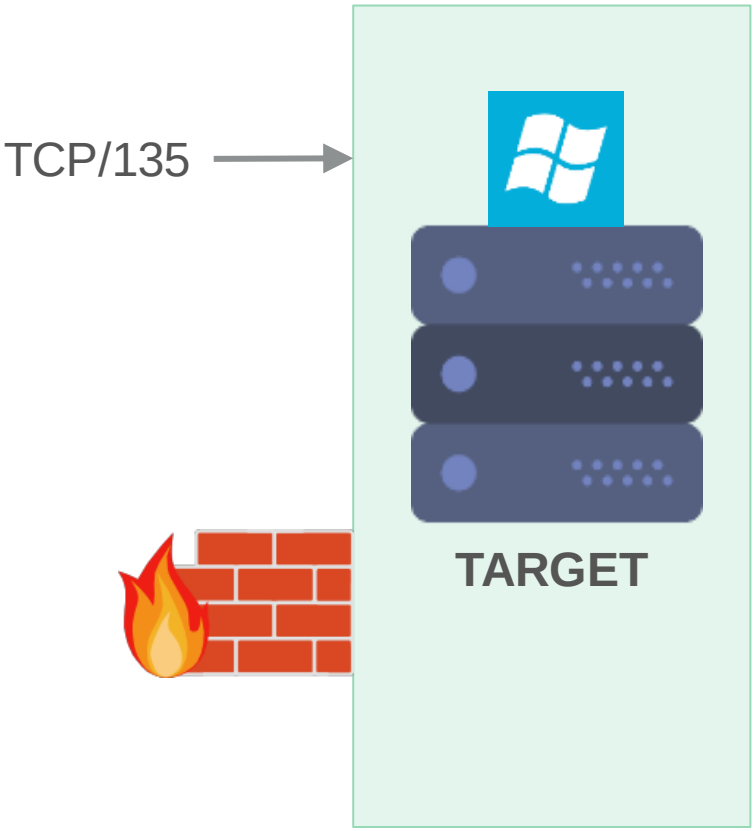
WMI as C2 Channel: a brief history

- 2014: WMI Shell technique (first!)
  - http://2014.hackitoergosum.org/slides/day1_WMI_Shell_Andrei_Dumitrescu.pdf
- 2014: APT29 attacks (non-public)
    - Custom class creation and storage in class properties
- 2015: Matt Graeber research on WMI for Blackhat 2015
  - https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent%20Asynchronous-And-Fileless-Backdoor-wp.pdf
  - Existing and new persistence mechanisms:
    - Registry modification and storage via StdRegProv
- 2016: WMImplant technique to bypass Device Guard

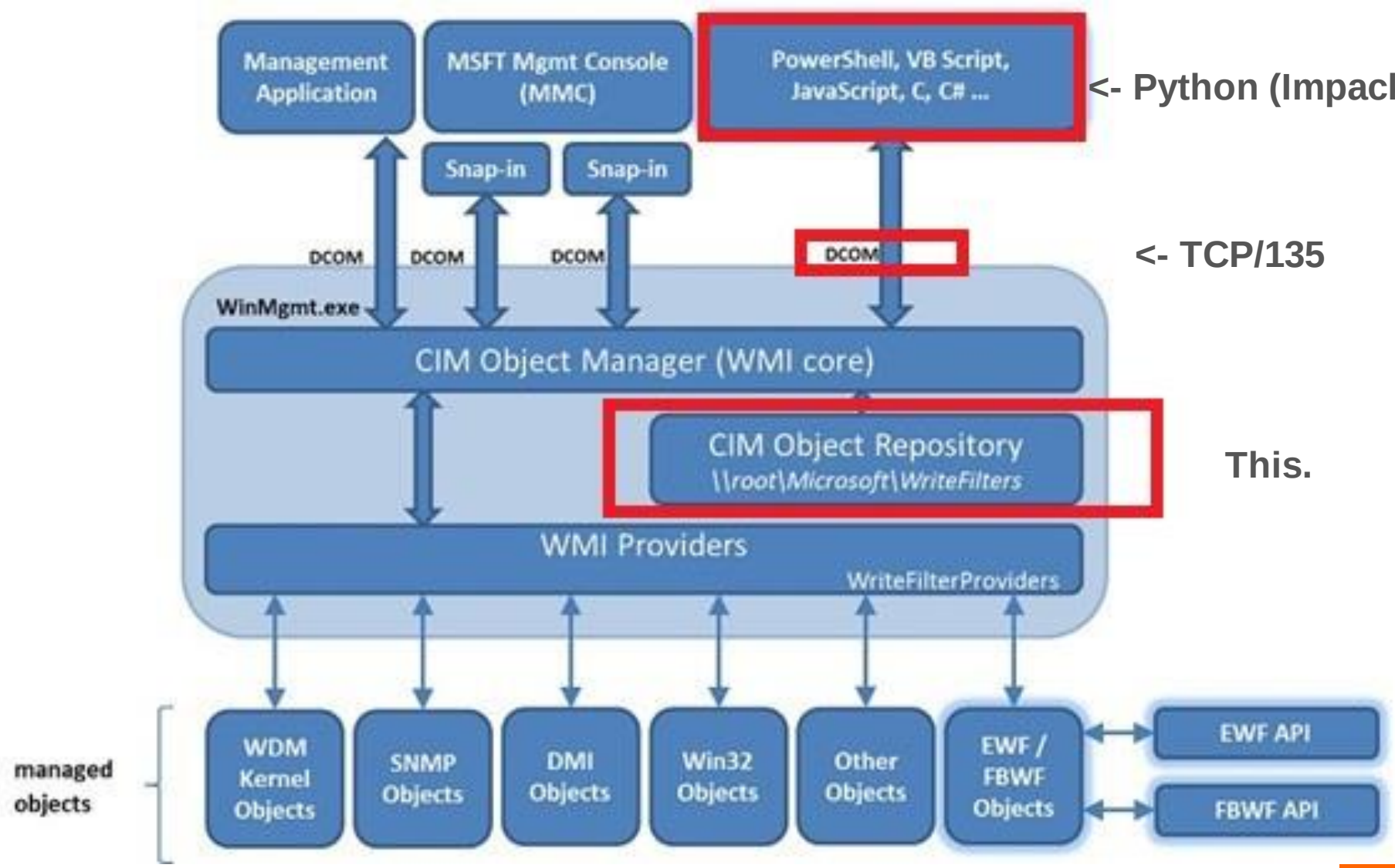
Attack scenario

HACKER*

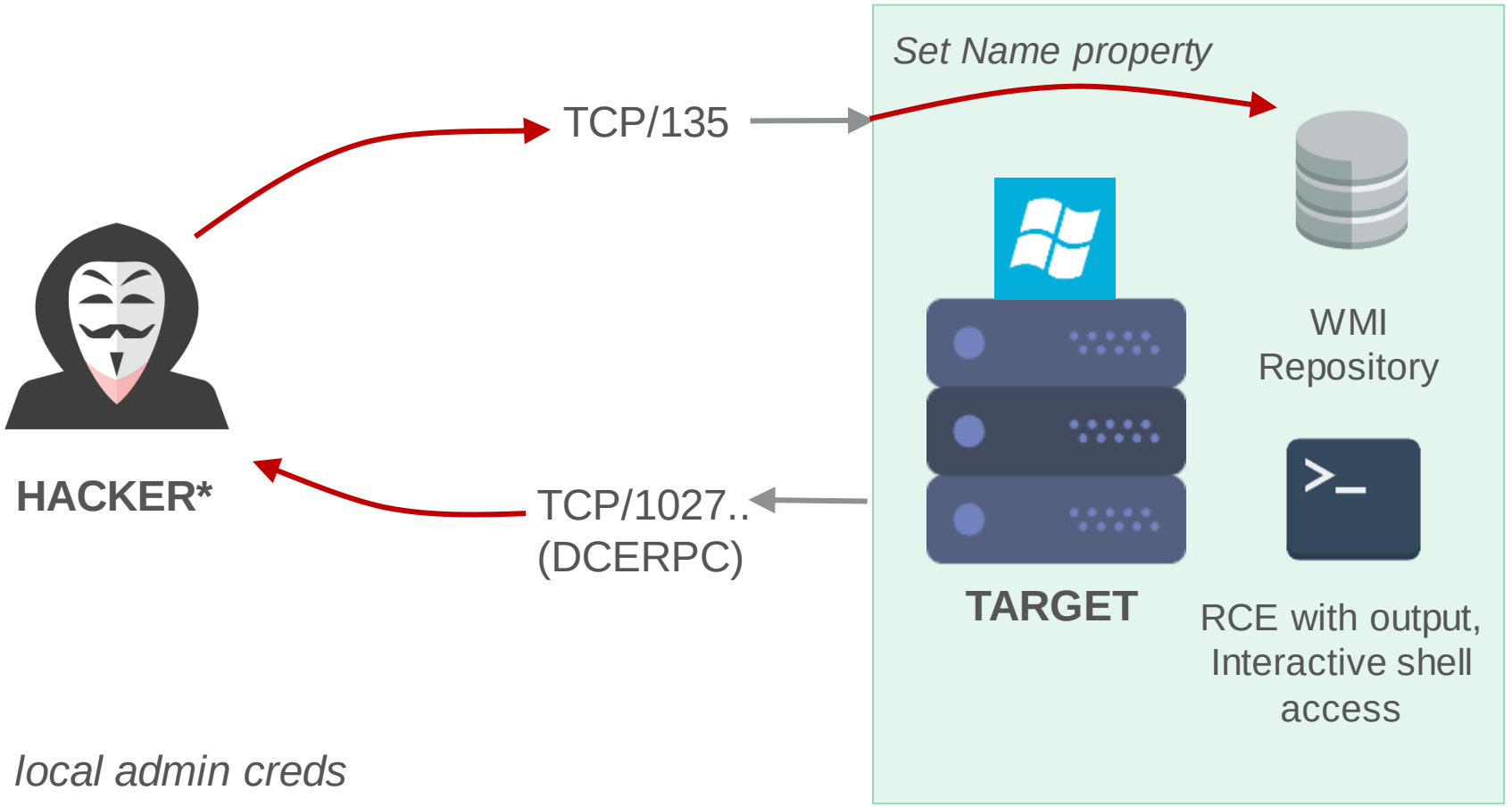
**has local admin creds*



WMI


Management Application
MSFT Mgmt Console (MMC)
PowerShell, VB Script, JavaScript, C, C# ...
<- Python (Impacket)
Snap-in
Snap-in
DCOM
DCOM
DCOM
DCOM
<- TCP/135
WinMgmt.exe
CIM Object Manager (WMI core)
CIM Object Repository
\\root\Microsoft\WriteFilters
This.
WMI Providers
WriteFilterProviders
managed objects
WDM Kernel Objects
SNMP Objects
DMI Objects
Win32 Objects
Other Objects
EWF / FBWF Objects
EWF API
FBWF API

WMI Shell

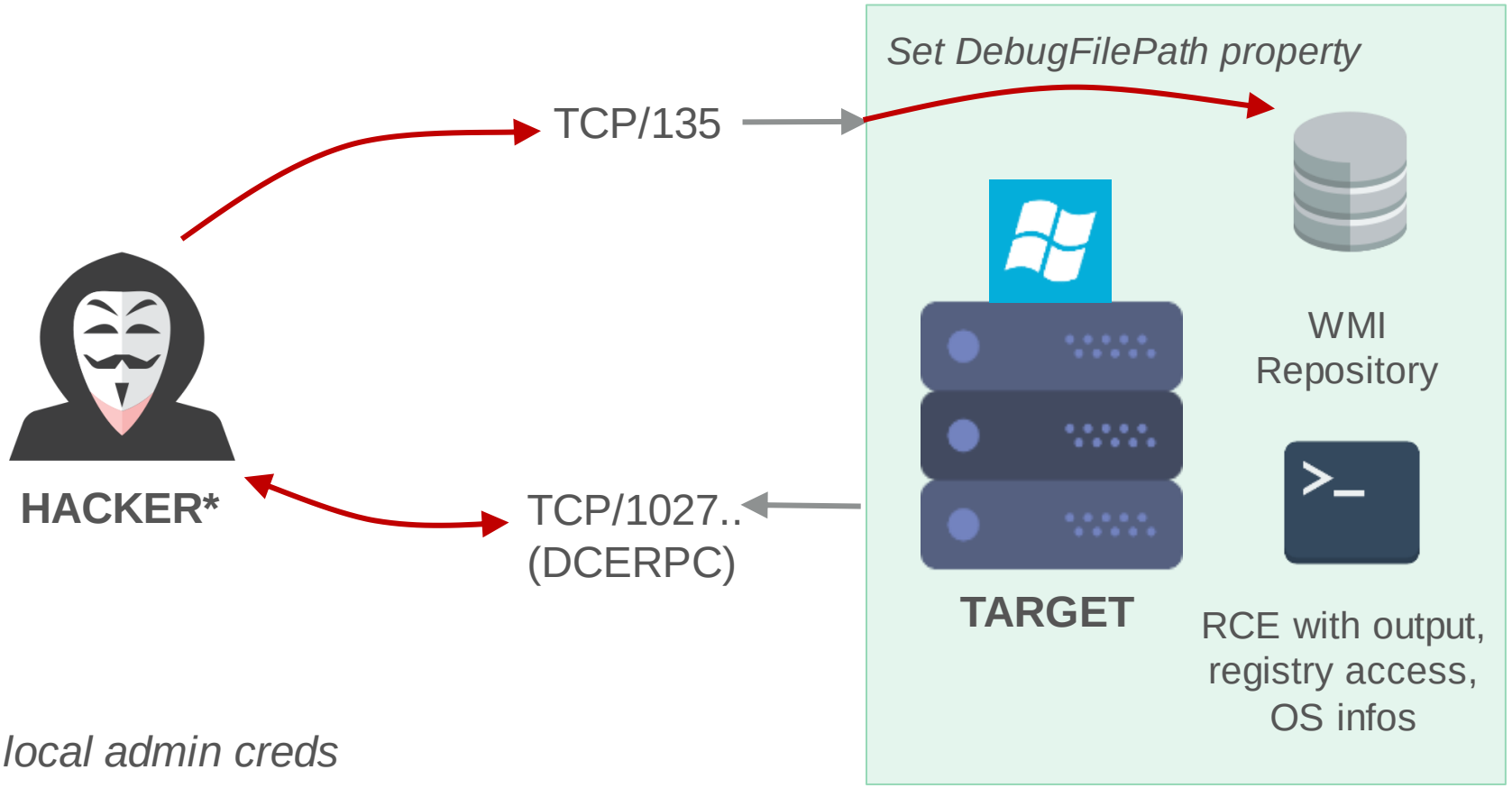


*has local admin creds

WMI Shell

- Implemented in Python and VBScript
- Interactive shell prompt
- Command execution, file upload
- Stores output in the ***Name*** properties of the *__Namespace* class in the *root/default* namespace
- *Name* has a **finite length** of 8000 characters

WMImplant



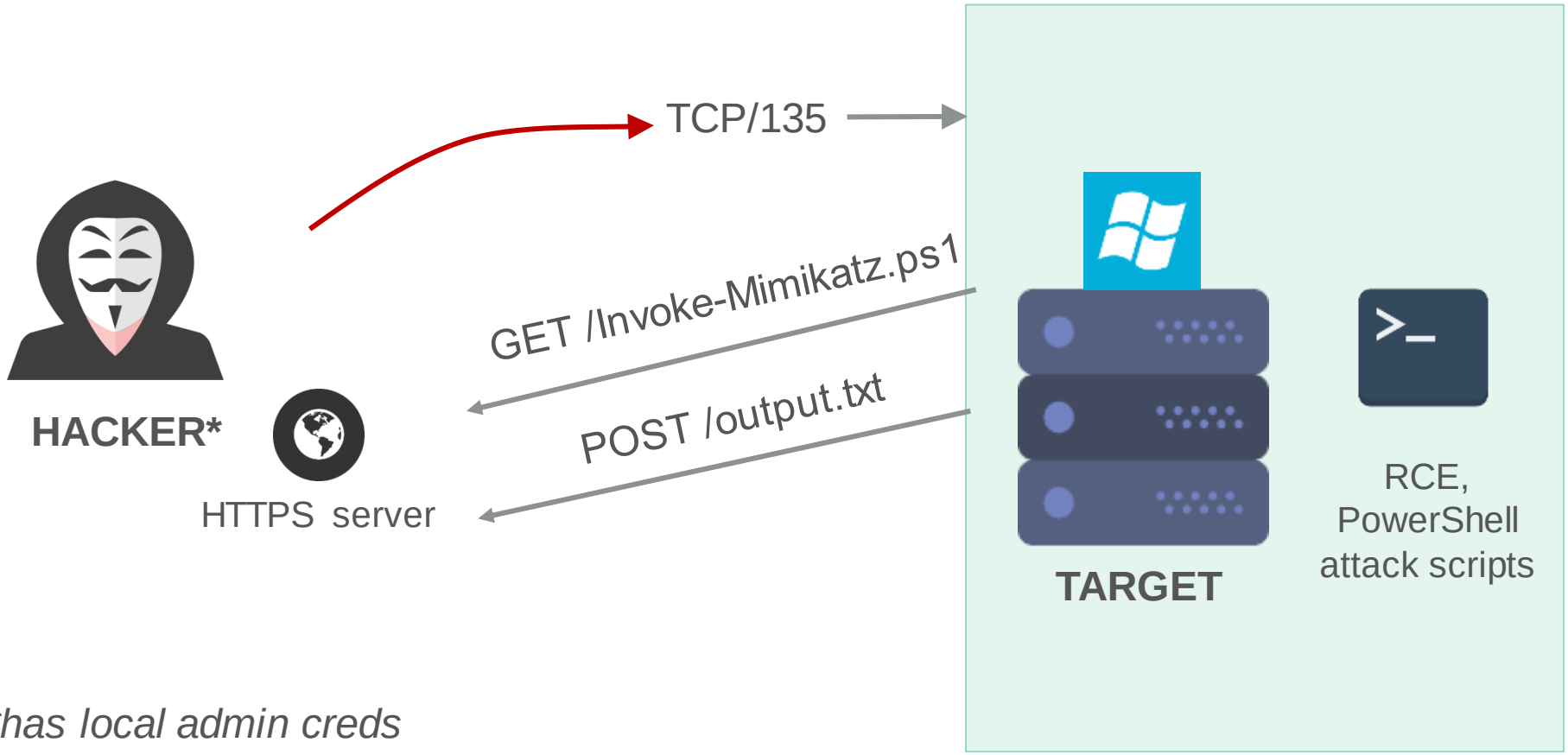
*has local admin creds

WMImplant

- Implemented entirely in Powershell
- Same network flow as WMI Shell
- More functions, non-interactive, bypasses Windows 10 protections
- Stores data in the ***DebugFilePath*** property of the *Win32_OSRecoveryConfiguration* class
- DebugFilePath has **"infinite" length** (at least **256 MB**)

orange

CrackMapExec



*has local admin creds

CrackMapExec

- Several “protocols” : SMB, MSSQL etc.
- Many modules, different execution styles
- HTTP server-based modules :
  - bloodhound.py
  - empire_exec.py
  - enum_chrome.py
  - get_keystrokes.py
  - get_netdomaincontroller.py
  - get_netrdpsession.py
  - invoke_sessiongopher.py
  - invoke_vnc.py
  - met_inject.py
  - **mimikatz.py**
  - mimikatz_enum_vault_creds.py
  - multirdp.py
  - netripper.py
  - …

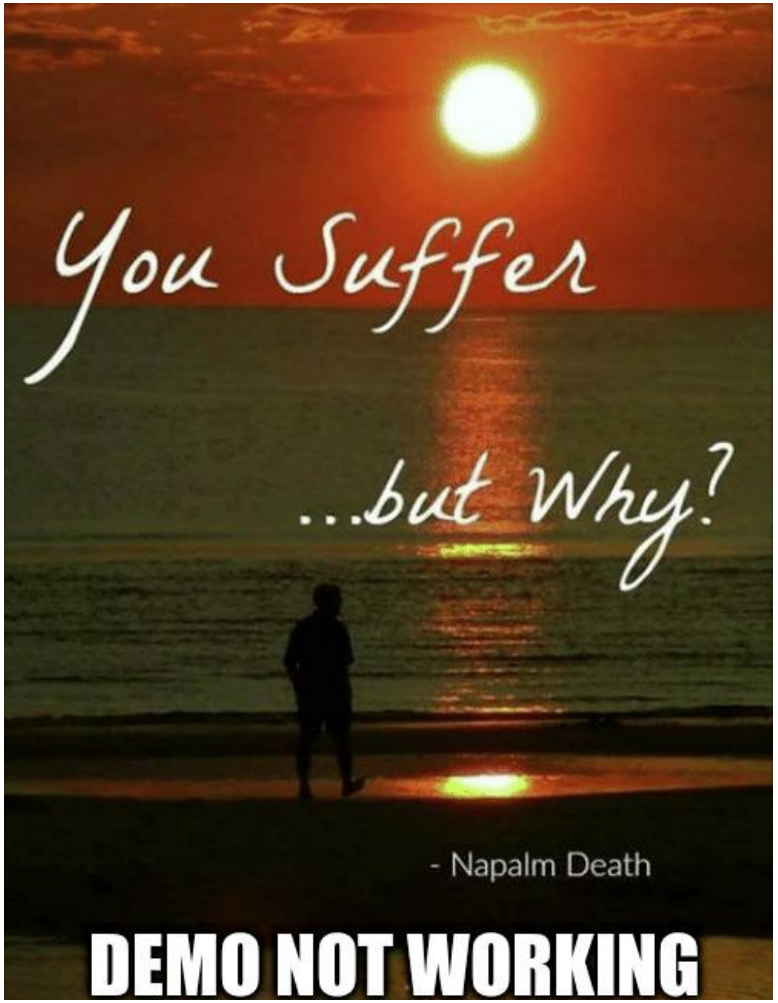
CrackMapExec – WMI protocol (new!)

- Tunneling every module through WMI.
- No HTTP server, network share access needed.
- *./cme/protocols/wmi.py*
  - **--query** (e.g. :
    *Select * From AntiVirusProduct*
    Select DebugFilePath From Win32_OSRecoveryConfiguration
    etc.)
  - **--update** (like
    wmic /node:10.0.0.42 recoveros set debugfilepath=xxx
    but with Impacket!)
  - **--execute** (like
    wmic /node:10.0.0.42 process call create "cmd.exe"
    but with Impacket!)

CrackMapExec – WMI protocol (new!)

DEMO TIME

CrackMapExec – WMI protocol (new!)



CrackMapExec – WMI protocol (new!)

WMI Attack tools

- CrackMapExec
  - https://github.com/byt3bl33d3r/CrackMapExec
  - https://github.com/byt3bl33d3r/CrackMapExec/tree/4.1.odev/cme/c2
- WMImplant
  - https://github.com/FortyNorthSecurity/WMImplant
- WMI Shell (archived):
  - https://github.com/Orange-Cyberdefense/wmi-shell
- CrackMapExec with **WMI protocol**:
  - **https://github.com/Orange-Cyberdefense/cme-wmi**

…and some WMI Logging

- MISC n° 91 mai 2017, Guichard Jean-Philip Wyttenbach Bruno:
  - https://connect.ed-diamond.com/MISC/MISC-091/Detecter-la-persistance-WMI
- Derbycon'18 : Detecting WMI exploitation, Michael Gough:
  - https://www.slideshare.net/Hackerhurricane/detecting-wmi-exploitation-v11
  - https://www.youtube.com/watch?v=w-UFEKR2lO8
- Some tips:
  - https://www.eideon.com/2018-03-02-THL03-WMIBackdoors/
  - https://github.com/marcurdy/dfir-toolset/blob/master/wmi-notes

WMI Attack tools