TrafficSpy: Disaggregating VPN-encrypted IoT Network Traffic for User Privacy Inference

Qi Li, Keyang Yu, Dong Chen, Mo Sha*, Long Cheng†
Department of Computer Science, Colorado School of Mines, Golden, USA
*Florida International University, Miami, USA
†Clemson University, USA

***Abstract*—People have been increasingly deploying the Internet of Things (IoT) devices to monitor and control their environments. Unfortunately, extensive recent research has shown that IoT devices are vulnerable to multiple adversarial attacks, which analyze their network traffic to reveal a wide range of sensitive private information about user in-home activities. Thus, smart home users recently have a keen interest in employing virtual private networks (VPN) to obscure their privacy information in their IoT network traffic. Our key insight is that VPN-encrypted IoT network traffic data is not anonymous, since this aggregate traffic data can still be disaggregated into individual IoT device traffic data. And this individual IoT device traffic may have an identifiable traffic signature that already embeds detailed user sensitive information.**

To explore the severity and extent of this privacy threat, we design a new factorial hidden Markov model (FHMM)-based smart home network traffic disaggregator—TrafficSpy that can accurately disaggregate VPN-encrypted whole-house IoT network traffic data into individual IoT device network traffic data. We evaluate TrafficSpy using VPN network traffic data from three smart homes. We find that TrafficSpy can disaggregate VPN traffic data into individual IoT device data accurately. We also show that the disaggregated traffic traces can be further attacked by smart and adaptive adversaries and thus reveal user sensitive information. TrafficSpy represents a serious privacy threat, but also a potentially useful tool that provides important contextual information for smart home monitoring and automation.

***Index Terms*—Disaggregation, IoT Privacy, Smart Homes, Machine Learning, Deep Learning**

I. Introduction

People are increasingly deploying the Internet of Things (IoT) devices in smart homes to monitor and control their environment. The total installed base of the IoT devices is projected to amount to 5.44 billions worldwide by 2025, a fivefold increase in 10 years [28]. Network traffic data generated by these IoT devices is recorded by Internet Service Providers (ISPs) and their third-parties to maintain customer services, such as generating monthly bills, personalizing data plan, and detecting network outages. In addition, recent IoT privacy study [17] shows that 72 out of 81 popular IoT devices are sharing data with third-parities (e.g., Google, Amazon and Akamai) completely unrelated to original manufacturer and far beyond basic necessary device configuration, including voice speakers, smart TVs, and streaming dongles. Unfortunately, significant recent research [8], [23], [6], [7], [5], [26], [1] has shown that IoT device traffic data has significant privacy threats. In addition, it is surprisingly easy for on-path adversaries to launch data analytics attacks to infer user sensitive information from IoT traffic data, since user in-home activity highly correlates with simple time-series data statistical metrics, such as mean, variance, and range.

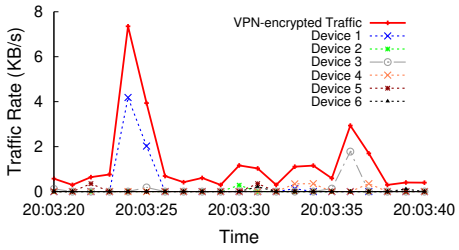


Fig. 1. An example of VPN-encrypted IoT traffic traces.

Recently, smart home users have a keen interest in employing virtual private networks (VPN) to prevent their privacy leakage [12], [18], [19]. When a VPN service is enabled on the router or gateway, VPN will establish a virtual peer-to-peer (P2P) encrypted "tunnel" over the Internet. This can secure the data traveling between the smart home and IoT manufacturers and/or remote cloud servers, which are usually required by smart home automation. Considering the tradeoff between speed and security, OpenVPN is the most widely adopted protocol to hide smart home user private information [22]. Figure 1 shows an example of OpenVPN-encrypted IoT traffic trace from a smart home. We can observe that different individual IoT device traffics are hidden under the VPN-encrypted IoT traffic to protect user privacy. Thus, such VPN-encrypted traffic data from smart homes is often not treated as sensitive: instead, it is routinely transmitted over Internet, shared with ISPs, IoT manufacturers, and third-parities. To make it worse, recent news [24], [25], [20] reported that 27 out of 117 VPN companies may share or sell user network traffic data to third-parties.

It is generally believed that VPN-encrypted network traffic is anonymous. However, our key insight in this work is that VPN-encrypted IoT traffic data can be disaggregated into individual IoT device traffic data. Since device-level IoT network traffic data may have a unique network traffic signature that already embeds detailed user in-home activity information [1], [31], such disaggregated per-device traffic

data may indirectly reveal user privacy information that might be interesting for insurance companies, marketers, or the government. To explore the severity and extent of this privacy threat, we design a new factorial hidden Markov model (FHMM)-based network traffic disaggregator—TrafficSpy that can accurately disaggregate VPN-encrypted whole-house IoT network traffic data into individual IoT device traffic data. Our hypothesis is that FHMM-enabled disaggregator is able to recover the unknown individual IoT device level network traffic rate signals, given only the observed aggregate VPN-encrypted network traffic rate measurements. In evaluating our hypothesis, we make the following contributions.

TrafficSpy Design. We present the design of network traffic disaggregator—TrafficSpy, which can accurately disaggregate VPN-encrypted whole-house IoT network traffic into individual IoT device network traffic. In essence, TrafficSpy employs network traffic data preprocessing, intelligent device traffic pattern learning, and FHMM-based network traffic disaggregation to accurately disaggregate whole-house aggregate traffic traces. We also design optimization techniques to further improve TrafficSpy's disaggregation performance.

Implementation and Evaluation. We implement TrafficSpy both simulator and prototype in `python` using widely-used open-source frameworks. We evaluate TrafficSpy using 105 days VPN network traffic rate traces from 3 different homes. We find that TrafficSpy can disaggregate VPN traffic traces into individual IoT device trace accurately. We also show that the disaggregated traffic traces can be attacked by Machine Learning (ML)- and Deep Learning (DL)-based smart and adaptive attacks, and thus reveal user sensitive information.

Releasing Anonymized Datasets and Code. TrafficSpy represents a serious privacy threat, but also a potentially useful tool for smart home monitoring and automation. Our approaches to disaggregate VPN-encrypted smart home network traces and adversarial attack models are quite general, and can be applied to address similar problems in other research domains, such as smart grid and medical e-health system. We release source code and datasets to broad IoT research communities on our website [29].

II. Background and Related Work

A. Problem Statement

The aim of network traffic disaggregation in a smart home is to provide accurate individual IoT device traffic consumption estimates based on the VPN-encrypted whole-house IoT network traffic consumption. In essence, given a specific smart home, we assume there are N IoT devices, and for each IoT device, the traffic rate signal is represented as $X_i = (X_i1, X_i2, X_i3, ..., X_it)(1 \leq t \leq n)$ where n denotes the duration of recorded traffic rate trace and $X_it \in \mathbb{R}$ represents the network traffic rate measured in KB/s or MB/s by on-path network observers, such as Internet Service Providers (ISPs), VPN providers, and/or third-parties. Then, we can describe the main network traffic rate as the summation of the individual IoT device traffic rate signals with the following form: $Y_t = \sum_{i=1}^{N} X_it + \epsilon_t$ where ϵ_t is an error term which is mainly comprised of network traffic noises generated by VPN service, router or gateway maintenance, and IoT devices' heartbeat background network traffic. This signal—Y_t is assumed to be the aggregation of network traffic consumed by the component IoT devices in a building. Thus, our goal of the network traffic disaggregation problem is to recover the unknown network traffic rate signals—X_i given only the observed aggregate VPN-encrypted network traffic rate measurements—Y.

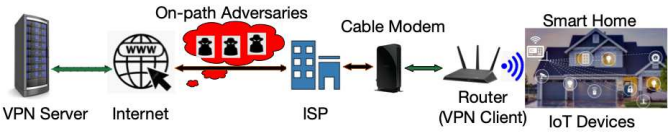


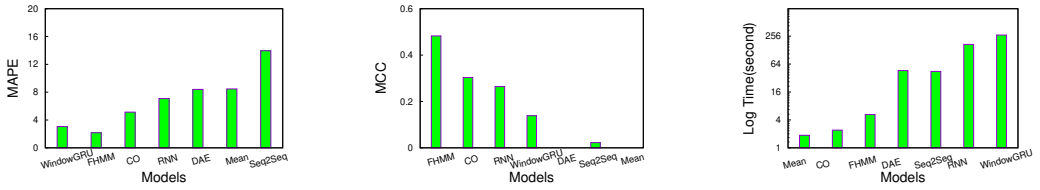
Fig. 2. Overview of our privacy threat model.

B. Privacy Threat Model

TrafficSpy assumes either a software or hardware VPN router is deployed in a smart home. A VPN wraps all smart home traffic from IoT devices in an additional transport layer. By doing so, the VPN can aggregate all the traffic into a single traffic flow with the source and destination addresses of the VPN endpoints. As shown in Figure 2, we are broadly concerned with the capabilities of ISPs, IoT manufactures, on-path network observers, VPN service providers, and third-parties to infer user in-home activities from VPN-encrypted smart home network traffic. The network traffic rate metadata, including inbound/outbound traffic rates, network protocols, source, and destination IPs, package sizes and etc., are accessible to these entities. The potential adversaries may be interested in inferring user activities from smart home network traffic to gain profit, while users usually do not want to share such privacy-sensitive information with others. We assume adversaries can use any data analytics techniques to infer certain types of information from the observed patterns in recorded traffic traces (but they cannot manipulate the traffic). This is a typical assumption in existing privacy leakage analysis works [2], [9], [30].

In particular, we are concerned with 3 types of privacy attacks: i) *Learning occupancy from the data.* This includes whether a home is occupied and when; ii) *Learning user in-home activities from the traffic data.* User activities may include when users come and go, when they perform their daily activities, such as going to bed, waking up, watching TV, listening to music, playing online games, as well as more complex questions, such as whether a household has a baby, and whether they go on vacation on weekends; iii) *Learning network traffic pattern information from the data.* This includes whether a particular IoT device is present in a home, what model of an IoT device is present, and how much traffic the home consumes on it monthly.

Attack Scenario #1: An external adversary from ISPs, IoT device manufacturers or third-parties is actively monitoring



(a) MAPE (b) MCC (c) Training Time

Fig. 3. Comparison of 7 disaggregation approaches based on (a) MAPE, (b) MCC, and (c) Training Time using 20.5 days traffic rate (kB/min) trace.

the VPN aggregate traffic traces for a target home and then uses data analytics approaches to learn the indirect user privacy information that might be interesting for insurance companies, marketers, or government.

Attack Scenario #2: To infer the type of IoT devices and user activities at a certain home, an external on-path adversary intends to acquire the real-time VPN aggregate IoT network traffic traces. The attacker can leverage ML/DL-based disaggregation models to separate the VPN aggregate traffic traces into per-device traffic traces. Then, the external attacker may further identify the IoT devices using per-device traffic traces and launch cyberattacks on them.

C. Related Work

There is not significant research in IoT VPN traffic disaggregation area. The most relevant ones are the netmeter energy data disaggregation work in Non-intrusive Load Monitoring (NILM) community. We outlined and implemented a wide range of the most recent design alternatives [16], [3], [11], [14], [4], [32], [33] that could be adapted to disaggregate VPN traces in smart homes. **Combinatorial Optimization (CO)** [3] assumes each device has a linear model such that we can identify which device is "on" and "off" at each time—t. Thus, the aggregated signal at time—t can be described as the sum of all the individual load signal of the devices that are "on" at that time. **MEAN** algorithm [11] only calculates and keeps the mean state values for each device. During the modeling training process, the mean values are updated every time the same device is "on". **Denoising Autoencoder (DAE)** is a special kind of deep neural networks that can extract or reconstruct a particular component from noisy input. Multiple trained models are required to disaggregate the aggregate readings since DAE can only denoise on a per-device basis. **Recurrent Neural Network (RNN)** [14] receives a sequence of main readings and outputs a single value of power consumption of the target appliance. The RNN approach might have the vanishing gradient problem in modeling training process. **Sequence-to-Sequence (Seq2seq)** model [32], [14] learns a regression mapping from the main sequence to the corresponding target device sequence for a single-channel blind source separation (BSS) with the same timestamps. The Seq2seq is an essentially posterior density estimator. **Sliding Window Gated Recurrent Units Network (WindowGRU)** [16] is an improvement to the RNN approach that uses long short-term memory (LSTM) neurons and suffers from their high computational cost. WindowGRU replaces LSTM neurons with light-weight Gated Recurrent Unites (GRU). Theneural networks of WindowGRU will analyze the time slice $[t, t-w]$ to predict the per-device consumption at time—t where w is the sliding window size. **Factorial Hidden Markov Model (FHMM)** [33] is a generalization of the multi-HMMs in which the hidden state is "weighted" into multi-state variables, and thus can model the whole-house as a factorial HMMs model. The operational states of each device are learned via Markow chain, and the dependence between different time slots are also learned.

Observation and Summary. Figure 3 (a) shows that WindowGRU and FHMM yield the lowest average Mean Absolute Percentage Error (MAPE) values as of 3.054 and 2.170, respectively. They achieve the best disaggregation accuracy. Meanwhile, Figure 3 (b) shows that FHMM approach yields the highest average Matthews Correlation Coefficient (MCC) value of 0.483, which indicating that FHMM approach achieves the best accuracy when distinguishing one individual IoT device from other IoT devices in the same household. Unsurprisingly, MEAN approach reports MAPE and MCC as of 8.466 and 0, respectively. This is mainly due to the fact that MEAN algorithm tends to work in "always on" mode to predict the per-device network traffic. In addition, Figure 3 (c) reports that DAE, Seq2seq, and WindowGRU all have very long training overhead. This is mainly due to their architecture design in which multi-layer neural networks are typically employed to learn the transition features. MEAN and CO yield the lowest training overhead, since they mainly depend on empirical threshold analytics instead. To achieve the tradeoff among MAPE, MCC and training overhead, FHMM approach is the most suited technique to address our IoT traffic disaggregation problem. These valuable insights will guide the development of our proposed technique—TrafficSpy.

III. TrafficSpy Design

A. System Design

Figure 4 shows the system structure of our disaggregation system—TrafficSpy that has multiple steps, including network traffic converting, statistics, preprocessing, training, model building, disaggregation and adaptive evaluation. In essence, TrafficSpy will first convert all the network traffic rate traces into the common timeseries dataset format inspired

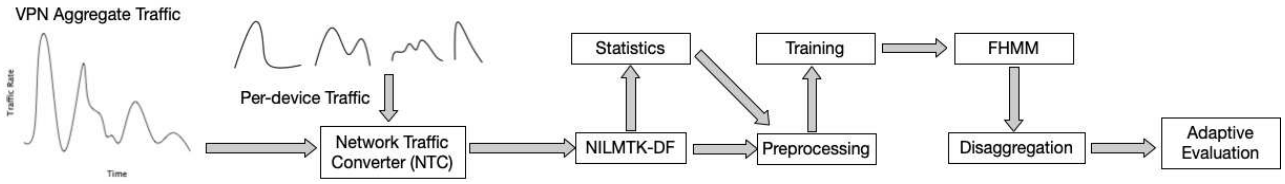


Fig. 4. The system pipeline of TrafficSpy.

by the REDD format [15] which is widely used in many NILM-based disaggregation problems. Next, TrafficSpy will perform statistical analytics on the converted common data file to ensure its correctness and effectiveness. Then, TrafficSpy will preprocess the data into NumPy arrays and split the whole dataset into training, testing and validation datasets. After that, TrafficSpy will build leverage the Factorial Hidden Markov Model (FHMM) model to learn the relationships among different IoT device network traffic consumption signals. Eventually, TrafficSpy will apply the FHMM-based approach to disaggregate VPN whole-house traffic traces into individual device traffic traces. TrafficSpy also integrates with smart and adaptive attack models that can directly evaluate user privacy leakage degree in the disaggregated per-device traffic traces.

B. Network Traffic Data Preprocessing

Network Traffic Converting. Given a new target home, the inputs for TrafficSpy are the aggregate whole-house network traffic rate traces from the new house, and the individual device traffic rate traces that TrafficSpy learns from other smart homes or public online repositories. The aggregate network traffic rate traces are the traffic volume data that might be observed by on-path adversaries. TrafficSpy leverages the per-device traffic rate traces as the groundtruth data to learn or calibrate disaggregation models. TrafficSpy employs network traffic converter interface (NTCC) to collect and convert these per-device historical traffic rate traces both "online" and "offline". In particular, NTCC also provides users with the public APIs to collect per-device traffic rates.
Network Traffic Sampling. In addition to the above-mentioned traffic data converting, TrafficSpy also provide automatic up-sampling and down-sampling functions in NTCC to support different granularity level aggregated whole house traffic data disaggregation. This automatic sampling process of TrafficSpy enables external on-path adversaries to launch multiple ML-based or DL-based user privacy attacks at different granularity traffic rate traces.

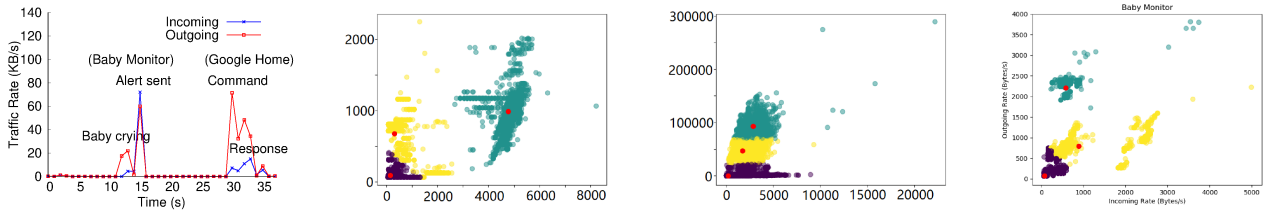
C. Intelligent Device Traffic Signature Learning

Next, as shown in Figure 5 (a), TrafficSpy will learn individual IoT device traffic rate signatures from its historical traffic rate signatures, which are used in its later to train disaggregation algorithms. Different IoT devices typically have different network traffic signatures. In particular, the non-interactive background traffic patterns in which there are no occupant engaged might be similar from house to house for a specific IoT device. Thus, it is possible to observe the similar background traffic for IoT devices across different smart homes. However, different smart home occupants may operate their interactive IoT devices in different ways. That says, even for the same IoT device, it would be challenging for us to use some homes' interactive traffic loads and patterns as the proxy to "predict" the same IoT device traffic loads and patterns at new homes. To mitigate this issue, we propose to leverage K-Means clustering algorithm-based approach to learn the interactive load patterns. As shown in Figure 5 (b) $\sim$ (d), our analytics using K-Means clustering algorithm where $K = 3$ has shown that IoT devices typically have identifiable three traffic consumption patterns, including low or background (in purple), medium (in yellow), and high (in cyan) traffic mode. In particular, the background traffic volume for IoT devices typically should be the same in different smart homes. Eventually, we apply the K-Means clustering algorithm where $K = 1$ on all IoT devices to infer the threshold—$T_{background}$ that we can leverage to model the background traffic, and also the medium/high threshold—T_{active} that enables TrafficSpy to more efficiently disaggregate whole-house aggregate traffic.

Note that, TrafficSpy's approach to estimating the traffic flows of IoT devices in a smart home is orthogonal to the other aspects of the technique and is thus "pluggable," such that we could use other machine learning approaches to estimate background traffic flows.

D. Building FHMM-based Traffic Disaggregation Model

As we studied in Section II, our insight is that considering the MAPE, MCC and training overhead benchmarking metrics, FHMM approach is the most well suited technique to address our smart home VPN traffic disaggregation problem. In this section, we will explain how we leverage FHMM approach to build TrafficSpy's disaggregation process.
Learning Process. The Hidden Markov Model (HMM) [10] is a widely used algorithm to learn probalilitstic results from time series data. Hidden Markov Model (HMM) assumes a sequence of observations $Y_t = Y_1, Y_2, Y_3, ..., Y_T$ can be modeled using a specific probilistic relationships to a sequence of hidden IoT device status $S_t = (S_1, S_2, S_3, ..., S_T)$. This HMM-based structure will form a Markov chain to convey the past per-device network traffic consumption information. The formal relationships can be described as the following joint probability factors:



(a) Traffic Signature Learning (b) Insteon Camera (c) TP-Link Camera (d) Baby Monitor

Fig. 5. The illustration of traffic signature learning and applying K-Means on 3 IoT devices to identify traffic loads (e.g., background/low, medium, high).

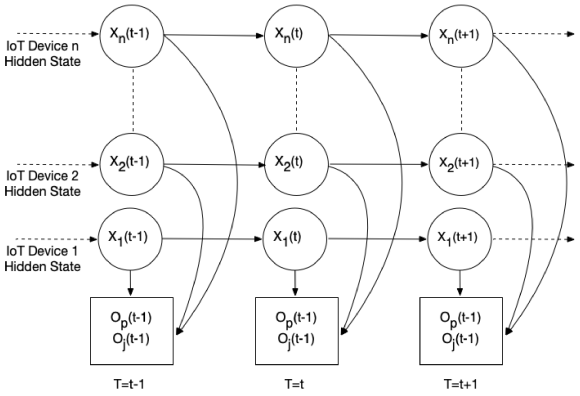


Fig. 6. The structure overview of TrafficSpy's FHMM approach.

$$P(S_t, Y_t) = P(S_1)\cdot P(Y_1 \mid S_1)\cdot \prod_{t=2}^{T} P(S_t \mid S_{t-1})\cdot P(Y_t \mid S_t) \tag{1}$$

Thus, in the HMM model, S_t is independent of $(S_1, S_2, S_3, ..., S_{t-2})$ given S_{t-1}. While, FHMM [33] is a generalization of the multi-HMMs in which the hidden state is "weighted" into multi-state variables, and thus can model the whole-house as a factorial HMMs model. In particular, the operational status of each device or appliance are learned via Markow chain at each time—t, and the dependence between different time slots are also learned in the process. These make FHMM well suited for our distributed manner problem—whole-house network traffic disaggregation.

Figure 6 shows the structure of TrafficSpy's FHMM approach. The FHMM-based approach can be described as follows,

$$P(S_t \mid S_{t-1}) = \prod_{n=1}^{N} P(St^{(n)} \mid S_{t-1}^{(n)}) \tag{2}$$

where $P^{(n)}$ denotes the different transitions matrices, and $\prod^{(n)}$ represents the initial probabilities. Then, we can have:

$$P(S_t, Y_t) = P(S_1)\cdot P(Y_1 \mid S_1)\cdot \prod_{t=2}^{T} P(Y_t \mid S_t)\cdot \prod_{n=1}^{N} P(S_t^{(n)} \mid S_{t-1}^{(n)}) \tag{3}$$

We leverage the Gaussian model to capture the continuous IoT device traffic rate observations Y_t at different time t, whose mean is a linear function of the state variables defined as follows,

$$P(Y_t \mid S_t) = |C|^{-\frac{1}{2}} (2\pi)^{-\frac{D}{2}} exp[-\frac{1}{2}(Y_t - \mu_t)' \cdot C^{-1} \cdot (Y_t - \mu_t)] \tag{4}$$

where $\mu_t = \sum_{n=1}^{N} W^n \cdot S_t^{(n)}$. Here, $W^{(n)}$ represents a DxN matrix in which columns are indicating the contributions towards mean values for each $S_t^{(n)}$ setting. C is the DxD covariance matrix, and $'$ represents the operation of matrix transpose, and $|C|$ denotes the determinant operator of C.

Optimization Process. However, the approach we build up so far is intractable for large value of N which is the number of IoT devices in a smart home. That says it is often expensive in both training time and memory capacity to perform the exact inference solutions when $N > 30$. To mitigate this issue, we apply the approximate FHMM technology by relaxing the hidden state values and transforming the exact FHMM inference problem into a convex problem [33].

E. Adaptive User Privacy Inference

To further explore the severity and extent of the privacy threat, TrafficSpy also provides a full stack of ML-based and DL-based adversarial inference models.

Feature Selection. To identify principle features of IoT traffic rate data, we empirically examine the statistical features based on timeseries motifs in each IoT device traffic traces, including duration, mean, maximum and minimum values, standard deviations, range, Skewness, variation coefficient, kurtosis, area under the curve (AUC), etc. We leverage Principal Component Analysis (PCA) to analyze the principle features from IoT network traffic rate traces.

Machine Learning or Deep Learning Classifier Selection. We then investigate the most widely used ML classifiers in prior IoT traffic research work to build ML-based adversarial attack models, including Logistic Regression, Support Vector Machines (SVMs), and Random Forest. In particular, we also benchmark different kernels for SVMs, including linear, linear passive-aggressive, linear ridge, polynomial with 1∼10 degrees, and radial basis function (RBF). We also design a convolutional neural networks (CNNs)-based deep learning approach to infere user in-home activities. The CNNs architecture of TrafficSpy is comprised of input, convolutional layers (ReLU), max pooling, fully-connected layers and output. Two fully-connected layers with ReLU

and another fully-connected layer (without ReLU) are added to further process the outputs.

Adaptive Adversarial Attacks on Disaggregated Traffics. In addition, TrafficSpy also enables a set of deeper adaptive adversarial attacks on the disaggregated individual IoT traffic traces. Our hypothesis is that online attackers may gain their knowledge-level about the target house and thus can integrate this information with their attack models to better infer use in-home private activity information. To do so, we fine-tune or calibrate the model parameter space using the knowledge-level information when building the ML-based or DL-based attack models. This will examine TrafficSpy's ability to explore the user privacy threat of the disaggregated individual traffic rate traces.

F. Online Optimizations

Individual Traffic Rate Adjustment. TrafficSpy also introduces optimization techniques to further improve the disaggregation performance. TrafficSpy also dynamically adjusts its background traffic threshold and rate of traffic spikes over time to match the expected rate each period. TrafficSpy only ensures these time periods look the same with respect to each other, regardless of whether a home is occupied or not. In addition, TrafficSpy also indexes its traffic rate signatures database based on each IoT device's traffic rate signature's real time-of-use. At any time, TrafficSpy tries to select from the past traffic rate signatures that occurred near that time.

IV. Implementation

We implement TrafficSpy in `python` using widely available open-source frameworks, including Pandas, Scikit-learn and PyCUDA. The simulator takes a home's VPN aggregate whole-house network traffic race traces as input and applies different disaggregation techniques outlined in Section III. The outputs of TrafficSpy are the separated individual IoT device traffic rate traces. To implement the 7 different disaggregation algorithms discussed in Section II and Section V, we leverage the open source NILM toolkit (NILMK) V0.2 to build those disaggregation approaches. As shown in Figure 7, we also deploy a prototype TrafficSpy using Raspberry Pi 4 Model B-based hardware (BCM2711) in a "mock" smart home on our campus to demonstrate TrafficSpy's ability to disaggregate smart home network traffic rate in real-time.

Note that, we use Raspberry Pi 4 here to build a prototype to demonstrate and validate TrafficSpy's performance. On-path adversaries can deploy TrafficSpy on their servers or gateways to disaggregate VPN traces and infer user private information. *We do not assume or require external adversaries to have Raspberry Pi installed in their end.* We use the Scikit-learn machine learning library in `python` to build our machine learning attack approaches, including Logistic Regression, Support Vector Machines (SVMs), and Random Forest. For CNNs-based attack approaches, we implement based on the framework from VGGnet. For user in-home activities, we implement LSTM-based user in-home activities modeling using Keras model library.

Fig. 7. The overview of our PrivacyGuard prototype.

V. Experimental Evaluation

A. Datasets

Dataset 1: UNSW. We downloaded the publicly-available IoT traffic rate traces from UNSW Sydney [27] that includes second level network traffic traces of 22 IoT devices for 20.5 days. To evaluate our TrafficSpy approaches, we process the IoT traffic metadata traces to IoT traffic rate data for disaggregation and also label all the user activities for adaptive adversarial evaluation.

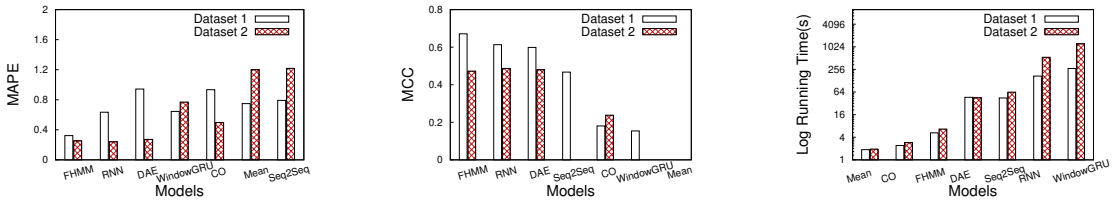
Dataset 2: Real Home VPN Dataset. We deploy IoT devices in a volunteer's house and set up a router-based hardware VPN environment. The home is a private townhouse apartment with 3 occupants operating 22 IoT devices daily. We configure CyberGhost VPN service using OpenVPN protocol on the router—Netgear R6700-V3 (DD-WRT firmware), and VPN servers are dynamically chosen to get better speed for torrent downloading. All the 22 IoT devices are hidden behind this router that is hosting the VPN client service to encrypt the communication channel. We have collected 105 days of VPN aggregate whole-house network traffic data. Table I shows the IoT device samples of this dataset.

Dataset 3: User Activity Groundtruth Dataset. To label user activity in public datasets rather than our own, we develop a script to search motifs in aggregated traffic spikes. We cluster and process groundtruth user activities data comprehensively. For our own datasets, we have been logging user activities in our smart homes. We anonymize groundtruth datasets along with the above-mentioned datasets.

Ethical Consideration for Data Collection. Data collection participants were one-to-one interviewed and provided user privacy consent and agreement. Before sharing datasets, we remove user identical information and sampled the datasets.

B. Experimental Setup

We implement a general version of Combinatorial Optimization (CO), MEAN Approach, Denoising Autoencoder (DAE), Recurrent Neural Network (RNN), Sequence-to-Sequence (Seq2seq), and Sliding Window Gated Recurrent Units Network (WindowGRU) using prior work proposed in [3], [11], [1], [9], [13], [14], [32], [16]. We then implement TrafficSpy, including network traffic data preprocessing, intelligent device traffic pattern learning, and FHMM-based traffic disaggregation to accurately disaggregate whole-house aggregate traffic traces into individual IoT device traffic rates.



(a) MAPE (b) MCC (c) Training Overhead

Fig. 8. The performance comparison of applying different traffic disaggregation approaches on both UNSW dataset and VPN dataset.

Device	MAC Address	IP Address
Amazon Echo	14:0A:C5:8A:73:76	192.168.1.116
Belkin Switch	24:F5:A2:FF:91:1D	192.168.1.110
Google Home	20:DF:B9:5C:72:62	192.168.1.128
Insteon Hub	28:6D:97:77:5F:D5	192.168.1.143
Philips Hue	EC:B5:FA:04:AE:96	192.168.1.123
SmartHome Hub	00:0E:F3:45:73:31	192.168.1.124
XiaoMi Camera	D4:B7:61:5F:B4:6D	192.168.1.102

TABLE I
LIST OF IOT DEVICE SAMPLES IN DATASET 2.

C. Evaluating Metrics

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

$$MAPE = \frac{100}{n}\sum_{t=0}^{n}|\frac{S_t - P_t}{S_t}| \quad (5)$$

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). To quantify the disaggregation accuracy, we use the MCC [21], a standard measure of a classifier's performance for imbalanced dataset, where values are in the range -1.0 to 1.0, with 1.0 being perfect individual traffic disaggregation, 0.0 being random individual traffic disaggregation, and -1.0 indicating individual traffic disaggregation is always wrong.

D. Experimental Results

1) Quantifying Disaggregation Accuracy of Different Approaches: We first quantify disaggregation accuracy when applying 7 disaggregation approaches on both Dataset #1 and Dataset #2, including CO, MEAN, DAE, RNN, Seq2seq, WindowGRU, and TrafficSpy. Figure 8 shows the quantitative comparison results using MAPE (a), MCC (b), and training overhead (c). Unsurprisingly, CO and MEAN approaches report the worst MAPE and MCC results and the best training overhead. This is mainly due to fact that CO assumes each device has binary "on" and "off" status and MEAN assumes all the devices are "on" and tries to use their mean values to separate them. In addition, both CO and MEAN leverage thresholds to disaggregate the VPN aggregate traffic, and thus do not require significant training overhead. We observe that TrafficSpy and neural networks-based approaches (e.g., RNN, DAE) are reporting reasonably accurate disaggregation results. In particular, TrafficSpy (FHMM) yields the best MCC and the worst MAPE. Figure 8 (c) reports training overhead across 7 approaches. We find that excluding CO and MEAN, TrafficSpy reports the shortest training time.

Results: *Comparing with the other 6 recent disaggregation approaches, TrafficSpy is the best performing disaggregation approach across two different datasets. TrafficSpy yields average MAPE as 0.323, which is ∼2 times better than the baseline approach—MEAN and ∼3 times better than CO.*

2) Quantifying Disaggregation Accuracy When Varying Granularity of Traffic Traces: We next evaluate traffic disaggregation effect on different trace rate traces that have different level of granularities, such as 1 minute, 3 minutes, 5 minutes, 10 minutes, 30 minutes and 60 minutes. As shown in Figure 9 (a), as expected, higher granularity results in lower MAPE results. This is mainly due to the facts that 1) TrafficSpy performs consistently well on different traffic rate traces at different granularities, 2) fewer fluctuations and spikes are observed in higher resolution traffic rate traces. When traffic rate traces are becoming coarser, error percentages are declining.

Results: *TrafficSpy's accuracy is a linear function of the granularities of IoT traffic rate traces. TrafficSpy yields the MAPE of 0.15 when disaggregating on 10 minutes level network traffic rate traces, which is ∼4 times less than as the baseline MEAN approach.*

3) Quantifying TrafficSpy's Scalable Performance: Next, we examine the performance effect on TrafficSpy when disaggregating network traffic traces having different amount of IoT devices. In doing so, we are evaluating the scalability performance of TrafficSpy. Figure 9 (b) shows that the larger amount of the IoT devices in a smart home has resulted in a better overall disaggregation performance. We observe that TrafficSpy achieves lower MAPEs when the number of IoT devices increases from 3 to 40.

Results: *Larger smart home size has resulted in lower MAPEs of TrafficSpy. With the growth of the number of the IoT devices, TrafficSpy achieves better overall performance.*

4) Quantifying TrafficSpy's Performance When Adding New Device in Smart Homes: Next, we examine the performance effect on TrafficSpy when disaggregating the network

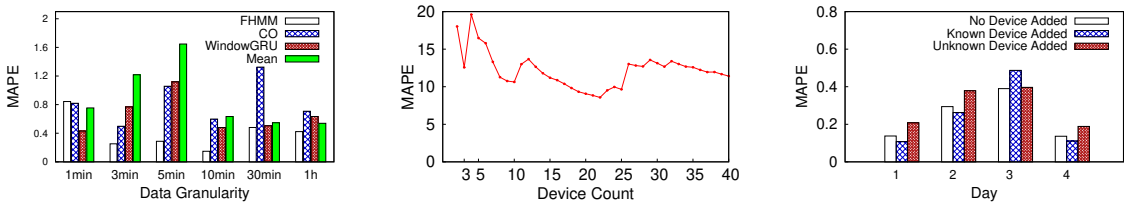


Fig. 9. The disaggregation accuracy comparison on different traffic granularities (a), different amount of IoT devices (b), and adding new IoT device (c).

traffic traces that have new IoT devices installed. In doing so, we are evaluating the robust performance of TrafficSpy. As shown in Figure 9 (c), TrafficSpy reports similar or slightly lower disaggregation accuracy in MAPE (∼6.62%) when smart home installs new IoT device that TrafficSpy knows its traffic signatures. We also find that TrafficSpy yields similar or slight worse disaggregation accuracy in MAPE (∼6.61%) when smart home owner installs new IoT device that TrafficSpy does not know its traffic signatures. Note that, constant repeated traffic patterns in a unknown IoT device trace can be linked to the new IoT device when TrafficSpy learns its traffic signature from other smart homes or public IoT traffic repositories. **Results**: *TrafficSpy reports 6.62% changes in MAPEs when smart home users installing a new known and unknown IoT device. TrafficSpy is robust to new IoT device becomes online.*

5) Attacking User Occupancy: We then examine the effectiveness of user occupancy detection from the individual IoT device traces that are disaggregated by different approaches. Figure 10 (a) shows the comparison results of attacking on per-device traces. We observe that TrafficSpy (FHMM) yields the best MCC as of 0.636 and is the best performing approach to detect user occupancy status.

Results: *Comparing with other disaggregation approaches, TrafficSpy achieves the best MCC value as of 0.636 to attack user occupancy status in a target home.*

6) Attacking User In-home Activities: We next benchmark the effectiveness of inferring user activities from individual IoT device traces that are disaggregated by 7 different disaggregation approaches. We leverage ML/DL-based attack models that we proposed in Section III. Unsurprisingly, as shown in Figure 10 (b), we observe that TrafficSpy (FHMM) yields the best MCCs as of 0.523, 0.456, and 0.581 across three different user in-home activities attacking. TrafficSpy is constantly the top-1 disaggregator.

Results: *Comparing with other 6 disaggregation approaches, TrafficSpy constantly achieves the best MCC values when attacking user in-home activities. TrafficSpy exposes a serious user privacy threat.*

7) Inferring User In-home Activities by Adaptive Adversary: We next examine the effect of different knowledge-level adaptive adversary on privacy inference from TrafficSpy's separated traffic. We define attacker knowledge-level as the percentages of traffic rate testing dataset that an external adversary can leverage to calibrate its attack models to infer user in-home activity. 0% indicates that the external adversary has no knowledge of the target home testing dataset and no cross-validation is performed to train the attack models. While, 100% means the external adversary has observed all the groundtruth traffic patterns for each user activity so that the attack models are "perfectly" trained. Figure 10 (c) shows the ability of TrafficSpy to infer user private information when adaptive adversary having more knowledge about our attack model. To report MCCs in Figure 10 (c), we apply KNN-based, SVM-based, Decision Tree (DT)-based attack models on per-device traffic traces generated by TrafficSpy to infer 4 user activities and report the mean value of the MCCs, respectively. We find that TrafficSpy's MCC increases from 0.47 to 0.91. This is because inferring user in-home activity is surprisingly easy as discussed in Section II.

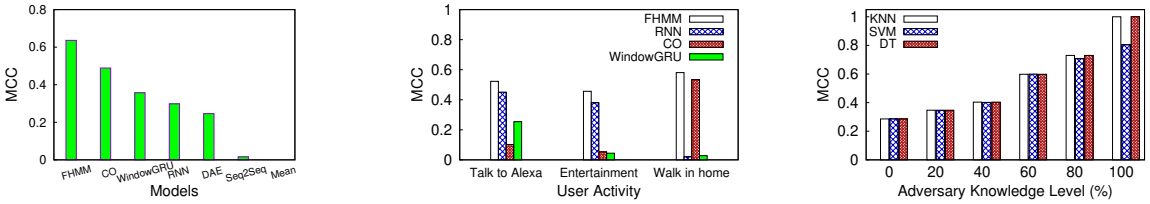
Results: *TrafficSpy's MCC is increasing from 0.47 to 0.91 when varying adversary knowledge level from 0% to 100%. Thus, TrafficSpy exposes a serious user privacy threat.*

E. Designing User Privacy-preserving Approaches

Our experimental evaluation results have shown that on-path adversaries can infer and fingerprint users' sensitive privacy information such as occupancy and user in-home activities by only analyzing VPN-encrypted IoT network traffic traces. We are developing a new low-cost, open-source and traffic reshaping based user privacy defense system to significantly reduce the private information leaked through IoT device VPN traces, while still permitting sophisticated control that is necessary in smart home management. In essence, we plan to use intelligent traffic signature learning, long short-term memory (LSTM)-based artificial traffic signature injection, and partial traffic reshaping to obfuscate user in-home privacy. We also plan to develop optimization techniques to further reduce traffic reshaping overhead.

VI. Conclusion and Future Work

We design a new open-source factorial hidden Markov model (FHMM)-based network traffic disaggregator—TrafficSpy that can accurately disaggregate VPN-encrypted whole-house IoT traffic traces into individual IoT device traffic traces. We evaluate TrafficSpy by using 105 days IoT network traffic traces and deploying a hardware prototype. We find that TrafficSpy can accurately disaggregate VPN traffic traces. Our evaluation also show that we can accurately further detect 4 different kind of user in-home private information. Thus, TrafficSpy represents a serious privacy threat. We plan to deploy TrafficSpy in more smart homes



(a) Occupancy (sorted using MCC) (b) User Activity (c) Different Adversary Knowledge

Fig. 10. The comparison of training overhead, occupancy detection and user activity detection.

which more commercially-available IoT devices to further benchmark and improve its online performance.

ACKNOWLEDGEMENT

We would like to thank the anonymous reviewers for providing us their insightful comments and valuable feedback, which significantly improved the quality of this paper. Qi, Keyang and Dong are supported by Mines/NREL Nexus Seed Grants. Mo is supported by NSF CNS-2150010.

REFERENCES

[1] 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.

[2] Noah Apthorpe, Dillon Reisman, Srikanth Sundaresan, Arvind Narayanan, and Nick Feamster. Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic. *arXiv preprint arXiv:1708.05044*, 2017.

[3] MH Bjorndal, Alberto Caprara, Peter I Cowling, Federico Della Croce, H Lourenco, Federico Malucelli, Alex J Orman, David Pisinger, Cesar Rego, and Juan Jose Salazar. Some thoughts on combinatorial optimisation. *Journal of Operational Research*, 83(2):253–270, 1995.

[4] Hafsa Bousbiat, Christoph Klemenjak, and Wilfried Elmenreich. Exploring time series imaging for load disaggregation. In *Proceedings of BuildSys*, pages 254–257, 2020.

[5] Phuthipong Bovornkeeratiroj, Srinivasan Iyengar, Stephen Lee, David Irwin, and Prashant Shenoy. Repel: A utility-preserving privacy system for iot-based energy meters. In *IEEE/ACM IoTDI*, pages 79–91, 2020.

[6] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. A systematic approach to developing and evaluating website fingerprinting defenses. In *ACM CCS*, pages 227–238, 2014.

[7] Dong Chen, David Irwin, Prashant Shenoy, and Jeannie Albrecht. Combined heat and privacy: Preventing occupancy detection from smart meters. In *IEEE Percom*, pages 208–215, 2014.

[8] Wenbo Ding and Hongxin Hu. On the safety of iot device physical interaction control. In *ACM CCS*, pages 832–846, 2018.

[9] 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 *IEEE Security & Privacy*, 2012.

[10] Sean R Eddy. Hidden markov models. *Current opinion in structural biology*, 6(3):361–365, 1996.

[11] George William Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.

[12] Global Market Insights. Virtual Private Network Market: Top Trends Boosting The Industry Demand Through 2027. https://www.totaltele.com/512209/Virtual-Private-Network-Market-Top-trends-boosting-the-industry-demand-through-2027, Jan 2022.

[13] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an efficient website fingerprinting defense. In *Symposium on Research in Computer Security*. Springer, 2016.

[14] Jack Kelly and William Knottenbelt. Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of BuildSys*, pages 55–64, 2015.

[15] J Zico Kolter and Matthew J Johnson. Redd: A public data set for energy disaggregation research. In *Workshop on data mining applications in sustainability (SIGKDD), San Diego*, volume 25, pages 59–62, 2011.

[16] 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*, pages 1–6, 2018.

[17] Nicole Lindsey. Smart Devices Leaking Data to Tech Giants Raises New Iot Privacy Issues. https://www.cpomagazine.com/data-privacy/smart-devices-leaking-data-to-tech-giants-raises-new-iot-privacy-issues/, Oct 2019.

[18] TBRC Business Research Pvt Ltd. Virtual Private Network Market Focuses On Providing Fast and Uninterrupted Connectivity. https://www.einnews.com/pr_news/559887757/virtual-private-network-vpn-market-focuses-on-providing-fast-and-uninterrupted-connectivity.

[19] TBRC Business Research Pvt Ltd. Using a Vpn to Protect Your Privacy. https://www.smarthome.com/blogs/tips-tricks/using-a-vpn-to-protect-your-privacy, Jan 2022.

[20] Kim Martin. Do Vpns Sell Your Data to Third-parties? https://thenextweb.com/news/be-cautious-free-vpns-are-selling-your-data-to-3rd-parties, Oct 21th 2021.

[21] Matthews Correlation Coefficient. https://en.wikipedia.org/wiki/Matthews%_correlation%_coefficient.

[22] OpenVPN. Openvpn Solutions for Secure Iot Communication. https://openvpn.net/for/iot-communications/, Jan 29 2022.

[23] 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.

[24] Nathan Resnick. Be Cautious Free Vpns Are Selling Your Data to Third Parties. https://thenextweb.com/news/be-cautious-free-vpns-are-selling-your-data-to-3rd-parties, Oct 21th 2021.

[25] Adi Robertson. A Vpn can Stop Internet Companies from Selling Your Data — But It's Not a Magic Bullet. https://www.theverge.com/2017/3/25/15056290/vpn-isp-internet-privacy-security-fcc-repeal, 2017.

[26] 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.

[27] Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Classifying iot devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, 2018.

[28] Statista. Internet of Things Connected Devices Installed base Worldwide from 2015 to 2025 (in billions). https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/, 2016.

[29] trafficspy. TtrafficSpy. https://github.com/cyber-physical-systems/trafficspy, Aug 29 2022.

[30] Rahmadi Trimananda, Janus Varmarken, Athina Markopoulou, and Brian Demsky. Pingpong: Packet-level signatures for smart home device events. In *NDSS*, 2019.

[31] Keyang Yu, Qi Li, Dong Chen, Mohammad Rahman, and Shiqiang Wang. Privacyguard: Enhancing smart home user privacy. In *IPSN*, pages 62–76, 2021.

[32] 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.

[33] 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.