PRINTSHOP: ASSESSING OS AND CAPABILITIES OF SERIAL PRINT DEVICES

1st Micah Flack
The Beacom College of Computer and Cyber Sciences
Dakota State University
Idaho Falls, USA
micah.flack@trojans.dsu.edu

***Abstract*—(OUTLINE FILLER WORDS) pharetra sit amet aliquam id diam maecenas ultricies mi eget mauris pharetra et ultrices neque ornare aenean euismod elementum nisi quis eleifend quam adipiscing vitae proin sagittis nisl rhoncus mattis rhoncus urna neque viverra justo nec ultrices dui sapien eget mi proin sed libero enim sed faucibus turpis in eu mi bibendum neque egestas congue quisque egestas diam in arcu cursus euismod quis viverra nibh cras pulvinar mattis nunc sed blandit libero volutpat sed cras ornare arcu dui vivamus arcu felis bibendum ut tristique et egestas quis ipsum suspendisse ultrices gravida dictum fusce ut placerat orci nulla pellentesque dignissim enim. (OUTLINE FILLER WORDS).**

***Index Terms*—serial devices, thermal printer, PoS, ICS, badusb, hardware hacking, embedded devices.**

I. Introduction

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 a 30.5% increase in credit card payment fraud compared to 2012. By comparison, since 2020, there has been a 14.6% increase in credit-card-related fraud. Which excludes 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 stress how crucial the security of payment systems is, 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, this does not include mobile payment apps such as Venmo, CashApp, Zelle, or Paypal [3]. There are many reasons, but the types of systems being targeted vary greatly in terms of the hardware and software supported, as well as, how the transactions are handled with the payment processor.

Serial devices pose a significant threat to the PoS security landscape, as well as, industrial control systems due to the limited security features of the devices. Because these devices implement no host monitoring, minimal hardware protections, dubious component suppliers, and unchecked communication with their host. Furthermore, the origin of the manufacturer, supplied components, firmware, and supporting libraries is a separate area of research; which, could provide further scrutiny into where these devices are deployed and in what environments.

A. Research Objectives

The research questions this study aims to answer are as follows:

- **RQ1:** What is the baseline or minimum hardware these devices are running?
- **RQ2:** What software is being used on these devices? OS, libraries...
- **RQ3:** Can the software/firmware be modified? FreeRTOS, ReconOS, and VXWorks.
- **RQ4:** If so, how much can be modified in memory? Is manually reflashing possible?
- **RQ5:** Assuming reflashing is possible, can the original OS keep original functions and be used as a HID clone or hub?

Manufacturers must understand the hardware and software capabilities of their peripheral devices. During the design process, they should be asking whether the hardware can support adding unintended functionality at the application and physical layers. And, with what we know about the USB standard and developing real-time operating systems, can that functionality be used to recreate a dual-purpose device? This research answers all of the questions presented.

Methodology

There are several parts to the research methodology. First, technical information and datasheets were collected for each of the identified devices. Then, device capabilities were verified before beginning device teardown and flash recovery. During the disassembly, each component was documented and further technical information was gathered from respective manufacturers. The format for presenting the collected data is described later in section I-D.

B. Research Approach

For this research, the quantitative approach and survey research was used [4, 5]. Because the goal of the research was to gather and examine, point-in-time, data across a sampled population of serial printer devices. By using quantitative survey research, it was possible to evaluate which devices are vulnerable to the attacks hypothesized, as well as, which devices are the most eligible for future design artifact research (i.e., creation of modified OS for HID cloning).

C. Cross-Sectional Survey

Using cross-sectional surveys [5] has multiple benefits. It can be used to represent data as it is taken, rather than over a long period. The study method also focuses on providing summaries that describe the patterns and context between collected data, and how it relates to the research questions.

D. Data Collection Process

The data collection process began with gathering technical specifications from device manufacturers. Typically, these contain information about the capabilities of the intended device functions. For a printer, this could contain information ranging from hardware specifications (e.g., CPU, architecture, memory) to things like printed pages per minute. This information forms the baseline for the device survey. Afterward, further specifications were gathered for components as each device was disassembled and examined. Roughly, the types and format of gathered device specifications appear as shown in Figure I-D (e.g., SNBC BTP-S80 is used here).

Following the previous example, the next step in the data collection process would be identifying the SoC. If there is no beforehand knowledge, the SoC can be identified by comparing gathered datasheets during the components discovery. This is easily accomplished using an online service like FindChips or AllDataSheets [6]. The expected type and format for SoCs are described in Figure I-D.

The process for gathering flash/memory chip specifications was similar; identify the serial number and manufacturer, then find the component datasheet. Gathering the pin layouts and format is also useful for later stages, should the manual flash need to be recovered. The expected format for memory chips can be seen in Figure I-D.

A final report will be created detailing each of these tables for the devices and their identified core components. Operating system features and protections will be loosely summarized for each device, there is no set reporting format or requirements. Using the final report will aid in the process of designing an artifact for future research and testing.

E. Hardware Assessment

NIST SP 800-115 [8] provides general guidelines for performing information security testing and assessment, however, there is little information regarding hardware reverse engineering and firmware analysis. Their guidelines are aimed more towards single/multi-tasking operating systems like Windows or Unix-like, those where network logging and listener agents are feasible. For the targeted devices in this research proposal, a different approach is needed that evaluates the hardware protections of the SoC and flash memory.

Analysis of device components, once disassembled, requires using a hardware debugger tool with the correct interface. The majority of the targeted devices use joint test action group (JTAG) or single wire debugging (SWD) headers. By referring to the manufacturer datasheet for a given SoC, it was possible to identify the pin layout for serial debugging access.

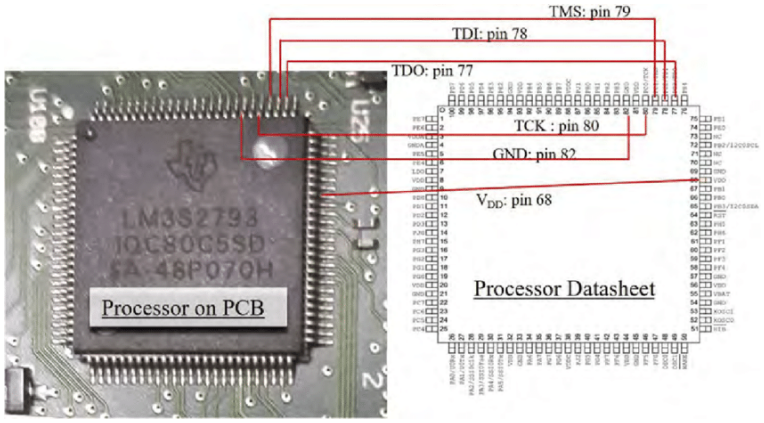

Fig. 1. JTAG pin out example for Texas Instruments LM3S2793

Figure 1 is an example showing what the physical SoC looks like on a PCB compared to the pin layout described in the datasheet. The dot in the top left of the SoC denotes the beginning of the pin layout. Counting in a counter-clockwise method indicates the pin number and the associated functions. For instance, to access the JTAG debug interface on the LM3S2793:

- TDO: pin 77
- TDI: pin 78
- TMS: pin 79
- TCK: pin 80
- GND: pin 82
- V_{DD}: pin 68

Using this information, a device like the JTAGULATOR [9] can be connected and enumerate or verify pin layouts as described. Ball joint SoCs require a different process and are much harder to debug if there is no visible header available on the board. Once an interface is connected, if debugger access is not disabled, the researcher can interact with the bootloader to further investigate enabled protections and recover flash storage.

If the JTAG is disabled, the researcher would attempt to recover flash manually using a device like the Segger J-Link [10]. The Segger has pre-defined and existing support for working with flash memory and flash breakpoints. Whereas, using OpenOCD with the JTAGULATOR would require time crafting custom configurations. Assuming there are no access protections to the flash memory, the researcher could begin performing firmware analysis to identify the operating system or potential vulnerabilities. Documenting the size and address range of memory regions was a key part of the process.

II. Related Works

A. RTOS: Software and Security

[11] introduces several embedded kernels and discusses their differences regarding 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

TABLE I
DEVICE SPECIFICATIONS FOR SNBC BTP-S80

Specifications	
Max print speed	120mm (Two-Color), 150mm (Grayscale), 250mm (Mono)
Printing method	Direct Thermal
Paper roll type	9 x 7, 82.5 x 80 x 57.5mm
Bar code support	UPC-A, UPC-E, EAN8, EAN13, Code39, Code93, CODE128, CODABAR, ITF, PDF417, QR Code, Maxicode
Printer interpreter	ESC/POS
Interfaces	Serial+USB+Ethernet USB+Parallel USB+Serial USB+Bluetooth USB+WiFi USB Only
Supported OS	32-bit (Windows XP/2000/POSReady) 64bit (Windows XP/Server 2012) 32/64bit (Windows 10/8.1/8/7/Server 2008/Server 2003/Vista) Other (Linux/OPOS/BYJavaPOS Windows/BYJavaPOS Linux)
Development Kit	Android, iOS
Data Buffer	Receive Buffer RAM: 64KB RAM Bitmap: 128KB Flash Bitmap: 512KB
Power Supply	AC 100 $\sim$ 240V, 50/60 Hz Adapter
Current/Power Usage	2.0A / 60W
Safety and EMI	FCC/UL

TABLE II
SOC TECHNICAL SPECS EXAMPLE USING STELLARIS LM3S2793 MICROCONTROLLER

Specifications	
Architecture	32-bit ARM
Platform	ARM Cortex-M3
Frequency	80-MHz, 100DMIPS performance
Memory	128KB single-cycle Flash memory 64KB single-cycle SRAM
Firmware	Internal ROM loaded with StellarisWare
Advanced Comm. Interfaces	UART, SSI, I2C, I2S, CAN
Debug Interfaces	JTAG, SWD
Package format	100-pin LQFP 108-ball pin BGA

TABLE III
MEMORY SPECIFICATIONS EXAMPLE USING INFINEON TECHNOLOGIES S25FL064P [7]

Specifications	
Single power supply operation	2.7 to 3.6V
Software Features	SPI Bus Compatible Serial Interface
Memory architecture	Uniform 64KB sectors 256 byte page size
Programming	Page programming (up to 256 bytes) Operations are page-by-page basis Accelerated mode via 9V W#/ACC pin Quad page programming
Erase commands	Bulk erase function Sector erase for 64KB sectors Sub-sector erase for 4KB and 8KB sectors
Protections	W#/ACC pin used with Status Register Bits to protect specified memory regions and configure parts as read-only; One time programmable area for permanent and secure identification
Package format	16-pin SO 8-contact WSON 24-ball BGA, 5x5 pin config 24 ball BGA, 6x6 pin config

such operating systems because their "real-time driven design is largely incompatible with the overhead produced by security mechanisms." For many applications, there is a trade-off with RTOS where performance is the main criterion and security is not a priority. [12] 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.

B. 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. [13] describes a method for hot-patching downstream RTOS devices without needing to shut down 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 a 64 MHz MCU clock. This appears to be an effective method for attackers to sideload client or server implants without risking detection.

C. BadUSB-like Devices

BadUSB is a well-known and documented attack vector. One of the most popular hacker tools is built on the concept [14]. 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 [15].

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 [16]. Although the experiment environment will not use such software, it is an important distinction to make.

RESULTS

Due to time constraints and under-budgeting, the actual sample size for this research is smaller than initially proposed. The original proposal estimated a small sample population of three to five of the most common, accessible serial printers. However, the shipment of additional printers would have taken longer than the submission deadline. All data presented is limited to available equipment at the time of reporting, the SNBC BTP-S80 serial printer.

D. Device Disassembly

The SNBC BTP-S80 is a common thermal printer used for providing receipts for PoS systems and immediate reporting for industrial control systems (ICS). This model features three buttons on the front face of the printer. Starting from the top: paper roll release, auto-feeding, and power button.

Fig. 2. SNBC BTP-S80

From the rear of the device, we can see some of the available I/O. There are several ways to interface with the thermal printer. The host device can connect using an internet address via the RJ45 ethernet connector or over serial using the USB type-b and RS232. We can also see a screw on either side of the expansion card containing the RJ45 jack and RS232 connector. The manufactures website shows that this slot is interchangeable and can provide different functionality depending on the end-users needs. Some configurations are setup as shown in Figure 3, others feature wireless adapters utilizing 2.4Ghz networking.

Removing each of the screws allows us to remove the chassis from the outer shell. Here we can see the motherboard (leftmost) and the expansion cards (topside, right of the motherboard). The expansion cards are divided into two parts, as shown in Figure 5.

The left expansion card allows the user of the device to swap networking stacks. In this configuration it features the RJ45 Ethernet and RS232 serial connector. The right expansion card provides USB connectivity and power via the barrel plug connector.

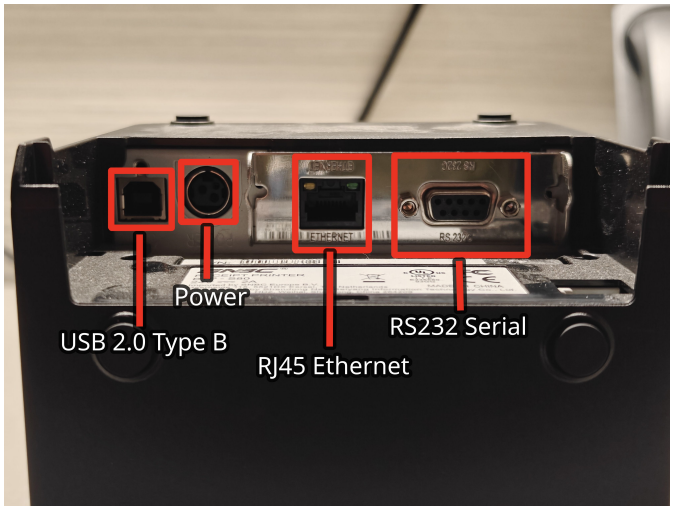

Fig. 3. SNBC BTP-S80 labeled I/O

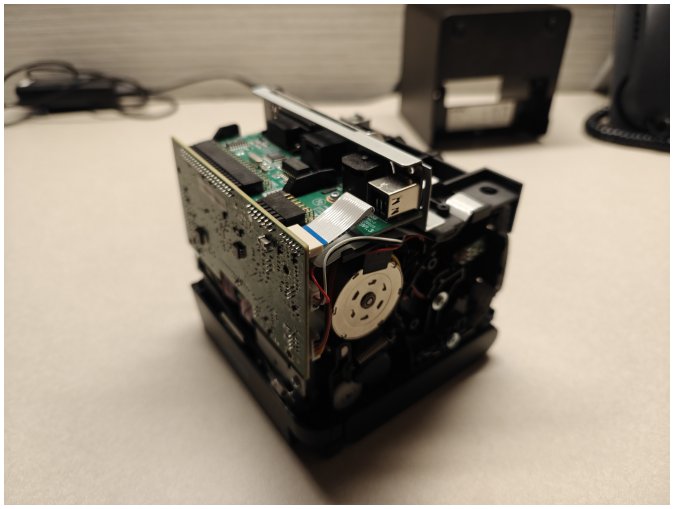

Fig. 4. SNBC BTP-S80 inner chassis

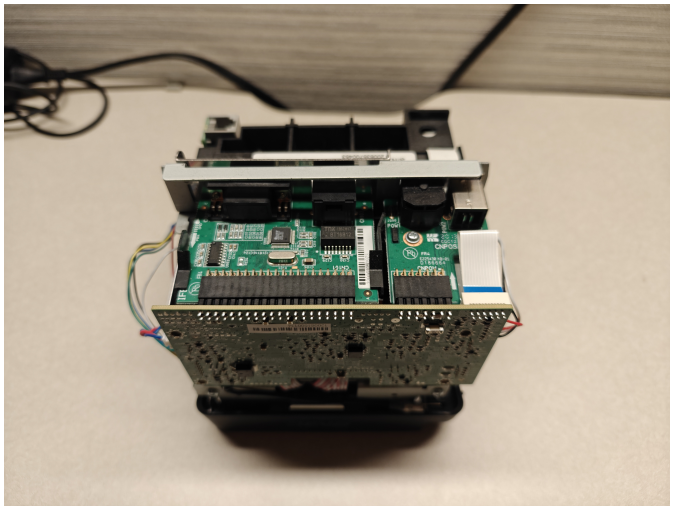

Fig. 5. SNBC BTP-S80 expansion cards

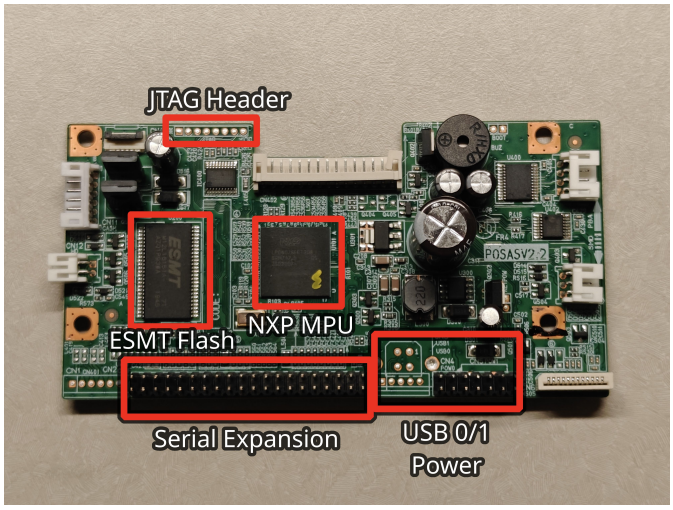

Fig. 6. Motherboard components

The disassembly is completed after carefully disconnecting each cable, taking note of their respective connectors, and preparing the boards for component identification. There are plenty of components within the device, however, our concern is only the motherboard and two expansion cards. We are only interested in researching the components used for processing and storing data.

E. Identified Components

Component identification and analysis is divided into two parts. The first being analysis of the motherboard, and the second being the serial expansion card. Analysis of the power delivery and USB type-b expansion card is not necessary since it features no controllers nor any flash to analyze. In some cases, these might still be used for fuzzing and debugging because labeled connectors are provided, however, this can be ignored with most single wire debugging tools (e.g., Jtagulator or Bus Pirate).

The list of motherboard components, as shown in Figure 6:

- NXP MPU,
- ESMT flash,

The list of expansion components, as shown in Figure 7:

- SIPEX RS-232 Transceiver,
- ASIX Ethernet Controller,

Some words...

F. Technical Resources

Outline ↓

- List datasheets and where they were sourced
- Explanation of which sites/resources and why

pharetra sit amet aliquam id diam maecenas ultricies mi eget mauris pharetra et ultrices neque ornare aenean euismod elementum nisi quis eleifend quam adipiscing vitae proin sagittis nisl rhoncus mattis rhoncus urna neque viverra justo nec ultrices dui sapien eget mi proin sed libero enim sed

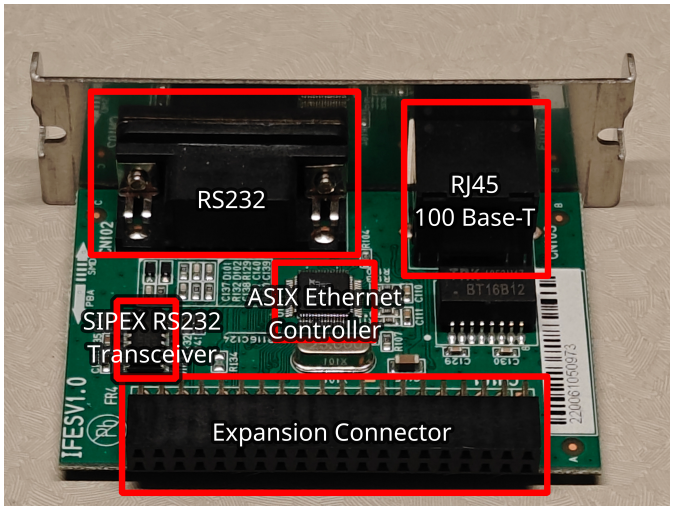

Fig. 7. Expansion card components

faucibus turpis in eu mi bibendum neque egestas congue quisque egestas diam in arcu cursus euismod quis viverra nibh cras pulvinar mattis nunc sed blandit libero volutpat sed cras ornare arcu dui vivamus arcu felis bibendum ut tristique et egestas quis ipsum suspendisse ultrices gravida dictum fusce ut placerat orci nulla pellentesque dignissim enim.

G. Firmware Analysis

Outline ↓

- List bootloader information
- Details about recovered memory regions (e.g., addr ranges, size, perms)
- Known libraries

pharetra sit amet aliquam id diam maecenas ultricies mi eget mauris pharetra et ultrices neque ornare aenean euismod elementum nisi quis eleifend quam adipiscing vitae proin sagittis nisl rhoncus mattis rhoncus urna neque viverra justo nec ultrices dui sapien eget mi proin sed libero enim sed faucibus turpis in eu mi bibendum neque egestas congue quisque egestas diam in arcu cursus euismod quis viverra nibh cras pulvinar mattis nunc sed blandit libero volutpat sed cras ornare arcu dui vivamus arcu felis bibendum ut tristique et egestas quis ipsum suspendisse ultrices gravida dictum fusce ut placerat orci nulla pellentesque dignissim enim.

H. Security Protections

Outline ↓

- Summary of physical protections (e.g., too much information silkscreened on PCB)
- List hardware protections (e.g., did manufacturer block debug access, are memory regions locked)
- Discuss software protections (e.g., access to bootloader, identifiable CVEs/CWEs, attributable libraries/functions)

pharetra sit amet aliquam id diam maecenas ultricies mi eget mauris pharetra et ultrices neque ornare aenean euismod elementum nisi quis eleifend quam adipiscing vitae proin sagittis nisl rhoncus mattis rhoncus urna neque viverra justo nec ultrices dui sapien eget mi proin sed libero enim sed faucibus turpis in eu mi bibendum neque egestas congue quisque egestas diam in arcu cursus euismod quis viverra nibh cras pulvinar mattis nunc sed blandit libero volutpat sed cras ornare arcu dui vivamus arcu felis bibendum ut tristique et egestas quis ipsum suspendisse ultrices gravida dictum fusce ut placerat orci nulla pellentesque dignissim enim.

Conclusion

Outline ↓

- Reiterate the introduction, research questions, and how the results of the research answered those questions.
- Provide supporting statements for future research and highlight areas that were loosely documented/known and would have been solid preliminary research.
- Can these devices be extended? Research questions, yes/no...

in aliquam sem fringilla ut morbi tincidunt augue interdum velit euismod in pellentesque massa placerat duis ultricies lacus sed turpis tincidunt id aliquet risus feugiat in ante metus dictum at tempor commodo ullamcorper a lacus vestibulum sed arcu non odio euismod lacinia at quis risus sed vulputate odio ut enim blandit volutpat maecenas volutpat blandit aliquam etiam erat velit scelerisque in dictum non consectetur a erat nam at lectus urna duis convallis convallis tellus id interdum velit laoreet id donec ultrices tincidunt arcu non sodales neque sodales ut etiam sit amet nisl purus in mollis nunc sed id semper risus in hendrerit gravida rutrum quisque non tellus orci ac auctor augue mauris augue neque gravida in fermentum et sollicitudin ac orci phasellus egestas tellus rutrum tellus pellentesque eu tincidunt tortor aliquam nulla facilisi cras fermentum odio eu feugiat pretium nibh ipsum consequat nisl vel pretium lectus quam id leo in vitae turpis massa sed elementum tempus egestas sed sed risus pretium quam vulputate dignissim suspendisse in est ante in nibh mauris cursus mattis molestie a iaculis at erat pellentesque adipiscing commodo elit at imperdiet dui accumsan sit amet nulla facilisi morbi tempus iaculis urna id volutpat lacus laoreet non curabitur

References

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

[2] *Consumer Sentinel Network Data Book for January - December 2011*, https://www.ftc.gov/reports/consumer-sentinel-network-data-book-january-december-2011, Oct. 2023. (visited on 10/23/2023).

[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] R. Babbie, *The Basics of Social Research*. Cengage Learning, 2017, ISBN: 978-1-305-58586-7.

[5] J. Creswell and J. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. SAGE Publications, 2017, ISBN: 978-1-5063-8671-3.

[6] *Findchips: Electronic Part Search*, https://www.findchips.com/. (visited on 11/06/2023).

[7] *S25FL064P Series NOR Flash Datasheets – Mouser*, https://www.mouser.com/c/ds/semiconductors/memory-ics/nor-flash/?series=S25FL064P. (visited on 11/06/2023).

[8] "NIST SP 800-115," *NIST*, Jan. 2020. (visited on 11/06/2023).

[9] *JTAGulator*, Grand Idea Studio, Nov. 2023. (visited on 11/06/2023).

[10] *SEGGER J-Link debug probes*, https://www.segger.com/products/debug-probes/j-link/. (visited on 11/06/2023).

[11] 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.

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

[13] Y. He *et al.*, "{RapidPatch}: Firmware Hotpatching for {Real-Time} Embedded Devices," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 2225–2242, ISBN: 978-1-939133-31-1. (visited on 10/24/2023).

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

[15] 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. 2015, pp. 261–270, ISBN: 978-1-4503-3682-6. DOI: 10.1145/2818000.2818040. (visited on 10/24/2023).

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