Bash Bunny

¿Qué es Bash Bunny?

Bash Bunny es un dispositivo de ataque USB fabricado por una empresa estadounidense Hak5. A primera vista, el dispositivo parece una unidad USB USB innecesariamente gorda. Sin embargo, si echaras un vistazo dentro, encontrarás que a pesar de su tamaño, contiene un ejemplo de hardware muy respetable por un procesador ARM de cuatro núcleos:

- Quad-core ARM Cortex A7
- 32 K L1/512 K L2 Cache
- Memoria DDR3 de 512 MB

- 8 GB SLC NAND Disk

El hardware cobra vida gracias a Linux, en concreto Debian 8 Jessie.

Si puedes acceder físicamente a un dispositivo, el **Hak5 Bash Bunny** te dará acceso electrónico. En pocas palabras: es la plataforma de ataque USB más potente del mundo.

En detalle, es una multiplataforma, multicarga, multiherramienta capaz de emular y abusar simultáneamente de dispositivos de confianza - dispositivos de entrada, dispositivos de almacenamiento, dispositivos de red.

Disfrazado como una unidad USB normal, infinitamente configurable, y respaldado por el repositorio de carga útil Hak5, el Bash Bunny es una herramienta de hacking físico todo en uno.

Rubber Ducky vs Bash Bunny

Similitudes entre Rubber Ducky y Bash Bunny:

1. **Fabricante**: Ambas herramientas son desarrolladas por **Hak5**, enfocadas en pruebas de penetración y hacking ético.
2. **Automatización**: Ambas permiten automatizar tareas mediante scripts, simulando la interacción de un teclado o una conexión USB.
3. **Plataforma portátil**: Son dispositivos compactos y fáciles de transportar, diseñados para simular ataques físicos.
4. **Uso de scripts**: Utilizan lenguajes de scripting fáciles de aprender:Rubber Ducky: Duckyscript.Bash Bunny: Bash y lenguajes similares a scripts de Linux.
5. **Objetivo**: Sirven para pruebas de seguridad, como extracción de datos, instalación de backdoors o recolección de credenciales.

Diferencias principales:

Rubber Ducky	Bash Bunny
Función principal: Simula un teclado USB para enviar comandos rápidamente al sistema objetivo.	**Función principal**: Simula varios dispositivos USB (teclado, almacenamiento, adaptador Ethernet, etc.).
Lenguaje de scripting: Duckyscript, más básico y directo.	**Lenguaje de scripting**: Basado en Bash, con mayor flexibilidad y complejidad.
Versatilidad: Limitado a ataques basados en teclado.	**Versatilidad**: Puede ejecutar ataques más avanzados y múltiples funciones simultáneamente.
Almacenamiento: Capacidad de almacenamiento limitada.	**Almacenamiento**: Mayor capacidad, con soporte para múltiples payloads.
Potencia de hardware: Hardware más simple.	**Potencia de hardware**: Hardware más avanzado y procesador más rápido.
Coste: Más económico.	**Coste**: Más caro debido a sus capacidades extendidas.

¿Cuándo usar cada uno?

- **Rubber Ducky**: Ideal para ataques rápidos, como el uso de scripts para inyectar comandos en segundos.
- **Bash Bunny**: Más adecuado para escenarios complejos que requieran múltiples dispositivos simulados y capacidades extendidas.

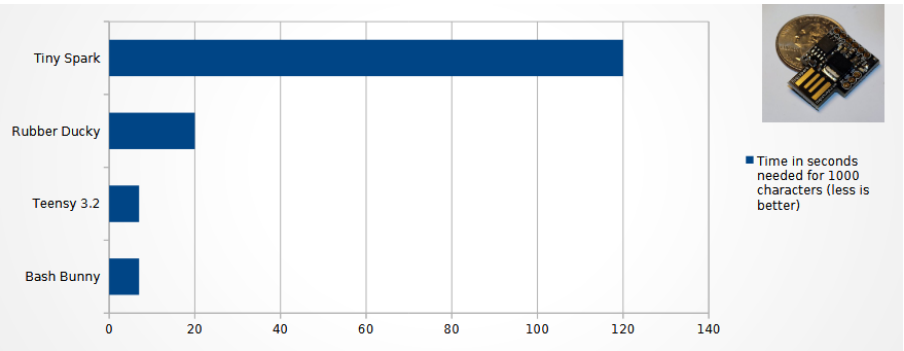
Ambos son herramientas poderosas que, usadas éticamente, ayudan a identificar vulnerabilidades en la seguridad de sistemas.

Bash Bunny Speed

La innovadora plataforma de carga útil que introdujo ataques USB multivector ha evolucionado. Realiza ataques encubiertos o tareas de automatización de TI más rápido que nunca con sólo el movimiento de un interruptor.

El NUEVO Bash Bunny Mark II va de enchufe a pwn en **7 segundos**, así que cuando la luz se vuelve verde la máquina está hackeada.

Ahora con **un rendimiento más rápido, Geofencing inalámbrico, disparadores remotos y soporte MicroSD**, el Bash Bunny es una herramienta aún más impresionante para tu arsenal de equipo rojo. Simultáneamente imita múltiples dispositivos de confianza para engañar a los objetivos para que divulguen información sensible sin desencadenar defensas. El Bash Bunny es realmente la plataforma de ataque USB más avanzada del mundo.



Bash Bunny Mark I y Bash Bunny Mark II

Las diferencias entre el **Bash Bunny Mark I** y el **Bash Bunny Mark II** se centran principalmente en mejoras de hardware, capacidad de almacenamiento y funcionalidad:

1. Hardware

- **Procesador**:

Mark I: Procesador menos potente, lo que limita la velocidad de ejecución de scripts complejos.

Mark II: Procesador actualizado para un mejor rendimiento y tiempos de ejecución más rápidos.

- **Memoria RAM**:

Mark I: Memoria RAM más limitada, lo que afecta el rendimiento multitarea.

Mark II: RAM mejorada, permitiendo ejecutar payloads más complejos y simultáneos.

2. Almacenamiento

- **Capacidad de almacenamiento**:

Mark I: Menor espacio de almacenamiento interno, lo que limita la cantidad de scripts o archivos grandes que se pueden almacenar.

Mark II: Mayor capacidad de almacenamiento interno, ideal para guardar múltiples payloads y herramientas.

- **Velocidad de transferencia**:

Mark I: Velocidad de transferencia de datos más baja al actuar como dispositivo de almacenamiento.

Mark II: Mejora significativa en la velocidad de transferencia, lo que lo hace más eficiente.

3. Compatibilidad y funcionalidad

- **Compatibilidad con sistemas operativos**: Ambos modelos son compatibles con la mayoría de sistemas operativos, pero el **Mark II** incluye mejoras para evitar detección y manejar sistemas más recientes con parches avanzados.
- **Soporte de payloads**:

Mark I: Puede ejecutar payloads básicos.

Mark II: Permite ejecutar payloads más avanzados y con mayor complejidad, aprovechando el hardware mejorado.

- **Gestión de modos USB**:

Mark II ofrece un mejor manejo de la simulación de dispositivos USB (almacenamiento, teclado, adaptador de red, etc.) de forma más rápida y eficiente.

4. Consumo energético

- **Mark I**: Consumo energético mayor y menos optimizado, afectando la duración en sistemas que dependen de baterías (como laptops).
- **Mark II**: Mejor optimización energética, permitiendo un uso más prolongado sin consumir tanto.

5. Precio

- **Mark I**: Más económico debido a su hardware más antiguo.
- **Mark II**: Precio más alto por las mejoras en rendimiento y capacidad.

Resumen

El Bash Bunny **Mark II** es una evolución directa del **Mark I**, diseñado para ser más rápido, más potente y con mayor capacidad de almacenamiento. Mientras que el **Mark I** sigue siendo útil para tareas básicas, el **Mark II** es más adecuado para escenarios avanzados y entornos con mayores exigencias.

¿Dónde comprar Bash Bunny?

Sólo hay un vendedor oficial y esa es la tienda Hak5. El precio del Bash Bunny es de unos $119. Sin embargo, dado que los almacenes Hak5 se encuentran sólo en el continente americano, hay que añadir un precio significativo para el transporte al precio final, que nos lleva a unos $145.

El precio podría variar según ofertas.

Unboxing Bash Bunny

La entrega es relativamente rápida, ya que podemos tener el Bash Bunny en nuestro buzón aproximadamente en una semana, comenzando desde el momento de pedido.

El Bash Bunny está en nuestra mesa. Así que, ¿ahora qué?

Echa un vistazo al contenido del elegante embalaje:

Esta es la **versión Mark II**, que incluye **geofencing inalámbrico, disparadores remotos, soporte microSD** y **un rendimiento más rápido.**

Resumen de plataformas

Independientemente del sistema operativo (MacOS, Linux, Windows, Android), todos los dispositivos modernos implementan la noción de dispositivos de confianza, es decir, dispositivos en los que un sistema confía y acepta automáticamente *sin necesidad de confirmación* ni controladores.

Existen varias categorías de dispositivos de confianza, entre ellas:

- HID ("Human Input Devices"): teclados, ratones, etc.
- Dispositivos de almacenamiento: unidades flash, etc.
- Dispositivos de Red - Adaptadores Ethernet, etc

El Conejo Bash puede emular todos estos dispositivos, simultáneamente - y luego abusar de esta confianza a través de Payloads scriptables.

Escribe o personaliza fácilmente tu propio payload, o utiliza uno de los cientos disponibles en el repositorio de Bash Bunny.

Se pueden almacenar varias cargas útiles y seleccionarlas mediante un interruptor físico. El LED RGB proporciona información instantánea y encubierta sobre el estado de la carga útil.

Bash Bunny es una potente máquina Linux de cuatro núcleos con todas las funciones en un paquete diminuto, accesible a través de una interfaz serie.

Conectando Bash Bunny en Linux

1. Conecta el Bash Bunny al puerto USB

Inserta el dispositivo Bash Bunny en un puerto USB de tu sistema Linux. Dependiendo del payload activo y el modo en que esté configurado, el Bash Bunny puede comportarse como:

- Un dispositivo de almacenamiento USB.
- Un adaptador Ethernet USB.

2. Accede al modo de almacenamiento (armar payloads)

Cambia el interruptor de posición:

El Bash Bunny tiene un interruptor físico con dos posiciones:

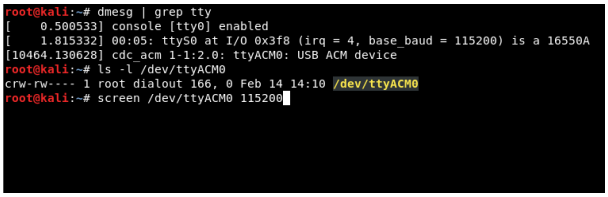
- **Switch 1 (Payload 1):** Modo normal para ejecutar payloads.
- **Switch 2 (Payload 2):** Otro modo normal para ejecutar payloads.
- **Deslízalo hacia el medio:** Entra en **modo de almacenamiento**, lo que permite acceder al Bash Bunny como un pendrive para configurar scripts.

Montar el almacenamiento USB

En la mayoría de las distribuciones de Linux, el almacenamiento del Bash Bunny se monta automáticamente al conectarlo. Si no ocurre automáticamente, puedes montarlo manualmente:

sudo mount /dev/sdX /mnt

(Reemplaza **`/dev/sdX`** con el identificador del dispositivo que aparece al conectar el Bash Bunny, que puedes encontrar con **`lsblk`** o **`dmesg`**).



Configura tus payloads:

- Navega hasta la carpeta `/payloads` en el almacenamiento.
- Coloca tus scripts y configúralos según el modo deseado.

3. Configuración de red (si está en modo Ethernet)

Si el Bash Bunny está configurado para actuar como un adaptador Ethernet USB, sigue estos pasos:

Verifica la conexión USB Ethernet

Comprueba si tu sistema reconoce la interfaz:

ip link

Deberías ver una nueva interfaz, como `**usb0**` o similar.

Configura la interfaz USB Ethernet

Asigna una dirección IP a la interfaz para conectarte al Bash Bunny:

sudo ifconfig usb0 172.16.64.1 netmask 255.255.255.0

Habilita el servidor SSH del Bash Bunny

Por defecto, el Bash Bunny tiene habilitado un servidor SSH en la IP `**172.16.64.1**`. Conéctate usando SSH:

ssh root@172.16.64.1

- Contraseña predeterminada: **hak5bunny**.
- Una vez dentro, puedes administrar el sistema operativo del Bash Bunny, subir scripts o ejecutar comandos.

Carga y prueba de payloads

1. Cambia el interruptor a la posición del payload deseado (1 o 2).
2. Desconecta y reconecta el Bash Bunny para ejecutar el payload.
3. Repite según sea necesario para ajustar los scripts.

Consejos adicionales

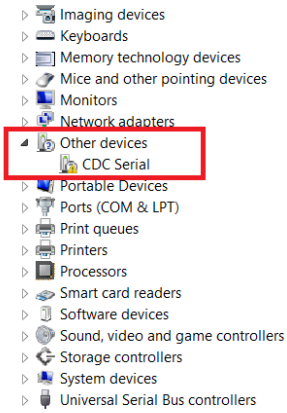
Actualizar firmware: Antes de usar el Bash Bunny, asegúrate de que tiene el firmware más reciente. Puedes descargarlo desde la página de Hak5.

Instalar dependencias: Algunos payloads requieren dependencias adicionales que puedes instalar conectándote al dispositivo vía SSH.

Documentación oficial: Consulta la wiki de Hak5 para detalles específicos de payloads y configuración avanzada.

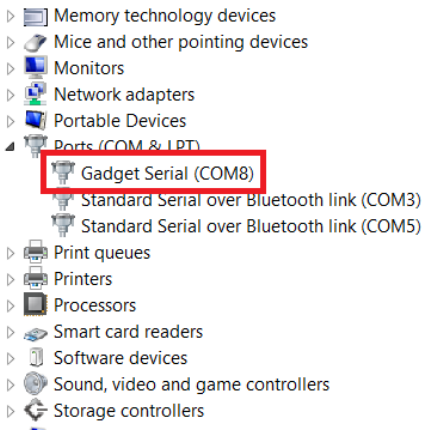
Conectando Bash Bunny en Windows

En caso de que el dispositivo se pusiera en funcionamiento en Windows, a menudo nos encontramos con una situación, cuando el puerto serie de Bash Bunny no es reconocido correctamente por el sistema:

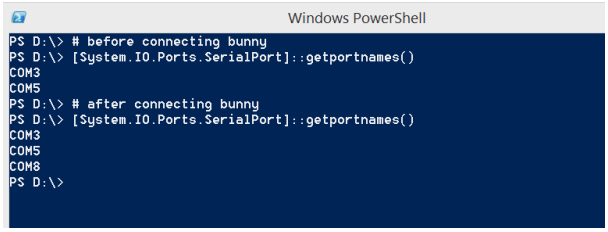


Si este es el caso, se requiere una instalación manual de los controladores ubicados en el almacenamiento extraíble Bash Bunny. Sin embargo, el problema es que estos controladores no están firmados, por lo que es necesario desactivar la firma de controladores con Advanced Startup y reiniciar el sistema antes de la instalación. Después de instalar los controladores, es más que recomendable que vuelvas a habilitar la firma de controladores.

El puerto serie se reconoce ahora correctamente después de la instalación de los controladores, y en este caso, fue asignado como COM8.

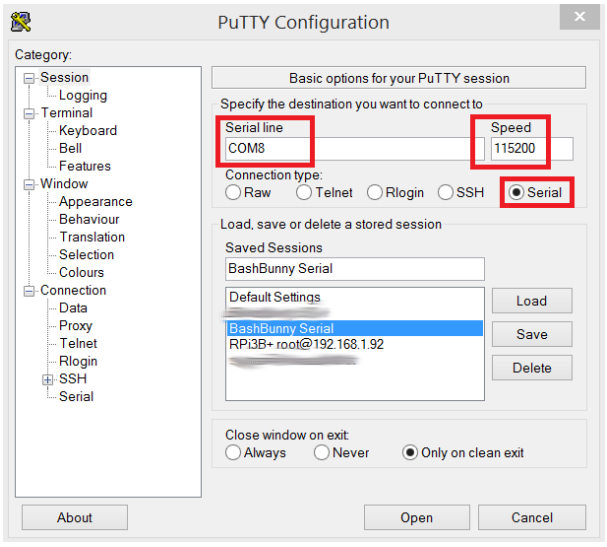


Alternativamente, fuera del Administrador de dispositivos, el número COM se puede descubrir con PowerShell:

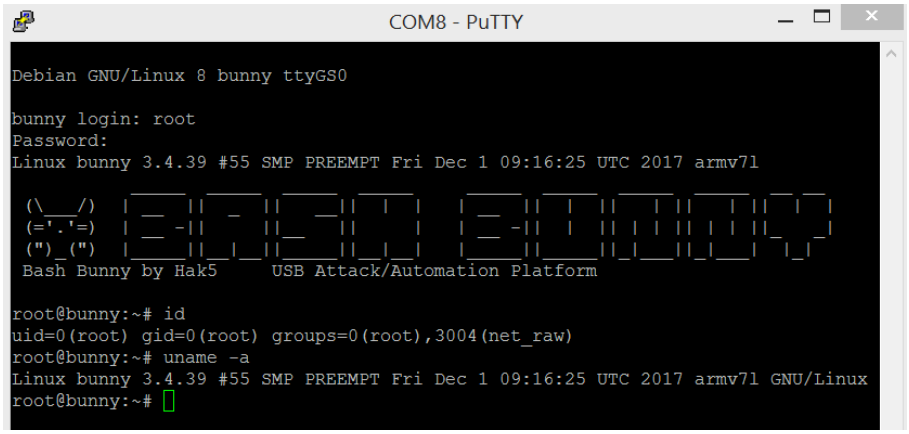


En cuanto a Windows como objetivo de ataques, aquí el Bash Bunny funciona Fuera de la caja, ya que los controladores para la interfaz de red y el almacenamiento extraíble se identifican e instalan correctamente y automáticamente.

Ahora, es posible conectarse a la consola Bash Bunny a través del puerto COM8 utilizando la herramienta Putty:



Después de conectarse a la consola, es posible comprobar que realmente hay Linux de plena base en la plataforma ARM en este pequeño dispositivo.



Ahora, cuando ponemos el Bash Bunny en operación para poder administrarlo completamente, es hora de una actualización. Las actualizaciones de Bash Bunny se pueden dividir en tres tipos:

1. Actualización de firmware,
2. Actualización del sistema operativo (Debian Linux),
3. Actualización de cargas útiles.

Actualización de firmware

Para actualizar el firmware, el Bash Bunny tiene que ser cambiado a modo de armado y conectarse a un puerto USB en el ordenador. Desechemos la última versión de firmware. El firmware se descarga en el formato de archivo tar.gz (por ejemplo, ch-fw.1.5-298.tar.gz) y lo copiaremos sin desempaquetar al almacenamiento extraíble en la carpeta raíz.

Ahora tienes que desconectar el Bash Bunny. Es cierto que la vida es demasiado corta para la desconexión segura de los dispositivos USB de la computadora, pero si no desea pasar las siguientes horas tratando de poner en funcionamiento el Bash Bunny, recomiendo encarecidamente este paso.

Ahora todo lo que tienes que hacer es desconectar el Bash Bunny y reconectarlo con el puerto USB y la actualización del firmware comenzará automáticamente. Se tarda unos 5 minutos en total; el Bash Bunny normalmente arranca y puedes empezar a usarlo.

Actualización del sistema operativo (Debian Linux)

Esta situación no es diferente de la forma en que actualizas a Debian en un escritorio o servidor. Bajo los permisos de usuario raíz, ejecuta:

apt-get update && apt-get dist-upgrade && apt-get autoremovemove

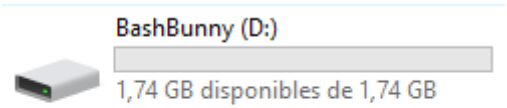
Puedes encontrar un problema aquí, que es que el Bush Bunny no tiene acceso a Internet a nivel del sistema operativo. En ese caso, echa un vistazo a la Wiki oficial, donde encontrará el manual para Linux, así como para Windows.

Actualización de cargas útiles

La actualización de la base de datos de carga útil es entonces el paso final. Piensa en un solo guión de ataque bajo el término carga útil. La actual base de datos de carga útil se puede descargar del Github oficial:

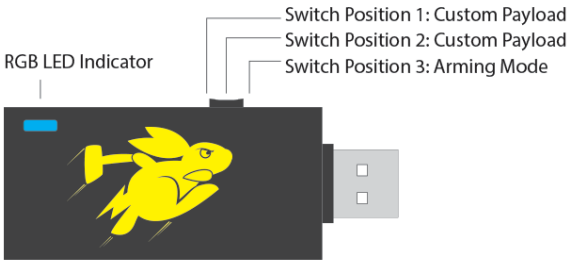
- https://github.com/hak5/bashshunny-payloads

Después de descargar, todo lo que tienes que hacer es copiar los archivos al almacenamiento de datos. En este punto, tu Bash Bunny está completamente actualizado, y puedes empezar a jugar con las cargas útiles como tales.



Conmutar posiciones

En la posición de conmutación 3 (más cercano al enchufe USB) el Bash Bunny arrancará en *modo de armamento*, permitiendo tanto el almacenamiento en serie como masivo. Desde este modo dedicado, las cargas útiles de Bash Bunny se pueden gestionar a través de Almacenamiento de Masas y la consola Serial puede acceder a la shell Linux.

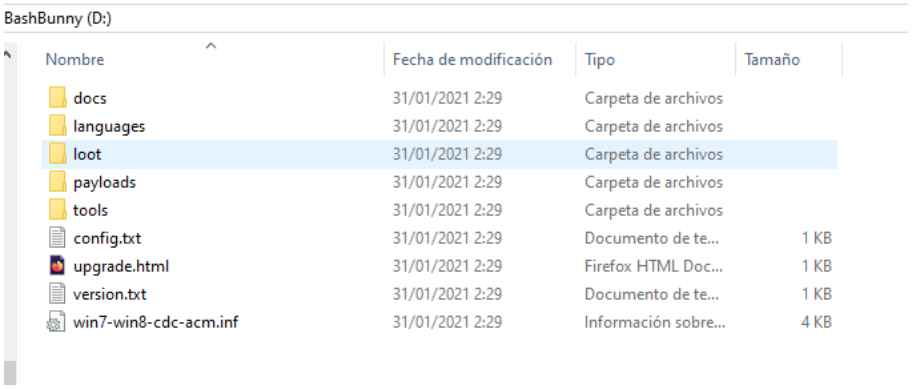


Estructura de almacenamiento masivo

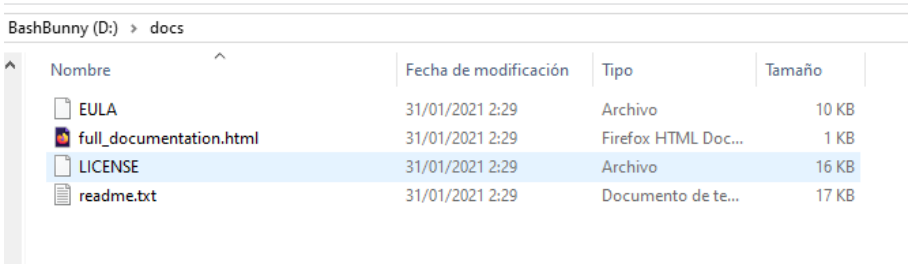
- */docs* – Aquí tienes la documentación.
- */languages* – Para instalar idiomas de teclado HID adicionales.
- */loot* – utilizada por cargas útiles para almacenar registros y otros datos.
- */tools* – Herramientas usadas para instalar paquetes de deb adicionales y otras herramientas..
- */payloads* – Aquí van las cargas útiles activas, bibliotecas y extensiones.

- */payloads/switch1* y */payloads/switch2* – Aquí van los payload.txt y archivos acompañantes que se ejecutarán en el arranque cuando el interruptor de conejito bash esté en la posición correspondiente.
- */payloads/library* – Hogar de la biblioteca de cargas útiles que se puede descargar desde el [repositorio Bash Bunny Payload git]
- */payloads/library/extensions* – Aquí van las Bash Bunny extensions.

Bash Bunny Mark II Nota: Si una tarjeta MicroSD está presente en el arranque en las posiciones de conmutación 1 o 2, /root/udisk se enlazará a la raíz de la tarjeta MicroSD. De lo contrario, la partición de udisk se comportará como de costumbre en la SSD interna.

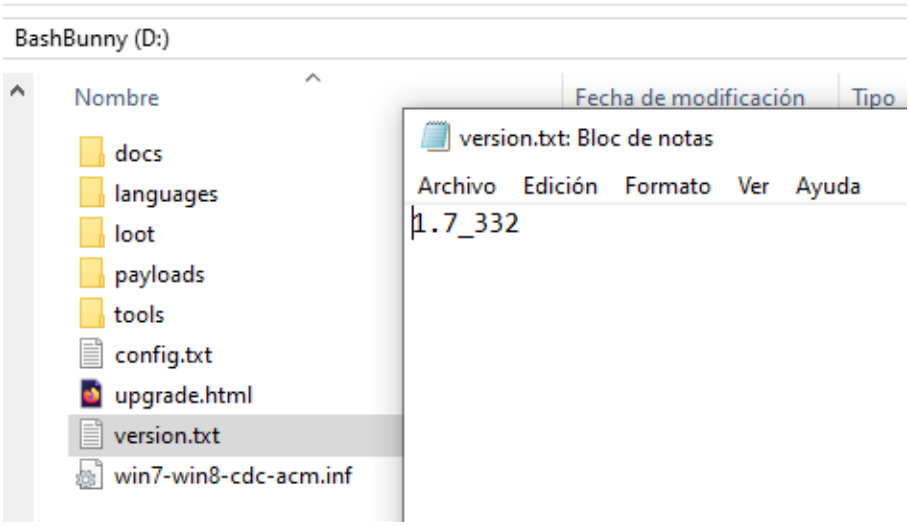


***/docs* – Aquí tienes la documentación:**



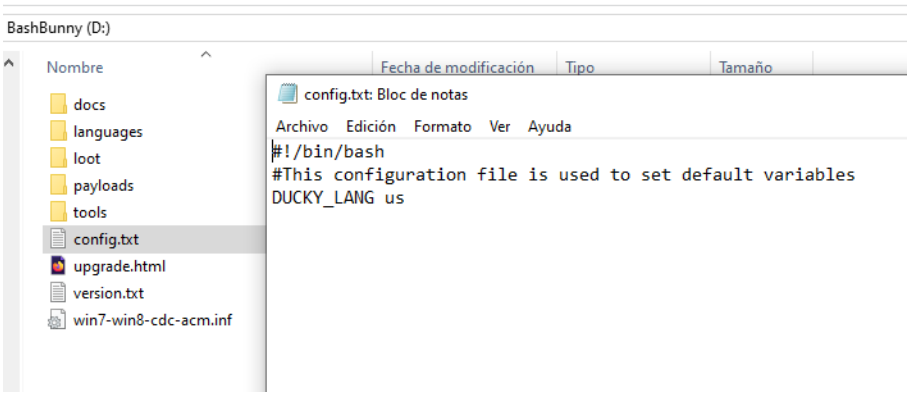
version.txt

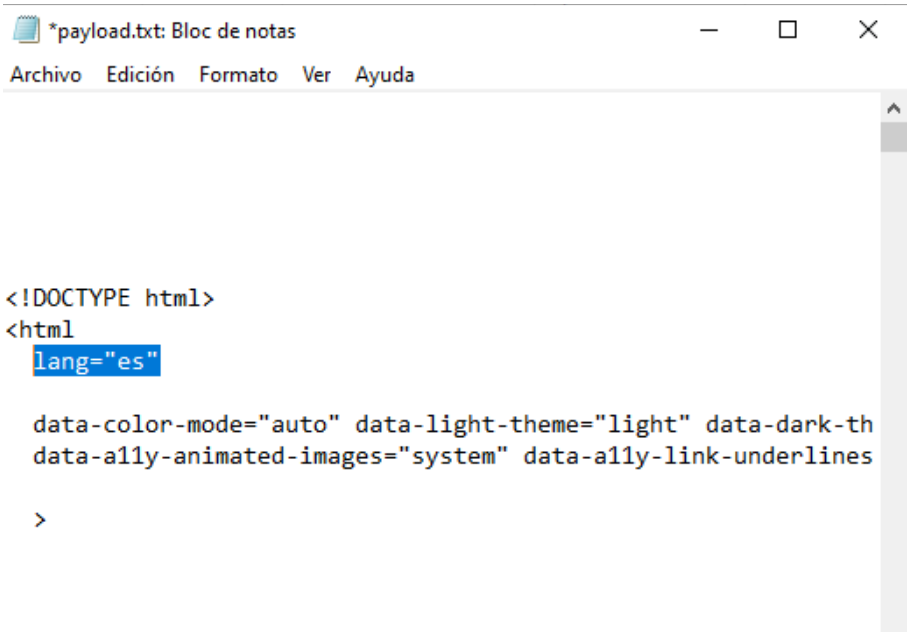
Aquí tienes la versión de tu conejito:



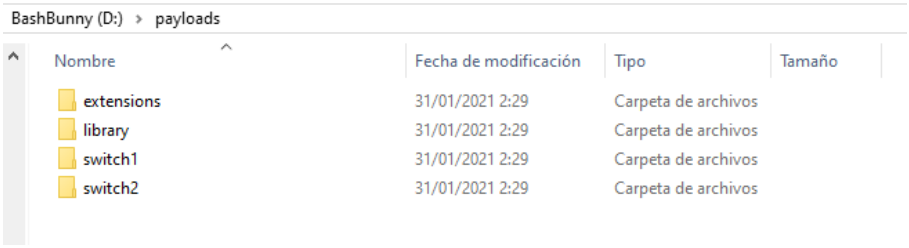
config.txt

Aquí tenemos el idioma que ejecuta de forma predeterminada. Si tu payload utiliza otro idioma deberás indicarlo en el payload y tenerlo instalado en */languages.*

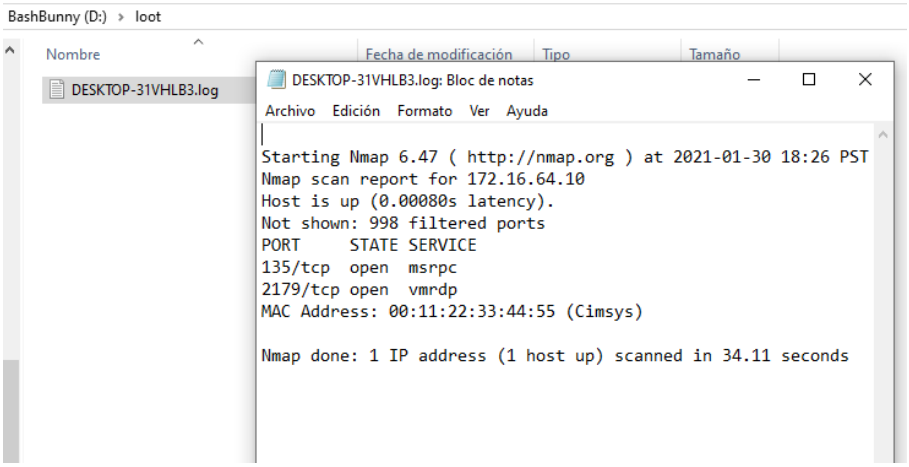




/payloads/switch1 y ***/payloads/switch2*** – Aquí van los payload.txt y archivos acompañantes, como un script de Powershell, que se ejecutarán en el arranque cuando el interruptor de conejito bash esté en la posición correspondiente:



/loot – Aquí se guardarán las "recompensas". Si el payload ejecuta por ejemplo un escaneado de puertos con Nmap, puedes pedirle que te guarde el resultado en esta carpeta:



Indicaciones de estado de LED

LED Estado Indicaciones

LED	Status
Verde (parpadeante)	Booting up
Azul (parpadeante)	Arming Mode
Rojo (parpadeante)	Recovery Mode or Firmware Flashing *from v1.0* **NO DESENCHUFAR**
Rojo/Azul Alternado	Recovery Mode or Firmware Flashing *from v1.1+* **NO DESENCHUFAR**

Instalación de herramientas adicionales

Mientras que muchas herramientas se pueden instalar en el Bash Bunny como lo harías en cualquier computadora Linux basada en Debian, como *apt install*, *git clone*, una carpeta de herramientas dedicada de la partición de almacenamiento en masa simplifica el proceso. Accesible desde el modo de blindado, las herramientas en formato .deb o directorios enteros se pueden copiar fácilmente /tools en la raíz de la partición de almacenamiento de

masas. Luego, en el siguiente inicio del modo Armado de Bash Bunny, estas herramientas se instalarán. LED SETUP (Luz magenta).

En el arranque en modo de armado, cualquier archivo .deb colocado en la carpeta de herramientas se instalará con dpkg. Entonces cualquier archivo o directorio restante se trasladará a /tools en el sistema de archivos raíz.

Algunas cargas útiles pueden requerir herramientas adicionales de terceros. Por ejemplo, el rdp.checker La carga útil requiere impacket para ubicarse en /tools/impacket. Esto se puede instalar copiando el directorio impacket o un archivo impacket.deb al directorio /tools y arrancar en modo de armado. La carga útil del rdp-chever también hace uso de la REQUIRETOOL extensión, que comprueba la existencia de esta herramienta y sale con un parpadeo rojo LED FAIL que indica si la herramienta no se encuentra.

Una lista de herramientas precompiladas está disponible en este hilo del foro.

Instalación de idiomas adicionales

Las cargas útiles de Bash Bunny pueden ejecutar ataques de inyección de pulsación de teclas similares al USB Rubber Ducky mediante el uso de HID ATTACKMODE. De forma predeterminada, este modo utiliza un diseño de teclado de EE.UU.

La comunidad puede desarrollar diseños adicionales de teclado. Instalar diseños adicionales de teclado es similar al uso de la carpeta de herramientas en la raíz de la partición de almacenamiento del USB.

En el modo de boot-up en el modo de armado, se instalará cualquier archivo de dos letras-country-code.json ubicado en la carpeta /languages en la raíz de la partición de almacenamiento del USB. El archivo permanecerá en /languages después de la instalación.

Con un nuevo archivo de idioma instalado, se puede especificar el diseño del teclado de una carga útil mediante el uso de la extensión **DUCKY-LANG**.

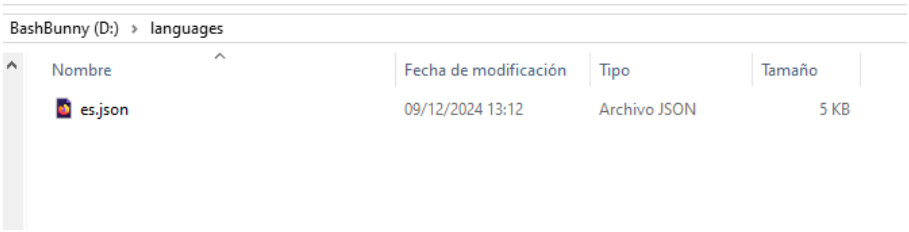
Esta extensión acepta un código de país de dos letras.

Ejemplo:

DUCKY_LANG us

Puedes descargar el archivo .json del idioma que necesites desde aquí:

https://github.com/hak5/bashbunny-payloads/tree/master/languages



Consideraciones para Mark II

El Bash Bunny Mark II añade exfiltración masiva, geoesencia inalámbrica y funcionalidad de gatillo remoto a través de un lector de tarjetas MicroSD XC y radio bluetooth de baja energía. Todas las cargas útiles de primera generación son compatibles con el Bash Bunny Mark II. Dos consideraciones a tener en cuenta al desarrollar y desplegar cargas útiles para el Bash Bunny Mark II; Wireless y Storage.

WIRELESS

Si lo desea, WAIT_FOR_PRESENT o WAIT_FOR_NOT_PRESENT

Las extensiones se pueden utilizar para geoesgrima y disparadores remotos. Al usar estas extensiones, el paisaje inalámbrico bluetooth se leerá temporalmente a /tmp/bt-observation

STORAGE

Algunos puntos clave a tener en cuenta al usar una tarjeta MicroSD con el Bash Bunny Mark II:

Modo de armado

Para cargar cargas útiles, arranca el Bash Bunny **sin una tarjeta MicroSD presente.**

- Las cargas útiles se ejecutan **únicamente desde el almacenamiento interno.**
- Si una tarjeta MicroSD está presente en el modo de armado, se pasará al host.

Consideraciones de carga útil

- Si ATTACKMODE STORAGE está activa:En caso de que esté presente una tarjeta MicroSD, la tarjeta MicroSD se presentará al objetivo. En caso de que una tarjeta MicroSD no esté presente, la partición interna de udisk se presentará al objetivo.
- Por defecto, *después de cargar cargas útiles durante* el *arranque*, el udisk **no** se **monta desde la perspectiva del Bash Bunny.** Para montar el udisk desde la perspectiva del Bash Bunny, edita el comando `udisk mount`

Consideraciones de montaje

- La partición de udisk, ya sea interna o MicroSD, sólo se puede montar en un dispositivo a la vez.
- El /root/udisk: El directorio aparecerá en blanco a menos que udisk mount- ha sido ejecutado.
- Escribir a /root/udisk cuando desmontamos no tendrá efecto en la partición udisk real.
- Si ambos ATTACKMODE STORAGE y .udisk están montados, un comportamiento inesperado puede ocurrir, ya que la partición no puede ser manejada tanto por ambos.

Consideraciones de formato

- La tarjeta MicroSD debe dividirse con una sola partición formateada con un sistema de archivos adecuado al destino.

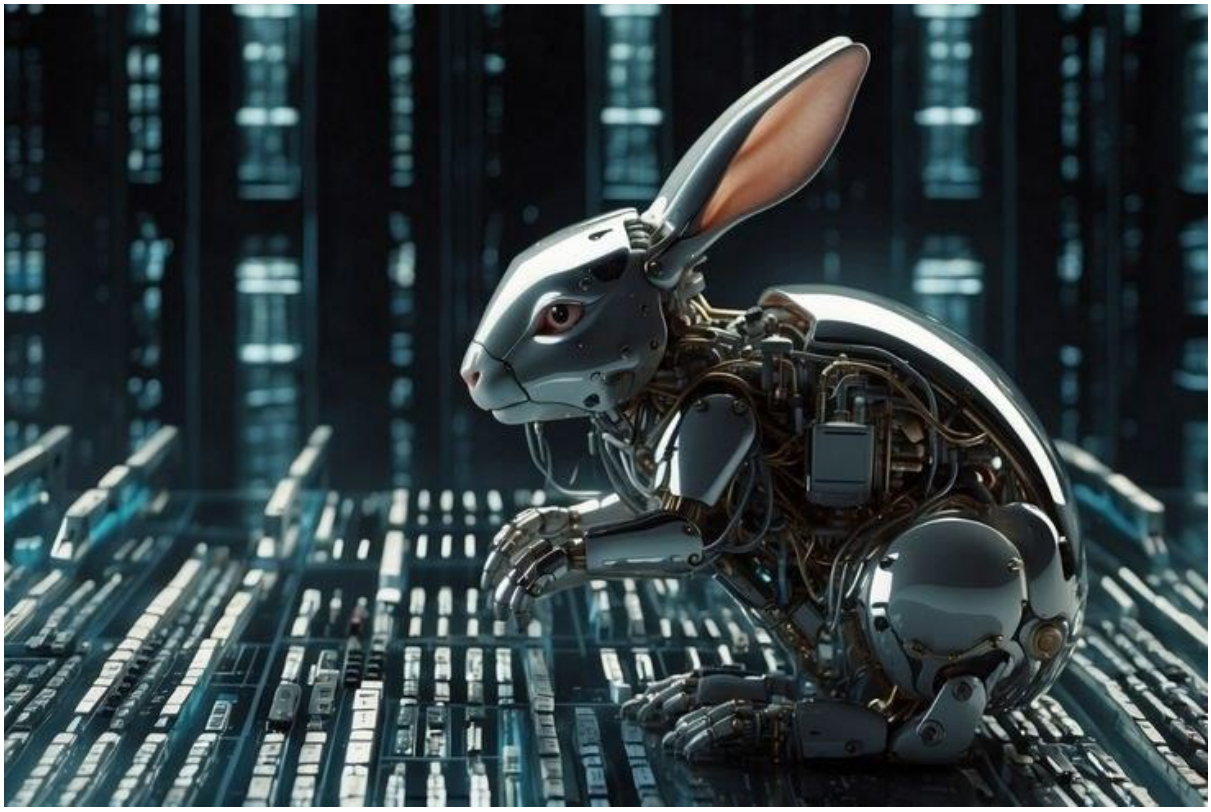
Por ejemplo, para los objetivos de Windows: FAT32, ExFAT, NTFS.

Por ejemplo, para objetivos de Mac: FAT32, ExFAT, APFS

Por ejemplo, para los objetivos de Linux: FAT32, ExFAT, EXT

- Si bien el objetivo puede soportar varios sistemas de archivos, el host (Bash Bunny) actualmente sólo admite EXT y FAT32. Los sistemas de archivos adicionales (ExFAT) pueden incluirse en futuras versiones de firmware.

Consideraciones para la creación de Payloads



Puedes empezar probando los payloads disponibles en la página de GitHub de Hak5. Esta es una buena manera de empezar pero seguramente debas adaptar el payload a tus necesidades concretas.

Biblioteca de carga útil para el Bash Bunny de Hak5

Este repositorio contiene cargas útiles y extensiones para el Hak5 Bash Bunny. Se enumeran las cargas útiles desarrolladas por la comunidad y se anima a los desarrolladores a crear solicitudes de retirada para hacer cambios o enviar nuevas cargas útiles.Las cargas útiles aquí están escritas en DuckyScriptTM y Bash oficiales específicamente para el Bash Bunny. Hak5 NO garantiza la funcionalidad de carga útil.

Consideraciones para escribir un Payload:

Escribir un payload para el **Bash Bunny** requiere tener en cuenta varios aspectos clave para garantizar que funcione correctamente y cumpla su propósito. Aquí tienes las consideraciones más importantes:

1. Estructura básica del payload

Un payload debe estar bien organizado y seguir las convenciones del Bash Bunny:

- **Directorio de ubicación**:Los payloads se colocan en la carpeta /payloads/switch1 o /payloads/switch2, según el interruptor físico del dispositivo.

- **Archivo principal**:El script principal del payload debe llamarse payload.txt. Este archivo contiene las instrucciones que el Bash Bunny ejecutará.

Ejemplo de estructura básica:

/payloads

├── switch1/

│ └── payload.txt

└── switch2/

└── payload.txt

2. Uso correcto de Bash y funciones del Bash Bunny

- **Lenguaje de scripting**: Los payloads están basados en Bash, por lo que debes tener conocimientos básicos de scripting en Linux.
- **Comandos predefinidos del Bash Bunny**: Aprovecha las funciones integradas como:ATTACKMODE para definir el tipo de dispositivo que emula el Bash Bunny (almacenamiento, teclado, Ethernet, etc.).LED para indicar el estado del payload (usado para depuración).Q (Quick Command) para enviar pulsaciones de teclado cuando el dispositivo actúa como teclado USB.

Ejemplo:

```
ATTACKMODE HID STORAGE  # Emula teclado y almacenamiento USB
LED ATTACK              # Cambia el LED al color de "Ataque en curso"
Q DELAY 500             # Espera 500 ms
Q STRING whoami         # Escribe "whoami" en la consola del objetivo
Q ENTER                 # Presiona Enter
LED FINISH              # Indica que el ataque ha terminado
```

3. Configuración del modo USB

El Bash Bunny puede actuar como diferentes tipos de dispositivos USB, como:

- **HID (teclado)**
- **Mass Storage (almacenamiento)**
- **Ethernet**Define correctamente el modo usando el comando ATTACKMODE.

Ejemplo:

ATTACKMODE HID # Emula un teclado USB

4. Considerar la velocidad de ejecución

- **Delays (retrasos)**: Los sistemas objetivo pueden tardar en procesar entradas del teclado o montar dispositivos. Asegúrate de usar comandos como Q DELAY para evitar que el payload falle debido a la velocidad excesiva.
- **Sincronización**: Agrega pausas estratégicas entre comandos para garantizar que el dispositivo objetivo procese cada acción antes de continuar.

5. Compatibilidad del sistema objetivo

- **Sistema operativo**: Asegúrate de que el payload esté diseñado para el sistema operativo específico del objetivo (Windows, macOS, Linux).Para Windows: Usa comandos CMD o PowerShell.Para macOS/Linux: Usa comandos de terminal Bash.
- **Localización del teclado**: Configura el Bash Bunny para que emule un teclado con el diseño correcto (por ejemplo, US, ES, etc.).

Ejemplo:

```
DUCKY_LANG es          # Configura el teclado al español
```

6. Seguridad y depuración

- **Usa el LED para indicar estados**: Configura el LED del Bash Bunny para mostrar el progreso del payload (inicialización, ataque, error, finalización).
- **Depura antes de usar en objetivos reales**: Prueba el payload en un entorno controlado para identificar errores.

Ejemplo de estados del LED:

```
LED SETUP              # Preparación
LED ATTACK             # Ataque en curso
LED FINISH             # Ataque finalizado
```

7. Estructura modular del script

Divide el payload en secciones claras para facilitar su mantenimiento:

1. **Preparación**: Configuración inicial del modo USB y LEDs.
2. **Ejecución**: Comandos principales del ataque.
3. **Limpieza**: Finaliza el ataque y regresa a un estado seguro.

Ejemplo:

```
# Configuración inicial
ATTACKMODE HID
LED SETUP

# Payload principal
Q STRING whoami
Q ENTER
LED ATTACK

# Finalización
LED FINISH
```

8. Respeto ético y legal

- **Autorización**: Usa el Bash Bunny solo en entornos donde tengas permiso explícito para realizar pruebas de seguridad.
- **Pruebas éticas**: Diseña payloads que ayuden a identificar vulnerabilidades sin causar daño al sistema objetivo.

9. Optimización del almacenamiento

- **Mantén los payloads ligeros**: El Bash Bunny tiene un almacenamiento limitado, por lo que los archivos y dependencias deben ser mínimos.
- **Carga dependencias externas solo si es necesario**: Si el payload requiere herramientas adicionales, asegúrate de descargarlas o instalarlas solo cuando sean imprescindibles.

10. Documentación del payload

Incluye un archivo README.md junto al payload que explique:

- Su propósito.
- Cómo configurarlo.
- Dependencias necesarias.
- Consideraciones especiales.