Classification of Wearable Device Bluetooth Data

1. Hamza Ahmed, Sid Burre, Preethi Chidambaram, Matthew Whelan
[1] University of Virginia, Department of Computer Science, Charlottesville, Virginia

Abstract

The increasing prevalence of Bluetooth communications has raised concern about privacy and security risks associated with the protocol. This study aimed to investigate the feasibility of analysing Bluetooth traffic to identify sensitive information. Specifically, we collected transmission data between an iPhone 14 Pro and an Apple Watch Series 4, and focused on the L2CAP protocol, which is responsible for data transfer between devices. We collected transmission records of various applications including phone calls, messages, and music playback, to ensure different levels of security and packet patterns. We then filtered the data using the L2CAP protocol and used machine learning algorithms to classify the traffic. Our results show that it is possible to accurately classify Bluetooth traffic based on the type of communication, with an overall accuracy of 92%. Our study demonstrates the potential of analyzing Bluetooth traffic for privacy and security applications, and provides insights for future research in this area.

Introduction

As the use of Bluetooth devices continues to increase, there is a growing concern about the potential privacy and security risks associated with these devices. With an estimated 5.5 billion new devices being introduced this year, it is important to be aware of the sensitive information that these devices can contain and the potential for eavesdropping [1]. Bluetooth technology enables secure wireless communication between devices, but it also means that information can be transmitted and intercepted without physical access to the device. This raises the possibility of eavesdropping, where attackers can intercept the communication between devices and access sensitive information such as credit card details, authentication codes, and phone call traffic. Furthermore, the potential for side channel attacks, gaining information from packet pattern analysis, has been clearly exploited [2].

More technically, bluetooth uses frequency hopping spread spectrum (FHSS) to transmit data wirelessly. This method involves transmitting a signal over a range of frequencies, typically 79 different frequencies in the 2.4 GHz band, instead of just one frequency. Each frequency is used for a short period of time before hopping to the next frequency in the pattern. The hopping pattern is determined by the Bluetooth master device's MAC address and clock, which all slave devices synchronize with. This process is done through adaptive frequency hopping (AFH), which dynamically adjusts the hopping patterns based on the current radio frequency (RF) environment to ensure optimal performance and minimize interference [2]. In this way, eavesdropping is a complex process due to Bluetooth's adaptive nature, specific hopping pattern and short transmission time. Many inexpensive sniffers like Ubertooth generally require observing the pairing while attempting to follow the frequency hopping of the connection, but do so with questionable accuracy [3]. However, expensive sniffers listen concurrently on all channels and accurately sniff traffic without the need to ever observe the pairing. Therefore,

we decided that using a sniffer would be out of the scope of the project, so we looked towards more feasible options for Bluetooth sniffing.

In summary, the contributions of this work are as follows:

- Demonstrate traffic-analysis attacks on Bluetooth traffic captured from wearable devices, showcasing the ability to analyze and interpret the encrypted traffic.
- Develop a machine learning classification methodology that accurately categorizes Bluetooth traffic based on its communication type, achieving an impressive overall accuracy rate of 92%.

Related Work

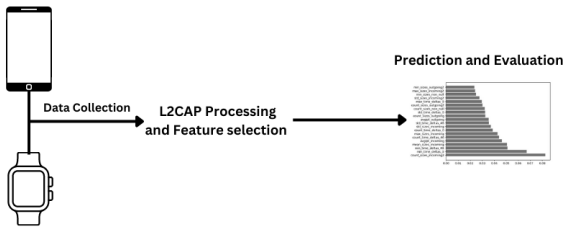
There is a rising interest in literature surrounding security vulnerabilities of Bluetooth communication and techniques for Bluetooth sniffing. Although the flaws of Bluetooth encryption are well documented [4], true sniffing of Bluetooth traffic has been considered a significant challenge. There exist a plethora of current technologies to sniff Bluetooth traffic. The first is GNURadio/USRP which can be used to decode Bluetooth packets [5]. But due to the significant overhead of signal processing and the delay associated with frequency switching, this technique is limited to only one channel at a time, severely limiting the scope and capabilities of such a technique. A technique that relies on the firmware configuration of Bluetooth chipsets exists, allowing the radio to report packet-level diagnosis in sniffing mode [6]. But for this exploit to function, the sniffer would need to be paired with the target device, impractical in an adversarial environment. There is a novel dual-radio architecture to learn the hopping sequence of Bluetooth networks and predict adaptive hopping behavior, with a packet capture rate of over 90% [5]. There are also Software-defined Radio (SDR) based sniffers that allow for full spectrum Bluetooth analysis using a robust de-anonymization technique to deliver high accuracy [6].

Sophisticated sniffers that rely on specialised hardware to monitor all sub channels simultaneously exist but until recently have cost on the order of tens of thousands of dollars. Although other stop-gap methods of packet sniffing via low-end sniffers exist, due to the poor sniffing performance, misprediction of hopping behaviour, and excessive packet corruption, low-end sniffers are largely impractical [7]. With a decrease in sniffing-related hardware costs, the practicality of sniffing-based side-channel attacks improves greatly.

In terms of literature analyzing sniffed traffic, one study looks at connections between mobile devices and in-vehicle infotainment systems to attempt to exploit vulnerabilities in these systems [3]. The wireless communication used both WiFi and Bluetooth, and packets were collected and examined by expensive sniffers in a traditional Man in the Middle (MitM) attack. However, this research had some limitations as it used limited configurations and was not able to analyze encrypted WiFi traffic. This reinforced this paper's intention to only analyze Bluetooth traffic in different environments. Another study reviewed Bluetooth traffic from wearable devices to mobile devices to infer sensitive information about users including health status, physical activity, and location [4]. A commercial wide-band scanner was used to capture upwards of 10,700 Bluetooth signals, and they used machine learning to try to predict user actions given their bluetooth traffic. Specifically, this meant multi-class classification problems using random forest classifiers with 7 Classic and 7 LE devices that gave an average precision of 0.97. This paper has significant implications for accurately identifying Bluetooth traffic in other contexts. To expand on their findings, the scope will be limited in this paper to just one device and look at a specific set of applications that communicate sensitive information. Likewise, this will help focus more on actually using the app instead of simply opening it.

Methods

Data collection was first performed in order to test and train the classifier. Apple devices were chosen to be



studied since they are known for security and should patterns may not be as easily identifiable. Data collection was performed between an iPhone 14 Pro with iOS 16.0.2 and an Apple Watch Series 4 with watchOS 9.3.1. These devices were both set up in a controlled environment to ensure no other bluetooth traffic could be detected. Packet collection was conducted using PacketLogger in XCode, allowing direct access to the packet stream.

In order to train and test the classifiers, many different types of data had to be collected. Data was collected by running the Xcode packet logger as the experiment was being conducted. Each sample was collected for two minutes when possible to ensure that the number of packets were the same, but this was not possible for every data type. To get a variety of samples, many different data types were measured. The data types measured included: Phone Call (answered), Phone Call (notification), Messages, Duo notification, ECG Data, Strength Training, Outdoor walks, and Spotify music playback. These data types were chosen to ensure different levels in security and changes in packet patterns. 10 sets of data were collected for each sample type with all other applications closed on the devices.

The data then had to be filtered, and the L2CAP protocol was chosen as the filtrication protocol. L2CAP stands for Logical Link Control Adaptation Protocol, a bluetooth protocol responsible for segmentation and the reassembly of data. L2CAP sits above the Host Controller Interface, HCI, and transfers packets to this layer. The L2CAP protocol was chosen since it was found to be associated with data transfer between devices, and the patterns would be dependent on this protocol.

After filtration, relevant features were identified that would be utilized in the classifiers to help inform the predictions. Relevant features included time between packets, packet size, average interpacket time, and other statistics such as maximum and minimum values regarding time deltas and packet size.

The following classifiers were then applied to 80/20 percent split of data for training and testing. The classifiers included: naive-bayes, support vector machines, k nearest neighbors, and random forest implementations. SciKit learn python packages were used for the classification implementations and the data was then analyzed and compared.

Results

10 sets of each action were recorded via a ~2 minute packet capture via packetScanner into a .pklg format. Subsequently, the raw .pklg files were examined using WireShark and exported as a .csv file. To extract the transmission data, only L2CAP protocol packets were used, lower level packets such as HCI_ACL and HCI_CMD/HCI_EVT were discarded. From the L2CAP packets, the following features were extracted:

Attribute	Average	Minimum	Mean	Max	Std. Dev.
Packet Size	✓	✓	✓	✓	✓
Incoming Packet Size	✓	✓	✓	✓	✓
Outgoing Packet Size	✓	✓	✓	✓	✓
Incoming Unique Packet Size	✓	✓	✓	✓	✓
Outgoing Unique Packet Size	✓	✓	✓	✓	✓
Time Deltas	✓	✓	✓	✓	✓

As well as Average Incoming/Outgoing inter-packet times with the following formula:

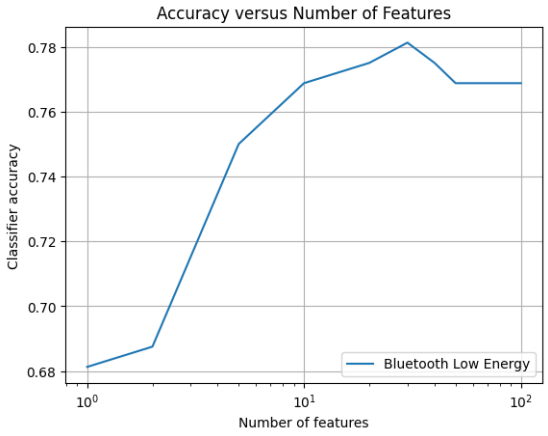
Where P is the set of sent/received packets and tsi is the timestamp of packet i. This value captures the “burstiness” of the inter-device communication. In classifying time deltas, we also capture features on varying ranges of time deltas. So not only on all time

deltas but also on deltas strictly larger than 46 ms and 1005 ms as well.

Prior to training, we split our dataset of application-specific packet capture features into 80% training and 20% testing sets. On this dataset, the following models were used with test-set accuracies listed below.

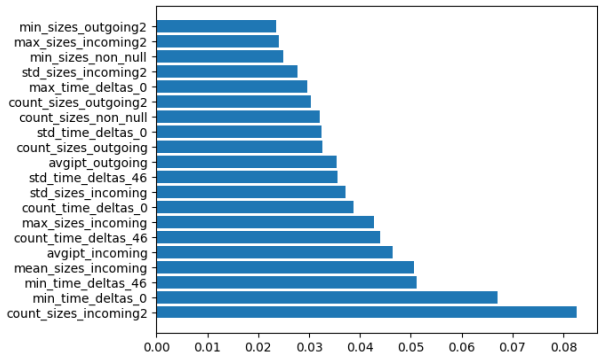
Model	Test Set Accuracy (%)
Naive Bayes Classifier	93.75
Support Vector Machine	75.00
K-Nearest Neighbors Model	50.00
Random Forest Classifier	81.25

As we tested the Random Forest Classifier, we provided it with a subset of the features that we found were best and produced the following graph.

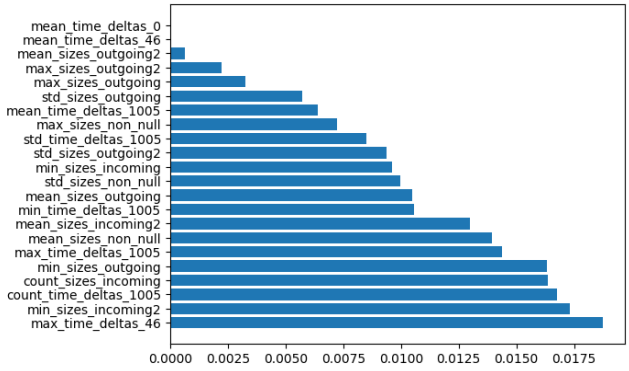


This graph shows how the Random Forest Classifier performed with the best possible features given a subset of the features. To determine the best possible subset, the Random Forest Classifier was trained and tested on each possible subset of a certain size; the best classifier accuracy for each size of subset is displayed above.

The final component of our project was assessing feature importance, essentially how specific features were highly important to the application classification schema and how other features were largely irrelevant to the schema. The best configuration of the Random Forest Classifier ranked the following features with great importance:



And the following features with least importance:



Discussion

The feature extraction and parameter filtering played a significant role in our model accuracy as it produced important results regarding feature importance. Our model ranked the number of incoming packets as being the most important feature and the maximum time deltas, or inter-packet times, as the least important feature. Given these rankings, the random forest classifier was able to more accurately predict the application being used as shown in [Figure Title for Line Graph]. Since we did not implement any feature filtering for the other classifiers, we were not able to achieve optimal results for those, with the exception of the naive-bayes classifier, which surprisingly had the best results.

While the results of our study demonstrated that machine learning algorithms can accurately predict user activity based on bluetooth traffic data, there were

several limitations that should be considered in future research. One of the primary limitations with this project was the difficulty in collecting sufficient data to train a robust model. This was largely due to the fact that we performed manual data collection for various applications, which limited the amount of data we were able to obtain for each individual application. In order to address this problem, future research should investigate the use of automated data collection methods that do not require the use of an expensive commercial scanner. One plausible method would be to set up a model to consume a stream of data that is being automatically generated from a specific application and repeat that process for various different applications. This modification would also address the dynamic nature of our current work because having a system that is able to collect and analyze real time input expands the scope of this research and the number of practical applications for it. Another limitation in our research was that we focused largely on messaging and communication applications, which is not representative of user phone and smartwatch activity. Expanding the type of applications that are analyzed to include those regarding fitness and social media can help assess the model's performance in different contexts. Additionally, including different devices will allow future research to draw broader conclusions about bluetooth traffic data.

Contributions

Hamza Ahmed performed data collection and testing. Sid Burre and Preethi Chidambaram worked on classification methods and data analysis. Matthew Whelan performed background research and literature review.

References

[1] Bluetooth Special Interest Group. "2022 Bluetooth Market Update." Retrieved from Bluetooth website: [Online]. Available: https://www.bluetooth.com/2022-market-update/

[2] J. Marcel, "How Bluetooth Technology Uses Adaptive Frequency Hopping to Overcome Packet Interference," Bluetooth.com. [Online]. Available: https://www.bluetooth.com/blog/how-bluetooth-technology-uses-adaptive-frequency-hopping-to-overcome-packet-interference/

[3] Y. Zhang and Z. Lin, "When Good Becomes Evil: Tracking Bluetooth Low Energy Devices via Allowlist-Based Side Channel and Its Countermeasure," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, in CCS '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 3181–3194. doi: 10.1145/3548606.3559372.

[4] M. Ryan, "Bluetooth: With low energy comes low security," in Proceedings of the 7th USENIX Workshop on Offensive Technologies (WOOT), 2013.

[5] N. B. N. Ibn Minar, "Bluetooth Security Threats And Solutions: A Survey," *IJDPS*, vol. 3, no. 1, pp. 127–148, Jan. 2012, doi: 10.5121/ijdps.2012.3110.

[6] D. Spill and A. Bittau, "BlueSniff: Eve meets Alice and Bluetooth," in Proc. 1st USENIX Workshop Offensive Technol. (WOOT). Berkeley, CA, USA: USENIX Association, 2007, pp. 1–10.

[7] Y. Li, J. Barthelemy, S. Sun, P. Perez, and B. Moran, "A Case Study of WiFi Sniffing Performance Evaluation," *IEEE Access*, vol. 8, pp. 129224–129235, 2020, doi: 10.1109/ACCESS.2020.3008533.

[8] Sniffer Firmware of CC2540. Accessed: Apr. 25, 2023. [Online]. Available: https://e2e.ti.com/support/wireless_connectivity/f/538/t/197748

[9] L. Barman, A. Dumur, A. Pyrgelis, and J.-P. Hubaux, "Every Byte Matters: Traffic Analysis of Bluetooth Wearable Devices," Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., vol. 5, no. 2, Article 54, June 2021, 45 pages. https://doi.org/10.1145/3463512

[10] Y. Shin, S. Kim, W. Jo, and T. Shon, "Digital Forensic Case Studies for In-Vehicle Infotainment Systems Using Android Auto and Apple CarPlay,"

Sensors, vol. 22, no. 19, pp. 7196, 2022. https://doi.org/10.3390/s22197196

[11] W. Albazrqaoe, J. Huang, and G. Xing, "A Practical Bluetooth Traffic Sniffing System: Design, Implementation, and Countermeasure," in IEEE/ACM Transactions on Networking, vol. 27, no. 1, pp. 71-84, Feb. 2019. https://doi.org/10.1109/TNET.2018.2880970

[12] M. Cominelli, F. Gringoli, P. Patras, M. Lind, and G. Noubir, "Even black cats cannot stay hidden in the dark: Full-band de-anonymization of bluetooth classic devices," in Proceedings of the 2020 IEEE Symposium on Security and Privacy (S&P), 2020.

[13] S. Shrestha, E. Irby, R. Thapa, and S. Das, "SoK: A Systematic Literature Review of Bluetooth Security Threats and Mitigation Measures," in Proceedings of the International Symposium on Emerging Information Security and Applications (EISA) - Copenhagen, Denmark, 2021. Retrieved from http://dx.doi.org/10.2139/ssrn.3959316

[14] A. Barua, M. A. Al Alamin, M. S. Hossain, and E. Hossain, "Security and Privacy Threats for Bluetooth Low Energy in IoT and Wearable Devices: A Comprehensive Survey," in IEEE Open Journal of the Communications Society, vol. 3, pp. 251-281, 2022. https://doi.org/10.1109/OJCOMS.2022.3149732

[15] J. J. Vinagre Díaz, A. B. Rodríguez González and M. R. Wilby, "Bluetooth Traffic Monitoring Systems for Travel Time Estimation on Freeways," in IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 1, pp. 123-132, Jan. 2016, doi: 10.1109/TITS.2015.2459013.

[16] S. Boudabous, J. Garbiso, B. Leroy, S. Clemencon and H. Labiod, "Traffic Analysis Based on Bluetooth Passive Scanning," 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring), Kuala Lumpur, Malaysia, 2019, pp. 1-6, doi: 10.1109/VTCSpring.2019.8746452.