Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería de Computadores

Trabajo Fin de Grado

Análisis de procedimientos de escalada de privilegios basado en el framework MITRE ATT&CK

Autor: Kevin van Liebergen Ávila

Tutor: Jose Javier Martínez Herráiz

Director: Javier Junquera Sánchez

2020

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Computadores

Trabajo Fin de Grado

Análisis de procedimientos de escalada de privilegios basado en el framework MITRE ATT&CK

Autor: Kevin van Liebergen Ávila

Tutor: Jose Javier Martínez Herráiz

Director: Javier Junquera Sánchez

Tribunal:

Presidente: .....

Vocal 1º: .....

Vocal 2º: .....

Calificación: .....

Fecha: .....

Dedicado a todo aquel que me ha insistido en no abandonar este viaje,
al fin y al cabo la vida es como montar en bicicleta,
si quieres mantener el equilibrio, tienes que seguir avanzando...

"In real open source, you have the right to control your own destiny"
Linus Torvalds (Creator of the Linux kernel)

Agradecimientos

Live fast, die old,
and make very sure everyone
knows you were there

Alan Cox

Al finalizar este trabajo no me paran de aparecer imágenes de gente que me ha acompañado en estos años, en primer lugar quería agradecer a mi familia la motivación que me han brindado y la paciencia que han tenido conmigo, a Jose Javier Martínez Herráiz por ofrecerme la oportunidad de entrar laboralmente en el mundo de la ciberseguridad con la cátedra de investigación, a los UAHrros, ukis, amigos de mi barrio, a mis compañeros de la cátedra y por toda las personas que empezaron y lamentablemente no pudieron continuar este viaje conmigo. ¡Muchas gracias!

Como dijo Hobbes "la información es poder", y gracias a que la información es libre y accesible la tecnología y el mundo pueden avanzar. Este granito de arena va destinado, entre otras personas a la gente que cree y defiende el movimiento Open Source, porque gracias a ellos el mundo no contaría con los dispositivos móviles actuales ni con los servidores en los que se basan la mayoría de servicios de Internet, porque como dijo mi tutor Javier Junquera; "aunque cueste reconocerlo, no seríamos las mismas personas si nunca hubiese existido Wikipedia [1]".

El desarrollo de esta herramienta se encuentra alojada actualmente en mi repositorio personal de github https://github.com/KevinLiebergen.

[1] https://es.wikipedia.org/

Resumen

Este documento pretende mostrar y analizar las distintas técnicas existentes que han sido utilizadas en ataques reales para escalar privilegios en una máquina. Basándonos en el framework MITTRE ATT&CK se ha realizado un estudio para auditar los sistemas operativos Windows y Linux frente a un equipo Blue Team.

Se ha propuesto desarrollar una herramienta que descubra y notifique los puntos débiles de un sistema para escalar privilegios, estudiaremos cuál es el funcionamiento de los sistemas Windows y Linux y con ello abordaremos las principales debilidades que tienen estos sistemas, para posteriormente notificarlas en caso de que no se encuentre debidamente configuradas.

Palabras clave: Escalada de privilegios, Pentesting, Seguridad en Windows, Seguridad en Linux, MITTRE ATT&CK.

Abstract

This document aims to show and analyze the various existing techniques that have been used in real attacks to escalate privileges on a machine. Based on the MITTRE ATT&CK framework, a study has been carried out to audit the Windows and Linux operating systems against a Blue Team.

We have proposed to develop a tool that discovers and notifies the weak points of a system for escalating privileges. We will study how the Windows and Linux systems work and with this we will cover the main weaknesses that these systems have, to later notify them in case they are not properly configured.

Keywords: Privilege Escalation, Pentesting, Security in Windows, Security in Linux, MITTRE ATT&CK.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1 Presentación

Un *pentesting* o test de penetración consiste en atacar y comprometer un sistema informático de forma que se recojan las vulnerabilidades que tiene dicho sistema para poder prevenirse frente a ataques externos.

A nivel general existen varias fases dentro de un proceso de *pentesting* que es necesario seguir de manera ordenada. Estas fases, omitiendo el pre-acuerdo para establecer el tipo de caja de dicho *pentesting* (blanca, gris o negra) o la elaboración del informe son: Fase de reconocimiento, análisis de vulnerabilidades, enumeración, explotación y post-explotación.

Un *pentesting* se concluye cuando se obtiene control absoluto sobre el sistema a auditar, es decir, cuando se obtienen los máximos privilegios de la máquina y asimismo se consigue persistencia para acceder cuando se desee a ella. En otras palabras, conseguir los máximos privilegios de la máquina, privilegios de administrador, es uno de los últimos y más importantes pasos dentro de la fase de post-explotación, por lo que es necesario que el sistema se encuentre a prueba de escalada de privilegios.

Este trabajo pretende crear un procedimiento para la búsqueda de vulnerabilidades y fallos de configuración para escalar privilegios en sistemas Windows y Linux basados en el framework MITRE ATT&CK. Para ello, estudiaremos y analizaremos las distintas técnicas que se recogen acerca de la escalada de privilegios. Se analizarán los distintos tipos de usuarios y cómo interactúan con el sistema operativo según si pertenecen a Windows o Linux. Asimismo se explicará el funcionamiento interno de los sistemas para conocer una visión global y posteriormente aunar dicho conocimiento con las técnicas de intrusión y poder elaborar un procedimiento eficaz.

Después se elaborarán dos herramientas, en función del sistemas operativo, para poder desarrollar el procedimiento creado para que pueda servir sobre equipos Blue Team, conocer que puntos críticos ofrece el sistema y cómo mitigarlos. La finalidad es que la herramienta sea modular, para poder añadir nuevas técnicas según se vaya actualizando el framework MITRE ATT&CK, y poder añadir nuevas tácticas.

1.2 ¿Qué es MITTRE ATT&CK?

El trabajo se ha basado en MITTRE ATT&CK, una base global de conocimiento donde se recogen diversas técnicas y tácticas empleadas en ataques reales de todo el mundo, se utiliza para establecer una base para el desarrollo de metodologías tanto para el sector privado, gobierno, productos de ciberseguridad y dentro

de la comunidad cíber. Se va a proceder a estudiar las técnicas en Sistemas Linux y Windows de la táctica escalada de privilegios para desarrollar la herramienta que analice dichas vulnerabilidades.

Esto nos ha llevado a tener que recopilar investigaciones y herramientas que existen actualmente como `pentestmonkey` [1] o `PEASS` [2].

1.3 Justificación

Se ha decidido realizar esta propuesta innovadora para encontrar una herramienta que facilite encontrar las brechas de seguridad de un sistema, de tal manera que en futuros *Blue Teams* se acelere el proceso de defensa frente a la escalada de privilegios.

También se ha elegido el tema la escalada de privilegios al poseer conocimiento sobre Sistemas Linux y por el afán de poseer mayor conocimiento sobre los Sistemas Windows.

Otro de los motivos de la elección de este tema es debido a que múltiples malwares, pentestings, ataques reales no terminan de infectar un ordenador al no poseer privilegios de administrador, la escalada de privilegios es una sección de elevada importancia que es necesario fortificar correctamente.

Además se ha elegido el framework MITTRE ATT&CK por ser una base global donde se recogen ataques reales hacia empresas y encontrarse en continuo desarrollo y avance, tanto es así que es necesario comentar que actualmente mientras se realiza esta investigación (Mayo 2020) se encuentra vigente la versión v6.3 y se encuentran desarrollando la nueva versión beta de MITTRE ATT&CK [3], futura versión v7.0, en el que se agrupan distintas técnicas como subtécnicas aportando un mayor nivel de abstracción 1.1. Asimismo el trabajo incluye un estudio sobre los límites y carencias de MITTRE ATT&CK, técnicas que no aparecen en el framework pero que permiten escalar privilegios a niveles de administrador. Más adelante se detallará en MITTRE ATT&CK como funciona y estructura dicho framework.

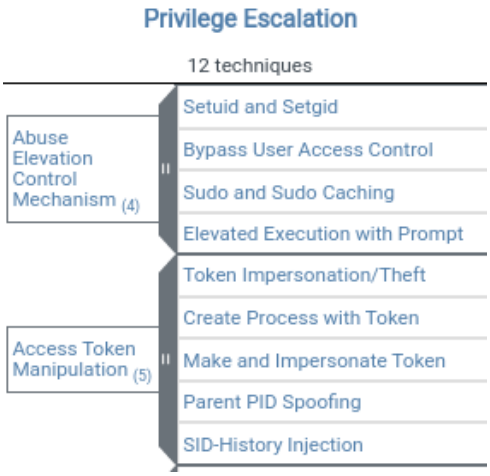

Figura 1.1: MITTRE ATT&CK con subtécnicas en versión beta

1.4 Objetivos

Los objetivos concretos del trabajo es, entre otros, desarrollar una herramienta para defendernos frente a escaladas privilegios, esto es, que se utilice para Blue Team. Para esta parte se está desarrollando

[1] https://github.com/pentestmonkey/windows-privesc-check
[2] https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite
[3] https://attack.mitre.org/beta/

actualmente Mitre Shield [4], una base de conocimiento de defensa para organizar lo aprendido acerca de la defensa activa y el compromiso del adversario, puede ser interesante estudiarlo y basarnos en dichas técnicas. La finalidad es que la herramienta posea modularidad para poder añadir y eliminar técnicas, así también podremos reutilizar la herramienta para orientarlo a otras tácticas del framework como obtener persistencia. Partiremos de las técnicas encontradas en MITTRE ATT&CK y después se estudiarán los trabajos previos.

Además otro de los objetivos es estudiar qué vulnerabilidades aparecen en trabajos previos y no se encuentran en el framework MITTRE ATT&CK para comparar las diferencias y carencias entre ellas y establecer una herramienta más potente.

Para finalizar se ejecutará la herramienta sobre máquinas vulnerables para asegurar de la eficacia de esta, se ejecutará sobre una máquina en producción y máquinas virtuales localizadas bajo el ordenador local. Las imágenes de las máquinas poseen una vulnerabilidad para poder atacar y escalar privilegios y se han descargado bajo la página oficial de VulnHub [5]. Con ello se comprobará la eficacia para conocer si realmente es productiva y poder observar que carencias posee en relación con las herramientas Open Source que existen actualmente.

1.5 Reconocimientos

Este proyecto ha recibido financiación del programa de investigación e innovación Horizonte 2020 de la Unión Europea en virtud del acuerdo de subvención No. 826284 (ProTego).

[4] https://shield.mitre.org/
[5] https://www.vulnhub.com/

Capítulo 2

Estudio teórico

Scientists dream about doing great things.
Engineers do them

James A Michener

A continuación se van a exponer las herramientas en las que se ha basado el proyecto y explicar los diferentes métodos de intrusión y explotación, el funcionamiento de ambos sistemas (tanto por gestión de usuarios como de servicios) y las herramientas desarrolladas anteriormente, sin entrar en demasiados detalles y aportando una visión global de la compilación de otras investigaciones.

2.1 MITTRE ATT&CK

MITTRE ATT&CK es una matriz donde se abordan diversas tácticas (tales como persistencia, escalada de privilegios, evasión de defensa, credenciales de acceso) y técnicas que van a implementar esas tácticas (en nuestro caso la táctica de escalada de privilegios tendrá múltiples técnicas como Access Token Manipulation, Accesibility Features, etc). La matriz define estas tácticas cuando el sistema ya se encuentra comprometido, qué es posible realizar cuando una víctima se ha comprometido, cualquier actividad realizada anteriormente se encuentra dentro del marco PRE-ATT&CK (ver figura 2.1).

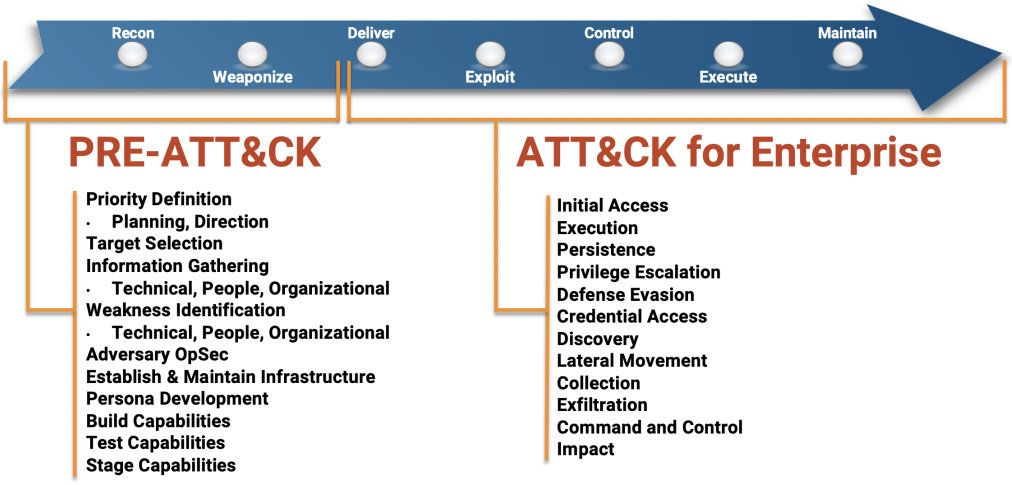

Figura 2.1: Marco PRE-ATT&CK y ATT&CK

Como se ha comentado cada táctica o columna de la matriz hace referencia a una determinada fase cuando un adversario ha conseguido perpetrar una máquina (ordenador personal o servidor) y pretende continuar con dicho ataque, por ejemplo, que el adversario intente ejecutar código malicioso, consiga mantener persistencia, ganar permisos de alto nivel, evitar ser detectado y un largo etcétera. Cada táctica tiene asociadas sus técnicas que se organizan en la misma columna que la táctica en color azul (2.2). Cada técnica contiene un identificador y puede pertenecer a una o varias tácticas, además se muestra información relacionada con dicha técnica como la plataforma sobre la que afecta y la descripción en sí (2.3).

Persistence	Privilege Escalation	Defense Evasion
.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation
Accessibility Features	Accessibility Features	Binary Padding
Account Manipulation	AppCert DLLs	BITS Jobs
AppCert DLLs	AppInit DLLs	Bypass User Account Control
AppInit DLLs	Application Shimming	Clear Command History
Application Shimming	Bypass User Account Control	CMSTP

Figura 2.2: Matriz ATT&CK

ID: T1182
Tactic: Persistence, Privilege Escalation
Platform: Windows
Permissions Required: Administrator, SYSTEM
Effective Permissions: Administrator, SYSTEM
Data Sources: Loaded DLLs, Process monitoring, Windows Registry
Version: 1.0
Created: 16 January 2018
Last Modified: 16 July 2019

Figura 2.3: Información sobre la técnica

Además se proveen de ejemplos sobre ataques reales que han utilizado esta técnica y han podido comprometer a millones de máquinas en todo el mundo, asimismo, el framework dota de unas tablas con mitigaciones para conocer cómo protegernos de esto, por ello es importante conocerlo desde el punto de vista defensivo y ofensivo. Desde el enfoque de un equipo Blue Team es necesario conocer dichas vulnerabilidades para mitigar este ataque, y desde el punto de vista ofensivo para saber cómo aprovecharnos de ello.

Procedure Examples

Name	Description
Honeybee	Honeybee's service-based DLL implant can execute a downloaded file with parameters specified using `CreateProcessAsUser`.[3]
PUNCHBUGGY	PUNCHBUGGY can establish using a AppCertDLLs Registry key.[2]

Mitigations

Mitigation	Description
Execution Prevention	Adversaries install new AppCertDLL binaries to execute this technique. Identify and block potentially malicious software executed through AppCertDLLs functionality by using application whitelisting tools, like Windows Defender Application Control, AppLocker, or Software Restriction Policies where appropriate.

Figura 2.4: Ejemplos reales y mitigaciones para remediar dicha técnica

2.2 Tipos de cuentas

Como se ha comentado el objetivo de este documento es concluir que vulnerabilidades tiene el sistema para que un atacante no pueda conseguir los máximos privilegios de un usuario. Algunas cuentas de usuario que querrán conseguir son los niveles *SYSTEM* o *root*, de administrador local, de cuentas de usuario con acceso de administrador o cuentas de usuario con acceso a un sistema que realice cierta función con privilegios. Para el proyecto actual analizaremos los distintos tipos de cuentas que existen y cómo lo gestionan los distintos sistemas operativos.

2.2.1 Sistemas Windows

Para los sistemas con infraestructuras Windows todas las acciones existentes se realizan desde algún tipo de cuenta, acciones que pueden ir desde realizar servicios asociados a los drivers que se ejecutan cuando el ordenador arranca hasta cualquier acción que realiza un usuario normal, como escribir en un fichero. Una cuenta al fin y al cabo es un conjunto de información que indica al sistema operativo las acciones que puede realizar un determinado usuario sobre ficheros o carpetas específicas.

Estas cuentas pueden ser utilizadas por personas o por ciertos programas como los *System Services* programas que se ejecutan en segundo plano, similar a los *Daemons* de Unix. Cuando un usuario se loggea en un sistema (acceso a perfil de usuario) el sistema produce un token de acceso. Este token incluye la identidad de seguridad y los miembros del grupo del usuario asociado al proceso, por lo que cada proceso que se ejecute por este usuario tendrá una copia del token de acceso del usuario, de esta manera Windows puede controlar qué usuarios o aplicaciones tienen permisos a determinados recursos.

En Windows, una cuenta local [3] es aquella cuyo nombre de usuario y contraseña cifrada se almacenan en el propio ordenador. Cuando se inicia la sesión como usuario local, el ordenador verifica su propia lista de usuarios y su propio archivo de contraseñas para ver si se le permite iniciar la sesión en la computadora. La propia computadora aplica entonces todos los permisos y restricciones que se le asignan para ese sistema.

Un usuario de dominio [4] es aquel cuyo nombre de usuario y contraseña están almacenados en un Domain Controller en lugar de la computadora en la que el usuario se está conectando. Cuando inicia sesión como usuario del dominio, la computadora pregunta al controlador del dominio qué privilegios se le asignan. Cuando el equipo recibe una respuesta adecuada del controlador del dominio, inicia la sesión con los permisos y restricciones apropiados

- Cuentas locales

  Estas cuentas de usuario se guardan en el servidor local y son las principales cuentas de seguridad para un servidor *standalone*. Se dividen en:

  - Cuentas de usuario local

    Son cuentas que poseen un perfil de usuario, las cuentas por defecto que Windows crea son:

    * Administrador
      Con SID S-1-5-21-domain-500 es la primera cuenta creada durante la instalación. Controla todo el ordenador, directorios, servicios y otros recursos, permite instalar nuevos programas y decide la creación de nuevos usuarios, no puede ser borrado. El objetivo es evitar que consigan dicha cuenta.
      Por defecto la cuenta de Administrador forma parte del grupo de Administradores, que tampoco puede ser borrado del grupo.

    * Invitado
      Por defecto esta cuenta se encuentra desactivada. Todos pueden acceder debido a que no es necesario contraseña para ingresar, el escritorio se comparte entre todos los que acceden. No puede instalar nuevos programas pero sí ejecutarlos y es la única cuenta miembro del grupo invitados (su SID siempre tiene el valor S-1-5-32-546).

  Además es posible crear nuevas cuentas de usuarios manualmente, esta cuenta se denomina:

    * Usuario estándar
      Esta cuenta dependiendo de si posee permisos de Administrador puede instalar o no nuevos programas, independientemente puede ejecutar dichos programas. Se recomienda utilizar una cuenta de usuario por cada persona que utilice el ordenador, de esta forma cada usuario tendrá su propio escritorio y programas instalados.

  – Cuentas de sistema local

    También conocido como *Service Account*, son cuentas que Windows incorpora por defecto para proporcionar ciertas limitaciones de seguridad para los servicios que se ejecutan, de esta manera no todas las cuentas de servicio ofrece privilegios de Administrador. Estas *System Accounts* o *Service Accounts* son pseudo-cuentas que Windows crea antes de que un usuario normal se logee, se incorpora para ejecutar servicios Windows y servicios de terceras partes. Según la funcionalidad de cada servicio existen diversos tipos de *Service Account* [5], las principales son:

    * SYSTEM o LocalSystem
      La cuenta más importante del sistema. Esta cuenta de Windows incorporada internamente incluye entre sus tokens NT AUTHORITY/SYSTEM (grupo más poderoso de Windows) y BUILTIN\Administrators, del que se detallará en más profundidad posteriormente.
      Sin embargo las principales diferencias entre la *LocalSystem Account* y una cuenta *root* de Linux radica principalmente que la cuenta *LocalSystem* no tiene perfil de usuario y existen ciertos objetos a los que el sistema no puede acceder directamente (sin embargo mediante un token puede ganar acceso a ello) [6].
    * NETWORK SERVICE
      Cuenta local utilizado por el gestor de control de servicios (Service Control Manager). Cuenta con menos privilegios y presenta credenciales de los servidores remotos. Pertenece al grupo NT AUTHORITY\NetworkService.
    * LOCAL SERVICE
      También se utiliza por el gestor de control de servicios, posee los mínimos privilegios del ordenador local y credenciales anónimas sobre la red. Pertenece al grupo NT AUTHORITY\LocalService.

- Cuentas de dominio.

  Además de las cuentas locales, es posible crear cuentas de usuario para una organización. Las cuentas de dominio se crean y localizan como objetos en *Active Directory Domain Services (ADDS)*, contenedores lógicos que permiten administrar usuarios, grupos, ordenadores y recursos de una red. Para ello es necesario crear un dominio e instalar un controlador de dominios en un Windows Server.

  – Cuentas de usuarios de dominio

    * Administrador de dominios
      Tiene control total sobre todos los recursos del servidor local, puede restringir ordenadores según la política de la empresa.

* Invitado
  Tiene acceso limitado y por defecto está desactivado.
* Usuario de dominio
  Cuenta en la que el nombre de usuario y contraseña se encuentra guardado en un controlador de dominio en lugar del ordenador local.

– Cuentas de sistema local
  Al ser un equipo común posee las mismas cuentas de sistema que las cuentas locales, `SYSTEM`, `NETWORK SERVICE`, `LOCAL SERVICE`...

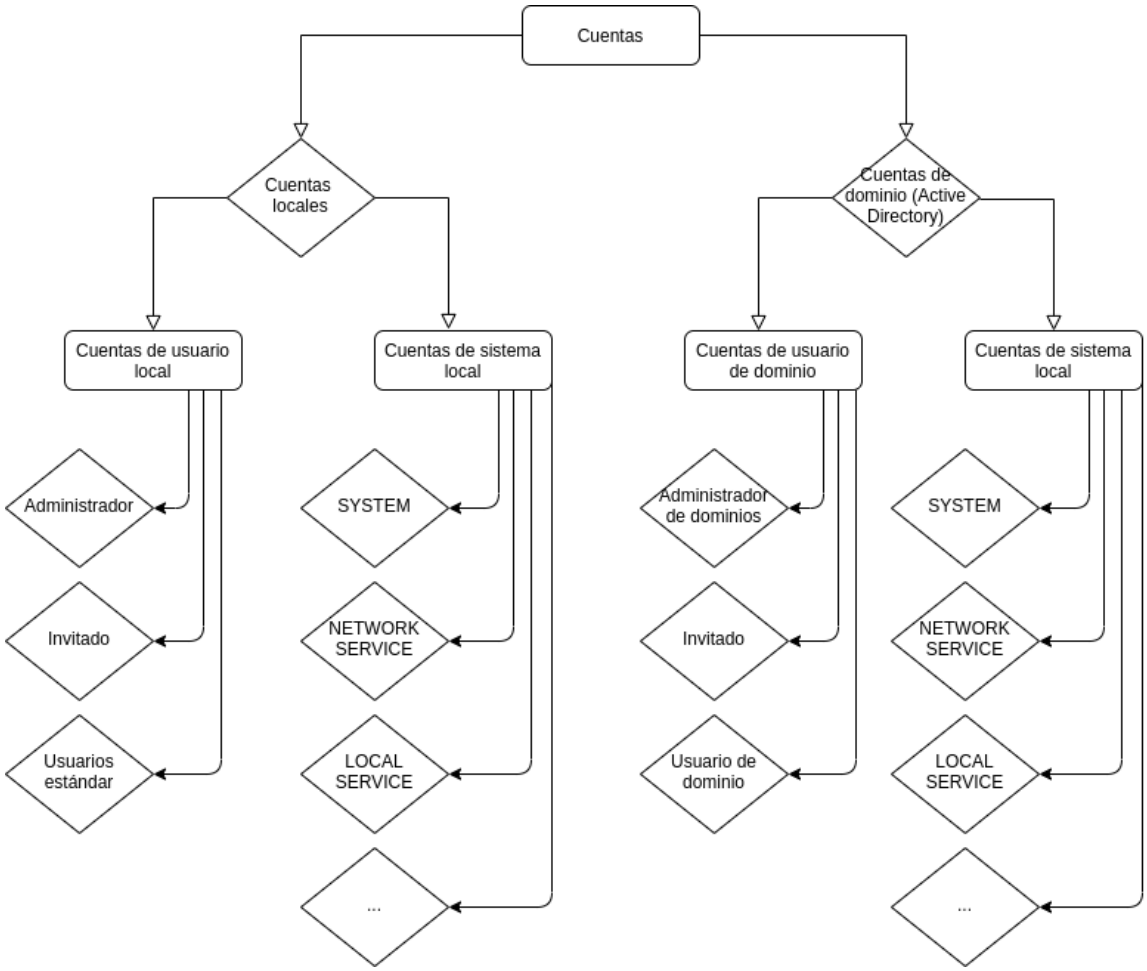

Figura 2.5: Cuentas de Windows

Con todo esto, existen diferencias entre la cuenta con mayor privilegios de una *Service Account* (`SYSTEM`) y la cuenta con máximos privilegios de una *User account* (Administrador). Si un ordenador está unido a un dominio, los procesos que corren tras `SYSTEM` pueden acceder los servidores de dominio como una cuenta de dominio. Sin embargo, los procesos que se ejecutan como Administrador no tienen acceso a los ordenadores de dominio por defecto [7].

2.2.2 Sistemas Linux

Como se ha explicado, al encender un sistema Windows los primeros servicios que se ejecutan son gestionados por las `Service Accounts` que realizan tareas internas del sistema (`System Services`),

más adelante entran en acción las `User Accounts` cuando un usuario se loggea en el sistema. En los sistemas Linux el primer servicio que se ejecuta se denomina Init [8], un `daemon` que se ejecuta durante todo el ciclo de un sistema, desde que se enciende hasta que se apaga. Init es el primer proceso que se crea y por ello se le asigna el PID 1, y todos los procesos se crean a partir de él.

El dueño del servicio Init es el usuario *root* o superusuario, usuario con todos los privilegios posibles del sistema, puede realizar cualquier servicio y el propósito será descubrir las vulnerabilidades que existen para que un ciberdelincuente pueda llegar hasta él para poder mitigarlas.

Por debajo de esta cuenta se encuentra la cuenta de usuario regular, creada cuando un usuario quiere acceder al sistema, dicho usuario por defecto no tiene permisos de administrador, sin embargo puede dar órdenes a *root* mediante los comandos `sudo` y `su` para acceder a determinadas funciones que tiene acotadas, siempre y cuando tenga los permisos correspondientes en `/etc/sudoers`, para realizar una escalada de privilegios legítima.

El último tipo de usuario es la cuenta de servicio o *Service Account*, creadas por lo paquetes de instalación, estas cuentas son usadas por servicios para ejecutar procesos y funciones, nunca se deben de utilizar para trabajar y por ello no disponen de un perfil de usuario y no disponen de un "login" apropiado (si se observa no se dispone de una shell en el fichero `/etc/passwd` asociado a los *Service Account*).

En Linux existe un equivalente al Active Directory de Windows, que se denomina FreeIPA[1], es un gestor de paquetes de identidades que agrupa OpenLDAP, Kerberos, DNS, NTP. Sin embargo no se va a centrar en ello debido a que la idea es focalizarse en cuentas locales.

2.3 SSOO

En esta sección se va a explicar acerca del funcionamiento interno de los sistemas Windows y Linux, para aportar una visión global y posteriormente conocer qué puntos de entrada y explotación tienen dichos sistemas.

2.3.1 Sistema operativo Windows

Para los sistemas Windows las principales características y diferencias que poseen frente a otros sistemas son los siguientes:

2.3.1.1 Editor de registro - Windows Registry

Es una colección de base de datos que almacena los ajustes de configuración de bajo nivel de los programas y el sistema operativo. Posee información acerca de ficheros de configuración de hardware y software para que Windows conozca cómo actuar frente a determinados archivos. Se accede a través el `Registry Editor` o `regedit` y se estructuran en "colmenas registro". Cuando un programa se instala se añaden distintas referencias, instrucciones o claves de registro al Windows Registry para especificar directorios o información relacionada con el programa 2.6.

La gran desventaja de este servicio es que toda la información de configuración se encuentra centralizada, siendo un punto crítico en sistemas Windows.

[1] https://www.freeipa.org/page/Main_Page

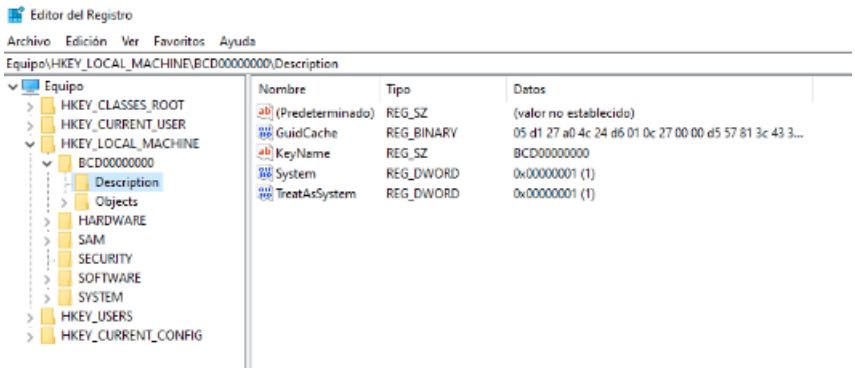

Figura 2.6: Regedit

2.3.1.2 Windows API functions - WinAPI

Es un conjunto de APIs disponible en los sistemas operativos Windows. El sistema operativo interactúa con las WinAPI para desarrollar aplicaciones. Las funciones de la API de Windows se guardan en DLLs como funciones exportadas para interactuar de manera más sencilla con los programas.

Los componentes principales de la API de Windows son:

- WinBase

  Las funciones del kernel, CreateFile, CreateProcess, etc.

- WinUser

  Las funciones de interfaz, CreateWindow, RegisterClass, etc.

- WinGDI

  Las funciones gráficas, Ellipse, SelectObject, etc.

- Controles comunes

  Controles estándar, vistas de lista, controles deslizantes, etc.

2.3.1.3 Import Address Table

El IAT es una tabla de llamada de módulos en espacio de usuario. Los ejecutables de Windows poseen una o más IATs que guardan la dirección de funciones específicas importadas de las DLLs, es decir, un puntero que apunta a funciones específicas que normalmente suelen utilizarse funciones API.

Además de una “tabla de llamada” la IAT provee una vista sobre las diferentes localizaciones de donde se guardan dichas llamadas.

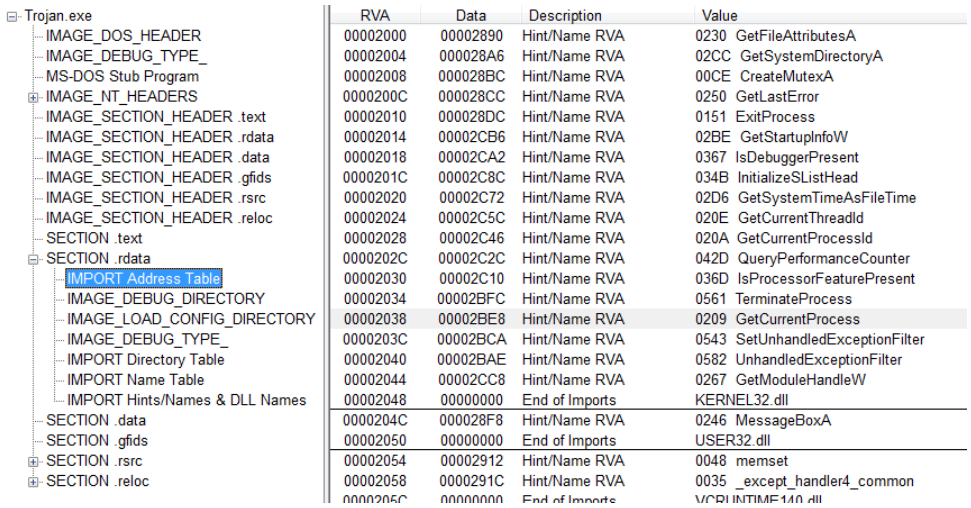

Figura 2.7: Import Address Table [1]

2.3.1.4 Active Directory Domain Services (ADDS)

Las ADDS son un servicio de Windows que permite gestionar y guardar información sobre recursos en una red, como la instalación o actualización de las aplicaciones o la gestión usuarios que se ejecutan en nuestra red, cuando un ordenador se encuentra corriendo este servicio se denomina `domain controller`.

2.3.1.5 Misceláneo

- svchost.exe

  El proceso `svchost` se encuentra corriendo siempre y ejecuta constantemente DLLs, existen múltiples `svchost.exe` para segmentar riesgos y organizados por grupos lógicos.

- Ficheros con extensión .msc

  Son ficheros "Microsoft Saved Console", existen multitud de fichero msc que permiten tener personalizaciones de la consola o módulos para usar conectores para ejecutar tareas administrativas. Están asociadas con la consola de gestión de Microsoft (MMC), un framework para tareas administrativas donde estas herramientas operan.

- sigcheck

  Sigcheck es una herramienta para mostrar las versiones del sistema, información sobre el timestamp, estado de virus total y detalles acerca de firmas digitales.

2.3.2 Sistema operativo Linux

En sistemas Linux todo es tratado como un fichero, incluidos los procesos y dispositivos hardware. Estos ficheros se organizan en forma de árbol de tal manera que la raíz de todos los ficheros es el símbolo `/`. De él emanan los demás ficheros como si de ramas se trataran. Los directorios más importantes del sistema Linux son:

2.3.2.1 Directorios

- `/bin`

Contiene los ejecutables y comandos más importantes del sistema.

- `/etc`

  Directorio donde se almacena los archivos de configuración del sistema.

- `/var`

  Almacena información variable, desde ficheros de registro (logs), información del sistema, usuarios, bases de datos, etc.

- `/home`

  Directorio donde se almacena la información personal de cada usuario que utilice el sistema.

- `/usr`

  Donde se almacenan las aplicaciones no locales, de tal manera que cualquier usuario del sistema puede utilizar las aplicaciones que existen en él.

- `/tmp`

  Directorio temporal utilizado por las aplicaciones para generar ficheros en tiempo real. Este directorio tiene todos los permisos activados por lo que cualquier usuario puede leer, escribir y ejecutar archivos que se encuentren en dicho directorio.

2.3.2.2 Permisos de ficheros

En Linux existen tres niveles de permisos según la cuenta que se utiliza: Usuario, Grupo y otros. Cuando se lista los permisos de un fichero aparecen diez bits asociados a ellos. El primer bit corresponde con el tipo de archivo, archivo común (-), directorio (d), link (l) o binario (b).

Los siguientes nueve bits corresponden a los permisos del archivo, dividiéndose en grupos de tres. Los primeros tres bits corresponden a los permisos del usuario, los tres siguientes al grupo del archivo y los últimos tres bits para el resto de usuarios.

En función de las letras que aparezcan se otorgará permisos de lectura (read, r), escritura (write, w) y ejecución (execution, x) para la cuenta a la que se pertenezca.

En la figura 2.8 se muestra que el archivo `ejemplo.txt` es un archivo común (debido al primer símbolo, -), tiene permisos de lectura y escritura para el usuario kevin, permiso de lectura para el grupo kevin y tambien permiso de lectura para otros.

Es necesario añadir que existe un tipo de bit s (Setuid and Setgid) que se explicará más adelante y puede conllevar en una posible escalada de privilegios.

```
~/prueba
→ ls -l
total 0
-rw-r--r-- 1 kevin kevin 0 jul  1 16:56 ejemplo.txt
```

Figura 2.8: Permisos de un archivo

2.3.2.3 Ficheros críticos

A continuación se van a listar algunos ficheros del sistema, que si se encuentran indebidamente configurado puede suponer un peligro para el propio sistema.

- `/etc/passwd`

  Fichero donde se especifican información acerca de las cuentas del sistema, si aparece una 'x' despues del nombre de la cuenta, la contraseña de la cuenta aparece hasheada en el fichero '/etc/shadow'. Si el fichero tiene permisos de escritura puede comprometerse para escalar privilegios.

- `/etc/shadow`

  Fichero donde se almacenan las contraseñas hasheadas de las cuentas. Dicho fichero se vuelve un problema cuando posee permisos de escritura y/o escritura para cualquier usuario.

- `/etc/sudoers`

  Archivo de configuración de sudo, especifica qué usuarios permiten escalar privilegios de manera controlada y para qué programas.

- `/tmp`

  Al tener permisos de lectura, escritura y ejecución es posible crear y ejecutar exploits bajo este directorio.

- `/etc/services`

  El fichero `/etc/services` especifica el nombre, número de puerto, protocolo que se utiliza y alias de los servicios de red, es posible cambiar los alias de los distintos protocolos y abrir un servicio sospechoso en dicho fichero y simular que se ejecuta otro protocolo.

- `crontab` o `/etc/init.d`

  Es posible conseguir persistencia si se inserta código malicioso en *daemons* o ficheros programados (cron), si dichos procesos se ejecutan bajo permisos de administrador es posible escalar privilegios. *systemV* maneja procesos a través de shell scripts alojados en `/etc/init*`, sin embargo *systemD* maneja procesos de arranque a través de ficheros con extensión `.service`.

2.4 Técnicas de escalada de privilegios

Como se ha comentado, cada táctica tiene asociadas unas técnicas basadas en ataques reales, estas técnicas se han dividido según el sistema operativo vulnerable, dichas técnicas se van a analizar para aportar una visión global de las vulnerabilidades y conocer cómo funcionará el script a desarrollar [9].

Además se ha adjuntado una tabla como se puede apreciar en el Apéndice A para aportar una visión más esquemática de dichas técnicas. Los nombres de las técnicas se han mantenido en inglés debido a que es el idioma nativo de MITTRE ATT&CK.

2.4.1 Windows

Comenzando por los sistemas operativos Windows, cuyo objetivo como se ha comentado es mitigar cualquier vulnerabilidad para denegar que cualquier usuario no legítimo pueda conseguir la cuenta de usuario `Administrador` o la cuenta de servicio `LocalSystem Account` o también denominada `SYSTEM Account`, que pertenece al grupo `NT AUTHORITY/SYSTEM` como se ha descrito anteriormente.

2.4.1.1 Access Token Manipulation

Los procesos Windows funcionan bajo tokens de acceso, el dueño de un proceso en ejecución tiene asociado un token, y en función del tipo de este puede realizar ciertas funciones y/o aplicar determinadas restricciones. Los administradores normalmente se logean como usuarios normales y utilizan el comando `runas` para elevar privilegios de manera controlada.

Bajo el comando `> runas /user:prueba cmd` un usuario puede abrir un cmd con los permisos del usuario `prueba`. El objetivo es robar el token de acceso para ejecutar procesos con privilegios de administrador. Normalmente se utiliza para elevar privilegios de nivel de administrador a `SYSTEM`, sin embargo cualquier usuario puede utilizar las funciones de la API de Windows y el comando `runas`.

Los tokens de acceso se pueden conseguir de tres modos diferentes:

- Token Impersonation/Theft - Robo/Suplantación de Token

  Mediante el método `DuplicateToken(Ex)` es posible conseguir un duplicado del token de acceso. Después con los métodos `ImpersonateLoggedOnUser` y `SetThreadToken` se asigna el token a un hilo específico.

- Create Process with a Token - Creación de un Proceso con Token

  Un usuario duplica el token de un proceso abierto mediante el método `DuplicateToken(Ex)` y seguidamente lo utiliza para crear un nuevo proceso con el token duplicado con el método `CreateProcessWithTokenW`.

- Make and Impersonate token - Creación y Suplantación de Token

  Un usuario puede crear una sesión de inicio mediante la función `LogonUser`. Esta función devuelve una copia del token de acceso de la nueva sesión y asignarlo a un hilo mediante la función `SetThreadToken`.

2.4.1.2 Accesibility Features

Windows ofrece características de accesibilidad para adaptar a las personas a sus necesidades auditivas, de visión, etc. Sin embargo es posible aprovecharse de estas características para abrir una shell o crear una puerta trasera.

Las `Accesibility Features` más importantes son `C:\Windows\System32\setch.exe` y `C:\Windows\System32\utilman.exe` como aparecen en las figuras 2.9 y 2.10 respectivamente. Existen dos métodos para aprovecharse de este fallo dependiendo de la versión de Windows:

- Binary replacement

  Se sustituye el binario legítimo, por ejemplo `utilman.exe` por una shell, `cmd.exe` de tal manera que cuando se vaya a ejecutar el programa legítimo se abra un cmd con permisos administrativos.

- Debugger method

  Se modifican las claves de registro para abrir un cmd en lugar de un depurador, esto causará que se abra una shell también con permisos administrativos.

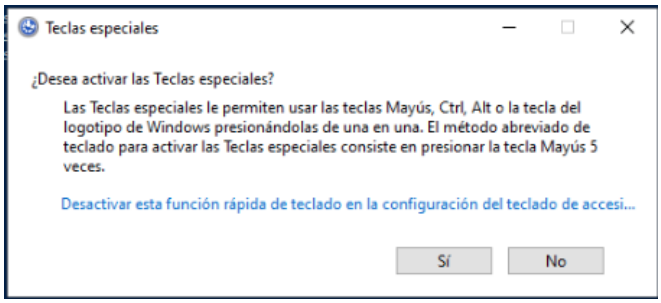

Figura 2.9: setch.exe

Figura 2.10: Botón para abrir utilman.exe

2.4.1.3 AppCert DLLs

Las DLLs que se especifican bajo AppCertDLLs en la clave de registro `HKLM\System\CurrentControlSet\Control\Session Manager` se cargan en los procesos que usan las funciones de la API Win32 como `CreateProcess`, `CreateProcessAsUser`, `CreateProcessWithLogonW`, `CreateProcessWithTokenW` y `WinExec`.

Al igual que en Process Injection podemos aprovecharnos de esto para escalar privilegios realizando que cargue una DLL maliciosa.

2.4.1.4 AppInit DLLs

Las DLLs que se encuentran bajo AppInit_DLLs en la clave de registro `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows` o en `HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Windows` se cargan debido a la llamada de la biblioteca user32.dll (la mayoría de programas cargan dicha librería).

Al igual que en la sección anterior y en Process Injection los usuarios ilegítimos se pueden aprovechar de estos valores para escalar privilegios realizando que cargue una DLL maliciosa. Sin embargo esta característica esta deshabilitada para sistemas Windows 8 y posteriores si está activado el booteo seguro.

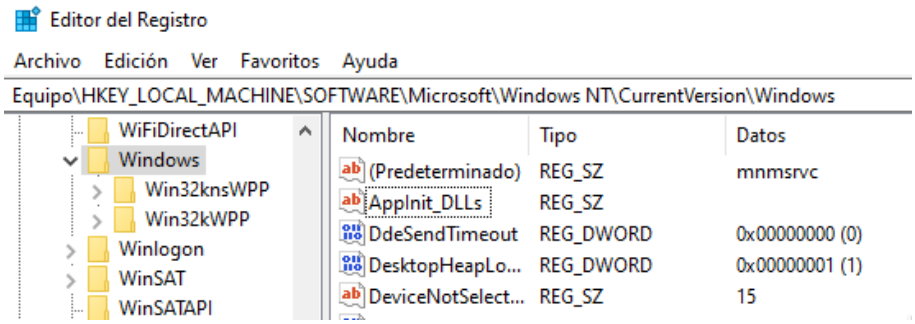

Figura 2.11: Registry Key HKLM\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Windows

2.4.1.5 Application Shimming

El framework `Application Shim` se creó para permitir la retrocompatibilidad (Que un sistema ejecute programas orientado a versiones anteriores a la suya [10]).

Los `Shims` permiten aplicar parches sin reescribir código, esta infraestructura implementa una especie de API hooking (Intercepta las llamadas y modifica el comportamiento y el flujo de las llamadas API

[11]). Las llamadas a binarios externos se realiza a través del `Import Address Table` (IAT). Una llamada a Windows a partir de una aplicación normal se realiza como se describe en la figura 2.12.

Figura 2.12: Llamada al sistema operativo a través del IAT [2]

Cuando se ejecuta un programa, se referencia a la caché Shim para determinar si es necesario utilizar la base de datos shim (archivos .sdb). En caso afirmativo la base de datos redirige el código para comunicarse con el sistema operativo Windows tal y como se describe en la figura 2.13.

Figura 2.13: Llamada al sistema operativo mediante Shim [2]

Sin embargo las Shims se localizan fuera del modo de usuario por lo que se necesitan permisos de administrador para instalar una. No obstante existen ciertas Shims como `RedirectEXE, InjectDLL, DisableNX, DisableSEH y GetProcAddress` que realizan ciertas acciones que utilizadas incorrectamente permiten aprovecharnos del sistema para elevar privilegios [12].

2.4.1.6 Bypass User Account Control

El User Account Control permite elevar privilegios a un programa solicitando al usuario confirmación. Si el usuario actual se encuentra dentro del grupo de Administradores aparecerá una pantalla como en la figura 2.14, en caso contrario, si el usuario no tien permisos de Administrador se solicitará que ingrese la contraseña de algún administrador, en la figura 2.15 el usuario *prueba* no se encuentra dentro del grupo Administradores y debe escribir la contraseña del usuario administrador *test1*.

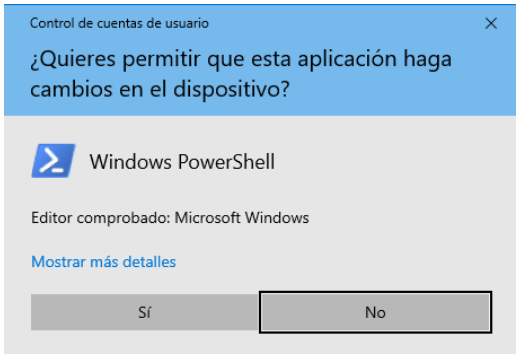

Figura 2.14: User Account control

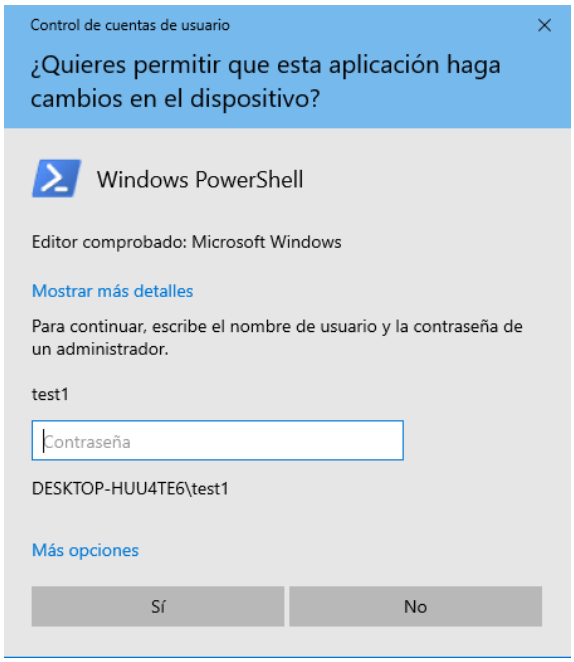

Figura 2.15: UAC con usuario sin permisos

Aunque el nivel de un usuario no esté configurado dentro del grupo Administradores, ciertos programas se ejecutan en un nivel privilegiado o ciertos objetos COM (La elevación COM o COM Elevation Moniker se utiliza para realizar una función específica y limitada que requiere privilegios elevados) sin preguntar al usuario a través del UAC, dichas aplicaciones pueden ser aprovechadas por adversarios para inyectar código malicioso y escalar privilegios, estas aplicaciones se muestran en el apéndice B y se ha conseguido gracias a *The WayBack Machine* [2], debido a que actualmente dicha entrada se encuentra eliminada [13]. Gracias a esta lista se descubrió que el ejecutable `RunDll32.exe` podía ejecutar código dentro de una DLL con permisos de administrador sin la ventana UAC.

Además, en Github existe un repositorio orientado a bypassear el Windows User Account Control mediante diferentes maneras, como se detallará más adelante [14].

2.4.1.7 DLL Search Order Hijacking

Un sistema puede contener múltiples versiones de una DLL, las aplicaciones siguen un órden a la hora de buscar dichas DLLs, si no se encuentra en un directorio busca en el siguiente y así sucesivamente hasta encontrar la DLL deseada. Es posible inyectar una DLL maliciosa antes de una legítima aprovechándonos del órden de búsqueda (DLL preloading), siempre que la ruta tenga permisos de escritura. Si la DLL legítima se ejecutara con permisos de alto nivel una DLL maliciosa se puede aprovechar para escalar privilegios.

También es posible modificar archivos `.manifest` (ficheros XML con información acerca de clases COM, librerías y ensamblado) o `.local` (ficheros de redirección hacia el directorio de la aplicación [15]) para que el programa cargue DLLs maliciosas. Uno de los repositorios más conocidos para crear DLL malignas es PowerSploit [16].

El órden de búsqueda de DLLs que realizan las aplicaciones es [17]:

1. El directorio desde donde se ejecutó la aplicación

[2] https://archive.org/web/

2. El directorio actual
3. El directorio del sistema (`GetSystemDirectory`)
4. El directorio de Windows (`GetWindowsDirectory`)
5. Los directorios que se enumeran en la variable de entorno `PATH`

2.4.1.8 Exploitation for Privilege Escalation

Las vulnerabilidades del kernel, bugs en servicios o errores de programación en el sistema se pueden aprovechar para explotar y conseguir permisos de administrador, estas vulnerabilidades pueden estar recogidas en CVEs (Lista de vulnerabilidades existentes [3]), MS (Estandarización del ciclo de parches de Microsoft [4]) o en *zero days*.

Existen repositorios que analizan la información del sistema en Windows (mediante `> systeminfo`) y arroja las vulnerabilidades del sistema [18] [19] [20], CVEs y MS. Con ello se pueden encontrar exploits públicos para aprovecharnos de las vulnerabilidades encontradas [21] [22].

2.4.1.9 File System Permissions Weakness

Todos los procesos ejecutan programas específicos, si estos programas o los directorios donde se ubican los programas, no se encuentran correctamente configurados (en base a sus permisos) es posible sobreescribir o reemplazar dicho ejecutable por uno malicioso, y si el sistema corre el proceso con privilegios de administrador, el binario malicioso que lo reemplace también correrá bajo esos permisos.

Una variante de esta técnica es reemplazar servicios Windows en lugar de procesos, para ejecutar el servicio malicioso bajo los permisos de la *Service Account* que la ejecute 2.2.1.

Otra variante son los instaladores auto-extraíbles, cuando se instala un programa es común que se creen directorios y subdirectorios bajo la carpeta `%TEMP%` para descomprimir ciertos archivos. Como la carpeta `%TEMP%` tiene permisos de escritura, si el programa tiene mal configurados los permisos de los ficheros es posible poder crear archivos maliciosos para sustituir o sobreescribir los originales para que se ejecuten bajo el instalador, si el instalador se ejecuta bajo permisos de Administrador se podrá escalar privilegios. Esta técnica está relacionada con DLL Search Order Hijacking (2.4.1.7) y Bypass User Account Control (2.4.1.6).

2.4.1.10 New Service

Como se ha comentado anteriormente, un servicio es una aplicación que corre o ejecuta en segundo plano sin la interacción del usuario, similar a los *daemon* en Unix. Cuando un ordenador bootea, ejecuta determinados servicios (acerca del reporte de errores, logging de eventos, etc), la información de estos servicios se encuentran en el registro de Windows.

Mediante el registro de Windows es posible instalar servicios que se ejecuten al bootear el sistema, creando servicios con nombres del sistema operativo o de Software legítimos. Para instalar estos servicios se necesitan permisos de administrador, pero se ejecutarán bajo permisos SYSTEM, con ello se escalará privilegios de Administrador a SYSTEM.

[3] https://cve.mitre.org/
[4] https://blog.rapid7.com/2014/02/03/new-ms08-067/

2.4.1.11 Path Interception

Se denomina interceptación de la ruta o Path Interception cuando un ejecutable malicioso se encuentra en una ruta específica, de manera que se ejecute por una aplicación en lugar del ejecutable legítimo. Si el ejecutable corre con privilegios de Administrador se podrán escalar privilegios. Existen distintos tipos de interceptación de ruta:

- Unquoted Paths

  En Windows se pueden ejecutar programas sin utilizar su extensión. Esto se puede explotar si las rutas de los servicios que se almacenan en el registo de clave de Windows no se especifican entre comillas dobles y la ruta tiene algún espacio. Debido al orden que utiliza Windows para encontrar ficheros en la siguiente ruta `C:\Program Files\Application\program.exe` primero buscará el binario `C:\Program.exe` debido al espacio y si no existe continuará con la ruta proporcionada. Si se consigue crear un ejecutable con nombre `Program.exe` se ejecutará antes y si se ejecuta con permisos de administrador se conseguirá escalar privilegios. Sin embargo esto se puede remediar añadiendo simplemente comillas dobles entre la ruta.

- PATH Environment Variable Misconfiguration

  Muchas aplicaciones especifican rutas relativas de la variable de entorno `PATH` para funcionar. Las variables de entorno de Windows se pueden contemplar tal y como aparece en la figura 2.16 si se especifica a una ruta antes del directorio Windows (`C:\Windows\System32`) se ejecutará en su lugar si tiene el mismo nombre por el programa que lo llame.

  Por ejemplo si el directorio `C:\ejemplo` se encuentra antes de `C:\Windows\System32`, y ambos contienen el ejecutable `ejecuta.exe` desde el cmd se ejecutará el fichero localizado en el directorio `C:\ejemplo`.

- Search Order Hijacking

  Al igual que en la sección DLL Search Order Hijacking (2.4.1.7) un adversario puede aprovecharse del orden de carga de un proceso si no especifica la ruta. El orden de búsqueda aparece en la sección 2.4.1.7 y también es necesario tener en cuenta el órden de ejecución según la extensión (extensiones `.com` se ejecutan antes que las `.exe`) [23].

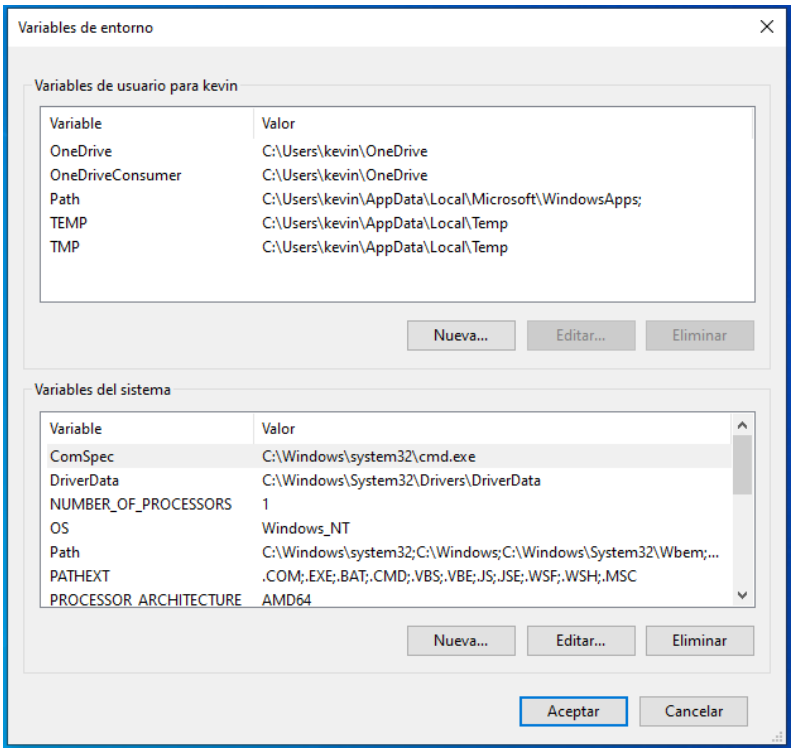

Figura 2.16: Variables de entorno de Windows

2.4.1.12 PowerShell profile

PowerShell es una interfaz por línea de comandos interactiva, también establece un entorno de Scripting. Cuando se abre PowerShell se ejecuta un perfil de PowerShell independientemente para cada usuario (`profile.ps1`), es el equivalente al `.bashrc` en Sistemas Unix [24]

Sin embargo es posible modificar y customizar estos perfiles PowerShell, permitiendo la ejecución de código remoto, llamadas a funciones y métodos. Cada vez que se abra una ventana de PowerShell se ejecutará el perfil PowerShell asociado, si el usuario contiene privilegios de Administrador o SYSTEM se ejecutará el binario con privilegios administrativos.

2.4.1.13 Process Injection

La finalidad de esta técnica es inyectar código en la memoria de otro proceso ya existente, consiguiendo acceso a los recursos de red, sistema y memoria, pudiendo elevar privilegios. Esta técnica consigue bypassear antivirus y herramientas de evasión de malware debido a que se ejecuta bajo un proceso legítimo.

En sistemas Windows existen multitud de tipos de inyección de procesos:

- Dynamic-link library (DLL) injection

  Se refiere a escribir una ruta hacia una DLL maliciosa para que lo ejecute una aplicación legítima.

- Portable executable injection

  Escribe código malicioso directamente en el proceso y después invoca dicha parte con un código adicional. Esta inyección presenta el problema de la relocalización de direcciones, que se arregla mediante variaciones como reflective DLL injection y memory module [25].

- Thread execution hijacking

  Se refiere a la inyección de código malicioso o de una DLL en un hilo de un proceso.

- Asynchronous Procedure Call (APC)

  Las APC son funciones que se ejecutan asincrónicamente en un hilo, cuando el APC está a la cola de un hilo el sistema emite una interrupción de Software y se ejecutará cuando entre en un estado alterable. Es posible inyectar código malicioso a la cola APC de un hilo, como realiza "Early Bird Injection", que crea un proceso suspendido en el cual el código malicioso se ejecutará antes que el programa principal porque se pone a la cola, y después reanuda el proceso suspendido [26].

- Thread Local Storage (TLS)

  Se refiere a modificar el puntero de un ejecutable para apuntar hacia un código malicioso antes de que se ejecute `AddressOfEntryPoint` (Comienzo normal de una ejecución).

2.4.1.14 Scheduled task

Existen herramientas para programas tareas o servicios como son `at` y `schtasks`, siendo el equivalente de `cron` en Sistemas Unix.

Una tarea se puede programar a través de otro ordenador, conectándose a través de llamadas de procedimiento remoto (RPC). Para que sea esto posible es necesario que se encuentre dentro del grupo Administradores del sistema remoto.

Un adversario puede aprovechar esta temporización de tareas para ejecutar programas cuando un ordenador se enciende o de forma programada para escalar privilegios de SYSTEM Account.

2.4.1.15 Service Registry Permissions Weakness

La configuración de servicios Windows se aloja bajo `HKLM\SYSTEM\CurrentControlSet\Services`. Si se puede modificar valores del servicio en el registro de Windows, que se encuentra controlada bajo Listas de Control de Acceso, es posible escalar privilegios hasta niveles de Administrador o SYSTEM modificando el binario para apuntar a un ejecutable malicioso. Además es posible modificar los parámetros de fallo `FailureCommand` para realizar la misma acción.

2.4.1.16 SID-History Injection

Un SID o Identificador de Seguridad es un valor único que identifica una cuenta de usuario, grupo o de sesión. Este SID se fija por medio del Controlador de Dominio de Windows y se guarda en una base de datos. Cuando un usuario se loggea, el sistema devuelve dicho SID y lo fija en el token de acceso del usuario. Además existe un historial SID que contiene múltiples identificadores de seguridad que se han utilizado previamente para cambiar de un dominio a otro, al crear un nuevo SID el anterior queda añadido a dicho historial.

Es posible inyectar SIDs en el historial (Con permisos de Administrador) para suplantar a usuarios y grupos, y con ello escalar privilegios para poseer permisos de una SYSTEM Account, los SIDs a inyectar se puede conocer a través del Administrador de Dominio o de los SIDs ya conocidos [27].

2.4.1.17 Valid accounts

Por medio de ataques de ingeniería social, reconocimiento, recolección de información y Man-in-the-Middle es posible conseguir credenciales de un sistema. Con las credenciales de un sistema es posible bypassear ciertos servicios y/o escalar privilegios hasta nivel de Administrador.

Es necesario atender a los pastes que existen en Internet o realizar técnicas de OSINT y/o por medio de Dorks porque es posible que por error se encuentren tanto contraseñas como claves privadas de ciertos servicios. Asimismo si el ordenador está conectado a la red es posible pivotar hacia otras máquinas.

2.4.1.18 Web Shell

Cuando una página Web tiene una vulnerabilidad es posible inyectar un archivo que abra una shell reversa hacia el cliente, con esto es posible interactuar con el sistema de la página Web y ejecutar comandos.

Cuando el sistema ejecute el fichero para abrir una Shell es posible que se abra con privilegios SYSTEM si por defecto el dueño del servicio se ejecuta con dichos servicios.

2.4.1.19 Extra Window Memory Injection

Antes de abrir una ventana se registra una clase de Windows en el que se estipula la apariencia de esta. Cuando se registra la nueva ventana, la aplicación puede incluir una cabecera de hasta 40 bytes que se añadirá a la memoria de dicha instancia, esto se conoce como Extra Window Memory (EWM). Sin embargo se puede inyectar en estos 40 bytes un puntero de 32 bits hacia una función maliciosa, permitiendo acceso a la memoria del proceso y poder elevar privilegios. Y mediante las funciones API `GetWindowLong` y `SetWindowLong` es posible cambiar valores y apuntar a funciones escritos en la sección compartida.

La finalidad es aprovechar una sección compartida de la memoria del proceso, añadiendo un puntero en el Extra Window Memory para posteriormente invocar la ejecución devolviendo el control de ejecución a la dirección en el EWM del proceso. Esta ejecución que se concede puede permitir el acceso a la memoria del proceso y escalar privilegios.

2.4.1.20 Hooking

Los procesos de Windows utilizan su API para comunicarse con los recursos del sistema, y con ello de interactuar con las DLLs debido a que existen ciertas DLLs que guardan funciones de la API de Windows. Con Hooking nos referimos a interceptar las llamadas de dichas funciones, se pueden realizar de distintas maneras:

- Hooks Procedures

  Intercepta y ejecuta código malicioso para responder a distintos procesos.

- Import address table (IAT) hooking

  Se reemplaza la IAT, cuando una aplicación llama a una API localizada en una DLL, la función maliciosa reemplazada se ejecuta en lugar de la original.

- Inline hooking

  A diferencia del punto anterior, se modifica la función API en lugar de la IAT.

Es posibe cargar y ejecutar código malicioso a causa de otro proceso, pudiendo escalar privilegios. También es posible esconder procesos y claves de registro pudiendo ofuscar malware.

2.4.1.21 Image File Execution Options Injection

La opción Image File Execution permite ejecutar un programa con el depurador. Esto es posible realizarlo directamente creando una clave de registro de Windows (bajo la ruta `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\<ejecutable>`) o utilizando la herramienta *Gflags*.

También es posible monitorizar un objeto cuando un programa se cierra silenciosamente con el modo *Silent Process Exit* de GFlags. El término cerrar silenciosamente se alude cuando el proceso llama a ExitProcess o TerminateProcess y también es posible realizarlo mediante claves de registro de Windows o *Gflags* (ofrece una capa de abstracción para escribir sobre claves de registros).

Ejecutando un programa malicioso cuando un binario legítimo cierra evade `autoruns.exe` (Herramienta que analiza ciertos archivos). Un ejemplo de ls registros que son necesarios crear para abrir un programa malicioso cuando `notepad.exe` cierra es [28]:

- `reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\notepad.exe" /v GlobalFlag /t REG_DWORD /d 512`
- `reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit \notepad.exe" /v ReportingMode /t REG_DWORD /d 1`
- `reg add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit \notepad.exe" /v MonitorProcess /d "C:\temp\evil.exe"`

2.4.1.22 Parent PID Spoofing

Otra técnica utilizada es *spoofear* (hacerse pasar por) el PPID (PID Padre) de un proceso para generar un nuevo proceso, si el proceso padre tiene privilegios de Administrador el proceso hijo también contará con ellos. Asimismo los nuevos procesos evaden procesos de monitorización.

Un ejemplo para asignar el PPID de un proceso es mediante la función API `CreateProcess`, esta función se utiliza para asignar el PPID en el User Account Control para elevar legítimamente privilegios.

Existen APPs que sus interfaces gráficas son inseguras y permiten generar un cmd en lugar de un archivo, es el caso de Paint o Notepad, si se abre la consola con permisos de Administrador se puede conseguir un cmd con los mismos permisos abriendo la ruta del cmd, `c:\windows\system32\cmd.exe` como se indica en la figura 2.17.

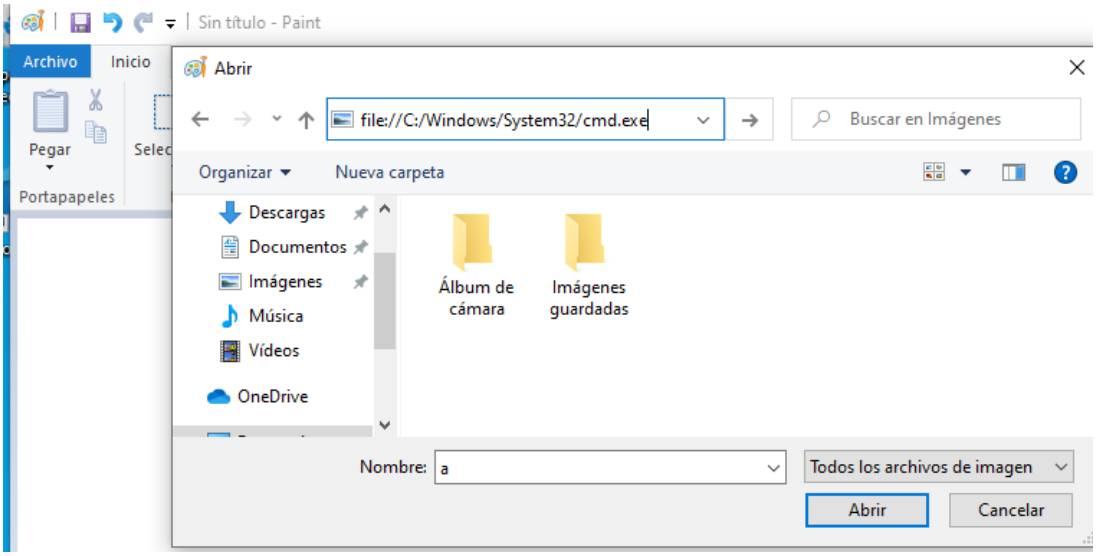

Figura 2.17: Inserción de la ruta del cmd

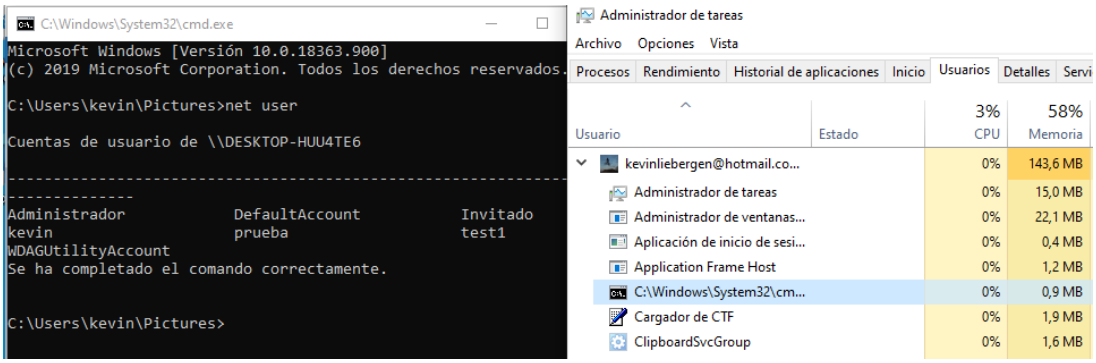

Figura 2.18: El cmd creado bajo el usuario dueño del proceso paint

2.4.1.23 Port Monitors

Se puede fijar una monitorización de un puerto a través de la API de Windows para insertar una DLL en el arranque del sistema, esta DLL (que se tiene que insertar bajo permisos de Administrador) se cargará mediante `spoolsv.exe` que corre bajo permisos de la cuenta de servicio SYSTEM.

Se puede cargar una DLL maliciosa si se puede escribir en el registro `HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors`.

2.4.2 Linux

Continuando con los Sistemas Linux la cuenta con mayor privilegios es la denominada cuenta *root*. Esta cuenta se puede conseguir de manera ilegítima por medio de las siguientes técnicas:

2.4.2.1 Exploitation for Privilege Escalation

Al igual que para sistemas Windows, mediante la técnica Exploitation for Privilege Escalation es posible aprovecharse de las vulnerabilidades en el kernel del sistema o fallos en la configuración, el objetivo principal es conocer los CVEs [5] del sistema desactualizado.

[5] https://cve.mitre.org/

Existen diversos repositorios [29] [30] [31] que facilitan la búsqueda de CVEs y otros [32] [22] que proporcionan exploits públicos, lo que permite explotar el sistema y poder elevar privilegios.

2.4.2.2 Process Injection

Al igual que en Sistemas Windows, la finalidad de los Process Injection es inyectar código en la memoria de un proceso.

Para los sistemas Linux mediante:

- Variables de entorno LD_PRELOAD, LD_LIBRARY_PATH

  Con ello es posible cargar un objecto compartido (`.so`) para interceptar llamadas API y escalar privilegios.

- Ptrace system calls

  La llamada al sistema ptrace permite que un proceso padre puede controlar la ejecución de una imagen y poder cambiar su memoria y registros. Con esta llamada es posible añadir código a un proceso corriendo.

- /proc/[pid]/mem

  Es posible leer la memoria de los procesos localizados en /proc/[pid]/mem y con ello escribir datos en él para escalar privilegios.

- VDSO hijacking

  Virtual Dynamic Shared Object es un mecanismo para acelerar ciertas llamadas al sistema, es un área de memoria que se localiza en espacio de usuario (dinámicamente) que ofrece algunas funcionalidades de kernel de una manera segura. Es parecido a las syscalls o llamadas del sistema, la diferencia es que las sycalls se ejecutan en el espacio de kernel y cambiar entre el espacio de usuario y el kernel es muy lento, por lo que algunas llamadas al sistema que no son necesarias para correr en el kernel se ejecutan en el espacio de usuario [33]. Existen ataques que inyectan binarios sobre la librería compartida `linux-vdso.so`

2.4.2.3 Setuid and Setgid

Los Set User id y Set Group id son bits especiales que pueden asignarse a un archivo y/o directorio. Cuando algunos de estos bits se encuentran activos en una aplicación, y un usuario ejecute dicho fichero, el proceso adquirirá los permisos del propietario del fichero ejecutado, los adversarios aprovechan esta técnica para poder abrir una terminal si es posible y escalar privilegios a nivel de *root*.

Este bit se creó debido a la necesidad de ejecutar ciertos archivos como *root*, en lugar de modificar el archivo `/etc/sudoers` por sencillez se puede activar el bit setuid y setgid para sus aplicaciones.

Para mostrar un ejemplo se ha configurado mal el fichero vim activando el bit setuid como aparece en la imagen 2.19, con esta configuración es posible ejecutar vim con permisos de *root* y por ende abrir ficheros privilegiados como `/etc/shadow` 2.20.

Además vim ofrece la posibilidad de ejecutar comandos como *root* como aparece en las imágenes 2.21 y 2.22.

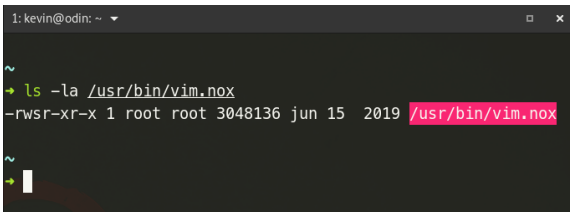

Figura 2.19: Setuid activado en ejecutable vim.nox

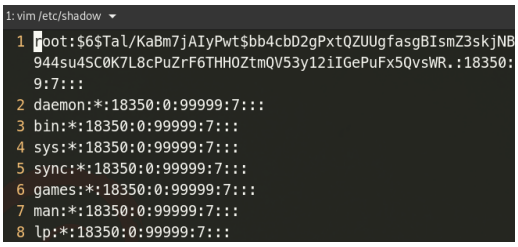

Figura 2.20: Es posible abrir ficheros con permisos de root

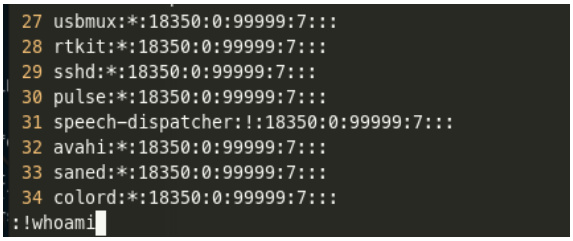

Figura 2.21: Desde vim se puede ejecutar comandos

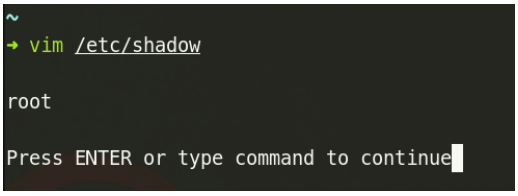

Figura 2.22: Salida del comando anterior

2.4.2.4 Sudo

En el fichero `/etc/sudoers` se detalla que usuarios permiten ejecutar determinados comandos y qué mínimo permiso tienen. Sin embargo el fichero `/etc/sudoers` tiene permisos de administrador por lo que no es posible contemplar dicho fichero si no existe una mala configuración o si no se encuentra en el grupo *root*. Sin embargo es posible contemplar los permisos del usuario actual mediante `sudo -l`.

El fichero `/etc/sudoers` ofrece la función de ejecutar servicios sin insertar la contraseña si se especifica bajo la sentencia `test ALL=(ALL) NOPASSWD: ALL`, siendo `test` usuario del sistema. Con ello no se solicitará la contraseña para ejecutar ningún servicio con permisos de administrador en el sistema.

Los criminales se pueden aprovechar de esta técnica para escalar privilegios y poder ejecutar ciertos servicios como *root* si el fichero tiene permisos de escritura.

2.4.2.5 Sudo caching

Sudo tiene ciertas características especiales como `timestamp_timeout`, define el tiempo que no se volverá a pedir la contraseña desde que se inserta, que por defecto es de cinco minutos. El sistema puede cachear credenciales al crear un fichero en `/var/run/sudo/ts` o `/var/db/sudo` con un timestamp de cuando se insertó por última vez el comando `sudo`.

Además existe el parámetro `tty_tickets` que trata a un terminal como una nueva sesión independiente (Cada vez que se abre una nueva terminal será necesario escribir la contraseña de Administrador para escalar privilegios). Sin embargo si está desactivado, mediante la línea `Defaults !tty_tickets` se puede tratar una nueva sesión de terminal dependiente de otra. Si en la primera sesión se ha introducido ya la contraseña las nuevas sesiones de terminal no se volverá a pedir, de esta manera se comportaba el malware OSX Proton [34].

Es posible monitorizar el archivo `/var/run/sudo/ts` para saber si se ha escrito sudo en algún terminal (esté dentro del rango `timestamp_timeout`). Con esto es posible inyectar comandos con sudo

sin la necesidad de insertar la contraseña en el mismo terminal, o en distinto si está la `tty_tickets` desactivada.

2.4.2.6 Valid Accounts

Al igual que en los sistemas Windows, que aparece más detallado, la técnica Valid accounts engloba el robo de credenciales de distinto tipo para conseguir introducirse en el sistema y escalar privilegios. Además es necesario tener en cuenta cambiar las contraseñas por defecto de los servicios que se instalan en el sistema.

2.4.2.7 Web Shell

Al igual que en sistemas Windows, la técnica Web Shell establece que una web shell puede proporcionar un conjunto de funciones para ejecutar comandos. Si el proceso que abre una Web Shell contiene permisos de Administrador la shell también contará con ellos y se habrá escalado privilegios.

2.5 Trabajos previos

Para finalizar el estudio teórico se ha realizado un estudio de mercado para conocer las herramientas comerciales y Open Source existentes orientadas a escaladas de privilegios, se han seleccionado dos programas importantes utilizados a diario en ambientes laborales o en CTFs cuya finalidad es encontrar fallas en el sistema para poder escalar privilegios, las herramientas que se han seleccionado son las que se muestran a continuación, PentestMonkey y Privilege Escalation Awesome Scripts SUITE:

2.5.1 PentestMonkey

PentestMonkey [6] [7] es una herramienta cuya función es encontrar problemas de configuración que permiten escalar privilegios hacia otros usuario o acceder a otras aplicaciones, su fuerza radica en la búsqueda de ficheros con determinados problemas que se explican a continuación.

1. Lo primero que realiza PentestMonkey es generar un fichero de caché denominado `files_cache.$$` siendo `$$` el número del PID actual, este archivo almacenará todos los ficheros del sistema junto con sus permisos.
2. Una vez se ha generado el fichero de caché se procede a comprobar ficheros críticos con permisos de lectura por cualquier usuario (World-readable). Dichos ficheros que encuentre los imprimirá por pantalla al igual que los ficheros que encuentre que cumplan los siguientes puntos.
3. Seguidamente PentestMonkey procede a buscar ficheros con permisos de escritura por grupo (Group writable).
4. Posteriormente se procede buscar ficheros que almacenan historiales con permisos de lectura (history readable).
5. Una vez finalizado el punto anterior se comprueban ficheros con permisos de lectura y ejecución (homedirs_executable) como `/run/sshd, /var/www`, etc.

[6] https://github.com/pentestmonkey/unix-privesc-check
[7] https://github.com/pentestmonkey/windows-privesc-check

6. Más adelante se procede a buscar archivos `.jar` con permisos de lectura.

7. Mediante key_material se listan posibles claves críticas (con extensión `.cer`, `.crt`, `.pem`, `.p12`, `.keystore` y `.key`).

8. Seguidamente mediante passwd_hashes se listan usuarios sin contraseñas buscando en el fichero `/etc/passwd`.

9. Con privileged_banned se listan determinados ficheros que llaman a funciones banneadas como `alloca`, `gets`, `memcpy`, `scanf`, `sprintf`, `sscanf`, `strcat`, `strcpy`, `strlen`, `strncat`, `strncpy`, `strtok`, `swprintf`, `vsnprintf`, `vsprintf`, `vswprintf`, `wcscat`, `wcscpy`, `wcslen`, `wcsncat`, `wcsncpy`, `wcstok` y `wmemcpy`, basadas en la lista de APIs banneadas de Microsoft [35].

10. A continuación con privileged_path se comprueba qué ficheros utilizan la variable de entorno PATH para llamar a otros ficheros (Rutas relativas).

11. Con privileged_pie (Position Independent Executable) se comprueba que los ficheros cumplen las normas ASLR basados en [36].

12. Por medio de privileged_random se procede a comprobar si existen ficheros con permisos de administrador que llaman a funciones aleatorias.

13. Posteriormente la sección privileged_relro o privileged Relocation Read Only comprueba si posee soporte relro o los ficheros poseen técnicas para hardenizar binarios ELF [37].

14. Además mediante privileged_ssp se buscan ficheros que no tengan protección contra buffer Overflow (Stack smashing Protector).

15. Asimismo privileged_tmp localiza si existen ficheros con permisos de administrador que llaman a archivos temporales (bajo el directorio `/tmp`).

16. Seguidamente se procede a buscar ficheros con el bit setgid o setuid activados.

17. Para finalizar shadow_hashes busca usuarios con contraseñas en texto plano.

```
credentials_permissions () {
        pattern="${1}"
        file_show_non_symlink_perms " ${pattern}$" | while read filename permissions userid groupid
        do
                case "${permissions}" in
                        ???????r??)
                                stdio_message_warn "credentials" "${filename} is owned by user ${userid
                                ;;
                        ????r?????)
                                if [ "`group_is_in_group_name \"${groupid}\"`" -eq 1 ]
                                then
                                        stdio_message_warn "credentials" "${filename} is owned by user
                                else
                                        stdio_message_log "credentials" "${filename} is owned by user $
                                fi
                                ;;
                esac
        done
}
```

Figura 2.23: Búsqueda de permisos de ficheros

Con toda la información recabada es posible manualmente analizar cada fichero, investigar la vulnerabilidad y tratar de escalar privilegios, a modo de ejemplo se ha adjuntado la salida de la herramienta ejecutada sobre mi ordenador personal para poder visualizar las distintas partes del programa [8].

2.5.2 Privilege Escalation Awesome Scripts SUITE

PEASS es una herramienta que busca ficheros para realizar posibles escaladas de privilegios de manera local para sistemas Windows y Linux, además la salida se resalta con distintos colores para reconocer los problemas de configuración de una manera más sencilla.

Esta herramienta comprueba un checklist distinto en función de si se realiza sobre sistemas Windows [9] y Linux [10], actualmente se va a comentar cómo funciona la herramienta para sistemas Linux.

- Información básica

  En esta sección se analiza la información más básica del sistema y del usuario actual, tal y como la versión del sistema operativo, kernel, grupos al que pertenece con su PID, nombre del host y análisis de ejecutables críticos como ping, nc, nmap, etc.

- Información de sistema

  En esta sección se procede a buscar información detallada acerca del sistema como la versiones del sistema operativo, kernel y sudo. Además se detalla información acerca del *PATH*, fecha actual, ficheros de sistema, variables de entorno.

  Asimismo se investiga la información relacionada como si se encuentra presente grsecurity [11], si se encuentra habilitado ASLR, si el sistema se encuentra conectado con una impresora o si es un contenedor.

- Dispositivos montados bajo la ruta `/dev`.

  También se comprueban si existen sistemas de ficheros sin montar.

- Software disponible

  Programas interesantes instalados bajo la ruta `/usr/bin/` y compiladores que se encuentran en el sistema.

- Procesos, cron, servicios, temporizadores, sockets

  Se imprimen los procesos inesperados ejecutados por *root*, se comprueban los permisos que tiene los binarios del sistema controlados por *root*.

  Además la herramienta busca los procesos que se encuentran controlados por temporizadores como cron, realiza búsquedas sobre los ficheros `/etc/cron.d`, `/etc/cron.daily`, `/etc/cron.hourly`, `/etc/cron.monthly`, `/etc/cron.weekly` y `/tmp/crontab.iwzfvl`.

  También se buscan servicios con versiones obsoletas. Asimismo se comprueba el *PATH* utilizado por systemd y se analizan los ficheros con extensión *.service* para comprobar si tiene permisos de escritura o ejecutan rutas relativas.

[8] https://gist.github.com/KevinLiebergen/48d1d68ebe9c0ef35b28da93d8651efe
[9] https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/blob/master/winPEAS/winPEASexe/README.md
[10] https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/blob/master/linPEAS/README.md
[11] https://grsecurity.net/

La herramienta también analiza y enumera los Timers, alternativa a cron, ficheros con terminación *.timer* que controla eventos o daemons.

Igualmente analiza ficheros con extensión `.socket`, HTTP sockets y ficheros de configuración D-Bus.

- Información de red

  Por otro lado, muestra información acerca del nombre del host, hostname y DNS, imprime por pantalla redes vecinas, tablas de enrutamiento, reglas firewall, muestra los puertos activos y comprueba si puede sniffear tráfico con tcpdump.

- Información de usuarios

  Información de usuario actual (`id`), comprobación de llaves PGP, clipboards, /etc/sudoers sin contraseña, políticas Pkexec, lista usuarios con consola, todos los usuarios y grupos posibles, últimos inicios de sesión y políticas de contraseña.

- Información de software

  En este grupo se buscan las base de datos (MySQL, PostgreSQL), servidores Web, PHPCookies, ficheros de CMS (WordPRess, Drupal), Tomcat, búsqueda de ficheros `.ovpn` para conexiones VPN, ficheros ssh (Claves públicas y privadas, known_hosts, ssh config) y certificados.

  Asimismo se buscan credenciales Cloud (AWS, Azure, GCP), ficheros de configuración Kibana, configuración de port knocking (`Knock.config`), ficheros logstash, elasticsearch, Vault-ssh, AD cached, sesiones de pantalla, de terminal (tmux) y ficheros redis (`redis.conf`), dovecot, mosquitto, nei4j, Cloud-Init y Erlang cookie.

- Ficheros interesantes

  Para finalizar se analizan posibles ficheros interesantes, por ejemplo ficheros que contengan el bit SUID y/o SGID activado, directorios con permisos de escritura en `/etc/ld.so.conf.d/`, con privilegios adicionales (Capabilities), ficheros con listas de control de acceso, con extensión `.sh` en directorios que se incluyan en la variable de entorno `PATH`, ficheros no usuales en `/root`, scripts en `/etc/profile.d/`, permisos de lectura de los ficheros bajo `/etc/shadow o /root/`, si existen usuarios con hashes en `/etc/passwd o /etc/group` y credenciales en `/etc/fstab`.

  Búsqueda de ficheros que pertenecen a *root* en `/home`, también con permisos de lectura al usuario actual por medio de grupo pero no a todos los usuarios, ficheros interesantes modificados en los últimos cinco minutos, ficheros log con permisos de escritura, ficheros de correo electrónico, backups, extracción de tablas de bases de datos, ficheros dentro de `/var/www/`, historiales con permisos de lectura, ficheros `.git, .profile, .sudo_as_admin_successful, .bashrc, httpd.conf, .plan, .htpasswd, .gitconfig, .git-credentials, .git, .svn, .rhosts, hosts.equiv, Dockerfile, docker-compose.yml`, búsqueda de contraseñas dentro de `.bash_history`, papelera de reciclaje y bajo `.zsh_history`.

  Búsqueda de ficheros ocultos (salvo en `/sys`), ficheros con permisos de lectura en `/tmp, /var/tmp, /var/backups`, ficheros interesantes con permisos de escritura siendo el dueño el propio usuario o con permisos de escritura por cualquier usuario (incluso por las cuentas de servicio).

  Ficheros interesantes con permisos de escritura por grupo que no se encuentran en `/home`, búsqueda de contraseñas en ficheros PHP, IPs dentro de logs, búsqueda de correos electrónicos y contraseñas dentro de logs, búsqueda de las palabras *password* o *credential* en los títulos de ficheros en `/home`.

Para completar los puntos de la herramienta PEASS se realiza una búsqueda de las palabras *pwd*, *passw*, *username* y de posibles variables de contraseñas dentro de ficheros bajo `/home`, `/var/www`, `/var/backups`, `/tmp`, `/etc`, `/root` y `/mnt`.

Al igual que con la herramienta *pentestmonkey*, es necesario analizar cada fichero, que no es una tarea trivial, e ir buscando ficheros o servicios críticos con los que será posible escalar privilegios. Asimismo también se ha adjuntado para una mejor comprensión del programa la salida de la herramienta ejecutada sobre mi ordenador personal, de esta manera se conseguirá visualizar de mejor manera las distintas partes del programa [12].

[12] https://gist.github.com/KevinLiebergen/6b86a1caacbb57248353bac3fe31592b

Capítulo 3

Descripción experimental

Science is about knowing;
engineering is about doing.

Henry Petroski

3.1 Introducción

Las dos herramientas desarrolladas se encargan de comprobar que el sistema cumple cada una de las técnicas mencionadas anteriormente basadas en la base global MITTRE ATT&CK v6.3 en función del sistema operativo sobre el que se realiza.

El programa analiza cada una de las técnicas mencionadas, comprueba los permisos de los ficheros, en función de si cumplen o no ciertas técnicas se mostrará un mensaje de error, en rojo, o de notificación, en verde si el sistema se encuentra securizado para esa técnica.

3.2 Arquitectura

Como se ha comentado se han desarrollado dos programas, uno para cada sistema operativo, ciertas fracciones de código se han implementado en shell para Linux, y PowerShell en Windows debido a sus facilidades para:

- Leer ficheros de configuración
- Administrar ficheros
- Reaundar, para y crear procesos
- Ejecutar aplicaciones
- Crear y eliminar directorios
- Gestión de recursos de red
- Redireccionar / Pipelines

Sin embargo existen otras consideraciones ajenas a la gestión del sistema operativo como el procesamiento y la iteración sobre distintos objetos, por ello el desarrollo de la herramienta se ha visto limitado por la escasa flexibilidad que ofrece la programación en shell y PowerShell, el manejo de errores y la falta de librerías, debido a estas características la herramienta se ha implementado con el lenguaje de programación Python en su versión 3 [1].

A continuación se van a enumerar las bibliotecas de Python utilizadas para poder desarrollar la herrramienta.

- os

  Esta biblioteca proporciona de una manera sencilla la capacidad de comunicarse con el sistema operativo, ideal para realizar búsquedas, consultar estados de los servicios y demás acciones que requieren de una interacción con el sistema operativo.

- subprocess

  La biblioteca subprocess es parecida a os pero añade más funcionalidades como crear nuevos procesos, conectar y redirigir salidas y poder guardar las salidas.

- urllib

  Urrlib es una biblioteca que agrupa diversos módulos para trabajar con URLs y peticiones HTTP/S, ideal para comprobar si el sistema tiene activado un servidor web en la técnica Web Shell.

Para el paradigma de programación se ha descartado la programación orientada a objetos debido a que no es necesario la creación ni instanciación de diversos objetos ni tampoco cambiar el estado de las clases, por lo tanto se ha desarrollado siguiendo el paradigma de programación funcional por la necesariedad de un programa modular, para que en el futuro permita la adaptabilidad a nuevas versiones y sea posible añadir distintas tácticas.

Además se ha centrado en desarrollar el programa de una manera declarativa, aportando un nivel de abstracción alto sobre el programa principal (`main.py`) que importa las distintas técnicas mencionadas en el capítulo anterior que se alojan en el directorio `techniques/`, de esta manera la programación declarativa posibilita el mantenimiento independientemente sobre las futuras implementaciones de la aplicación y permite modularizar cada apartado sobre distintos ficheros.

Como se ha comentado la base de la herramienta se encuentra desarrollada en python y existen ciertas partes en PowerShell si la herramienta es para Windows o en Shell si se quiere analizar sistemas Linux. En la figura 3.1 se observa el fichero principal `main.py` interactúa con la carpeta `techniques/` donde según la técnica actúa de una manera u otra, las herramientas pueden producir varias salidas siendo éstas la creación de un fichero de texto, la salida por la terminal, o que se guarde en una base de datos para posteriormente conectarlo a un servidor web y crear una página web.

[1] https://www.python.org/download/releases/3.0

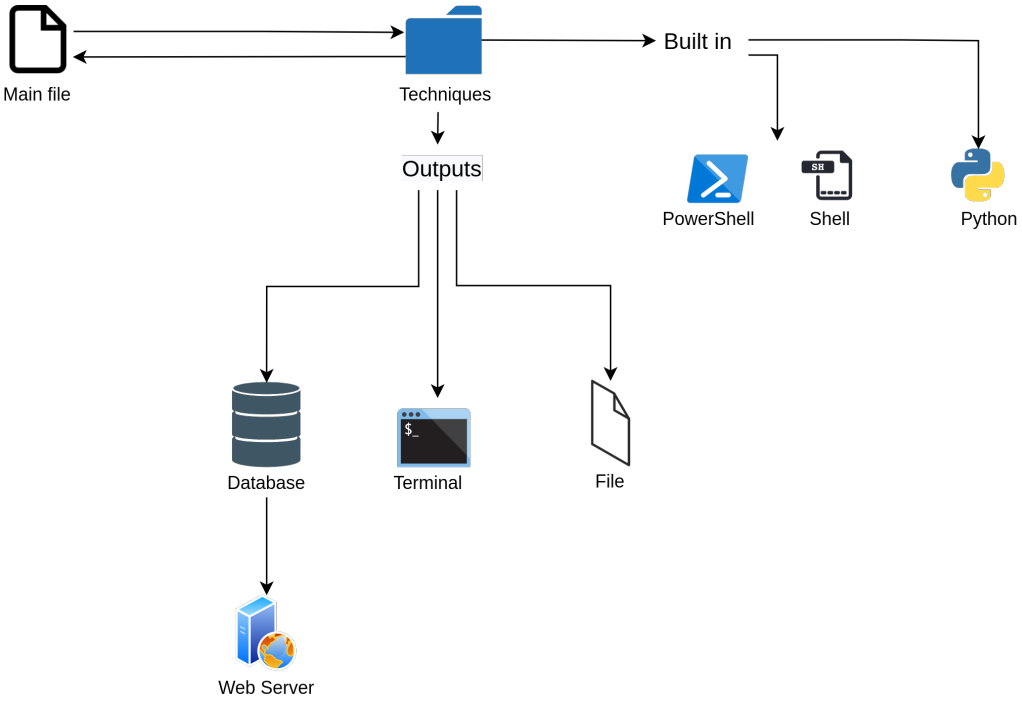

Figura 3.1: Diseño del programa principal

3.3 Evaluación de la herramienta

Para probar la eficacia se ha ejecutado la herramienta desarrollada sobre una máquina en producción y sobre dos máquinas virtuales vulnerables, las imágenes de las máquinas virtuales se han descargado de la página oficial de VulnHub [2]. Las imagenes que se han seleccionado han sido NullByte [3] y Raven [4], la máquina en producción se tratará de un servidor denominado ProTego JBCA.

Por todo esto, es necesario adquirir un conocimiento profundo sobre el funcionamiento de los sistemas operativos y de las técnicas del framework a estudiar para poder extraer conclusiones experimentales.

[2] https://www.vulnhub.com/
[3] https://www.vulnhub.com/entry/nullbyte-1,126/
[4] https://www.vulnhub.com/entry/raven-1,256/

Capítulo 4

Herramienta desarrollada

The power of Open Source is the power of the people.
The people rule

Philippe Kahn

4.1 Introducción

Como se ha comentado, se va a aunar la información estudiada anteriormente para poder crear una metodología que notifique cuando un sistema no se encuentre debidamente configurado, y con ello como mitigar dicho punto de entrada, tanto para sistemas Windows como para sistemas Linux.

4.2 Windows

El fichero principal, `main.py` se encarga de importar y ejecutar las veintitrés técnicas descritas anteriormente, se ha diseñado el programa para extraer la salida por la terminal de su vulnerabilidad con su mitigación.

4.2.1 Access Token Manipulation

En la figura 4.1 se hace referencia a las herramientas que es necesario fijar para cerciorar de que un sistema no pueda duplicar tokens de acceso que puedan desembocar en escaladas de privilegios no controlables.

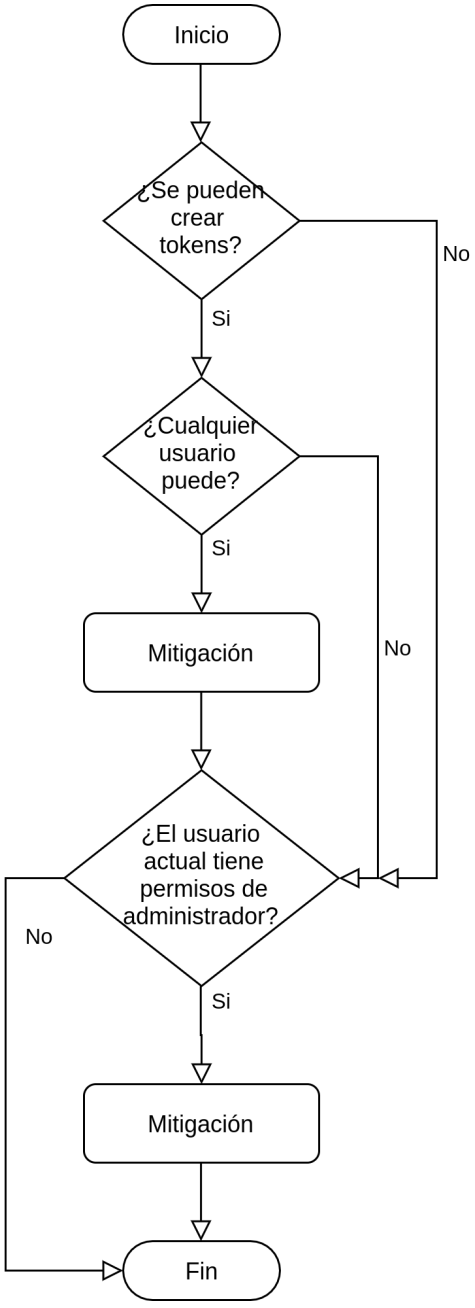

Figura 4.1: Control de flujo de la técnica Access Token Manipulation

4.2.2 Accesibility Features

Como se explicó anteriormente, los ejecutables `setch.exe` y `utilman.exe` son puntos vulnerables cuando se inicia una aplicación por lo que es necesario atender a tres grandes cuestiones que aparecen en la figura 4.2 para conocer si el equipo se encuentra protegido frente a esta técnica, si el sistema utiliza alguna herramienta de whitelist, si administra conexiones RDP y si utiliza autenticación a nivel de red.

Figura 4.2: Control de flujo de la técnica Accessibility Features

4.2.3 AppCert DLLs

Es necesario protegernos frente a librerías especificadas bajo AppCertDLLs realizando una búsqueda de software tal y como establece el diagrama de flujo 4.3.

Figura 4.3: Control de flujo de la técnica Appcert DLLs

4.2.4 AppInit DLLs

Similar a la técnica anterior en esta caso es necesario buscar bajo otra clave de registro. Esta funcionalidad está deshabilitada en las nuevas versiones de Windows con ajustes determinados por lo que es necesario comprobar como se muestra en 4.4.

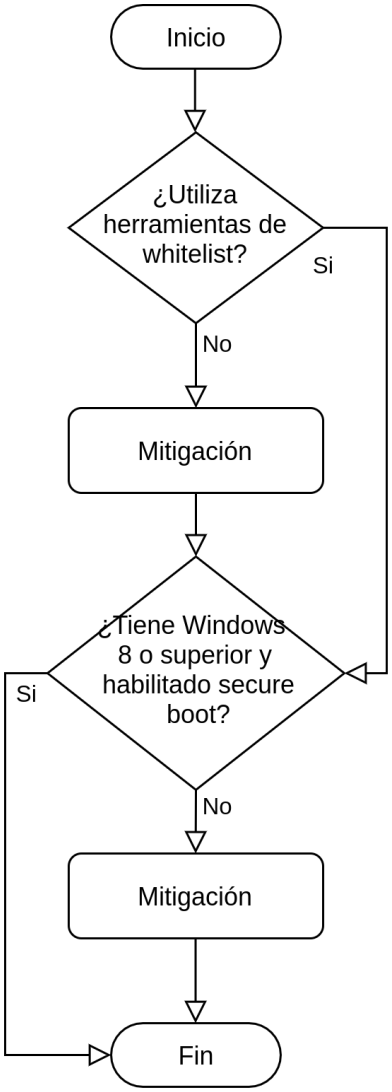

Figura 4.4: Control de flujo de la técnica AppInit DLLs

4.2.5 Application Shiming

Se conoce que la compatibilidad que ofrece Application Shiming ofrece una vulnerabilidad si se encuentra mal configurada en el sistema por lo que el fichero revisa dos aspectos importantes como muestra la figura 4.5 para que no modifiquen el kernel y se ejecuten como Administrador.

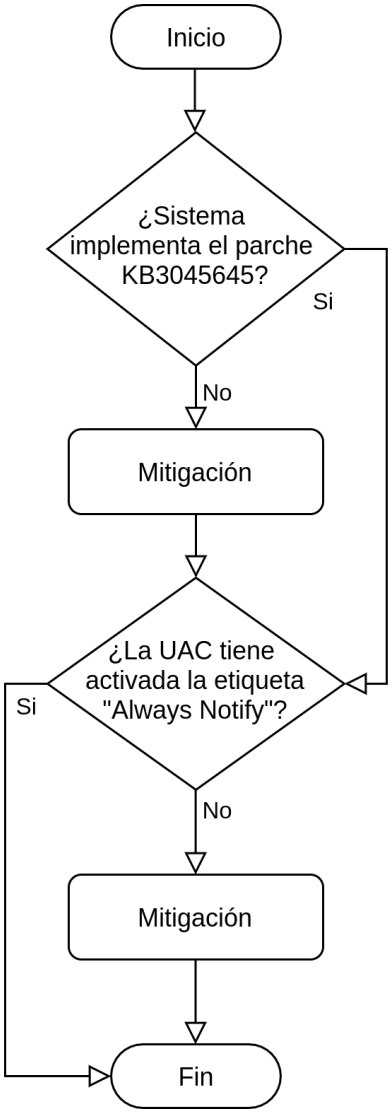

Figura 4.5: Control de flujo de la técnica Application Shimming

4.2.6 Bypass User Account Control

Existen múltiples exploits públicos para bypassear UAC por lo que para esta técnica se ejecutará uno de los repositorios más importantes [1] para poder listar las vulnerabilidades y seguidamente verificar las configuraciones de seguridad como se aprecia en las últimas acciones del diagrama de flujo 4.6. Existen otros métods para bypassear UAC basados en movimientos laterales que sus mitigaciones se cubren en otras técnicas.

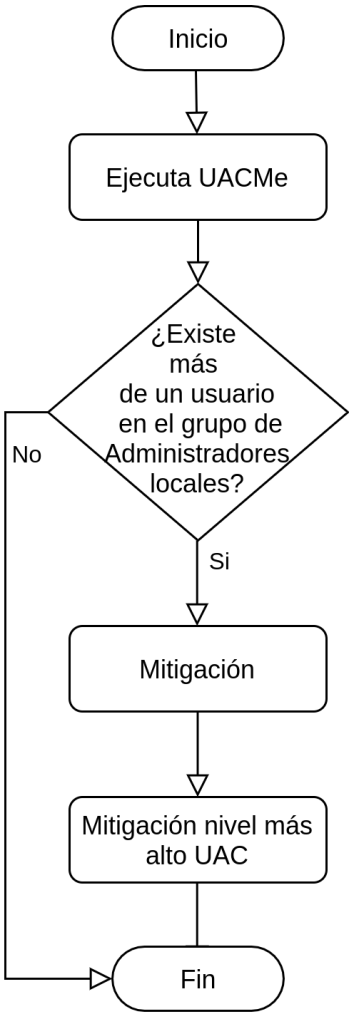

Figura 4.6: Control de flujo de la técnica Bypass User Account Control

[1] https://github.com/hfiref0x/UACME

4.2.7 DLL Search Order Hijacking

El secuestro de DLLs basados en cambios de orden es una de las técnicas más comunes para escalar privilegios, por ello es necesario seguir rigurosamente las medidas para no reemplazar el orden de búsqueda, esto se hace comprobando la configuración del sistema como aparece en el diagrama 4.7

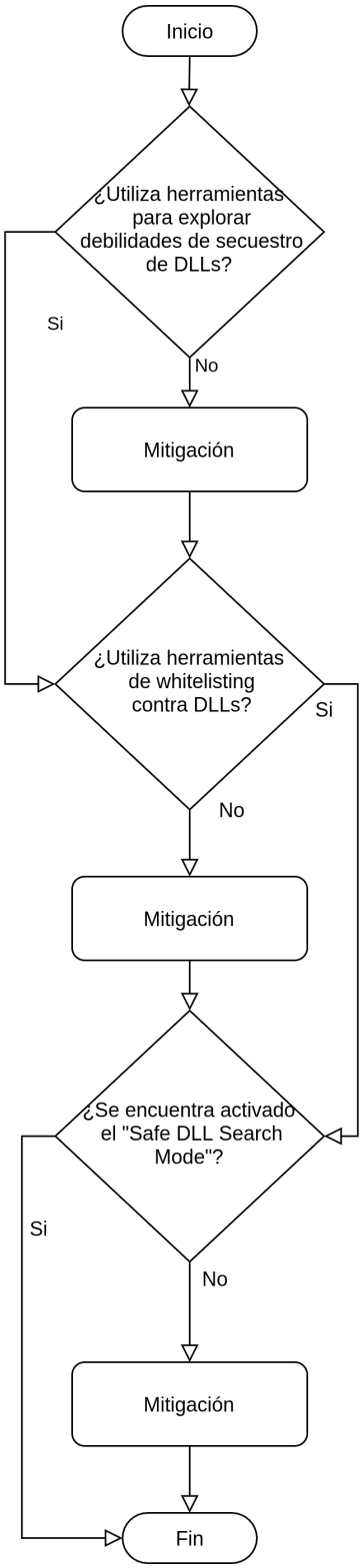

Figura 4.7: Control de flujo de la técnica DLL Search Order Hijacking

4.2.8 Exploitation for Privilege Escalation

El diagrama de flujo de la técnica *Exploitation for Privilege Escalation* funciona de la siguiente manera: en primer lugar ejecuta una herramienta que sugiere si existen vulnerabilidades en el sistema atendiendo ala versión del sistema operativo y kernel e indaga los CVEs públicos que exisen en bases de datos públicas.

Como aparece en 4.8 Seguidamente se comprueba si existen programas actualizables, si existen programas se lanza un mensaje de notificación que aparece qué realizar al respecto. Posteriormente se comprueba si el sistema se encuentra dockerizado o virtualizado y se concluye con un mensaje que aconseje la utilización de un programa de amenazas, el personal que ejecute la herramienta debe conocer si ya se encuentra instalado algún programa de amenazas.

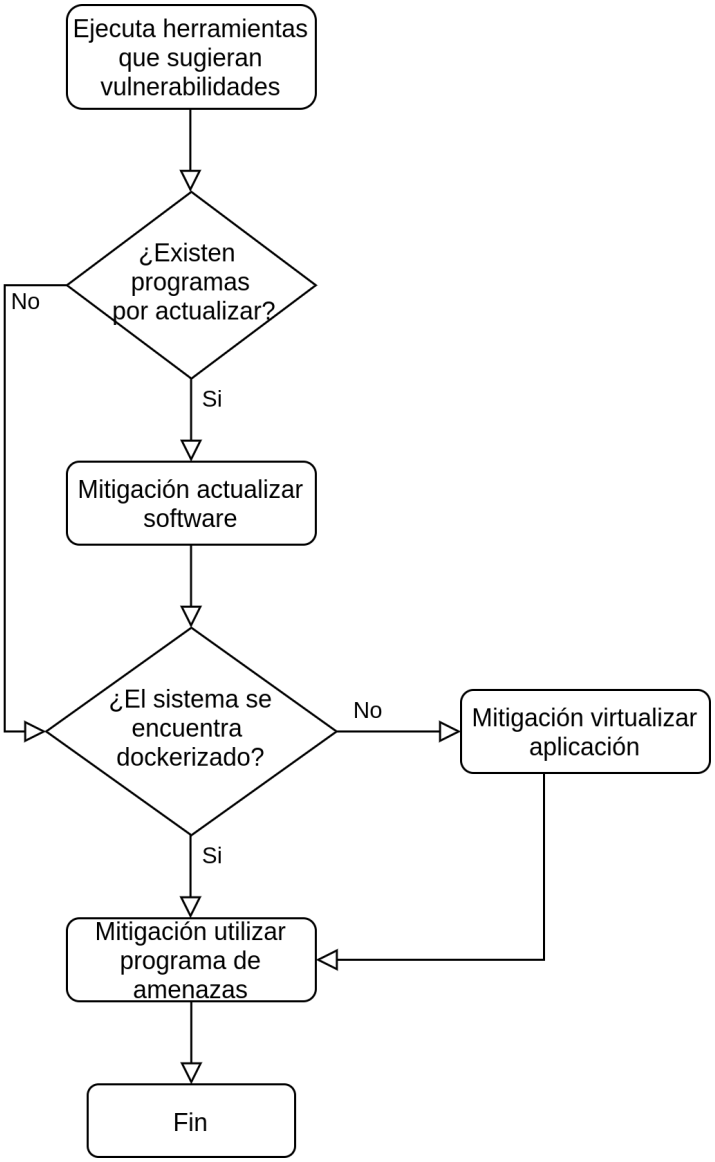

Figura 4.8: Control de flujo de la técnica Exploitation for Privilege Escalation

4.2.9 File System Permissions Weakness

En esta técnica se procede a buscar herramientas como el framework PowerSploit para mitigar debilidades en los permisos de los ficheros y directorios y a buscar fallos de configuración de los permisos del UAC y de los usuarios como aparece en 4.9.

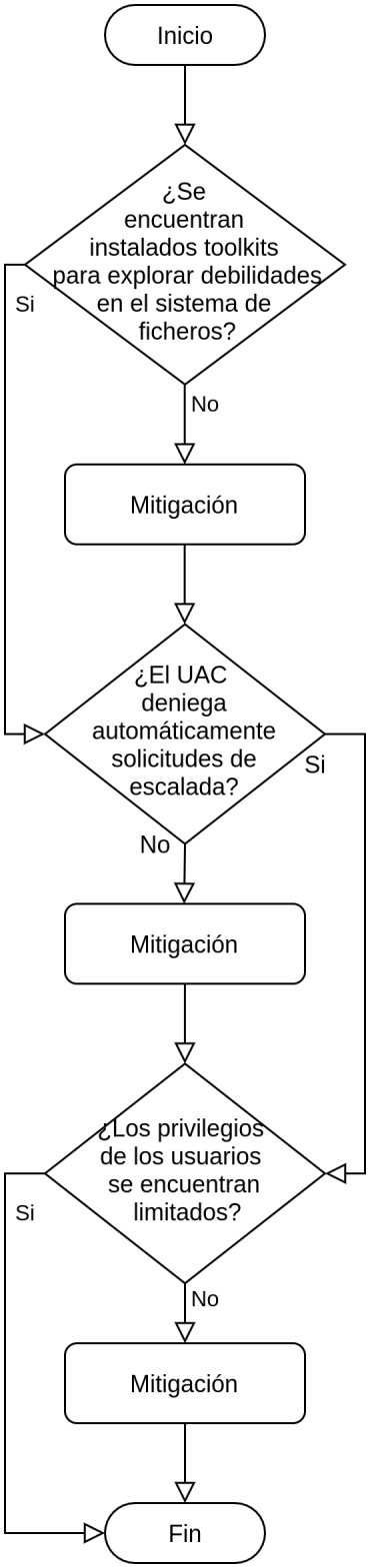

Figura 4.9: Control de flujo de la técnica File System Permissions Weakness

4.2.10 New Service

La creación de nuevos servicios para ejecutarse en el arranque del sistema se deben de realizar con privilegios de administrador para evitar añadir software malicioso, se intentará la creación de nuevos servicios y se buscarán herramientas como Sysinternals para detectar este tipo de amenaza con su posterior mitigación como en 4.10.

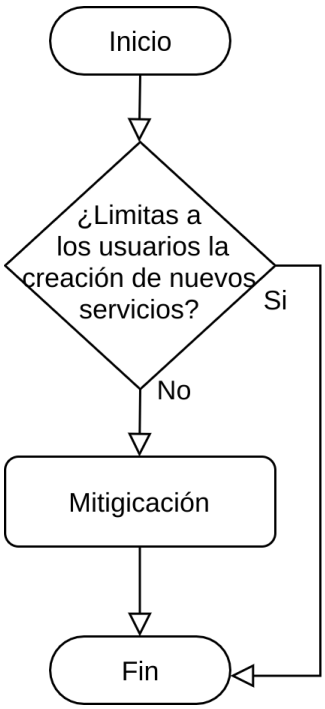

Figura 4.10: Control de flujo de la técnica New Service

4.2.11 Path Interception

Para mitigar la interceptación y reemplazo de rutas debemos de atender a los tipos de Path Interception que existen y analizar cuidadosamente cómo mitigarlo, para ello nos debemos de basar en herramientas externas que analicen las rutas de los ficheros de configuración, utilizar whitelists y asegurar de que los usuarios tienen los permisos correctos y no crean ficheros en cualquier ruta.

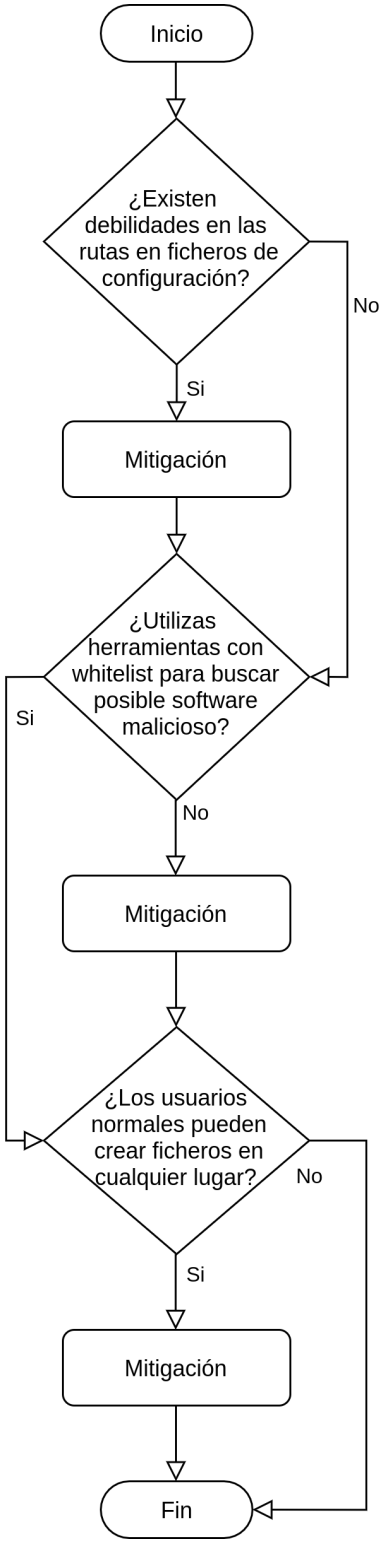

Figura 4.11: Control de flujo de la técnica Path Interception

4.2.12 PowerShell Profile

En la figura 4.12 se revisan los perfiles PowerShell que traen por defecto en distintas rutas como `$PsHome\Profile.ps1`, se revisan las firmas en los perfiles.

Por último independientemente de si se cumplen las condiciones anteriores se recomienda evitar los perfiles PowerShell y se busca si los scripts contienen la etiqueta `-No Profile` para evitar que se ejecuten scripts.

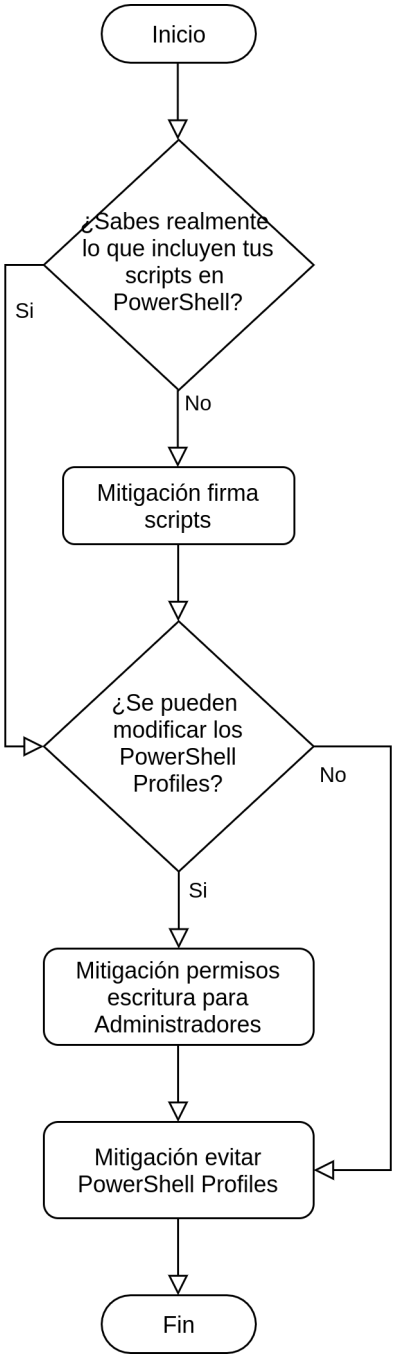

Figura 4.12: Control de flujo de la técnica PowerShell Profile

4.2.13 Process Injection

Una de las técnicas que más se han aprovechado los ciberdelincuentes, es necesario controlar los endpoints como aparece en 4.13 para evitar inyecciones DLLs, secuestro de hilos y llamadas a procedimientos asíncronos entre otros. Es necesario monitorizar llamadas de la API de Windows como `WriteProcessMemory` y variables de entorno como `LD_PRELOAD`.

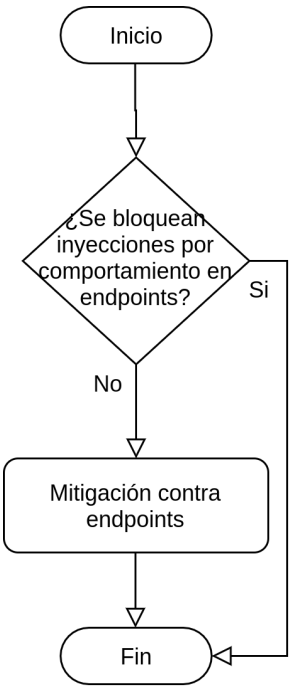

Figura 4.13: Control de flujo de la técnica Process Injection

4.2.14 Scheduled Task

Es necesario analizar en Scheduled task la forma en la que se crean las tareas programas, en el diagrama 4.14 se examina si el sistema contienen toolkits como PowerSploit debido a que implementan módulos que explora debilidades en tareas programadas, configurar el registro de clave en `HKLM\SYSTEM\CurrentControlSet\Control\Lsa\SubmitControl` para ejecutar tareas bajo el permiso del usuario en lugar de `SYSTEM` y configurar el `Group Policy Objects` adecuadamente para aumentar la seguridad del sistema.

Por último se lanza un mensaje de notificación para recordar limitar los privilegios de las cuentas de usuarios al crear tareas programadas.

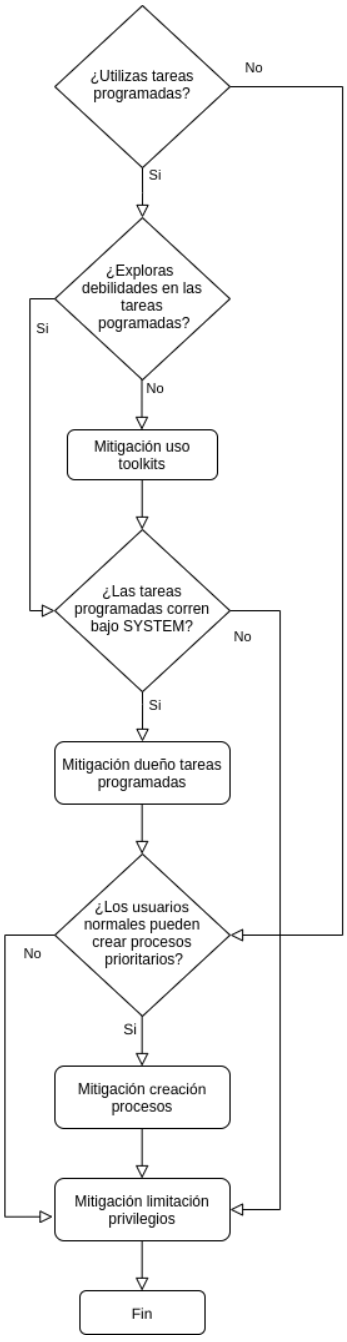

Figura 4.14: Control de flujo de la técnica Scheduled Task

4.2.15 Service Registry Permissions Weakness

Es necesario analizar los permisos de escritura bajo `HKLM\SYSTEM\CurrentControlSet\Services` para que no se manipulen los servicios como se demuestra en 4.15.

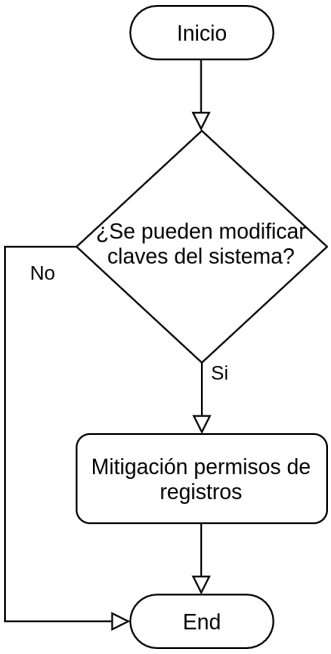

Figura 4.15: Control de flujo de la técnica Service Registry Permissions Weakness

4.2.16 SID-History Injection

La configuración del Active Directory permite controlar y filtrar el identificador de Windows para mitigar la escalada de privilegios mediante la inserción del `SID-History` por lo que es necesario configurar el Active Directory.

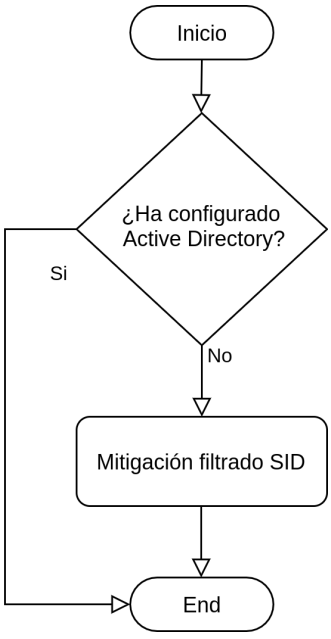

Figura 4.16: Control de flujo de la técnica SID-History Injection

4.2.17 Valid Accounts

La técnica Valid Accounts, que engloba el robo de credenciales se encarga de realizar una búsqueda de posibles contraseñas sobre los ficheros críticos e imprimirlos por pantalla. Si existen una posible contraseña se imprimirá una notificación e independendientemente delresultado anterior procederá a buscar posibles claves ssh (*.pub,*.ssh,*id_rsa) en el sistema. En función del resultado anterior se imprimirá o no un mensaje de cómo mitigar y prevenir esta vulnerabilidad según el diagrama 4.17.

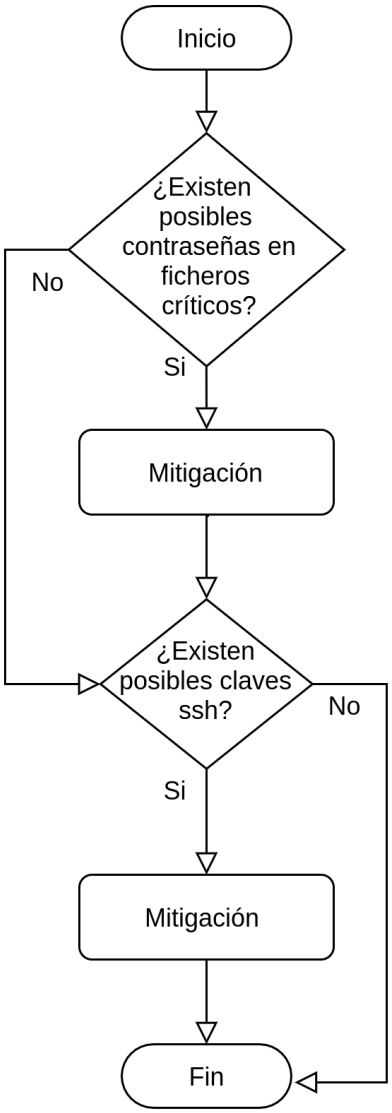

Figura 4.17: Control de flujo de la técnica Valid Accounts

4.2.18 Web Shell

El diagrama de flujo de la técnica Web Shell 4.18 se enfoca desde la comprobación de la existencia de un servidor web, y en caso afirmativo escanear y analizar la información de este, listar los usuarios que permiten abrir una shell y con ello su mitigación correspondiente.

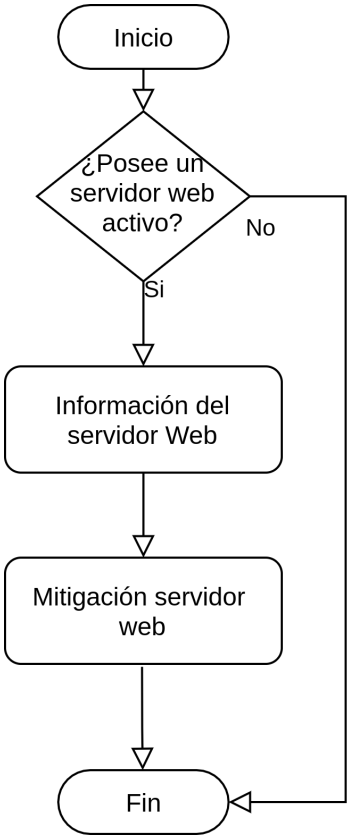

Figura 4.18: Control de flujo de la técnica Web Shell

4.2.19 Técnicas no mitigables

Estas técnicas de a continuación no tienen establecidas una mitigación concreta debido a su elevada complejidad por lo que la única solución es una monitorización regular sobre determinados servicios.

4.2.19.1 Extra Window Memory Injection

En este caso para mitigar la técnica Extra Window Memory Injection es necesario monitorizar las llamadas de la API de Windows `GetWindowLong`, `SetWindowLong` y `SendNotifyMessage` regularmente.

4.2.19.2 Hooking

La manera para mitigar Hooking es realizar un monitoreo regular sobre las llamadas `SetWindowsHookEx` y `EventHook`, verificar la integridad de procesos realizando análisis estáticos y dinámicos y analizar procesos que realizan conexiones sospechsas al exterior.

4.2.19.3 Image File Execution Options

Monitorización de procesos que se encuentren depurando bajo las etiquetas `DEBUG_PROCESS` y `DEBUG_ONLY_THIS_PROCESS`, o llamadas que tengan indicativos de creación de registros como `RegCreateKeyEx` `RegSetValueEx`.

4.2.19.4 Parent PID Spoofing

Monitoriza regularmente las llamadas `CreateProcess` / `CreateProcessA` y rastrea el flujo mediante Event Tracing for Windows (ETW).

4.2.19.5 Port Monitors

Es necesario monitorizar regularmente llamadas API, ejecutar Autoruns y monitorizar registros `HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors` para cerciorarse de que en el sistema no se encuentra comprometido.

4.3 Linux

El fichero `main.py` importa las siete técnicas descritas anteriormente en el orden indicado en la imagen 4.19, cuyo diagrama de flujo se expondrá a continuación. Se ha desarrollado la herramienta generando como output la salida por terminal, siendo la estructura del proyecto la que aparece en la figura 4.20.

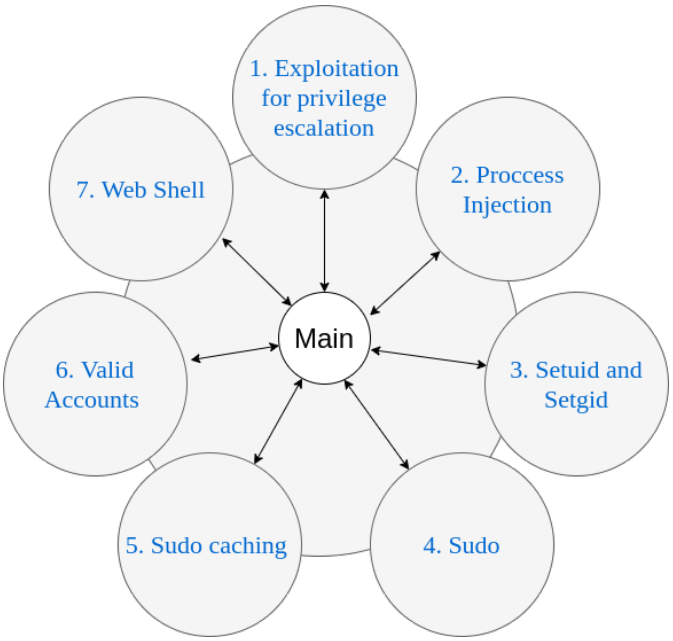

Figura 4.19: Interconexión del main con las distintas técnicas

```
Escalada-privilegios-tfg/project/lin on  master
→ tree -I __pycache__
.
├── main.py
└── techniques
    ├── config
    │   ├── constants.py
    │   ├── sudoers_permissions.py
    │   └── utilities.py
    ├── exploitation.py
    ├── process_injection.py
    ├── setuid.py
    ├── sudo_caching.py
    ├── sudo.py
    ├── valid_accounts.py
    └── web_shell.py

2 directories, 11 files
```

Figura 4.20: Estructura de la herramienta desarrollada en Linux

4.3.1 Exploitation for Privilege Escalation

En la figura 4.21 se puede observar el diagrama de flujo de la técnica *Exploitation for Privilege Escalation* que funciona de la siguiente manera: en primer lugar ejecuta el script Linux Exploit Suggester [2] y Linux Exploit Suggester 2 [3] para analizar la versión del sistema y kernel e indaga los CVEs públicos que exisen en bases de datos públicas.

Seguidamente se ejecuta el comando `$ apt list --upgradable` para ver si existen programas actualizables, se compara desde el último `$ sudo apt-get update` que realizó el usuario legítimo, si existen programas se lanza un mensaje de notificación que aparece qué realizar al respecto. Posteriormente se comprueba si el sistema se encuentra dockerizado o virtualizado y se concluye con un mensaje que aconseje la utilización de un programa de amenazas.

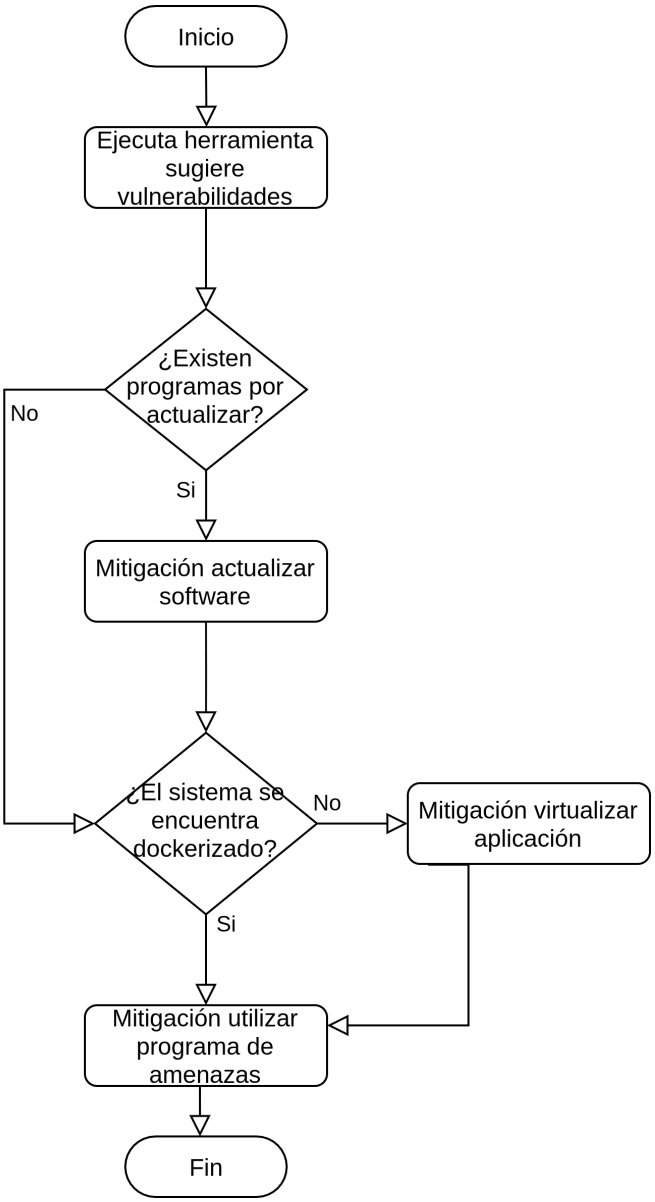

Figura 4.21: Control de flujo de la técnica Exploitation for privilege escalation

[2] https://github.com/mzet-/linux-exploit-suggester
[3] https://github.com/jondonas/linux-exploit-suggester-2

4.3.2 Process Injection

Para la técnica *Process Injection* se verifica y recomienda tener protección frente a ptrace y endpoints debido a la alta peligrosidad que ocasiona tenerlo mal configurado como se comentó en Process Injection. En caso de que `ptrace` se encuentre mal configurado imprimirá un mensaje del problema con su correspondiente mitigación tal y como indica la figura 4.22.

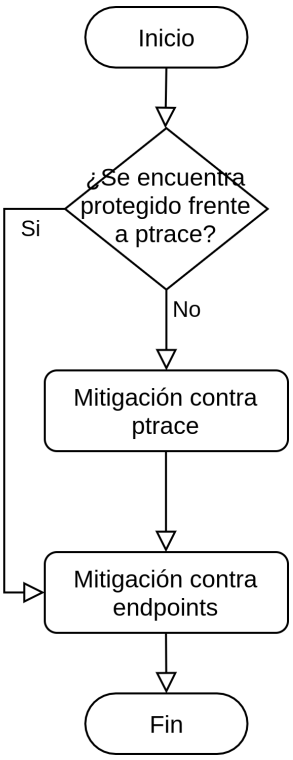

Figura 4.22: Control de flujo de la técnica Process Injection

4.3.3 Setuid and Setgid

Para la técnica Setuid and Setgid realiza el diagrama de flujo de la imagen 4.23, se ha procedido a realizar tres tipos de búsqueda o `find`, para archivos con el bit setuid activado, buscando archivos con la máscara `-??s??????` siendo `?` cualquier permiso y siendo *root* el dueño del fichero, mediante el comando:

```
$ find / -user root -perm /4000 -exec ls -l {} \; 2>/dev/null
```

ficheros con el bit setgid activado, buscando archivos con la máscara `-?????s???` siendo `?` cualquier permiso y siendo *root* el dueño del grupo del fichero, mediante el comando:

```
$ find / -user root -type f -perm /2000 -exec ls -l {} \; 2>/dev/null
```

y directorios con el bit setgid activado, buscando archivos con la máscara `d?????s???` siendo `?` cualquier permiso y siendo *root* el dueño del grupo del fichero, mediante el comando:

```
$ find / -user root -type d -perm /2000 -exec ls -ld {} \; 2>/dev/null
```

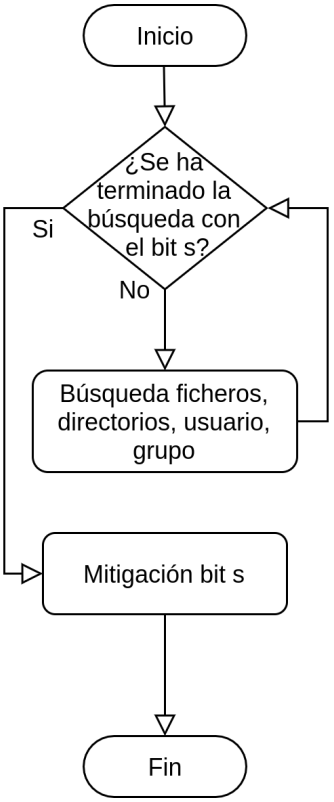

Figura 4.23: Control de flujo de la técnica setuid

4.3.4 Sudo

El fichero *sudo.py* se basa en la técnica Sudo, en la imagen 4.24 se observa que se analiza la técnica de la siguiente manera, se analiza los permisos del fichero `/etc/sudoers`, si contiene permisos de lectura procede a imprimir por pantalla dicho y fichero y comienza a analizar si contiene fallos de configuración, esto es, comprueba si existen procesos que no requieran la inserción de contraseñas para ejecutar como administrador, esto se consigue como se explicó mediante el uso de la etiqueta `NOPASSWD` activada. En caso afirmativo se procede a imprimir por pantalla el fallo y su mitigación asociada.

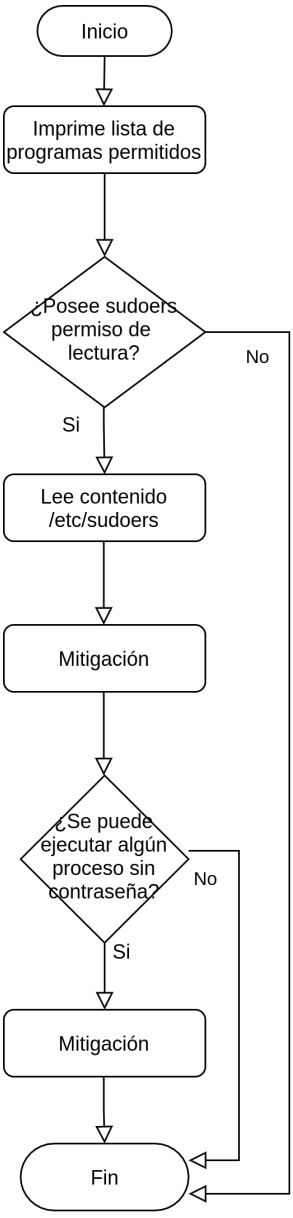

Figura 4.24: Control de flujo de la técnica sudo

4.3.5 Sudo Caching

La siguiente técnica se encuentra en una estrecha relación con la anterior técnica y, en la nueva versión de MITRE ATT&CK se han aunado ambas. Sudo caching contempla si dentro del fichero `/etc/sudoers` se encuentran deshabilitados los `!tty_tickets` que ocasiona que sea posible abrir una nueva terminal e introducir comandos como superusuario si en otra terminal ya se escribió la contraseña. Si se encuentra deshabilitado lanza un mensaje sobre el peligro y cómo mitigarlo y comprueba si el `timestamp` o el tiempo que debe de transcurrir para que vuelva a preguntar la contraseña se encuentre a 0. En caso de no ser así lanzará otro mensaje acerca de como mitigar dicha configuración tal y como se muestra en la figura 4.25.

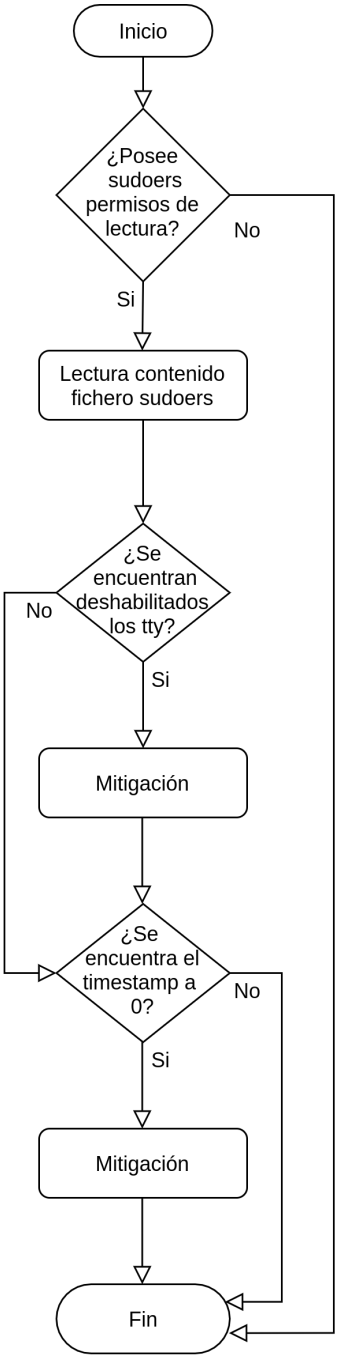

Figura 4.25: Control de flujo de la técnica sudo caching

4.3.6 Valid Accounts

La técnica Valid Accounts, que engloba el robo de credenciales se encarga de realizar una búsqueda de posibles contraseñas sobre los ficheros críticos `_history`, `.sudo_as_admin_successful`, `.profile`, `.bashrc`, `httpd.conf`, `.plan`, `.htpasswd`, `.gitconfig`, `.git-credentials`, `.git`, `.svn`, `.rhost`, `hosts.equiv`, `Dockerfile`, `docker-compose.yml` e imprimirlos por pantalla. Si existen una posible contraseña se imprimirá una notificación e independendientemente del resultado anterior procederá a buscar posibles claves ssh (`*.pub`, `*.ssh`, `*id_rsa`) en el sistema. En función del resultado anterior se imprimirá o no un mensaje de cómo mitigar y prevenir esta vulnerabilidad como aparece en la figura 4.26.

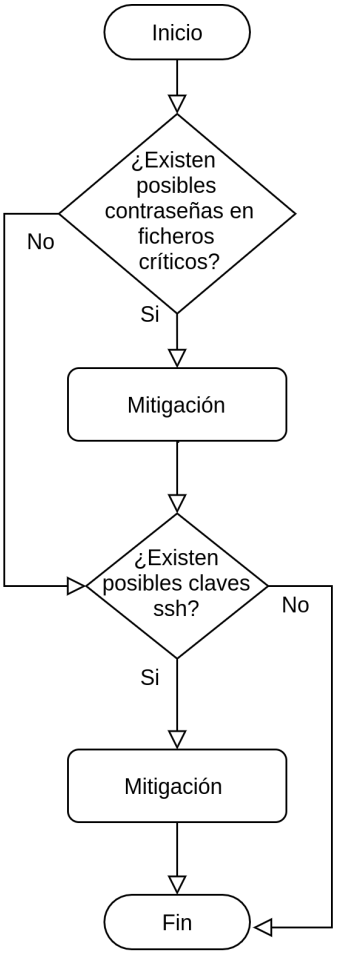

Figura 4.26: Control de flujo de la técnica Valid Accounts

4.3.7 Web Shell

Por último, el diagrama de flujo de la técnica *Web Shell* podemos observar en la figura 4.27 se enfoca desde la comprobación de la existencia del servidor web, y en caso afirmativo escanear y analizar la información de este, listar los usuarios que permiten abrir una shell y con ello la mitigación correspondiente.

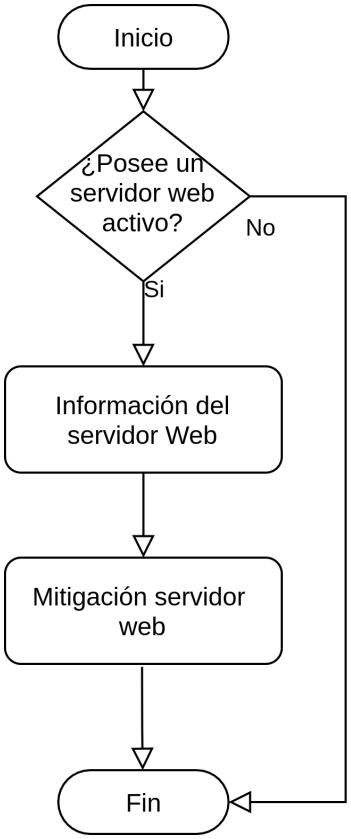

Figura 4.27: Control de flujo de la técnica Web Shell

Capítulo 5

Resultados

In theory, theory and practice are the same.
In practice, they are not.

Albert Einstein

5.1 Introducción

Para conseguir los resultados ha sido necesario estudiar el framework MITRE ATT&CK, comparar dichas técnicas con trabajos previos como PentestMonkey [1] y Privilege Escalation Awesome Scripts SUITE [2] para poder extraer las diferencias entre ellas, establecer un procedimiento e implementarlo para ejecutarlo sobre distintos sistemas y establecer resultados experimentales.

5.2 Resultados no experimentales

Analizando las herramientas y trabajos previos como se ha mencionado anteriormente se ha podido concluir que actualmente no existen herramientas comerciales eficaces, sino que predominan las heramientas Open Source. La carencia de herramientas comerciales puede ser debido a la falta de información, falta de conocimiento sobre escaladas de privilegios o del framework MITRE ATT&CK, o a la poca rentabilidad que es posible extraer debido a la eficacia de las herramientas Open Source.

Con ello se estudiaron qué técnicas de MITRE ATT&CK implementan las herramientas comunes y cuales no y gracias a ello se pudo recoger algunas carencias del framework. MITRE no contempla la búsqueda de determinados servicios en los sistemas Linux como son los `capabilities` [3], que nos permiten gestionar que permisos tiene un proceso para acceder a las partes del kernel independientemente del usuario que lo lance.

Igualmente MITRE ATT&CK tampoco incluye la búsqueda de ficheros críticos con permisos asignados erróneamente como son `/etc/passwd` o `/etc/shadow` que modificándolos permiten escalar privilegios.

También se encontrado carencias debido a la ausencia de búsqueda de servicios mediante cron que se ejecuten bajo permisos de *root* y de que la variable global `PATH` no se encuentre corrupta.

[1] https://github.com/pentestmonkey/unix-privesc-check
[2] https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite
[3] https://man7.org/linux/man-pages/man7/capabilities.7.html

Además se ha concluido por el número de técnicas y la complejidad de mitigar que el sistema operativo Windows por norma general es más vulnerable frente a ataques, sobre todo por malware y pentestings.

5.3 Resultados del framework

Se ha desarrollado un framework que cumple con los requisitos planeados para evitar una escalada de privilegios ilegítima, notificando al usuario los puntos vulnerables del sistema, el diseño incluye las técnicas y cubre las tecnologías óptimas de implementación para cada plataforma.

A la hora de desarrollar dicha herramienta en Windows se ha demostrado por su complejidad que no ha sido posible implementar mitigaciones para las técnicas Extra Window Memory Injection, Hooking, Image File Execution Options Injection, Parent PID SPoofing, Port Monitors.

Por ello actualmente la manera de mantener sin vulnerabilidades estos sistemas Windows es un análisis manual de forma periódica por parte de un operario para detectar las técnicas no mitigables o evaluables después de ejecutar la herramienta para detectar dichos fallos de seguridad. Es decir se hace imprescindible que se realice protocolos de evaluación del sistema y evaluación de vulnerabilidades para

Por ello se hace imprescindible que para un equipo Blue Team se realicen protocolos de evaluación del sistema para detectar las técnicas no mitigables o evaluables para asegurar la seguridad operativa (análisis, evaluación, prevencion, mitigatión y control) del sistema y con ello de la empresa.

Para los sistemas Linux al encontrar mitigación para las siete técnicas descritas basta con ejecutar la herramienta para conocer que vulnerabilidades tiene el sistema sin la necesidad de un operario que realice de manera manual un análisis.

Windows 10 tiene la ejecución de scripts deshabilitados por defecto por lo que es una primera medida contra la ejecución de malwares en PowerShell y con ello un problema para orientar la herramienta hacia Red Team.

Asimismo es necesario analizar la salida de las herramientas para indagar sobre los posibles vectores de ataque dentro del sistema.

5.4 Resultados experimentales en máquina Linux

Para experimentar se ha ejecutado las herramientas desarrolladas sobre distintas máquinas para comprobar la eficacia de estas, se han descargado dos máquinas virtuales en formato `.ova` y se han importado en VirtualBox, los resultados han sido los siguientes:

5.4.1 Máquina virtual NullByte

La máquina virtual elegida para este ejemplo se denomina NullByte y está disponible en la página oficial Vulnhub [4]. Esta imagen es de tipo *boot2root* en el que es necesario acceder al sistema y conseguir los privilegios de administrador. Esta sección se centrará únicamente en la escalada de privilegios y cómo protegernos frente a esto.

Gracias a una vulnerabilidad ha sido posible introducirse en el sistema, y como se muestra en la figura 5.1 no es posible ejecutar servicios de root.

[4] https://www.vulnhub.com/entry/nullbyte-1,126/

```
ramses@NullByte:~$ whoami
ramses
ramses@NullByte:~$ sudo cat /etc/shadow
[sudo] password for ramses:
ramses is not in the sudoers file.  This incident will be reported.
ramses@NullByte:~$
```

Figura 5.1: ramses no posee permisos de root

Una vez esto, se transfiere la herramienta creada al sistema a auditar y se ejecuta, al analizar la salida se observa un ejecutable curioso denominado `/var/www/backup/procwatch` con el sticky key activado, como se muestra en la figura 5.2

```
######################( Searching files with bit setuid activated )########

-rwsr-xr-x 1 root root 562536 Mar 23  2015 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 13796 Nov 28  2014 /usr/lib/policykit-1/polkit-agent
-rwsr-xr-x 1 root root 5372 Feb 25  2014 /usr/lib/eject/dmcrypt-get-device
-rwsr-xr-x 1 root root 9540 Apr 15  2015 /usr/lib/pt_chown
-rwsr-xr-- 1 root messagebus 362672 May 28  2015 /usr/lib/dbus-1.0/dbus-dae
er
-rwsr-sr-x 1 root mail 96192 Feb 12  2015 /usr/bin/procmail
-rwsr-xr-x 1 root root 52344 Nov 20  2014 /usr/bin/chfn
-rwsr-xr-x 1 root root 38740 Nov 20  2014 /usr/bin/newgrp
-rwsr-xr-x 1 root root 43576 Nov 20  2014 /usr/bin/chsh
-rwsr-xr-x 1 root root 78072 Nov 20  2014 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 18064 Nov 28  2014 /usr/bin/pkexec
-rwsr-xr-x 1 root root 53112 Nov 20  2014 /usr/bin/passwd
-rwsr-xr-x 1 root root 176400 Mar 12  2015 /usr/bin/sudo
-rwsr-xr-x 1 root root 1081076 Feb 18  2015 /usr/sbin/exim4
-rwsr-xr-x 1 root root 4932 Aug  2  2015 /var/www/backup/procwatch
-rwsr-xr-x 1 root root 38868 Nov 20  2014 /bin/su
-rwsr-xr-x 1 root root 34684 Mar 30  2015 /bin/mount
-rwsr-xr-x 1 root root 26344 Mar 30  2015 /bin/umount
-rwsr-xr-x 1 root root 96760 Aug 13  2014 /sbin/mount.nfs
```

Figura 5.2: Procwatch tiene bit setuid activado

Al ejecutar el binario se observa en la figura 5.3 que realiza el comando `ps` bajo permisos de administrador.

```
ramses@NullByte:/var/www/backup$ ls -la
total 20
drwxrwxrwx 2 root root 4096 Sep 28 00:40 .
drwxr-xr-x 4 root root 4096 Aug  2  2015 ..
-rwsr-xr-x 1 root root 4932 Aug  2  2015 procwatch
-rw-r--r-- 1 root root   28 Aug  2  2015 readme.txt
ramses@NullByte:/var/www/backup$ ./procwatch
  PID TTY          TIME CMD
 1408 pts/0    00:00:00 procwatch
 1409 pts/0    00:00:00 sh
 1410 pts/0    00:00:00 ps
ramses@NullByte:/var/www/backup$
```

Figura 5.3: Binario procwatch ejecuta ps como root

Esto se considera una vulnerabilidad debido a que si se modifica el binario `ps` por `sh` abrirá una shell

con permisos de *root*, por lo que se procede a sustituir el binario mediante un link simbólico como aparece en la figura 5.4 mediante el comando `ln -s`.

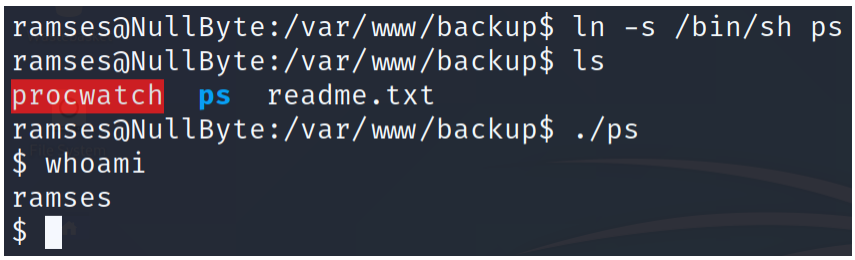

Figura 5.4: Link simbólico de ps a sh

Se modifica el orden de búsqueda de los binarios alterando la variable global `PATH` en 5.5 y se ejecuta el binario procwatch, en la figura 5.6 se ha abierto una shell con permisos de *root*.

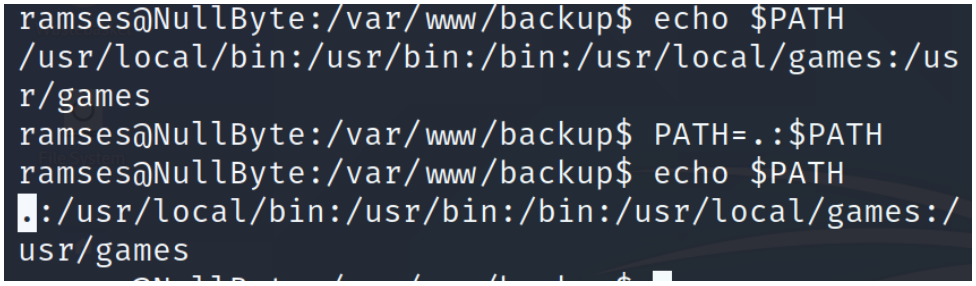

Figura 5.5: Variable global PATH modificada

Figura 5.6: Procwatch ejecuta una shell con permisos de root

Así es como un ciberdelincuente puede comprometer un sistema debido a una vulnerabilidad de este tipo, por lo que por parte de un equipo Blue Team es necesario mitigar este punto crítico. En este caso es necesario controlar el bit setuid y si es innecesario eliminarlo.

Además es importante comentar que el framework MITRE ATT&CK no contempla los permisos de la variable global `PATH`, la escalada de privilegios ha podido ser efectiva debido a que el sistema ha buscado el ejecutable ps en el directorio donde nos encontramos antes de en otros. Si se hubiera podido denegar la escritura sobre la variable `PATH` no hubiera tenido éxito dicha escalada.

5.4.2 Máquina virtual Raven

Para la máquina virtual Raven [5] se procede a realizar la misma metodología que el punto anterior, como se observa en la figura 5.7 el sistema tiene una aplicación web. Debido a una vulnerabilidad se consigue introducir en el sistema aunque sin permisos de *root*, debido a que no podemos leer el fichero `/etc/shadow`, figura 5.8.

[5] https://www.vulnhub.com/entry/raven-1,256/

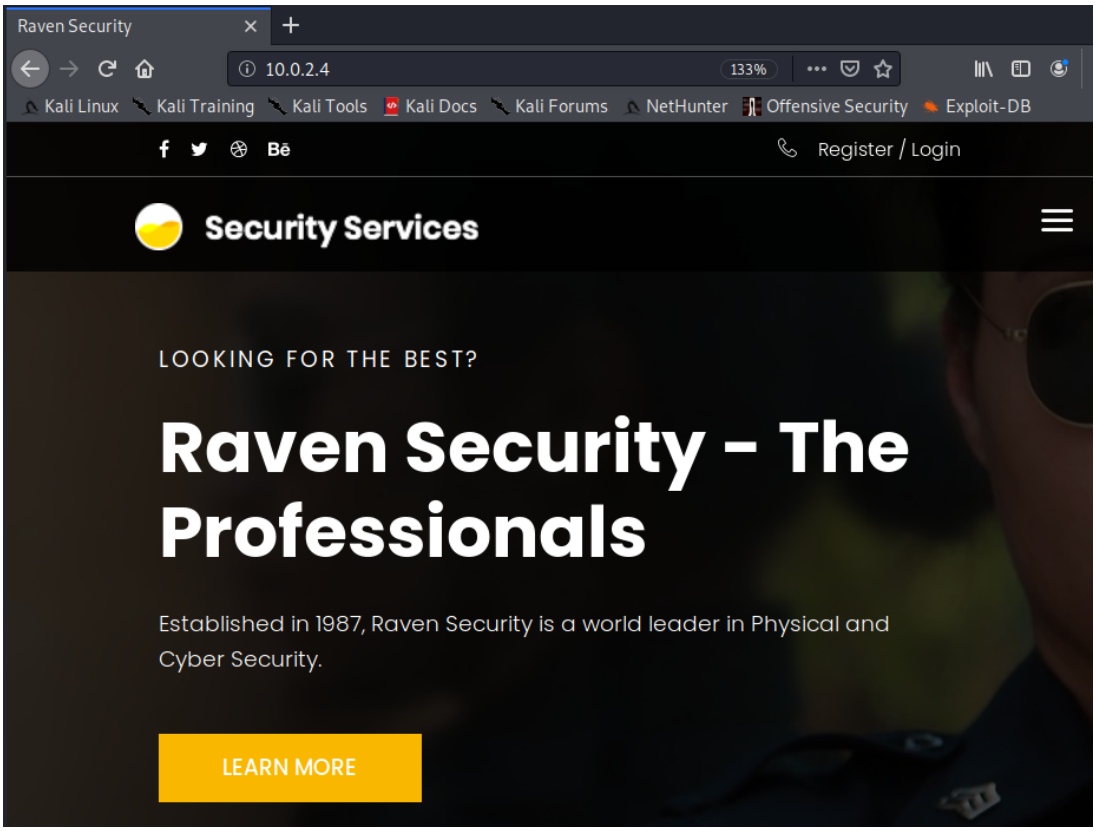

Figura 5.7: Página principal del Web Server Raven

```
$ whoami
steven
$ sudo cat /etc/shadow
[sudo] password for steven:
Sorry, user steven is not allowed to execute '/bin/cat /etc/shadow' as root
$
```

Figura 5.8: Steven no posee permisos de root

Lanzando la herramienta desarrollada se observa en la técnica Sudo, que Python se puede ejecutar sin introducir la contraseña en el sistema, según la imagen 5.9. Por lo que se procede a abrir la shell de Python mediante *sudo*, como python permite ejecutar comandos del sistema es posible realizarlos mediante permisos de administrador.

```
##########################( Searching /etc/sudoers permissions )############

-r--r----- 1 root root 713 Aug 13  2018 /etc/sudoers


Matching Defaults entries for steven on raven:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/u

User steven may run the following commands on raven:
    (ALL) NOPASSWD: /usr/bin/python
```

Figura 5.9: Python se puede ejecutar como superusuario sin insertar contraseña

```
steven@Raven:/tmp/lin$ sudo python
Python 2.7.9 (default, Jun 29 2016, 13:08:31)
[GCC 4.9.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>>
>>> os.system("whoami")
root
0
>>> os.system("cat /etc/shadow")
root:$6$rFGuQUz8$02awL8e4/jdcf3NSYRv/7pDY.gmiLLspy5j/LhVuCNb0IjGUU22Tyf
999:7:::
```

Figura 5.10: Desde Python se puede ejecutar comandos como superusuario

Por parte de un equipo Blue Team el ejecutable Python es un punto crítico y una amenaza para el sistema, es necesario que se pida la contraseña se solicite siempre para que los usuarios no puedan ejecutar procesos con privilegios de `root`.

5.4.3 Servidor producción ProTego JBCA

A continuación hemos ejecutado la herramienta sobre un servidor en producción con ubuntu 18.04, de todos los resultados se ha creído necesario comentar los que aparecen en las figuras 5.11 y 5.12.

Para la primera imagen, la figura 5.11 se observan unos CVEs para el sistema actual, de los cuales el primero, `CVE-2018-18955` tiene un grado de exposición alto, al lado de cada CVE aparecen detalles y un enlace hacia su exploit.

```
Possible Exploits:

[+] [CVE-2018-18955] subuid_shell

   Details: https://bugs.chromium.org/p/project-zero/issues/detail?id=1712
   Exposure: probable
   Tags: [ ubuntu=18.04 ]{kernel:4.15.0-20-generic},fedora=28{kernel:4.16.3-301.fc28}
   Download URL: https://github.com/offensive-security/exploitdb-bin-sploits/raw/mast
   Comments: CONFIG_USER_NS needs to be enabled

[+] [CVE-2019-18634] sudo pwfeedback

   Details: https://dylankatz.com/Analysis-of-CVE-2019-18634/
   Exposure: less probable
   Tags: mint=19
   Download URL: https://github.com/saleemrashid/sudo-cve-2019-18634/raw/master/explo
   Comments: sudo configuration requires pwfeedback to be enabled.

[+] [CVE-2019-15666] XFRM_UAF
```

Figura 5.11: Exploits posibles que nos arroja la herramienta sobre un servidor en producción

Para la segunda imagen, la figura 5.12 podemos observar que para la técnica *Valid Accounts*, que incluye la búsqueda de posibles contraseñas en ficheros ha encontrado la contraseña de la base de datos a la que esta conectada, y unos posibles tokens de una API que es necesario analizar.

Desde el punto de vista de un operario que ejecute la herramienta es necesario seguir las mitigaciones que se aconseja así como actualizar el sistema, los programas y guardar contraseñas en ficheros cifrados.

a traves de la conexion a la bbdd pueda acceder a un usuario con mas privilegios, aunqeu no sea root

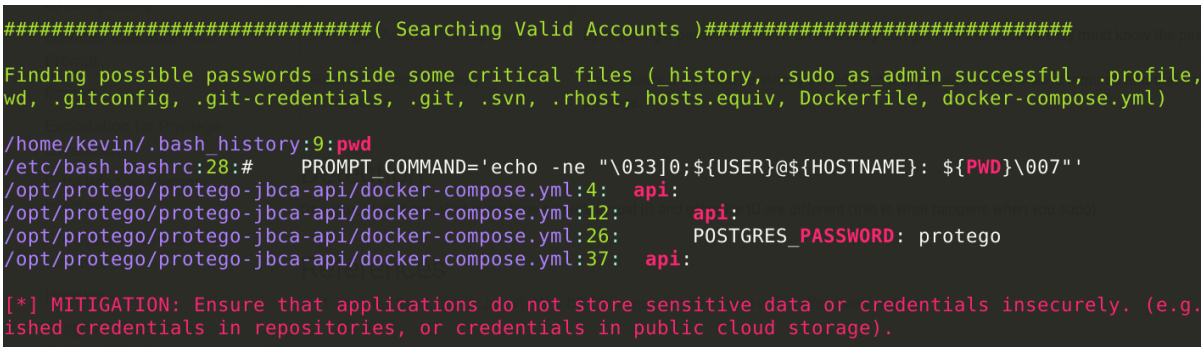

Figura 5.12: Posibles contraseñas dentro de ficheros

Capítulo 6

Conclusiones y trabajo futuro

Engineers like to solve problems.
If there are no problems handily available,
they will create their own problems.

Scott Adams

6.1 Conclusiones

Con este trabajo se ha resuelto el objetivo establecido, elaborar un framework basado en MITRE ATT&CK orientado a la escalada de privilegios,

Se ha concluido además que el framework posee carencias frente a otras herramientas Open Source por lo comentado anteriormente y viceversa, que personalmente sería bueno que se incluyesen en versiones futuras. Para la salida de las herramienta cabe destacar que es necesario distinguir algunas técnicas analizando la viabilidad para aplicar dicha mitigación o no, debido a que algunas soluciones no sean triviales y aunque a priori no se permita si se indaga puede suponer escalar privilegios.

Además, se ha concluido que por la dificultad de mitigar las técnicas Windows es un sistema operativo más vulnerable.

Asimismo se ha afirmado la idea de aunar ciertas técnicas con grandes similitudes en una sola, y esto se ha visto reflejada en la nueva versión v7.0.

6.2 Trabajo futuro

Del desarrollo de este framework y la creación de la herramienta se han generado varias ideas a desarrollar en un futuro:

- Actualizar a versión actual

  Durante el desarrollo del trabajo se encontraba bajo desarrollo la versión 7.0, que ya ha sido publicada de manera oficial, sería conveniente analizar qué cambios existen entre ambas versiones y poder actualizar el framework.

- Añadir distintas tácticas

Asimismo, es interesante poder añadir diversas tácticas así como la persistencia para poder abarcar un mayor nivel de seguridad a la hora de recibir una intrusión.

- Añadir aspectos que no cubre MITRE

  Como se ha comentado, las herramientas Open Source no abarcan todas las técnicas del framework MITRE ATT&CK, sin embargo, MITRE ATT&CK tampoco abarca todos los métodos de intrusión que indagan las herramientas Open Source. Por ello se puede crear una metodología que amplíe los distintos puntos de intrusión hacia los que abarca dichas herramientas Open Source.

- Implementar distintos outputs

  Por ejemplo el salvaguardado de la salida por un fichero o en una base de datos para poder crear una aplicación web y mostrar los resultados en él.

- Orientar la herramienta a Red Teaming

  También es interesante poder orientar la herramienta para equipos Red Team, para ello sería necesario migrar las herramientas y que sean escritas únicamente en Shell para sistemas Linux y en PowerShell para sistemas Windows, dado que son lenguajes nativos que incluyen dichos sistemas operativos.

Bibliografía

[1] Sina Karvandi, "Import Address Table (IAT) in action," Apr. 2017. [Online]. Available: https://rayanfam.com/topics/import-address-table-in-action/

[2] Microsoft Team, "Understanding Shims," Dec. 2012. [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-7/dd837644(v%3dws.10)

[3] D. Montemayor, "Windows Local Accounts," Feb. 2019. [Online]. Available: https://docs.microsoft.com/en-us/windows/security/identity-protection/access-control/local-accounts

[4] D. Halfin, "Active Directory Accounts," Aug. 2019. [Online]. Available: https://docs.microsoft.com/en-us/windows/security/identity-protection/access-control/active-directory-accounts

[5] Daniel Simpson, "Service Accounts," Apr. 2017. [Online]. Available: https://docs.microsoft.com/es-es/windows/security/identity-protection/access-control/service-accounts

[6] Polynomial, "System Account in Windows," Sep. 2014. [Online]. Available: https://security.stackexchange.com/questions/66743/system-account-in-windows

[7] H. Johnston, "windows - root vs Administrator vs SYSTEM," Nov. 2012. [Online]. Available: https://superuser.com/questions/504136/root-vs-administrator-vs-system

[8] "init - ArchWiki." [Online]. Available: https://wiki.archlinux.org/index.php/Init

[9] "Privilege Escalation Tactics." [Online]. Available: https://attack.mitre.org/tactics/TA0004/

[10] Contributors, "Retrocompatibilidad," Jul. 2019. [Online]. Available: https://es.wikipedia.org/w/index.php?title=Retrocompatibilidad&oldid=117384701

[11] SecRat, "API Hooking," Apr. 2014, library Catalog: resources.infosecinstitute.com Section: Hacking. [Online]. Available: https://resources.infosecinstitute.com/api-hooking/

[12] Ashkan Hosseini, "Ten process injection techniques," Jul. 2017. [Online]. Available: https://www.elastic.co/blog/ten-process-injection-techniques-technical-survey-common-and-trending-process

[13] "Windows 7 Release Candidate auto-elevate white list," Nov. 2014. [Online]. Available: https://web.archive.org/web/20141127035941/http://www.withinwindows.com/2009/05/02/short-windows-7-release-candidate-auto-elevate-white-list/

[14] hfiref0x, "UACME," Jun. 2020. [Online]. Available: https://github.com/hfiref0x/UACME

[15] M. Satran, "Dynamic-Link Library Redirection - Win32 apps," May 2018. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/dlls/dynamic-link-library-redirection

[16] Contributors, "PowerSploit," library Catalog: github.com. [Online]. Available: https://github.com/PowerShellMafia/PowerSploit

[17] D. Batchelor, "Dynamic-Link Library Search Order - Win32 apps," May 2018, library Catalog: docs.microsoft.com. [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/dlls/dynamic-link-library-search-order

[18] Arris Huijgen, "Windows Exploit Suggester Next Generation," Jun. 2020, original-date: 2019-03-02T15:25:40Z. [Online]. Available: https://github.com/bitsadmin/wesng

[19] R. Mouse, "Sherlock," May 2020, original-date: 2017-04-02T16:01:53Z. [Online]. Available: https://github.com/rasta-mouse/Sherlock

[20] Contributors, "Watson," May 2020, original-date: 2018-10-08T18:16:36Z. [Online]. Available: https://github.com/rasta-mouse/Watson

[21] Caledonia Project, "Windows kernel exploits," Jun. 2020, original-date: 2017-04-25T04:02:31Z. [Online]. Available: https://github.com/SecWiki/windows-kernel-exploits

[22] "Offensive Security's Exploit Database Archive," library Catalog: www.exploit-db.com. [Online]. Available: https://www.exploit-db.com/

[23] Microsoft Team, "Environment Property," Oct. 2011, library Catalog: docs.microsoft.com. [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/fd7hxfdd(v=vs.85)

[24] Sean Wheeler, "PowerShell Profiles," Nov. 2017, library Catalog: docs.microsoft.com. [Online]. Available: https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_profiles

[25] Joe Desimone, "Hunting In Memory," Jun. 2019, library Catalog: www.elastic.co. [Online]. Available: https://www.elastic.co/blog/hunting-memory

[26] Hod Gavriel, Boris Erbesfeld, "'Early Bird' Code Injection," Apr. 2018, library Catalog: www.cyberbit.com. [Online]. Available: /blog/endpoint-security/new-early-bird-code-injection-technique-discovered/

[27] Contributors, "Well-known security identifiers in Windows operating systems." [Online]. Available: https://support.microsoft.com/en-us/help/243330/well-known-security-identifiers-in-windows-operating-systems

[28] O. Moe, "Persistence using GlobalFlags in Image File Execution Options," Apr. 2018. [Online]. Available: https://oddvar.moe/2018/04/10/persistence-using-globalflags-in-image-file-execution-options-hidden-from-autoruns-exe/

[29] J. Donas, "Linux exploit suggester 2," Jun. 2020. [Online]. Available: https://github.com/jondonas/linux-exploit-suggester-2

[30] mzet, "Linux exploit suggester," Jun. 2020. [Online]. Available: https://github.com/mzet-/linux-exploit-suggester

[31] G. 75, "Inspector," May 2020. [Online]. Available: https://github.com/graniet/Inspector

[32] Contributors, "SecWiki/linux-kernel-exploits," Jun. 2020. [Online]. Available: https://github.com/SecWiki/linux-kernel-exploits

[33] "vdso." [Online]. Available: https://man7.org/linux/man-pages/man7/vdso.7.html

[34] A. Serper, "Proton.B: What this Mac malware actually does," library Catalog: www.cybereason.com. [Online]. Available: https://www.cybereason.com/blog/labs-proton-b-what-this-mac-malware-actually-does

[35] Tim Rains, "Microsoft's Free Security Tools – banned.h," Aug. 2012. [Online]. Available: https://www.microsoft.com/security/blog/2012/08/30/microsofts-free-security-tools-banned-h/

[36] Steve Grubb, "Security Assessment Tools." [Online]. Available: http://people.redhat.com/sgrubb/security/

[37] Huzaifa Sidhpurwala, "Hardening ELF binaries using Relocation Read-Only (RELRO)." [Online]. Available: https://www.redhat.com/en/blog/hardening-elf-binaries-using-relocation-read-only-relro

Apéndice A

Técnicas según Sistema Operativo

Las técnicas de escalada de privilegios desglosadas según Sistema Operativo son:

	Windows	Linux	macOS
1	Access Token Manipulation	Exploitation for privilege escalation	Dylib Hijacking
2	Accesibility Features	Process Injection	Elevated Execution with Prompt
3	AppCert DLLs	Setuid and Setgid	Emond (Event Monitor Daemond)
4	AppInit DLLs	Sudo	Exploitation for privilege escalation
5	Application Shimming	Sudo caching	Launch Daemon
6	Bypass User Account Control	Valid Accounts	Plist modification
7	DLL Search Order Hijacking	Web Shell	Process Injection
8	Exploitation for privilege escalation		Setuid and Setgid
9	Extra Window Memory Injection		Startup Items
10	File System Permissions Weakness		Sudo
11	Hooking		Sudo caching
12	Image File Executions Options Injection		Valid Accounts
13	New Service		Web Shell
14	Parent PID Spoofing		
15	Path Interception		
16	Port monitor		
17	PowerShelll Profile		
18	Process Injection		
19	Scheduled Task		
20	Service Registry Permissions Weakness		
21	SID-History Injection		
22	Valid Accounts		
23	Web Shell		

Apéndice B

Procesos que se ejecutan con privilegios sin mostrar UAC

La lista de los procesos que elevan privilegios sin mostrar un aviso de la UAC para Windows 7 son:

- Windows\ehomeM\cx2Prov.exe
- Windows\System32\AdapterTroubleshooter.exe
- Windows\System32\BitLockerWizardElev.exe
- Windows\System32\bthudtask.exe
- Windows\System32\chkntfs.exe
- Windows\System32\cleanmgr.exe
- Windows\System32\cliconfg.exe
- Windows\System32\CompMgmtLauncher.exe
- Windows\System32\ComputerDefaults.exe
- Windows\System32\dccw.exe
- Windows\System32\dcomcnfg.exe
- Windows\System32\DeviceEject.exe
- Windows\System32\DeviceProperties.exe
- Windows\System32\dfrgui.exe
- Windows\System32\djoin.exe
- Windows\System32\eudcedit.exe
- Windows\System32\eventvwr.exe
- Windows\System32\FXSUNATD.exe
- Windows\System32\hdwwiz.exe

- Windows\System32\ieUnatt.exe
- Windows\System32\iscsicli.exe
- Windows\System32\iscsicpl.exe
- Windows\System32\lpksetup.exe
- Windows\System32\MdSched.exe
- Windows\System32\msconfig.exe
- Windows\System32\msdt.exe
- Windows\System32\msra.exe
- Windows\System32\MultiDigiMon.exe
- Windows\System32\Netplwiz.exe
- Windows\System32\newdev.exe
- Windows\System32\ntprint.exe
- Windows\System32\ocsetup.exe
- Windows\System32\odbcad32.exe
- Windows\System32\OptionalFeatures.exe
- Windows\System32\perfmon.exe
- Windows\System32\printui.exe
- Windows\System32\rdpshell.exe
- Windows\System32\recdisc.exe
- Windows\System32\rrinstaller.exe
- Windows\System32\rstrui.exe
- Windows\System32\sdbinst.exe
- Windows\System32\sdclt.exe
- Windows\System32\shrpubw.exe
- Windows\System32\slui.exe
- Windows\System32\SndVol.exe
- Windows\System32\spinstall.exe
- Windows\System32\SystemPropertiesAdvanced.exe
- Windows\System32\SystemPropertiesComputerName.exe
- Windows\System32\SystemPropertiesDataExecutionPrevention.exe
- Windows\System32\SystemPropertiesHardware.exe

- Windows\System32\SystemPropertiesPerformance.exe
- Windows\System32\SystemPropertiesProtection.exe
- Windows\System32\SystemPropertiesRemote.exe
- Windows\System32\taskmgr.exe
- Windows\System32\tcmsetup.exe
- Windows\System32\TpmInit.exe
- Windows\System32\verifier.exe
- Windows\System32\wisptis.exe
- Windows\System32\wusa.exe
- Windows\System32\DriverStoreFileRepository\bth.inf_x86_neutral_65c949576945c2a9\fsquirt.exe
- Windows\System32\oobe\setupsqm.exe
- Windows\System32\sysprep\sysprep.exe

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR

Universidad
de Alcalá