PRINTSHOP: ASSESSING OS AND CAPABILITIES OF SERIAL PRINT DEVICES

Research Proposal

Doctor of Philosophy

in

Cyber Operations

September 21, 2023

By

Micah Flack

Beacom College of Computer and Cyber Sciences

TABLE OF CONTENTS

Table of Contents **ii**

List of Tables **iii**

List of Figures **iv**

1 Introduction **1**

1.1 Overview 1

2 Related Works **4**

2.1 RTOS: Software and Security 4

2.2 Embedded Firmware Patching 4

2.3 BadUSB-like Devices 5

2.4 Summary 6

3 Proposed Research **7**

3.1 Research Objectives 7

3.2 Research Questions/Hypotheses 7

3.3 Methodology 7

4 Timeline **9**

5 Conclusion **10**

References **11**

LIST OF TABLES

Table 4.1 Research lifecycle 9

LIST OF FIGURES

Figure 1.1 Comparison of common PoS systems 2

Figure 1.2 SNBC BTP-S80 3

Introduction

1.1 Overview

According to the Federal Trade Commission (FTC), there were 37,932 reports of credit card fraud in 2012 and 87,451 reports in 2022. This marks an increase of credit card payment fraud by an estimated, 30.5%. By comparison, since 2020, there has been a 14.6% increase in credit card related fraud. Which does not include the millions of other fraud reports the FTC receives every year. In 2022 alone, there were around 5.1 million fraud, identity theft, and miscellaneous reports in total [1], [2]. The statistics for these reports stresses how crucial the security of payment systems are, both physical and online. And, the need to secure them grows every year.

This research primarily focuses on physical point-of-sale (PoS) systems or terminals and their hardware (serial accessories), rather than online solutions. For instance, not mobile payment apps like Venmo, CashApp, Zelle, or Paypal [3]. There are many reasons, but the types of systems being targeted varies greatly in terms of the hardware and software supported, as well as, how the transactions are handled with the payment processor.

Figure 1.1 shows us two similar looking point-of-sale systems, albeit one is much older looking. However, the operating system and required hardware is very different. Typically, unless you have the Square provided terminal, their software/client is installed onto an Android or iOS device and connected to a Square compatible card reader [4]. Whereas, the SurePoS, NCR, or other common EFTPoS system will run a proprietary OS derived from Windows or Linux [5]. Furthermore, these PoS tend to require some form of printing receipts as record keeping for the business owner and customer. And these devices also vary in terms of processing capabilities and operating system.

(a) Square PoS

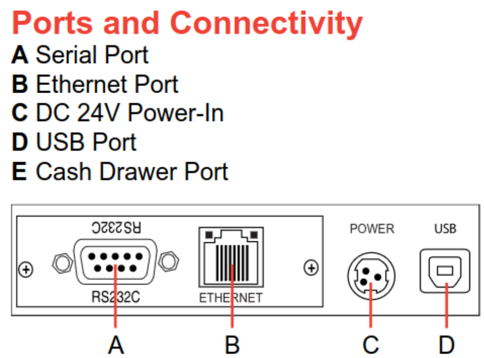
(b) SurePoS

Figure 1.1: Comparison of common PoS systems

For instance, a common thermal printer seen with PoS systems, integrated with fuel pumps, or other industrial control equipment, is the SNBC BTP-S80 thermal printer [6], [7]. There are multiple versions of the device with support for Bluetooth, USB only, or combination of USB/Serial/Ethernet. The bluetooth hardware is provided over an accessory 25-pin serial connection, with more I/O as a serial connection via RS232C connector and USB Type-B. It has driver support for various platforms: Android, iOS, Windows, Linux, and MacOS. The most interesting aspects are the processor, an Arm Cortex M4 clocked at 3.54MHz, and the operating system, a proprietary version of FreeRTOS. The system architecture is Armv7E-M with JTAG/SWD hardware debugging support [8], [9].

By default, the printer has enough headroom to process ESC/POS commands for printing paper and a webserver for debugging or general diagnostics. In theory, the uncompromised device could be flashed with modified firmware to act as a decoy and human-input-device (HID) against the host PoS. In this paper, we propose exploring the processing capabilities and extensibility of FreeRTOS to act as a dual HID clone and printer for continued research.

(a) BTP-S80



(b) Printer I/O

Figure 1.2: SNBC BTP-S80

The goal of this research is to further establish academic works in regards to embedded printer devices testing and security. This area is loosely documented within academia and only mentioned vaguely in relation to statistical reports or applied research using entirely different environments. Through this research we hope to apply gainful conclusions towards the development of an embedded environment for vulnerability assessment, penetration testing, and hardware-to-software interoperability against device hosts. Some examples of how the research could be applied in the future vary: BadUSB/BashBunny [10], JuiceShop [11], DVWA [12], or Webgoat [13]; no such work exists for embedded systems within the point-of-sale or serial printer context.

Related Works

2.1 RTOS: Software and Security

[14] introduces several embedded kernels and discusses their differences in regard to developing a secure mass storage device. For this research, we are primarily interested in RTOS-like kernels because of existing support for a sample device like the SNBC BTP-S80 printer. However, the paper criticizes such operating systems because their "real-time driven design is barely compatible with the overhead produced by security mechanisms." For many applications, there is a trade off with RTOS where performance is the main criteria and security is not a priority. [15] introduces several common RTOS and discusses their security issues. Notably, most RTOS are susceptible to code injection, cryptography inefficiency, unprotected shared memory, priority inversion, denial of service attacks, privilege escalation, and inter-process communication vulnerabilities. Depending on the MPU (microprocessor unit), the vendor has hardware protections like Intel SGX or Arm Trust Zone. These are all areas that can be used for pivoting onto the device, especially shared memory and privilege escalation. If the target device firmware is outdated (or, even libraries used by the firmware) and there are known CVEs that can be repeatedly exploited, persistence mechanisms are not a requirement to gain routine access.

2.2 Embedded Firmware Patching

Typically, updating the firmware for a device or even delivering patches requires a complete shutdown and hardware debug access (if supported). In some cases, the reflashing is unsupported through the operating system or bootloader and the flash memory needs to be reprogrammed. [16] describes a method for hotpatching downstream RTOS devices without needing to shutdown or reboot. Any changes made are permanent and as effective as traditional delivery methods. RapidPatch was capable of patching over 90% of vulnerabilities for the affected device, only needing at least 64KB or more memory and 64 MHz MCU clock. This appears to be an effective method for attackers to sideload client or server implants without risking detection.

2.3 BadUSB-like Devices

BadUSB is a well-known and documented attack vector. One of the most popular hacker tools is built-on the concept [10]. However, there are some limitations:

- Precision of attacks is limited since scripts or effects are typically deployed blind. There is no knowledge of the user environment nor ability to interact with functional user interface mechanisms (e.g., a mouse clicking a button).
- Limited to the USB 2.0 standard. Meaning, no support for video adapters like HDMI, DisplayPort, or PowerDelivery like with USB 3.0.
- There are existing methods for limiting USB access from the host, such as GoodUSB [17].

GoodUSB supports the Linux USB stack, so another solution would be required for Windows systems or RTOS. This all depends on the environment of the connected host, the PoS system. It is entirely possible that the PoS could have software like Crowdstrike Falcon deployed, which would monitor system behavior and mass storage device access [18]. Although the experiment environment will not use such software, it is an important distinction to make.

In [19], they describe several attacks at each of the applicable layers to USB attacks: the human, application, transport, and physical layers. These attacks would typically require some human element for deployment, but that is not the focus of the research (e.g., social engineering versus hardware hacking). Whereas the physical layer could allow signal eavesdropping or injection. This could enable a modified printer to overvolt the host (USBKiller [20]) to cause physical damage or perform other side-channel attacks [21]. Either of those methods would require investigating the device hardware to determine what level of control the bootloader or operating system has over power delivery.

2.4 Summary

As demonstrated by the previous works, vulnerability assessment of an embedded device is a well documented process. However, the extent that a serial thermal printer (e.g., Figure 1.2) can be maliciously expanded through a modified FreeRTOS image, while supporting original functionality, has not. And, given success in the assessment, it could suggest room for continual and improved research.

Proposed Research

3.1 Research Objectives

Section Outline:

- The goal of the research is to get an idea of what the potential ”threat map” looks like.
- With the technical ”specs” for the hardware and OS, can we manipulate device capabilities?
- What capabilities can we extend or add?

3.2 Research Questions/Hypotheses

Section Outline:

- What is the baseline or minimum hardware these devices are running?
- What software is being used on these devices? OS, libraries...
- Can the software/firmware be modified?
- If so, how much can be modified? In memory? Reflashed?
- Given hardware baseline, what does that performance support? Intended functions plus a webserver?

3.3 Methodology

Likely, but not limited to the following:

- Gather recent research within the last 5-10 years for manufacturer/device shares of the market
- Gather technical sheets and specs for the most popular devices
- Take note of the hardware specs for each device as well as firmware used
- Create default/debug images of each popular devices' firmware - what is natively supported?
- Is there room to add functionality without crippling original function

Timeline

The timeline for the research proposal is divided into four parts: review, surveying, disassembly, and writing. The dates provided are rough estimates and will vary as the project progresses. Each part is described as follows (refer to Table 4.1):

- **Review**: Review the proposal before beginning the research process to familiarize with the defined methodology, processes, and scope of project.
- **Surveying**: Gather the necessary technical data and acquire the devices.
- **Disassembly**: Each device is torn down, components identified, and documented.
- **Writing**: All data collected, pictures taken, and documents created will be gathered to create a formal report as dictated by the research purpose.

Task	Duration	Start Date	End Date
Review	7 days	06 Nov 23	13 Nov 23
Reread proposal paper	1 days	06 Nov 23	07 Nov 23
Review each related works	3 days	07 Nov 23	10 Nov 23
Take notes	1 day	10 Nov 23	11 Nov 23
Prepare work environment	2 day	11 Nov 23	13 Nov 23
Surveying	7 days	13 Nov 23	20 Nov 23
Gather list of most popular printers and acquire	4 days	14 Nov 23	17 Nov 23
Collect technical datasheets and specifications	2 days	18 Nov 23	19 Nov 23
Verify I/O and document before teardown	1 days	20 Nov 23	20 Nov 23
Disassembly	14 days	20 Nov 23	04 Dec 23
Disassemble each printer	3 days	20 Nov 23	22 Nov 23
Document interior/exterior of devices	1 days	23 Nov 23	23 Nov 23
Identify device components	3 days	24 Nov 23	26 Nov 23
Document interior/exterior of devices	3 days	27 Nov 23	29 Nov 23
Attempt hardware debug	1 days	30 Nov 23	30 Nov 23
Identify software/hardware protections	4 days	01 Dec 23	04 Dec 23
Writing	14 days	04 Dec 23	18 Dec 23

Table 4.1: Research lifecycle

Conclusion

Section Outline:

- Restate the research objectives, questions, what the research topic is and how this research will answer them.
- Briefly mention significance/impact in relation to the topic.
- Thank readers.

References

[1] "Consumer Sentinel Network Data Book for January - December 2011," Federal Trade Commission. (Oct. 22, 2023), [Online]. Available: https://www.ftc.gov/reports/consumer-sentinel-network-data-book-january-december-2011 (visited on 10/23/2023).

[2] C. FortheSentinel, "Consumer Sentinel Network Data Book 2022," 2022.

[3] Y. Wang, C. Hahn, and K. Sutrave, "Mobile payment security, threats, and challenges," in *2016 Second International Conference on Mobile and Secure Services (MobiSecServ)*, Feb. 2016, pp. 1–5. DOI: 10.1109/MOBISECSERV.2016.7440226.

[4] J. Ondrus and K. Lyytinen, "Mobile Payments Market: Towards Another Clash of the Titans?" In *2011 10th International Conference on Mobile Business*, Jun. 2011, pp. 166–172. DOI: 10.1109/ICMB.2011.41. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6047067?casa_token=jp6ioVlqPjQAAAAA:L69Yx3rjP2tvbnS5zWF8-eMm8sUNjXR586jm2QV6hLH8T2MyuueyIFCeMDqjhNWtUh-_2ERWgQ (visited on 10/23/2023).

[5] S. T. Ebimobowei, Z. Enebraye Peter, and Y. Pual, "THE ROLE OF SOFTWARE IN A CASHLESS ECONOMY (CASE STUDY NIGERIA)," *International Journal of Research -GRANTHAALAYAH*, vol. 6, no. 1, pp. 177–186, Jan. 31, 2018, ISSN: 2350-0530, 2394-3629. DOI: 10.29121/granthaalayah.v6.i1.2018.1607. [Online]. Available: https://www.granthaalayahpublication.org/journals/index.php/granthaalayah/article/view/IJRG17_A11_812 (visited on 10/23/2023).

[6] "SNBC BTP-S80 Thermal Printer - Black Cabinet (USB/Serial/Ethernet)," CRS Inc. (), [Online]. Available: https://www.crs-usa.com/products/snbc-btp-s80-thermal-printer-black-(usb-serial-ethernet) (visited on 10/23/2023).

[7] "SNBC New Beiyang-Intelligent Micro-Super, Smart Express Cabinet, Barcode Label Printer, Ticket Printer_Electronics_Receipt/Log Printer, Barcode/Label Printer, Special Scanning Products, Mixed Print Scanning Products, Smart Express Cabinet, Smart Micro-Super_New Beiyang specializes in the development, production, sales and service of intelligent print identification and system integration products. It provides leading products and complete, one-stop application solutions for various industries around the world. It is the only core design in the industry in the country through independent innovation. Manufacturing technology and forming a large-scale production enterprise." (), [Online]. Available: https://www.snbc.com.cn/ (visited on 10/23/2023).

[8] "Cortex-M4." (), [Online]. Available: `https://developer.arm.com/Processors/Cortex-M4` (visited on 10/23/2023).

[9] "FreeRTOS - Market leading RTOS (Real Time Operating System) for embedded systems with Internet of Things extensions," FreeRTOS. (), [Online]. Available: `https://www.freertos.org/index.html` (visited on 10/23/2023).

[10] Hak5. "Bash Bunny," Hak5. (), [Online]. Available: `https://shop.hak5.org/products/bash-bunny` (visited on 10/23/2023).

[11] "OWASP Juice Shop — OWASP Foundation." (), [Online]. Available: `https://owasp.org/www-project-juice-shop/` (visited on 10/23/2023).

[12] R. Wood, *DAMN VULNERABLE WEB APPLICATION*, Oct. 22, 2023. [Online]. Available: `https://github.com/digininja/DVWA` (visited on 10/23/2023).

[13] "OWASP WebGoat — OWASP Foundation." (), [Online]. Available: `https://owasp.org/www-project-webgoat/` (visited on 10/23/2023).

[14] R. Benadjila, M. Renard, P. Trebuchet, P. Thierry, and A. Michelizza, "Wookey : Usb devices strike back," 2018. [Online]. Available: `https://api.semanticscholar.org/CorpusID:199552896`.

[15] W. D. Yu, D. Baheti, and J. Wai, "Real-Time Operating System Security,"

[16] Y. He, Z. Zou, K. Sun, *et al.*, "{RapidPatch}: Firmware Hotpatching for {Real-Time} Embedded Devices," presented at the 31st USENIX Security Symposium (USENIX Security 22), 2022, pp. 2225–2242, ISBN: 978-1-939133-31-1. [Online]. Available: `https://www.usenix.org/conference/usenixsecurity22/presentation/he-yi` (visited on 10/24/2023).

[17] D. J. Tian, A. Bates, and K. Butler, "Defending Against Malicious USB Firmware with GoodUSB," in *Proceedings of the 31st Annual Computer Security Applications Conference*, ser. ACSAC '15, New York, NY, USA: Association for Computing Machinery, Dec. 7, 2015, pp. 261–270, ISBN: 978-1-4503-3682-6. DOI: `10.1145/2818000.2818040`. [Online]. Available: `https://doi.org/10.1145/2818000.2818040` (visited on 10/24/2023).

[18] J. Backer, "Sdn-controlled isolation orchestration to support end-user autonomy," Ph.D. dissertation, WORCESTER POLYTECHNIC INSTITUTE, 2021.

[19] 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. [Online]. Available: https://ieeexplore.ieee.org/document/8418652 (visited on 11/05/2023).

[20] "USB Kill devices for pentesting & law-enforcement," USBKill. (), [Online]. Available: https://usbkill.com/ (visited on 11/05/2023).

[21] K. Sridhar, S. Prasad, L. Punitha, and S. Karunakaran, "EMI issues of universal serial bus and solutions," in *8th International Conference on Electromagnetic Interference and Compatibility*, Dec. 2003, pp. 97–100. DOI: 10.1109/ICEMIC.2003.237887. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1287775 (visited on 11/05/2023).