FlexHash: Hybrid Locality Sensitive Hashing for IoT Device Identification

Nathan Thom*, Jay Thom*, Batyr Charyyev, Emily Hand, Shamik Sengupta
Department of Computer Science and Engineering, University of Nevada, Reno, USA
1664 N. Virginia street m/s 0171 Reno, NV 89557 775-784-6905
Email: nathanthom@nevada.unr.edu, jthom@unr.edu, bcharyyev@unr.edu, emhand@unr.edu, ssengupta@unr.edu

***Abstract*—Recent growth in the utilization of IoT has offered convenience and utility, but has also increased security risk. Many devices lack the capacity to support adequate encryption or other common means of protection, and are often designed for easy connection *out-of-the-box* exposing vulnerabilities related to default configurations. Managing IoT devices in a network can be difficult as MAC addresses are easily spoofed, creating a need for techniques to properly identify and monitor membership. Many of the proposed solutions for IoT device identification require complex feature extraction and engineering. In addition, little work has been done to identify individual devices from among identical peers. We propose a novel hashing algorithm, FlexHash, and show that we are able to identify identical devices with a very high degree of accuracy using only a single packet of network traffic. By applying *hybrid* locality sensitive hashing in combination with machine learning our approach achieves accuracy scores as high as 98% for identical devices and 99% for identifying device genre.**

***Index Terms*—IoT Security, Traffic Fingerprinting, IoT Device Identification, Locality Sensitive Hashing, Machine Learning**

I. Introduction

The utility and ease-of-use offered by Internet of Things (IoT) devices offers many benefits, but has also introduced increased risk to data and network systems. For this reason, new methods are required for reliably tracking device behavior and network membership. Various techniques for IoT device identification through network traffic fingerprinting with machine learning have been proposed in the literature, however many require complex feature extraction and engineering which can introduce a high degree of overhead, and require extensive domain knowledge. In addition, the classification of identical devices has not been adequately addressed, an issue that is important when monitoring multiple homogeneous devices. Previous studies also classify devices in lab environments limited to only traffic generated by known devices rather than in realistic network environments. For this reason, we also include background noise in the form of random network traffic, as well as traffic generated from unknown IoT devices to show our system performs well in a realistic setting.

We propose FlexHash, a hybrid locality sensitive hashing (LSH) method. Because IoT devices are typically simple in their functionality, similarity hashes produced by LSH are useful for fingerprinting network traffic. Our approach extends this utility in two ways; first, it combines the benefits of LSH with the power of machine learning, allowing for highly accurate single packet device identification. Second, it optimizes the similarity hashing function by providing the ability to adjust various parameters in the hashing process, allowing the output to be tuned for use with specific sets of devices. By using the hash of the traffic data we avoid the need for feature selection and extraction, a computationally expensive process. We evaluate our method by classifying 3 groups of 8 identical devices; 8 smart plugs, 8 smart light bulbs, and 8 web cameras using *single packets* data. We further validate this method by introducing realistic background noise.

Contributions of this paper are as follows:

- We develop FlexHash, a novel locality sensitive hashing algorithm that enables adjustments to the hashing parameters (accumulator length, window size, and n-grams).
- We implement a network traffic fingerprinting method combining FlexHash with machine learning, and perform accurate IoT device identification requiring only a *single packet* of data.
- We evaluate this system by classifying device genre and *identical devices* in the presence of similar peers while also including realistic *background noise*.
- We collect traffic data from three categories of 8 identical IoT devices, which we share with the research community at *github.com/UNR-IoT-Fingerprinting/FlexHash*.

In the rest of the paper; Section II presents a review of previous studies in this field. Section III provides details of the proposed method and describes the data set used. Section IV presents results and experimental evaluation focusing on the identification of devices by genre, identification of individual devices from a group identical peers, and perfomance with IoT and random network noise. Section V concludes the paper.

II. Related Work

IoT device identification based on captured network traffic has been presented in the literature using a variety of techniques. Existing studies on device identification extract the most representative features from traffic data and pass them to the device identification system. Recent approaches include the use of temporal features [1], network layer features [2], application layer features [3], flow-based features [4], and single-packet identification [5]. Extracted features from network traffic are typically fed into a device identification system. Such

* Equal contributing authors

systems depend on various machine learning approaches such as natural language processing [6], one-class classifiers [7], multi-class classifiers [8], and neural networks [9]. Other studies have focused on a comparison of machine learning models identifying IoT devices based on network traffic [10].

Locality sensitive hashing, also referred to as similarity-preserving hashing, produces similar hash values for similar inputs whereas traditional cryptographic hashes produce entirely different outputs if the inputs are not identical. Nilsimsa [11], ssdeep [12], sdhash [13], and tlsh [14] are the most commonly known n-gram based LSH approaches. In general, these hash functions operate by using a sliding window of a specific size that moves over the input stream one byte at a time. In each iteration, trigrams are generated from the current characters in a window which are then passed to a hash function to generate integer index values. Using these values, frequencies of the observed trigrams in the input are stored in an accumulator array and a hash digest is created based on these stored values. We propose a novel n-gram hash function *FlexHash* and demonstrate its effectiveness in identifying IoT devices. Our method provides an approach combining hybrid hashing and machine learning models to achieve greater accuracy and a more robust solution.

III. Methodology

This section will provide a description of the proposed hashing method *FlexHash* used in combination with machine learning algorithms to identify IoT devices, and will provide details of the data set used for evaluation.

A. *FlexHash*

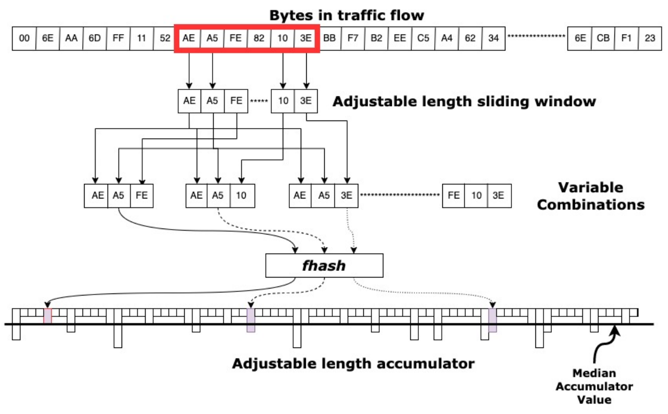
Compared with other n-gram based approaches, FlexHash provides greater flexibility and improved results (Table IV) by allowing parameter tuning (sliding window, n-gram, and digest length) to be used with specific devices. Table I provides the tunable parameters along with the set of ranges chosen for each parameter. Details of FlexHash's functionality are provided in Figure 1. FlexHash utilizes an adjustable size sliding window that reads the input one byte at a time. At

TABLE I: Tunable Parameters of FlexHash

Accumulator Size (in bits)	128, 256, 512, 1024
Window Size	4, 5, 6
Combination Size	[2: Window Size]

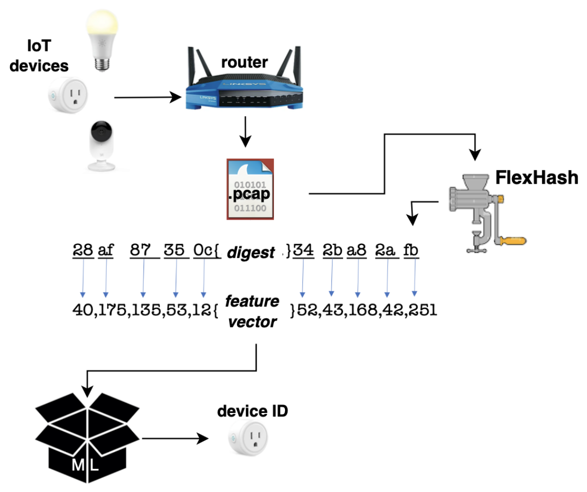
each iteration, all combinations of bytes in the window are generated and given to the hash function *fhash*. This function (fhash) multiplies the integer value of each combination. It then adds a prime number to the result to facilitate greater hash distribution. The remainder of this result (modulo) divided by the length of the accumulator denotes the current index. The value in that index of the accumulator is incremented by one. As a result, the accumulator holds frequencies of the observed combinations in an input. The final hash value of the input will be the same length as the accumulator. For every index in the accumulator, if the value at a given index exceeds the median of all values then the value of that index of the hash is set to 1, otherwise it is set to 0. The window size and accumulator size are adjustable and can be tuned based on the problem and input characteristics. The n-gram (combination) size is also adjustable with the upper limit being the current window size.

To perform optimal parameter selection for a particular device, we run combinations of parameters and test which set provides the best results. Optimal parameters vary based on the device type, making the adjustable property of FlexHash vital to improving accuracy and other performance metrics, especially when identifying identical devices. Optimal parameters for devices found in our evaluations are *1024, 6, and 2* for accumulator, window, and combination size for smart plug and light bulb, and *1024, 4, and 2* for web cameras.



Bytes in traffic flow
00 6E AA 6D FF 11 52 AE A5 FE 82 10 3E BB F7 B2 EE C5 A4 62 34 6E CB F1 23
AE A5 FE 10 3E
Adjustable length sliding window
AE A5 FE
AE A5 10
AE A5 3E
FE 10 3E
Variable Combinations
fhash
Adjustable length accumulator
Median Accumulator Value

Fig. 1: FlexHash functionality.



IoT devices
router
.pcap
FlexHash
28 af 87 35 0c{ digest }34 2b a8 2a fb
40,175,135,53,12{ feature vector }52,43,168,42,251
ML
device ID

Fig. 2: The network traffic is continuously monitored by the device identification system. The traffic data is processed by FlexHash and converted into feature vectors. These feature vectors are given as input to the pre-trained ML model to identify the device that generated the traffic.

B. Identification of IoT Devices

The overview of the IoT device identification system is presented in Figure 2. Traffic data is captured from IoT devices in a network and hashes of the traffic data are generated with FlexHash. These hashes are converted to feature vectors by converting each byte value in the resultant digest to base-10 numerical values ranging from 0-255. For instance, byte value ”FB” would be converted to the integer value 251. These feature vectors are then passed to our device identification system to train the underlying machine-learning model. Once a model is trained the identification system is ready for deployment on a router or micro-controller to serve as a gateway to a network. The device identification system will continuously monitor the traffic data to identify new devices joining the network, or identify changes indicating anomalous behavior. This is done by generating the digest of randomly captured traffic data in the form of packets, converting them to numeric feature vectors, and passing them to a pre-trained machine learning classifier.

As a machine learning framework we use Autogluon-Tabular [15]. We evaluate base models such as LightGBM, XGBoost, extremely randomized trees, extra trees, and multi-layer stack ensembles and select the model with the best performance. The proposed device identification system is different from existing systems as it does not require feature selection from traffic data and focuses on device identification with a single packet.

C. Dataset

We focus on three categories of devices; smart plugs (*Ghome Smart Plug*), smart light bulbs (*General Electric CYNC Full-Color Smart Bulb*), and web cameras (*YI 1080p Home Camera*). Each category of device contains 8 identical devices for a total of 24 devices. Devices are connected to the network and allowed to complete their initial setup phase and then .pcap data is collected on a resting state for 24-hours. Data is then sanitized and header checksums are recalculated using *editcap*, replacing all MAC and IP addresses of the devices with 11:11:11:11:11:11 and 1.1.1.1 respectively to avoid bias introduced by unique addresses. Digests of each packet for each parameter combination are generated and converted to feature vectors as described above. For evaluations, the data was split into 78.8% train, 1.2% validation, and 20% test, and measured in terms of precision (i. e., $TP/(TP + FP)$), recall (i. e., $TP/(TP+FN)$), f1-score (i. e., $2/(1/precision+1/recall)$), and accuracy (i. e., $(TP + TN)/(TP + FP + TN + FN)$) where TP, TN, FP, and FN stand for true positive, true negative, false positive and false negative.

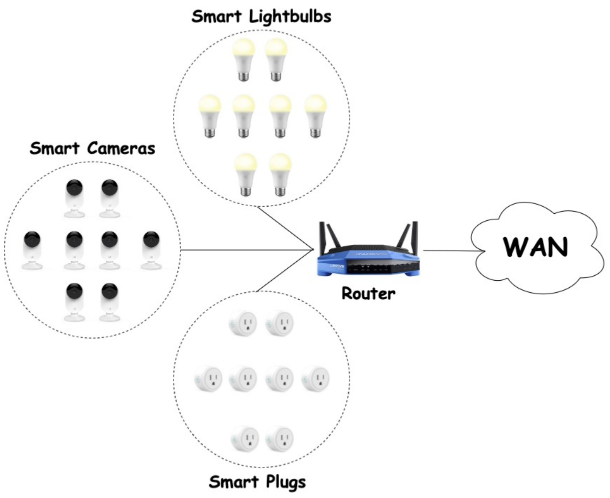


Fig. 3: Traffic data was collected from 3 categories of devices, representing simple to complex set of devices. Each set contains 8 identical devices. Data is collected for 24 hours.

TABLE II: Average performance in identifying device genre in the presence of noise and without noise.

Device	Accuracy		F1 Score	
	Without Noise	With Noise	Without Noise	With Noise
Smart Plugs	99.98	99.89	99.97	99.90
Smart Lights	99.88	99.41	99.92	99.57
Smart Cameras	99.99	99.98	99.99	99.98

IV. Experimental Analysis and Results

In this section, we analyze the performance of our device identification system using FlexHash with the following experiments.

A. Identify devices in the presence of background noise

In a live network traffic noise is inevitable. Thus, it is crucial to explore system performance in a noisy environment. This demonstrates the ability to apply device identification techniques in a realistic network setting when monitoring for IoT device membership or when searching for unknown devices by category. To achieve realistic results, we introduce two types of background noise: *IoT noise* (random IoT traffic from [16]) and *network noise* (random network traffic from a large set of heterogeneous devices. We create hashes of the noise data and add it to device data set labeled as either *network-noise* or *iot-noise*.

B. Identify devices by genre

The identification of devices by genre enables inferring the device type for traffic captured in real-time via a single packet. By monitoring a network and testing random packets we can predict the likelihood that a particular packet belongs to a genre, i.e. the packet is *probably* a camera, or *probably* a smart bulb, etc. To do this, all 24 devices are re-labeled as either smart bulb, smart plug, or web camera, and a multi-class classifier is built. The implication here is that in a network scenario, randomly captured packets identified as being generated from some IoT device can be further categorized as a specific device type. This is useful for investigating unknown or rogue devices on a network that could potentially compromise security.

Results for this experiment are presented in Table II. Results for identification by genre are above 99% for all devices both

TABLE III: Average performance in identifying identical devices from the same category in the presence of noise and without noise.

	Accuracy		F1 Score	
Device	Without Noise	With Noise	Without Noise	With Noise
Smart Plugs	85.79	85.02	85.77	84.91
Smart Bulbs	93.63	89.09	93.60	84.75
Web Cameras	97.89	98.61	97.78	98.55

with and without background noise. We are also able to differentiate the noise type from known device genres, achieving an accuracy for *iot-noise* above 98% and for *network-noise* above 97%.

C. Identification of individual devices from identical peers

In this subsection, we work to identify an individual device from a group of identical peer devices. Rather than attempting to identify a heterogeneous set of unique devices, we tackle the more difficult task of identifying groups of identical devices, a task under-explored in the literature. This type of identification is critical in tracking device behavior and membership as there are often multiple individuals of the same device appearing in a network. Experiments are performed both with and without background noise and results compared.

Average results in terms of both accuracy and F1 score for this experiment are shown in Table III. We see that without background noise, web cameras achieve an accuracy above 97% on average while smart bulbs and smart plugs achieve results above 85% and 93% respectively. We note that in the case of smart plugs, some imbalance was found in the data samples for a 24 hour period, with three of the eight plugs generating considerably more traffic than the other five, possibly accounting for this change in performance. Interestingly, devices with greater complexity appear to perform better than simpler devices when identifying an individual from identical peers. We note here that FlexHash and our device identification system achieve accuracy results in this more difficult scenario that are either competitive or superior to results offered by other approaches in the literature while using only a single packet sample. When running the same experiment with background noise, web cameras, smart bulbs, and smart plugs achieve an average accuracy of 98%, 89%, and 85% respectively. With noise added, we only see a slight degradation of performance in the smart bulbs, with web cameras and smart plugs performing at nearly an equal accuracy.

TABLE IV: Comparison of average performance, FlexHash vs Nilsimsa in identifying identical devices from the same category.

	Accuracy		F1 Score	
Device	FlexHash	Nilsimsa	FlexHash	Nilsimsa
Smart Plugs	**85.79**	72.97	**85.77**	73.27
Smart Bulbs	**93.63**	78.04	**93.60**	80.48
Web Cameras	**97.89**	84.74	**97.78**	84.66

We compare the performance of FlexHash with Nilsimsa because in our previous studies in device identification [17] with LSH we observed that Nilsimsa outperforms other hashing methods such as those mentioned above. Thus, we can assume that better results with FlexHash over Nilsimsa will imply better results over other hashing techniques as well. Average identification results for this experiment are presented in Table IV

Identification in this experiment is done without background noise. For web cameras FlexHash achieves an accuracy of 97.74% compared to 84.74% for Nilsimsa. For smart bulbs FlexHash achieves an accuracy of 93.63% compared to 78.04% for Nilsimsa. Finally, for smart plugs FlexHash achieves an accuracy of 85.79% compared to 72.97% for Nilsimsa. For web cameras, smart bulbs, and smart plugs this represents a percent increase of 13.00%, 15.59%, and 12.82% respectively. We see here that in every case FlexHash achieves a significant increase in accuracy over Nilsimsa hashing, an indication of the effectiveness of FlexHash's tunable parameters.

V. Conclusion

In this paper, we introduced the LSH method *FlexHash*, which enables highly accurate *single packet* traffic fingerprinting without the complexities of feature extraction and engineering. We propose FlexHash in combination with machine learning which can monitor devices on a network to aid in device membership, identification, and anomaly detection. We evaluated the proposed system in identifying devices by individuals among identical peers and by genre in the presence of background traffic noise, and make our data set of identical devices available to the research community.

Acknowledgment

This material is based upon work supported by the National Science Foundation under award number 2019164. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] M. Mazhar and Z. Shafiq, "Characterizing smart home iot traffic in the wild," in *2020 IEEE/ACM IoTDI*. IEEE, 2020, pp. 203–215.

[2] Y. Meidan, M. Bohadana, A. Shabtai, J. Guarnizo, M. Ochoa, N. Tippenhauer, and Y. Elovici, "Profiliot: A machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the symposium on applied computing*, 2017, pp. 506–509.

[3] I. Ullah and Q. Mahmoud, "Network traffic flow based machine learning technique for iot device identification," in *2021 IEEE SysCon*. IEEE, 2021, pp. 1–8.

[4] A. Sivanathan, H. Gharakheili, and V. Sivaraman, "Inferring iot device types from network behavior using unsupervised clustering," in *2019 IEEE LCN*. IEEE, 2019, pp. 230–233.

[5] R. Chowdhury, S. Aneja, N. Aneja, and E. Abas, "Network traffic analysis based iot device identification," in *Proceedings of the 2020 the 4th International Conference on Big Data and IoT*, 2020, pp. 79–89.

[6] F. Le, J. Ortiz, D. Verma, and D. Kandlur, *Policy-Based Identification of IoT Devices Vendor and Type by DNS Traffic Analysis*. Cham: Springer International Publishing, 2019, pp. 180–201.

[7] Y. Meidan, V. Sachidananda, Y. Elovici, and A. Shabtai, "Privacy-preserving detection of iot devices connected behind a nat in a smart home setup," in *arXiv:1905.13430*, 2019.

[8] B.Bezawada, M.Bachani, J. Peterson, H.Shirazi, I. Ray, and I. Ray, "Behavioral fingerprinting of iot devices," in *Proceedings of the 2018 WRKSHP on Attacks and Solutions in Hardware Security*, 2018, p. 41–50.

[9] S. Dong, Z. Li, D. Tang, J. Chen, M. Syn, and K. Zhang, "Your smart home can't keep a secret: Towards automated fingerprinting of iot traffic with neural networks," in *arXiv:1909.00104*, 2019.

[10] A. J. Pinheiro, J. de M. Bezerra, C. A. Burgardt, and D. R. Campelo, "Identifying iot devices and events based on packet length from encrypted traffic," *Computer Communications*, vol. 144, pp. 8 – 17, 2019.

[11] E.Damiani, S.Vimercati, S.Paraboschi, and P.Samarati, "An open digest-based technique for spam detection." *ISCA PDCS*, vol. 2004, p. 559:564, 2004.

[12] N.Sarantinos, C.Benzaid, O.Arabiat, and A.Al-Nemrat, "Forensic malware analysis: The value of fuzzy hashing algorithms in identifying similarities," in *Trust/BigDataSE/ISPA*. IEEE, 2016, pp. 1782–1787.

[13] F. Breitinger and H. Baier, "Properties of a similarity preserving hash function and their realization in sdhash," in *2012 Information Security for South Africa*. IEEE, 2012, pp. 1–8.

[14] J. Oliver, C. Cheng, and Y. Chen, "Tlsh–a locality sensitive hash," in *2013 IEEE CTC*. IEEE, 2013, pp. 7–13.

[15] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola, "Autogluon-tabular: Robust and accurate automl for structured data," *arXiv preprint arXiv:2003.06505*, 2020.

[16] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. Sadeghi, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in *2017 IEEE ICDCS*, 2017, p. 2177:2184.

[17] J. Thom, N. Thom, S. Sengupta, and E. Hand, "Smart recon: Network traffic fingerprinting for iot device identification," in *2022 IEEE CCWC*. IEEE, 2022, pp. 0072–0079.