VoiceAttack: Fingerprinting Voice Command on VPN protected Smart Home Speakers

Xiaoguang Guo, Keyang Yu, Qi Li, Dong Chen
Department of Computer Science
Colorado School of Mines
Golden, Colorado, USA
{xiaoguang_guo,yukeyang,liqi,dongchen}@mines.edu

ABSTRACT

Recently, there are growing security and privacy concerns regarding smart voice speakers, such as Amazon Alexa and Google Home. Extensive prior research has shown that it is surprisingly easy to infer Amazon Alexa voice commands over their network traffic data. To prevent these traffic analytics (TA)-based inference attacks, smart home owners are considering deploying virtual private networks (VPNs) to safeguard their smart speakers. In this work, we design a new machine learning (ML) and deep learning (DL)-powered attack framework—VoiceAttack that could still accurately fingerprint voice commands on VPN-encrypted voice speaker network traffic. We evaluate VoiceAttack under 5 different real-world settings using Amazon Alexa and Google Home. Our results show that VoiceAttack could correctly infer voice command sentences with a Matthews Correlation Coefficient (MCC) of 0.68 in a closed-world setting and infer voice command categories with an MCC of 0.84 in an open-world setting by eavesdropping VPN-encrypted network traffic rates. This presents a significant risk to user privacy and security, as it suggests that external on-path attackers could still potentially intercept and decipher users' voice commands despite the VPN encryption. We then further examine the sensitivity of voice speaker commands to VoiceAttack. We find that 134 voice speaker commands are highly vulnerable to VoiceAttack and 3 commands are less sensitive. We also present a proof-of-concept defense approach—VoiceDefense, which could effectively mitigate TA attack accuracy on Amazon Echo and Google Home by ~50 .

CCS CONCEPTS

• **Security and privacy → Privacy-preserving protocols**; *Privacy protections*; *Information flow*; **Vulnerability management**.

KEYWORDS

Voice Speakers, IoT Systems, User Privacy, ML Systems, Traffic Analysis, Smart Homes

ACM Reference Format:
Xiaoguang Guo, Keyang Yu, Qi Li, Dong Chen. 2024. VoiceAttack: Fingerprinting Voice Command on VPN-protected Smart Home Speakers. In *The 11th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys'24), November 7–8, 2024, Hangzhou, China.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3671127.3698171

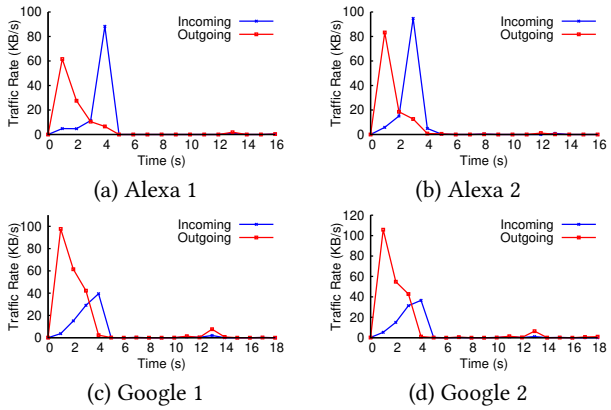
BuildSys'24, November 7–8, 2024, Hangzhou, China

ACM ISBN 979-8-4007-0706-3/24/11
https://doi.org/10.1145/3671127.3698171

1 INTRODUCTION

A smart speaker is a voice-activated Internet of Things (IoT) device, within which is a virtual smart assistant that can help people with everyday tasks. These tasks could include playing music, making appointments and controlling other IoT devices in the house. Another example, Amazon's voice speakers use an assistant called Alexa and when you give voice commands such as "what is the weather like tomorrow?", Alexa will answer the question with current location's weather information. Alexa also supports voice commands that require web-search responses and third-party enabled Alexa "skills" [2]. Most of the skills available on the Amazon Skill Store are developed by third-party developers to extend the capabilities of voice assistants. Smart voice speakers, such as Amazon Echo and Google Home, are increasingly deployed in smart homes for automation. The global volume in the smart voice speakers segment was forecast to continuously increase between 2024 and 2028 by in total 108.2 million pieces (~47.56) [39].

Extensive recent research [1, 10, 23, 28, 45] has shown that it is surprisingly easy to fingerprint simple voice commands using network traffic data of Amazon Alexa and Google Home. The most recent work [28] has presented that Amazon Alexa voice commands can be fingerprinted with ~93 accuracy, indicating a serious passive and local adversarial attack on smart speaker user privacy. Another work [1] has shown that they actually could achieve the accuracy of 70~80 when fingerprinting fine-grained user activities. However, these approaches often assume ideal smart home settings. For instance, recent approaches [10, 23, 28, 45] typically focused on local adversaries in their threat models when designing their systems. In the work [23], the authors assumed they have knowledge of the starting and ending times of each smart speaker traffic trace, as well as the source and destination IP addresses of each network traffic packet. In another work [45], the authors assumed local attackers know the model of a smart speaker, and also the IP addresses of the smart speaker and its remote voice servicing server. Although the most recent work [1] considers both local and external attack scenarios in their system design, the proposed approach depended on precise detection of voice invocation traffic flows, which can be challenging to achieve when a smart home has multiple IoT devices operating simultaneously. In real practices, attacks mentioned above proves challenging in real-world smart home environments, especially those protected by VPNs where only encrypted traffic is accessible. The precise timing and "pure"



(a) Alexa 1 (b) Alexa 2

(c) Google 1 (d) Google 2

Figure 1: Traffic volumes produced by Amazon Echo (in a and b) and Google Home (in c and d) for the same command "What's the weather today?"

patterns of voice speaker traffic necessary for designing such fingerprinting attacks are not readily obtainable under these conditions.

We consider a practical setting where home owners could deploy virtual private networks (VPNs) to safeguard their smart speakers. Thus, adversaries are not necessarily to be local and do not have access to the exact time when and what activities are active on smart speakers. Unlike prior works, we do not assume attackers know the source and destination IP addresses of smart speakers and their remote service providing servers. Figure 1 illustrates the VPN encrypted traffic rate traces of Amazon Alexa and Google Home, with each command repeated twice by users. *Our key insight is that user voice commands might exhibit distinct network traffic patterns and are already embedded in their (encrypted) outgoing/incoming network traffic data.* These voice commands can potentially be inferred or recovered through advanced traffic analysis (TA) techniques, such as machine learning (ML) or deep learning (DL)-powered fingerprinting attacks.

To this end, we are answering these two questions: (1) "*Can we design a new fingerprinting attack that could infer voice commands across different voice platforms via only VPN-protected smart voice speaker network traffic traces under read-world settings?*"; (2) "*Can we further design a new proof-of-concept approach that could potentially and effectively prevent this new fingerprint attack?*". To address these questions, we design a new ML/DL-powered attack framework—VoiceAttack that could still accurately fingerprint voice commands on VPN-encrypted voice speaker network traffic. In doing so, we are making the following contributions.

Challenges. We first explore and emphasize the key challenges in designing our new fingerprint attack—*VoiceAttack* that could accurately fingerprint voice commands via VPN-encrypted traffic.

System Design of VoiceAttack. We present the design of VoiceAttack, which could accurately fingerprint voice commands via VPN-encrypted voice speaker network traffic. VoiceAttack leverages a deep learning (DL) model to gather and preprocess VPN-encrypted voice speaker network traffic datasets for training. Next, VoiceAttack employs a novel smart speaker detection model and machine learning (ML)-enabled traffic filtering model to automatically distinguish between different smart speaker models based on their respective network traffic patterns. Following this, VoiceAttack implements an new LSTM-based fingerprinting attack with exceptional precision at both the sentence and category levels of voice commands. This attack accurately categorizes and identifies voice commands, revealing sensitive information about the user's voice interactions with their smart voice speakers.

Implementation and Evaluation. We implement VoiceAttack using the most widely used programming language—*Python*. To assess and benchmark VoiceAttack, we conduct evaluations under 5 different real smart home deployments. Our results show that VoiceAttack could correctly infer voice commands sentences with MCC as of 0.68 in closed-world setting and infer voice commands categories with an MCC of 0.84 in open-world setting by eavesdropping VPN-encrypted network traffic, respectively. This presents a significant risk to user privacy and security, as it suggests that external on-path attackers could still potentially intercept and decipher users' voice commands despite the VPN encryption. We then further examine the sensitivity of voice commands to VoiceAttack. We find that 134 commands highly vulnerable to VoiceAttack. We also present a proof-of-concept defense approach—VoiceDefense, which could effectively mitigate TA attack accuracy by ~ 50 .

Releasing Datasets. Our new approaches to benchmark TA attacks and their defenses for smart voice speakers are quite general, and could be applied to address similar user privacy issues of other IoT devices or in other domains. We released the datasets of VoiceAttack to IoT research communities on our lab GitHub page [42].

2 BACKGROUND AND RELATED WORK

2.1 Overview of Smart Voice Speaker System

Smart voice speakers (e.g., Amazon Echo, Google Home, Bose Smart Speaker, Denon Home Speaker, Harman Kardon Astra Speaker, Edifier Wi-Fi Speaker) typically have built-in voice assistants, such as Amazon Alexa or Google Home, providing user interactive features. The Alexa voice commands can be categorized into built-in commands and skills-enabled commands. Built-in commands are the voice commands that are directly processed by Amazon's Alexa Service. For instance, a user can ask Amazon Alexa the built-in command—"What is the weather today?". The Amazon Alexa will then transmit the audio command to the Amazon Alexa Servicing Server. The Alexa service contains various services, including Weather, Wikipedia, Time, and others. For a weather request, the service accesses the built-in weather information. There, the server prepares the request for "weather information" at the user's location and response to the user. Another example, the users may ask some "skills" related voice commands. When the user issues the command "Turn on the master bedroom light," Alexa forwards the audio command to their remote Alexa Service server. The server communicates with the Alexa Skills server, which functions as the marketplace for skills where third parties develop the majority of the available skills in the Amazon Skill Store. These skills are designed to enhance and expand the capabilities of the voice assistant. Eventually, the feedback control commands are transmitted via the device cloud to the light switch, instructing it to turn on the light.

Figure 2: Overview of our threat model.

2.2 Threat Model

As shown in Figure 2, we are broadly concerned with the ability of external adversaries (e.g., Internet service provider, on-path network observers, manufactures, cloud storage providers, and their third-parties) to fingerprint voice commands via VPN-encrypted network traffic traces of smart voice speakers. Note that, unlike the recent approaches [10, 23, 28, 45], we do not assume we only have local adversaries in their threat models when designing their fingerprinting systems. Instead, we are focusing on a more practical real-world setting, where adversaries are on the path from a smart home Internet router to voice speaker servicing servers. Thus, these external adversaries can only sniff or access to VPN-encrypted network traffic rates of smart voice speakers. Unlike prir works, the adversaries are not assumed to have the knowledge of "internal" metadata, such as IP addresses of the smart speaker and its remote voice servicing server, original source and destination IP addresses of each network packet, and starting and ending times of each smart speaker traffic trace.

We assume external adversaries can use any sophisticated TA inference techniques, such as ML, DL, or other statistical methods to fingerprint voice commands using the recorded voice assistant traffic rates. Thus, fingerprinting voice commands at both sentence and category levels in these smart homes is considered as an opposition to smart home users' privacy preferences. And these potential adversaries may be incentives to fingerprint user voice commands from smart speaker network traffic traces where users do not want to share this privacy-sensitive information with them.

We are concerned with three types of user privacy TA attacks: i) *Fingerprinting user voice command sentences.* This includes what voice commands a user has asked and when; ii) *Fingerprinting user voice command categories.* This includes what categories a user command belongs to and when. In particular, even if voice commands are inaccurately fingerprinted, adversaries could still deduce their categories, potentially revealing some sensitive user information; iii) *Fingerprinting user voice command keywords.* This includes what keywords or phrases that a voice command contains and when. This becomes particularly intriguing and potentially "useful" for external adversaries when sentence-level fingerprinting fails. In addition to the above-mentioned short-term user voice commands fingerprinting, adversaries may infer more comprehensive and longer-term user sensitive activities, such as whether a household has a baby, and whether they go on vacation on weekends.

Attack Scenario 1: *Fingerprinting user sentence-level voice commands.* To fingerprint user sentence-level voice commands, an external Internet on-path adversary intends to acquire the voice assistant network traffic data, and leverage ML- and DL-based traffic analysis (TA) attacking approaches to fingerprint the voice commands that are embedded in the traffic traces. This fingerprint attack could result in user sensitive information leakage.

Attack Scenario 2: *Fingerprinting user category-level and keyword-level voice commands.* In addition to the prior attack scenario, an external adversary could also use advanced ML or DL techniques to develop a fingerprinting attack that could infer the categories and keywords of use voice commands via VPN-encrypted smart speaker network traffic traces.

2.3 Related Work

Although there has been relatively less emphasis on fingerprinting voice commands from smart speakers, there is a recently growing interest [1, 10, 23, 28, 45] in this area. Meanwhile, there is a substantial body of work dedicated to enhancing the *fingerprinting* of website network traffic and IoT device activities. We classify them into three categories, including fingerprinting website traffic, fingerprinting IoT device, and fingerprinting voice commands.

Fingerprinting Website Network Traffic. There are significant works [12, 13, 16, 27, 31, 44, 54] in fingerprinting website traffic using statistical models or ML models. Researchers showed it is feasible to infer user website visiting history by analyzing the network traffic generated by user website accessing. However, voice command traffic contrasts notably with website visiting traffic. Voice command data tends to be sparser and of shorter duration. Unlike browsing websites, where data flow is more continuous, voice commands of smart speakers consist of brief bursts of traffic flows.

Fingerprinting IoT Device Activities. People have also been developing new approaches to identify IoT device activities using their network traffic traces [4, 15, 19, 36, 40, 43, 49]. In the work [4] and [15], the authors proposed an approach that leveraged ML models such as feed-forward neural network (FFNN), long short-term memory (LSTM), and support vector machine (SVM) techniques to recognize users' activities of daily living (ADL). In the work [19], Hafeez et al. presented a semi-supervised learning to distinguish between malicious and benign device behaviors using network traffic generated by IoT devices. Shahid et al. [36] designed a Random Forest classifier based approach to recognize the type of IoT devices by analyzing their outgoing and incoming traffic packages. Wan et al. presented a model to profile diverse user activities with distinct IoT device event sequences, which are extracted from smart home network traffic based on their TCP/IP data packet signatures [43]. Xue et al. [49] proposed an approach to infer user activity patterns from a sequence of device events by deterministically extracting a small number of representative user activity patterns from the sequence of device events. Most of these approaches either rely on access to detailed network traffic header metadata or depend on the sequential IP packets and intervals to fingerprint user activities. However, these approaches cannot be simply applied to our research problem here, as such "perfect" data are not available in VPN-encrypted voice speaker traffic rates.

Fingerprinting Voice Commands of Smart Speakers. Kennedy et al. [23] introduced a passive voice command fingerprinting attack on smart home speakers. Their work demonstrated that a passive attacker, who can only eavesdrop encrypted traffic between a smart home speaker and a cloud server, can use ML models to infer users' voice commands and compromise the privacy of voice assistant users. Their experimental results on an Amazon Echo using a dataset of 100 common voice commands suggested that

	Platforms	Command Categories	Adversary Type	Targeted Traffic	Detection
Kennedy et al. [23]	Alexa, Google Assistant	Simple	Local	Filtered	Activity
Wang et al. [45]	Alexa, Google Assistant	Simple	Local	Filtered	Activity
Ahmed et al. [1]	Alexa, Google Assistant, Siri	Simple, Skills	Non-local	All, Aggregated	Invocation
Charyyev et al. [10]	Alexa	Simple	Local	Filtered	Activity
Mao et al. [28]	Alexa	Simple	Local	Filtered	Activity
Ours	Alexa, Google Assistant	Simple, Skills	External	All, Aggregated	Activity

Table 1: The comparison of recent smart speaker fingerprinting systems.

voice command fingerprinting attacks can correctly infer 33.8 of voice commands by eavesdropping on encrypted traffic in a closed world setting. Wang et al. [45] implemented proof-of-concept attacks by leveraging deep learning model in a open-world analysis. Their experimental results indicated disturbing privacy concerns on voice assistant network traffic data. Their attacks can correctly infer voice commands over encrypted traffic with 92.89 accuracy on Amazon Echo on a closed world setting. Charyyev et al. [10] proposed a method that utilizes the network traffic of the speakers to fingerprint the voice commands of users without a need for extracting traffic features with ML algorithms. Their approach correctly inferred 42 of voice commands while ML models infer 22 to 34 . Mao et al. [28] developed a new DL model using time-series related features (e.g., direction, time, and size of packets of smart speaker network traffic traces) to improve the attack accuracy on the same dataset to 93.36 . Ahmed et al. [1] presented a new taxonomy of fingerprinting voice commands on the voice assistants. They first detected the invocation/activation of voice assistants followed by what specific voice command is issued. Their results have shown that it is possible to detect when a voice assistant is activated with 99 accuracy and then utilize the subsequent traffic patterns to infer more fine-grained user activities with around 77∼80 accuracy.

Observation. As shown in Table 1, prior works [10, 23, 28, 45] mainly focused on local adversaries in their approaches. Specially, the approach in [23] assumed 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). They also assumed that the attackers know the source and destination of packets by observing packet headers. The work [45] further assumed the local eavesdropper can observe and use IP address of a smart speaker to filter out the "pure" smart speaker traffic traces. That being said, their research relies on the availability of IP addresses, which are only accessible to local adversaries, to separate traffic traces. Prior works [10, 23, 28, 45] leverage the activities or patterns that are embedded in the speaker traffic metadata and traces to fingerprint voice commands. In contrast, the most recent work [1] uses "invocation" traffic spikes. This notable work [1] relaxed the strict assumptions by initially detecting invocation commands (e.g.,"Hi Alexa"). They then trimmed the network traffic immediately following these invocation commands for fingerprinting. Prior works [10, 23, 28, 45] mainly considered the simple built-in voice commands and skill-enabled commands fingerprinting are not fully evaluated. Existing approaches have not been fully evaluated in VPN-encrypted voice speaker scenarios.

Insight. Although prir research has made significant efforts to explore the severity of fingerprinting voice commands, these approaches have not been considered or evaluated in practical environments where users deploy VPNs to protect their traffic. Additionally, they may not account for scenarios where users have multiple interleaving IoT devices online, leading to potential noise from other devices existing in smart speaker traffic. Specifically, invocation commands used by the most recent work [1] may become unreliable and inaccessible from VPN-encrypted voice speaker traffic. Prior works [10, 23, 28, 45] assumed the access to traffic metadata, which can only be sniffed by local adversaries, to build their fingerprinting approaches. In our real-world setting, we focus on external adversaries who only have access to network rates. *They do not have access to the detailed metadata assumed in prior works* [10, 23, 28, 45]. Unlike many prior works assume attacker are given the model of the targeted voice speaker, a new fingerprinting approach could automatically recognize the smart speaker model is necessary. These valuable insights will guide the design, implementation, and evaluation of our proposed fingerprinting system—*VoiceAttack*.

3 CHALLENGES

VPN-encrypted Speaker Traffic. We focus on a real-world setting where users are free to utilize encryption techniques, particularly VPNs, to conceal their voice speaker traffic data. External on-path adversaries can only observe the VPN-encrypted traffic rate volume traces of smart speakers. Thus, many existing approaches, which assume detailed access to traffic metadata, including voice speaker IP addresses, source and destination IP addresses or domains of traffic packets, model information, accurate invocation commands, etc., will not work in this scenario. To address this challenge, we extract a wide set of principal features that could distinguish different voice commands using the VPN-encrypted traffic rates.

Traffic "Noise" from Other Devices. Another challenge is the potential "noise" from other "interleaving" device traffic. In real-world settings, there may be sometimes multiple active devices using Internet traffic simultaneously. Another recent research [51] showed that among 72,370 IoT traffic spikes, 15.19 have significant aggregation due to the interleaved IoT device usages in 34-day traffic traces of four smart homes. This could bring some noise into voice speaker traffic rates. Prior works either did not consider this situation or leveraged internal IP addresses which are mostly inaccessible in real-world settings to filter out the "pure" voice command traffic traces. To overcome this challenge, we design a new ML-enabled traffic filtering algorithm to automatically detect voice speaker model and also obtain voice speaker command traffic.

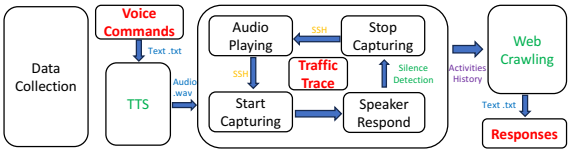


Figure 3: The operational pipeline of our data collection.

Missing Similarity and Sensitivity Analysis. Prior approaches have examined their efficiency using different metrics. One missing in understanding the fingerprint accuracy of voice commands is the analysis of similarity and sensitivity. To address this issue, we propose to comprehensively evaluate the performance of fingerprinting voice commands at both the sentence, category, and keyword levels. We also examine the effect on fingerprinting accuracy of semantic similarity or relatedness of different voice commands.

4 DATA COLLECTION

4.1 VPN-encrypted Speaker Traffic Collection

Figure 3 illustrates the design of our traffic data collection module.
Voice Command Dataset Preparation. To prepare the voice command datasets, we leverage the voice command categories listed on the Amazon Alexa official website [3]. We compile 10 voice commands for each category, resulting in the collection of 200 voice commands for each voice speaker platform. These datasets consist of basic and skills voice commands of Alexa and Google Home. These textual voice commands are converted into audio files using the Coqui TTS pre-trained text-to-speech conversion model [41]. The TTS synthesis is performed using the tacotron2-DCA model from Coqui AI, trained on the LJ Speech dataset.
Raw Traffic Trace Capturing. We first configure Raspberry Pi 4 Model B to operate as a VPN-enabled router running on OpenWRT. We then use another Raspberry Pi 4 Model B with a speaker to play all the collected auto files from the previous step. We employ *tcpdump* on the VPN router to collect both incoming and outgoing traffic generated by the invocation of voices speakers. We leverage an open-source library—SoundDevice [38] to detect silence level of voice speakers and thus proactively terminate a traffic trace recording process. This "silence" indicates that a complete interactive session between the voice speakers and their remote serving servers has concluded. To ensure the quality of our data collection, we repeat voice command 200 times.
Traffic Data Cleaning from Outliers. During our data collection process, we observe some outlier data points caused by various factors such as Internet connection failures, delayed responses, or instances where the audio command was not recognized correctly for voice assistants (e.g., "audio is not intended for Alexa."). To ensure only valid and relevant traffic data was collected into our datasets, we leverage the custom web crawler to extract the textual content of each response from the voice speaker activity history.

5 SYSTEM DESIGN

Figure 4 depicts the pipeline of our new voice command fingerprinting framework—*VoiceAttack*.

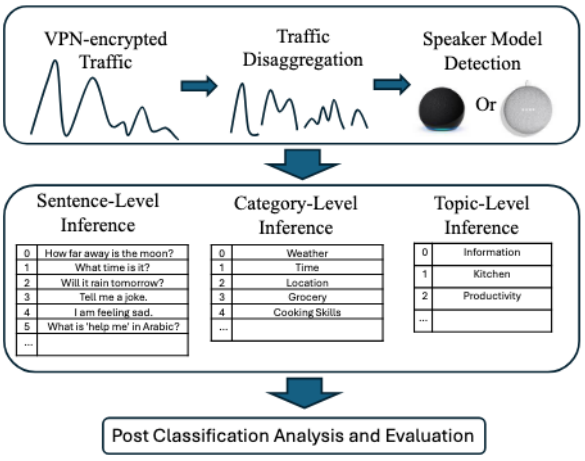


Figure 4: The operational pipeline of our VoiceAttack system.

5.1 Voice Speaker Network Traffic Detection

As discussed in Section 3, the first challenge is the potential noise from other interleaving device traffic. In real-world deployments, there could be some times multiple active devices using Internet traffic simultaneously. To address this issue, we first design a new ML/DL enabled traffic detection module to automatically obtain voice speaker command traffic and thus detect the voice speaker model (either Amazon Alexa or Google Home). There is not significant research in IoT VPN-encrypted traffic detection area. The most relevant ones are the recent work [25] and the netmeter energy data disaggregation works [7, 22, 24, 52, 53] in Non-intrusive Load Monitoring (NILM) community. The aim of network traffic detection is to provide accurate individual IoT device traffic consumption estimates based on the VPN-encrypted whole-house IoT network traffic consumption. We are highly motivated by these works and benchmark and tailor the major disaggregation algorithms in NILMTK framework [6]. Figure 5 shows the comparison results of different disaggregation algorithms. We find that LSTM-based approach yields the lowest Mean Absolute Percentage Error (MAPE), indicating the highest detection accuracy. Moreover, the MAPE exhibits only marginal changes as the number of noise devices increases from 1 to 25. Figure 6 illustrates the tailored LSTM traffic detection results when Amazon Alexa and other five IoT devices become online simultaneously. Thus, we select LSTM as our traffic detection approach for our *VoiceAttack* framework. *Note that the detection algorithms in our system design are orthogonal to new approaches and models.* Detecting the voice speaker model becomes trivial once we acquire the"pure" network traffic traces.

5.2 Voice Speaker Traffic Preprocessing

Unlike browsing websites or other IoT device traffic, where traffic flow is more continuous, voice commands of smart speakers consist of brief/sharp bursts of traffic flows. To capture the features embedded in these sharp and short traffic flows, we then preprocess our voice command network traffic traces. Our data preprocessing process is systematically divided into three stages, each designed to refine traffic data for later optimal feature extraction.

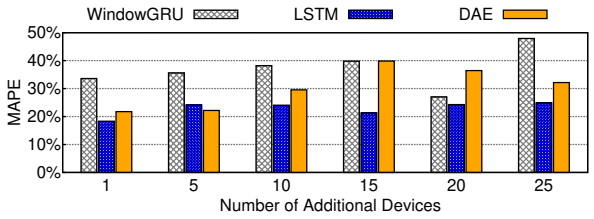


Figure 5: The performance of traffic detection methods.

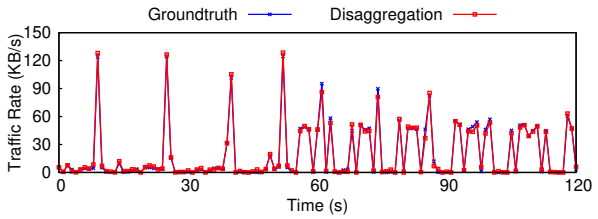


Figure 6: The visual comparison of traffic detection.

- **Trimming:** Initial segments of the traffic, typically the first 0.5 to 1 second, are primarily invocation commands and thus distill the data to the most informative segments for fingerprinting.
- **Resampling:** To reconstruct the intrinsic patterns embedded in the traffic traces more effectively, we resample the traffic data at intervals of 0.1, 0.2, 0.5, and 1 second. This process is critical for aligning temporal variations to a standard time-series framework.
- **Filtering:** The traffic traces undergo a filtering process. Traffic flows are bifurcated into incoming flows (from cloud servers to smart speakers) and outgoing flows (from smart speakers to cloud servers), thus segregating the data for focused examination.
- **Standardization:** The principal features are often recorded using different measurement units. To address this issue, we leverage standard scaling, min-max scaling, and Box-Cox transformation to further process our training dataset.

5.3 Principal Traffic Feature Extraction

We then identify and extract the principal features from the clean voice speaker traffic. Firstly, we empirically identify the potential candidate principal features from the voice speaker traffic. Then, we employ Principal Component Analysis (PCA) to extract the principal features from these identified potential features. The multi-dimensional candidate features we learn from prior works may come from six different categories, including *Basic Statistics*, *Fourier Transform Results*, *Spectral Features*, *Energy and Power*, *Time Series Analysis*, and *Distribution Characteristics*. Our basic statistics include the mean, standard deviation, median, mean absolute deviation, skewness, kurtosis, and duration. Our Fourier transform features include FFT mean and FFT standard deviation. Regarding spectral features, we mainly consider spectral centroid, entropy, rolloff, flux, flatness, and kurtosis. We then apply energy and power-related metrics as our features, such as root mean square (RMS), signal magnitude area (SMA), total harmonic distortion (THD), signal-to-noise ratio (SNR), and signal energy. Since voice speaker traffic is described in time-series data format, we also use autocorrelation, crest factor, first difference mean, first difference variance, cumulative sum, smoothness, number of peaks, and waveform length to capture the relationship between time-series traffic and voice commands. Lastly, we also leverage distribution metrics, including entropy, entropy of energy, entropy package distribution (EPD), and max autocorrelation peak (MAP). Totally, we have empirically identified 31 features as the potential principal features.

Figure 7 illustrates the relationship between these five important traffic features. Some features exhibit high correlation with others. For instance, "sma" exhibits a linear relationship with "sd_dev",

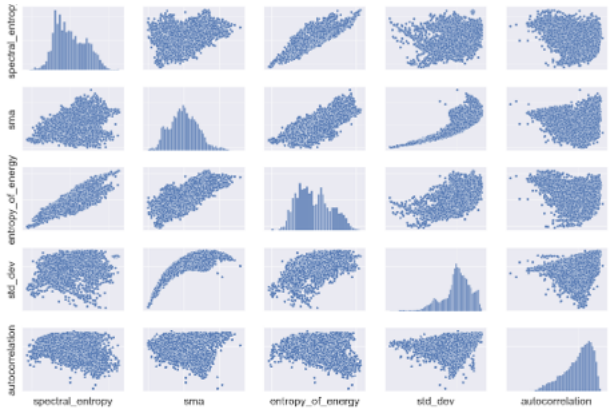


Figure 7: The correlation matrix of different traffic features.

while the "entropy of energy" and the "spectral entropy" demonstrate a strong linear correlation. Therefore, we cannot simply feed these features directly into our fingerprinting models, as they might lead to overfitting. To address this problem, we leverage principal component analysis (PCA) which simplifies the complexity in high-dimensional data while retaining trends and patterns. In addition, PCA provides dimensionality reduction that has been used to optimize the training time and solve part of the overfitting problem. We employ PCA to automatically learn the principle features for voice command fingerprinting. When applying PCA on our voice command datasets, the input is the whole dataset ignoring the binary class labels. PCA computes the covariance matrix, and eigenvectors and corresponding eigenvalues. Eventually, we transform our samples onto the new subspace. We find that starting from 25 components, the PCA's cumulative explained variance is close to 100 , and individual explained variance is approaching 0 . That said, our approach is able to reduce dimensions from 66 (combined incoming and outgoing) to 25 while preserving most of the feature information. We use the 25 principle components as the inputs for our fingerprinting models which are discussed next.

5.4 Command Fingerprinting Model Selection

Our next step is to build our new voice command fingerprinting model to infer the voice commands using the learned principal traffic feature set. We focus on selecting the optimal ML/DL classifier that could yield the best accuracy for voice command fingerprinting. We investigate the most widely used ML/DL models in prior fingerprinting relate works, including Long short-term memory (LSTM),

Multi-Layer Perceptron (MLP), Random Forest, Extra Trees, Logistical Regression, Support vector machines (SVMs), Extreme Gradient Boosting (XGBoot), Decision Tree, k-nearest neighbors (KNN), and Naive Bayes. In particular, we also benchmark the different kernels for SVMs, including linear, linear passive-aggressive, linear ridge, polynomial with 1~10 degrees, and radial basis function (RBF). In addition, we also fine-tune the tree-based models (e.g., Random Forest, Decision Tree, Extra Tree and XGBoost) to ensure we are reporting the best performance for those ML approaches.

Fingerprinting voice command at sentence-level. The results in Figure 8 show that the LSTM classifier yields the best MCC as ~0.7 when fingerprinting voice command at sentence level, which is two times better than the Naive Bayes classifier. Unsurprisingly, LSTM reports the best accuracy. LSTM has been very successful in the literature, especially when working with sequences of words and paragraph datasets, particularly those represented as time series data format. MLP reports decent but suboptimal results and is often used as a baseline point of comparison to benchmark other models that may appear better suited for the research problem. Note that, although other ML models yield lower accuracy, their performance on fingerprinting user voice commands at the sentence level is still better than prior work [23], where their inference accuracy is 33.8 on non-VPN-encrypted voice speaker traffic. This is mainly due to the more significant and efficient extraction process of our principal features. The results have shown that user voice commands are embedded in their network traffic rate traces and could be revealed by a well-performing traffic inference fingerprint attack. Figure 9 illustrates the detailed structure of our LSTM classifier.

Fingerprinting voice command at category-level and topic-level. Furthermore, our framework also supports more precise voice command fingerprinting, such as category-level and keyword-level. The insight lies in the fact that while we may not always be able to perfectly fingerprint voice command sentences, we can still deduce their categories and topics. This is because the semantic similarity between the groundtruth and predicted commands could be significant. To do this, we train a LSTM model using the Amazon recommended 20 categories. We also train another LSTM-powered classifier to fingerprint voice command topics.

5.5 Command Similarity & Sensitivity Analysis

Prior approaches have examined their results using different metrics and one big missing in understanding the fingerprint accuracy of voice commands is the analysis of their similarity and sensitivity. To this end, we are answering the question: "Why are we able to effectively fingerprint some voice commands while struggling with others?". We look into this question using semantic relatedness/similarity algorithms, including the Wu & Palmer measure (WUP) [48], the Resnik similarity (RES) [34], and the Lin measure (LIN) [26]. We can analyze the correlation between these relatedness metrics and the prediction outcomes of our fingerprint attacks to assess the attack results. To reflect this insight on our voice command fingerprinting accuracy, we also report Top-k voice command inference results. Top-k accuracy classification score aims at computing the number of times where the correct "label" is among the top K labels predicted (ranked by predicted scores) by a classifier.

5.6 Full Stack Benchmarking and Evaluation

Different than prior works, VoiceAttack also provides a full stack benchmarking on system performance using Matthews Correlation Coefficient (MCC) [29], F1 score [18], and Mean Absolute Percentage Error (MAPE), and also comprehensive similarity and sensitivity analysis using semantic relatedness/similarity algorithms, including Wu & Palmer measure (WUP) [48], Resnik similarity (RES) [34], and Lin measure (LIN) [26]. VoiceAttack could also assess fingerprinting systems using their training time and resource overhead.

6 IMPLEMENTATION

We implement *VoiceAttack* using the most widely used programming language—*Python*. To capture VPN-encrypted voice speaker traffic traces for our model training, we build a prototype by using two Raspberry Pi 4 Model B units, Amazon Echo Dot, Google Home, and a USB speaker. We flash one Raspberry Pi to playback audio input files and it acts as the "users" to initialize the conversations with the two voice speakers. We flash the second Raspberry Pi with OpenWrt and configure it as the host device for capturing traffic data. In addition, we deploy CyberGhost VPN on this host server. We employ `paramiko` to establish secure SSH communications and `sounddevice` to monitor audio outputs for initiating and concluding network traffic captures based on audio activity detected from smart speakers. We then further preprocess the captured traffic traces (represented in `pcap` format) by filtering, trimming, and resampling. We implement the system components of *VoiceAttack* using widely available open-source frameworks, including NILMTK, Scikit-learn, and TensorFlow. We leverage standard scaling, min-max scaling, and Box-Cox transformation to preprocess our datasets. We first use TensorFlow to implement MLP and LSTM. We then use Scikit-learn library to implement all other fingerprint ML and DL models, including Random Forest, Extra Trees, Logistical Regression, SVMs, XGBoot, Decision Tree, KNN, and Naive Bayes.

7 EXPERIMENTAL EVALUATION

7.1 Evaluation Scenarios

Scenario 1: Amazon Alexa Voice Command Traffic. This dataset comprises traffic traces captured from interactions with an Amazon Echo device, encapsulated in .pcap files. It has 200 unique voice commands categorized into 20 distinct classes, with each class containing 10 specific commands. To ensure balanced data collection, each command was executed 200 times, resulting in sufficient traffic data for model training. This dataset is structured to facilitate both sentence-level and category-level fingerprinting.

Scenario 2: Google Home Voice Command Traffic. An identical environment for collecting Google Home data was configured in another smart home environment. This dataset comprises network traffic traces captured from interactions with a Google Home voice assistant, encapsulated in .pcap files as well. The voice commands and category information remains identical to the Amazon Alexa dataset to maintain consistency in our experiments.

Scenario 3 ~ 5: Voice Command Traffic with additional IoT Devices. To demonstrate the effectiveness of our fingerprinting methods of VoiceAttack, we setup 25 additional IoT devices, including Belkin Wemo Switch, Sengled Smartbulb, Xiaomi Smart Camera,

Figure 8: The comparison results of different voice command fingerprinting models.

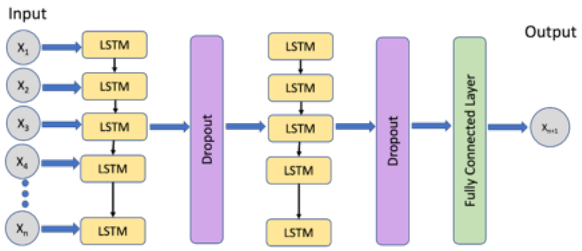


Figure 9: The overview of our LSTM architecture design.

Inference Type	Sentence-Level			Category-Level		
Traffic Flow	In	Out	I+O	In	Out	I+O
Amazon Alexa	0.5431	0.4928	0.6844	0.6247	0.6065	0.7280
Google Home	0.4814	0.4564	0.5702	0.5697	0.5397	0.6232
Alexa+5 Dev	0.5145	0.4372	0.5870	0.6130	0.5603	0.6782
Google+5 Dev	0.3756	0.3925	0.4424	0.4699	0.4988	0.5410
Alexa+15 Dev	0.5155	0.4518	0.5883	0.6061	0.5557	0.6867
Google+15 Dev	0.3733	0.4083	0.4529	0.4840	0.4981	0.5549
Alexa+25 Dev	0.5293	0.4461	0.6071	0.6208	0.5785	0.7034
Google+25 Dev	0.3780	0.3770	0.4735	0.4864	0.4813	0.5670

Table 2: The VoiceAttack's fingerprinting performance.

two Samsung SmartThings Hubs, two Philips Hue Bridges, Raspberry Pi 3, Eufycam, Dropcam, two Netgear Routers, two TP-Link Smart Plugs, and Netatmo Welcome Camera, Triby Speaker, Withings Smart Scale, Withings Baby Monitor, Withings Aura Smart Sleep Sensor, Sengled Smart Lightbulb, Belkin Wemo Motion Sensor, PIX-STAR Photo-frame, HP Printer, and Netatmo Weather Station. This dataset is then subjected to traffic disaggregation, followed by sentence-level and category-level fingerprinting.

Ethical Consideration for Data Collection. We collected network traffic data using DL models to explore the severity and extent of user privacy threat from smart speakers. Our long-term goal is to provide system solutions to enable people to regain the control of privacy leakages through their voice speaker data. We did not collect data from real individuals. We removed device identical information and sampled the datasets. *We followed our institution's Institutional Review Board (IRB) exempt process.*

7.2 Evaluation Metrics

Mean Absolute Percentage Error (MAPE). To quantify the disaggregation accuracy of VoiceAttack, we compute the MAPE [17], between the ground truth individual IoT device (e.g., Amazon Alexa) traffic consumption and the VoiceAttack infers over all time intervals t. A lower MAPE indicates higher accuracy with a 0 MAPE being perfectly aggregated traffic disaggregation.

$$M \quad PE \quad \frac{100}{n} \sum_{t \ 0} \left| \frac{S_t \quad P_t}{S_t} \right| \tag{1}$$

where n describes the duration of traffic disaggregation, S_t denotes the per-device groundtruth traffic consumption, and P_t indicates the predicted per-device traffic rate at time t.

Matthews Correlation Coefficient (MCC). We use the MCC [29], a standard measure of a classifier's performance, where values are in the range 1.0 to 1.0, with 1.0 being perfect voice command detection, 0.0 being random voice command detection, and 1.0 indicating voice command inference is always wrong. The expression for computing MCC is below.

$$\frac{TP * TN \quad FP * FN}{(TP + FP)(TP + FN)(TN + FP)(TN + FN)} \tag{2}$$

7.3 Experimental Results

7.3.1 Quantifying VoiceAttack's fingerprinting performance using different traffic flows. First, we evaluate the fingerprinting accuracy of VoiceAttack using different traffic flows under 5 different real-world settings, including incoming traffic, outgoing traffic, and a combination of both, with and without additional IoT devices. Table 2 describes the fingerprinting accuracy when applying VoiceAttack on incoming traffic, outgoing traffic and the combination of the two, respectively. Unsurprisingly, VoiceAttack yields the best MCC when training on both incoming and outgoing traffic flows for sentence-level voice command fingerprinting. In addition, VoiceAttack achieves better accuracy when trained on incoming traffic compared to training on outgoing traffic. We also find that VoiceAttack yields the MCCs of 0.50 and 0.58 when fingerprinting disaggregated Amazon Alexa traffic, which is the same as when fingerprinting "pure" Amazon Alexa traffic. In addition, VoiceAttack yields similar MCCs on disaggregated traffic for both Amazon Alexa and Google Home, regardless of whether there are 5, 15, or 25 additional IoT devices. This is mainly due to the fact that incoming traffic is less affected by "user" interactions, as it is prepared by the voice speaker cloud server. Therefore, incoming traffic is more reliable and easier for us to fingerprint compared to outgoing traffic. Similarly, we observe that VoiceAttack illustrates the similar "trends" when fingerprinting voice command at category-level.

Results: *VoiceAttack yields the best MCC when training on both incoming and outgoing traffic to fingerprint voice command at both*

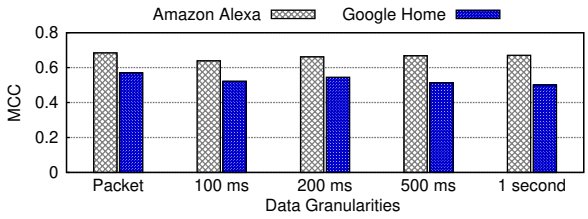


Figure 10: The comparison of VoiceAttack's accuracy on different traffic granularities.

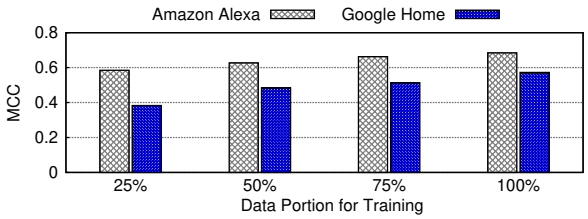


Figure 11: The comparison results when VoiceAttack training on different data sizes.

sentence and category levels. VoiceAttack achieves similarly good accuracy when fingerprinting the disaggregated voice speaker traffic and "pure" voice speaker traffic. VoiceAttack also achieves better accuracy when trained using incoming traffic compared to outgoing traffic.

7.3.2 Quantifying VoiceAttack's fingerprinting accuracy on different granularities of voice speaker traces. We next evaluate voice command fingerprinting effect on different traffic rates that have different level of granularities, such as 1 package, 0.1 second, 0.2 second, 0.5 second, and 1 second. Note that a voice command traffic may only last very few seconds in real practices. As shown in Figure 10, as expected, higher granularity results in lower fingerprinting accuracy in MCC. VoiceAttack's accuracy only drops marginally on both Amazon Alexa and Google Home. This is mainly due to the facts that 1) VoiceAttack performs consistently well on voice speaker traffic traces at different granularities, 2) fewer fluctuations and motifs can be observed in higher resolution traffic rate traces. When voice speaker traffic rates are becoming coarser, some principal features, such as standard deviation, variation coefficient and AUC, become less distinguishable and are thus hidden.
Results: *VoiceAttack performs consistently well on smart speaker traffic at different granularities. Ranging from 0.1 second to 1 second, VoiceAttack's accuracy in MCC only drops marginally from 0.68 to 0.67 on Amazon Alexa and 0.57 to 0.5 on Google Home, respectively.*

7.3.3 Quantifying VoiceAttack's fingerprinting performance using different traffic data sizes. Next, we evaluate VoiceAttack's fingerprinting performance when varying voice speaker traffic training dataset sizes. Figure 11 illustrates the comparison results of VoiceAttack's accuracy in MCC when training on different sizes of voice speaker traffic. As shown in Figure 11, VoiceAttack's accuracy is a linear function of training data sizes. For Amazon Alexa, VoiceAttack's accuracy in MCC increases from 0.58 to 0.68. Similarly, VoiceAttack's fingerprinting accuracy on Google Home traffic traces grows from 0.38 to 0.57. This is mainly due to the fact that when VoiceAttack has access to a larger training data size, we can extract more principal features and thus train a more accurate fingerprinting classifier. Hence, VoiceAttack demonstrates greater potential when training on larger datasets is feasible.
Results: *VoiceAttack's accuracy is a linear function of training data sizes. Even with a training data size of only 25 , VoiceAttack still achieves an MCC of 0.58 and 0.38 on voice speaker traffic traces from Amazon Alexa and Google Home, respectively.*

7.3.4 Quantifying VoiceAttack's performance on an open world setting. We also consider an open-world setting to better understand the severity of our new VoiceAttack in more real practices. We curated an open-world voice command dataset comprising voice commands observed during VoiceAttack's training phase and those never encountered before in its training dataset. We collect fingerprinting accuracy under different scenarios. For the voice commands that VoiceAttack encountered during its training process, it achieves a MCC accuracy of 0.82 when fingerprinting at sentence-level, indicating consistency with closed-world setting evaluations. For the voice commands that VoiceAttack never encountered during its training process, VoiceAttack achieves a MCC accuracy of 0.27 when fingerprinting at sentence-level. Interestingly, VoiceAttack yields the MCCs as of 0.84 and 0.93 at category-level and topic-level. This is primarily attributed to the fact that while VoiceAttack may not have encountered the entire voice command sentence before, there may be some overlapping category and topic information of commands included its training dataset. We also find that using only incoming traffic for fingerprinting, VoiceAttack still could yield the MCCs as of 0.20 and 0.86 at sentence-level and category-level, respectively. This further suggests that, despite variations in accents or language styles among users in real-world settings, VoiceAttack's fingerprinting performance is not significantly affected.
Results: *VoiceAttack could yield the fingerprinting accuracy in MCC as of 0.27 and 0.84 at sentence-level and category-level in open-world setting, respectively. VoiceAttack's results demonstrate potential robustness to the variability of outgoing traffic flows of voice commands.*

7.3.5 Voice Command Similarity and Sensitivity Analysis. Next, we will look into voice command similarity and sensitivity. For instance, for the voice command—"What's the weather going to be like this weekend?", VoiceAttack yields an accuracy of 71.05 and Top-5 accuracy of 94.74 when performing sentence-level fingerprinting. We examine the Top-5 most frequently selected predictions. The results revealed that the most frequent voice commands are predominantly related to meteorological inquiries. Specifically,"Will there be a thunderstorm this week?" emerged as the leading prediction, constituting 18.95 of the Top-5 selections. This was closely followed by "What's the weather like in San Francisco?" and "Is it going to be sunny tomorrow?", each accounting for 17.36 of the selections. Additionally,"What's the wind speed right now?" was represented with a frequency of 6.31 . These findings underscore a significant inclination towards weather-related queries within the voice command system. Meanwhile, we compute semantic similarities between voice commands using three different semantic similarity metrics: Wu-Palmer (WUP), Resnik (RES), and Lin (LIN). Table 3 illustrates the semantic similarity results for the different predictions. The WUP scores here range from 0.324 to 0.392, indicating a moderate level of similarity. RES scores range from 1.25

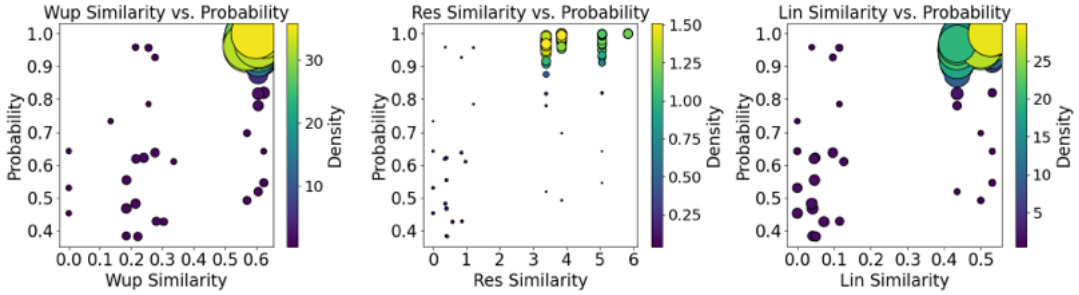


Figure 12: The correlation between semantic similarity metrics and prediction probability.

to 2.49, indicating potential overlapping between the predictions and the groundtruth. The LIN score shows a range similar to WUP, from 0.066 to 0.278, reflecting the detailed semantic overlap. These together suggest that while the commands share common terms and structures related to weather, their specific focus differs (e.g., general weather vs. specific phenomena like thunderstorms).

We also examine the correlation between semantic similarity and prediction probability. Figure 12 illustrates the relationship. We find that the three different metrics show a similar trend, indicating the correlation between semantic similarity metrics and prediction probability. When the metrics are closer to 1, VoiceAttack will yield a very high detection probability. Conversely, when the metrics are closer to 0, VoiceAttack has a lower detection probability.

Results: *The voice command fingerprinting accuracy of VoiceAttack shows a strong correlation with semantic similarities. The stronger the correlation, the higher the fingerprinting accuracy.*

7.3.6 Quantifying the performance of countermeasure approach—VoiceDefense. Eventually, we look into the potential preventing techniques to mitigate VoiceAttack's fingerprinting attack. While there isn't a direct approach we can apply to address our problem, there exists a substantial body of work focused on prevention through traffic reshaping approaches [5, 8, 9, 11, 14, 20, 21, 30, 32, 33, 35, 37, 46, 47, 50, 51] to thwart privacy attacks on IoT traffic rate traces. Motivated by these approaches, we propose a concept of proof defense—VoiceDefense. The fundamental concept behind VoiceDefense is to inject random "noise" traffic when active outgoing traffic is detected. This action serves to confuse VoiceAttack's disaggregation module, effectively masking voice speaker outgoing traffic. Furthermore, we will also initiate invalid requests via the voice speaker, resulting in additional spikes in incoming traffic. Through this two-directional traffic "reshaping," we find that VoiceDefense can decrease VoiceAttack's fingerprinting accuracy in MCC from 0.68 to 0.31 and from 0.57 to 0.24 on Amazon Alexa and Google Home, respectively. Additionally with the introduction of noise from 25 additional devices, VoiceDefense can decrease voiceAttack's fingerprinting accuracy in MCC from 0.61 to 0.30 and 0.47 to 0.21 on Amazon Alexa and Google Home, respectively. Note that we defer the comprehensive implementation and evaluation of such an approach for voice assistants to our upcoming work.

Results: *We design a new concept of proof defense—VoiceDefense, it could decrease VoiceAttack's fingerprinting accuracy by ~50 .*

Voice Command Pair	WUP	RES	LIN
Weekend vs. Thunderstorm Week	0.344	1.444	0.066
Weekend vs. San Francisco	0.392	2.490	0.278
Weekend vs. Sunny Tomorrow	0.385	1.707	0.202
Weekend vs. Wind Speed Now	0.324	1.255	0.159
Thunderstorm vs. San Francisco	0.330	1.789	0.015
Thunderstorm vs. Sunny Tomorrow	0.275	0.822	0.091
Thunderstorm vs. Wind Speed Now	0.293	1.292	0.025
San Francisco vs. Sunny Tomorrow	0.228	0.547	0.057
San Francisco vs. Wind Speed Now	0.382	1.946	0.245
Sunny Tomorrow vs. Wind Speed Now	0.267	0.564	0.073

Table 3: The semantic similarities between voice commands.

8 CONCLUSION AND FUTURE WORK

We design a new voice command inference attack framework—VoiceAttack that could still accurately fingerprint voice commands via VPN-encrypted voice speaker network traffic. Our results have shown that VoiceAttack could accurately fingerprint voice commands at sentence-level in both closed-world setting and open-world setting by eavesdropping VPN-encrypted network traffic, respectively. VoiceAttack presents a significant risk to user privacy and security, as it demonstrates that external on-path attackers could still potentially intercept and decipher users' voice commands despite the VPN encryption. We find that 134 commands are highly vulnerable to VoiceAttack. We also present a proof-of-concept defense approach—VoiceDefense, which could effectively mitigate inference attack accuracy on Amazon Echo and Google Home.

We plan to collect additional voice speaker traffic traces using Amazon Alexa and Google Home, and also validate our findings on another popular voice assistant—Siri. We will also develop more comprehensive and cost-effective approaches to defend against VoiceAttack while maintaining reasonable traffic overhead.

Acknowledgment. We thank the anonymous reviewers for their suggestions for improving the paper. This research is supported by NSF grant CNS-2238701. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] Dilawer Ahmed, Aafaq Sabir, and Anupam Das. Spying through your voice assistants: realistic voice command fingerprinting. In *32nd USENIX Security*

Symposium (USENIX Security 23), pages 2419–2436, 2023.
[2] Voice Interaction Models. https://developer.amazon.com/en-US/docs/alexa/ask-overviews/voice-interaction-models.html.
[3] Alexa Skills. https://developer.amazon.com/en-US/docs/alexa/ask-overviews/list-of-skills.html.
[4] Fahed Alkhabbas, Sadi Alawadi, Romina Spalazzese, and Paul Davidsson. Activity recognition and user preference learning for automated configuration of iot environments. In *Proceedings of the 10th International Conference on the Internet of Things*, pages 1–8, 2020.
[5] Noah Apthorpe, Danny Yuxing Huang, Dillon Reisman, Arvind Narayanan, and Nick Feamster. Keeping the smart home private with smart (er) iot traffic shaping. *Proceedings on Privacy Enhancing Technologies*, 2019(3):128–148, 2019.
[6] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, and Mani Srivastava. Nilmtk: An open source toolkit for non-intrusive load monitoring. In *Proceedings of the 5th international conference on Future energy systems*, pages 265–276, 2014.
[7] Hafsa Bousbiat, Christoph Klemenjak, and Wilfried Elmenreich. Exploring time series imaging for load disaggregation. In *Proceedings of BuildSys*, 2020.
[8] Phuthipong Bovornkeeratiroj, Srinivasan Iyengar, Stephen Lee, David Irwin, and Prashant Shenoy. Repel: A utility-preserving privacy system for iot-based energy meters. In *2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 79–91. IEEE, 2020.
[9] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. A systematic approach to developing and evaluating website fingerprinting defenses. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 227–238. ACM, 2014.
[10] Batyr Charyyev and Mehmet Hadi Gunes. Voice command fingerprinting with locality sensitive hashes. In *Proceedings of the 2020 Joint Workshop on CPS&IoT Security and Privacy*, pages 87–92, 2020.
[11] Dong Chen, David Irwin, Prashant Shenoy, and Jeannie Albrecht. Combined heat and privacy: Preventing occupancy detection from smart meters. In *2014 IEEE International Conference on Pervasive Computing and Communications*, 2014.
[12] Weiqi Cui, Tao Chen, Christian Fields, Julianna Chen, Anthony Sierra, and Eric Chan-Tin. Revisiting assumptions for website fingerprinting attacks. In *Proceedings of the 2019 ACM asia conference on computer and communications security*.
[13] Weiqi Cui, Jiangmin Yu, Yanmin Gong, and Eric Chan-Tin. Realistic cover traffic to mitigate website fingerprinting attacks. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 1579–1584. IEEE, 2018.
[14] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In *2012 IEEE symposium on security and privacy*, pages 332–346. IEEE, 2012.
[15] Patricia Franco, Jose Manuel Martinez, Young-Chon Kim, and Mohamed A Ahmed. Iot based approach for load monitoring and activity recognition in smart homes. *ieee access*, 9:45325–45339, 2021.
[16] Xun Gong, Negar Kiyavash, and Nikita Borisov. Fingerprinting websites using remote traffic analysis. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 684–686, 2010.
[17] Paul Goodwin and Richard Lawton. On the asymmetry of the symmetric mape. *International journal of forecasting*, 15(4):405–408, 1999.
[18] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer, 2005.
[19] Ibbad Hafeez, Aaron Yi Ding, Markku Antikainen, and Sasu Tarkoma. Real-time iot device activity detection in edge networks. In *International Conference on Network and System Security*, pages 221–236. Springer, 2018.
[20] Md Kamrul Hasan, Husne Ara Rubaiyeat, Yong-Koo Lee, and Sungyoung Lee. A reconfigurable hmm for activity recognition. In *2008 10th International Conference on Advanced Communication Technology*, volume 1, pages 843–846. IEEE, 2008.
[21] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an efficient website fingerprinting defense. In *European Symposium on Research in Computer Security*, pages 27–46. Springer, 2016.
[22] Jack Kelly and William Knottenbelt. Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of BuildSys*, pages 55–64, 2015.
[23] Sean Kennedy, Haipeng Li, Chenggang Wang, Hao Liu, Boyang Wang, and Wenhai Sun. I can hear your alexa: Voice command fingerprinting on smart home speakers. In *2019 IEEE Conference on Communications and Network Security (CNS)*, pages 232–240. IEEE, 2019.
[24] Odysseas Krystalakos, Christoforos Nalmpantis, and Dimitris Vrakas. Sliding window approach for online energy disaggregation using artificial neural networks. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*.
[25] Qi Li, Keyang Yu, Dong Chen, Mo Sha, and Long Cheng. Trafficspy: Disaggregating vpn-encrypted iot network traffic for user privacy inference. In *2022 IEEE Conference on Communications and Network Security (CNS)*, pages 145–153, 2022.
[26] Dekang Lin et al. An information-theoretic definition of similarity. In *Icml*, volume 98, pages 296–304, 1998.
[27] Liming Lu, Ee-Chien Chang, and Mun Choon Chan. Website fingerprinting and identification using ordered feature sequences. In *Computer Security–ESORICS 2010: 15th European Symposium on Research in Computer Security, Athens, Greece, September 20-22, 2010. Proceedings 15*, pages 199–214. Springer, 2010.
[28] Jianghan Mao, Chenyu Wang, Yanhui Guo, Guoai Xu, Shoufeng Cao, Xuanwen Zhang, and Zixiang Bi. A novel model for voice command fingerprinting using deep learning. *Journal of Information Security and Applications*, 65:103085, 2022.
[29] Matthews Correlation Coefficient.
[30] Homin Park, Can Basaran, Taejoon Park, and Sang Hyuk Son. Energy-efficient privacy protection for smart home environments using behavioral semantics. *Sensors*, 14(9):16235–16257, 2014.
[31] Tobias Pulls and Rasmus Dahlberg. Website fingerprinting with website oracles. *Proceedings on Privacy Enhancing Technologies*, 2020.
[32] Vasanthan Raghavan, Greg Ver Steeg, Aram Galstyan, and Alexander G Tartakovsky. Coupled hidden markov models for user activity in social networks. In *2013 IEEE International Conference on Multimedia and Expo Workshops*, 2013.
[33] Vasanthan Raghavan, Greg Ver Steeg, Aram Galstyan, and Alexander G Tartakovsky. Modeling temporal activity patterns in dynamic social networks. *IEEE Transactions on Computational Social Systems*, 1(1):89–107, 2014.
[34] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*, 1995.
[35] Karsten Rothmeier, Nicolas Pflanzl, Joschka Hüllmann, and Mike Preuss. Prediction of player churn and disengagement based on user activity data of a freemium online strategy game. *IEEE Transactions on Games*, 2020.
[36] Mustafizur R Shahid, Gregory Blanc, Zonghua Zhang, and Hervé Debar. Iot devices recognition through network traffic analysis. In *2018 IEEE international conference on big data (big data)*, pages 5187–5192. IEEE, 2018.
[37] Vitaly Shmatikov and Ming-Hsiu Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In *European Symposium on Research in Computer Security*, pages 18–33. Springer, 2006.
[38] Sound Device. https://github.com/spatialaudio/python-sounddevice.
[39] Sales volume of the smart speakers industry worldwide 2018-2028. https://www.statista.com/forecasts/1367982/smart-speaker-market-volume-worldwide.
[40] Abdulhamit Subasi, Mariam Radhwan, Rabea Kurdi, and Kholoud Khateeb. Iot based mobile healthcare system for human activity recognition. In *2018 15th learning and technology conference (L&T)*, pages 29–34. IEEE, 2018.
[41] Tts is a Library for Advanced Text-to-Speech Generation. https://github.com/coqui-ai/TTS.
[42] VoiceAttack. https://github.com/cyber-physical-systems/VoiceAttack.
[43] Yinxin Wan, Kuai Xu, Feng Wang, and Guoliang Xue. Iotmosaic: Inferring user activities from iot network traffic in smart homes. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 370–379. IEEE, 2022.
[44] Chenggang Wang, Jimmy Dani, Xiang Li, Xiaodong Jia, and Boyang Wang. Adaptive fingerprinting: Website fingerprinting over few encrypted traffic. In *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*, pages 149–160, 2021.
[45] Chenggang Wang, Sean Kennedy, Haipeng Li, King Hudson, Gowtham Atluri, Xuetao Wei, Wenhai Sun, and Boyang Wang. Fingerprinting encrypted voice traffic on smart speakers with deep learning. In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020.
[46] Tao Wang and Ian Goldberg. Walkie-talkie: An efficient defense against passive website fingerprinting attacks. In *26th USENIX Security Symposium*, 2017.
[47] Wei Wang, Mehul Motani, and Vikram Srinivasan. Dependent link padding algorithms for low latency anonymity systems. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 323–332, 2008.
[48] Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. *arXiv preprint cmp-lg/9406033*, 1994.
[49] Guoliang Xue, Yinxin Wan, Xuanli Lin, Kuai Xu, and Feng Wang. An effective machine learning based algorithm for inferring user activities from iot device events. *IEEE Journal on Selected Areas in Communications*, 40(9):2733–2745, 2022.
[50] Keyang Yu and Dong Chen. Paros: The missing "puzzle" in smart home router operating systems. In *2023 32nd International Conference on Computer Communications and Networks (ICCCN)*, pages 1–10, 2023.
[51] Keyang Yu, Qi Li, Dong Chen, Mohammad Rahman, and Shiqiang Wang. Privacyguard: Enhancing smart home user privacy. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week 2021)*, pages 62–76, 2021.
[52] Chaoyun Zhang, Mingjun Zhong, Zongzuo Wang, Nigel Goddard, and Charles Sutton. Sequence-to-point learning with neural networks for non-intrusive load monitoring. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
[53] Mingjun Zhong, Nigel Goddard, and Charles Sutton. Signal aggregate constraints in additive factorial hmms, with application to energy disaggregation. *Advances in Neural Information Processing Systems*, 27:3590–3598, 2014.
[54] Zhongliu Zhuo, Yang Zhang, Zhi-li Zhang, Xiaosong Zhang, and Jingzhong Zhang. Website fingerprinting attack on anonymity networks based on profile hidden markov model. *IEEE Transactions on Information Forensics and Security*, 13(5):1081–1095, 2017.