Effective IoT Device Discovery and Identification for Contextualized Privacy Insights Using Smartphones: Final Report
CS 4605 Team 1: What the hack?

Suha Sabi Hussain
shussain64@gatech.edu

Bozhidar Konstantinov Manev
bmanev3@gatech.edu

Zekai Wang
zwang3114@gatech.edu

Abstract

The pervasiveness of Internet-of-Things (IoT) devices has led to an erosion of privacy, presenting a significant challenge for its continued adoption and development. To empower technical and non-technical users alike, we intended to develop a method for IoT device discovery and identification using smartphones such that contextualized privacy insights could be delivered. We conducted a rigorous evaluation of leading techniques for this purpose, analyzing their performance, limitations, and ability to be deployed on smartphones without the use of external hardware. We identified six tools that, when combined, effectively detect and discover a wide variety of IoT devices. Drawing upon our evaluation, we proposed a general software architecture, conducted a preliminary exploration of the applicability of Large Language Models (LLMs), and designed a Figma prototype. We intend for our work to eventually become a practical mobile application that empowers users to make real-time decisions about their privacy by detecting and identifying nearby IoT devices.

Introduction

The ubiquity of Internet-of-Things (IoT) devices has brought about a new era of connectivity, but it has also introduced significant privacy concerns. In response to this,we have embarked on a journey to develop a method for IoT device discovery and identification using smartphones. Our goal is to provide users with contextualized privacy insights, thereby empowering them to make informed decisions about their privacy in real-time. One of our main requirements was to ensure that our chosen methods could be deployed on smartphones without the use of additional hardware; otherwise, the resulting tool is unlikely to be convenient and usable, especially for non-technical users.

Research into existing methods for IoT device discovery and identification is fragmented, complex, and unreliable, resulting in the need for a rigorous evaluation. This is a non-trivial task, requiring a deep understanding of the performance and limitations of existing techniques, and their potential for deployment on smartphones without the need for additional hardware. Our evaluation led us to identify six tools that, when combined, can effectively detect and discover a wide range of IoT devices.

We leveraged these insights to propose a general software architecture, and we explored the potential of LLMs for this task. Our proposed system is designed to collect and analyze information from various sources, including the networking module, radio, image sensors, IMU, and ToF sensors. This data, which spans the application, transport, network, data link, and physical layers, is then fed into an LLM. An object recognition model is used to localize the IoT device, while the LLM attempts to detect and identify it. We also created a Figma prototype to encode our proposed workflow.

In essence, our work represents a significant step towards developing a method for IoT device discovery and identification. The proposed system, once fully realized, promises to provide users with the tools they need to protect their privacy in an increasingly connected world. Future work will focus on refining our software architecture and application, conducting a rigorous assessment of LLMs, developing a suitable computer vision model, and creating a Minimum Viable Product (MVP) for the Android app. More broadly, this endeavor aims to empower users to take control of their privacy in the IoT era.

Background

IoT devices can violate the privacy of users in opaque and complex ways, making it difficult for users to control their information flows and improve their privacy. This is especially impactful for smart home devices. Privacy Plumber is an augmented reality (AR) app developed by Cruz et al. [2] that aimed to address this issue. This app used network traffic analysis, specifically through ARP spoofing, to overlay data flow information about devices localized by the smartphone's camera. This empowered even non-technical users to make contextualized and real-time decisions about their privacy.

However, Privacy Plumber's primary issue was that it was unable to discover and identify non-registered IoT devices. This flaw greatly limited its applicability as hidden devices, most notably hidden cameras, are one of the most prominent and pervasive privacy violations enabled by IoT devices.

Researchers have developed many methods for IoT device discovery and identification, but these methods are typically restricted to specific contexts and subsets without considering usability. Similar to Privacy Plumber, Sharma et al. built an AR app for IoT device discovery based upon analyzing the timing data of 802.11 packets [15]. DevTag integrated many previous methods reliant on network traffic analysis to put forth a potential benchmark, but was restricted to a subset of IoT devices [19]. Yan et al.'s tool identified devices that disguised themselves through rogue WiFi connections through a novel network traffic analysis procedure [20]. Huang et al. applied network traffic analysis through ARP spoofing [6]. Other network traffic analysis approaches include Chowdhury et al. [1], Sang et al. [14], Thom et al. [17], Ma et al. [10], Kurmi et al. [7], Miettinen et al. [11], and Venhoef et al. [18]. Feng et al. designed a tool that detects devices based on their electromagnetic emanations [3]. Radio frequency (RF) and mmWave signal probes have also been leveraged [8] as well as responses to light [9] and IMU measurements [16]. For hidden camera detection in particular, both image sensors [21] and smartphone time-of-flight (ToF) sensors [13] have been employed. Olsson et al. used a smartphone to detect jammers [12]. Command and control techniques for mobile devices could be repurposed for device discovery and identification [4, 5].

Evaluation

Our technical evaluation was necessary to design our software architecture. We wanted our software to be comprehensive and detect a wide variety of IoT devices. In addition, we wanted our software to be run on a smartphone without the need for any additional hardware, so that it can be easily used by non-technical users. However, existing approaches vary wildly in the types of IoT devices they can effectively analyze; this is a consequence of the complexity and variety of the IoT stack. There are many kinds of hidden and non-hidden IoT devices that manipulate different channels of information and operate on different OSI layers. Much of this research is proprietary in nature. Additionally, these approaches assume different user capabilities, so not all of these approaches can be transferred to a smartphone. Augmenting this is the fact that the extent of the capabilities of smartphones are not widely known and are evolving. Therefore, our primary questions when conducting this evaluation were:

- What methods (when combined) will result in effective and comprehensive discovery and detection of IoT devices?
- What useful information can a smartphone collect for our use case without the use of external hardware?

Method Analysis

We identified 16 promising methods for comprehensive IoT device discovery and identification. For each method, we asked a series of questions:

- What technique and channel of information does this method rely upon?
- What types of IoT devices can this method detect? What types of IoT devices can it not detect?
- What OSI layer does this method map to?
- What hardware or sensor does this method use?
- What tools does this method utilize (if any)?
- Does this method use hardware other than a smartphone? Is that a necessary requirement?

Technical Evaluation

To answer these questions, for methods with open-source codebases or otherwise reproducible techniques, we ran their method on our own hardware. If this was not possible, we simulated their technique or relied upon manual analysis. The materials we used includes: 1 OnePlus 7, 3 A9 hidden cameras, 2 Amazon hidden cameras, and 1 laptop.

The OSI layers these methods used were the application, transport, network, data link, and physical layers. The sensors or hardware include networking modules, radios, antennas, cameras, IMUs, and time-of-flight (ToF) sensors. The main tools used were: Wireshark, tcpdump, scapy, SDR, hostapd, WPS, FingerBank, netdisco, arpspoof, Nmap, Ztag, ARE, Libpcap, and Zmap.

We ran a series of experiments to determine if the aforementioned tools and techniques could be run on smartphones. For certain tools listed, we elected to use supersets of that tool (for example, instead of only using hostapd, we used Aircrack-ng).

NetHunter, NMAP, and Scapy: In our evaluation, we successfully gained root access to an Android phone and proceeded to install a custom operating system - Nethunter. This specialized OS is equipped with an array of penetration testing tools and utilities. Our initial investigation focused on evaluating the networking scanning capabilities of the Android phone. This analysis ranged from basic ICMP and ARP ping tests to more extensive scanning techniques employing tools like NMAP. Furthermore, we extended our study by replicating specific components of the NMAP scanning suite using Scapy.

Wireshark and tcpdump: To validate and troubleshoot our implementations, we adopted a combination of tcpdump and Wireshark. During the experimentation phase, we established regular SSH connections to the Android phone, thereby gaining remote access to its command-line interface. To enhance our capabilities and streamline the analysis process, we enabled X-forwarding, which allowed us to execute Wireshark directly on the phone while concurrently streaming its graphical user interface to our host computer. This approach enabled us to conduct real-time network packet capture and analysis efficiently, providing a more user-friendly and integrated experience for network monitoring and debugging tasks.

ARP Spoofing, Aircrack-ng, and Deauthentication: We continued by replicating parts of the IoT Inspector detection algorithm, utilizing both arpspoof and a custom ARP spoofing script built with Scapy. The successful execution of these attacks reinforces the phone's potential as a viable attack vector. In our final evaluations, we concentrated on assessing the WiFi capabilities of the phone. We used the Aircrack-ng suite to successfully put the phone's wireless card into monitoring mode, allowing us to capture and analyze 802.11 traffic. The Nethunter OS incorporates custom drivers that enable packet injection. Our evaluation involved a series of experiments centered around deauthentication attacks, employing various tools like Wifite2, Wifijammer, and Ettercap.

Based on our findings, we determined that after successfully conducting networking scans and discovering IoT devices, the next crucial step is to implement proactive defense measures. During our study, we recognized that hidden cameras typically store video streams locally. However, by executing deauthentication attacks on hidden cameras, we effectively eliminated the possibility of real-time surveillance through the network.

We concluded that after a successful networking scan and IoT device discovery the next logical step is pro-active defense. We acknowledge that hidden cameras usually store video streams locally, but by deauthenticating hidden cameras from the network we eliminated the possibility of real-time surveillance.

Findings

From our evaluation, we determined that the following methods should be integrated into our software. These techniques were the most effective and, in conjunction with each other, can detect the vast majority of IoT devices without the use of external hardware.

- DevTag: This method collects network traffic data using passive monitoring and active probling. This data is analyzed using an ensemble of ML models and manual rules. This tool is able to detect IoT devices that communicate over a network using a non-proprietary, unencrypted protocol. The information collected is from the application, transport, network, and data link layers. The hardware used is the networking module. The tools used include Nmap, Ztag, ARE, libpcap, and Zmap.
- DeepFinger: This network traffic analysis approach can discover and detect IoT devices that use encrypted and proprietary protocols through deep clustering. The information collected is from the transport and network layers. The hardware used is the networking module.
- Digitus: This program exploits exploits electromagnetic emanations to discover low-end and office IoT devices that may or may not have transmitter modules or DRAM. The information collected is from the physical layer. The hardware used is the radio and antenna. It should be noted that the original paper relied on external hardware, but we discovered that this technique can be transferred to jailbroken smartphones through a software defined-radio (SDR) hack.
- Lumos: This system combines network traffic analysis focused on applying ML to timing information from 802.11 packets with localization informed by integrated visual odometry. The information collected is from the hardware and physical layers; the target is any IoT device connected to WiFI. The hardware requirements are the networking module, camera, and IMU. This techique relies on the use of Scapy.
- HeatDeCam: This is a thermal image analysis system to detect certain types of hidden cameras. It collects information from the physical layer using image sensors.
- LAPD: This technique uses ML detects and localizes hidden cameras using smartphone time-of-flight(ToF) sensors. This is categorized as a physical layer approach.

Application Development

Based on our findings, we proposed a generalized software architecture for our method. The system collects and analyzes information from the networking module and radio using DevTag, DeepFinger, and Digitus. It uses image sensors and the IMU to run Lumos and HeatDeCam, and it uses ToF sensors for LAPD. All of this information falls under the application, transport, network, data link, and physical layers. This data is sent to a LLM. An object recognition model attempts to localize the IoT device as the LLM attempts to detect and identify it. More broadly: after a user draws a bounding box, an object detection model identifies what the object may be. They send this information to a LLM that has access to the relevant data.

We conducted an initial exploration of the efficacy of using LLMs for this tasks by testing it against open-source datasets such as IoT-Sentenial. However, we ran into the issue of data leakage: many of these datasets are already present in LLMs. We proposed an initial methodology for further evaluation that uses LLMs with open-source datasets that do not contain the test datasets. We identified Pythia as a suitable canditate, prompt steering as a suitable prompt engineering methodology, and the EluetherAI evaluation harness as a suitable evaluator. In addition, we developed a Figma prototype, representing the AR and non-AR components of the suggested workflow, for the Android application we hope to develop for this tool.

Reflection

Challenges

We originally wanted to collaborate and integrate our function into an app called Privacy Plumber that delivers contextualized privacy insights through AR, but we were unsuccessful in establishing such collaboration. The evaluation took much longer than expected. We underestimated the complexity of the IoT stack and smartphone capabilities. Many of these methods and data sets that come up in our evaluations are proprietary and not reproducible. Their direction and methods used are also quite disjointed, meaning our evaluation had to be far more in depth than we expected.

In our attempts to deploy possible methods on the mobile platform, we realized consumer platform manufacturer specifically designed their products to actively conceal any ports and functions we want to use that is outside the range of an ordinary mobile application, resulting in an extremely unreliable platform, that is, when it is working in the first place. For instance, we soft bricked and overheated the phone and cameras multiple times and there were times the phone wouldn't even boot. We did not get to dive into development much. Initially, we expected that one or two methods would prove effective in the evaluation and we could simply deploy that, but the work was proven quite disparate, so we needed to thoughtfully design a new architecture. Overall, the lack of time and knowledge also affected our development progress. However, we gained a deep understanding of networking, IoT devices, covert channels, ML, and smartphone capabilities.

Future Work

To extend this project, the next steps should focus on realizing our proposed software architecture and application. A systematic and rigorous assessment should determine if and how an LLM can be used for this purpose, completing our inital exploration of their utility. In addition, the proposed general architecture includes an object recognition component. More work is necessary to develop a suitable computer vision model with the requisite data. For the localization and augmented reality components in particular, integrated visual odometry approaches are likely to prove useful. The creation of a Minimum Viable Product (MVP) for the Android app is also a crucial step forward.

The effectiveness of the product is centered around user experience and usability. Conducting user studies based on interaction prototypes will provide valuable insights, especially regarding the inclusion of and distinction between different modes to explore contextualized privacy insights. For instance, it may prove more beneficial to explicitly separate hidden camera detection from other types of devices. A major weakness of our work was the lack of a comprehensive benchmark. We posit the existence of a dataset collected from an environment with a wide variety of IoT devices such as the Aware Home. This dataset could encode information useful for a technical evaluation. It could also encode useful data regarding user interaction with different devices that could inform how contextualized privacy insights are delivered. The efficacy of these insights are tied to how understandable they are to non-technical users.

This project was rooted in a desire to improve the privacy and security of the IoT world. Therefore, it is reasonable to conclude that privacy and security should be imbued in this method. Employing privacy-enhancing technologies (PETs) and stronger security measures, particularly for the LLM component of the application, will ensure user data is protected. Relevant PETs include differential privacy, encrypted machine learning, and federated learning. Moreoever, repurposing existing exploitation techniques with mobile devices for this use case will help to broaden the application's capabilities and effectiveness.

Conclusion

Our project resulted in a systematic analysis of the current state of IoT device discovery and identification. We identified which of the most promising methods can be combined in order to ensure that the vast majority of IoT devices can be effectively detected and that this method can be deployed on smartphones without the use of external hardware. This is a concrete milestone on the path to developing a mobile app that can deliver contextualized privacy insights based upon IoT device discovery and identification. While we were unable to make a MVP of the mobile app, our findings led to the proposal of a general software architecture, an initial exploration of LLMs, and a Figma prototype. These deliverables are critical for the continued development of this project. In addition, our work presents a pragmatic and practical assessment of IoT device discovery and identification as a whole. To that end, this project was an excellent demonstration of and lesson in the complexity of the IoT stack, the mismatch between expectations and reality with regards to smartphone capabilities, and the difficulty in pursuing privacy and security for mobile and ubiquitous computing applications.

References

[1] Chowdhury, R. R., Idris, A. C., and Abas, P. E. Packet-level and ieee 802.11 mac frame-level analysis for iot device identification. *Turkish Journal of Electrical Engineering and Computer Sciences 30*, 5 (2022), 1941–1961.

[2] Cruz, S., Danek, L., Liu, S., Kraemer, C., Wang, Z., Feamster, N., Huang, D. Y., Yao, Y., and Hester, J. Augmented reality's potential for identifying and mitigating home privacy leaks. In *Proceedings 2023 Symposium on Usable Security* (2023), Internet Society.

[3] Feng, J., Zhao, T., Sarkar, S., Konrad, D., Jacques, T., Cabric, D., and Sehatbakhsh, N. Fingerprinting iot devices using latent physical side-channels. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 7*, 2 (2023), 1–26.

[4] Guri, M. Gairoscope: Leaking data from air-gapped computers to nearby smartphones using speakers-to-gyro communication. In *2021 18th International Conference on Privacy, Security and Trust (PST)* (2021), IEEE, pp. 1–10.

[5] Hasan, R., Saxena, N., Haleviz, T., Zawoad, S., and Rinehart, D. Sensing-enabled channels for hard-to-detect command and control of mobile devices. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security* (New York, NY, USA, 2013), ASIA CCS '13, Association for Computing Machinery, p. 469–480.

[6] Huang, D. Y., Apthorpe, N., Li, F., Acar, G., and Feamster, N. Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4*, 2 (2020), 1–21.

[7] Kurmi, J., and Matam, R. Device identification in iot networks using network trace fingerprinting. In *2022 International Conference on Smart Applications, Communications and Networking (SmartNets)* (2022), IEEE, pp. 1–6.

[8] Li, Z., Yang, Z., Song, C., Li, C., Peng, Z., and Xu, W. E-eye: Hidden electronics recognition through mmwave nonlinear effects. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems* (2018), pp. 68–81.

[9] Liu, T., Liu, Z., Huang, J., Tan, R., and Tan, Z. Detecting wireless spy cameras via stimulating and probing. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services* (New York, NY, USA, 2018), MobiSys '18, Association for Computing Machinery, p. 243–255.

[10] Ma, X., Qu, J., Li, J., Lui, J. C., Li, Z., and Guan, X. Pinpointing hidden iot devices via spatial-temporal traffic fingerprinting. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications* (2020), IEEE, pp. 894–903.

[11] Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A.-R., and Tarkoma, S. Iot sentinel: Automated device-type identification for security enforcement in iot. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)* (2017), IEEE, pp. 2177–2184.

[12] Olsson, G. K., Nilsson, S., Axell, E., Larsson, E. G., and Papadimitratos, P. Using mobile phones for participatory detection and localization of a gnss jammer. In *2023 IEEE/ION Position, Location and Navigation Symposium (PLANS)* (2023), pp. 536–541.

[13] Sami, S., Tan, S. R. X., Sun, B., and Han, J. On utilizing smartphone time-of-flight sensors to detect hidden spy cameras. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems* (New York, NY, USA, 2021), SenSys '21, Association for Computing Machinery, p. 384–385.

[14] Sang, Y., Yang, J., Zhang, Y., Li, S., and Chang, P. Toward iot device fingerprinting from proprietary protocol traffic via key-blocks aware approach. *Computers & Security 131* (2023), 103145.

[15] Sharma, R. A., Soltanaghaei, E., Rowe, A., and Sekar, V. Lumos: Identifying and localizing diverse hidden IoT devices in an unfamiliar environment. In *31st USENIX Security Symposium (USENIX Security 22)* (Boston, MA, Aug. 2022), USENIX Association, pp. 1095–1112.

[16] Singh, A. D., Garcia, L., Noor, J., and Srivastava, M. I always feel like somebody's sensing me! a framework to detect, identify, and localize clandestine wireless sensors. In *30th USENIX Security Symposium (USENIX Security 21)* (Aug. 2021), USENIX Association, pp. 1829–1846.

[17] Thom, J., Thom, N., Sengupta, S., and Hand, E. Smart recon: Network traffic fingerprinting for iot device identification. In *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)* (2022), pp. 0072–0079.

[18] Vanhoef, M., Matte, C., Cunche, M., Cardoso, L. S., and Piessens, F. Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms. In *Proceedings of the 11th ACM on Asia conference on computer and communications security* (2016), pp. 413–424.

[19] Wan, S., Li, Q., Wang, H., Li, H., and Sun, L. Devtag: A benchmark for fingerprinting iot devices. *IEEE Internet of Things Journal 10*, 7 (2022), 6388–6399.

[20] Yan, D., Yan, Y., Yang, P., Song, W.-Z., Li, X.-Y., and Liu, P. Real-time identification of rogue wifi connections in the wild. *IEEE Internet of Things Journal 10*, 7 (2022), 6042–6058.

[21] Yu, Z., Li, Z., Chang, Y., Fong, S., Liu, J., and Zhang, N. Heatdecam: Detecting hidden spy cameras via thermal emissions. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* (New York, NY, USA, 2022), CCS '22, Association for Computing Machinery, p. 3107–3120.