COMP4337 - Securing Wireless Networks

Research Report

Automatic fingerprinting of vulnerable BLE IoT devices with static UUIDs from mobile apps

Chris Joy (z5113243), Sen Li (z5183990)

23rd of April 2020

Paper Summary

We will examine a paper published in proceedings of the 2019 ACM SIGSAC Conference, in which security researchers from Ohio State University examine a method of automatically fingerprinting vulnerable Bluetooth low-energy (BLE) IoT devices with static UUIDs from mobile apps [1]. As a result of this work, the researchers built an automated analysis tool (BLEScope), aimed at helping developers find the attack surfaces they have uncovered.

Background

The two common types of attacks performed on BLE devices are nearby and remote attacks. Bluetooth, being a short-range communication technology, is only able to transmit only up to 100 meters away, using a transmitter [1]. The means nearby attackers are constraint to a particular distance. This constraint has lead to the development of Bluetooth adapters that can sniff BLE devices from up to 1000 meters. Remote attacks can be conducted by installing malware on the BLE device (via social engineering, backdoors or direct exploitation) [1]. After setting up the attack, remote attackers will be able to hack nearby devices, assuming Bluetooth is enabled on those devices. In this paper, the primary focus is nearby attacks. They build a special long-range Bluetooth antenna for USD$150, extending their reach to 1km [1].

In order to establish a connection, BLE devices initially broadcast packets intended to advertise their availability. These packets contain UUIDs, in which neighbouring devices have access. UUIDs are unique generated IDs used to distinguish different devices. Some BLE devices are assigned UUID on a chip-level, others generated by the OS [2]. Smartphone apps leverage these UUIDs in order to identify the device. After identifying the device, the are devices are able to pair with each other, after approval from the requestee, which enables further data communication [1].

Attack Overview

This paper explores a fundamental design flaw in the communication protocols used by BLE devices. It explores a method, used by attackers to fingerprint BLE devices using the static UUIDs accessible from companion apps [1]. As a result of this study, researchers also discover many apps utilise the “Just Works” pairing, opening up a security vulnerability in which attackers connect to those devices, without any app-level authentication required [1]. The key objective is to systematically deduce a set of insecure IoT devices that can be attacked, by using an automated tool. The automated tool should take in an input, containing a set of all IoT apps on the Google Play Store along with the UUID from the advertisement packet and produce an output containing the devices that are vulnerable to fingerprinting [1]. A higher-level overview of the steps taken by the tool is as follows. Please note the steps below is a simplified version of what was described in the actual paper and each of these steps break down into multiple other steps.

(1) Perform analysis on the Bluetooth API used by the given set of IoT App APKs (i.e. packaged android apps). This operation outputs a list of UUIDs and reconstructed hierarchies that can be used for UUID fingerprinting [1].

(2) After retrieving these UUIDs (along with their hierarchies), they then identify BLE devices that can be fingerprinted, by extracting the UUIDs specified in the app [1]. This extraction process is difficult (but still possible), since one UUID may map to multiple apps since BLE chips usually handle mapping of the UUID config on the OS [2].

(3) Finally, they identify devices that are vulnerable to unauthorised access via the app code, by checking if it used the “Just Works” protocol [1]. They check if the app uses any crypto hashing functions around the data transmission phase, if not they assume the data being transferred can be sniffed [1].

Literature Review

Past Research

In the past, there has been lots of work done by research to uncover mobile app vulnerabilities, due to the increasing reliance on mobile technologies. A lot of this work has been done through using methods such as taint analysis, that is a method which is used to check which variables can be modified as a result of user input [8]. The researchers only apply taint methods in their automated system after gaining access. They describe a unique approach, using fingerprinting, which isn’t a common approach. There’s only been another paper that applies this same approach by T. Gu and P.Mohapatra, where they examine methods of securing IoT Networks via fingerprinting-based device authentication [12].

Related Research

Over recent years, there has been an increased focus on IoT security. Majority of this work has been focused on discovering vulnerabilities on IoT devices, including smart locks, home appliances and watchers/bands [3]. However, only a few of them focus on BTE devices. Of these papers, the majority of them focus on thread models including credential leakages, use of unencrypted channels [5], policy misconfiguration [6] and memory corruptions [6]. There are a number of attack surfaces that have been exposed as a result of that research, including denial of service (DoS), man-in-the-middle (MiTM) and gaining root access [1]. However, compared to those papers that typically focus on single device types and are relatively small in scale, the research presented in this paper systematically studies all kinds of BTE devices by automating key discovery steps, resulting in a scalable system [1].

Inspired Work

As a result of research in this space, there has been a rapid emergence of security systems & protocols, such as systems employed by solutions such as SmartAuth [9], BLE-Guardian [11] and Flow-fence [10]. These systems primarily aimed at providing a commercialised solution, whereas BLEScope, produced by the authors of this paper, developed open-source software solutions & hardware, aimed at increasing public awareness of this attack.

Strengths of this Work

Experimental Setup

One of the key aspects we admire in this work is how the researchers set up a framework and clearly defined the scope, including constraints arisen from ethical considerations. The evaluation metrics derived from the following two sets of experiments [1]:

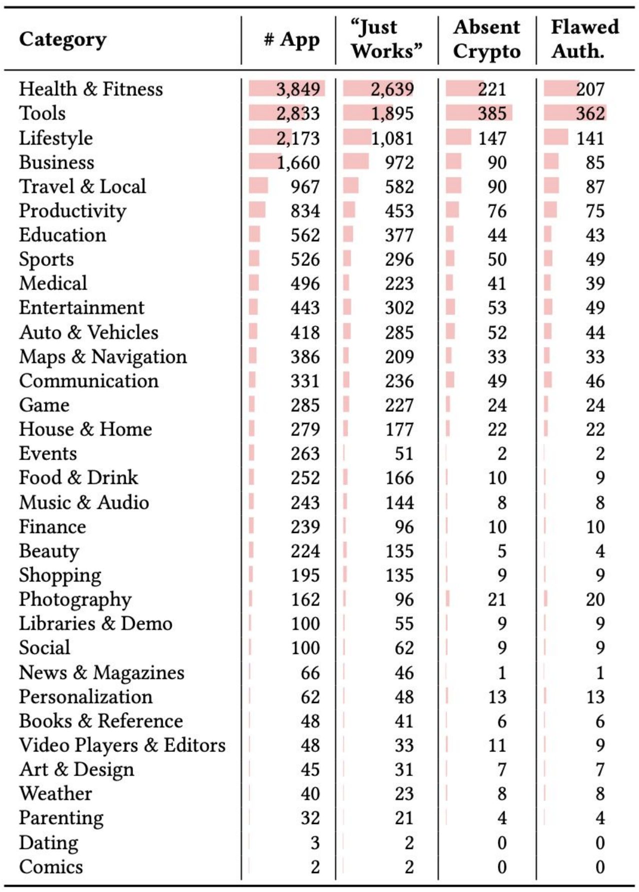
(1) Static analysis of Android apps found on the Google Play Store, which included other sorts of analysis (including value-set analysis [3]), conducted on a Linux-based server running on a machine with two Intel Xeon E5-2695 CPUs [1].

(2) Passive sniffing of the UUIDs present in the advertising packets. The researchers built an extended Bluetooth sniffer using a Raspberry PI and a special BLE Antenna [1].

One hurdle faced by relying on Google Play Store apps was that Google didn't provide information that indicates if the app is a BLE IoT type. Their API only provided a limited amount of meta-data, so the researchers applied a heuristic in search of BLE IoT apps. They manually checked the app manifest file and found 135k apps from 2 million free apps, which was the initial sample size. However, this heuristic wasn't perfect, when they later discovered another 18k BTE apps, after running them and checking if it used the Bluetooth interface [1].

Results

The researchers discovered a large number of apps in the Google Play Store that is vulnerable to this app-level authentication attack [1]. This meant that devices with these vulnerable apps were susceptible to being controlled by attackers. In order to raise public awareness and diagnose vulnerabilities, the researchers built an automated mobile application analysis toolkit called BLEScope [1]. They ran BLEScope on all free BLE-enabled IoT apps in the Google Play Store and discovered around 1700 vulnerable mobile apps in total [1]. They also ran a field test in a heavily populated area (of around 1.3 square miles) and given 5,822 BLE devices, they were able to identify 5,509 (~95%) of devices were susceptible to this fingerprinting attack [1]. Furthermore, of the devices that were susceptible, the researchers discovered 430 (7.5%) of devices were vulnerable to unauthorised pairing & control [1]. As you see in the table below, the number one category of affected apps is Tools (e.g. Smart home apps etc). They also note that they contacted every vendor responsible for maintaining the affected apps, disclosing specifics about the vulnerability they'd discovered. This was automated via a script.



Category	# App	"Just Works"	Absent Crypto	Flawed Auth.
Health & Fitness	3,849	2,639	221	207
Tools	2,833	1,895	385	362
Lifestyle	2,173	1,081	147	141
Business	1,660	972	90	85
Travel & Local	967	582	90	87
Productivity	834	453	76	75
Education	562	377	44	43
Sports	526	296	50	49
Medical	496	223	41	39
Entertainment	443	302	53	49
Auto & Vehicles	418	285	52	44
Maps & Navigation	386	209	33	33
Communication	331	236	49	46
Game	285	227	24	24
House & Home	279	177	22	22
Events	263	51	2	2
Food & Drink	252	166	10	9
Music & Audio	243	144	8	8
Finance	239	96	10	10
Beauty	224	135	5	4
Shopping	195	135	9	9
Photography	162	96	21	20
Libraries & Demo	100	55	9	9
Social	100	62	9	9
News & Magazines	66	46	1	1
Personalization	62	48	13	13
Books & Reference	48	41	6	6
Video Players & Editors	48	33	11	9
Art & Design	45	31	7	7
Weather	40	23	8	8
Parenting	32	21	4	4
Dating	3	2	0	0
Comics	2	2	0	0

Table 1. Number of affected BLE Apps by Category [1]

Weaknesses of this Work

Limitations

BLEScope has helped identify a large number of vulnerable apps through UUID fingerprinting, however, there are limitations to its capabilities that can be improved. The authors state that the mechanism used by BLEScope to identify flawed authentication practices in mobile apps consists of a very strict hard-coded ruleset which requires all parameters to be set [1]. As a result, this leads to many false negatives since the system is not able to identify which data contains the authentication tokens [7]. This results in some apps, sending data via BLE peripherals, which does not mean the data is being used for authentication. Building a generalised solution is not feasible since app developers use a variety of different design patterns in dealing with auth.

Ethical Challenges

There have been a few restrictions that have been put in place, preventing researchers from conducting tests on real IoT devices that contain sensitive and private user information. They were not able to actively connect to any of the devices that they'd identified during their passive scan of advertising packets (used to obtain the UUID). This resulted in only being able to collect service UUIDs that had been exposed in those advertising packets [1]. As a result, it was impossible for the authors to reverse engineer the hierarchy of UUIDs [4] because it requires a connection to the device [1]. They stated that because real attackers weren't restricted due to ethical consideration, the actual number of vulnerabilities may be larger than that was discussed in the paper.

Complexity of Ideas

Originality of ideas

The researchers present a unique concept that hasn't been explored in this particular context. Fingerprinting attacks have been explored by other network contexts including ToR [13]. In our personal opinion, this work demonstrates a unique attack that would have wide implications given the rise of IoT devices and our reliance on this technology. In order to raise public awareness, the authors published all the software they developed throughout the study, as open source software. We think this should be built into the Google Play Store Console, where Google would automatically alert developers if their apps are vulnerable to this attack.

Countermeasures

The researchers present a list of countermeasures that can prevent attacks from multiple levels including, protocol-level, application level and channel-level protection [1]. All of the suggestions are low-level software-based improvements, requiring users to update their device firmware. These patches could in theory be delivered via over-the-air (OTA) updates.

References

[1] CCS '19: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, November 2019. Pages 1469–1483

[2] Gogul Balakrishnan and Thomas Reps. 2004. Analyzing memory accesses in x86 executables. In International conference on compiler construction. Springer. Pages 5–23.

[3] Redjem Bouhenguel, Imad Mahgoub, and Mohammad Ilyas. 2008. Bluetooth security in wearable computing applications. In 2008 international symposium on high capacity optical networks and enabling technologies. IEEE, Pages 182–186.

[4] Jiongyi Chen, Wenrui Diao, Qingchuan Zhao, Chaoshun Zuo, Zhiqiang Lin, XiaoFeng Wang, Wing Cheong Lau, Menghan Sun, Ronghai Yang, and Kehuan Zhang. 2018. IoTFuzzer: Discovering Memory Corruptions in IoT Through App- based Fuzzing.. In NDSS.

[5] Brian Cusack, Bryce Antony, Gerard Ward, and Shaunak Mody. 2017. Assessment of security vulnerabilities in wearable devices. (2017).

[6] Britt Cyr, Webb Horn, Daniela Miao, and Michael Specter. 2014. Security analysis of wearable fitness devices (fitbit). Massachusets Institute of Technology (2014), Pages 1.

[7] Aveek K Das, Parth H Pathak, Chen-Nee Chuah, and Prasant Mohapatra. 2016. Uncovering privacy leakage in ble network traffic of wearable fitness trackers. In Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications. ACM. Pages 99–104.

[8] Jonathan Salwan, Shell Storm. Taint-analysis-and-pattern-matching-with-Pin, 2013-08-08, Accessed on 20 April 2020: http://shell-storm.org/blog/Taint-analysis-and-pattern-matching-with-Pin/

[9] Yuan Tian, Nan Zhang, Yueh-Hsun Lin, XiaoFeng Wang, Blase Ur, Xianzheng Guo, and Patrick Tague. 2017. Smartauth: User-centered authorization for the internet of things. In 26th USENIX Security Symposium (USENIX Security 17). Pages 361–378.

[10] Earlence Fernandes, Justin Paupore, Amir Rahmati, Daniel Simionato, Mauro Conti, and Atul Prakash. 2016. Flowfence: Practical data protection for emerging IoT application frameworks. In 25th USENIX Security Symposium (USENIX Security 16). Pages 531–548.

[11] Kassem Fawaz, Kyu-Han Kim, and Kang G Shin. 2016. Protecting Privacy of BLE Device Users. In 25th USENIX Security Symposium (USENIX Security 16). Pages 1205–1221.

[12] T. Gu and P. Mohapatra, "BF-IoT: Securing the IoT Networks via Fingerprinting-Based Device Authentication," 2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), Chengdu, 2018. Pages. 254-262

[13] 2013. Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. Association for Computing Machinery, New York, NY, USA..

Project Log

Date	**Task**
16/04/2020	● Reading through the paper ● Initialising the google doc & presentation
17/04/2020	● Video call via teams to assign tasks ● Structuring the presentation
19/04/2020	● Report boilerplate with subheading ● Working on the presentation ● Working on report introduction
20/04/2020	● Compiling resources for literature review ● Working on the literature review ● Trying to gain access to unsw library's resources ○ Configuring VPN for this ● Working on presentation slides - main concepts
21/04/2020	● Finishing of the literature review ● Improving referencing within the report, to make it more consistent with the actual paper. ● Trying to find github repo for BLEScope
22/04/2020	● Recording the presentation ● Finishing of the report ○ Strengths ○ Weaknesses ○ Complexities / originality ● Fixing up report presentation & reference links ● Proof-reading and using grammarly to check report ● Refining report template
23/04/2020	● Uploading the youtube video & posting it on the group forum
24/04/2020	● Finishing touches on the report (rephrasing complex sentences etc) ● Further presentation refinements. ● Submitting the report via email.