BadUSB

BADUSB

MALWARE, MÁS ALLÁ QUE SOLO SOFTWARE

Por Victor Augusto López Hernández
Universidad de San Carlos de Guatemala
Noviembre de 2014

BADUSB

MALWARE, MÁS ALLÁ QUE SOLO SOFTWARE

ÍNDICE

ABSTRACT

When we talk about malware and how to solve the issues that are caused by them, we think about some security solution, antivirus, antispyware, and the list continues, but, what happen when your hardware turns into that malware and becomes almost undetectable to software solution, this is what BadUSB is about. BadUSB is a modification to the firmware of a usb device, in order to look and work like a normal usb device, but doing something else at the same time, for example, being a keylogger.

Victor Augusto
López Hernández

Universidad de San Carlos
de Guatemala

Premio Universitario
ESET 2014

INTRODUCCIÓN

Con el avance que tienen las tecnologías de la información día tras día, las personas utilizan cada día más las mismas de manera común y rutinaria, usándolas desde un Smartphone de gama baja, puesto que los precios cada vez son más accesibles, hasta hacer uso de grandes servidores de datos cuando se registran en algún servicio en línea.

Muchas cosas que tendíamos a hacer presencialmente ahora se pueden realizar en algún servicio en línea, compras, trámites, operaciones bancarias; intercambiamos en nuestros equipos electrónicos información atractiva para atacantes, que hacen uso de malware para obtenerla. Así que, al mismo tiempo que adquirimos un dispositivo electrónico nuevo, una computadora o un teléfono, pensamos en qué solución de seguridad podemos utilizar para evitar cualquier tipo de infección; podríamos verlo como una carrera entre quienes desarrollan el malware que nos ataca, y quienes desarrollan las soluciones de seguridad que nos protegen, una carrera que posiblemente jamás tenga final, y es aquí donde importa pensar en nuevas estrategias de infección para los atacantes que se encuentran con la barrera de las soluciones de seguridad informática.

Durante muchos años, la carrera ha sido a nivel de software, pero, si a nivel de software las personas tienden a protegerse ¿por qué no intentar una estrategia nueva y diferente? Hace algunos años un gusano llamado Stuxnet fue el primero en intentarlo, afectando a los Siemens S7PLCs [1].

Es común encontrar gente que confía en conectar pendrives USB a sus computadoras una vez que algún antivirus ha hecho una búsqueda de virus o software indeseado, pero BadUSB va más allá de eso, y explota la vulnerabilidad que algunos dispositivos USB tienen al permitir actualizaciones de firmware, y por lo que en algún momento, la seguridad informática deberá considerarse más allá que solo en software.

¿CÓMO FUNCIONA USB?

FUNCIONAMIENTO E IDENTIFICACIÓN DE DISPOSITIVOS

USB apareció a mediados de los años 90 con el objetivo de estandarizar la conexión de periféricos a la computadora, como por ejemplo, teclados, cámaras, impresoras, adaptadores, entre otros.

USB conecta diferentes dispositivos electrónicos a un host controlador a través de hubs, estos dispositivos, son llamados funciones, porque cada dispositivo físicamente puede tener muchas funciones, por ejemplo, una webcam que además tiene un micrófono incorporado. Los hubs son dispositivos de propósito especial que no se consideran funciones, aunque siempre debe existir un hub que es conocido como root, o raíz, que se encuentra conectado directamente al host controlador. El host controlador y hub raíz en la práctica, es cada uno de los puertos USB implementados en una computadora.

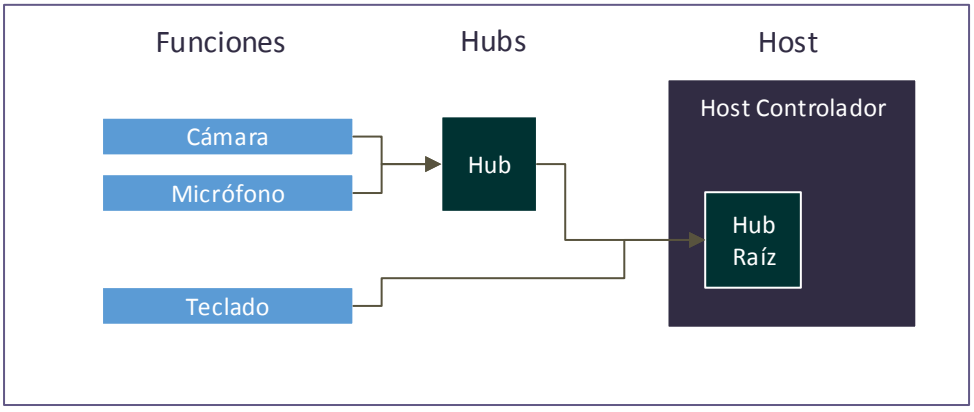


Figura 1. Diagrama de funcionamiento de USB

Cada una de estas funciones o dispositivos, tienen asociados canales lógicos, estos canales son conexiones desde el host controlador a una unidad lógica en el dispositivo llamada “endpoint”; cada endpoint tiene dos canales unidireccionales; los endpoints están numerados de cero a quince, sin embargo, el endpoint 0 está reservado para control. Además, existen descriptores de clase que indican qué tipo de dispositivo se está conectando, que a su vez también pueden tener sub-descriptores. [2]

Continuando con el ejemplo de los dispositivos conectados en la Figura 1, podemos obtener la siguiente tabla de información al realizar la conexión.

Identificador	Webcam	Teclado
Descriptor de clase	0x01 - Audio Class 0x0E - Video Class	0x03 - HID Class / Human Interface Device Class
Endpoints	0 - Control 1 - Video 2 - Audio	0 - Control 1 - Transferencia de datos

Tabla 1. Tabla de información de conexión USB

PROCESO DE INICIALIZACIÓN

En la sección anterior se muestra el funcionamiento básico y la información que utiliza USB para realizar una conexión e identificarla, ésta es de suma importancia para el proceso de inicialización y que además será la base para hablar de BadUSB más adelante.

El proceso de inicialización comienza al conectarse el dispositivo USB a un host controlador, este envía una solicitud de registro y el host controlador le asigna una dirección de acceso, a continuación el dispositivo envía el descriptor con el que se identifica para que el host controlador asigne el driver adecuado para el dispositivo, este se configura y ya se puede operar entre ambos como normalmente estamos acostumbrados a hacerlo, este proceso se puede repetir de ser necesario, por ejemplo, al cambiar el modo de conexión USB de nuestro Smartphone, sin desconectarlo físicamente de la computadora, en la computadora podemos observar cómo se desconecta y vuelve a conectarse como un dispositivo diferente.

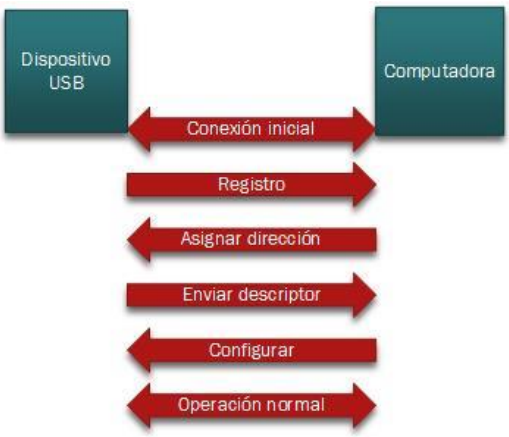


Figura 2. Inicialización USB

El escenario de conexión de un dispositivo USB ahora está completo. Un dispositivo USB se puede identificar como un dispositivo con una gran cantidad de capacidades, tantas como sus descriptores de clase; es capaz de contener en una misma conexión muchas funciones diferentes, tantas como los endpoints pueden proveer y además, este se puede registrar la cantidad de veces que se desee, lo que hace éste tipo de conexión bastante versátil, pero al mismo tiempo, abriendo las posibilidades a que un dispositivo que creemos que hace algo, en realidad, haga más de lo que creemos sin darnos cuenta, y, debido a que es parte del funcionamiento normal y esperado de una conexión USB puede pasar inadvertida para las soluciones de seguridad informática actuales.

Esto es lo que BadUSB trata de utilizar a su favor, convertir el funcionamiento normal de una conexión USB en algo beneficioso para un atacante, realizar cambios en el equipo u obtener información y al mismo tiempo, lograr pasar inadvertido.

BADUSB

EL INICIO

BadUSB fue mostrado por tres investigadores alemanes, Kasten Nohl, Sascha Krißler y Jakob Lell, y consiste en la modificación del firmware del microcontrolador que maneja de las conexiones USB de algún dispositivo electrónico con el objetivo de realizar la instalación de malware por medio de la emulación de un teclado, redireccionamiento DNS, entre otras muchas opciones de ataque al mismo tiempo que mantiene la función original del dispositivo para pasar inadvertido por el usuario. [3]

Si bien BadUSB cuando fue presentado se enfocó en pendrives USB, es posible que pudiera ser también escrito para cualquier dispositivo que cuente con conexión USB. Al ver la forma en que se realiza una conexión USB, se puede notar, que por la forma en que se realiza únicamente es necesario cambiar el descriptor de clase, para que un dispositivo sea identificado como uno diferente, y las instrucciones que se deben ejecutar pueden ser escritas directamente en firmware.

Sin embargo, implementarlo implica ciertas dificultades respecto del malware sobre software al que estamos acostumbrados, éstas comienzan con la poca facilidad que se tiene para acceder al firmware de un dispositivo, de entender el mismo, si no se cuenta con conocimiento previo de lenguaje ensamblador, y que no todos los dispositivos USB permiten la actualización del firmware del microcontrolador que realiza las conexiones USB, sin embargo, la amenaza sigue siendo real, puesto que de vencer las dificultades mencionadas, la amenaza es casi indetectable.

ALGUNOS ATAQUES

Cuando BadUSB se mostró por primera vez en julio de 2014, durante la conferencia “Black Hat USA 2014”, los investigadores mostraron solamente como se podía explotar el funcionamiento de USB, y aunque después se liberó el código del mismo en GitHub [4], la dificultad de implementación seguía existiendo, sin embargo, posteriormente, algunas personas comenzaron a experimentar, liberando código con el que las opciones de ataque se multiplican. [5]

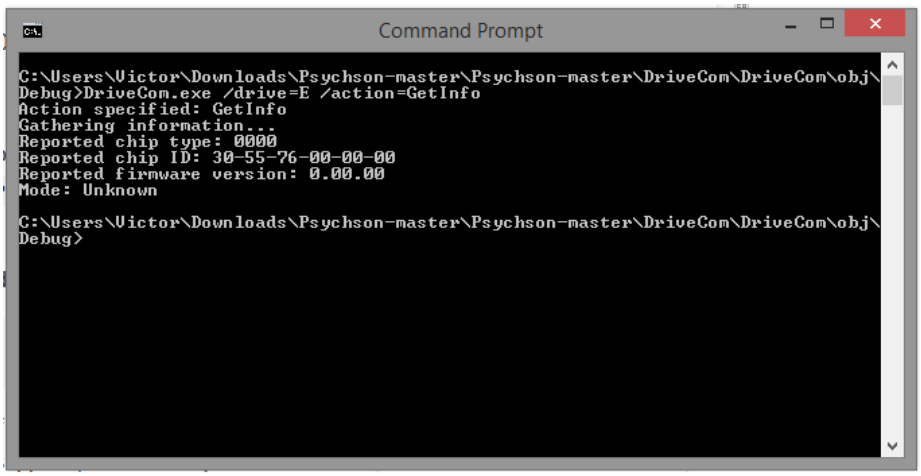
Se mostrarán únicamente dos formas de ataque, de las cuales se puede encontrar código ya implementado y liberado, puesto que implica menor esfuerzo, y hace que las posibilidades de que sea utilizado una vez se comprenda el código y su funcionamiento, aumentan, además de que puedan ser escritas variaciones del mismo si se tienen conocimientos de lenguaje C y una base de lenguaje ensamblador.

SIMULACIÓN DE UN TECLADO USB

Utilizando el sistema operativo Windows, y descargando el paquete con el código liberado que está disponible en GitHub por los investigadores, es posible experimentar con BadUSB de la siguiente manera:

Para comenzar debemos configurar el entorno necesario para las pruebas y compilación. Se debe contar con Visual Studio 2012 (para ésta prueba se está utilizando la versión 2013), puesto que será necesario para compilar las herramientas que se utilizarán, siendo Drivecom la más importante puesto que es la que nos permite sobrescribir el firmware del dispositivo, y se debe instalar Small Device C Compiler, para compilar el firmware, escrito en lenguaje C, además de los parches que se aplicarán.

Para poder saber el tipo de controlador que se tiene, se puede utilizar la herramienta DriveCom.exe, que deberá ser compilada en Visual Studio, indicándole la letra que identifica al dispositivo y la acción a realizar, en este caso, *GetInfo*, para obtener la información del dispositivo indicado (Captura de pantalla 1).



Captura de pantalla 1. Ejecución de DriveCom.exe

PS2251-03 es el único tipo de microcontrolador soportado por el código que fue liberado, sin embargo, el código puede reescribirse para cualquier tipo en caso de tener uno diferente si se busca acerca del firmware y la hoja de datos de dicho dispositivo en internet.

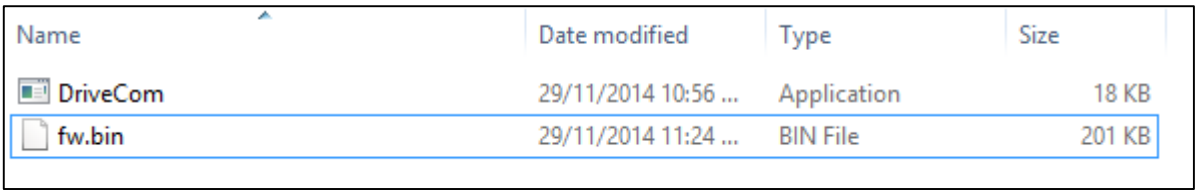
Algunos pendrive USB que se encuentran actualmente en el mercado a la venta, que tienen dicho tipo de microcontrolador y son soportados por el código liberado sin ninguna modificación son los siguientes [6]:

- Patriot 8GB Supersonic Xpress
- Kingston DataTraveler 3.0 T111 8GB
- Silicon power marvel M60 64GB
- Patriot Stellar 64 Gb Phison
- Toshiba TransMemory-MX USB 3.0 16GB
- Toshiba TransMemory-MX USB 3.0 8GB
- Kingston DataTraveler G4 64 GB
- Patriot PSF16GXPUSB Supersonic Xpress 16GB
- Silicon Power 32GB Blaze B30 (SP032GBUF3B30V1K)
- Kingston Digital 8GB USB 3.0 DataTraveler (DT100G3/8GB)
- SanDisk Ultra 16Gb USB 3.0 SDCZ48-016G

Es posible obtener el firmware original, nuevamente utilizando la herramienta Drivecom e indicando la letra que identifica al dispositivo del que deseamos extraer el firmware, además, de indicar el nombre del archivo donde se guardara el firmware leído (Captura de pantalla 2), el archivo se guardara en el mismo directorio donde esté Drivecom (Captura de pantalla 3).

```
C:\t>Drivecom.exe /drive=D /action=DumpFirmware /burner=BN03V104M.BIN /firmware=
fw.bin
Action specified: DumpFirmware
```

Captura de pantalla 2. Comando Drivecom para obtener Firmware.



Captura de pantalla 3. Archivo .bin del firmware que se obtuvo.

Para instalar un firmware diferente, o bien, nuestro firmware con las nuevas instrucciones de ataque, se puede realizar utilizando la herramienta Drivecom e indicando nuevamente, la letra que identifica al dispositivo, además del nombre del archivo que contiene el nuevo firmware (Captura de pantalla 4).

```
C:\t>Drivecom.exe /drive=K /action=SendFirmware /burner=BN03U104M.BIN /firmware=
fw.bin
Action specified: SendFirmware
```

Captura de pantalla 4. Comando Drivecom para instalar Firmware.

Las modificaciones que se realicen al firmware dependerán del atacante y de lo que quiere lograr, pero para fines de éste ejemplo y simular un teclado USB, en el fragmento de código mostrado en la captura de pantalla 5, podemos ver una bandera llamada send_keys_enabled que habilitara o deshabilitara el envío de datos, en caso de que ya hayan sido enviados anteriormente, un contador simple llamado wait_counter y una constante que tiene el valor de espera antes de enviar la secuencia de caracteres deseada(Captura de pantalla 6), llamada KEY_DELAY, cuando esta condición es verdadera, se envían los datos simulando un teclado conectado a un puerto USB.

```
if (send_keys_enabled && wait_counter >= KEY_DELAY)
{
    if (keyData[key_index])
    {
        //Send this key, with some padding before, since something's wonky with endpoint 3
        SendKey(0x00, 0x00);
        SendKey(0x00, 0x00);
        SendKey(0x00, 0x00);
        SendKey(0x00, 0x00);
        SendKey(keyData[key_index], keyData[key_index + 1]);
        SendKey(0x00, 0x00);
    }
    else
    {
        //Wait a while
        wait_counter = 0;
        wait_tick = 0;
    }

    //Move to next key
    key_index += 2;

    //Are we done?
    if (key_index >= sizeof(keyData))
    {
        send_keys_enabled = 0;
    }
}
```

Captura de pantalla 5. Condición para envió de los datos.

```
#define KEY_DELAY 8192
#define KEY_BUFFER_SIZE 0x2000
static const BYTE keyData[KEY_BUFFER_SIZE] = { /*0x00, 0xFF, 0x00, 0xFF, 0x00, 0xFF, 0x00, 0xFF, 0x00,
    0xFF, 0x00, 0xFF, 0x00, 0xFF, 0x00, 0xFF, 0x00, 0xFF, 0x00, 0xFF, 0x00, 0xC3,
    0x15, 0x08, 0x00, 0xFF, 0x00, 0xF5, 0x11, 0x00, 0x12, 0x00, 0x17, 0x00, 0x08, 0x00, 0x13,
    0x00, 0x04, 0x00, 0x07, 0x00, 0x00, 0xFF, 0x00, 0xF5, 0x28, 0x00, 0x00, 0xFF, 0x00, 0xFF,
    0x00, 0xF0, 0x0B, 0x02, 0x08, 0x00, 0x0F, 0x00, 0x0F, 0x00, 0x12, 0x00, 0x2C, 0x00, 0x1A,
    0x02, 0x12, 0x00, 0x15, 0x00, 0x0F, 0x00, 0x07, 0x00, 0x1E, 0x02, 0x1E, 0x02, 0x1E, 0x02,
    0x28, 0x00*/ 0x12, 0x34, 0x56, 0x78 };
int key_index = 0;
```

Captura de pantalla 6. Datos en cola para ser enviados simulando un teclado.

REDIRECCIÓN DNS SOBRE ANDROID

Cuando BadUSB fue mostrado por primera vez también se hizo una demostración de filtrado DNS, que cambiaba el Gateway al que se conectaba un navegador al acceder a determinados sitios web como Paypal.com y antes de llegar a la red, se redirigían las conexiones hacia el pendrive USB que se utilizó para la demostración, siendo una forma viable de robo de información personal, una aplicación un poco más compleja que solamente enviar comandos simulando un teclado USB, pero más beneficiosa para algunos atacantes.

Para que éste ejemplo con Android funcione, los scripts a utilizar son más cortos, pero se necesitan herramientas adicionales, dado que Android puede ser identificado como un adaptador de red por una computadora, se aprovechara esta funcionalidad para ocultar el ataque.

La primer limitación de este ataque sin embargo, comienza con que el teléfono usado debe tener privilegios a nivel de root, debe instalarse busybox, crear un archivo que contenga los dominios host que se desean filtrar y a partir de allí, ejecutar los scripts así que, puede considerarse como un ataque que no se puede realizar sin estar consciente del mismo, pero que igualmente puede ser una amenaza de parte de un atacante que está consciente de ello.

El script básicamente desactiva la interfaz USB, reconfigura lo necesario e interrumpe la reconfiguración hasta el próximo reinicio, por último, con el fragmento de código mostrado en la Captura de pantalla 7, se realiza el filtrado DNS.

```
65  # Configure interface, firewall and packet forwarding
66  busybox ifconfig $INTERFACE inet 192.168.100.1 netmask 255.255.255.0 up
67  iptables -I FORWARD -i $INTERFACE -j ACCEPT
68  iptables -t nat -A POSTROUTING -j MASQUERADE
69  echo 1 > /proc/sys/net/ipv4/ip_forward
70
71  chmod 644 /data/local/tmp/hosts
72  # Start dnsmasq
73  dnsmasq -H /data/local/tmp/hosts -i $INTERFACE -R -S $UPSTREAM_NS -F 192.168.100.100,192.168.100.200 -x $BADANDROID_DIR/dnsmasq.pid
```

Captura de pantalla 7. Fragmento del script de filtrado DNS sobre Android

¿UNA AMENAZA REAL?

Aunque la cantidad de ataques que han demostrado ser posibles utilizando el método aún no es muy grande, el potencial que el método tiene es bastante amplio, podemos imaginar todo lo que es posible hacer a nivel de software, backdoors, troyanos, virus, cualquier tipo de malware como el que estamos habituados a ver puede ser instalado a través de BadUSB, un ejemplo de lo que es posible fue mostrado por Stuxnet, uno de sus predecesores, que es un gusano con funciones rootkit, ocultándose y mostrando sus funciones como legítimas, pionero al utilizar una estrategia similar de infección, sin embargo la limitante del método es, la cantidad de

dispositivos que permiten la actualización de firmware, Phison, el fabricante de los microcontroladores con los que el método de BadUSB ha sido probado tiene un 5% del mercado de microcontroladores que manejan conexiones USB. Aunque aún no se ha probado con otros fabricantes la amenaza es real, pero su propagación puede ser limitada debido a que no todos los fabricantes cuentan con las mismas normas de seguridad y por lo tanto, algunos podrían estar protegidos contra la actualización del firmware. [7] El problema con limitar la actualización de firmware está en que perdemos cierta versatilidad si se desea parchar algo, o agregar funcionalidades al dispositivo.

Pero, otro punto a tomar en cuenta, es la liberación del código de BadUSB, no es necesario tomar el camino largo, y tomar en nuestras manos el trabajo de hacer la ingeniería inversa de algún dispositivo para comprobar que es cierto, con suerte, ya habremos adquirido un dispositivo compatible con el código que se liberó, y esto hace que muchas personas curiosas empiecen a experimentar, siendo la imaginación la limitante del potencial de ataque en este aspecto.

Hasta el momento, no hay una solución o defensa definitiva al problema que supone BadUSB debido a que, solo solucionan el problema parcialmente. Se ha planteado únicamente como una solución a implementar en los dispositivos USB nuevos, la limitación de actualización de firmware.

CONCLUSIONES

Si bien, a lo largo del trabajo se demuestra que el método es efectivo y es real, la probabilidad de ser infectado por éste tipo del malware es baja por ahora, en el futuro más personas podrían comenzar a experimentar violar la seguridad a este nivel, esto entonces, debería ser un antecedente para los fabricantes de microcontroladores, para que comiencen a implementar medidas más rigurosas de seguridad en sus dispositivos.

Aunque podría ser visto como un punto de vista extremista, de la misma manera en que no confiamos de todo el software que podemos encontrar en línea para descargar a menos que tengamos seguridad de su origen y sus funciones, en el futuro, podríamos comenzar a pensar en tener esa misma precaución con los dispositivos electrónicos que adquirimos que tienen dudosa procedencia.

A nivel de software, algunos de los posibles ataques que éste método puede realizar pueden ser bloqueados con alguna solución de seguridad informática, pero, en un intento por crear malware de manera más creativa, ejecutar acciones durante el arranque del sistema operativo podría ser posible, por lo que mejorar la seguridad a todos los niveles es siempre algo para tomar en cuenta.

REFERENCIAS

1. Stuxnet WhitePaper Updated, Andrew Ginter
   http://blog.industrialdefender.com/?p=516

2. Serial Programming/USB
   http://en.wikibooks.org/wiki/Serial_Programming/USB.htm

3. BadUSB – On accessories that turn evil, SR Labs
   http://srlabs.de/blog/wp-content/uploads/2014/11/SRLabs-BadUSB-Pacsec-v2.pdf

4. GitHub – adamcaudill / Psychson
   http://github.com/adamcaudill/Psychson.htm

5. It's a case of good news/bad news with the BadUSB firmware exploit, Roger Grimes
   www.infoworld.com/article/2692408/data-security/badusb-is-deadly-but-hackers-wont-use-it.html

6. GitHub – adamcaudill / Psychson, Known Supported Devices
   http://github.com/adamcaudill/Psychson/wiki/Known-Supported-Devices.htm

7. Análisis de BadUSB, la nueva amenanaza (que no es apocalipsis), Josep Albors
   http://www.welivesecurity.com/la-es/2014/08/11/analisis-badusb-nueva-amenaza-no-es-apocalipsis.htm