IOTHound: Environment-agnostic Device Identification and Monitoring

IoT Conference 2020

Prashant Anantharaman,[1*] Liwei Song,[2*]
Ioannis Agadakos,[3] Gabriela Ciocarlie,[3] Bogdan Copos,[4]
Ulf Lindqvist[3] and Michael Locasto[3]
[1]Dartmouth College [2]Princeton University
[3]SRI International [4]Google Inc.

Overview

- **Introduction**
- IoTHound: System Design
- Evaluation
- Conclusions

Introduction

- Researchers have found several flaws in IoT devices that can give complete control to an attacker
- Attackers have exploited Smart Home devices running Bluetooth, Zigbee, and Wi-Fi
- How do we find out if one of our devices have been exploited in a generalizable way?

Researchers Allege 'Systemic' Privacy, Security Flaws in Popular IoT Devices

Author:
Lindsey O'Donnell

January 29, 2019
/ 8:00 am

Report: Smart bulbs have a major security problem

by **Brandon Vigliarolo** in **Security**
on February 5, 2020, 7:54 AM PST

Many Philips Hue smart light bulbs have a firmware flaw that leads hackers into an entire network, Check Point Research found.

IoTHound: Device Identification and Monitoring

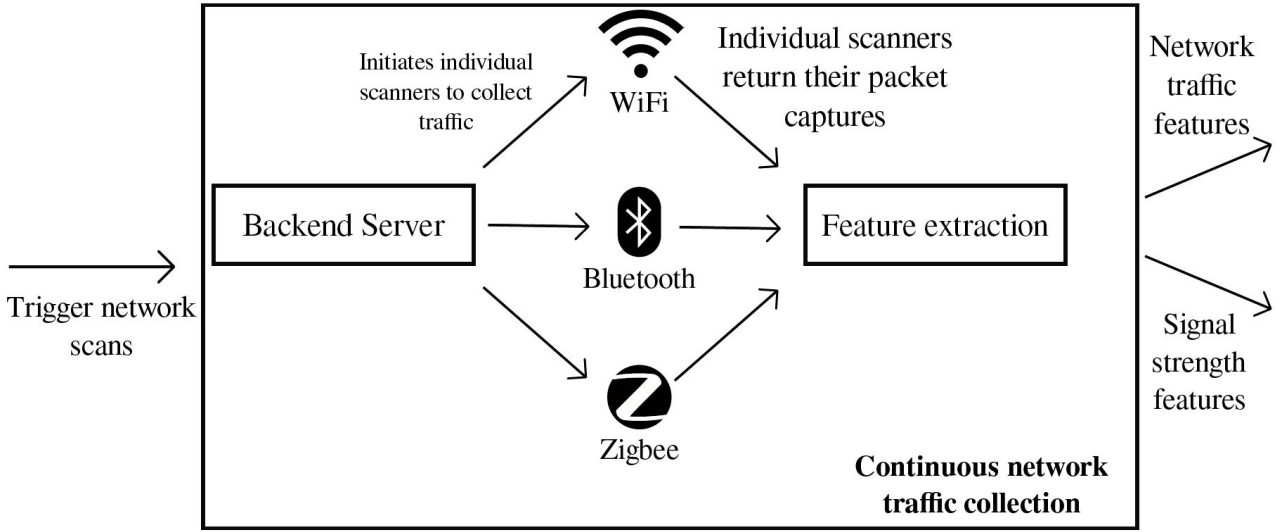
- We identify device types using network traffic across Zigbee, Bluetooth, and Wi-Fi
  - IoT devices do not change their communication patterns often
  - We used unsupervised clustering to analyze properties of the network
- We detected anomalies in network traffic by observing clusters over time
- We consider also physical properties of the devices (RSSI)
- To evaluate our Type Identification component:
  - We built a testbed with 21 commercial off-the-shelf IoT devices
  - We used a dataset publicly made available by Alrawi *et al*.
- To evaluate our Anomaly Detection component:
  - We used a known exploit against a Philips Hub and compared it against its normal operations
  - We built a testbed with 7 Raspberry Pis; we simulated various attack scenarios

Overview

- Introduction
- **IoTHound: System Design**
- Evaluation
- Conclusions

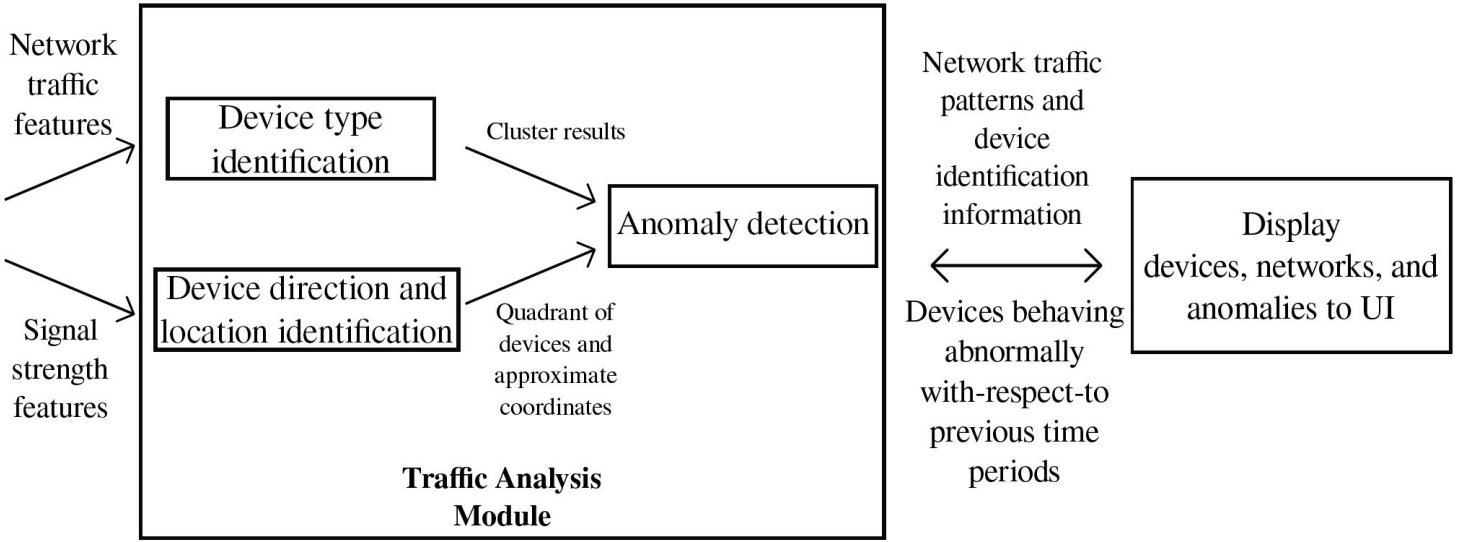
System Overview

- We used a Sniffing device for Zigbee, while tapping into the Android phone for Bluetooth, and the router for Wi-Fi/LAN traffic
- We extracted RSSI features to help with direction estimation, and extracted network traffic features for our device type identification and anomaly detection components



System Overview (Continued)

- The direction estimation and device type identification components send their cluster centroid values to the anomaly detection module
- We also built a UI for users to display devices and their types, network flows, and anomalies



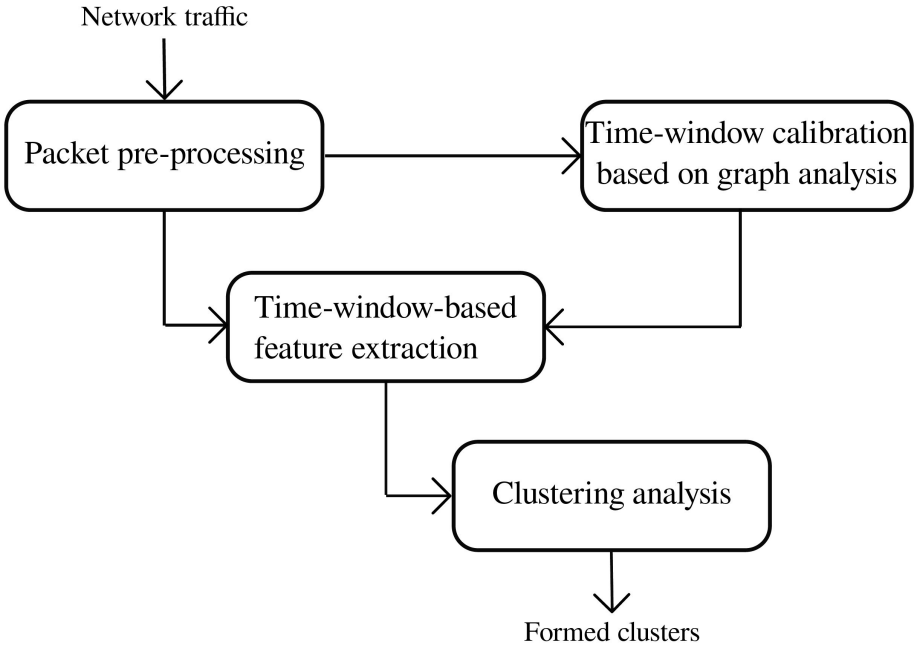
Device type identification

Extract traffic feature vectors in a time-window:

- Features include inter-packet time, payload length, packet length, payload entropy, packet entropy
- Compute statistical values: mean, std, min, max, quartiles Q1, Q2, Q3

Calibrate the time-window value by differentiating between central and end-point devices

HDBSCAN clustering analysis on extracted feature vectors



Device direction identification

- With a router with *N* antennas, we have *N(N+1)/2* possible pairs of antenna
- We find the differential signal strength received in milliwatts as:

$$\delta P_{Ai} P_{Aj} = P_{Ai} - P_{Aj}$$

- We trained an SVM using an *rbf* kernel with the ${}^{N}C_{2}$ values for differential signal strength
- We trained two models using eight smartphones placed in the directions from the routers
  - Four directions from the router
  - Eight directions from the router

Anomaly detection

- After a time period, we store the cluster centroids generated by the device identification component
- Each device may have multiple clusters and centroids labeled against them
- Across time periods, we test if these centroids:
  - Deviate by more than a margin,
  - Are not present in Time period 2, or
  - New clusters are identified in Time period 2 that were not present in Time period 1.

Overview

- Introduction
- IoTHound: System Design
- **Evaluation**
- Conclusions

Evaluation Overview

We try to answer the following questions.

- Is our clustering technique able to cluster different devices of the same type into the same cluster?
- Is our technique generalizable for other datasets?
- Can we detect the direction and direction changes of devices effectively?
- Can we effectively detect various anomalous scenarios?

Device type identification

- Collect Wi-Fi traffics for 7 hours
- Achieve average clustering accuracy above 99%
- Each device has the clustering accuracy above 94%
- Similar success on clustering Zigbee and Bluetooth traffics with 97% and 90% average accuracy

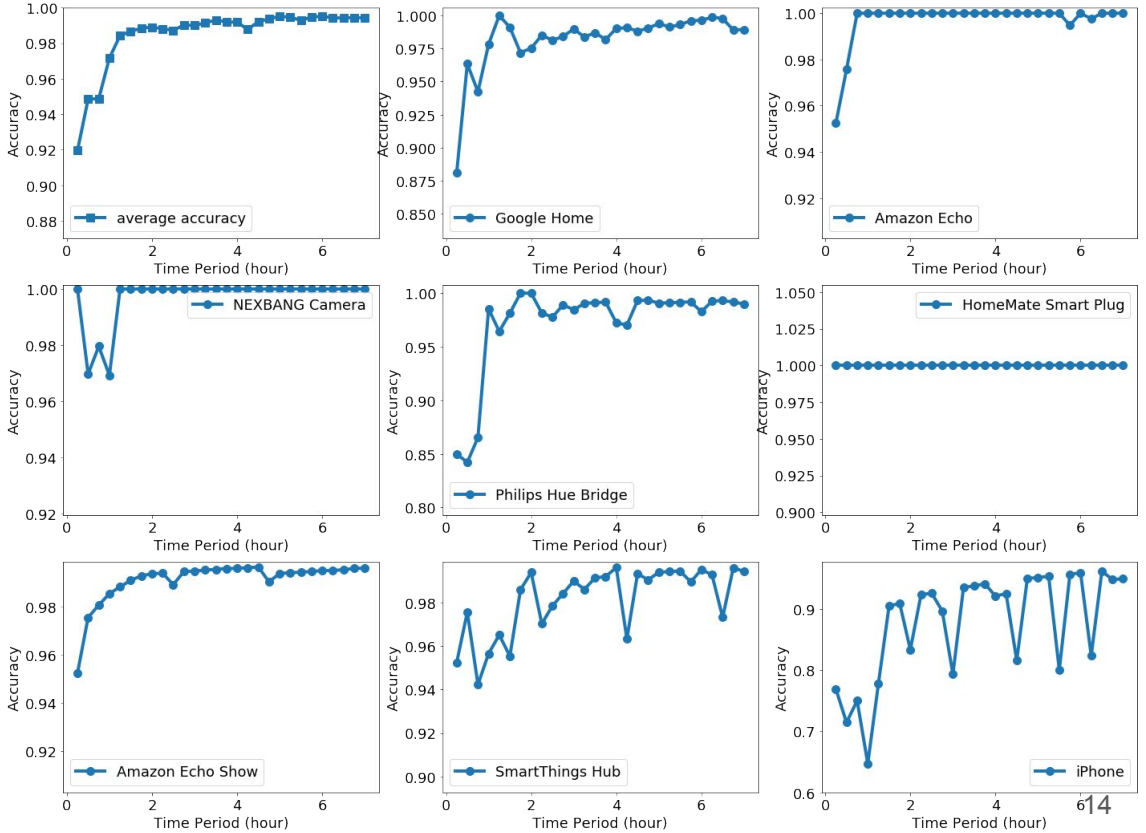
Table 1: Wi-Fi device-type identification performance. IoTHound achieves an average clustering accuracy of 99.42% and all devices have clustering accuracy above 94%.

Device Type	Number of Packets	Number of Examples	Clustering Accuracy
Google Home[a]	545086	1042	98.94%
Amazon Echo	253392	521	100.00%
NEXBANG Camera	1246	458	100.00%
Philips Hue Bridge	9954	498	99.00%
HomeMate Smart Plug	4192	521	100.00%
Amazon Echo Show	650880	521	99.62%
SmartThings Hub	14793	521	99.42%
iPhone	3614	59	94.92%

[a] We used two Google Home devices during the experiments.

Device type identification (Continued)

- Clustering performance is improved with more collected traffic data
- For tested Wi-Fi devices, having 2 hours (or more) of traffics results in enough high accuracy



Device type identification (Continued)

Our proposed method also works well on a public IoT dataset provided by Alrawi et al. (IEEE S&P'19)

Table 2: Device-type identification performance for a testbed by Alrawi et al. [1]. We selected ten devices from various manufacturers to show our results.

IoT Device	Day 1		Day 2		Day 3	
	Number of examples	Clustering accuracy	Number of examples	Clustering accuracy	Number of examples	Clustering accuracy
Wink 2 Hub	1542	98.51%	1543	98.77%	1540	99.61%
Philips Hue Bridge	1542	100%	1543	100%	1543	100%
Sonos Speaker	1540	99.61%	1543	100%	1543	99.87%
Amazon Echo	1542	98.57%	1543	98.64%	1543	99.48%
Ring Doorbell	1542	100%	1543	99.87%	1129	99.73%
Logitech Circle Camera	1542	100%	1543	99.55%	1543	80.88%
WeMo Crockpot	401	98.25%	399	86.22%	392	76.79%
August Doorbell Cam	1539	90.05%	1542	98.18%	1499	99.00%
Chamberlain Garage Opener	754	86.87%	734	89.51%	730	90.14%
Samsung SmartTV	1519	97.50%	1519	99.01%	184	79.35%

Device direction identification

- Collected data for 2 days and 1 day in two different office spaces under normal operation
- Results deteriorate drastically with less data

Table 3: Accuracy of the SVM direction classification using SSD RSSI measurements.

Testbed	Distance	Accuracy (4 Directions)	Accuracy (8 Directions)	Training points
Testbed 1	2 meters	95%	81%	5000
Testbed 2	2 meters	85%	52%	1500

Anomaly detection

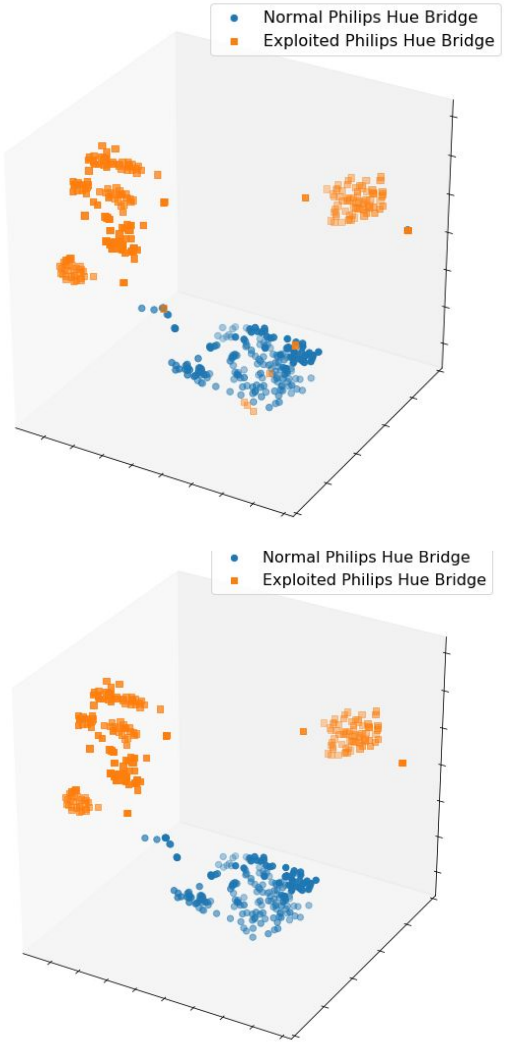
- Scenario 1 was the base scenario. Scenario 2 used a cryptocurrency miner on RPi-7
- Scenario 3 ran a dissimilar set of protocols on RPi-1, RPi-2, RPi-4, RPi-6, and RPi-7
- Scenario 4 ran IRC bots on devices 3 and 6 and an HTTP client on RPi-6

Table 4: For each Raspberry Pi, we computed the average distance of the centroids for the current and previous scenario, and we marked the abnormalities in bold.

Scenarios	RPi-1	RPi-2	RPi-3	RPi-4	RPi-5	RPi-6	RPi-7
Scenario 1 → Scenario 2	0.61	0.61	0.61	0.61	0.28	0.28	**2.07**
Scenario 2 → Scenario 3	**11.19**	**1.81**	0.65	**9.59**	0.49	**22.93**	**4.83**
Scenario 3 → Scenario 4	0.79	0.79	**9.70**	**6.60**	0.42	**21.42**	1.40

Anomaly detection (continued)

- The ground truth (top) and clusters detected by IoTHound (bottom)
- There is quite a significant deviation between the normal operation and the operations with exploited with an additional HTTP client



Overview

- Introduction
- IoTHound: System Design
- Evaluation
- **Conclusions**

Conclusions

- We presented IoTHound: a tool to monitor IoT devices in a network-agnostic way
- We used the cyber and physical properties of IoT devices to categorize and monitor them
- We demonstrated the effectiveness of our direction estimation, device identification, and anomaly detection components

Thank You!
Questions?

- Prashant Anantharaman: pa@cs.dartmouth.edu
- Liwei Song : liweis@princeton.edu
- Ioannis Agadakos : ioannis.agadakos@sri.com
- Gabriela Ciocarlie : gfc2101@caa.columbia.edu

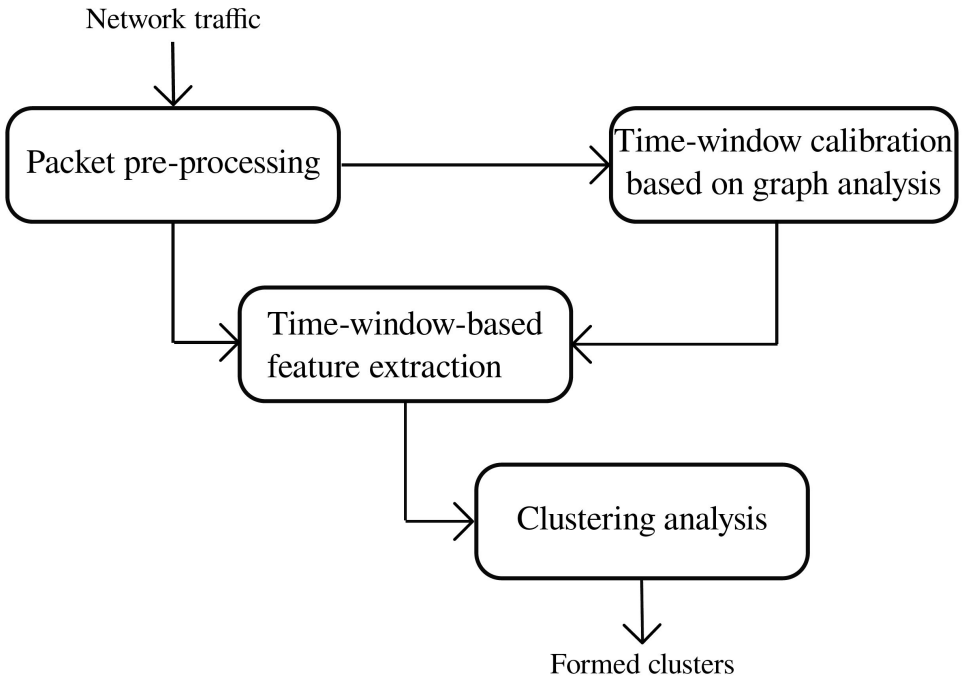
Device type identification

Features:

- Inter-packet arrival time
- Total packet entropy
- Payload entropy
- Total packet length
- Payload length

We computed the seven statistical features for all of these features.

- We segmented the network traffic into time-windows before performing clustering.



Device type identification

Table 5: Zigbee device-type identification performance. IoTHound achieves an average accuracy of 97.91%, and all devices, except for SmartThings devices, achieve clustering accuracy higher than 96%.

Device Type	Number of Packets	Number of Examples	Clustering Accuracy
SmartThings Hub	1478	973	96.20%
Philips Hue Bridge	30029	2869	99.97%
Philips Hue Bloom[b]	50359	5737	99.84%
SmartThings Arrival Sensor	633	626	100.00%
SmartThings Waterleak Sensor	43	43	100.00%
SmartThings Multipurpose Sensor	79	66	0.00%
SmartThings Motion Sensor	91	64	37.50%
SmartThings Outlet	968	308	77.27%

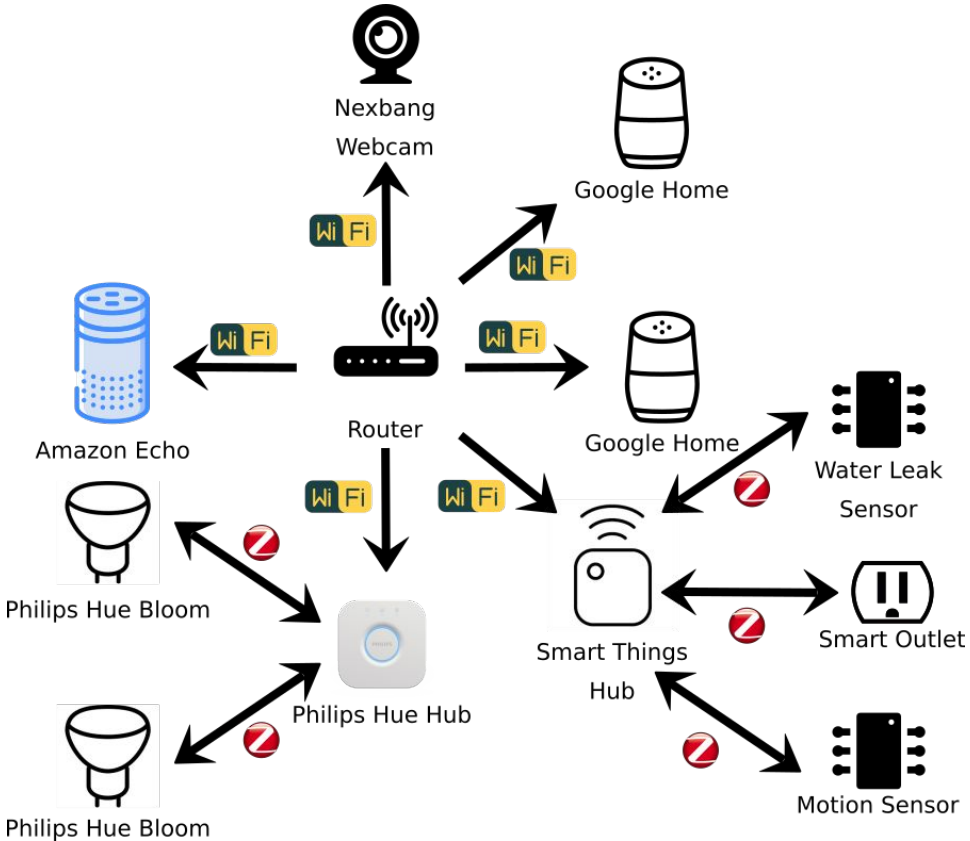
[b] We have two Philips Hue Bloom devices during the experiments.

Device type identification

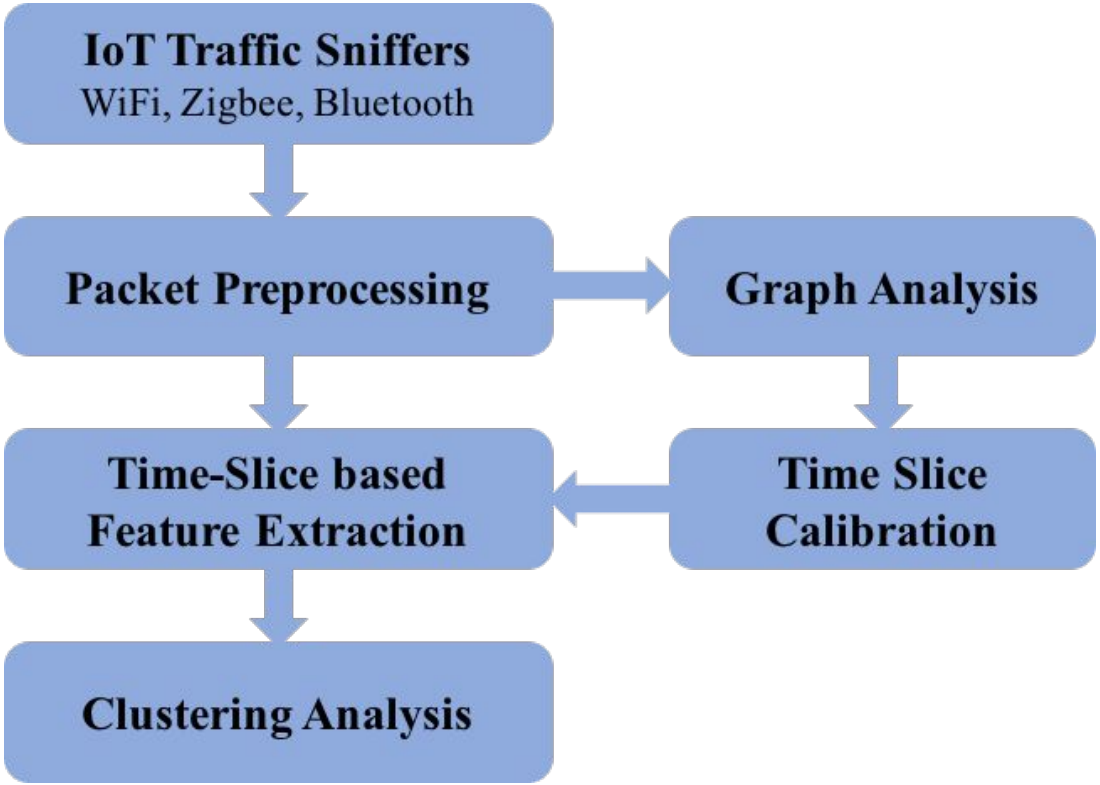
Table 6: Bluetooth device-type identification performance. IoTHound achieves an average clustering accuracy of 90.96%, all devices have clustering accuracy higher than 71%.

Device Type	Number of Packets	Number of Examples	Clustering Accuracy
Android Phone (Google Pixel)	48389	160	95.63%
Amazon Echo Speaker	83	11	72.73%
Beats Headphone	802	46	86.96%
Google Home Speaker	424	46	86.96%
GE Smart Plug	516	9	100.00%
Moto Smart Watch (Moto 360)	698	46	91.30%
Fitbit Tracker (Charge HR)	635	14	71.43%

Experimental Setup



Overall Approach



IoT Traffic Sniffers

- **WiFi Sniffer:** tcpdump
- **Zigbee Sniffer:** Texas Instruments CC2531emk USB dongle
- **Bluetooth Sniffer:** Bluetooth HCI snoop log on Android Phone

Packet Preprocessing

- Record both sender and receiver’s addresses.
- Use the sender’s IP/MAC address as the label for each packet.
- Extract timing and volume-based information for each packet: timestamp, payload length, packet length, payload entropy, and packet entropy, where entropy is calculated as following...

Time-Slice Based Feature Extraction

- Divided the whole traffic as many small slots with fixed time length (discuss how to choose slot time size in the calibration part)
- Inside the time slot, we extract one feature vector for each sender’s address based on statistics of packet timing and volume-based information.
- More specifically, we get a feature vector with size of 35 by calculating the mean, standard deviation, minimum, maximum, three quartiles for inter-packet time, payload length, payload entropy, packet length, and packet entropy.

Clustering Analysis

- Adopt hdbscan to perform clustering analysis.
- Density-based method, only need to set min_cluster_size for hdbscan function.
-

Graph Analysis

- Construct a directed graph based on the network traffic
- Identify router in WiFi traffic, Hubs in Zigbee traffic and phone in Bluetooth traffic

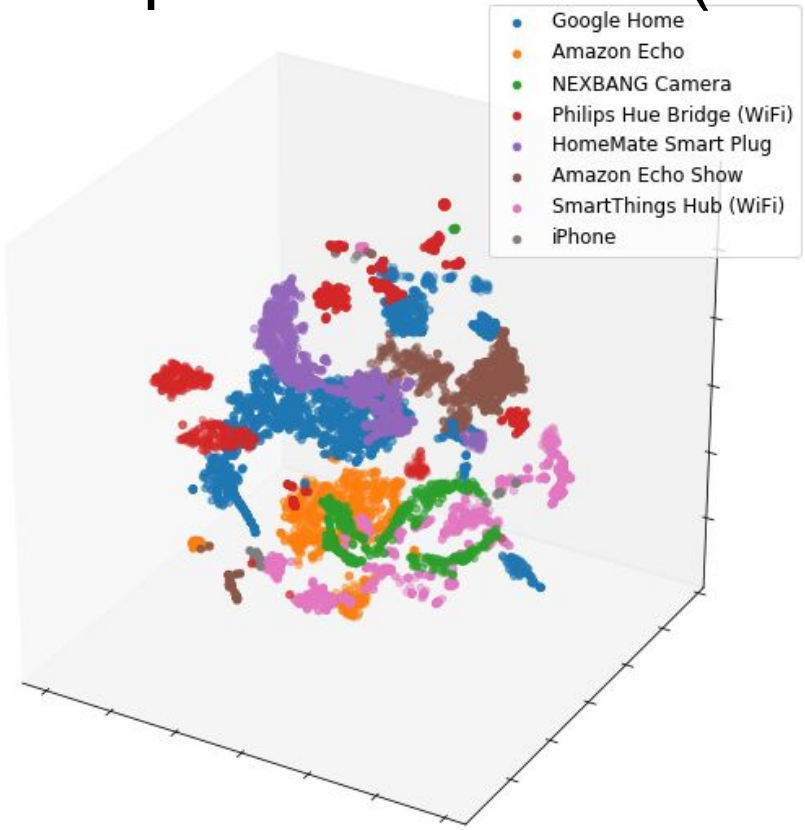
Time Slice Calibration

- We focus the differentiation between router (hub) and non-router (non-hub) traffic
- Repeat feature extraction and clustering analysis with different slot size values
- Choose the slot size with highest accuracy

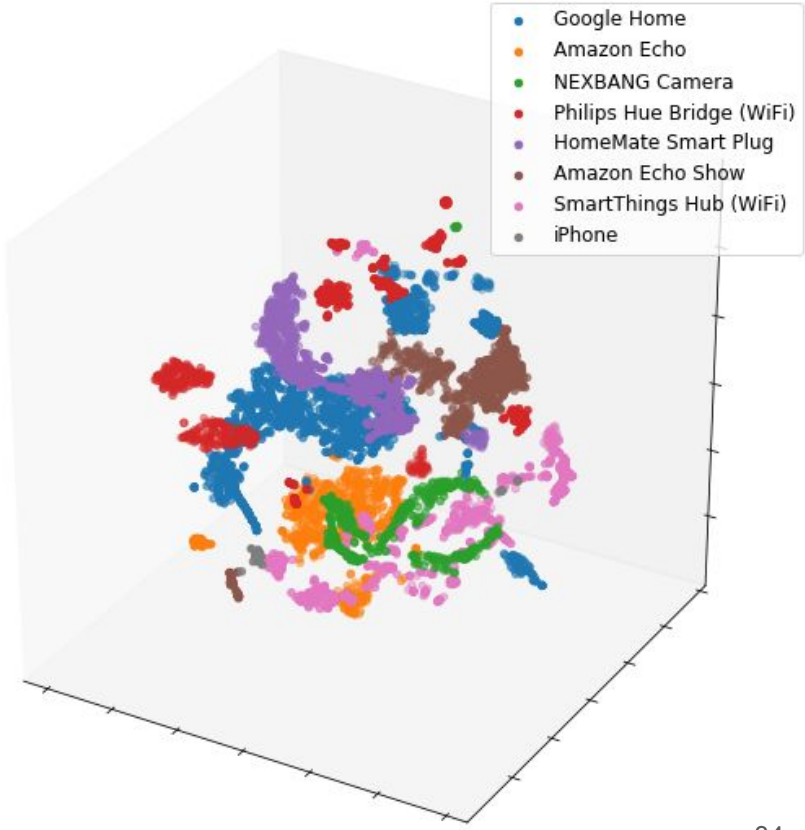
Experiment Setup (WiFi)

- **WiFi devices:** two Google Home devices, Amazon Echo, Amazon Echo Show, Philips Hue Bridge, SmartThings Hub, NEXBANG Camera, HomeMate Smart Plug, iPhone
- 7-hour WiFi traffic collection
- **Cluster accuracy**
- **Performance over time period**
- **Calibration effectiveness**

Experiment Result (WiFi)



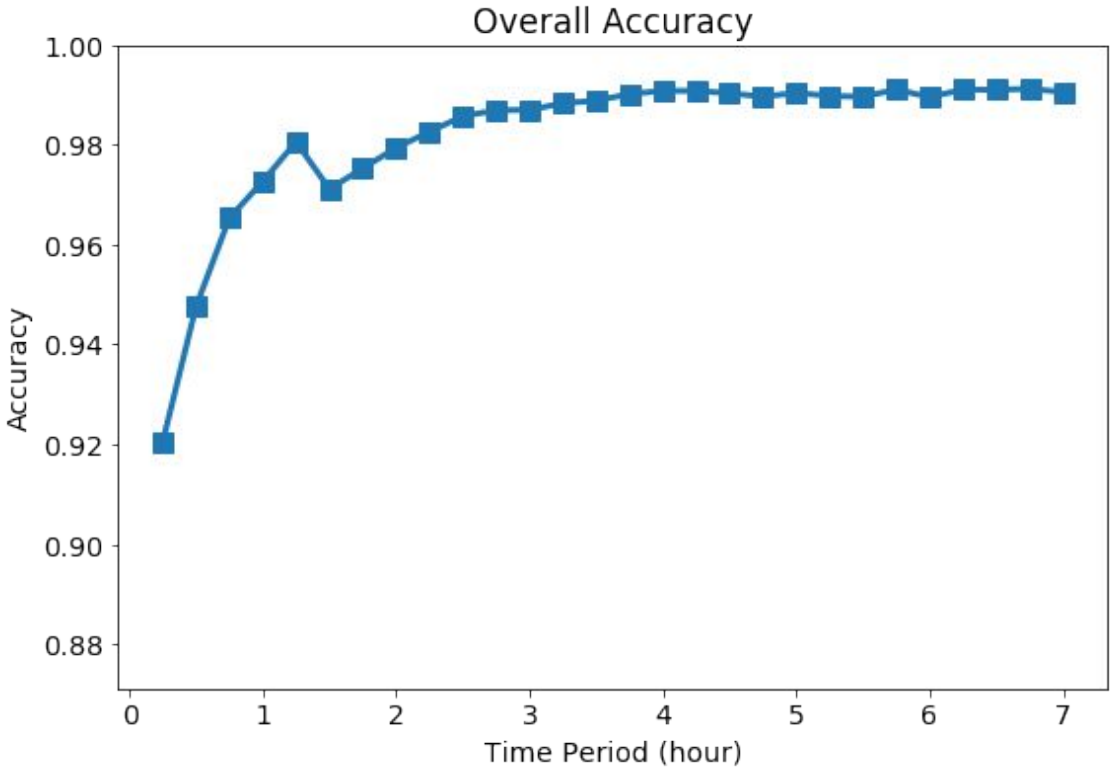
Ground Truth

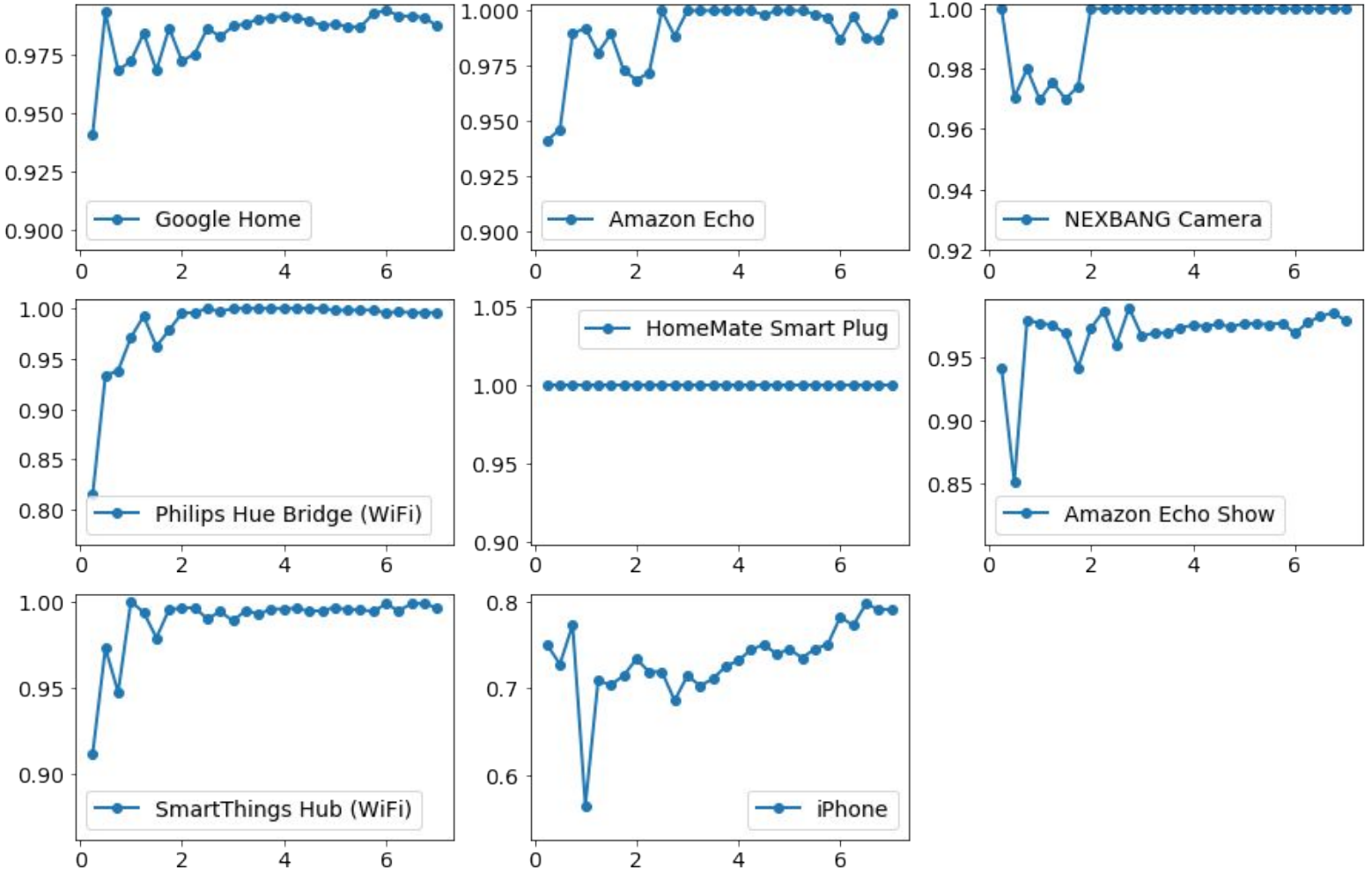


Cluster Result

Device type	Number of packets	Number of examples	Cluster accuracy
Google Home	545086	1656	98.73%
Amazon Echo	253392	828	99.88%
Echo Show	650880	828	97.83%
Philips Hue Bridge	9954	725	99.59%
SmartThings Hub	14793	828	99.64%
NEXBANG Camera	1246	464	100.00%
Smart Plug	4192	828	100.00%
iPhone	3614	62	79.03%
Total	1483157	6219	99.05%

Performance over time period



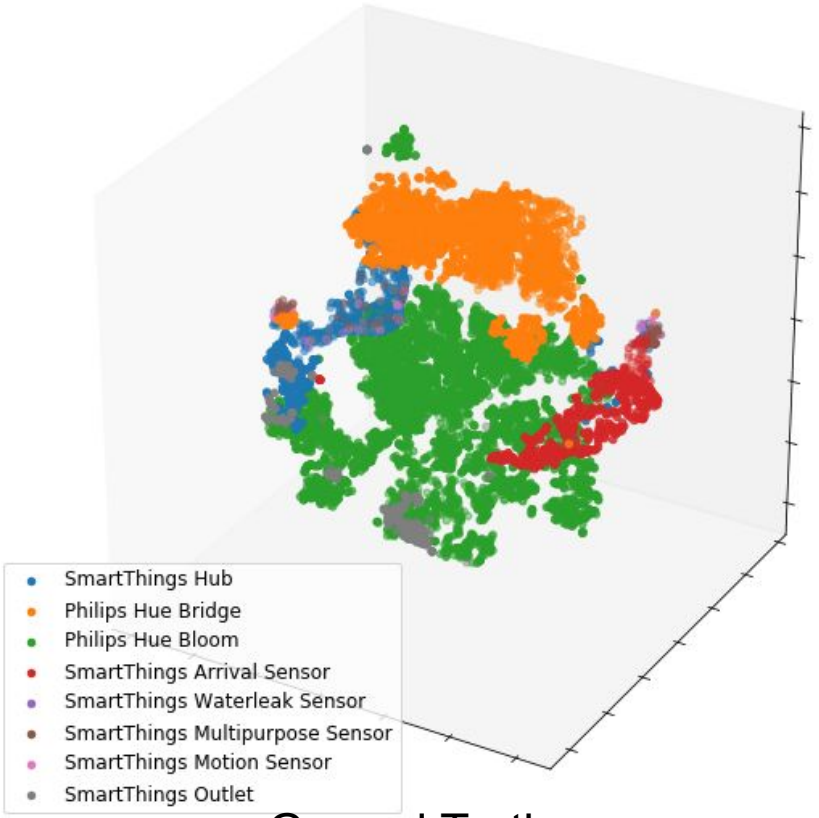

Google Home
Amazon Echo
NEXBANG Camera
Philips Hue Bridge (WiFi)
HomeMate Smart Plug
Amazon Echo Show
SmartThings Hub (WiFi)
iPhone

Effectiveness of Calibration Process

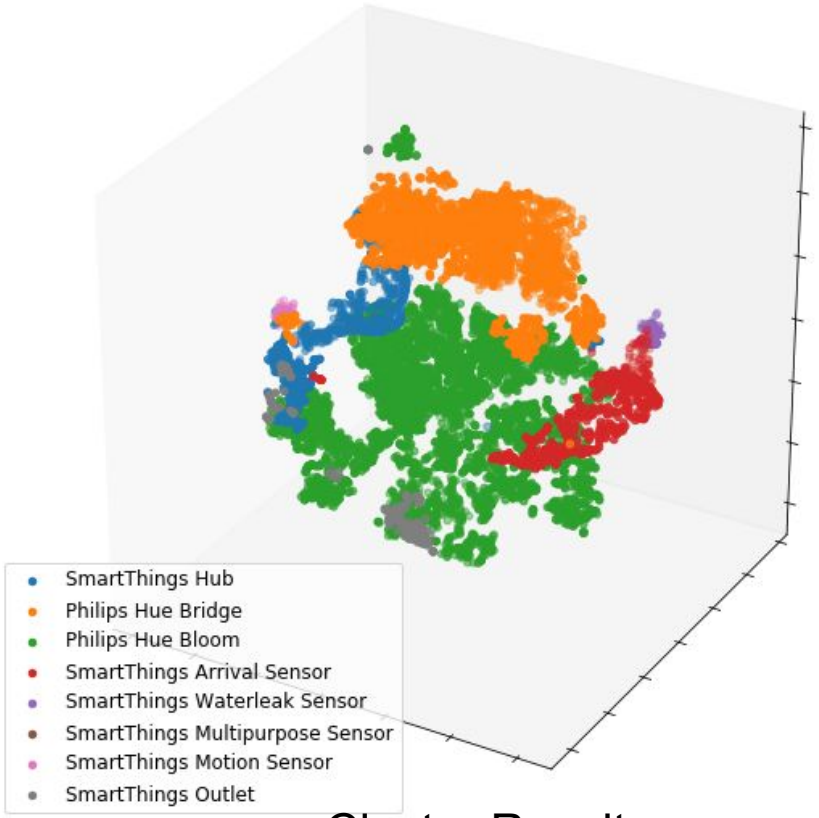
Experiment Setup (Zigbee)

- **Zigbee devices:** SmartThings Hub, Philips Hue Bridge, two Philips Hue Blooms, SmartThings Arrival Sensor, SmartThings Waterleak Sensor, SmartThings Multipurpose Sensor, SmartThings Motion Sensor, SmartThings Outlet.
- 7-hour Zigbee traffic collection

Experiment Result (Zigbee)



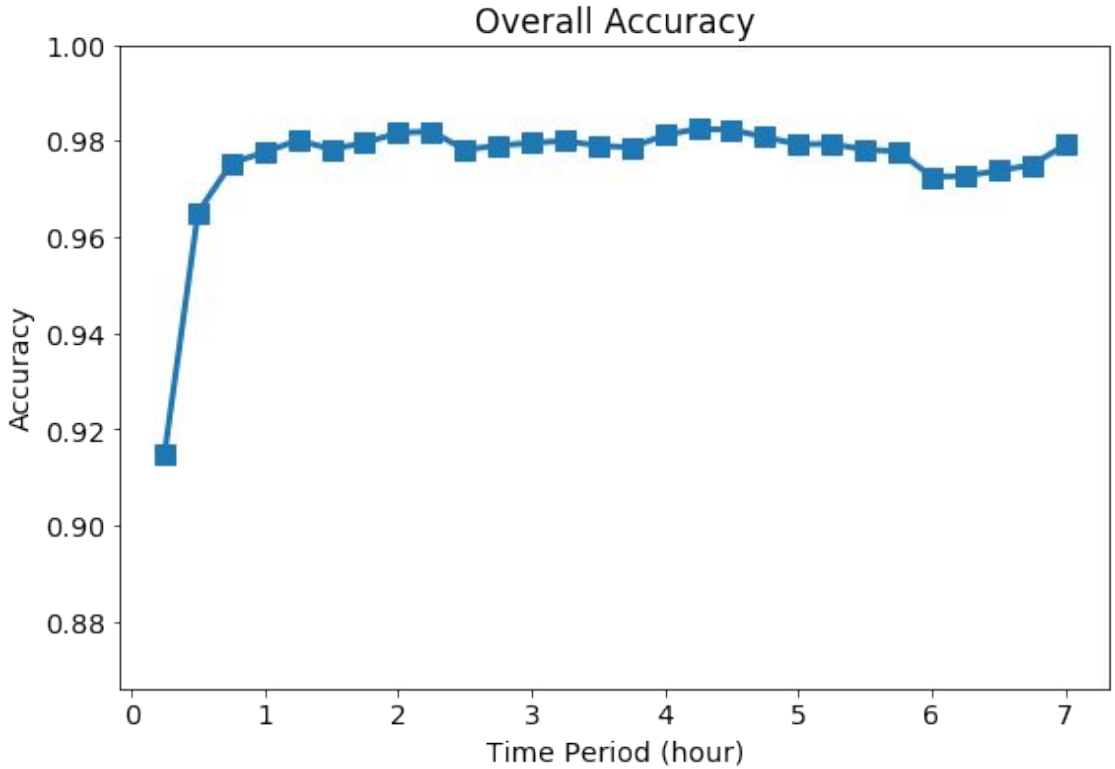
Ground Truth

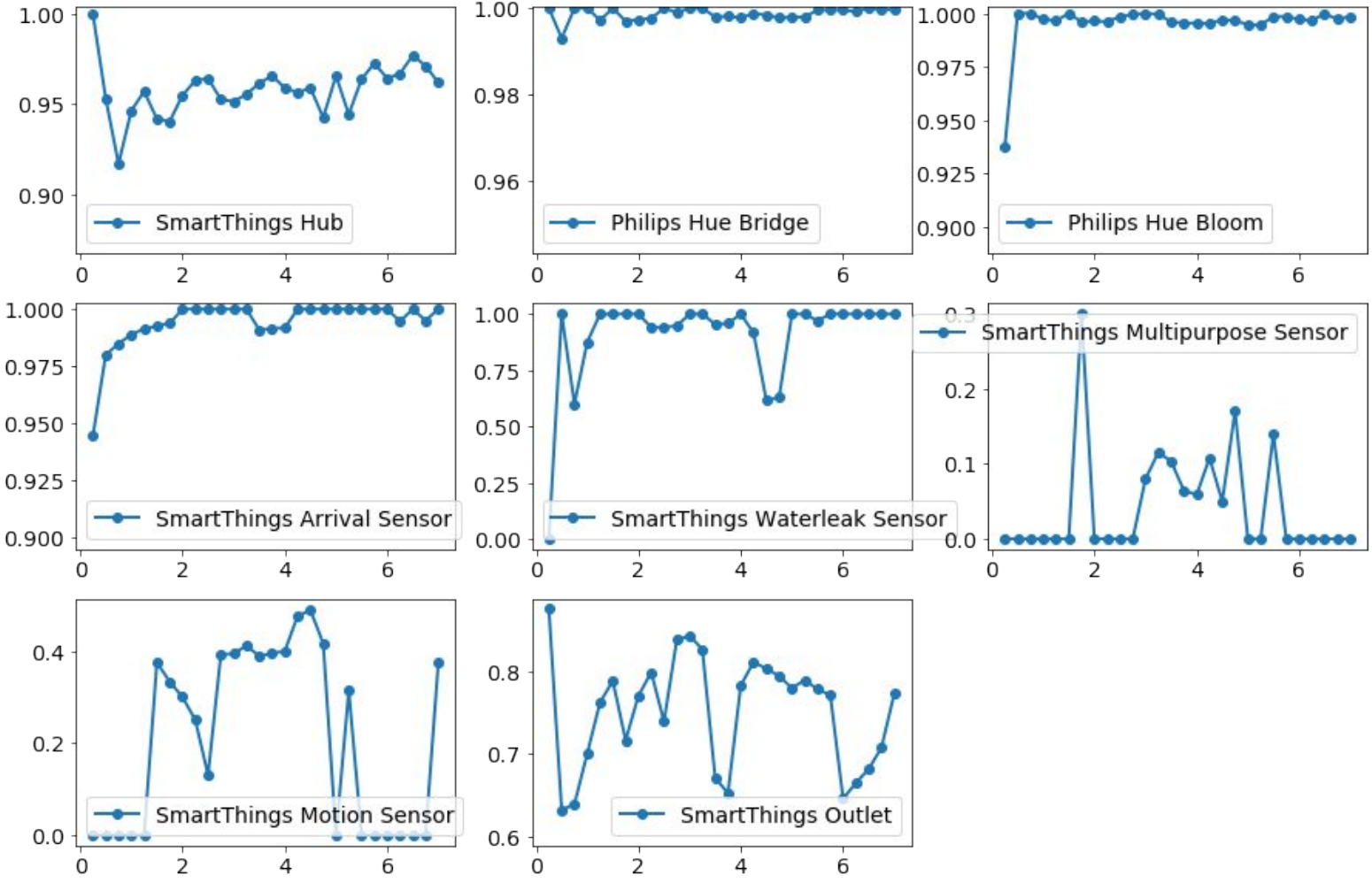


Cluster Result

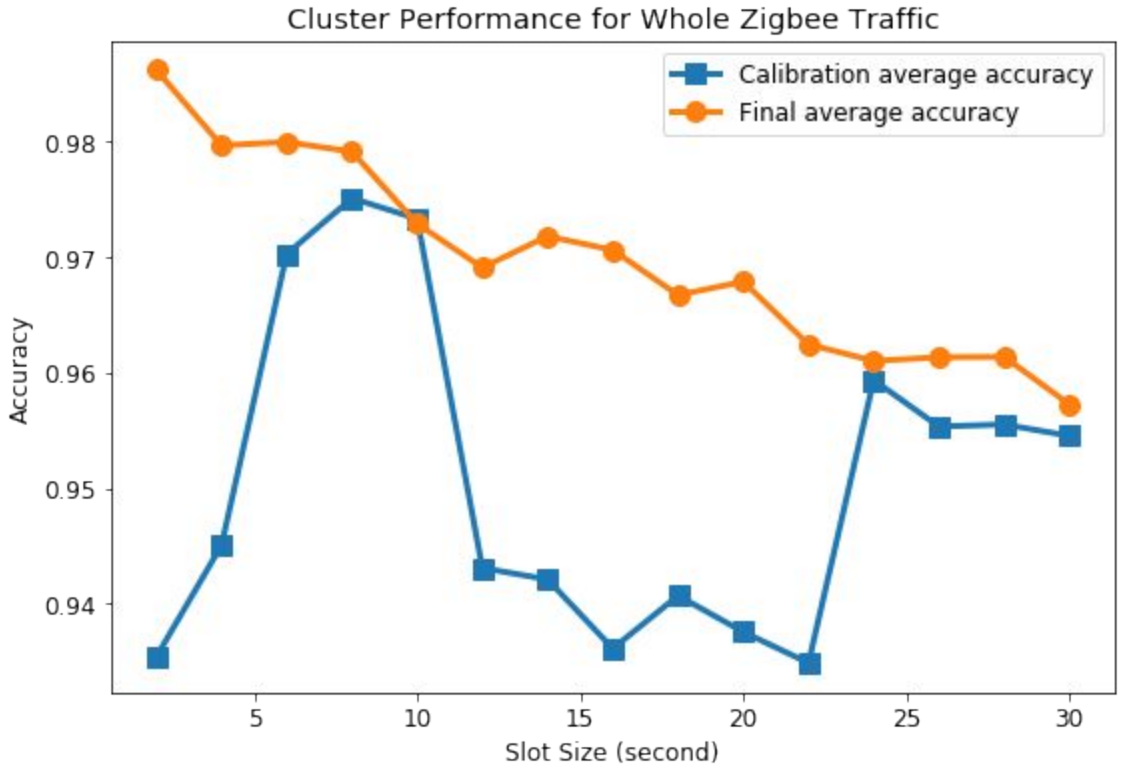
Device type	Number of packets	Number of samples	Cluster accuracy
SmartThings Hub	1478	973	96.20%
Philips Hue Bridge	30029	2869	99.97%
Philips Hue Bloom	50359	5737	99.84%
Arrival Sensor	633	626	100.00%
Waterleak Sensor	43	43	100.00%
Multipurpose Sensor	79	66	0.00%
Motion Sensor	91	64	37.50%
Outlet	968	308	77.27%
Total	83680	10686	99.05%

Performance over time period




SmartThings Hub
Philips Hue Bridge
Philips Hue Bloom
SmartThings Arrival Sensor
SmartThings Waterleak Sensor
SmartThings Multipurpose Sensor
SmartThings Motion Sensor
SmartThings Outlet

Effectiveness of Calibration Process



Experimental Setup (WiFi Traffic)

- **Device list:** two Google Home devices, Amazon Echo, Philips Hue, SmartThings Hub, NEXBANG Smart Camera, iPhone.
- **WiFi sniffer:** tcpdump tool
- We label the TCP packets according to their source ip addresses.

Experimental Setup (Zigbee Traffic)

- **Device list:** Philips Hue bridge, two Philips Hue blooms, Smartthings Hub, Smart outlet, Water leak sensor, Motion sensor
- **Zigbee sniffer:** Texas Instruments CC2531emk USB dongle (https://github.com/andrewdodd/ccsniffpiper)
- We ignore IEEE 802.15.4 (not Zigbee) packets and broadcast packets.
- Similarly, the Zigbee packet is labeled by its source MAC address

Features used

The following features are extracted for both WiFi and Zigbee packets:

- Packet inter-arrival time (mean, std, min, max, the first quartile, the third quartile)
- Payload length (mean, std, min, max, the first quartile, the third quartile)
- Total packet length - payload length (mean, std, min, max, the first quartile, the third quartile)
- Entropy of payload- (https://arxiv.org/pdf/1804.03852.pdf) (mean, std, min, max, the first quartile, the third quartile)

Approach - feature extraction

- Step 1: Read from the pcap and extract useful packet information (timestamp, payload length, packet length, entropy) using scapy.
- Step 2: Split whole data into many small time slots based on the timestamp information.
- Step 3: For each time slot, find all packets sent from the same device, and calculate a feature vector mentioned before.

Approach - clustering analysis

- **HDBSAN** (Hierarchical Density-Based Spatial Clustering of Applications with Noise) is adopted.
- Compared to traditional DBSCAN, HDBSCAN has the advantages of finding clusters of varying densities and being more robust to parameter selection.
- HDBSCAN will label some points as noisy examples, which do not belong to any clusters. For the noisy point, we find its nearest neighbour and label it to the same cluster as the neighbour.
- To compare clustering results with ground truth labels and compute the accuracy, we relabel each cluster as its majority samples' ground truth label.
- There could be multiple clusters for the same device, as the traffic for certain devices are separated that way in the graph. This is relabelled based on the majority in that cluster.

Approach - slot time calibration

- There are also UDP packets collected from WiFi traffic, we separate them as two groups -- router packet and non-router packet.
- We perform feature extraction and clustering analysis on those UDP packets with varying slot time values, and finally choose the value with highest cluster accuracy.

Experiments

- We collected WiFi and Zigbee traffic for around 1 hour, get a 1.5GB pcap file for WiFi packets and 4.1 MB for Zigbee packets.
- The best slot time value during our calibration process is 25 seconds. We anticipate that this value will vary from network to network.

Clustering Results for WiFi Traffic

Device Type	No. of packets	No. of data instances*	Cluster Accuracy
Google Home	93390	133	100%
Amazon Echo	35023	126	95.24%
Philips Hue Hub	7812	120	99.17%
Smartthing Hub	2901	131	100%
Smart Webcam	10494	29	100%
iPhone	19119	102	86.27%
Overall Accuracy			**96.72%**

*A data instance is the feature vector from one sending device within a single time slice.

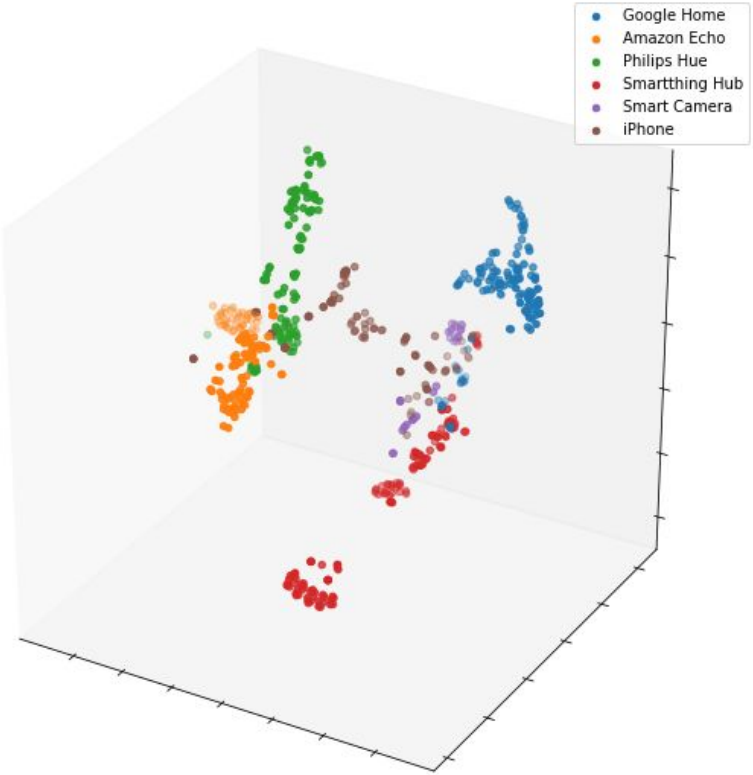
Clustering Results for Zigbee Traffic

Device Type	No. of packets	No. of data instances	Cluster Accuracy
Smartthing Hub	344	81	93.83%
Philips Hue Hub	8524	140	100%
Philips Hue Bloom	12458	140	100%
Water Leak Sensor	163	46	89.13%
Smart Outlet	1097	77	87.01%
Motion Sensor	73	20	60%
Overall Accuracy			**94.44%**

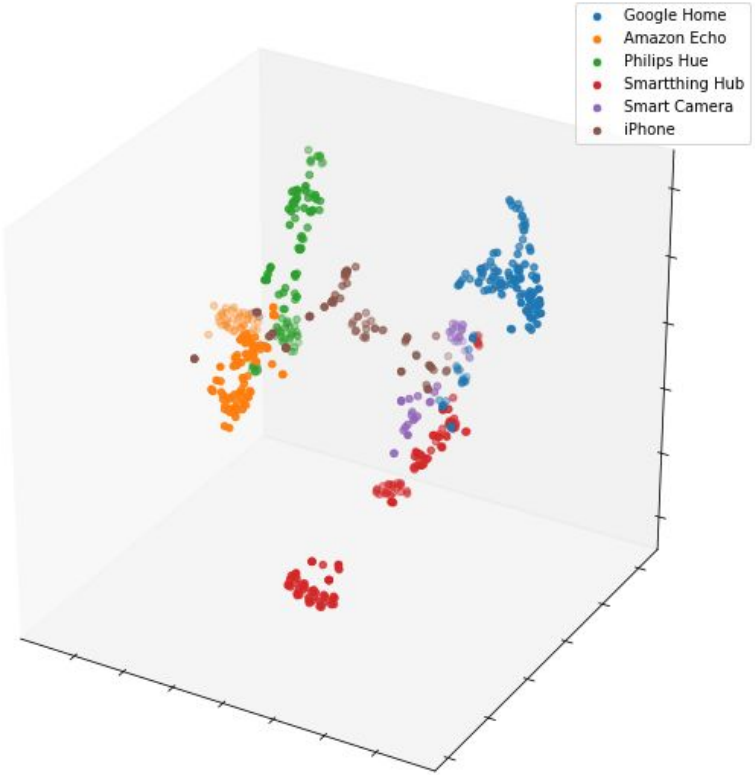
Understanding the results

- We use TSNE method in scikit-learn package to visualize high dimensional data (reduce our high dimensional data to 3 dimensions).
- Our traffic clustering works for device type fingerprinting, two Google Home devices (Philips Hue Blooms) have nearly same feature patterns, so we use same label for them.

Clustering Results - WiFi Traffic

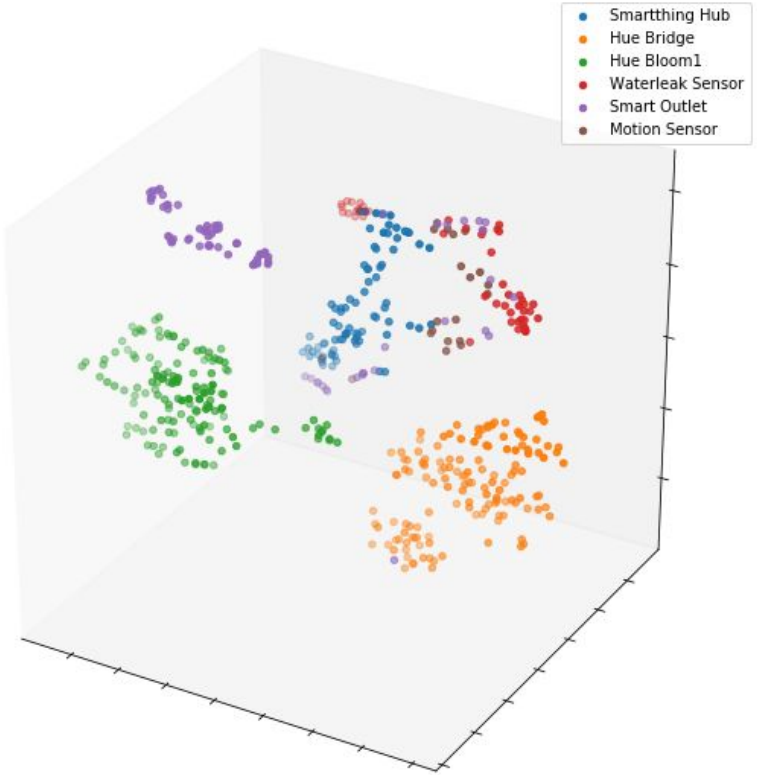


Ground Truth

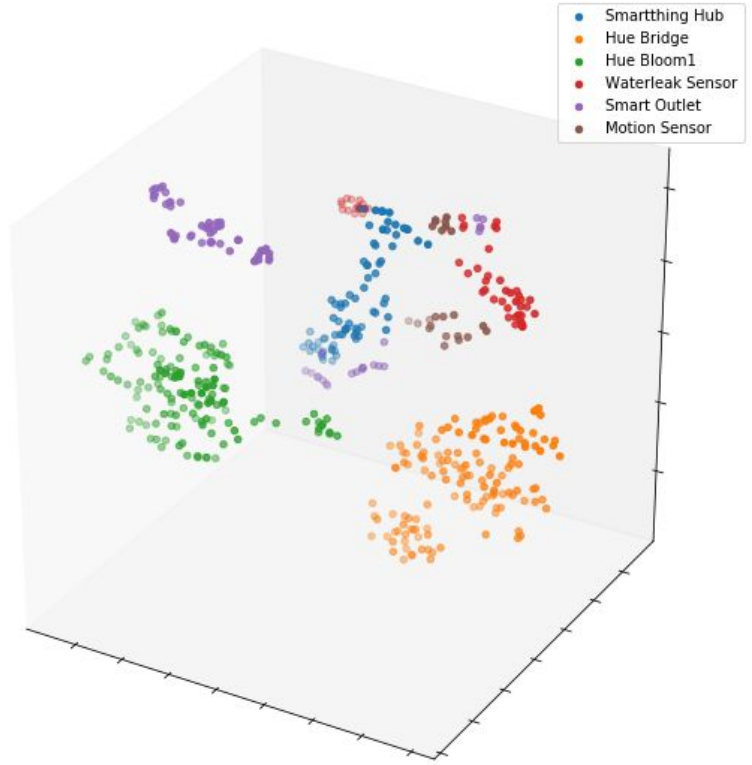


HDBSCAN Cluster

Clustering Results - Zigbee Traffic



Ground Truth



HDBSCAN Cluster

Next Steps

- Extract the same features for BLE traffic.
- Classifying devices as sensors, actuators and activators based on network traffic.
- Another experiment with longer time period.