HACKEANDO COMPUTADORES POR EL PUERTO USB
HACKING COMPUTERS BY USB PORT

DANIEL CARRERA SAINZ
BRUNO TORRALBO FERNANDEZ

DEPARTAMENTO DE ARQUITECTURA DE COMPUTADORES Y AUTOMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo Fin de Grado en Ingeniería de Software

Septiembre 2022

Director:

Luis Piñuel Moreno

Resumen

Los ataques HID (Human Interface Device) consisten en conectar un pequeño sistema empotrado al puerto USB del computador víctima para que este lo identifique como un dispositivo USB válido y poder explotar sus vulnerabilidades por este medio. Ciñéndose a Raspberry Pi Zero W como sistema empotrado atacante, los objetivos de este trabajo consisten en estudiar el funcionamiento del puerto USB, así como la vulnerabilidad que posee y que hace posible que se pueda atacar con facilidad, las herramientas existentes para estos ataques (como P4wnP1 A.L.O.A, Ethsploiter, entre otros), ponerlas a prueba en diversos sistemas, para posteriormente ofrecer soluciones para evitar que puedan llegar a ejecutarse estas herramientas.

Para cada herramienta utilizada haremos un tutorial de instalación, una explicación de su funcionamiento, una pequeña demostración en diferentes entornos y una posible solución para hacer frente a las vulnerabilidades explotadas. Además, explicaremos los diferentes tipos de ataques HID, como montar cualquier imagen en una Raspberry Pi Zero, la definición de exploit y en qué se diferencia con una vulneravilidad y los distintos tipos de estos. Complementandolo con nuestra conclusión respecto a estos tipos de ataques muy presentes en el mundo real debido a su facilidad en realizarlos y a su dificultad en defenderse de ellos.

Palabras clave

USB, HID, Raspberry Pi, Exploit, Hash, P4wnP1, Thief, SSH, Kali, Vulnerabilidad

Keywords

USB, HID, Raspberry Pi, Exploit, Hash, P4wnP1, Thief, SSH, Kali, Vulnerability

Índice general

Agradecimientos

En primer lugar, nos gustaría dar las gracias a nuestros amigos y familiares por acompañarnos y apoyarnos a lo largo de estos años en la realización del Grado de Ingeniería de Software.

Además, de dar las gracias a nuestro director Luis Piñuel Moreno por apoyarnos, guiarnos y darnos la oportunidad de hacer posible la realización de este TFG ya que es un tema que nos llamaba mucho la atención a pesar de no ser de Sistemas y queríamos adentrarnos en el mundo de la ciberseguridad.

Capítulo 1

Introducción

En este trabajo investigaremos sobre los ataques HID. Para ello, haremos una breve introducción desde la historia del puerto USB hasta como funciona el estándar HID. Después, haremos enfoque en los diferentes tipos de ataques y mismo ataque en diferentes entornos, todos ellos usando una Raspberry Pi Zero W como herramienta atacante. Analizaremos más exhaustivamente los exploits desarrollados en los capítulos siguientes.

Este trabajo consta con parte de investigación sobre cómo funciona el puerto USB, los diversos ataques HID y las múltiples opciones para albergarlos.

Además de entender cómo funcionan estos exploits una vez es atacada la víctima en múltiples entornos, se proporcionarán potenciales soluciones a ellos, además de demostraciones de dichas soluciones empleadas y breves tutoriales de instalación para su uso.

1.1. Motivación

Lo que nos ha llevado a elegir este tema fue el gran desconocimiento que se tiene sobre este tipo de ataques, para comprender el grave peligro que supone tanto para un usuario medio como para una empresa, además de que queríamos iniciarnos en el mundo de la ciberseguridad y este proyecto ha sido la oportunidad para llevarlo a cabo. El peligro del que hablamos es algo tan común como que con una simple memoria flash, un dispositivo que na-

die pensaría que podría ocasionarnos graves problemas si lo conectásemos a nuestros equipos personales, hay personas que no se paran a pensar en ello y lo conectan por curiosidad de ver su contenido. Incluso estudiantes de informática, tal y como reflejó el experimento realizado en la biblioteca de la facultad de Informática, experimento al que hacemos referencia en la conclusión de este trabajo. Esto nos hace preguntarnos por qué algo que lleva incorporado en los ordenadores desde 1996, 26 años después sigue sin ser corregido.

1.2. Objetivos del trabajo

El objetivo de este Trabajo de Fin de Grado es *estudiar las herramientas existentes (P4wnP1 A.L.O.A, Responder,...) para atacar sistemas operativos*, ponerlas a prueba en diversos sistemas, para posteriormente ofrecer soluciones para evitar las vulnerabilidades que explotan. Para ello, partiremos de una Raspberry Pi Zero W, una tarjeta microSD en la que cargar la herramienta en cuestión y un cable microUSB para conectar la Raspberry Pi Zero W a cualquier dispositivo.

Los objetivos a nivel de aprendizaje personal son los siguientes:

- Enfrentarnos durante el desarrollo del trabajo con conceptos y técnicas que no hayamos estudiado en el grado y que no nos resulten familiares (como podrían ser conceptos enfocados a la programación), para poder así ampliar nuestros conocimientos y competencias.

- Descubrir y utilizar diferentes herramientas nuevas que nos ayuden a conocer de primera mano la capacidad de explotar vulnerabilidades en los sistemas operativos.

1.3. Plan de trabajo

En primer lugar, nos familiarizaremos con conceptos dentro del pentesting[1], ya que son ajenos a lo que hemos dado en el grado. Investigaremos los distintos tipos de ataques HID,

[1]Técnica que consiste en atacar diferentes entornos o sistemas con el objetivo de detectar y prevenir posibles fallos.

además de las posibles víctimas. Realizaremos estudios de los distintos tipos de vulnerabilidades a explotar a través del puerto USB, así como haremos una investigación de varios exploits que podremos analizar en este trabajo, para finalmente elegir alguno de ellos para analizarlo y estudiarlo. Además, nos familiarizaremos con la Raspberry Pi Zero W, ya que es el dispositivo que usaremos para inyectar los exploits.

Los exploits seleccionados los estudiaremos analizando cómo funcionan, alguna posible solución, en qué sistema operativo son más efectivos... Esto lo desarrollaremos en los capítulos 5, 6 y 7.

1.4. Estructura de la memoria

Esta memoria se estructura en 9 capítulos:

- Capítulo 1: Introducción sobre el tema del Trabajo de Fin de Grado y descripción de los objetivos y plan de trabajo.
- Capítulo 2: Se realiza una breve introducción sobre el Protocolo USB y como funciona, el estándar HID, kernel y la diferencia entre Kernel de linux y windows.
- Capítulo 3: En este capítulo se explica brevemente qué es un exploit, los diferentes tipos, la diferencia entre exploit y vulnerabilidad y los diferentes ataques HID.
- Capítulo 4: Se realiza una breve descripción sobre la Raspberry Pi Zero W, ya que es la herramienta que hemos utilizado en todo este tiempo de investigación. Además, acompañado de un breve tutorial para su configuración.
- Capítulo 5: Se realiza una pequeña descripción del exploit P4wnP1.
- Capítulo 6: En este capítulo se describe el exploit P4wnP1 ALOA.
- Capítulo 7: Se describe el exploit Password Thief, mediante el uso del exploit Responder.
- Capítulo 8: Se extraen una serie de conclusiones finales tras la realización del trabajo y se describe la aportación de cada alumno al proyecto.
- Capítulo 9: Capítulos 1 y Conclusión traducidos a inglés.

Capítulo 2

USB

En este capítulo, haremos una breve descripción introductoria sobre el puerto USB, el estándar HID y cómo funciona, y el Kernel para que más tarde, cuando estudiemos los exploits, estar familiarizados con ellos. Este capítulo esta basado en la documentación oficial de USB, tal y como se puede apreciar en la referencia.[10]

2.1. Historia USB

En informática, los términos USB [4] (siglas de Universal Serial Bus, es decir, Bus Universal en Serie) o BUS *"se refieren a un estándar de conexión y transmisión eléctrica y de datos, entre computadores, dispositivos periféricos y otros aparatos electrónicos.*

Dicho sistema consiste en un bus de comunicaciones guiado por protocolos, cables y conectores de serie universal, que surgió como un modo de universalizar la conexión de los dispositivos a los distintos modelos de computadores.

Conviene aclarar que un bus, en arquitectura computacional, se refiere a un sistema digital de transmisión de datos entre computadores y sus componentes, fabricado en un circuito impreso con resistores y condensadores, y de utilización común en la informática de hoy.

El USB surgió en 1996 en su versión 1.0, como una iniciativa de Intel, Microsoft, IBM,

Compaq, DEC, NEC y Nortel, entonces incompatibles entre sí, por estandarizar los puertos de conexión de sus productos.

Dos años después la especificación 1.1 ya era de uso masivo, y desde entonces su utilización devino la norma, reemplazando a conectores como el puerto serie, puerto paralelo, puerto de juegos, entre otros.

Actualmente la mayoría de los periféricos emplean conectores USB: punteros, unidades flash, teclados, joysticks, escáneres, cámaras, parlantes, teléfonos celulares, etc. Esto ofrece un sinfín de ventajas, más allá de la compatibilidad extrema: los periféricos pueden conectarse en cualquier momento y ser reconocidos al instante, permite la transmisión conjunta de datos y electricidad, y además permite velocidades de transmisión de hasta 1250 Mbps (en su estándar vigente)."

En este proyecto vamos a profundizar en los USB con una baja velocidad de transmisión (1.0): Su tasa de transferencia es de hasta 1,5 Mbit/s (188 kB/s). Su principal utilización es por dispositivos de interfaz humana (Human Interface Device, HID en inglés), como teclados, ratones o memorias USB, entre otros. Al tratarse de la primera especificación y ser utilizada para estos dispositivos, es la que mayores fallos de seguridad posee, ya que el firmware de estos dispositivos está muy desactualizado, a pesar de los parches que han ido actualizando, a lo largo del tiempo, se van encontrando nuevas vulnerabilidades a explotar.

La mayoria de estos fallos son, en su mayoria, causados por fallos de implementación que no han sido contemplados antes de incluirlos en el Kernel. El Kernel de Linux, en lo relacionado con los drivers de dispositivos USB, debería pasar un conjunto de auditorías de seguridad, con el objetivo de detectar estos fallos para después buscar solucionarlos. Ya que, al igual que hay expertos que se dedican a encontrar estos fallos para después corregirlos, hay atacantes que se dedican a detectar nuevas vías a explotar.

Este tipo de vulnerabilidades USB, necesitan que el dispositivo atacante sea conectado de forma física para poder explotarlas, haciendo imposible que el ataque sea de forma remota sin conexión a la máquina.

2.2. Protocolo USB

El bus universal en serie (USB) es una arquitectura de comunicaciones que da a los ordenadores personales (PC) la capacidad de interconectar una gran variedad de dispositivos. Por tanto, el USB es un enlace de comunicación en serie. Los protocolos USB tienen la capacidad de configurar dispositivos, ya sea en el momento del arranque o cuando ya se encuentran en funcionamiento, es decir, en tiempo de ejecución.

Estos dispositivos están distribuidos en varias clases de dispositivos en función de su comportamiento y los protocolos que tengan comunes, ya que sirven funciones similares. Un ejemplo de clase con su respectivo dispositivo es el caso de la clase Audio con el dispositivo Altavoz, o la clase Display con el dispositivo Monitor.

2.2.1. Pipes y Endpoints

El flujo de datos que se produce entre el Host y un dispositivo USB está organizado de forma lógica en tuberías (pipes). Por tanto, podemos definir una tubería como el canal lógico de datos entre ambos. Este flujo, además, está basado en la conexión entre un endpoint (punto final del dispositivo) y la capa de software que controla el hardware USB en el Host.

Los endpoints tienen una configuración realizada por el fabricante de serie, con una identificación única por dispositivo y son agrupados en conjuntos llamados interfaces, de forma que un periférico USB puede controlarse mediante la comunicación de varios endpoints.

El host se comunica de forma física con los dispositivos a través del par de señales diferenciales D+/D- que pasan por el cable.

Desde el punto de vista lógico, el software USB configura y administra los dispositivos a través de la tubería por defecto (llamada tubería de control[1]). Esta tubería utiliza el

[1]Se usan para recibir y responder a las solicitudes de control USB y los datos de clase, transmitir datos cuando son consultados por el controlador y recibir los datos del host

endpoint 0 que, como requisito, debe de estar soportado por todos los dispositivos USB para su correcto funcionamiento.

Durante el proceso de enumeración que se produce una vez se conecta el periférico, el Host obtiene a través de esta tubería (tubería de control) la información y el resto de endpoints de los que dispone el dispositivo. Finalmente, le asigna una dirección única en el BUS. Esto se produce en el nivel lógico, es decir, el más bajo. Por el contrario, en el nivel más alto, el software o el driver que hace uso del dispositivo, se comunica con él directamente mediante las tuberías que, a su vez, están conectadas a los endpoints. Como ejemplo, un dispositivo de almacenamiento puede tener un endpoint establecido para el envío de comandos de lectura y escritura ,y otro para la transferencia de datos.

Por otra parte, existen otro tipo de tuberías que reciben los datos asíncronos del dispositivo y transmiten los datos de baja latencia al dispositivo, y estas reciben el nombre de tuberías de interrupción.

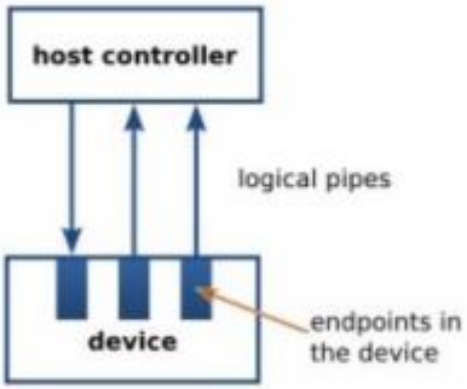


Figura 2.1: *Representación del flujo de datos de las pipes (tuberías).*

2.3. Descriptores

La información que el Host necesita para identificar un dispositivo USB y sus características se encuentra almacenada en los descriptores. Gracias a ellos, le permiten al Host buscar y cargar el driver correcto para su funcionamiento. Podemos organizar los descriptores en función de la información que indican, tal y como lo catalogan en la documentación USB, referenciada anteriormente [10]:

- *Descriptor de dispositivo: Indica qué versión de USB es compatible con el dispositivo, el código del fabricante y el código del producto. También muestra el número de descriptores de configuración activos.*
- *Descriptor de interfaz: Contienen el número de endpoints asociados a una interfaz determinada y la clase de dispositivo a la que pertenece dicha interfaz.*
- *Descriptor de endpoint: Indica la cantidad de endpoints, qué tipo de transferencias se van a realizar (interrupción, control o masiva, entre otros), la longitud y la dirección en la que circula desde el Host: (pueden ser dos opciones: IN = Se realiza desde el dispositivo al Host, OUT = Se realiza desde el Host al dispositivo).*
- *Descriptor de configuración: Un modo de operación del dispositivo es lo que se denomina como configuración (por ejemplo, una videocámara podría actuar como tal o como una cámara web). Al conectar el dispositivo al computador, se debe seleccionar una sola configuración y no puede haber ninguna otra configuración activa que no sea la seleccionada. Además, indica qué potencia necesita el dispositivo para funcionar correctamente, así como el número de interfaces que están asociados a esa configuración.*
- *Descriptor de string: Contiene una cadena de texto, normalmente el nombre del fabricante o el número de serie del dispositivo, y todos los descriptores pueden hacer uso de esta cadena..*

Figura 2.2: *Representación [6]*

El descriptor del dispositivo de clase HID es el encargado de identificar que descriptores están presentes e idicar el tamaño. A su vez, hay dos tipos, el de report que es el encargado de describir cada fragmento de datos que el dispositivo genera y los datos que se miden; y los físicos que son descriptores que se encargan de proporcionar información sobre la parte humana que se utiliza para activar los controles de un dispositivo.

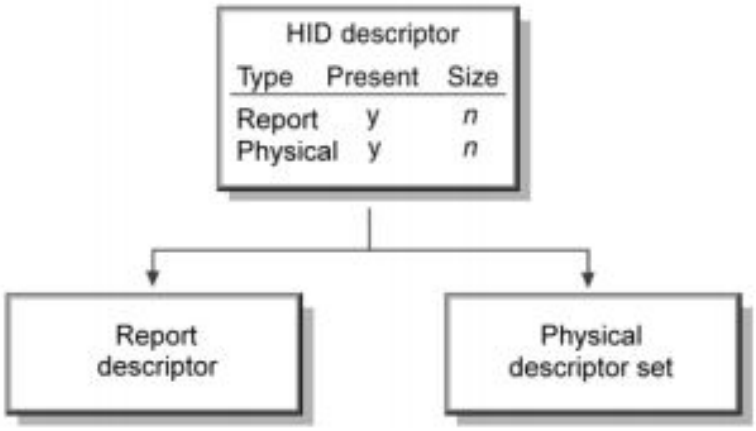


Figura 2.3: *[6]*

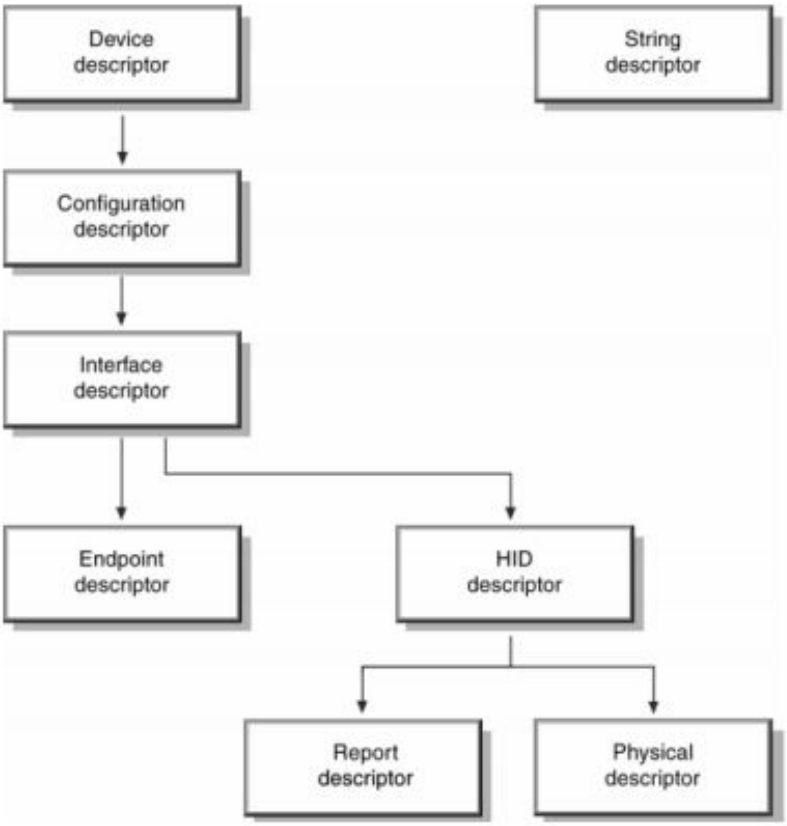


Figura 2.4: *Estructura jerarquizada de un descriptor.[6]*

2.4. Descripción General

Toda la información de un dispositivo se almacena en segmentos de su memoria ROM (Memoria de solo lectura o, por sus siglas en inglés, Read Only Memory), siendo cada uno de esos segmentos un descriptor, como se ha detallado anteriormente. Por consecuente, un dispositivo HID utiliza sus drivers para enrutar y recuperar todos los datos que son almacenados en el descriptor.

Ademas de la clasificación anteriormente hecha sobre los descriptores, podemos definir los que se encargan de describir el formato y el significado que admite el dispositivo USB como descriptor de informe. Mientras que a los datos reales que son intercambiados entre el Host y el dispositivo, como informes(reports)

2.5. Protocolo HID

HID (Human Interface Device) es una de las clases que nos aporta USB. La creación de esta clase tenía como objetivo hacer de forma sencilla la implementación de drivers y aplicaciones de usuario para los dispositivos conectados con los que interactúa el usuario frecuentemente. En HID, el intercambio de datos se hace a través de informes. No son más que estructuras con un cierto formato flexible y pueden manejar muchos tipos de datos. Cada informe tiene un tamaño fijo, y el sistema sabe cómo responder a los datos de cada uno.

2.6. Resumen Funcionamiento

El modo de funcionamiento HID incluye los dispositivos más usados por los usuarios, como pueden ser un teclado, un ratón... Son dispositivos muy extendidos cuyo driver se integra por defecto en el sistema, lo que nos evita asociar un driver a un dispositivo cada vez que los conectamos, para que todo el sistema funcione correctamente.

Lo primero que se hace es inicializar el puerto USB: se ejecuta la función "USBInitialize()", función encargada de habilitar el USB. Mientras tanto, se ejecuta la función "USBTask()", encargada de gestionar los estados del USB y manejar los eventos. La importancia de esta función recae en que debe ser llamada periódicamente, ya que es la encargada de mantener el USB funcionando.

Cuando se establece la conexión entre el cable USB al host, en primer lugar, se produce la enumeración por el cual el host (el propio computador) identifica al dispositivo y, si no existe ningún problema, se añade a la lista de dispositivos USB (tal y como el host lo tenga configurado en ese momento). Al tratarse de un dispositivo USB, que en su firmware[2] implementa la clase HID, tiene incorporados los drivers necesarios para su correcta comunicación,

[2]programa de software que permite controlar y comunicarse con el hardware de un equipo de forma directa.

y esta es la principal ventaja que nos facilita el estandar HID, haciendo innecesaria la instalación de drivers específicos para cada dispositivo. Esta comunicación entre el dispositivo y el driver se realiza a través de tuberías, ya sean de tipo control o de interrrupción.

En resumen, el HID función driver del dispositivo USB (USB device) se comunica con el HID class driver (USB host) usando la tubería de control, que está instaurada por defecto. Es obligatorio usar la tubería de interrupción en la recepción del host, mientras que en la recepción del dispositivo es opcional.

Como comentábamos anteriormente, en los dispositivos que implementan la clase HID la información se transmite y recibe a través de unas estructuras de datos llamadas informes o reports. El tamaño y tipos de datos de esa estructura queda definida en el firmware del PIC[3] concretamente en el archivo fuente (ex_usb_hid.c) y en el archivo que define los descriptores (usb_desc_hid.h).

[3]circuito integrado programable, el cual contiene todos los componentes para poder realizar y controlar una tarea

2.7. Kernel

2.7.1. ¿Qué es?

El Kernel (o núcleo) es la parte central de un sistema operativo (S.O.), y es el encargado de realizar toda la comunicación entre el software y el hardware de un ordenador. Es el encargado de evitar sobrecargas en el sistema, gasto innecesario de recursos, gestionar procesadores, entre otras funciones. Por tanto, es el encargado de gestionar los diferentes recursos mediante servicios de llamadas al S.O. En otros términos, es lo que denominaríamos como el motor de un S.O., ya que es el principal responsable de que el computador funcione correctamente, tanto el software como el hardware.

2.7.2. Diferencias entre el kernel de Linux y el kernel de Windows

El Kernel de Windows es inaccesible, por tanto, nadie ajeno a Microsoft puede modificarlo, mientras que el Kernel de Linux es de código abierto, lo que permite que existan distribuciones de Linux diferentes cuya finalidad puede ser totalmente distinta a las otras, como por ejemplo la distribución de Kali Linux, usada en este proyecto, diseñada para su uso en ciberseguridad.

Al ser código abierto, el Kernel de Linux frecuenta más la posibilidad de que se encuentre más vulnerable debido a que el desarrollador carezca del suficiente conocimiento o experiencia.

Capítulo 3

Exploit

3.1. ¿Qué es?

Un exploit [16] es *cualquier ataque que aprovecha las vulnerabilidades conocidas de las aplicaciones, las redes, los sistemas operativos o el hardware*. Por lo general, los exploits, aunque no son un programa o aplicación software como tal, pueden adoptar la forma de uno o de una secuencia de código específica diseñada para provocar efectos imprevistos, habitualmente dañinos para el usuario atacado, tales como hacerse con el control de los computadores o robar credenciales (tal y como hacemos en el exploit estudiado en el capítulo 7). Este tipo de acciones se producen, normalmente, sin que el usuario tenga conocimiento de que ha sido o está siendo atacado.

Tanto el software (ya sea un sistema operativo o las aplicaciones) como las redes incluyen una protección integrada por defecto frente a estas vulnerabilidades y a los atacantes que intenten aprovecharlas, aunque esta protección, en algunos casos, no es suficiente y debe reforzarse mediante la instalación y uso de herramientas antivirus externas (a ser posible, de pago, ya que ofrecen mejor protección frente a estos ataques que los antivirus gratuitos), además de mantener el sistema operativo actualizado.

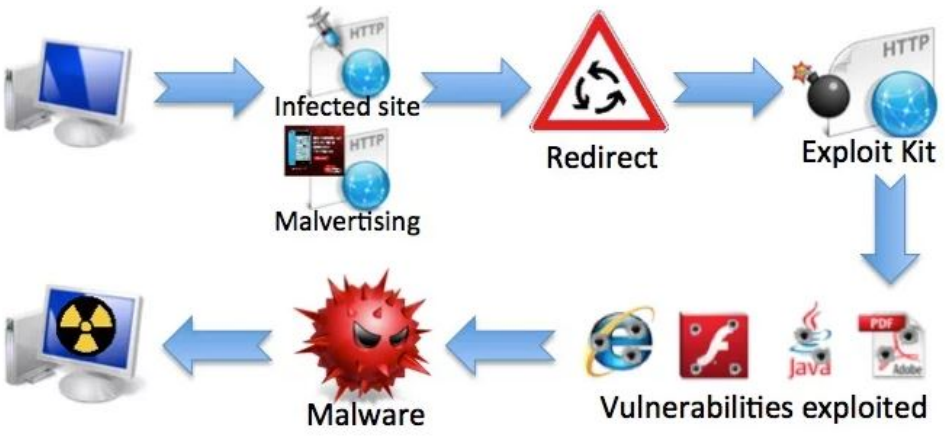


Figura 3.1: *Ruta de ataque [8].*

3.2. Diferencia entre vulnerabilidad y exploit

Como se ha mencionado anteriormente, un exploit es cualquier ataque que aprovecha las vulnerabilidades, mientras que una vulnerabilidad es el punto débil o defecto de seguridad que puede proporcionar el acceso a los atacantes. En un caso idílico, si no existe ninguna vulnerabilidad, no puede haber ningún exploit que pueda funcionar.

Aunque un sistema presente vulnerabilidades, no significa que puedan ser potencialmente explotables en su totalidad ya que, dependiendo de cada una de ellas en particular, impide que los atacantes descifren una secuencia de código válida con la que poder explotarla. En otras palabras, todas las vulnerabilidades pueden no ser aprovechables, ya que pueden llegar a darse casos extremos: unas pueden llevar a un camino cerrado del cual el atacante no pueda sacar información o instalar ningún tipo de software malicioso mientras que otras pueden provocar comportamientos inesperados, como por ejemplo bloquear el sistema o que funcione incorrectamente. Además, una vulnerabilidad puede aprovecharse para producir ataques DoS (denegación de servicio) o DDoS (denegación de servicio distribuida), en los que los asaltantes pueden atacar un sitio web en concreto o un sistema sin necesidad de

utilizar un exploit (estos dos ataques se utilizan para sobrecargar un servidor con más peticiones de las que admite, haciendo que el servidor colapse y necesite reiniciarse, con el tiempo que necesita para ello, haciendo imposible su uso mientras dure el proceso). [17]

3.3. ¿Un exploit es un malware?

La respuesta es no, son dos términos diferentes. Malware es *cualquier tipo de software malicioso para un sistema operativo*, mientras que un exploit es, como hemos mencionado anteriormente, *cualquier ataque que aprovecha las vulnerabilidades*. Habitualmente, ocurre que un atacante haga uso de un exploit para acceder al sistema por una vulnerabilidad para luego instalar malware, pero el exploit en sí no es malicioso.

3.4. ¿De dónde vienen los exploits?

Los exploits surgen de los investigadores que buscan vulnerabilidades para después encontrar la forma de explotarlas y finalmente buscar la forma de corregirlas, para intentar evitar a los desarrolladores maliciosos o ciberdelincuentes, mientras que estos buscan la forma de acceder a dispositivos ajenos a través de vulnerabilidades para, como se ha mencionado anteriormente, robar información del propio computador, de la red a la que se encuentre conectado o, simplemente, hacerse con el control del computador. Algunos pueden llegar incluso a crear conjuntos de exploits que, a menudo, son empaquetados en otro software para ejecutarse en el momento de la instalación o ejecución de ese software.

Los atacantes pueden comprar o alquilar estos paquetes en la deep web[1] para, posteriormente, ocultarlos en sitios web intervenidos o en anuncios publicitarios.

[1]Es todo el contenido online que no está indexado en los motores de búsqueda convencionales, tales como páginas privadas, bases de datos o páginas que han sido desindexadas por los motores de búsqueda.

3.5. ¿Cómo reconocer un ataque de exploit?

A simple vista, no hay signos reconocibles que permitan saber si un sistema está infectado o no, por lo que el usuario no tiene forma de verlo con lo que un usuario prácticamente no tiene forma de saberlo. Para intentar evitarlo, como hemos mencionado al inicio de este capítulo, es imprescindible tener el sistema operativo actualizado, con todos los parches de seguridad que los desarrolladores publiquen, ya sea para el propio sistema o para las aplicaciones que el usuario tenga instaladas ya que si se ha corregido alguna vulnerabilidad con un parche nuevo, nos quedamos indefensos frente a esa vulnerabilidad que ya es conocida y corregida, simplemente por no actualizar.

Cuando un exploit comienza a ejecutarse, el usuario no es consciente de eso hasta que se haya producido la instalación de algún malware aunque, normalmente, tampoco es consciente de dicha instalación ya que suele hacerse en segundo plano, es decir, mientras que el usuario está ejecutando otra aplicación, momento en el cual el usuario puede empezar a notar los efectos de ese malware, que pueden ser imperceptibles, como que el rendimiento del equipo se reduzca en pequeñas cantidades, que el usuario puede achacar a que alguna de las aplicaciones que tenga en ejecución esté consumiendo algo más de recursos de lo habitual y no le de importancia, hasta consecuencias más graves, como por ejemplo, el sistema operativo comience a bloquearse frecuentemente, se produzcan cambios de configuración inexplicables, ventanas emergentes de anuncios en lugares en los que no deberían aparecer o la pérdida de espacio de almacenamiento en el disco duro.

3.6. Tipos de exploits

Existen varias clasificaciones, siendo las más comunes dependiendo del conocimiento que se tenga sobre ellos o de las formas de acceso que se logran tener y del objetivo que tenga el atacante.

En primer lugar, los clasificaremos en función de su conocimiento de cara al público en dos tipos: conocidos y desconocidos (también llamados "día cero" (0-day)).

- `Exploits conocidos`: Son todos aquellos de los que se conoce su existencia y se sabe qué vulnerabilidades explotan, y así podemos tomar medidas preventivas para que nuestros computadores no lleguen a ser afectados. Estos exploits aparecen en la mayoría de las noticias de seguridad, además que surgen varios de ellos cada día, de la misma forma que se van descubriendo las vulnerabilidades que tratan de aprovechar. Debido a esto, es crucial analizar qué vulnerabilidades están siendo explotadas y comprobar que el sistema y aplicaciones se encuentran actualizados y, en el caso de no existir una actualización disponible por el momento, aplicar una serie de técnicas que ayuden a reducir el impacto las posibles amenazas. Existen webs especializadas en identificar e informar de todos los fallos y los parches para solucionarlos que surgen día a día, como por ejemplo Exploit Database.

- `Exploits desconocidos (0-day)`: Son los que utilizan sobre vulnerabilidades de las que el público no tiene constancia y, por tanto, suponen una grave amenaza, especialmente si se utilizan para ataques dirigidos (DoS o DDoS, mencionados anteriormente). Cuando este tipo de exploits entran en acción, como no se sabe cuál es la vulnerabilidad a la que atacan, es bastante probable que no existan todavía las medidas de seguridad necesarias para bloquearlos, y eso hace que sean prácticamente indetectables. Por ello es importante que los desarrolladores de software como los investigadores averiguen cuanto antes como funciona el exploit para llegar a corregir la vulnerabilidad y no estar indefenso frente a ese ataque.

Por último, los clasificaremos en función de sus formas de acceso y según el objetivo que tenga el atacante:

- `Exploits remotos`: Son todos los que no el atacante no necesita encontrarse físicamente frente al computador que quiere atacar.
- `Exploits locales`: Son aquellos en los que el atacante se encuentra de forma física frente al sistema que quiere atacar. Ya sea para conectar un dispositivo físicamente que luego este automatizado o atacando de forma personal.
- `Exploit cliente`: Este tipo de exploit es el más común actualmente. Para que se puedan ejecutar, el atacante no necesita estar delante del computador atacado, sino que es el propio usuario el que le da acceso sin tener conocimiento de ello. El ejemplo más claro es el "phising", que consiste en adjuntar un archivo en un correo electrónico que, a primera vista, no parece malicioso, pero si el usuario descarga ese archivo, entonces el exploit inicia su ejecución.

3.7. Tipos de ataques HID

Este apartado esta basado en el contenido del artículo [15] de un grupo de investigadores de la Universidad Ben-Gurion de Negev en Israel publicado en el año 2017, en la revista ScienceDirect, en el que identifican 29 formas distintas en las que podría usarse un dispositivo USB para comprometer un equipo. El objetivo de esa investigación fue poner en conciencia a los usuarios las numerosas posibilidades que tienen los atacantes para infectar un sistema y robar de forma encubierta los datos a través del puerto USB. La recomendación del equipo de investigación es que los dispositivos USB estén prohibidos o al menos controlados estrictamente en redes seguras.

Estos métodos de explotación fueron clasificados en cuatro categorías según la forma en la que se lleve a cabo el ataque:

1. *Ataques reprogramando el microcontrolador USB*:El dispositivo visualmente tiene una apariencia normal como puede ser un cargador usb , pero el microcontrolador esta programado de tal forma que inyecta pulsaciones de teclas.

2. *Ataques reprogramando el firmware del dispositivo USB*: Reprograman el firmware original del dispositivo USB para hacer uso de herramientas que aprovechan vulnerabilidades.

3. *Ataques basados en dispositivos USB no reprogramados*:Estos ataques se basan en aprovechar la forma de actuar de los SO con los prótocolos/estándares USB (Por ejemplo, el exploit Responder que lo describiremos más profundamente en el capítulo 7)

4. *Ataques eléctricos basados en USB*: Por ejemplo, USB Killer: destruye dispositivos de forma permanente al insertar un dispositivo USB que realiza una recarga eléctrico.

Capítulo 4

Raspberry Pi Zero W

4.1. Introducción

La Raspberry Pi Zero W [13] es la versión con WiFi y Bluetooth de la Raspberry Pi Zero. Es una computadora personal completa del tamaño de una tarjeta de crédito, compuesto por un procesador ARM, dos puertos USB y una conexión mini HDMI. Funciona con distintas distribuciones Linux (Raspbian, Noobs, XBMC) y soporta herramientas de software libre como Python o KOffice, entre otros. Las especificaciones de esta nueva Raspberry Pi son las mismas que la Zero, pero potenciadas con la conectividad sin cables.

La Raspberry Pi Zero es el modelo más pequeño Raspberry Pi. Debido al procesador ARM soporta el rango completo de distribuciones ARM GNU/Linux. Sus usos pueden ser muy variados, desde usarse para juegos, pruebas de pentesting, hasta proyectos bastante interesantes como la creación de un dron low cost, alarmas, proyectores...

Debido a su tamaño reducido, es fácil de ocultar, lo que le hace eficiente para ataques HID, ya que para este tipo de ataques es primordial pasar desapercibido ya que es un ataque que se realiza de forma presencial. Además, esta Raspberry al estar potenciada frente a su antecesora, la Raspberry Pi Zero, en temas de conectividad, hace que pueda actuar como un adaptador de Ethernet, lo que amplia el catálogo de posibles ataques.

4.2. Especificaciones

- LAN inalámbrica 802.11 b/g/n.
- 512 MB de memoria RAM.
- Puertos Mini HDMI y USB On-The-Go.
- 1 GHz, CPU de un solo núcleo.
- Bluetooth 4.1.
- Alimentación micro USB.
- Encabezado de 40 pines compatible con HAT.
- Video compuesto y restablecer encabezados.
- Conector de cámara CSI (solo v1.3).
- Dimensiones: 65mm x 30mm x 5mm.

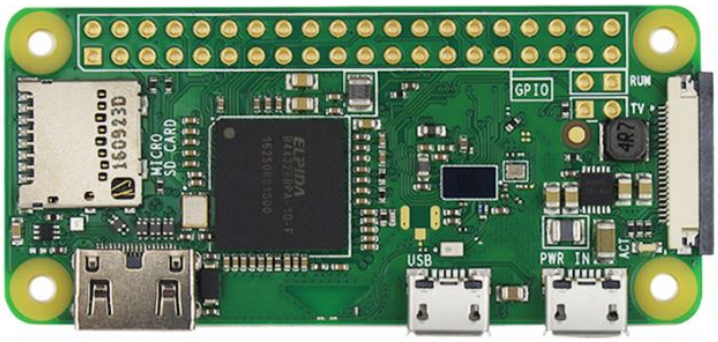


Figura 4.1: *Imagen de Raspberry Pi Zero W.[7]*

4.3. Tutorial de instalación

En primer lugar, debemos montar la imagen del SO que queremos instalar en la Raspberry. Para ello introduciremos la tarjeta microUSB en el ordenador. La montaremos a través de cualquier programa que pueda montar de imágenes (en nuestro caso, Raspberry Pi Imager, el cual se puede descargar desde su página web oficial).

Una vez montada, antes de introducir la memoria microSD en la Raspberry deberemos realizar los siguientes pasos para que sea un dispositivo Ethernet USB:

- Abrimos el archivo config.txt con cualquier editor de texto plano (Bloc de notas, Wordpad, etc).
- Agregamos un salto de línea al final del fichero y escribimos *dtoverlay=dwc2*.
- Abrimos el archivo archivo cmdline.tx con cualquier editor de texto plano.
- Buscamos la palabra *rootwait* y añadimos *modules-load=dwc2,g_ther* justo después.
- Creamos un archivo vacío llamado ssh, sin extensión.

Para configurar la conexión WiFi en nuestra Raspberry Pi sin necesidad de teclado o monitor, creamos un archivo que lo llamaremos wpa_supplicant.conf con la siguiente estructura:

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=ES

network={
        ssid="MiWiFi"
        psk="MiContraseña"
        key_mgmt=WPA-PSK
}
```

Figura 4.2: *Texto wpa_ supplicant.conf*

Insertamos la tarjeta SD en la Raspberry y conectamos el puerto USB a nuestro ordenador. Después de que la Raspberry se inicie, el Host debería de reconocer como un nuevo adaptador Ethernet. A continuación, abriremos PuTTy (emulador gratuito de terminal que soporta SSH) y haremos SSH a la Raspberry. Por defecto, el usuario es *pi* y la contraseña es *raspberry*.

Una vez completados estos pasos, estaríamos conectados a través de SSH a nuestra Raspberry Pi.

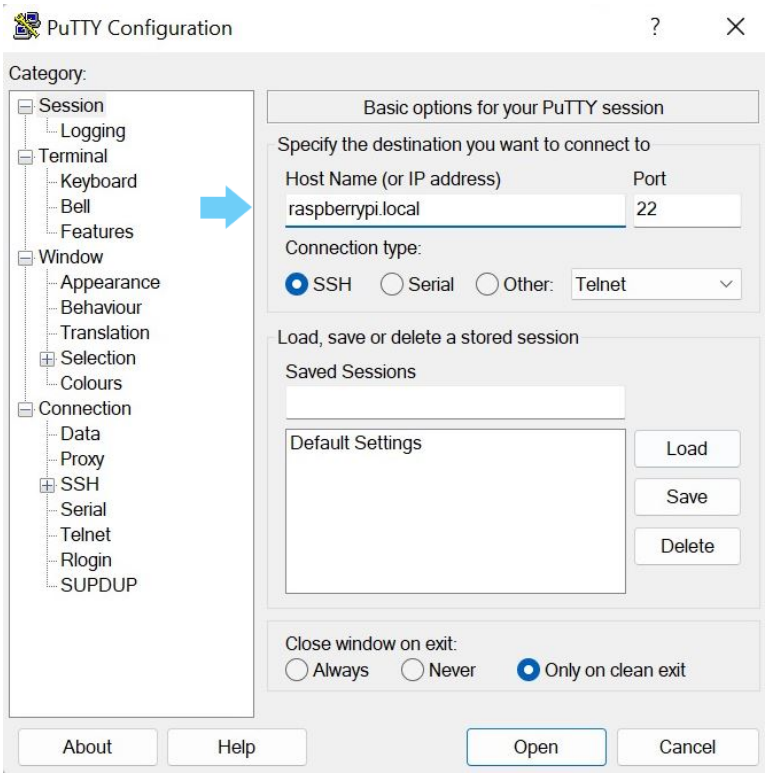


Figura 4.3: *Configuración Putty*

Capítulo 5

P4wnP1

Este exploit se describirá brevemente debido a los motivos detallados en la conclusión del mismo.

5.1. Descripción y características

P4wnP1 [11] es *una plataforma de ataque USB, totalmente modificable, creada para su uso en Raspberry Pi Zero o Raspberry Pi Zero W.*

El exploit P4wnP1 explota las vulnerabilidades que posee Windows a través de un puerto USB, ya que el sistema operativo puede reconocer la Raspberry (teniendo instalado este exploit) como un dispositivo USB habitual, como puede ser una memoria flash (pendrive o memoria USB) y, una vez conectada en el computador al que se le va a realizar el ataque, ejecuta el ataque que se haya cargado previamente en la configuración de dicho exploit.

Por medio de esta herramienta podremos habilitar e inyectar cientos de scripts y payloads[1]. Entre ellos, se encuentran los scripts P4wnP1 LockPicker Windows 10, HID Keyboard o HID Mouse, entre otros, los cuales explicaremos a continuación ya que han sido los que hemos probado.

[1]Conocido como carga útil, es el contenido del script, es decir, las instrucciones que se van a ejecutar

5.2. Payloads

5.2.1. P4wnP1 LockPicker Windows 10

El funcionamiento de este payload se basa en un ataque "de fuerza bruta", es decir, prueba combinaciones de caracteres alfanuméricos hasta conseguir cada uno de los caracteres que componen la contraseña del usuario del ordenador cuando el ordenador se encuentra encendido pero sin haber iniciado sesión.

Aunque, como su propio nombre lo indica, esté destinado a Windows 10, también se puede ejecutar en equipos con versiones anteriores de Windows, como Windows 7, Windows 8 o Windows 8.1, así como en dispositivos que tengan instalado un sistema basado en Linux o en MacOS (modificando previamente un archivo para indicar que se va a ejecutar en Windows, Linux o MacOS).

Dicho ataque funciona debido a que la gran mayoría de los computadores actuales hacen uso de la tecnología Plug and play, es decir, conectar el dispositivo USB al conector USB del computador y acceder al contenido del dispositivo, sin hacer comprobaciones adicionales en busca de contenido malicioso. Sin embargo, en muchos casos, como es el caso de las empresas, los dispositivos de instalación se encuentran limitados por autorizaciones, es decir, necesitan de una confirmación por parte del administrador del equipo o de un usuario con permisos de administrador para poder ejecutarse o instalar el contenido del dispositivo. En el caso de este ataque, al modificar el firmware del Ethernet/LAN y hacer creer al computador que es una tarjeta de red, este permite su instalación sin ningún tipo de limitación, ya que se encuentra en la lista blanca (lista de componentes y dispositivos reconocidos como no maliciosos), permitiendo el éxito del ataque.

Los computadores constantemente están generando tráfico aunque no lo estemos utilizando y la mayoría de estos confían en su red local, y es precisamente eso lo que aprovecha este exploit, capturar ese tráfico para poder generar y guardar la contraseña del usuario

mediante hashes[2].

Los hashes que contienen contraseñas de inicio de sesión se transmiten desde los computadores con Windows a las redes WiFi locales. Al interceptar y descifrar estos valores hash usando Responder[3] y John the Ripper[4], respectivamente, podemos capturar las credenciales de inicio de sesión del objetivo. Es decir, además de capturar los hash, se entregarán a John the Ripper para ser descifrados. En caso de que el ataque tenga éxito, este payload se encargará de escribir la contraseña capturada en la caja de texto del inicio de sesión para acceder al computador en cuestión.

Este ataque, como es de suponer, tiene mayores índices de éxito cuando el usuario al que se desea atacar dispone de una contraseña con débil seguridad (clave compuesta por su nombre y su fecha de nacimiento, por ejemplo) que cuando dispone de una contraseña con alta seguridad (clave compuesta por mayúsculas, minúsculas, números y caracteres especiales), ya que la complejidad para adivinar dicha contraseña aumenta exponencialmente.

5.2.2. HID Keyboard

Mediante este payload, podemos realizar un script para que, a través de él, se puedan inyectar scripts o pulsaciones de teclado al computador objetivo para, mediante comandos automatizados, acceder al directorio o directorios para extraer información.

5.2.3. HID Mouse

Este payload tiene un script por defecto el cual, mediante automatización de clicks y movimiento del ratón, es capaz de ejecutar el programa Microsoft Paint y realizar un dibujo sencillo (en el caso de que el sistema operativo atacado sea Windows). Modificando este

[2]Funciones criptográficas que codifican la información de los bloques en una serie de caracteres de longitud fija.

[3]Exploit que se detallará en los próximos capítulos.

[4]Herramienta usada para descifrar contraseñas y, usada por analistas de seguridad, para comprobar la seguridad de las mismas.

payload, a través de scripts personalizados, podemos llegar a realizar cualquier tipo de acción que solicite el uso del ratón.

5.3. Conclusión

Este exploit fue el primero que se intentó probar, con el payload P4wnP1 LockPicker Windows 10, ya que pensamos que era el ataque más interesante de todos los que ofrecía el exploit y, después de varios intentos fallidos de ejecución, primero estábamos convencidos de que no se ejecutaba correctamente por fallo nuestro pero, investigando más tarde, descubrimos que Windows lanzó un parche de seguridad en 2017 solucionando la vulnerabilidad que explotaba el ataque. Además, el creador de este exploit dejó de lanzar actualizaciones para desarrollar su sucesor, P4wnP1 ALOA. Debido a eso, seguimos intentando con otros de los payloads, pero la mayoría con el mismo resultado, así que decidimos probar con la versión ALOA que, a día de hoy, sigue en funcionamiento y no se ha conseguido solucionar. Este exploit lo explicaremos en el próximo capítulo.

Capítulo 6

P4wnP1 ALOA

En este capítulo explicaremos detalladamente el exploit P4wnP1 ALOA[12]. La imagen Raspberry Pi Zero W P4wnP1 ALOA (A Little Ofensive Application) es una versión altamente personalizada de Kali Linux. Le permite conectar la Raspberry Pi a una computadora y enviar comandos, o usar su red, todo sin tener que interactuar con ella.

El software P4wnP1 ALOA incluye una serie de características que tenía el P4wnP1 original, como la emulación de dispositivo USB Plug Play y Wi-Fi a través de una copia modificada del firmware Nexmon que permite ataques KARMA, compatibilidad con Bluetooth, canal encubierto de WiFi y, aunque se incluye el modo de monitor, NO es compatible, pero también agrega HIDScript, que es similar a DuckyScript para cargar scripts pero basado en JavaScript.

6.1. Funcionamiento del exploit

Este exploit se basa en el sistema operativo Kali Linux, especializado para ataques y ciberseguridad. Este sistema soporta los tres sistemas operativos existentes en la actualidad, Windows, Linux y MacOS. Prácticamente no existe ninguna defensa ante este tipo de ataque, salvo proteger los propios puertos USB del ordenador o dispositivo al que se conecte para que pueda reconocer la Raspberry y el usuario pueda bloquearla manualmente.

Contiene una interfaz gráfica básica con diferentes opciones de configuración para que el dispositivo conectado, en este caso una Raspberry Pi Zero W, sea reconocido en el equipo atacado como un dispositivo de almacenamiento o para que se inicie oculta sin dejar rastro en el equipo en el que está conectada.

Para lanzar ataques, es necesario que el equipo atacante se conecte a la red Wi-Fi que emite la Raspberry, una vez conectada al equipo atacado y se conecte mediante un navegador de internet a la dirección IP de la propia raspberry una vez ésta haya arrancado. De esta manera, se cargará dicha interfaz gráfica y se podrán ejecutar los ataques remotamente. Para la conexión, es necesario cualquier equipo que se pueda conectar a internet vía Wi-Fi, y puede ser un equipo de sobremesa, un ordenador portátil o un teléfono móvil inteligente

6.2. Tutorial de instalación

La instalación de este exploit es muy sencilla, teniendo dos opciones de instalación. La primera opción es descargar la imagen del exploit desde la página oficial de GitHub y montarla en la tarjeta microSD usando el programa mencionado anteriormente, Raspberry Pi Imager. Una vez el programa acabe de montar la imagen, hay que conectar la tarjeta microSD a la Raspberry Pi, y esta al ordenador que se desee atacar, para posteriormente proceder a la conexión desde nuestro equipo.

La segunda opción es mediante comandos utilizando la consola de comandos de Windows o el terminal de Linux. Primero hay que descargar la imagen, al igual que con la otra opción. Una vez descargada, hay que ejecutar el siguiente comando para instalar la imagen en la tarjeta microSD:

xzcat kali-linux-2022.3-raspberry-pi-zero-w-p4wnp1-aloa-armel.img.xz | sudo dd of=/dev/sdb bs=4M status=progress

Una vez acabe la ejecución, se desconecta la tarjeta microSD del equipo y se introduce

en la Raspberry, y se procede al igual que con la anterior opción.

6.3. Características

Entre las posibilidades de ataques y configuraciones que proporciona esta herramienta, se encuentran la ejecución de instrucciones contra un sistema operativo mediante la inyección de cadenas de pulsaciones de teclas (keystroke injection (HIDScript)) tan pronto como se conecte la Raspberry al computador atacado (mediante acciones automáticas (TriggerActions)), así como poder modificar la configuración de la señal WiFi que emite.

6.4. Materiales Necesarios

A continuación se lista lo necesario para instalar el exploit.

- `Raspberry Pi Zero W`
- `Cable microUSB`
- `Imagen del framework P4wnP1 ALOA`
- `Tarjeta de memoria microSD (recomendable de 16 GB de capacidad)`
- `Software para escribir la imagen en la tarjeta (Recomendamos Ether y Raspberry Pi Imager`

6.5. Opciones de configuración

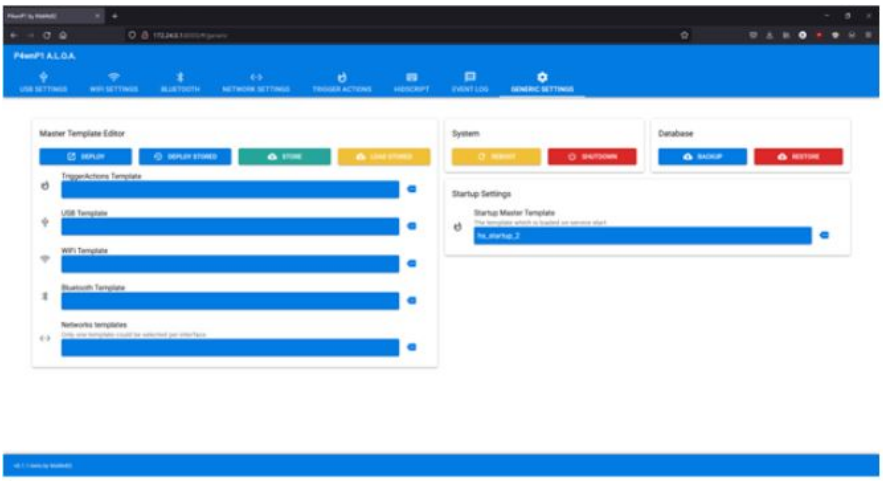


Figura 6.1: *Imagen de configuración genérica del exploit vía GUI.*

En ella se pueden cargar diversas opciones para que se ejecuten cuando se conecte la Raspberry a un dispositivo y arranque el sistema, tales como la configuración propia de WiFi o el comportamiento deseado al conectarse al dispositivo.

Desde esta GUI (Graphic User Interface / Interfaz gráfica de usuario) se pueden configurar todos los parámetros de comportamiento de la Raspberry en cuanto a la conexión a otro equipo, como son cambiar el nombre (SSID) de la red WiFi que proporciona el exploit, ejecución automática de scripts dependiendo de la acción que realice el usuario (como conectar la Raspberry al puerto USB del dispositivo, abrir algún programa o aplicación o escribir en el teclado, entre otros), cambiar el comportamiento en cuanto a cómo reconoce el computador la Raspberry cuando se conecta (en este caso, se reconoce como si se tratase de una memoria flash / pendrive). Además, contiene una opción para ejecutar manualmente los scripts que tiene cargados en memoria la propia Raspberry, así como un editor de texto sencillo para escribir el código que se desea ejecutar. Respecto a la ejecución de scripts, esta GUI permite ejecutar con éxito ataques de simulación de movimiento del cursor, pero no de pulsaciones de teclado.

6.6. Estudio proceso de ataque

El funcionamiento de este exploit se basa en ataques de HID, es decir, *instrucciones que simulan el comportamiento de dispositivos de interfaz humana, tales como un ratón o un teclado.*

El dispositivo atacado, en nuestro caso, computadores con el sistema operativo Windows, por defecto, confían en este tipo de dispositivos, ya que no son maliciosos para el sistema, con lo que no bloquean la ejecución del exploit debido a que reconocen la Raspberry como un dispositivo HID. Como los ataques que se ejecutan contra el sistema son instrucciones propias del sistema, instrucciones que un usuario puede ejecutar de forma habitual, pueden convertir este exploit en algo bastante perjudicial para nuestro computador, y las únicas formas de evitar el ataque son no conectar ningún dispositivo que no reconozcamos o bloquear el acceso de los puertos usb, como veremos más adelante.

El proceso para ejecutar un ataque consiste en conectar la Raspberry por USB al computador que se desea atacar, esperar entre 30 y 60 segundos hasta que el sistema de la Raspberry (Kali Linux) se encuentre en pleno funcionamiento y, desde nuestro propio computador, teléfono inteligente, o cualquier otro dispositivo con acceso a internet, conectarnos a la red WiFi que proporciona la Raspberry. Una vez establecida la conexión, se puede acceder al exploit por tres métodos: web, ssh o bluetooth. Para el desarrollo del exploit, se han hecho pruebas por vía web y ssh. Para acceder por web, hay que iniciar cualquier navegador de internet y acceder a la dirección IP de la raspberry (en nuestro caso, 172.24.0.1, y puerto 8000, quedando de esta manera: *172.24.0.1:8000*). Una vez dentro de la versión web, se pueden ejecutar varias opciones, como cambiar la configuración de la red WiFi o el comportamiento al conectar la Raspberry por USB.

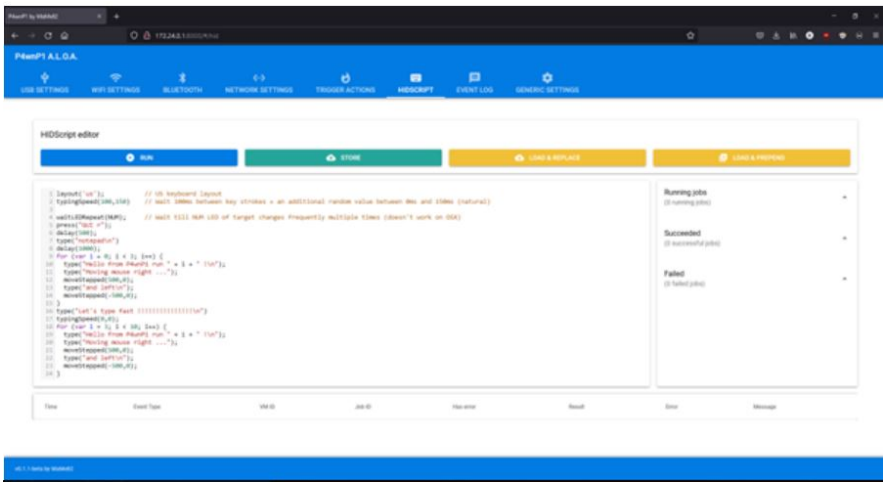


Figura 6.2: *Imagen de la ejecución del exploit P4wnP1 ALOA por interfaz web.*

Para acceder vía SSH, es necesario inciciar la consola de comandos de Windows, Windows Powershell o programas específicos para este tipo de conexiones, tales como Putty. Para el desarrollo de este apartado, se ha optado por Windows Powershell.

Este tipo de conexión es más compleja pero también tiene mayor funcionalidad, ya que permite ejecutar instrucciones directamente en el dispositivo sin tener que cargar un script completo, que también puede ejecutar. Con esta conexión se puede acceder al dispositivo en su totalidad, ya que permite ejecutar instrucciones como crear, copiar y modificar archivos, así como ejecutar programas que tenga instalados el usuario o hasta bloquear la sesión del usuario para luego ejecutar otra instrucción introduciendo la contraseña en el campo de texto. También permite la ejecución automática de instrucciones o exploits, pero es más complejo porque hay que modificar manualmente los archivos de configuración.

Para acceder por este método, hay que ejecutar el comando *ssh* seguido del usuario al que se quiere acceder de la Raspberry, seguido de un arroba (@) y la dirección IP de la Raspberry. Si los datos son correctos, se debe proporcionar la contraseña de dicho usuario para poder acceder. En nuestro caso, estos parámetros son *root* como usuario, *172.24.0.1* como dirección IP y, como contraseña, *toor*.

Una vez el inicio de sesión haya tenido éxito, se puede acceder a todo el contenido de la raspberry mediante comandos de Linux, ya que el sistema que tiene incorporado es Kali Linux, así como ejecutar instrucciones sencillas o complejas directamente con un comando P4wnP1cli hid -c, seguido de las instrucciones a ejecutar o una instrucción para ejecutar un script completo en concreto.

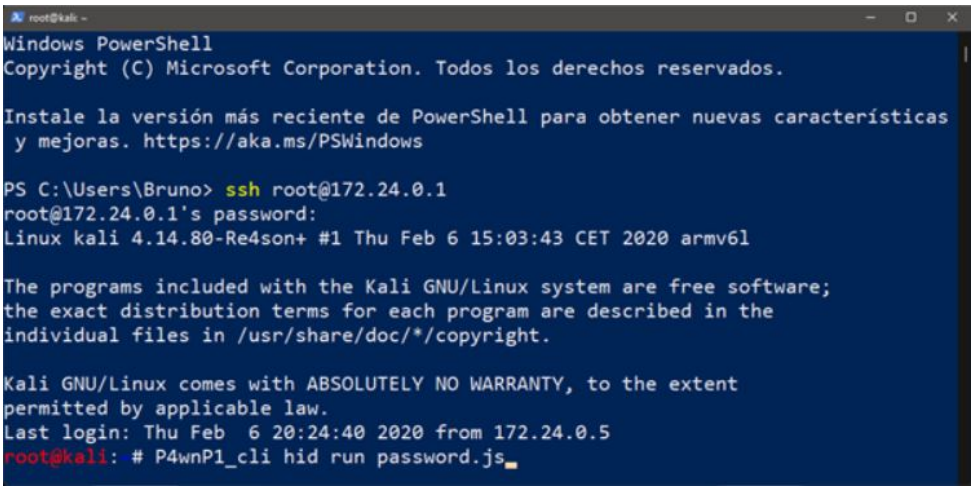


Figura 6.3: *Imagen correspondiente al inicio de sesión correcto y ejecución de un script.*

6.7. Sistemas operativos atacados

Windows 10, Windows 11. Debido a que los scripts que se pueden ejecutar pueden ser instrucciones propias de Windows, o sus propias variables, los ataques se ejecutan sin ningún tipo de problema o restricción. Se ha probado un script que simula el movimiento del ratón del equipo hacia arriba y hacia abajo, sin que el usuario mueva o interactúe de ninguna manera con el computador.

6.8. Posible defensa al ataque

Como se ha mencionado anteriormente, las únicas soluciones posibles a este ataque son no conectar ningún dispositivo usb que no reconozcamos o bloquear el acceso a los puertos usb mediante las políticas de grupo propias de Windows, las cuales bloquean el arranque y reproducción automática de los dispositivos usb que se conecten al equipo y sea el usuario el que decida qué dispositivo desbloquear, o implementar un sistema de lista blanca (white list), en la que se añadan los dispositivos reconocidos, ya sean propios o que tengamos la certeza de que no son perjudiciales para nuestro equipo, para después bloquear el acceso de todos los demás dispositivos que se conecten al equipo y, como en el caso de las políticas de grupo, desbloquear manualmente los dispositivos que deseemos.

Otras posibles defensas, aunque un poco más arriesgadas, serían tener un usuario sin permisos de administrador para ningún proceso o acción, con el cual iniciaríamos sesión con ese usuario y conectaríamos el dispositivo a nuestro computador para comprobar el contenido y, en caso de ser malicioso, no debería poder ejecutar ningún tipo de exploit ya que no tendría acceso como administrador del sistema, o en el caso de disponer de una máquina virtual, arrancar dicha máquina virtual, conectar el dispositivo y nos debería preguntar en qué máquina queremos abrir el dispositivo, si en el *host* (es decir, nuestro computador real) o en el *guest* (la máquina virtual). Nos referimos a estas soluciones como más arriesgadas debido a que, si el exploit tiene una configuración determinada, podría esquivar la seguridad de no

ser administrador para proporcionarse los permisos necesarios para serlo o para ejecutarse en nuestro computador real sin ninguna notificación.

Como demostración, la siguiente imagen muestra cómo una memoria flash (con el nombre "NUEVO VOL") es reconocida por la máquina virtual y sólo accesible de esa forma mientras que la máquina real ni siquiera lo muestra en el explorador de archivos:

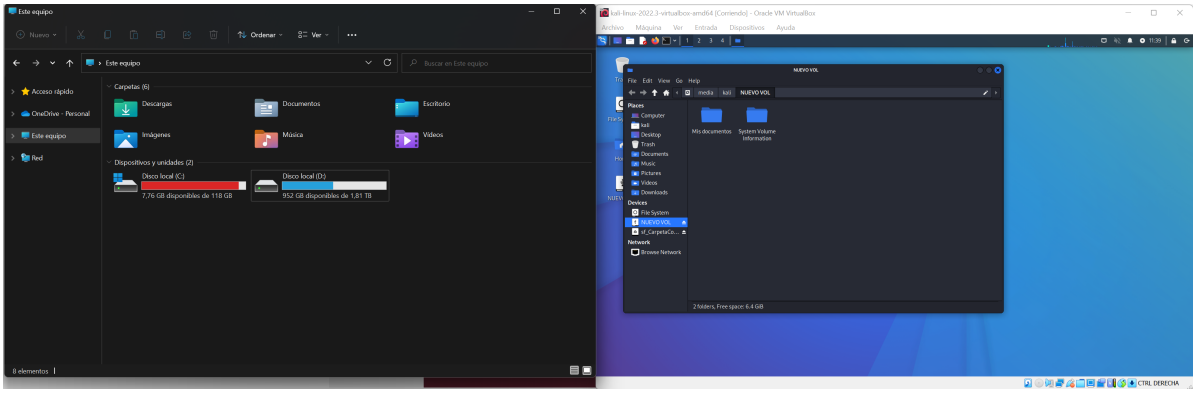


Figura 6.4: *La imagen de la izquierda corresponde al explorador de archivos de Windows 11 (máquina real), y la imagen de la derecha corresponde al sistema operativo Kali Linux (máquina virtual).*

6.9. Análisis del código

La mayor parte del código de esta herramienta está escrito en el lenguaje Go, que es un lenguaje de programación propio de Google pero basado en el lenguaje de programación C, teniendo una sintaxis muy similar, aunque también hace uso de librerías de Python para algunas funciones en concreto. A continuación, se muestran varios fragmentos de código pertenecientes a los ataques de teclado y ratón. Debido a que los archivos donde se encuentran estos fragmentos son bastante extensos, teniendo algunos más de 800 líneas de código, se ha optado por escoger los fragmentos que consideramos que son más interesantes o importantes en la ejecución de dichos ataques.

En la imagen superior vemos la función que crea un nuevo dispositivo de tipo ratón / mouse, creando un archivo que recogerá todos los eventos que produzca (en ambas direc-

```go
func NewMouse(devicePath string) (mouse *Mouse, err error) {
        //ToDo: check existence of deviceFile (+ is writable)
        mouse = &Mouse{
                devicePath: devicePath,
        }

        mouse.deviceFile, err = os.OpenFile(devicePath, os.O_WRONLY|os.O_CREATE|os.O_TRUNC, os.ModePerm)
        if err != nil {
                return nil,err
        }
        return
}
```

Figura 6.5: *Imagen correspondiente a la función para crear un nuevo dispositivo de ratón. [1]*

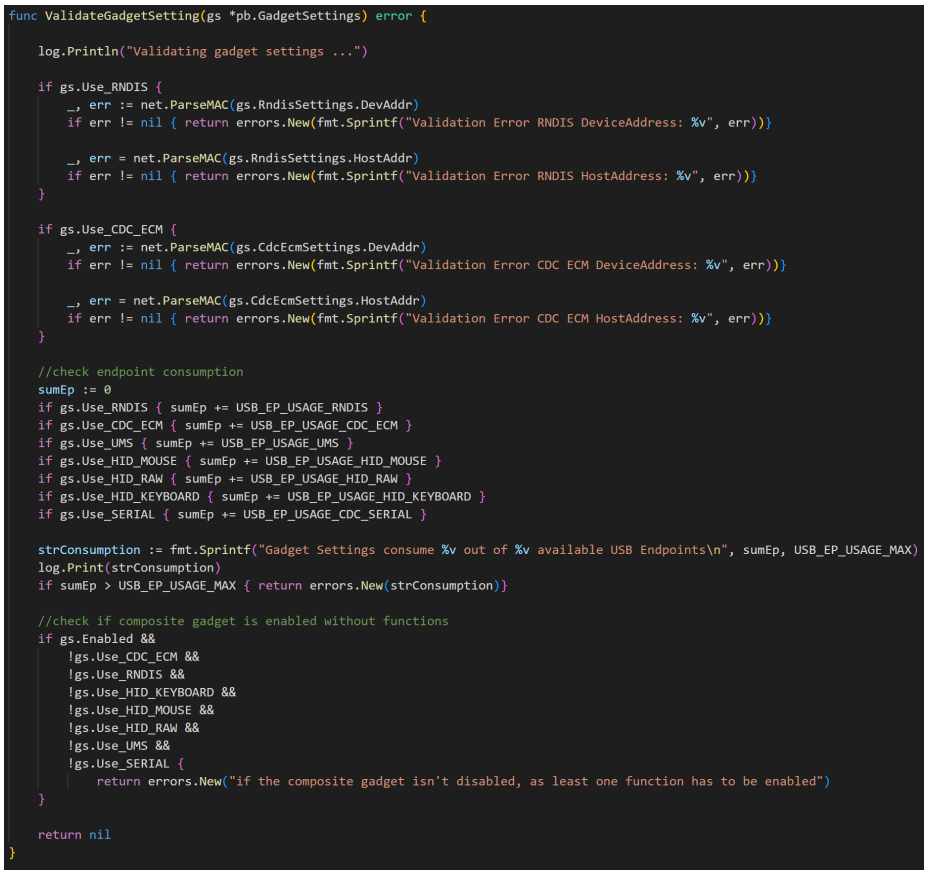
ciones, de la raspberry hacia el ordenador atacado (inputs) y del ordenador atacado a la raspberry (outputs)), con los permisos necesarios para su correcta ejecución.

```
func (bt *BtService) DeployBluetoothNetworkService(btNwSvc *pb.BluetoothNetworkService) (err error) {
	uuid := toolz.UUID_NETWORK_SERVER_NAP
	switch btNwSvc.Type {
	case pb.BluetoothNetworkServiceType_NAP:
		uuid = toolz.UUID_NETWORK_SERVER_NAP
	case pb.BluetoothNetworkServiceType_PANU:
		uuid = toolz.UUID_NETWORK_SERVER_PANU
	case pb.BluetoothNetworkServiceType_GN:
		uuid = toolz.UUID_NETWORK_SERVER_GN
	}
	if btNwSvc.ServerOrConnect {
		// start server for given network service
		if btNwSvc.RegisterOrUnregister {
			return bt.RegisterNetworkServer(uuid)
		} else {
			return bt.UnregisterNetworkServer(uuid)
		}
	} else {
		//(dis)connect from/to given network network service of given remote device

		if btNwSvc.RegisterOrUnregister {
			// register == connect
			return bt.ConnectNetwork(btNwSvc.MacOrName, uuid)
		} else {
			// unregister == disconnect
			return bt.DisconnectNetwork(btNwSvc.MacOrName)
		}
	}
}
```

Figura 6.6: *Imagen correspondiente a la función de desplegar una conexión Bluetooth. [2]*

En la imagen superior vemos la función que despliega una conexión Bluetooth, la cual primero comprueba qué tipo de servidor necesita crear y luego comprueba si la petición es para registrar un nuevo servidor o eliminarlo o si se trata de una nueva conexión al servidor o eliminarla.



```
func ValidateGadgetSetting(gs *pb.GadgetSettings) error {

    log.Println("Validating gadget settings ...")

    if gs.Use_RNDIS {
        _, err := net.ParseMAC(gs.RndisSettings.DevAddr)
        if err != nil { return errors.New(fmt.Sprintf("Validation Error RNDIS DeviceAddress: %v", err))}

        _, err = net.ParseMAC(gs.RndisSettings.HostAddr)
        if err != nil { return errors.New(fmt.Sprintf("Validation Error RNDIS HostAddress: %v", err))}
    }

    if gs.Use_CDC_ECM {
        _, err := net.ParseMAC(gs.CdcEcmSettings.DevAddr)
        if err != nil { return errors.New(fmt.Sprintf("Validation Error CDC ECM DeviceAddress: %v", err))}

        _, err = net.ParseMAC(gs.CdcEcmSettings.HostAddr)
        if err != nil { return errors.New(fmt.Sprintf("Validation Error CDC ECM HostAddress: %v", err))}
    }

    //check endpoint consumption
    sumEp := 0
    if gs.Use_RNDIS { sumEp += USB_EP_USAGE_RNDIS }
    if gs.Use_CDC_ECM { sumEp += USB_EP_USAGE_CDC_ECM }
    if gs.Use_UMS { sumEp += USB_EP_USAGE_UMS }
    if gs.Use_HID_MOUSE { sumEp += USB_EP_USAGE_HID_MOUSE }
    if gs.Use_HID_RAW { sumEp += USB_EP_USAGE_HID_RAW }
    if gs.Use_HID_KEYBOARD { sumEp += USB_EP_USAGE_HID_KEYBOARD }
    if gs.Use_SERIAL { sumEp += USB_EP_USAGE_CDC_SERIAL }

    strConsumption := fmt.Sprintf("Gadget Settings consume %v out of %v available USB Endpoints\n", sumEp, USB_EP_USAGE_MAX)
    log.Print(strConsumption)
    if sumEp > USB_EP_USAGE_MAX { return errors.New(strConsumption)}

    //check if composite gadget is enabled without functions
    if gs.Enabled &&
        !gs.Use_CDC_ECM &&
        !gs.Use_RNDIS &&
        !gs.Use_HID_KEYBOARD &&
        !gs.Use_HID_MOUSE &&
        !gs.Use_HID_RAW &&
        !gs.Use_UMS &&
        !gs.Use_SERIAL {
            return errors.New("if the composite gadget isn't disabled, as least one function has to be enabled")
    }

    return nil
}
```

Figura 6.7: *Imagen correspondiente a la función para comprobar las opciones del dispositivo. [3]*

En la imagen superior vemos la función que comprueba las opciones de las que dispone el dispositivo. Primero comprueba qué tipo de conexión usa, RNDIS[1] o CDC ECM[2], lanzando los correspondientes errores en caso de que la conexión falle. En caso de que disponga de más opciones de las que debería tener como máximo, lanza un error para informar de que está consumiendo un determinado número de endpoints por encima del máximo permitido. Por último, comprueba si el dispositivo está habilitado pero no dispone de ninguna función que pueda usar, caso en el que lanzará un error informando de que, si el dispositivo no está deshabilitado, debe tener al menos una función activada.

[1]Remote Network Driver Interface Specification, es un protocolo propio de Microsoft usado en la mayoría de USB que proporciona un enlace Ethernet virtual a la mayoría de sistemas operativos Windows y Linux.

[2]Communications Device Class – Ethernet Control Module subclass, es un driver del USB que se encarga de enviar y recibir paquetes Ehternet a través de USB.

6.10. Conclusión del exploit

Este exploit, al estar incorporado en una distribución de Kali Linux y su instalación es tan sencilla, hace que cualquier usuario con nociones básicas de programación o de instrucciones propias de Windows pueda adentrarse en el mundo de la ciberseguridad o del hacking de manera fácil ya que, como hemos explicado anteriormente, si se utiliza la versión GUI, es realmente intuitiva y lo suficientemente potente para realizar pequeños ataques o, simplemente, investigar para iniciarse en el campo, mientras que si se hace uso de la versión SSH, hay que tener un conocimiento más avanzado en torno a los comandos e instrucciones tanto de Linux como de Windows para poder llevar a cabo los ataques deseados.

Capítulo 7

Password Thief, exploit Responder

En este capítulo explicaremos el exploit Responder, ya que lo hemos investigado de cara a enfrentarnos a un exploit que aprovecha otro tipo de vulnerabilidad.

No hemos conseguido investigar este exploit tanto como nos hubiese sido deseado debido a que hemos dedicado la mayor parte del tiempo a entender como funciona el protocolo USB y el estándar HID, para comprender detalladamente el funcionamiento del exploit P4wnP1, que posteriormente tuvimos que adaptarnos al P4wnP1 ALOA por los motivos explicados anteriormente. A pesar de ello, vemos apropiado exponer los resultados que hemos obtenido a lo largo de nuestra pequeña investigación sobre este exploit, debido a que conseguimos ejecutarlo mediante Kali Linux, haciendo uso de un controlador de dominio, pero esa no era finalidad de este trabajo, ya que no estábamos usando la Raspberry, pese a que el funcionamiento sería algo parecido.

7.1. Funcionamiento del exploit

La Raspberry Pi, con la ejecución del exploit, funciona como un proxy falso, que envenena las comunicaciones DNS y DHCP para responder a las búsquedas de configuración de WPAD [1]. WPAD está habilitado por defecto en Windows, pero también es compatible con diferentes sistemas operativos. Los navegadores, tanto internos como web, transmiten las solicitudes del

[1]Web Proxy Auto Discovery, método utilizado por clientes para encontrar la URL de un archivo de configuración utilizando métodos de descubrimiento de DHCP y/o DNS

wpad.local , mientras que el exploit funciona de intermediario, capturando las credenciales. La Raspberry se hace pasar por WPAD gracias a que tiene un servidor DHCP, por tanto, esto aumenta las posibilidades de ataque. El proxy falso, envenena las comunicaciones, y se mantiene a la escucha de los hashes. Estos hashes aparecen en la pantalla o se puede cambiar la configuración de Responder.conf para establecer la ruta de guardado que, por defecto, se encuentran en la ruta *responder/logs/HTTP-NTLMv2–192.168.2.1.txt*.

7.2. Tutorial de instalación

Este exploit se encuentra instalado por defecto en el sistema operativo de Kali Linux, aunque es aconsejable usar el descargable que podemos encontrar en el GitHub[5] , ya que nos facilita tanto la personalización de las diferentes herramientas como de la configuración en la cual podemos indicar que protocolo activar o incluso indicar las IPs específicas a las que vamos escuchar y hacer de intermediario.

En la Raspberry Pi, los pasos a seguir son más complejos, debido a que debemos configurar las redes, en primer lugar debemos instalar el protocolo DHCP, y configurarlo para que use de interfaz el USB entre otras configuraciones. Todo estos pasos vienen redactados y explicados por un Arquitecto de Infraestructura en un artículo [14].

7.3. Materiales Necesarios

A continuación se lista lo necesario para instalar el exploit.

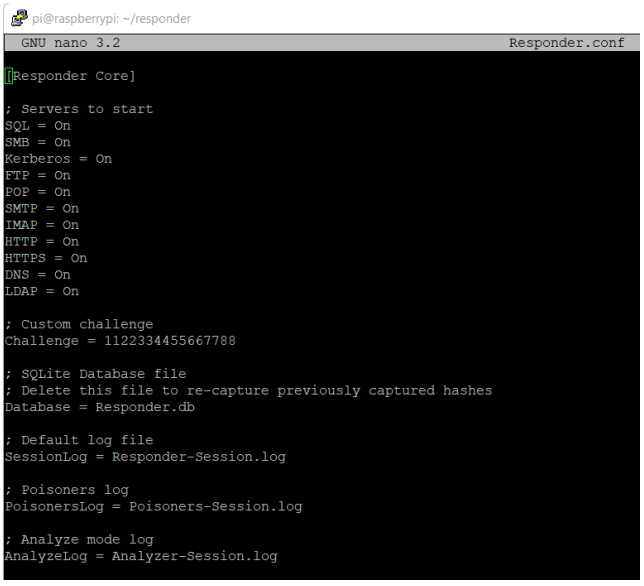
- `Raspberry Pi Zero 0 W`
- `Cable microUSB`
- `Imagen del framework Responder`
- `Tarjeta de memoria microSD (recomendable de 16 GB de capacidad)`

- `Software para escribir la imagen en la tarjeta (Recomendamos Ether y Raspberry Pi images)`

7.4. Opciones de configuración

En el archivo Responder.conf del exploit, se puede configurar y personalizar el exploit a nuestro gusto: desde poder activar y desactivar los servers (los cuales se lanzaran al ejecutar el exploit), como especificar que IP o rango de IPs son a las que vamos a permanecer a la escucha para obtener las credenciales como al directorio que vamos a redirigir las credenciales obtenidas.

En la siguiente imagen, podemos apreciar algunos de los campos que podemos modificar en Responder.conf:



A continuación, mostraremos los comandos del exploit para su ejecución.

```
--version             show program's version number and exit
-h, --help            show this help message and exit
-A, --analyze         Analyze mode. This option allows you to see NBT-NS,
                      BROWSER, LLMNR requests without responding.
-I eth0, --interface=eth0
                      Network interface to use
-b, --basic           Return a Basic HTTP authentication. Default: NTLM
-r, --wredir          Enable answers for netbios wredir suffix queries.
                      Answering to wredir will likely break stuff on the
                      network. Default: False
-d, --NBTNSdomain     Enable answers for netbios domain suffix queries.
                      Answering to domain suffixes will likely break stuff
                      on the network. Default: False
-f, --fingerprint     This option allows you to fingerprint a host that
                      issued an NBT-NS or LLMNR query.
-w, --wpad            Start the WPAD rogue proxy server. Default value is
                      False
-u UPSTREAM_PROXY, --upstream-proxy=UPSTREAM_PROXY
                      Upstream HTTP proxy used by the rogue WPAD Proxy for
                      outgoing requests (format: host:port)
-F, --ForceWpadAuth   Force NTLM/Basic authentication on wpad.dat file
                      retrieval. This may cause a login prompt. Default:
                      False
--lm                  Force LM hashing downgrade for Windows XP/2003 and
                      earlier. Default: False
-v, --verbose         Increase verbosity.
```

Figura 7.1: *Distintas comandos del exploit.[9]*

7.5. Estudio proceso de ataque

Tal y como se puede apreciar en la Figura 7.1, este exploit nos da la posibilidad de numerosas formas de ataque. Además, también nos ofrece el modo Análisis, que nos permite ver las diferentes solicitudes sin respuesta.

Como hemos explicado anteriormente, en este exploit nos hemos topado con algunos inconvenientes que no hemos podido solucionar por la falta de tiempo para dedicar a su investigación, pero describiremos cómo debe funcionar el ataque, ya que en el sistema operativo Kali Linux (mediante uso de una máquina virtual) hemos conseguido que se ejecute correctamente. Las imágenes mostradas en este capítulo son todas de Raspberry a excepción de la última, que se muestra el hash, que ha sido tomada directamente desde Kali Linux.

La funcionalidad de este exploit consiste en aprovechar las vulnerabilidades de una red interna, para imitar servicios como LLMNR [2], NBT-NS [3] y MDNS DNS de multidifusión,

[2]Resolución de nombre de multidifusión local de enlace
[3]Servicio de nombre de NetBIOS

lo cual, gracias a ello, obtienen los credenciales. Una vez ejecutado el comando básico del exploit, es decir, *sudo ./Responder.py -I usb0*, se ejecutara el exploit tal y como muestra la imagen, con una pequeña interfaz que nos muestra qué opciones están habilitadas y cuáles no, mientras se mantiene a la escucha de los eventos.

Debido a que Raspberry Pi se conecta a través del USB, la interfaz previamente configurada, tal y como nos indica el tutorial seguido en el apartado de la instalación[14], debemos de indicar dicha interfaz, en lugar de eth0 (que es la que suele estar configurada por defecto en Linux).

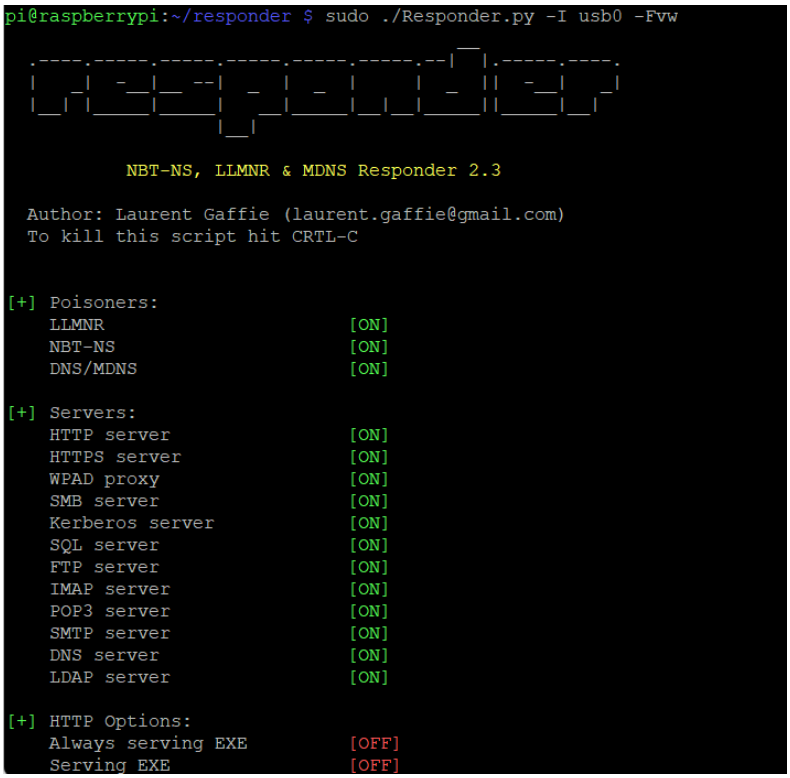
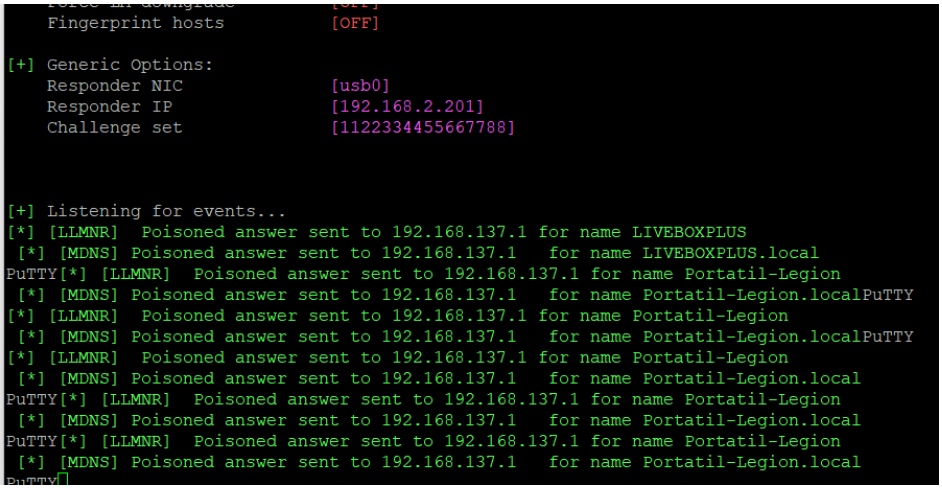
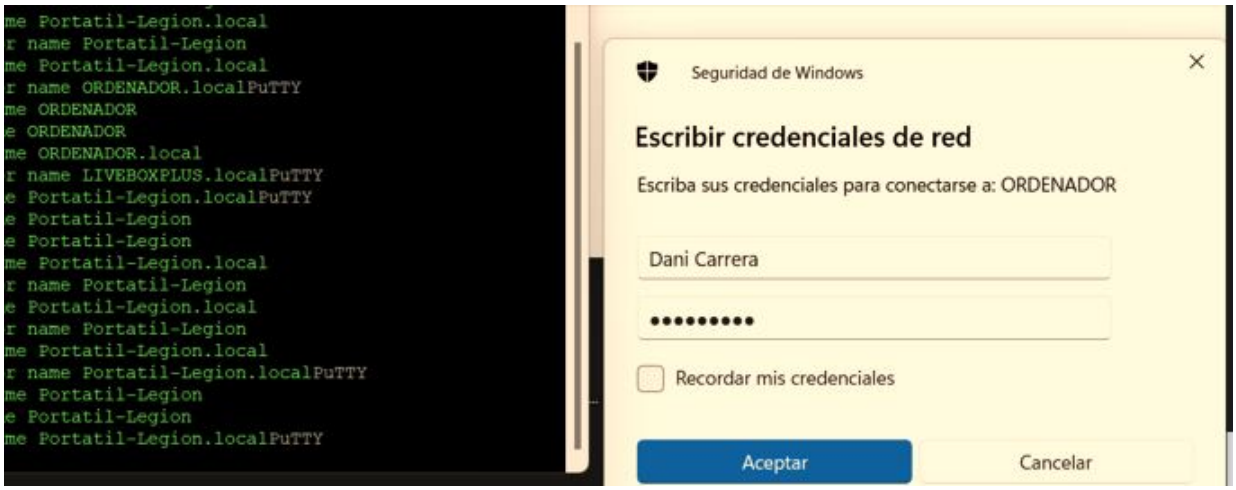


Figura 7.2: *Ejemplo de ejecución*

A continuación, se mantiene a la escucha de los eventos de las IPs o rango de IPs que hayamos configurado, tal y como se puede apreciar en la siguiente imagen.



Lo que hace el exploit Responder con la opción w, es crear un servidor WPAD falso, así cuando la víctima intenta obtener el archivo *wpad.dat*, es decir, cuando el DNS intenta obtener el archivo de configuración especial para establecer automáticamente su configuración de proxy, momento en que el exploit le pide una pantalla de autentificación con el nombre y contraseña del usuario que se encuentra en el dominio.



Al enviar las credenciales, el exploit las obtiene encriptadas en un hash. Este hash se guarda en un archivo de texto con formato .txt (en función de como hallamos configurado la ruta de destino en las opciones). En este punto, es en el que la Raspberry no obtiene respuesta del evento, mientras que en Kali Linux, atacando a un Windows 10 que hace de controlador de dominio, si interviene las credenciales obteniendo como resultado un hash en el momento que intentamos acceder a la ruta \\pintserver en el directorio de archivos. Tal y como podemos apreciar en la siguiente figura.

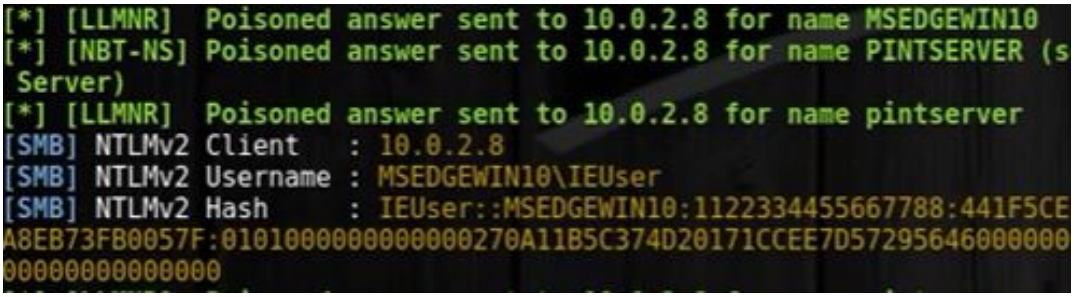


Figura 7.3: *Ejemplo escucha*

Una vez obtenido este hash, lo hemos desencriptado mediante la herramienta hashcat para que, una vez consiga desencriptarlo, devuelve la contraseña que tienen por defecto las máquinas de Windows "*Pasw0rd!*" .

7.6. Posible Defensa al ataque

Una solución simple pero efectiva, como en la mayoría de estos ataques, es la impenetrabilidad a un puerto USB o a un puerto de la red interna (en caso de que se use Kali Linux) de una persona u objeto desconocido. Pero en la práctica esto es prácticamente imposible de cumplir. Por tanto, de manera técnica se podría solucionar desactivando los servicios NBNS y LLMNR, que son los que se usan en la ejecución del exploit para envenenar, además de una entrada WPAD que apunte al proxy corporativo para no sufrir desviaciones indebidas.

7.7. Conclusión del exploit

A pesar de no haber podido dedicar el tiempo que hubiesemos querido en investigar el por qué a través de Kali Linux si se interviene en las DNS mientras que en la Raspberry Pi no, nos parece interesante hacer una breve descripción sobre lo que habiamos investigado debido a que es un exploit bastante cómodo de usar y efectivo para redes internas, que son las que se usan frecuentemente en las empresas. De ahí que esta herramienta/exploit sea bastante frecuente en las pruebas de Pentesting. A pesar de que con un cambio en las políticas o desactivando NBNS y LLMNR y creando una entrada WPAD que apunte al proxy corporativo, se sabe que muchas empresas medianas o pequeñas, no gastan recursos económicos en la seguridad informática de la empresa, quedando así vulnerables a varios ataques.

Capítulo 8

Conclusión

Con este trabajo hemos llegado a varias conclusiones. La primera es que, con la información adecuada y siguiendo los tutoriales que se encuentran en internet (tanto en GitHub como en foros de informática o páginas web, como las que hemos añadido en la bibliografía), puede no llegar a ser muy complicado probar determinados tipos de ataques, ya que no requieren amplios conocimientos sobre programación y son altamente efectivos, como es el caso del exploit P4wnP1 ALOA que, como hemos explicado anteriormente, sólo requiere montar la imagen en la tarjeta micro SD, introducirla en la Raspberry y conectarla a cualquier dispositivo, mientras que es bastante complicado entender su funcionamiento y poder defenderse ante los ataques, porque es absolutamente necesario analizar el código del exploit en profundidad.

La segunda conclusión que sacamos es el gran problema que supone este tipo de ataques, que se encuentran tanto las empresas y los organismos educativos como los usuarios. Al ser ataques que pueden simular ser un teclado o un ratón (dispositivos que hay que conectar por USB al ordenador en la mayoría de casos), es difícil defenderse, ya que el ordenador detecta el dispositivo como uno de los mencionados para poder hacer uso de sus funciones e interactuar con el humano.

Pero para que estos ataques tengan éxito, principalmente depende de una acción física (introducir el dispositivo atacante en el puerto USB). Por consecuente, una empresa se puede

defender de estos ataques simplemente no permitiendo interactuar con estos puertos, pero en la práctica puede llegar a no ser productivo, ya que siempre que se quiera hacer algo tan sencillo como introducir el USB de los auriculares para las reuniones de Microsoft Teams (algo cada vez más frecuente debido al teletrabajo), habría que solicitarlo a alguien con credenciales de administrador para asegurarse de que en el puerto USB no se ha introducido un dispositivo atacante.

Por tanto, este ataque también depende del desconocimiento de las personas sobre él. Hicimos la prueba tanto en las respectivas bibliotecas públicas de nuestra zona como en la de la facultad de informática, de dejar un USB normal y corriente, el cual no tenía ningún exploit o malware para evitar problemas legales, en la mesa en la que se encontraba uno de nosotros e irnos. Tarde o temprano alguien lo cogía e introducía en su portátil para ver qué contenía el dispositivo. Esto sucedió tanto en las bibliotecas públicas como en nuestra facultad.

En resumen, son ataques efectivos y difíciles de defender que pueden afectar a múltiples dispositivos dirigidos tanto a organizaciones como a un usuario individual.

8.1. Tareas realizadas por Daniel

Los primeros meses, nos dedicamos en la investigación sobre el tema tratado en todo el proyecto, ya que al ser un tema ajeno a lo que estabamos acostumbrados, teníamos que familiarizarnos. Estas primeras tomas de contacto se basaron en investigar qué tipos de ataques había, cómo funcionaban y los distintos exploits y la forma en la que se llevan a cabo para explotar las vulnerabilidades.

Después, realicé las distintas tareas de redacción de los primeros capítulos del trabajo, algunas de forma conjunta con Bruno y otras de forma personal, ya que era necesario que ambos supiéramos cómo funciona el protocolo USB, y el estándar HID.

Una vez acabada la parte de investigar y redactar lo mencionado anteriormente, nos encontramos con el problema de montar las imágenes en la Raspberry Pi, algo con lo que tampoco estábamos familiarizados. Después de varios intentos, con imágenes diferentes descargadas desde la página web oficial, no conseguíamos que se estableciera la comunicación SSH con éxito. Probamos varios tutoriales, algunos de ellos incompletos u otros anticuados. Por ejemplo, en los tutoriales se mencionaba el uso de la imagen Jessie Lite, imagen que, a partir de la versión del año 2017 ,no funcionaba la conexión SSH. Por eso, llegamos a la conclusión que introducir un pequeño tutorial en este trabajo.

También llevé a cabo la investigación de varios exploits, la primera de manera conjunta a Bruno, la de P4wnP1 que, a pesar de varios intentos fallidos, al ver que no conseguiamos los objetivos, decidimos repartirnos en la investigación de diferentes exploits.

A pesar de no estar incluídos en el trabajo, investigué sobre numerosos exploits como es el caso de Ethsploiter, whid injector, entre otros, y P4wnP1 ALOA, el cual finalmente nos decantamos por analizar de manera conjunta. Una vez avanzada la investigación, llegamos a la conclusión de separar las tareas, en las que Bruno se ocupaba de seguir analizando el

P4wnP1 ALOA mientras que yo me dedicaba a completar y redactar la memoria a través de la herramienta online Overleaf. Una vez avanzada la redacción, me dediqué a investigar sobre otro exploit para así hacer más extenso el trabajo. Finalmente, nos decantamos por el exploit Responder, debido a que usaba vulnerabilidades en redes. Al no recordar con certeza los conceptos sobre redes y protocolos, tuve que investigar y familiarizarme más con ellos. Realicé lecturas en diversos foros, revistas y demás medios de información sobre este tipo de exploits que aprovechan estas vulnerabilidades. Un recurso que utilicé bastante para informarme fueron los libros Pentesting con Kali y Python para Pentesting de 0xWord, llegando incluso a utilizarlas como fuentes y citas, tal y como se ve refllejada en la bibliogfrafía, ya que tenían apartados exclusivos de vulnerabilidades en IPv4.

Una vez con las bases más establecidas, realicé el estudio y explotación del exploit Responder buscando una posible solución para la vulnerabilidad que aprovecha. Tal y como se puede ver en el capítulo 7. Además de ayudar a mi compañero, con el exploit P4wnP1 ALOA, en el entendimiento del código y ver que vulnerabilidades aprovecha.

En resumen, considero que este trabajo ha sido realizado de manera conjunta por ambas partes a partes iguales.

8.2. Tareas realizadas por Bruno

Cuando empezamos el proyecto, como no estábamos familiarizados con el mundo de la ciberseguridad y, en este caso, el pentesting, ambos dedicamos bastante tiempo en buscar información sobre el USB, para poder enteder cómo funcionaba y qué sucedía en los ordenadores al conectar un dispositivo USB, así como buscar información acerca de una lista de posibles herramientas que nos proporcionó nuestro tutor. Una vez asentamos las bases del proyecto, comenzamos a decidir qué herramientas usaríamos, dividiéndonos dicha lista a partes iguales para avanzar con más rápidez.

Una vez decidimos con qué herramientas quedarnos, empezamos a probar juntos el exploit P4wnP1, con el cual tuvimos bastantes problemas para poder configurarlo inicialmente, desde no poder montar correctamente la imagen del exploit en la Raspberry, debido a que, dependiendo del programa que usáramos, nos configuraba el exploit de forma predeterminada con unos ajustes u otros, hasta configurar la conexión a internet de la Raspberry una vez montada la imagen, la cual se perdía cada vez que se reiniciaba la Raspberry, teniendo que volver a configurarla cada vez, haciendo que tuvieramos que dedicarle bastante tiempo. En este punto, decidimos que yo iba a continuar probando este exploit y mi compañero Daniel investigaría otros, para no quedarnos atascados los dos con el mismo problema y poder avanzar.

Después de conseguir configurarla correctamente, empecé a probar con el ataque P4wnP1 LockPicker Windows 10, pero después de varios intentos fallidos, investigué sobre dicho ataque, encontrando que Microsoft había parcheado la vulnerabilidad afectada con una actualización hace unos años. Como no podía seguir probando con este exploit, decidí probar su sucesor, P4wnP1 ALOA, el cual si que se ejecutaba sin problemas. Por lo tanto, nos decantamos por analizar e investigar juntos este exploit, realizando varias pruebas de ataques sencillos e inofensivos para el ordenador atacado.

Teniendo claro el funcionamiento de este exploit, decidimos repartirnos las tareas para intentar complementar el trabajo con otro exploit que aprovechara otro tipo de vulnerabilidad. Por tanto, yo me encargué de realizar varias pruebas grabadas en vídeo (del exploit P4wnP1 ALOA) de varios de esos ataques, como mover el cursor del ordenador atacado sin tocar el touchpad, o bloquear la sesión del usuario para luego introducir la contraseña para acceder al ordenador, así como probar algunos ataques con la ayuda de mi compañero Daniel. Además con ayuda de mi compañero, analizamos el código de GitHub y comencé la creación de un keylogger propio para seguir realizando pruebas, aunque no se ha podido implementar finalmente debido a que vimos más importante investigar otro exploit.

Ya habiendo acabado de investigar y probar sobre P4wnP1 ALOA, ayudé a mi compañero con su investigación de otros exploits, proporcionándole una lista con posibles exploits para probar si podían funcionar de alguna manera conjunta o independiente con P4wnP1 ALOA.

Los últimos días, nos repartimos los diferentes puntos restantes de esta memoria para redactar de manera conjunta y avanzar con más rapidez, comprobando lo que había escrito cada uno y aportando ideas y cambios para mejorar.

Además, ayudé a mi compañero con el exploit Responder, para intentar encontrar una solución para el problema que nos habíamos encontrado.

Para acabar, creo que hemos trabajado a la par en este proyecto, tanto a la hora de investigar como en la parte de pruebas, aportando cada uno al trabajo del otro algún detalle que no se había contemplado.

Capítulo 9

English

9.1. Introduction

In this project we will research about HID attacks. We will make a brief introduction of the history from the USB port to how the HID standard works. Then, we will focus on the different types of attacks and the use of the same attack in different environments, all of them using a Raspberry Pi Zero W as the attacker tool. We will take a closer look at developed exploits in the following chapters.

This project is composed of a part of research on how the USB port works, the several HID attacks and the multiple options to host them.

In addition to understanding how these exploits work once the victim is attacked in multiple environments, we will provide potential solutions to avoid the attack, as well as demonstrations of those and several installation tutorials for the correct use of the solutions.

9.2. Motivation

What has led us to choose this topic was the great lack of knowledge about this type of attack, to understand the serious danger that it poses both for an common user and for a company, in addition to the fact that we wanted to get started in the world of cybersecurity and this project has been the opportunity to carry it out. The danger we are talking about is something as common as a simple flash memory, a device that nobody would think could

cause us serious problems if we connected it to our personal computers, but there are people who unconsciously connect these memories without thinking about possible consequences. This action is carried out even by computer science students, as reflected in the experiment done in the library of the Faculty of Computer Science, an experiment which we refer to in the conclusion of this work. This made us wonder why something that has been incorporated into computers since 1996, has still not been corrected 26 years later.

9.3. Project objectives

The objective of this Final Degree Project is to *study the existing tools (P4wnP1 A.L.O.A, Ethsploiter,...) to attack operating systems*, put them to the test on various systems, to later offer solutions to avoid the vulnerabilities they exploit. To do this, we will start with a Raspberry Pi Zero W, a micro SD card in which load the tool in question and a micro USB cable to connect the Raspberry Pi Zero W to any device.

The objectives at the personal learning level are the following:

- Confront ourselves during the development of the work with concepts and techniques that we have not studied in the degree and they are not as familiar to us (as concepts focused on programming), in order to expand our knowledge and skills.
- Discover and use different new tools that help us learn by first-hand the ability to exploit vulnerabilities in operating systems.

9.4. Work plan

In the first place, we will become familiar with concepts within pentesting[1], as they are unfamiliar to what we have studied in the degree. We will research the different types of HID attacks, as well as the possible victims. We will carry out studies of the different types of vulnerabilities to be exploited through the USB port, as well as we will do a research of

[1]Technique that consists of attacking different environments or systems with the aim of detecting and preventing possible failures.

several exploits that we will be able to analyze in this work, to finally choose one of them to analyze and study. Also, we will get familiar with the Raspberry Pi Zero W, because that is what we will use to inject the exploits.

We will study the selected exploits analyzing how it works, some possible solution, in which operating system they are more effective... We will develope this on chapters number 5, 6 and 7.

9.5. Conclusion

In this work we have reached several conclusions. The first conclusion is that, with the right information and by following the tutorials that can be find in the internet (in GitHub and also in computing forums or web pages, like the pages we included in the bibliography), it can not be difficult to test some attacks, because they does not require extensive knowledge about programming and they have very high effectiveness, like P4wnP1 ALOA that, as we explained above, it only require to mount the image of the exploit in the micro SD card, put it on the Raspberry and connect it to any device, but is more complicated to understand the performance and could defend to the attacks, because it is absolutely necessary to analyze the exploit code in deep.

The second conclusion is that this is the biggest problem that companies and educational organizations as well as users face. As they are attacks that can pretend to simulate several devices such as a keyboard or a mouse (devices that have to connect to the USB port of the computer in most of the cases), is hard to defend against them, because the computer detect the device as one of the devices mentioned for use them and interact with the human.

But for these attacks to be successful, first of all it depends on a physical action (inserting the attacking device into the USB port). Consequently, a company can defend itself against these attacks simply by not allowing interaction with these ports, but when it comes to practice it can become unproductive, because whenever you want to do something as simple as inserting the USB of the headphones for Microsoft Teams meetings (something that is becoming more frequent due to teleworking), someone with administrator credentials would have to be requested to ensure that an attacking device has not been inserted into the USB port.

Therefore, this attack also depends on people's ignorance about it. We did the test both the respective public libraries in our area, as well as in the computer science faculty. We left a normal abandoned USB, which have not an exploit or malware to avoid legal problems, on the table where one of us was working and leave. Sooner or later someone did pick it

up and insert it into their laptop to see what the device contained. This happened both in public libraries and in our faculty.

In conclusion, those attacks are an effective and difficult to defend that can affect multiple devices targeted both to organizations and an individual user.

Bibliografía

[1] Código extraido. https://github.com/RoganDawes/P4wnP1_aloa/blob/master/hid/mouse.go, líneas 34 a 45.

[2] Código extraido. https://github.com/RoganDawes/P4wnP1_aloa/blob/master/service/bluetooth.go, líneas 159 a 187.

[3] Código extraido. https://github.com/RoganDawes/P4wnP1_aloa/blob/master/service/SubSysUSB.g líneas 207 a 262 (modificado para eliminar los comentarios).

[4] Descripción usb obtenida de. https://concepto.de/usb/ , accedido 2022.

[5] Github del exploit responder. https://github.com/lgandx/Responder , accedido 2022.

[6] Imagen obtenida de la documentación oficial,capítulo device class definition for human interface devices (hid). https://www.usb.org/sites/default/files/hid1_11.pdf , accedido 2022.

[7] Imagen obtenida de la página. https://www.tiendatec.es/raspberry-pi/gama-raspberry-pi/1130-raspberry-pi-zero-w-8472496016155.html , accedido 2022.

[8] Imagen obtenida de la siguiente url. https://www.malwarebytes.com/blog/news/2015/04/new-malwarebytes-anti-exploit-version-is-out, accedido 2022.

[9] Imagen obtenida de las opciones del exploit en el github. https://github.com/SpiderLabs/Responder , accedido 2022.

[10] Información obtenida de la documentación oficial. https://www.usb.org/document-library/device-class-definition-hid-111 , accedido 2022.

[11] Información sacada de la wiki de p4wnp1. https://p4wnp1.readthedocs.io/en/latest/ , accedido 2022.

[12] P4wnp1 aloa. https://github.com/RoganDawes/P4wnP1_aloa, accedido 2022.

[13] Página oficial raspberry pi. https://www.raspberrypi.com/products/raspberry-pi-zero-w/ , accedido 2022.

[14] Tutorial de instalación del exploit responder en raspberry pi. https://medium.com/codex/raspberry-pi-zero-password-thief-cb2bea8d6dc0 , accedido 2022.

[15] Ran Yahalom y Yuval Elovici Nir Nissim. Usb-based attacks. 2017. https://www.sciencedirect.com/science/article/pii/S0167404817301578.

[16] Germán Sánchez Garcés y Jose Miguel Soriano de la Cámara Pablo Gónzalez Pérez. Pentesting con kali. 3º Edición 2020. Definiciones y descripciones basadas en el Capitulo IV.

[17] Germán Sánchez Garcés y Jose Miguel Soriano de la Cámara Pablo Gónzalez Pérez. Pentesting con kali. 3º Edición 2020. Definiciones y descripciones basadas en el Capitulo III.