I Can Hear Your Alexa: Voice Command Fingerprinting on Smart Home Speakers

Sean Kennedy[†*], Haipeng Li[†*], Chenggang Wang[†*], Hao Liu[†], Boyang Wang[†], Wenhai Sun[§]
[†]University of Cincinnati, [§]Purdue University
{*kenneds6, li2hp, wang2c9, liu3ho*}*@mail.uc.edu, boyang.wang@uc.edu, whsun@purdue.edu*

***Abstract*—Millions of smart home speakers, such as Amazon Echo and Google Home, have been purchased by U.S. consumers. However, the security and privacy of smart home speakers have not been rigorously examined, which raise critical security and privacy concerns. In this paper, we investigate untold and severe privacy leakage of smart home speakers. Specifically, we examine a new passive attack, referred to as *voice command fingerprinting attack*, on smart home speakers. We demonstrate that a passive attacker, who can only eavesdrop encrypted traffic between a smart home speaker and a cloud server, can infer users' voice commands and compromise the privacy of millions of U.S. consumers. We formulate the attacks by harnessing machine learning algorithms. In addition to leveraging accuracy, we propose a new privacy metric, named *semantic distance*, to assess the privacy leakage with natural language processing. Our experiment results on a real-world dataset suggest that voice command fingerprinting attacks can correctly infer 33.8% of voice commands by eavesdropping encrypted traffic. Our results also show that existing padding methods can diminish an attacker's accuracy to 14.7%, but would cause high communication overhead (548%) and long time delay (330%).**

I. Introduction

A smart home speaker is a voice-controlled device, which is capable of making phone calls, playing music, controlling other IoT devices and providing real-time information, such as weather and traffic. For instance, a user can give a voice command "What is the weather today?" to an Amazon Echo after saying a *wake word* "Alexa", and the device will respond the voice command with current weather information. As of September 2018, Amazon Echo and Google Home dominate the smart home speaker market with a share of 70% and 24% respectively [1]. Over 19.8 million smart home speakers were sold in the third quarter of 2018. The market value of smart home speakers is expected to reach $30 billion in 2024 [1].

While the technology of smart home speakers is transforming the smart homes of U.S. consumers, its security and privacy capabilities have not been rigorously investigated, which leave critical security and privacy concerns. For instance, a private conversation of a couple living in Oregon was recorded by an Amazon Echo and sent to a contact without their knowledge [2]. Research studies show that an *active attacker* can maliciously activate a smart home speaker using background sound, which is similar as a wake word but cannot be noticed by a human [3]. In light of these security threats, it is imperative to investigate the security and privacy of smart home speakers.

*The first three authors contribute equally in this paper.

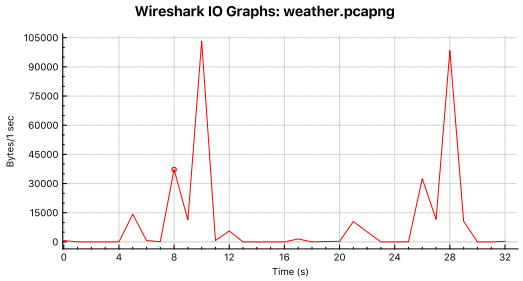

Fig. 1. The encrypted traffic of "*What is the weather?*" on Amazon Echo (2nd generation). This voice command is repeated twice in this figure.

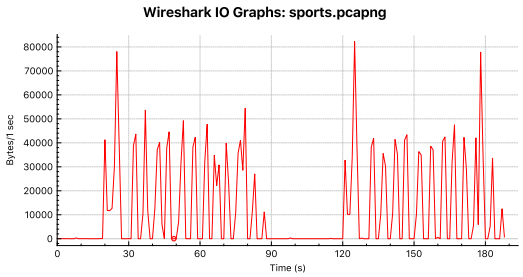

Fig. 2. The encrypted traffic of "*Sports updates.*" on Amazon Echo (2nd generation). This voice command is repeated twice in this figure.

In this paper, we study *untold and severe* privacy leakage of smart home speakers under a passive attack, which is referred to as *voice command fingerprinting attack*. Specifically, we demonstrate that a passive attacker, who can only eavesdrop encrypted traffic between a smart home speaker and a cloud server, can infer the content of a voice command, which is *supposed to be protected by existing smart home speakers*. The intuition behind this voice command fingerprinting attack is that every voice command and its response, although being encrypted, have a *unique* traffic pattern through *packet length*, *direction*, *order*, etc.. For instance, as shown in Fig. 1 and Fig. 2, the traffic pattern of a same voice command obtained from Amazon Echo is almost the same while the traffic pattern of the two different commands can be easily distinguished.

A voice command fingerprinting attack is essentially a type of traffic analysis [4], which can be formulated as a *machine learning* problem. Similar fingerprinting attacks have been widely investigated for identifying websites over encrypted traffic [5]–[14], but have not drawn attention on smart home speakers. Although the traffic analysis methods we apply

978-1-5386-7117-7/19/$31.00 ©2019 IEEE

in this paper are selected from existing research in website fingerprinting, we aim to answer two critical questions:

1) *Can website fingerprinting attack algorithms effectively classify encrypted traffic of voice commands?*
2) *What other critical factors should we consider in order to complement the privacy metric?*

The major contributions of this paper are summarized below:

- We apply and customize four traffic analysis methods in website fingerprinting to study the privacy leakage of smart home speakers under voice command fingerprinting attacks. These attacks utilize Jaccard similarity, Naive Bayes or AdaBoost to classify traffic traces. We build a real-world dataset by collecting $1,000$ encrypted traffic traces of an Amazon Echo (2nd generation) for 100 common voice commands. The traffic data was collected from October 2018 to December 2018. We implement voice command fingerprinting attacks in Python with 2,500 lines of code. Our dataset and code are available on Github [15] for research purposes only.
- The experimental results on our dataset show that an eavesdropper collecting encrypted traffic can correctly infer the content of a voice command with over 33.8% accuracy. Moreover, our study shows that the results of these attack algorithms under voice command fingerprinting are *not as effective* as their results in website fingerprinting, which suggests voice command fingerprinting is a more challenging problem.
- Most of the existing website fingerprinting attacks adopt accuracy as the only metric to determine the effectiveness of an attack, where a higher accuracy suggests a more severe privacy leakage. However, this all-or-nothing metric does not measure the cases where an attacker can infer a different but similar voice command. To address this limitation, we propose to leverage *semantic distance* to assess privacy leakage in addition to using accuracy. Specifically, we leverage a natural language processing primitive, named *doc2vec* [16], to evaluate the semantic distance between two voice commands, and calculate the average *normalized semantic distance* under the attacks. Our results demonstrate that semantic distance complements the privacy assessment.
- We apply one existing padding method, named BuFLO [8], as a countermeasure and examine its effectiveness under voice command fingerprinting attacks. Our results show that it can mitigate privacy leakage against the attacks, but would cause high communication overhead and extremely long time delay. For instance, BuFLO can diminish attack accuracy to 14.7%, but would have to introduce 932 KB (548%) additional communication overhead and cause 17.02 seconds (330%) time delay on average for each voice command and its response. This tradeoff is extremely high and would dramatically affect user experience of smart home speakers.
- Our findings suggest that voice command fingerprinting attacks are severe and could compromise the privacy of millions of smart homes. Consumers, service providers, stakeholders and the government should be aware of this serious privacy leakage. New protocols and countermeasures for smart home speakers need to be developed in order to mitigate privacy leakage against voice command fingerprinting attacks with *affordable tradeoffs*.

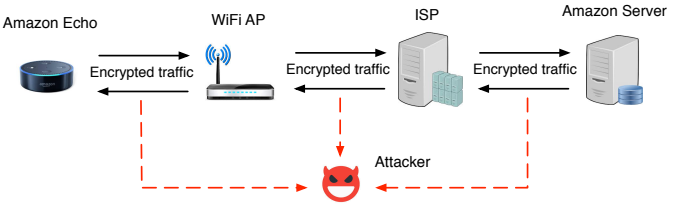

Fig. 3. The system model.

II. Background

A. System Model

Our system model is illustrated in Fig. 3. It includes a smart home speaker, a WiFi access point, an Internet Service Provider, and a cloud server. A user can speak a wake word, e.g., "*Alexa*", and send a voice command, e.g., "*What is the time?*", to a smart home speaker, e.g., Amazon Echo. The smart home speaker will forward this voice command (also referred to as a query in this paper) to the cloud server, e.g., Amazon. The cloud server will process the query, return a response to the smart home speaker, and the smart home speaker will play the response to the user.

We assume each voice command can be correctly recognized by the cloud server. The WiFi access point and the Internet Service Provider ensure the smart home speaker and the cloud server is well connected. We assume standard encryption algorithms and security protocols have been applied between the smart home speaker and the cloud server to secure the content of packets. For instance, Amazon Echo leverages TLS 1.2 and all the packets are encrypted by AES.

In this model, we assume there is a *passive attacker*, who can eavesdrop the traffic between a smart home speaker and the cloud server. This passive attacker could be an attacker on a user's local network or the Internet Service Provider. We assume that the attacker knows which type of smart home speaker a user has (e.g., whether it is an Amazon Echo or a Google Home). We also assume that this attacker learns the source and destination of packets by observing packet headers, but does not directly learn the content of packets.

We assume that each traffic trace contains one voice command and the corresponding response, and there is no noisy traffic produced by other devices or services. Similar as website fingerprinting attacks, we assume an attacker knows the start time and the end time of each traffic trace. *The objective of this attacker is to infer which voice command a user asks by analyzing an encrypted traffic trace it has captured.*

We study a *closed-world setting*, where this attacker knows a traffic trace is definitely associated with one of the voice commands in a given prior set. The closed-world setting

has been widely adopted in website fingerprinting [5]–[8], [10], [13]. A voice command fingerprinting attack can be formulated as a machine learning problem, where the attacker can produce labeled traffic traces as training data and utilize unlabeled traffic traces captured from a user's device as test data. We assume the training data for some time-sensitive voice commands (e.g., weather) can be generated by the attacker within a certain time frame (e.g., within a couple of hours) after the capture of test data, such that the responses of voice commands are the same or similar in plaintext.

B. Definitions

Each traffic trace is essentially a sequence of packets with timestamps, where each packet is either from or to a smart home speaker. Assume a voice command is denoted as Q, and the traffic trace of this voice command and its response is T_Q. We can represent a traffic trace as a sequence of tuples

$$T_Q = \langle (t_1, l_1, b_1), (t_2, l_2, b_2), ..., (t_m, l_m, b_m) \rangle \tag{1}$$

where m is the total number of packets, t_i is the timestamp, l_i is the packet length, and b_i is the direction of the i-th packet. Each timestamp is represented in seconds, where the timestamp of the first packet is set as $t_1 = 0$. Packet length $l \in [1, 1500]$ in bytes, where 1500 is the maximum packet size, and packet direction $b \in \{+1, -1\}$, where $+1$ indicates a packet is from a smart home speaker to the cloud server and -1 otherwise.

We refer the above format as the *generic form* of a traffic trace. Any feature that can be further utilized for a voice command fingerprinting attack can be extracted from the generic form. We assume the voice commands in a closed-world setting can be represented as a prior set $\mathcal{Q} = \{Q_1, ..., Q_M\}$, where M is the total number of voice commands (or the total number of classes).

C. Privacy Metric

Accuracy. The effectiveness of a voice command fingerprinting attack, or the privacy leakage of encrypted traffic on smart home speakers, can be evaluated based on *accuracy*. Specifically, assume there are N unlabeled traffic traces, an attacker can correctly infer the voice commands of α unlabeled traffic traces, then the accuracy of a voice command fingerprinting attack can be represented as

$$Accuracy = \frac{\alpha}{N} \tag{2}$$

Given a traffic trace T_Q and an attacker's guess Q' on this traffic trace, we say that this attacker infers this traffic trace correctly iff $Q = Q'$. A higher accuracy under an attack implies more privacy leakage.

Accuracy has been utilized in website fingerprinting as the main metric to assess privacy leakage. Same as website fingerprinting, we also assume the frequency distribution of labeled traffic traces in the prior set $\mathcal{Q}$ is uniformly distributed. In other words, without performing fingerprinting attacks, an adversary will take a random guess and this adversary's accuracy is $1/M$ in a closed-world setting, where M is the number of voice commands in the prior set.

Distance. Most of the website fingerprinting attacks leverage accuracy as the only metric to assess privacy leakage. However, this all-or-nothing metric [17] may not be adequate under a voice command fingerprinting attack. For instance, an attacker may infer a voice command incorrectly, i.e., $Q \neq Q'$, but the semantic meaning of two voice commands are very similar or close, which also indicates critical privacy leakage. For instance, the following two voice commands are not exactly the same but very similar.

What is the weather?
What is the weather tomorrow?

To address this limitation in the current literature, we propose to leverage *distance*, or more precisely *semantic distance*, as our metric to assess privacy leakage in addition to using accuracy. Specifically, given two voice commands Q and Q', we first transform each command into a *semantic vector* based on its semantics by leveraging a natural language processing primitive, named *doc2vec* [16]. Next, we evaluate the distance of the two vectors in *cosine similarity*. Assume the two semantic vectors of command Q and Q' are V_Q and $V_{Q'}$ respectively, the semantic distance of these two vectors can be calculated as

$$D(Q, Q') = \frac{V_Q \cdot V_{Q'}}{||V_Q|| ||V_{Q'}||} \tag{3}$$

where $D(Q, Q') \in [-1, 1]$. A semantic distance of 1 indicates two voice commands are completely the same while a value of -1 implies the two commands are totally opposite. We choose cosine similarity because it is commonly used for texts in natural language processing. We will briefly elaborate the details of doc2vec in the next subsection.

Unfortunately, we notice that *simply applying this semantic distance may still not be sufficient in a closed-world setting*. For instance, the semantic distance of two different voice commands Q and Q' suggests that the two are very dissimilar, but voice command Q could be the closest one to Q' in the prior set $\mathcal{Q}$. On the other hand, the semantic distance from voice command Q to Q' could be close to 1, but there might be other voice commands in the prior set $\mathcal{Q}$ that are more similar to Q.

To tackle this gap in the privacy assessment, we adopt *normalized semantic distance*. Specifically, assume an attacker infers Q' as the voice command of a traffic trace T_Q, where $Q \in \mathcal{Q}$. We compute a semantic distance of Q' to each voice command Q_i in prior set $\mathcal{Q}$, and sort these semantic distances descendingly. The order or the rank of $D(Q, Q')$ in this sorted list is the normalized semantic distance of voice command Q' to Q. The algorithm of calculating a normalized semantic distance is summarized in Algo. 1.

D. Doc2vec

Doc2vec [16] is a natural language processing primitive, which can transform a document or a paragraph into a vector. It was proposed by Le and Mikolov in 2014. Doc2vec is an unsupervised learning model built upon *word2vec* [18], which can

Algorithm 1: NormDistance$(Q, Q', \mathcal{Q})$

Input: Two voice commands Q and Q', and a prior set $\mathcal{Q}$
Output: A normalized semantic distance $D_{Norm}(Q, Q')$

$\mathcal{D} = \varnothing$;
$D_{Norm}(Q, Q') = 0$;
for $i = 1; i \leq M; i++$ **do**
  $d_i = D(Q', Q_i)$;
  $\mathcal{D} = \mathcal{D} \cup d_i$
end for
$\mathcal{D}^* = \{d_1^*, ..., d_M^*\} \leftarrow \mathsf{Sort}(\mathcal{D})$;
$d' = D(Q, Q')$;
for $j = 1; j \leq M; j++$ **do**
  if $d' == d_j^*$ **then**
    $D_{Norm}(Q, Q') = j - 1$;
    break;
  end if
end for
return $D_{Norm}(Q, Q')$;

present each word as a *fixed-length* vector. More importantly, doc2vec takes into consideration of the semantics of words and the order of words while generating the outputs, such that the distance of vectors produced by similar documents are close. It outperforms previous models, such as bag-of-words and bag-of-n-grams. More details regarding doc2vec can be found in [16]. For ease of presentation, we refer to a vector output by doc2vec as a semantic vector.

III. Voice Command Fingerprinting Attacks

A. Attack Algorithms

In this paper, we choose four existing attack algorithms in website fingerprinting, and apply them in voice command fingerprinting. We refer to the four algorithms as LL-Jaccard [5], LL-NB [5], VNG++ [8] and P-SVM [7] respectively. Previous studies have shown that these attack algorithms are very effective in website fingerprinting. For instance, LL-NB can achieve over 80% of accuracy in website fingerprinting. We elaborate on the details of each attack below.

LL-Jaccard and LL-NB. Both LL-Jaccard and LL-NB are proposed in [5]. The authors leverage *packet size* and *direction* to perform website fingerprinting. Each traffic trace is represented as a set, where each element is a positive or negative integer corresponding to the packet length and packet direction. This set can be obtained based on the generic form of a traffic trace described in Eq. 1.

The authors leverage two approaches to attack. We denote the first method as LL-Jaccard and the second method as LL-NB in this paper. LL-Jaccard utilizes Jaccard similarity to classify unlabeled traffic trace, where the Jaccard similarity of two sets X and Y can be computed as

$$J = \frac{|X \cap Y|}{|X \cup Y|} \tag{4}$$

Given a set Y for an unlabeled traffic trace and sets $\{X_1, ..., X_M\}$ for prior set $\mathcal{Q} = \{Q_1, ..., Q_M\}$, where X_j is obtained from Q_j and $j \in [1, M]$, the method finds $Q_i \in \mathcal{Q}$ such that

$$\arg\max_{i \in [1,M]} \frac{|X_i \cap Y|}{|X_i \cup Y|} \tag{5}$$

LL-NB exploits a Naive Bayes classifier, where the features in a set Y are assumed to be independent. The probability of a set Y, where $Y = \{y_1, ..., y_n\}$ belonging to a particular class Q_i can be calculated as

$$Pr(Q_i|Y) \propto Pr(Q_i) \prod_{j=1}^{n} Pr(y_j|Q_i) \tag{6}$$

and the classifier finds Q_i for Y such that

$$\arg\max_{i \in [1,M]} Pr(Q_i) \prod_{j=1}^{n} Pr(y_j|Q_i) \tag{7}$$

Details of this scheme can be found in [5] and additional details regarding Naive Bayes classifier can be found in [19].

VNG++. Dyer et al. [8] propose a website fingerprinting attack named VNG++ (which stands for variable n-gram), that leverages coarse information of traffic trace, including *total trace time*, *upstream/downstream total bytes*, and *bytes in traffic bursts*. A burst is defined as a sequence of consecutive packets going a same direction. For instance, given a sequence of 5 packets

$$(30, +1), (40, +1), (100, -1), (200, -1), (350, +1)$$

Variable n-gram converts it to a sequence of 3 bursts as

$$(70, +1), (300, -1), (350, +1)$$

Here n indicates the number of packets in each burst and it varies. This attack also leverages Naive Bayes classifier to infer the class of unlabeled traffic traces. According to the results in [8], VNG++ is more effective than LL-Jaccard and LL-NB in website fingerprinting, especially on obfuscated traffic data.

P-SVM. Panchenko et al. apply Support Vector Machine (SVM) to classify websites [7]. We denoted this attack as P-SVM in this paper. P-SVM considers features including *bursts* (denoted as size markers in [7]), *total transmitted bytes*, *the number of bursts*, *occurring packet sizes*, *percentage incoming packets* and *the number of packets*. We discard some features, such as packet size 52 and HTML markers, in P-SVM that obviously do not fit for voice command fingerprinting attacks.

In our experiments described in the next section, since the results based on the implementation with SVM only achieves 1.2% accuracy and we could not find optimized parameters of SVM to improve the attack results, we implement this attack with the same features but utilize AdaBoost as an alternative classifier.

B. Countermeasures

In addition to carrying out existing attacks, we also apply countermeasures in website fingerprinting and investigate the effectiveness of the countermeasures in the scenario of voice command fingerprinting.

Specifically, we implement one countermeasure, BuFLO (Buffered Fixed-Length Obfuscation) [8], which obfuscates traffic trace by sending packets at a fixed-size with fixed intervals. Previous work already demonstrated that BuFLO is more effective than other straightforward padding methods [8]. Given a traffic trace, the obfuscated trace with BuFLO is decided by three parameters, d, ρ and τ. Parameter d is the size of fixed-length packets, ρ is the rate or frequency at which packets need to be sent, and τ determines the minimum amount of time sending packets. As necessary tradeoffs, BuFLO would cost additional communication overhead and delay. Details of BuFLO can be found in [8].

C. Packet Capture Setting and Dataset

Setting. In our experiment, we use an Amazon Echo (2nd generation) as the smart home speaker, leverage a Raspberry Pi 3 Model B as a WiFi access point, and use Wireshark to capture traffic traces.

Dataset. We collect traffic traces of 100 popular voice commands for our study in a closed-world setting. Since we do not have access to the statistics of all the consumers' voice commands submitted to Amazon, we select 100 popular voice commands by mainly leveraging Amazon Echo service emails (from August 2018 to November 2018). Amazon sends weekly service emails entitled "*What's new with Alexa?*" to Echo users and indicates some most requested commands each week. In addition, we also utilize a recent study in [20] as supplementary references to select voice commands. Sciuto et al. [20] conducted a study on over 278,000 commands collected from 75 Amazon Echo users. The authors summarize statistic information and identify some popular commands.

For each voice command, we collect 10 traffic traces, where 8 of them are used as training traces and 2 are used as test traces. Overall, we have 800 traffic traces as training data, and 200 traffic traces as test data. All those traffic traces were captured from October 2018 to December 2018. The traffic traces are collected by two students on the same Amazon Echo (2nd generation). Each student collects 500 traffic traces, i.e., 5 traffic traces for each voice command in the list. The list of our 100 common voice commands and our traffic dataset can be found on Github [15].

Note that the number of classes we choose in this study is similar as the ones in website fingerprinting attacks. For instance, VNG++ considers attacks on 128 different websites. A recent website fingerprinting attack [13] based on deep learning uses 95 different websites. On the other hand, the scale of training data we have in this paper is smaller than the scale of training data in website fingerprinting, which normally consists of thousands or even millions of traffic traces.

The main reason is that the collection of website traffic traces can be performed automatically (e.g., by implementing a crawler), while the capture of voice command traffic traces produced by smart home speakers has to be conducted verbally and manually in our experiments. Not being able to expand the scale of our traffic dataset is also the main reason that we do not apply advanced attacks [13] based on deep learning algorithms.

IV. Performance Evaluation

In this section, we examine the performance of voice command fingerprinting based on our dataset. We first investigate the results under different attack algorithms, examine the impact of several parameters on privacy, and then evaluate the effectiveness of BuFLO.

A. Experiment Setting

We write code in Python to process traffic into generic forms, extract features of traffic traces, implement different attacks and produce obfuscated traffic traces. Our code has around 2,500 lines in total. It is available on Github [15] for research purposes only. We leverage `scikit-learn` library for machine learning algorithms. We run our experiments on Ubuntu 14.04 with 32GB memory and an Intel Core i7-7700 3.6GHz CPU.

B. Doc2vec Training

To obtain semantic vectors of voice commands, we utilize the `Gensim` library [21] to build doc2vec models. To train the model of doc2vec, we exploit two datasets, including Yahoo! Answers Manner Questions (version 2.0) [22] and Quora question pairs dataset [23]. The Yahoo! dataset is a subset of questions and answers posted on Yahoo! Answers since 2007. It includes 142,627 questions and corresponding answers. The size of this dataset is 104 MB. The Quora dataset includes queries from the online knowledge-sharing platform Quora. It includes over 400,000 lines of potential question duplicate pairs. The size of the Quora dataset is 58.2 MB.

We train two doc2vec models, one is based on Yahoo! dataset and the other one is produced by the Quora dataset. We train two different models based on two different datasets to avoid (or minimize) the impact of a training dataset itself on our observation in terms of semantic distance. In the training of the Yahoo! doc2vec model, we select the number of epochs as 100 and the length of semantic vectors as 300. It takes us 23.5 hours to train this doc2vec model. The size of this trained doc2vec model based on Yahoo! dataset is about 7.78 GB. Once we obtain this doc2vec model, it costs 0.02 seconds on average to produce a semantic vector for each voice command. We use the same parameters to train Quora doc2vec model. It takes 3.39 hours to train and the size of the trained model is 2.23 GB.

We also explore some other datasets to train doc2vec models in our experiments, but the results are not as good as the results based on this Yahoo! dataset or Quora dataset. It is likely because this Yahoo! dataset (or Quora dataset) includes questions and answers, which have very similar context as the voice commands sent to a smart home speaker. If we evaluate the semantic distance of two voice commands, "*What is the*

TABLE I
RESULTS OF VOICE COMMAND FINGERPRINTING ATTACKS ON OUR DATASET

	Accuracy	Semantic Distance (SD)		Normalized SD	
	Average	Mean (Yahoo!)	Mean (Quora)	Mean (Yahoo!)	Mean (Quora)
LL-Jaccard	17.4%	0.949	0.914	46.99	45.21
LL-NB	33.8%	0.955	0.924	34.11	33.25
VNG++	24.9%	0.950	0.916	43.80	42.67
P-SVM (AdaBoost)	33.4%	0.956	0.928	37.68	38.06
Random Guess	1%	N/A	N/A	49.5	49.5

weather?" and "*What is the weather tomorrow?*" using the doc2vec model trained by the Yahoo! dataset, the distance is 0.98. The distance of the two commands is 0.93 in the Quora doc2vec model.

C. Privacy Leakage under Attacks

Accuracy under Attacks. We first apply the four attacks, LL-Jaccard, LL-NB, VNG++ and P-SVM (AdaBoost), on our traffic dataset and report our results. As illustrated in Table I, an attacker can identify significant privacy information by applying a voice command fingerprinting attack. For instance, an attacker can correctly infer 33.8% of voice commands by running LL-NB. Other attacks are also very effective to reveal private information based on the encrypted traffic of a smart home speaker. For example, with VNG++ and P-SVM (AdaBoost), an attacker could guess 24.9% and 33.4% of voice commands respectively. All four attacks can achieve better results than random guessing (i.e., 1%).

On the other hand, our results under voice command fingerprinting also suggest that these attacks are *not as effective as* their results under website fingerprinting attacks. For instance, all these algorithms can achieve over 80% of accuracy in website fingerprinting. This observation also implies that voice command fingerprinting is more challenging than the problem of website fingerprinting. More advanced attack algorithms may need to be devised in order to help us better understand the privacy of smart home speakers.

Since our dataset is relatively small, we apply *5-fold cross validation* to evaluate the accuracy of each attack. For instance, to assess the accuracy of LL-Jaccard, we perform the attacks five times. Each time we take 8 traffic traces as training data and leave 2 traffic traces as test data. Different 8 training traffic traces are selected across the five tests. The average accuracy in the table is calculated based on the five tests in cross validation. We also implement 5-fold cross validation in LL-NB, VNG++ and P-SVM (AdaBoost).

For LL-NB and VNG++, to reduce the number of dimensions (or the number of features) for the Naive Bayes model, we further adopt *rounding* on some of the features considered in the two attacks. Specifically, for the results in Table I, we round the size of packets to the nearest multiple of 100 bytes in LL-NB, and we round the size of bursts to the nearest multiple of 5000 bytes in VNG++ in our experiments. Rounding is also used in VNG++ in website fingerprinting to reduce the number of dimensions [8].

Semantic Distance under Attacks. In addition to accuracy, we also evaluate the semantic distance of each attack. For instance, the average semantic distance in LL-Jaccard is 0.949 (Yahoo!) and 0.914 (Quora). We can see that for semantic distance, the actual value is different if the semantic vectors are produced by different trained doc2vec models. On the other hand, their corresponding normalized semantic distance is about the same even with different trained doc2vec models. For example, the normalized semantic distance in LL-NB is 34.11 (Yahoo!) and 33.25 (Quora) respectively. If an attacker performs random guessing, then the expected value of the normalized semantic distance is $\frac{1}{M}\sum_{i=0}^{M-1} i = 49.5$, where $M = 100$.

We can also observe that the results in normalized semantic distance are consistent with the results in accuracy in Table I, where an attacker with a higher accuracy has a lower normalized semantic distance. Moreover, we can observe that semantic distance, or more specifically, normalized semantic distance complements the privacy metric of voice command fingerprinting attacks. With normalized semantic distance, we have a better understanding regarding (on average) how close an attacker could infer under fingerprinting attacks. This evaluation cannot be assessed directly if only accuracy is utilized as the privacy metric.

D. The Impact of Different Parameters

With the initial results we have, we move forward to study the impact of several important parameters on the privacy leakage under voice command fingerprinting attacks.

Impacts of Rounding. Since both LL-NB and VNG++ need to select a rounding parameter in the attack, we also investigate the impact of the rounding parameter under the two attacks.

For LL-NB, the rounding parameter affects the size of each packet. we change the rounding parameter between $[10, 100]$ and apply the attack with different parameters. As shown in Fig. 4, the attack performs the best when the packet size is rounded to the multiple of 100, where the accuracy is 33.8% and the normalized semantic distance is 34.11 (Yahoo!). We can also see that the change of normalized semantic distance is consistent across outputs generated by the Yahoo! doc2vec model and the Quora doc2vec model.

For VNG++, the rounding parameter affects the burst size. According to the results in Fig. 5, we can see that the burst size does not significantly affect the results of the attacks if we change the value from 1,000 to 1,0000. Specifically, the accuracy is always around 25% and the normalized semantic distance is about 44 (Yahoo!).

Impacts of the Number of Epochs. We train a doc2vec model with different number of epochs and examine whether

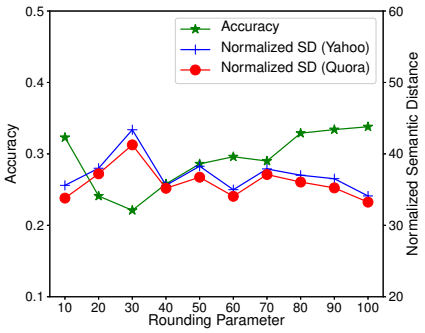

Fig. 4. The impact of rounding parameter on the attack results in LL-NB.

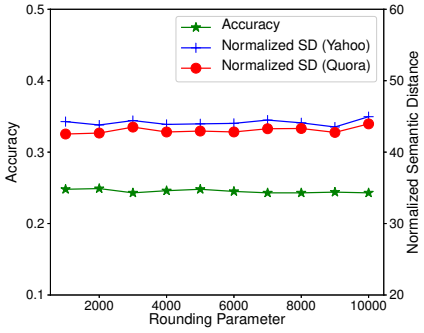

Fig. 5. The impact of rounding parameter on the attack resu;ts in VNG++.

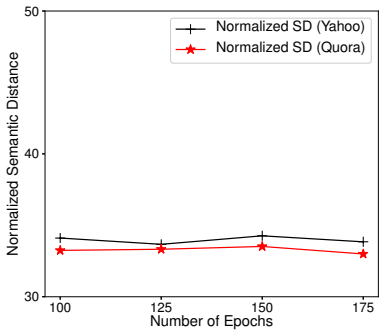

Fig. 6. The impact of the number of epochs on the attack results in LL-NB.

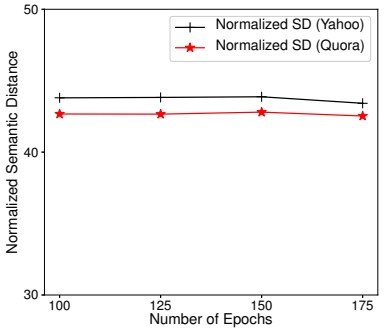

Fig. 7. The impact of the number of epochs on the attack results in VNG++.

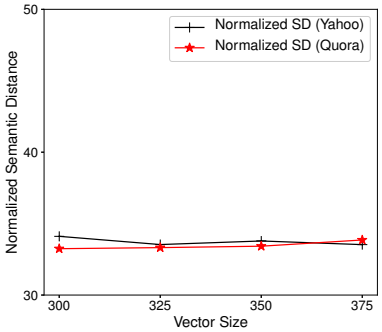

Fig. 8. The impact of the vector size on the attack results in LL-NB.

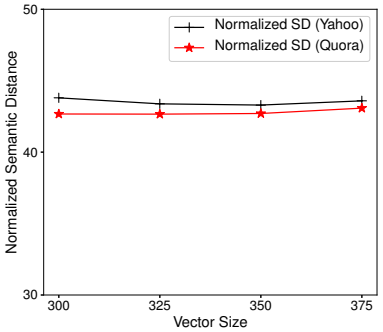

Fig. 9. The impact of the vector size on the attack results in VNG++.

this parameter will affect results in semantic distance. We can see from Fig. 6 and Fig. 7, the normalized semantic distance in LL-NB and VNG++ hardly change as we increase the number of epochs in the training process of the doc2vec model. This indicates that choosing the number of epochs as 100 is sufficient in our experiments.

Impacts of Vector Size. Similarly, we also study the impact of the size of semantic vectors on the results under different attack algorithms. As shown in Fig. 8 and Fig. 9, we find that the size of a semantic vector produced by a doc2vec model does not significantly affect the normalized semantic distance.

E. Privacy Leakage Mitigation with BuFLO

We apply BuFLO and explore the corresponding privacy leakage of voice command fingerprinting attacks. Our results suggest that BuFLO could enhance privacy of smart home speakers but would require significant amounts of tradeoffs in terms of communication overhead and time delay.

As shown in Table II, if we obfuscate traffic traces by padding each packet size to $d = 1000$ bytes, changing the frequency of packet size to $\rho = 50$ per second, and adjusting the minimum amount of communication time to $\tau = 20$ seconds, less information will be leaked under voice command fingerprinting attacks. For instance, the attack accuracy in LL-NB will drop to 5.6% and the normalized semantic distance will raise to 47.71. Similarly, the attack performance of other attacks will be reduced if BuFLO is applied. We can see that P-SVM (AdaBoost), which still achieves 14.7% accuracy, outperforms other attacks if obfuscation is applied.

TABLE II
RESULTS OF VOICE COMMAND FINGERPRINTING ATTACKS ON OBFUSCATED DATASETS PRODUCED BY BUFLO

$d = 1000, \rho = 50, \tau = 20$	Accuracy	Normalized SD (Yahoo!)
LL-Jaccard	1%	48.51
LL-NB	5.6%	47.71
VNG++	9.9%	47.68
P-SVM (AdaBoost)	14.7%	42.66
Random Guess	1%	49.5

On the other hand, we also find that BuFLO would introduce significant tradeoffs in order to preserve privacy. For example, on average, each voice command will need to produce 932 KB (548%) additional communication overhead, causing 17.02 seconds (330%) in time delay. With a long time delay, BuFLO would significantly affect the user experience of smart home speakers.

Impacts of Fixed Packet Size. We explore the impact of fixed packet size in BuFLO on the attack results. In Fig. 10, it shows that the attack accuracy will slightly decrease if we introduce more dummy packets in the traffic. In addition, we find that the normalized semantic distance does not necessarily increase with the increase of fixed packet size.

On the other hand, we find the additional communication overhead increases linearly with an increase on the fixed packet size. However, when the fixed packet size reaches to the maximum value $1,500$, it will no longer increase linearly. This is because no packets will be divided into two obfuscated packets, and therefore the overall number of packets in the

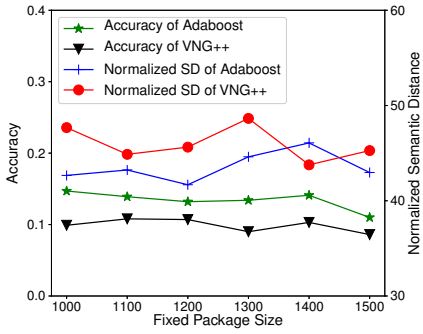

Fig. 10. The impact of fixed packet size (with $\rho = 50$ and $\tau = 20$) on the attack results in VNG++ and P-SVM (Adaboost).

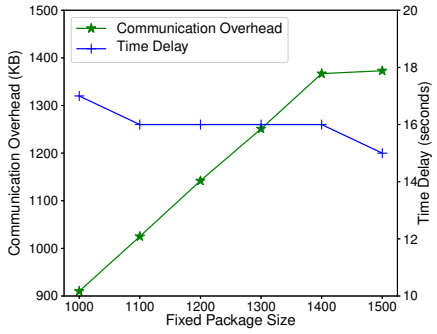

Fig. 11. The impact of BuFLO's fixed packet size (with $\rho = 50$ and $\tau = 20$) on the tradeoffs in time delay and communication overhead.

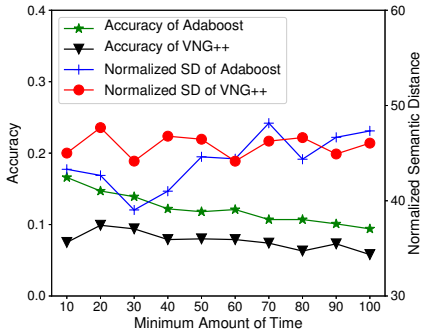

Fig. 12. The impact of minimum amount of time ($d = 1,000$ and $\rho = 50$) on the attack results in VNG++ and P-SVM (Adaboost).

obfuscated traffic is smaller. In terms of time delay, it slightly decreases as shown in Fig. 11.

Impacts of Minimum Amount of Time. We also study the impact of minimum amount of time on attack results. For instance, given fixed packet size is $1,000$, if we increase the minimum amount of time in BuFLO from 20 to 100, the attack accuracy will decrease. This suggests increasing the minimum amount of time will increase the privacy protection. However, both communication overhead and time delay will increase linearly with an increase of the minimum amount of time. We skip the corresponding figures due to space limitation.

V. Limitations and Future Works

While we are able to identify voice command fingerprinting attacks as new privacy risks with severe leakage, the research in this direction is still in its infancy and there are some notable limitations in our study. First, due to the limited resources, the scale of our traffic dataset is relatively small, which does not allow us to investigate more advanced attack algorithms, such as deep learning algorithms. According to the advantages of these algorithms in website fingerprinting [13], we expect these algorithms may reveal more information of voice commands with a higher accuracy or a closer distance. We will leave it as a future work.

Second, we select and customize four attack algorithms in website fingerprinting and apply them in voice command fingerprinting attacks. We do not enumerate all the attack algorithms in website fingerprinting. Therefore, we are not able to determine whether other existing attack algorithms can outperform the four selected algorithms or not in terms of inferring voice commands.

Third, while we try our best to select 100 common voice commands, the number of commands in a closed-world setting could be greater and the selection of voice commands could be different in a different study. A different list of voice commands may produce different accuracy and semantic distance. In addition, we do not consider an open-world setting in this study, which we will leave as a future work.

VI. Related Work

Website Fingerprinting Attacks. Many website fingerprinting attacks have been proposed in the current literature. Due to space limitation, we highlight several key attacks in this section. A more comprehensive survey of website fingerprinting attacks can be found in [24].

As mentioned in Sec. III, Liberatore and Levine [5] design a website fingerprinting attack by using Jaccard Similarity and Naive Bayes. The authors collected 480,000 samples for over 2,000 most-visited websites. The results show that both of the methods can achieve over 85% of accuracy.

Herrmann et al. [6] investigate a website fingerprinting attack on several different protocols, including OpenSSL, OpenVPN, and Tor networks. They examine the packet size and direction information. They implement the multinomial Naive Bayes classifier to infer the class of an unlabled traffic. Their results demonstrate that the proposed method can achieve over 90% accuracy on OpenSSH traffic but can only attain 3% accuracy in Tor networks with 775 websites.

As described in Sec. III, Panchenko et al. [7] select a set of features and leverages support vector machine as their classification algorithm. The attack can render over 55% of accuracy in Tor networks in a closed-world setting with 775 websites. In addition, this work also examines the attack in a open-world setting, where an unlabeled traffic may not be produced by a website in the training dataset. The devised method achieves over 73% false positive rate and 0.05% false negative rate in a open-world setting.

Dyer et al. [8] prove that previous countermeasures in [5], [6] are not effective if an attacker only analyzes coarse information. Specifically, using the Naive Bayes classifier, this study shows that their attacks based on coarse information can still achieve 61% accuracy on the traffic of Hermmann dataset obfuscated by previous countermeasures. The results of this work reaches a same level of accuracy as in [7] but it only requires coarse information. Moreover, the authors propose BuFLO, which can suppress an attacker's accuracy to 5% with over 400% bandwidth and 1.2 seconds in latency.

Wang et al. [10] design a website fingerprinting attack by applying a k-nearest neighbor classifier with weight adjustment. They consider features including total transmission, time, the number of incoming and outgoing packets, unique packet lengths, packet ordering, concentration of outgoing packets, and bursts. This study achieves 85% true positive rate and 0.6% false positive rate in an open-world setting.

Hayes et al. [11] propose a k-fingerprinting attack (k-FP). This attack leverages random decision forests to extract fingerprints. The authors utilize k-NN classifier and achieve 99% precision and 94% recall in an open world setting. The method can still render 96% precision and 68% recall if padding (e.g., WTF-PAD) is applied. Overdorf et al. [12] investigate the *fingerprintability* of Tor websites and analyze which features are vulnerable under fingerprinting attacks. The results show that the size-based features contribute most to the website identification and small sites are harder to identify.

Sirinam et al. [13] recently propose a website fingerprinting attack by leveraging Convolutional Neural Networks. This attack leverages order and direction. The attack achieves over 98% accuracy for 95 websites in a closed-world setting, and supplies 0.99 precision and 0.94 recall in an open-world setting. Schuster et al. [14] design an effective fingerprinting attack, which can identify which YouTube video a user watches by utilizing bursts in encrypted video streams. This attack leverages Convolutional Neural Networks, and achieves zero false positives with 0.988 recall.

Attacks on Smart Home Speakers. Apthorpe et al. [25] examine the encrypted traffic of IoT devices and suggest that people's activities at home can be easily inferred based on traffic generated by different IoT devices. Zhang et al. [3] design a Dolphin Attack, which can inject inaudible voice commands and maliciously activate smart home speakers without being noticed. Roy et al. [26] devise long-range attacks, which can launch inaudible voice commands 25ft away from smart home speakers. Kumar et al. [27] implement skill squatting attacks, which can leverages similar pronounced words to direct a user to malicious applications of smart home speakers. Another study of squatting attacks can be found in [28].

VII. Conclusion

We study voice command fingerprinting attacks on smart home speakers. We leverage semantic distance to complement the privacy assessment. Our findings suggest that this type of fingerprinting attack is severe and could compromise the privacy of millions of U.S. consumers. More importantly, efficient and effective countermeasures need to be developed in the future to mitigate privacy leakage without significantly affecting user experience of smart home speakers.

References

[1] "Amazon, google battle for dominance in smart home market." [Online]. Available: https://www.investors.com/stock-lists/tech-leaders/amazon-echo-google-home-smart-home/

[2] "An Oregon family's encounter with Amazon Alexa exposes the privacy problem of smart home devices." [Online]. Available: https://qz.com/1288743/amazon-alexa-echo-spying-on-users-raises-a-data-privacy-problem/

[3] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "DolphinAttack: Inaudible Voice Commands," in *Proc. of ACM CCS'17*, 2017.

[4] X. Fu, B. Graham, R. Bettati, W. Zhao, and D. Xuan, "Analytical and Emprical Analysis of Contermeasures to Traffic analysis Attacks," in *Proc. of ICPP'03*, 2003.

[5] M. Liberatore and B. N. Levine, "Inferring the Source of Encrypted HTTP Connections," in *Proc. of ACM CCS'06*, 2006.

[6] D. Hermann, R. Wendolsky, and H. Federrath, "Website Fingertinging: Attacking Popular Privacy Enhancing Tehnologies with the Multinomial Naive-Bayes Classifier," in *Proc. of ACM Workshop on Cloud Computing Security*, 2009.

[7] A. Panchenko, L. Niessen, and A. Zinnen, "Website Fingerprinting in Onion Routing Based Anonymization Networks," in *Proc. of Workshop on Privacy in the Electronic Society*, 2011.

[8] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, "Peek-a-Boo, I still See You: Why Efficient Traffic Analysis Countermeasures Fail," in *Proc. of IEEE S&P'12*, 2012.

[9] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proc. of ACM CCS'12*, 2012.

[10] T. Wang, X. Cui, R. Nithyannand, R. Johnson, and I. Goldberg, "Effective Attacks on Proable Denfenses for Website Fingerprinting," in *Proc. of 23rd USENIX Security Symposium*, 2014.

[11] J. Hayes and G. Danezis, "k-fingerprinting: A robust scalable website fingerprinting technique," in *USENIX Security Symposium*, 2016.

[12] R. Overdorf, M. Juarez, G. Acar, R. Greenstadt, and C. Diaz, "How unique is your .onion?: An analysis of the fingerprintability of tor onion services," in *Proc. of ACM CCS'17*, 2017.

[13] P. Sirinam, M. Imani, M. Juarez, and M. Wright, "Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning," in *Proc. of ACM CCS'18*, 2018.

[14] R. Schuster, V. Shmatikov, and E. Tromer, "Beauty and the Burst: Remote Identification of Encrypted Video Streams," in *Proc. of 26th USENIX Security Symposium*, 2017.

[15] "Voice Command Fingerprinting Attack and Dataset." [Online]. Available: https://github.com/SmartHomePrivacyProject/VCFingerprinting

[16] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. of the 31st International Conference on Machine Learning*, 2014.

[17] S. Li, H. Guo, and N. Hopper, "Measuring Information Leakage in Website Fingerprinting Attacks and Defenses," in *Proc. of ACM CCS'18*, 2018.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," https://arxiv.org/abs/1301.3781.

[19] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Staticstical Learning (With applications in R)*. Springer, 2013.

[20] A. Sciuto, A. Saini, J. Forlizzi, and J. I. Hong, ""Hey Alexa, What's Up?": Studies of In-Home Conversational Agent Usage," in *Proc. of the 2018 Designing Interactive Systems Conference*, 2018.

[21] "Gensim: topic modelling for humans." [Online]. Available: https://radimrehurek.com/gensim/models/word2vec.html

[22] "Yahoo! answers manner questions, version 2.0." [Online]. Available: https://webscope.sandbox.yahoo.com/catalog.php?datatype=l

[23] "First Quora Dataset Release: Question Pairs." [Online]. Available: https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

[24] E. Erdin, C. Zachor, and M. H. Gunes, "How to find hidden users: A survey of attacks on anonymity networks," *IEEE Communications Surveys Tutorials*, vol. 17, pp. 2296–2316, 2015.

[25] N. Apthorpe, D. Reisman, S. Sundaresan, A. Narayannan, and N. Feamster, "Spying on the Smart Home: Privacy Attacks and Defenses on Encrypted IoT Traffic," https://arxiv.org/abs/1708.05044.

[26] N. Roy, S. Shen, H. Hassanieh, and R. R. Choudhury, "Inaudible Voice Commands: The Long-Range Attack and Defense," in *Proc. of 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI'18)*, 2018.

[27] D. Kumar, R. Paccagnella, P. Murley, E. Hennenfent, J. Mason, A. Bates, and M. Bailey, "Skill Squatting Attacks on amazon Alexa," in *Proc. of 27th USENIX Security Symposium*, 2018.

[28] N. Zhang, X. Mi, X. Feng, X. Wang, Y. Tian, and F. Qian, "Understanding and Mitigating the Security Risks of Voice-Controlled Third-Party Skills on Amazon Alexa and Google Home," https://arxiv.org/pdf/1805.01525.pdf.