Raising awareness of cyber security concerns regarding the use of USB peripherals

Marian TICU
National Cyberint Center
marianticu89@yahoo.com

Abstract: The number and variety of types of attack vectors used by hackers are steadily increasing. The cyber attackers are constantly trying to adapt to the protective measures taken as a response to their actions and are searching for new methods to infect a system apart from the traditional infection methods that require the user to click on a link or run a file. One of these methods is the use of a USB device, specially built to incorporate malicious activities.
There are currently both open-source projects and commercial products that use programmable USB devices which can be configured for malware purposes. The attacker can choose between building its own device or buying it. Either way, we need to be informed in order to be prepared for proactive defense against these cyber-attacks vectors.
Keywords: awareness, cyber-attacks, rogue, USB, device, kill-switch, protection, malicious

INTRODUCTION

The USB protocol was first developed in 1996 by USB-Implementers Forum (USB-IF) in order to establish an universal communication protocol between different kinds of IT&C systems. Over the time, the protocol has gained a widespread popularity through its continuous improvement in terms of speed performance and small form factor interface size. In the nowadays, the USB physical interface or the USB port, as it is commonly referred to, it is being used to connect different kinds of devices (in example: hid, input/output, storage, audio-video or IoT devices) to a large set of hosts (in example: PCs, laptops, servers, tablets, smartphones and more). Starting from the versions USB 1.0 and 1.1 with speeds as low as 1.5 Mbps, the protocol specifications evolved over the years with version 2.0 reaching a maximum speed of 480 Mbps and with the version 3.0 that can transfer data at the maximum rate of 5Gbps. The latest version, USB 3.1, can make use of a maximum bandwidth of 10Gpbs The latest version, USB 3.1, can make use of a maximum bandwidth of 10Gpbs, that reinforces and strengthens once more the USB protocol in the leading position of the interoperability and data transfer ranking. One special particularity is that newer versions are backward compatible with the older devices meaning that a USB 2.0 device can be plugged and used with a superior USB 3.0 capable system.

The USB protocol is a host controlled communication which means that each and every transaction is first initiated by the host. In order to establish a USB transfer it requires the presence of two types of controllers: host and device. Two hosts can't exchange data directly, nor two devices with the exception of OTG (On-The-Go) devices. In the OTG, one device is setup as master and the other one as slave. In the USB protocol, the host is responsible for

initiating all the transfers. There are four types of transfers with different functionalities and specifications: control, bulk, interrupt, isochronous. The control transfer type is used when the host exchanges data with device in order to identify the device capabilities in order to assign the proper driver for it. The bulk is the preferred transfer mode when time is not critical. It is the fastest transfer mode and it is being used to transfer large files to devices like printers or disk drives. The interrupt transfer mode is used by HID devices (mice, keyboard, game controllers) to transmit the data without any delay based on the device generated interrupts. The isochronous transfer type it is being used for data streaming in cases when the data must be transferred at constant rates within a time limit and with an error acceptance or tolerance. Some of the devices that uses this transfer mode are audio and video devices.

Each USB device holds a descriptor hierarchy that is used by the host to identify the device and discover its capabilities. There are five common USB descriptors: device, configuration, interface, endpoint and string descriptors. An USB device can have only one device descriptor and this can offer contextual information about the device connected to the host. It holds a data structure that includes the vendor ID and product ID, the USB version, the maximum packet size and the number of the configurations defined on the device. Although the device can support multiple configurations, only one can be enabled at once. For each configuration, the descriptor specifies electrical connection details and interface information. The interface is associated with a device capability or functionality specified by some well known values for protocol, class or subclass fields in the interface descriptor. For example, if the class field in the interface has the value 0x03h then we can certainly conclude that the device attached to the host is an HID - human interface device, probably a keyboard or a mouse. If the same class field contains the value 0x07h then the device connected to the system is probably a printer.

A CYBER SECURITY PERSPECTIVE OF THE WELL-KNOWN USB BASED ATTACKS

Cyber attacks using USB devices are constantly expanding over the years and are using increasingly diversified methods. The attackers have turned their attention to develop custom devices or reprogramming existent USB devices for malicious purposes. This fact leads to a new infection method which is hard to identify even by highly skilled IT personnel because the devices, also known as "bad peripherals", act, look and feel as regular but it also incorporates other hidden functions.

Depending on the actions targeted by the hackers, the bad USB devices can be grouped into the following self-describing categories:

- Keystroke injection
- Malware delivery
- Data exfiltration
- Network traffic hijack
- Electrical damage
- Data alteration
- Video sniffing

The following section introduces a series of USB-based cyber-attacks along with their operating principles and associated category tags:

a) Rubber Ducky - operating principles: the device acts as a normal USB mass storage device and includes a hidden keyboard that can be used to send commands to the host computer by injecting keystrokes. The storage space can be used to deliver malware or to exfiltrate data to/from the host. Tags: keystroke injection, malware delivery, data exfiltration.

b) BashBunny - operating principles: small form factor computer running Linux and powered from the USB bus that can be used to simulate a keyboard or a mouse, a mass storage device or an USB external network card. Tags: malware delivery, keystroke injection, data exfiltration, network traffic hijack, data alteration.

c) PHUKD/URFUKED - operating principles: similar functions as the Rubber Ducky but can be configured to add a time delay to the malicious keystrokes injection. Tags: keystroke injection, malware delivery, data exfiltration.

d) USBdriveby - operating principles: emulates a keyboard and a mouse in order to install a backdoor

on a MacOS computer; it can also change the DNS settings of a host. Tags: keystroke injection.

e) Evilduino - operating principles: it uses an Arduino microcontroller to launch similar attacks as the PHUKD/URFUKED. Tags: keystroke injection, malware delivery, data exfiltration.

f) Unintended USB channel - operating principles: it can be used to exfiltrate data from a host computer by toggling the leds on a USB keyboard or by playing special sounds on speakers. Tags: data exfiltration.

g) TURNIPSCHOOL (COTTONMOUTH-1) - operating principles: special USB cable that hides inside it a microcontroller capable of sending and receiving data over RF communication. Tags: keystroke injection, data exfiltration, malware delivery.

h) RIT attack via USB mass storage - operating principles: the Read It Twice attack allows an attacker to install malware software as an update. This can be achieved because the "check firmware" and "install firmware" phases of the update process aren't executed in a single atomic operation, and the update package can be substituted after the "check" process. Tags: malware delivery.

i) KeySweeper - operating principles: it uses an Arduino to wirelessly sniff, decrypt, log and report the keystrokes from any Microsoft wireless keyboards. Tags: data exfiltration.

j) Default Gateway Override - operating principles: uses a microcontroller as an USB Ethernet adapter and can override DHCP settings in order to capture network traffic. Tags: network traffic hijack.

k) Smartphone-based HID attacks - operating principles: Android USB gadget driver that can simulate the keyboard and mouse. Tags: keystroke injection.

l) DNS override - operating principles: similar to the Default Gateway Override infection method, but this device overrides the DNS settings. Tags: data alteration.

m) Hidden Partition - operating principles: the USB flash drive has a hidden partition that can be used to save and exfiltrate data on it. Tags: data exfiltration.

n) Virtual machine break-out - operating principles: the firmware on a USB device can be used to evade from a virtual machine environment. Tags: malware delivery, data exfiltration.

o) Boot Sector Virus - operating principles: reprogrammed USB device can be used to infect a computer before boot. Tags: malware delivery.

p) iSeeYou - operating principles: the Apple internal iSight webcam can be reprogrammed to covertly capture the video without lighting the LED. Tags: video sniffing.

q) Autorun Exploits - operating principles: it can run specific applications from a flash drive if autorun is enabled. Tags: malware delivery.

r) Cold Boot - operating principles: it can dump data from RAM by fast rebooting the system and booting from a USB device with memory dump software. Tags: data exfiltration.

s) USB Thief - operating principles: the attacker infects portable software that can be run from an USB mass storage device; after being executed the malware will stole data and save it to the flash drive. Tags: data exfiltration.

t) USBee attack - operating principles: it is being used to exfiltrate data using an USB connector that can transmit data over electromagnetic emissions. Tags: data exfiltration.

u) USB Killer - operating principles: special USB device that can deliver an electrical surcharge to the target computer that conducts to damaging the system. Tags: electrical damage.

In addition, we can summarize that almost all the bad USB devices covered in this article, with small differences, share almost the same operating principles that involves reprogramming a microcontroller to emulate HIDs (keyboard and mouse), Mass Storage (flash drive) or RNDIS (network card) devices allowing the attacker to run predefined commands on target hosts, install and execute malware or sniff and steal data.

IS THERE ANY WAY OF IDENTIFYING THE ROGUE USB DEVICES?

Regardless of the above-mentioned methods and devices that an attacker might be using, except the USB killer, the data travels on the USB bus from host to device or vice-versa. I would describe the need as the possibility to unveil all hidden features of an USB device before using it while still maintaining considerable interest for capturing and analyzing the content of the exchanged data. Even though we are dealing with

some rogue USB devices which besides standard features they also incorporate some other hidden capabilities, in order to work on a target computer, the devices must be fully compliant with the USB protocol. In other words, this means that we can use the existing USB protocol debugging tools for evaluating and identifying USB functionalities including the hidden ones. From a technical perspective this can be accomplished by analyzing all the interfaces published by an USB device in the enumeration phases at the device, configuration and interface levels. In example, a standard storage device usually have only interfaces registered in the 0x08h base class which is associated to mass storage devices. The recording of another type of interfaces other than the ones intended for the scope of device, for example a 0x03h class interface which is associated with a keyboard or a mouse and it is registered by an USB memory stick, should be a relevant starting point for identifying rogue devices. What if we are dealing with a rogue keyboard registered as a normal standard HID device but it has malicious keystroke injection capabilities? In this particular case, in order to distinguish the legitimate user activity from the illegal, we must focus on capturing, analyzing and extracting information from the USB traffic. The activity of capturing the USB traffic is relatively easy compared to its analysis operation that requires advanced knowledge and skills in the field of USB protocol comprehension.

HOW CAN WE PROTECT OURSELVES OF NOT BECOMING A TARGET OF AN USB ATTACK?

INCREASED ATTENTION WHILE USING DOCKING CHARGING STATIONS IN PUBLIC PLACES

Our lives are increasingly interconnected with digital devices. In certain situations, we rely solely on the information provided by the electronic devices we own and we are conditioned by their availability and state of operation. Whether we are referring to an electronic boarding pass required for a flight, or do we need to use the GPS application on our smartphone to guide us in an unknown city, or do we need to access and respond to a critical e-mail message, these are just a few real life situations when our devices are indispensable in our lives. Every mentioned aspect is restricted around the availability of the electronic devices we are accustomed to, and in turn they are conditioned on the battery life. We often get to use the docking charging stations available in public places such as airport lounges, trains or waiting rooms. We use these charging facilities, even if our devices are partially charged whenever we have spare time, because we may not know when we may need them and we like to keep our electronic devices online as long as possible. Of course, the benefits of using these charging docks are visible but do we take into consideration the concerns regarding cyber security? Probably, not. Is there any possibility that these charging docks could be tampered by potential attackers? Probably, yes. How can we still use the charging docks without the fear that it could be infected and it could infect our devices? The solution is pretty straight-forward in terms of using charge-only cables or adapters. The USB socket has four pins, the pins inside, in the middle, are responsible for data transfer, while the outside pins are used to provide a 5 volt power supply. A charge-only cable uses only the charging pins. This is a workaround solution and it is not recognized by the USB Implementers Forum.

DO WE PROTECT OURSELVES FROM INFECTING OUR DEVICES OR DO WE PREVENT INFECTING OTHERS THROUGH OUR INFECTED DEVICES?

Starting from the hypothesis that an infected device can infect others we must acknowledge the fact that we can both fulfill two roles: one as a victim and afterwards, without our will, we can play the role of the attacker. The widespread of the USB protocol across a large set of devices and the diversified workflows and scenarios that assumes plugging in the same device between different information systems could introduce new concerns regarding cyber security. We adopted and already using in our lives a wide range of IoT, wearables, gadgets, fitness, health monitoring devices and more. Lots of them are using the USB protocol for transferring data. These devices are plugged in and plugged out numerous times to a lot more set of devices. Hypothetically speaking, what could

happen if all of these could get infected one day? We enjoy listening our favorite songs while driving the car from an USB memory stick. We could be using the same storage device to transfer some data to or from a computer, an IoT, a smart home automation solution, a video surveillance system and to a lot more set of devices. Some of the systems are not critical and their malfunctioning is not a cause for concern, but others are and our imagination is just a limit for scenarios that could happen. Maybe our climate control system could be hacked one day through an USB device when loading or storing configuration or firmware updates. By analogy, the theory can be extended to any equipment that has an UBS port, starting from home use apparels like a washing machine, an electric over or the alarm and the video surveillance systems and reaching topics that could go as far as infecting a car's entertainment system. Taking into consideration all these facts, is it justified to be more aware and to act with caution when we are establishing connections and we are making transfers between devices? Well, we probably should. In addition to all the mentioned aspects, we should not, whatsoever, connect lost and found USB capable devices to our systems. Not even from curiosity. The reasons explained above are, for the time being, purely theoretical, and have the role of raising awareness and stirring up talks rather than proposing solutions.

PROPOSALS FOR PROTECTION METHODS AGAINST USB-BASED CYBER ATTACKS

The most common scenario encountered in practice addresses the infecting of common computer systems, whether we refer to a PC, laptop or server. In this section we will take into consideration the possible solutions to our problem and we will discuss advantages or disadvantages.

One of the first approaches refers to limiting the use of the USB devices on a host to a restricted subset of well-known devices. This thought has already been implemented by many software solutions which basically are using a previously defined whitelist of allowed devices, blocking the communication for each and every other device. This kinds of software are building a whitelist using some unique fields that specifically identify one device. Nonetheless, there is an workaround because special crafted USB devices can be forged to spoof any required field in order to bypass the protection. Speaking in terms of directly connecting the USB device to the host, it could also theoretically exploit other sections of the protocol itself before reaching the phase where the software decides if the inserted device matches a record in the whitelist. If the exploitation succeeds, it could also disable any USB protection software and giving it a free undisturbed pass to the OS. Although it provides a certain level of limited security, this solution can prove useful in preventing cyber-attacks launched from unknown devices.

Another way of thinking a defense mechanism is to introduce another node in the communication channel. The design is similar to the proxy concept. Basically, a new device is inserted on the communication path between the device and the host itself and has the job to forward each packet to destination in a similar man-in-the-middle configuration. The proxy device must have both types of USB controllers, host and device, in its hardware configuration. In a theoretical model, the proxy solution could act as an Intrusion Prevention System borrowed from the computer networks domain. It could detect and block any USB transaction that matches a previously defined rule or it could identify any traffic anomaly starting from a baseline traffic. There are some experimental solutions developed by various computer scientists or industry specialists. Although this solution model is very advanced and can be used to detect and take actions on both statistical and anomaly data, it has some flaws. Given the circumstances that the proxy software solution runs on top of a operating system, one could directly exploit the operating system using an USB device before reaching the higher levels of the proxy application and infect it with malware. After infection, this could spread to the final host which we are trying to protect.

The last theoretical method proposed in this paper is the conclusion resulted from the previous two discussed methods. This means that we are taking into consideration the facts that installing an USB protection software directly on the host

or adding a new proxy in the USB communication path could improve protection but it won't fully solve the problem because the rogue USB device could exploit the host operating system at a lower level, before reaching the protection levels implemented by the application. The last method implies sniffing the USB traffic between the device and the host and mirroring it to a promiscuous capturing interface of a totally isolated system. This system should have the capability to analyze the USB traffic in a similar mode as an IPS, in a time slice as close as possible to near real time. In the eventuality of matching any USB transaction packet with some previously well-defined rules, the system should take the decision to physically interrupt the USB cable between the device and the host and thus prevent other further actions. This approach acts like an controlled USB kill switch depending on fulfilling of certain conditions. The proposed method can prove to be very useful if the system is fast enough to shut down the USB link as fast as possible after detecting a pattern matching a signature or an anomaly.

CONCLUSIONS

USB-based cyber-attacks are a reality of our times and we need to understand the risks we and our systems pose when using USB devices from unknown sources or when connecting our own USB devices to other systems. This paper offers a short brief over some of the USB based cyber-attacks and contributes to the general improvement of the awareness in term of cyber security.

Although in the field of cyber security there is no security solution that can guarantee a success rate of 100%, the methods proposed in this article can add an extra layer of security to your systems. It is always better to prevent than to correct possible actions of malicious rogue USB devices.

The rogue USB devices intended for malware purposes are using legitimate mechanisms in the USB protocol to perform both normal and malicious activities. These devices take advantage of the fact that the USB protocol was built to provide support for composite or multi-feature devices by implementing multi-interface configurations. Taking into consideration the fact that the USB protocol allows them to perfectly disguise along legitim devices, proves to be very difficult to defend against the actions performed by these devices. Device filtering using whitelisting proves to be inefficient because the attackers can spoof almost every string or parameter in the descriptor configurations.

The unpleasantness caused by these devices could be avoided by implementing a near real time USB traffic analyzer capable of matching and identifying patterns using a principle similar to a network IPS. The protection could be accomplished through the use of an USB kill switch which refers to physical interrupting or disconnecting the communication channel between the host and de device.

References

[1] Brian Anderson, Barbara Anderson. (2010). *Seven Deadliest USB Attacks*. Oxford, GB: Syngress, Elsevier, INC.
[2] Kamkar, S. (2018, 07 01). KeySweeper. Retrieved from samy.pl: https://samy.pl/keysweeper/
[3] Kamkar, S. (2018, 07 01). *USB driveby*. Retrieved from samy.pl: http://samy.pl/usbdriveby/
[4] Karsten Nohl, Sascha Kribler, Jakob Lell. (2018, 07 01). BadUSB - *On accessories that turn evil*. Retrieved from Security Research Labs: https://srlabs.de/wpcontent/uploads/2014/07/SRLabs-BadUSB-BlackHat-v1.pdf
[5] LLC, H. (n.d.). Hak5. Retrieved from BashBunny Documentation: https://www.hak5.org/gear/bash-bunny/docs
[6] Matthew Brocker and Stephen Checkoway, Johns. (2014). *iSeeYou: Disabling the MacBook*. Proceedings of the 23rd USENIX Security Symposium. San Diego, CA: USENIX Association.
[7] Nir Nissim and Ran Yahalom and Yuval Elovici. (2017). *USB-based attacks*. Computers & Security, 675-688.
[8] TURNIPSCHOOL. (2018, 07 01). Retrieved from NSA Playset: http://www.nsaplayset.org/turnipschool
[9] USB KILL. (2018, 07 01). Retrieved from https://www.usbkill.com/
[10] Zhaohui Wang, Angelos Stavrou. (2010). *Exploiting smart-phone USB connectivity for fun and profit*. ACSAC '10 Proceedings of the 26th Annual Computer Security Applications Conference (pp. 357-366). Austin, Texas, USA: ACM New York, NY, USA ©2010.
[11] USB Made Simple. (n.d.). Retrieved from http://www.usbmadesimple.co.uk/