International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-2 Number-4 Issue-7 December-2012

A Low Cost Hardware Trojan horse Device based on Unintended USB channels and a Solution

Pravin Phule
Department of Computer Engineering PICT, Pune, Maharashtra, India

Abstract

Nowadays every device is becoming available as a USB device. As a result of that these devices may be used to attack a network endpoint. This paper aims at implementing a Hardware Trojan horse device which when used by a malicious insider can attack a network endpoint to steal the confidential information over unintended USB channels. Endpoint Security Solutions are available to protect the stealing of information through USB Mass Storage, USB Printer interfaces but still they have no control over the USB audio and USB keyboard interfaces. So these interfaces can be used in an unintended way to attack a network endpoint and steal the information. This paper also focuses on methodologies that can be applied to block the unintended USB channels.

Keywords

Insider Threat, Hardware Trojan horse, Unintended USB channel, Human Interface Device

1. Introduction

USB peripherals provide almost all types of functionalities, from storage to Human Interface Devices. The USB Specification defines a single physical interface and a base protocol to be used for all USB devices [1]. USB devices are plug and play which means that computer system already contains the driver required to configure the device. Due to the wide use of Plug and play it has become the preferred way of attacking a computer system using Auto Run and Auto Play features [2].

Many security solutions are available to provide data theft protection from USB devices [3, 4, 5, and 6]. Endpoint security solutions (ESSs) are used to protect corporate data. Using Access Control Lists (ACL) ESSs can allow specific users or employees to access specific device but not all the devices. Such protection works well only if access is allowed to the devices whose serial numbers are available in ACL.

But ESSs are not well while regulating Human interface devices such as a USB mouse, USB keyboard, USB headsets or speakers etc. [3, 4, 5, 6]. So these interfaces can be exploited to develop Hardware Trojan horse Device and attack the network endpoint to steal the confidential information of an organization.

A USB channel becomes unintended when it is used to do something which is not defined by USB protocol [7]. For example consider the communications between a USB keyboard and a network endpoint. USB keyboard uses two intended USB channels: one to send key presses from USB keyboard to network endpoint and another to send the state of the keyboard LEDs (Caps Lock, Number Lock and Scroll Lock) from network endpoint to USB keyboard. An application on the network endpoint could create an unintended USB channel by causing the exfiltration of data in the form of toggling keyboard status LEDs [7].

2. Related Work And Literature Review

In 2005, D. Barral and D. Dewey [9] introduced a concept of USB Meta-Device. USB protocol relies on a USB device for its identification. Due to that a USB Meta-Device can programmed to identify itself as any USB device. USB Meta-Device could be programmed to identify itself as a device associated with a vulnerable driver loaded on the network endpoint. The approach used in this paper differs from the USB Meta-Device in that it does not require the presence of a vulnerable driver on the network endpoint.

In 2006, M. Al-Zarouni[10] proved that USB Auto Run and Auto Play features can be used to steal data and automatically execute harmful code on a network endpoint without user's knowledge. Endpoint security solutions are available to deal with this type of risk. The approach used in this paper differs in that Hardware Trojan horse Device can be used in the presence of Endpoint Security Solutions, as they generally do not regulate Human Interface Devices.

In 2006, G. Shah, A. Molina, and M. Blaze [11] introduced the concept of Keyboard Jitterbug. The keyboard Jitterbug device is also a way to exfiltrate the data. It can be inserted between keyboard and network endpoint to act as key logger. It can also be used to exfiltrate the data by adjusting the timing information of packets sent over the internet. However, it is different from the approach used in this paper because it requires that the network endpoint to maintain an interactive session through the Internet. Hardware Trojan horse Device can be used to exfiltrate data without relying on internet connectivity.

In 2009, J. Clark, S. Leblanc and S. Knight [7] introduced the concept of Hardware Trojan Horse Device based on Unintended USB channels and in 2011 they [6] extended the concept with the Risks Associated with USB Hardware Trojan Devices used by Insiders. This research [6, 7] is directly related to the approach used in this paper. But it has used PLX's Net 2280 Rdk [12] to develop a Hardware Trojan horse device which is costly device. As proposed in this paper it will use a low cost device: mbed NXP LPC11U24 Microcontroller [13] to develop a Hardware Trojan horse device. Also, this paper proposes a solution to block the unintended USB channel.

3. Concept of Hardware Trojan Horse Device

An application on the network endpoint could create an unintended USB channel by causing theexfiltration of data in the form of toggling keyboardstatus LEDs Similarly an application on the network endpoint could create an unintended USB audio channel using Isochronous Out and exfiltrate the data in the form of audio packets [1]. Refer Figure 1 for more details.

Only Human Interface Devices such as USB keyboard, USB speakers, headsets, USB mouse etc. are considered for this project. Because previous research has shown that these are the only USB devices which are not regulated by Endpoint Security Solutions. Other USB devices such as USB flash, USB printer etc. are well regulated by Endpoint Security Solutions [1, 2, and 3].

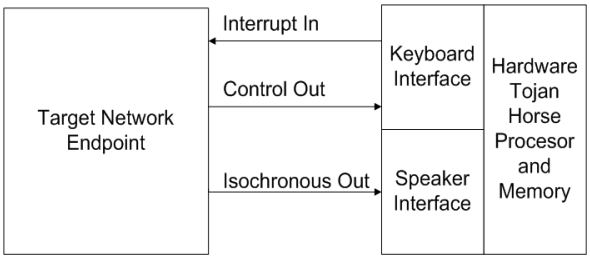

Figure 1: Concept of Hardware Trojan horse Device [7]

4. Proposed Solution

The solution can be implemented as software on the network endpoint. This software will keep a watch on what type of data is sent over USB keyboard and USB audio channel. If it found that some data is exfiltrated to USB keyboard or USB audio channel then immediately that activity is stopped or blocked. Refer Figure 2 for more details.

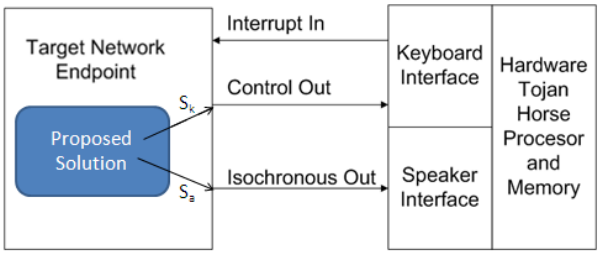

Figure 2: Proposed Solution

5. Implementation Methodology and Feasibility Assessment

There are four modules needed in the project as given below:

- Uploading Malicious code to network endpoint
- Steal data over the unintended USB keyboard channel and unintended USB audio channel
- Track and block unintended USB keyboard channel
- Track and block unintended USB audio channel

Uploading malicious code to network endpoint:
This module is to be implemented as a part of keyboard interface which will reside on mbed LPC11U24 microcontroller. This module will upload a malicious code to the network endpoint. The

algorithm for the same is given below.

```
//UploadCode(Uk, Dk) uploads malicious code over
//unintended USB keyboard channel to the network
//endpoint
//where Uk is an unintended USB keyboard channel
//Dk is a USB keyboard device or a device which is
//identifies itself as USB keyboard
Algorithm UploadCode(Uk, Dk)
{
While (there exist a character c in a file which
contains malicious code)
        {
                sendchar(c);
//sendchar(c) sends a character of //malicious code to
network //endpoint in the form of key press //events
}
}
```

Steal data over the unintended USB keyboard channel and the unintended USB audio channel:

This module will be a malicious code uploaded by UploadCode(U_k, D_k) which after executing on network endpoint will be able to encode data in the form of toggling keyboard LEDs and send it over the unintended USB keyboard channel as well as embed data in audio packet and play it over an unintended USB audio channel.

```
//Where U is an unintended USB channel
//Uk is an unintended USB keyboard channel
//Ua is an unintended USB audio channel
Algorithm stealData(U,D)
{
        //let f is the file to be stolen
        If (U=Uk)
        {
        While (there exist a character c in f)
                {
                        sendcharOverUk (c);
//sendcharOverUk(c) sends //a character of f over
Uk//in the form of toggling LEDs
}
        }
        Else If (U=Ua)
        {
//let a be the audio packet to be sent //overUa
                a = embedFileInAudioPacket(f);
//embedFileInAudioPacket(f) //embeds file f in an
audio packet //'a'
PlaySound(a);
//PlaySound (a) sends an audio //packet over Ua
}
}
```

Track and block the unintended USB keyboard channel:

This module is to be developed as a part of the security solution which will reside on network endpoint and will be able to detect and block the unintended USB keyboard channel. An algorithm for the same is given below.

```
//Let I be an illegal activity
//Let Ik be an illegal activity performed using Uk
Algorithm trackAndstopStealing(Ik)
{
        While (true)
        {
If (there exists an illegal activity Ik)
                {
Block the unintended USB channel Uk ;
}
}
}
```

Track and block the unintended USB audio channel:

This module is to be developed as a part of the security solution which will reside on network endpoint and will be able to detect and block the unintended USB audio channel. An algorithm for the same is given below.

```
Algorithm trackAndstopStealing(Ia)
{
        While (true)
        {
If (there exists an illegal activity Ia)
{
Block the unintended USB channel Ua ;
}
}
}
```

Algorithm trackAndstopStealing(I_k /I_a) is an NP-Hard problem [14]. Because it needs to track for an illegal activity (I_k /I_a). If there exists an unintended USB channel (U_k/U_a) and device (D_k/D_a) attached to that channel then we can say that there is some illegal activity going on. But it is not possible to recognize whether the USB channel being used is intended or unintended. So providing a solution to block an unintended USB channel becomes an NP-Hard problem.

Previous research [6, 7] proved that USB keyboard channel became unintended when it is used to exfiltrate data in the form of toggling keyboard LEDs. Toggling of keyboard LEDs can be detected

and can be blocked. So algorithm trackAndstopStealing(I_k) can be reduced to tracking for continuous toggling of keyboard LEDs and if found then stop it which can be detected and solved in polynomial time. Algorithm trackAndstopStealing(I_k) can be redefined as follows:

```
//Let I be an illegal activity
//Let I_k be an illegal activity performed using U_k
Algorithm trackAndstopStealing(I_k)
{
        While (true)
        {
If (keyboard LEDs toggling continuously)
                {
Stop continuous toggling of keyboard LEDs;
}
}
}
```

Previous research [6, 7] also proved that USB audio channel became unintended when it is used to exfiltrate data in the form of audio packets. We can record the audio being played and if it contains information other than audio then that audio can be blocked from playing. So algorithm trackAndstopStealing(I_a) can be reduced to recording the audio and if it found that audio packet contains information other than audio which can be detected and solved in polynomial time.

Algorithm trackAndstopStealing(I_a) can be redefined as follows:

```
//Let Ia be an illegal activity performed using U_a
Algorithm trackAndstopStealing(I_a)
{
        If (an audio packet contains information
        other than audio)
        {
                Block that audio from playing;
        }
}
```

6. Proposed Outcome of Project

The outcome is a Hardware Trojan horse device and software which will provide a solution to block exfiltration of data through unintended USB channels to the attached Hardware Trojan horse device as shown in Figure 2.

A Hardware Trojan horse device can easily attack the network endpoint as follows:-
A network endpoint's keyboard is replaced by a Hardware Trojan Horse. The Hardware Trojan Horse activates outside the organization's business hours and logs in to the network endpoint. It will then upload the applications necessary to create the Keyboard LED and Audio Channels between the network endpoint and the Hardware Trojan horse, and applications that can open a tunnel and a back door to the Internet. Then Hardware Trojan Horse will carry out its attack. It will perform a search of the network endpoint data, looking for the files containing uploaded keywords and writes the results of the search to a text file. These search results, consisting of the path to files containing keywords, are exfiltrated to the Hardware Trojan Horse using the Keyboard LED Channel. The search result will be analyzed by the Hardware Trojan horse, which will then cause the exfiltration of the files of interest from the network endpoint to the Hardware Trojan Horse using the Audio Channel [6, 7].

Exfiltration of files from network endpoint to attached device in the form of Keyboard LED status messages can be blocked as follows:-
Whenever there is no key press corresponding to NUM LOCK, SCROLL LOCK, and CAPS LOCK but still there is something that is toggling keyboard LEDs then such process can be caught and proposed software can disturb and consequently block such activity.

Exfiltration of files from the network endpoint to the attached device through the USB audio channel can be blocked as follows:-
Whenever audio is played proposed software can check what type of information is transferred to audio device in the form of audio packet and if it found that some data is exfiltrated to the USB audio channel then it can block that activity.

7. Conclusion

The researchers are not completely aware of the risks associated with USB devices. This paper focuses on the risk associated with unintended USB channels and proposes the possible solution to mitigate this risk. This paper also discusses the feasibility of the implementation using P, NP, NP-Hard and NP-Complete models. All the algorithms are proposed regardless of any technology or programming language.

Real time protection is nothing but addressing the threat before it becomes a problem [8]. This paper will address an insider threat before it becomes a widely used way to steal the corporate data.

Also it focuses on the deficiencies of Endpoint Security solutions. They rely on a device for identification. The device may identify itself as any Human Interface Device and can be used to steal information. Endpoint Security solutions have control over USB Mass Storage, USB Printer interfaces. But now there is a need that they should also regulate the data or information transferred over the USB keyboard or USB audio channels.

Acknowledgment

I would like to thank my college Pune Institute of Computer Technology, Pune, Maharashtra, India for blessing me the environment which is perfect for research. I would also like to thank the author J. Clark for his quick and accurate response to my queries about his research based on Unintended USB Channels.

References

[1] USB Implementers Forum, "USB 2.0 Specification," 2001, http://www.usb.org/developers/docs.
[2] S. Stasiukonis, "Social engineering, the USB way," 2006, http://www.darkreading.com/security/article/208803634/social-engineering-the-usb-way.html.
[3] Centenial Software, "DeviceWall: Endpoint Security homepage," 2012, http://www.frontrange.com/ProductsSolutions/Detail.aspx?id=9416.
[4] CheckPointSoftware, "Pointsec protector homepage," 2009 http://www.checkpoint.com/products/datasecurity/protector.
[5] DeviceLockInc., "Devicelock homepage," 2009, http://www.devicelock.com.
[6] J. Clark, S. Leblanc and S. Knight, "Risks Associated with USB Hardware Trojan Devices used by Insiders," in IEEE International Systems Conference (SysCon), April 2011.
[7] J. Clark, S. Leblanc and S. Knight, "Hardware Trojan horse device based on unintended USB channels," in 3rd International Conference on Network and System Security, pages 1-8, IEEE Computer Society, May 2009.
[8] Microsoft, "Microsoft Security Essentials Product Information page", 2012, http://windows.microsoft.com/en-US/windows/products/security-ssentials/product-information.
[9] D. Barral and D. Dewey, ""Plug and Root", the USB Key to the Kingdom," 2005, http://www.blackhat.com/presentations/bh-usa-05/BHnUSn05-Barrall-Dewey.pdf.
[10] M. Al-Zarouni, "The reality of risks from consented use of USB devices," in Proceedings of the 4th Australian Information Security Conference, Sep. 2006, pp. 5–15.
[11] G. Shah, A. Molina, and M. Blaze, "Keyboards and covert channels," in Proceedings of the 15th conference on USENIX Security Symposium, 2006.
[12] PLX Technology, "Net2280 home page," 2008, http://www.plxtech.com/products/net2000/net2280.asp.
[13] mbed, "mbed NXP LPC11U24 home page," 2012, http://mbed.org/handbook/mbed-NXP-LPC11U24.
[14] Eric W. Weisstein, "NP-Hard Problem," 2012,http://mathworld.wolfram.com/NP-HardProblem.html.

Pravin Phule, Pune, India, born on 13 May 1986, Post Graduate Student at Pune Institute of Computer Technology, Pune, Affiliated to the University of Pune, Maharashtra, India