# Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics

Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath and Vijay Sivaraman

**Abstract**—The Internet of Things (IoT) is being hailed as the next wave revolutionizing our society, and smart homes, enterprises, and cities are increasingly being equipped with a plethora of IoT devices. Yet, operators of such smart environments may not even be fully aware of their IoT assets, let alone whether each IoT device is functioning properly safe from cyber-attacks. In this paper, we address this challenge by developing a robust framework for IoT device classification using traffic characteristics obtained at the network level. Our contributions are fourfold. First, we instrument a smart environment with 28 different IoT devices spanning cameras, lights, plugs, motion sensors, appliances and health-monitors. We collect and synthesize traffic traces from this infrastructure for a period of 6 months, a subset of which we release as open data for the community to use. Second, we present insights into the underlying network traffic characteristics using statistical attributes such as activity cycles, port numbers, signalling patterns and cipher suites. Third, we develop a multi-stage machine learning based classification algorithm and demonstrate its ability to identify specific IoT devices with over 99% accuracy based on their network activity. Finally, we discuss the trade-offs between cost, speed, and performance involved in deploying the classification framework in real-time. Our study paves the way for operators of smart environments to monitor their IoT assets for presence, functionality, and cyber-security without requiring any specialized devices or protocols.

**Index Terms**—IoT, network characteristics, device visibility, classification, machine learning.

## 1 Introduction

The number of devices connecting to the Internet is ballooning, ushering in the era of the "Internet of Things" (IoT). IoT refers to the tens of billions of low cost devices that communicate with each other and with remote servers on the Internet autonomously. It comprises everyday objects such as lights, cameras, motion sensors, door locks, thermostats, power switches and household appliances, with shipments projected to reach nearly 20 billion by 2020 [1]. Thousands of IoT devices are expected to find their way in homes, enterprises, campuses and cities of the near future, engendering "smart" environments benefiting our society and our lives.

The proliferation of IoT, however, creates an important problem. Operators of smart environments can find it difficult to determine what IoT devices are connected to their network and further to ascertain whether each device is functioning normally. This is mainly attributed to the task of managing assets in an organization, which is typically distributed across different departments. For example, in a local council, lighting sensors may be installed by the facilities team, sewage and garbage sensors by the sanitation department and surveillance cameras by the local police division. Coordinating across various departments to obtain an inventory of IoT assets is time consuming, onerous and error-prone, making it nearly impossible to know precisely what IoT devices are operating on the network at any point in time. Obtaining "visibility" into IoT devices in a timely manner is of paramount importance to the operator, who is tasked with ensuring that devices are in appropriate network security segments, are provisioned for requisite quality of service, and can be quarantined rapidly when breached. The importance of visibility is emphasized in Cisco's most recent IoT security report [2], and further highlighted by two recent events: sensors of a fishtank that compromised a casino in Jul 2017 [3], and attacks on a University campus network from its own vending machines in Feb 2017 [4]. In both cases, network segmentation could have potentially prevented the attack and better visibility would have allowed rapid quarantining to limit the damage of the cyber-attack on the enterprise network.

One would expect that devices can be identified by their MAC address and DHCP negotiation. However, this faces several challenges: (a) IoT device manufacturers typically use NICs supplied by third-party vendors, and hence the Organizationally Unique Identifier (OUI) prefix of the MAC address may not convey any information about the IoT device; (b) MAC addresses can be spoofed by malicious devices; (c) many IoT devices do not set the Host Name option in their DHCP requests [5]; indeed we found that about half the IoT devices we studied do not reveal their host names, as shown in Table 1; (d) even when the IoT device exposes its host name it may not always be meaningful (e.g. WBP-EE4C for Withings baby monitor in Table 1); and lastly (e) these host names can be changed by the user (e.g. the HP printer can be given an arbitrary host name). For these reasons, relying on DHCP infrastructure is not a viable solution to correctly identify devices at scale.


- *A. Sivanathan, F. Loi, H. Habibi Gharakheili, C. Wijenayake, and V. Sivaraman are with the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, NSW 2052, Australia (e-mails: a.sivanathan@unsw.edu.au, f.loi@student.unsw.edu.au, h.habibi@unsw.edu.au, c.wijenayake@unsw.edu.au, vijay@unsw.edu.au).*
- *A. Radford is with Cisco Systems, Sydney, Australia (e-mail: aradford@cisco.com).*
- *A. Vishwanath is with IBM Research, Australia (e-mail: arvishwa@au1.ibm.com).*

TABLE 1
MAC address and DHCP host name of IoT devices used in our testbed.

| IoT device | MAC address | OUI | DHCP host name |
|---|---|---|---|
| Amazon Echo | 44:65:0d:56:cc:d3 | Amazon Technologies Inc. | |
| August Doorbell Cam | e0:76:d0:3f:00:ae | AMPAK Technology, Inc. | |
| Awair air quality monitor | 70:88:6b:10:0f:c6 | | Awair-4594 |
| Belkin Camera | b4:75:0e:ec:e5:a9 | Belkin International Inc. | NetCamHD |
| Belkin Motion Sensor | ec:1a:59:83:28:11 | Belkin International Inc. | |
| Belkin Switch | ec:1a:59:79:f4:89 | Belkin International Inc. | |
| Blipcare BP Meter | 74:6a:89:00:2e:25 | Rezolt Corporation | |
| Canary Camera | 7c:70:bc:5d:5e:dc | IEEE Registration Authority | Ambarella/C100F1615229 |
| Dropcam | 30:8c:fb:2f:e4:b2 | Dropcam | |
| Google Chromecast | 6c:ad:f8:5e:e4:61 | AzureWave Technology Inc. | Chromecast |
| Hello Barbie | 28:c2:dd:ff:a5:2d | AzureWave Technology Inc. | Barbie-A52D |
| HP Printer | 70:5a:0f:e4:9b:c0 | Hewlett Packard | HPE49BC0 |
| iHome PowerPlug | 74:c6:3b:29:d7:1d | AzureWave Technology Inc. | hap-29D71D |
| LiFX Bulb | d0:73:d5:01:83:08 | LIFI LABS MANAGEMENT PTY LTD | LIFX Bulb |
| NEST Smoke Sensor | 18:b4:30:25:be:e4 | Nest Labs Inc. | |
| Netatmo Camera | 70:ee:50:18:34:43 | Netatmo | netatmo-welcome-183443 |
| Netatmo Weather station | 70:ee:50:03:b8:ac | Netatmo | |
| Phillip Hue Lightbulb | 00:17:88:2b:9a:25 | Philips Lighting BV | Philips-hue |
| Pixstart photo frame | e0:76:d0:33:bb:85 | AMPAK Technology, Inc. | |
| Ring Door Bell | 88:4a:ea:31:66:9d | Texas Instruments | |
| Samsung Smart Cam | 00:16:6c:ab:6b:88 | Samsung Electronics Co.,Ltd | |
| Smart Things | d0:52:a8:00:67:5e | Physical Graph Corporation | SmartThings |
| TP-Link Camera | f4:f2:6d:93:51:f1 | TP-LINK TECHNOLOGIES CO.,LTD. | Little Cam |
| TP-Link Plug | 50:c7:bf:00:56:39 | TP-LINK TECHNOLOGIES CO.,LTD. | HS110(US) |
| Triby Speaker | 18:b7:9e:02:20:44 | Invoxia | |
| Withings Baby Monitor | 00:24:e4:10:ee:4c | Withings | WBP-EE4C |
| Withings Scale | 00:24:e4:1b:6f:96 | Withings | |
| Withings sleep sensor | 00:24:e4:20:28:c6 | Withings | WSD-28C6 |

In this paper, we address the above problem by developing a robust framework that classifies each IoT device separately in addition to one class of non-IoT devices with high accuracy using statistical attributes derived from network traffic characteristics. Qualitatively, most IoT devices are expected to send short bursts of data sporadically. Quantitatively, our preliminary work in [6] was one of the first attempts to study how much traffic IoT devices send in a burst and how long they idle between activities. We also evaluated how much signaling they perform (e.g. domain lookups using DNS or time synchronization using NTP) in comparison to the data traffic they generate. This paper significantly expands on our prior work by employing a more comprehensive set of attributes on trace data captured over a much longer duration (of 6 months) from a test-bed comprising 28 different IoT devices.

There is no doubt that it is becoming increasingly important to understand the nature of IoT traffic. Doing so helps contain unnecessary multicast/broadcast traffic, reducing the impact they have on other applications. It also enables operators of smart cities and enterprises to dimension their networks for appropriate performance levels in terms of reliability, loss, and latency needed by environmental, health, or safety applications. However, the most compelling reason for characterizing IoT traffic is to detect and mitigate cyber-security attacks. It is widely known that IoT devices are by their nature and design easy to infiltrate [7], [8], [9], [10], [11], [12]. New stories are emerging of how IoT devices have been compromised and used to launch large-scale attacks [13]. The large heterogeneity in IoT devices has led researchers to propose network-level security mechanisms that analyze traffic patterns to identify attacks (see [14] and our recent work [15]); success of these approaches relies on a good understanding of what "normal" IoT traffic profile looks like.

Our primary focus in this work is to establish a machine learning framework based on various network traffic characteristics to identify and classify the default (i.e. baseline) behavior of IoT devices on a network. Such a framework can potentially be used in the future to detect anomalous behavior of IoT devices (potentially due to cyber-attacks), and such anomaly detection schemes are beyond the scope of this paper. This paper fills an important gap in the literature relating to classification of IoT devices based on their network traffic characteristics. Our contributions are:

1) We instrument a living lab with 28 IoT devices emulating a smart environment. The devices include cameras, lights, plugs, motion sensors, appliances and health-monitors. We collect and synthesize data from this environment for a period of 6 months. A subset of our data is made available for the research community to use.
2) We identify key statistical attributes such as activity cycles, port numbers, signaling patterns and cipher suites, and use them to give insights into the underlying network traffic characteristics.
3) We develop a multi-stage machine learning based classification algorithm and demonstrate its ability to identify specific IoT devices with over 99% accuracy based on their network behavior.
4) We evaluate the deployment of the classification framework in real-time, by examining the trade-offs between costs, speed, and accuracy of the classifier.

The rest of this paper is organized as follows: §2 describes relevant prior work. We present our IoT setup and data traces in §3, and in §4 characterize traffic attributes of the various IoT devices. In §5 we propose a machine learning based multi-stage device classification method and evaluate its performance, followed by a discussion on the real-time operation of the proposed system in §6. The paper is concluded in §7.

## 2 Related Work

There is a large body of work characterizing general Internet traffic [16], [17], [18], [19]. These prior works largely focus on application detection (e.g. Web browsing, Gaming, Mail, Skype VoIP, Peer-to-Peer, etc.). However, studies focusing on characterizing IoT traffic (also referred to as machine-to-machine or M2M traffic) are still in their infancy.

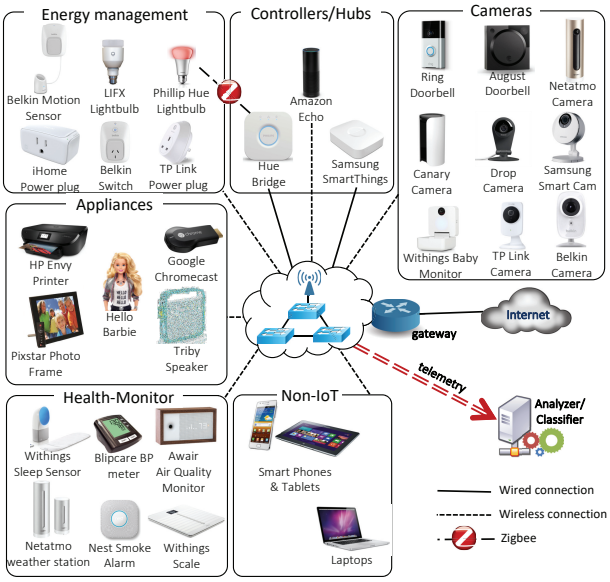



Fig. 1. Testbed architecture showing connected 28 different IoT devices along with several non-IoT devices, and telemetry collected across the infrastructure is fed to our classification models.

**Analysis of Empirical Traces:** The work in [20] is one of the first large-scale studies to delve into the nature of M2M traffic. It is motivated by the need to understand whether M2M traffic imposes new challenges for the design and management of cellular networks. The work uses a traffic trace spanning one week from a tier-1 cellular network operator and compares M2M traffic with traditional smartphone traffic from a number of different perspectives – temporal variations, mobility, network performance, and so on. The study informs network operators to be cognizant of these factors when managing their networks.

In [21], the authors note that the amount of traffic generated by a single M2M device is likely to be small, but the total traffic generated by hundreds or thousands of M2M devices would be substantial. These observations are to some extent corroborated by [22], [23], which note that a remote patient monitoring application is expected to generate about 0.35 MB per day and smart meters roughly 0.07 MB per day.

**Aggregated Traffic Model:** A Coupled Markov Modulated Poisson Processes framework to capture the behavior of a single machine-type communication as well as the collective behavior of tens of thousands of M2M devices is proposed in [24]. The complexity of the CMMPP framework is shown to grow linearly with the number of M2M devices, rendering it effective for large-scale synthesis of M2M traffic.

In [25], the authors show that it is possible to split the (traffic) state of an M2M device into three generic categories, namely periodic update, event driven, and payload exchange, and a number of modelling strategies that use these states are developed. An illustration of model fitting is shown via a use-case in fleet management comprising 1000 trucks run by a transportation company. The fitting is based on measured M2M traffic from a 2G/3G network. A simple model to estimate the volume of M2M traffic generated in a wireless sensor network enabled connected home is constructed in [26]. Since behavior of sensors is very application specific, the work identifies certain common communication patterns that can be attributed to any sensor device. Using these attributes, four generalized equations are proposed to estimate the volume of traffic generated by a sensor network enabled connected apartment/home.

**Use of Machine Learning:** Various machine-learning-based analytical methods have been proposed in the literature to classify traffic application or identify malwares/botnets for typical computer networks. The work in [27] uses deep learning to classify flow types such as HTTP, SMTP, Telnet, QUIC, Office365, and YouTube by considering six features namely source/destination port number, payload volume, TCP window size, inter-arrival time and direction of traffic that are extracted from the first 20 packets of a flow. The work carried out in [28] suggests that botnets exhibit identifiable traffic patterns that can be classified by considering features such as average time between successive flows, flow duration, inbound/outbound traffic volume, and Fourier transformation over the flow start times. Detection of malicious activity on the network was enhanced in [29] and [30] by combining these flow-level features with packet-level attributes including packet size, byte distribution of payload, inter arrival times of packets and TLS handshake metadata (i.e. cipher suite codes). Further, authors have released an open source *libpcap*-based tool called *Joy* [31] to extract these features from the passive capture of network traffic.

In the context of IoT, [32] uses machine learning to classify a single TCP flow from authorized devices on the network. It employs over 300 attributes (packet-level and flow-level), though the most influential ones are minimum, median and average of packets Time-To-Live (TTL), the ratio of total bytes transmitted and received, total number packets with reset (RST) flag, and the Alexa rank of server.

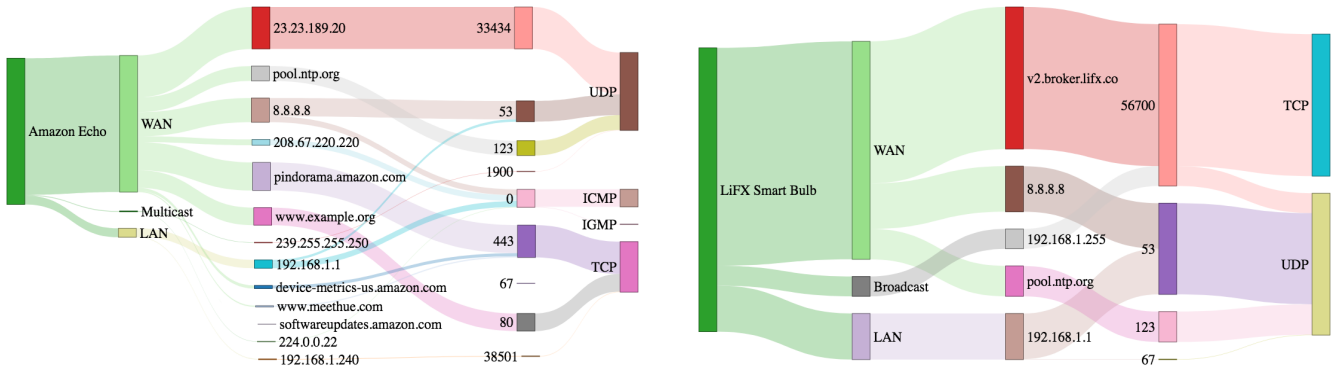
While all the above works make important contributions, they do not undertake fine-grained characterization and classification of IoT devices in a smart environment such as a home, city, campus or enterprise. Furthermore, statistical models are not developed that enable IoT device classification based on their network traffic characteristics. Most importantly, prior works do not make any data set publicly available for the research community to use and build upon. Our work overcomes these shortcomings.

## 3 IoT Traffic Collection and Synthesis

In this section, we describe our smart environment infrastructure for collecting and synthesizing traffic from various IoT devices.

### 3.1 Experimental Test-bed

A real-life architecture of a "smart environment" is depicted in Fig. 1 that serves a wide range of IoT and non-IoT devices over its (wired/wireless) network infrastructure and allows them to communicate with the Internet servers via a gateway. Our lab setup is a specialized implementation of this architecture, housed at our campus facility, comprises one node of TP-Link Archer C7 v2 WiFi access point (representing internal switch) collocated with the Internet gateway. The TP-Link access point, flashed with the OpenWrt firmware release Chaos Calmer (15.05.1, r48532), serves as the gateway to the public Internet. We also installed additional OpenWrt packages on the gateway, namely `tcpdump`




(a) Amazon Echo. (b) LiFX lightbulb.

Fig. 2. Sankey diagram of daily network activity for two representative IoT devices, Amazon Echo and LiFX lightbulb. A clear distinction is observed in terms of their communication patterns, i.e. the servers they talk to, and the port numbers and protocols used for data exchange.

`(4.5.1-4)` for capturing traffic, `bash (4.3.39-1)` for scripting, `block-mount` package for mounting external USB storage on the gateway, `kmod-usb-core` and `kmod-usb-storage (3.18.23-1)` for storing the traffic trace data on the USB storage.

In our lab setup, the WAN interface of the TP-Link access point is connected to the public Internet via the university network, while the IoT devices are connected to the LAN and WLAN interfaces respectively. Our smart environment has a total of 28 unique IoT devices representing different categories along with several non-IoT devices. Here, IoT refers to specific-purpose Internet connected devices (e.g. cameras and smoke sensors), while general-purpose devices (e.g. phones and laptops) fall into the non-IoT category.

The IoT devices include cameras (Nest Dropcam, Samsung SmartCam, Netatmo Welcome, Belkin camera, TP-Link Day Night Cloud camera, Withings Smart Baby Monitor, Canary camera, August door bell, Ring door bell), switches and triggers (iHome, TP-Link Smart Plug, Belkin Wemo Motion Sensor, Belkin Wemo Switch), hubs (Smart Things, Amazon Echo), air quality sensors (NEST Protect smoke alarm, Netatmo Weather station, Awair air quality monitor), electronics (Triby speaker, PIXSTAR Photoframe, HP Printer, Hello barbie, Google Chromecast), healthcare devices (Withings Smart scale, Withings Aura smart sleep sensor, Blipcare blood pressure meter) and light bulbs (Phips Hue and LiFX Smart Bulb). Several non-IoT devices were also connected to the testbed, such as laptops, mobile phones and an Android tablet. The tablet was used to configure the IoT devices as recommended by the respective device manufacturers.

### 3.2 Trace Data

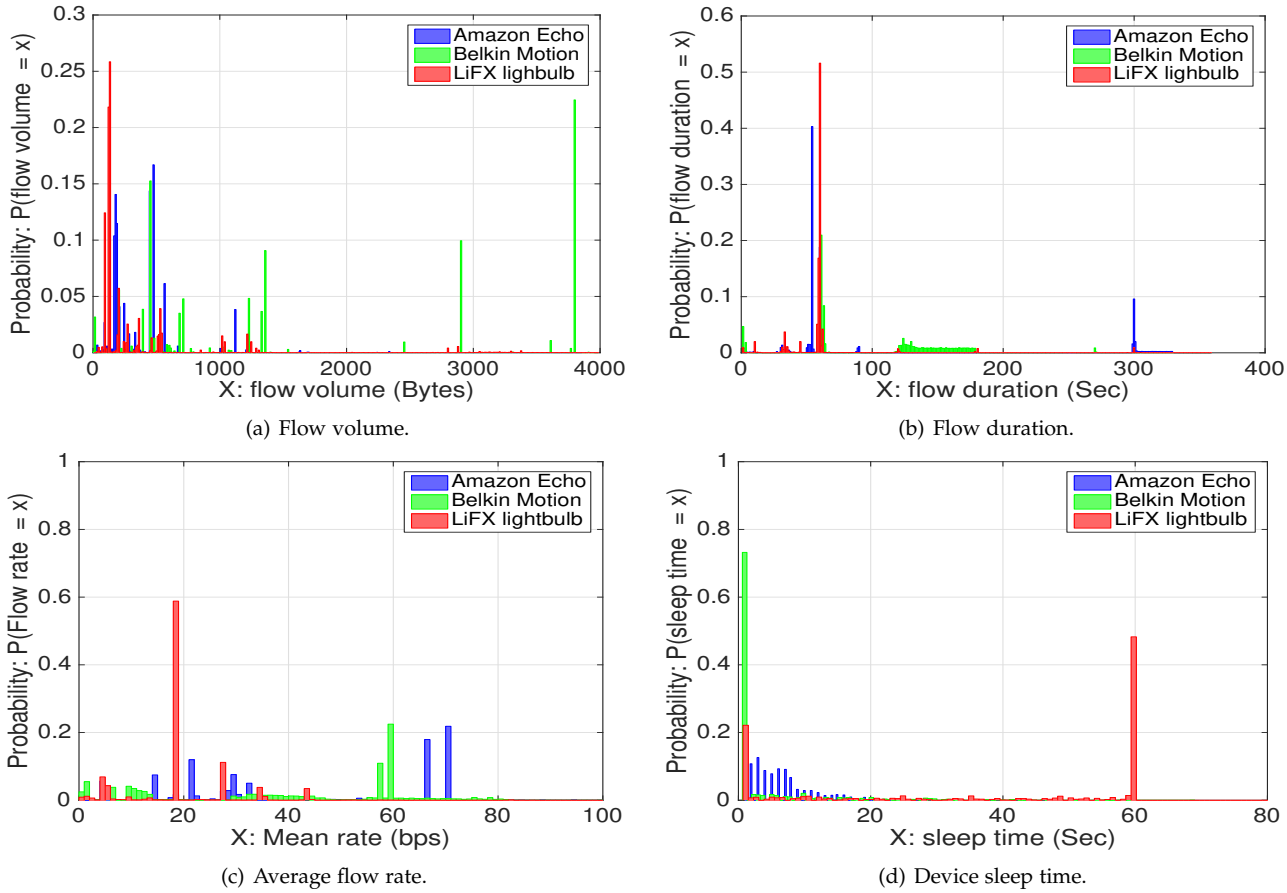
All the traffic on the LAN side was collected using the `tcpdump` tool running on OpenWrt [33]. It is important to have a one-to-one mapping between a physical device and a known MAC address (by virtue of being in the same LAN) or IP address (i.e. without NAT) in the traffic trace. Capturing traffic on the LAN allowed us to use MAC address as the identifier for a device to isolate its traffic from the traffic mix comprising many other devices in the network. We developed a script to automate the process of data collection and storage. The resulting traces were stored as `pcap` files on an external USB hard drive of 1 TB storage attached to the gateway. This setup permitted continuous logging of the traffic across several months.

We started logging the network traffic in our smart environment from 1-Oct-2016 to 13-Apr-2017, i.e. over a period of 26 weeks. The raw trace data contains packet headers and payload information. The process of data collection and storage begins at midnight local time each day using the `Cron` job on OpenWrt. We wrote a monitoring script on the OpenWrt to ensure that data collection/storage was proceeding smoothly. The script checks the processes running on the gateway at 5 second intervals. If the logging process is not running, then the script immediately restarts it, thereby limiting any data loss event to only 5 seconds. To make the trace data publicly available, we set up an Apache server on a virtual machine (VM) in our university data center and wrote a script to periodically transfer the trace data from the previous day, stored on the hard drive, onto the VM. The trace data from two weeks is openly available for download at: http://iotanalytics.unsw.edu.au/. The size of the daily logs varies between 61 MB and 2 GB, with an average of 365 MB.

## 4 IoT Traffic Characterization

We now present our observations using passive packet-level analysis of traffic from 28 IoT devices over the course of 26 weeks. We study a broad range of IoT traffic characteristics including activity patterns (e.g. distribution of volume/times during active/sleep periods), and signalling (e.g. domain names requested, server-side port numbers used and TLS handshake exchanges).

IoT traffic constitutes (i) traffic generated by the devices autonomously – e.g. DNS, NTP, etc. that are unaffected by human interaction, as well as (ii) traffic generated due to users interacting with the devices – e.g. Belkin Wemo sensor responding to detection of movement, Amazon Echo responding to voice commands issued by a user, LiFX lightbulb changing colour and intensity upon user request, Netatmo Welcome camera detecting an occupant and instructing the LiFX light bulb to turn on with a specific colour, and so on. Our dataset well captures these two types of IoT traffic from a lab that represents a living smart environment (i.e. covering periods over which humans are present or absent in the environment).




(a) Flow volume.

(b) Flow duration.

(c) Average flow rate.

(d) Device sleep time.

Fig. 3. Distribution of IoT activity pattern: (a) flow volume, (b) flow duration, (c) average flow rate and (d) device sleep time.

To provide insights into the IoT traffic characteristics, we show in Fig. 2 a Sankey plot of network traffic seen over a 24 hour period for Amazon Echo and LiFX lightbulb. These devices are chosen just for illustrative purposes. Each plot depicts the flow-level information generated by the respective device. Flows are: (a) either unicast or multicast/broadcast, (b) destined to either local hosts (LAN) or Internet servers (WAN), and (c) tied to protocols (TCP, UDP, ICMP or IGMP) and port numbers.

Fig. 2 provides a visual aid depicting the underlying traffic signature exhibited by the two devices. For example, DNS (port number 53) and NTP (port number 123) are used by both Amazon Echo and LiFX lightbulb. While Amazon Echo uses HTTP (port number 80), HTTPS (port number 443) and ICMP (port number 0), LiFX lightbulb does not use any of these applications. Further, each device seems to communicate to a unique port number on a WAN server; TCP 33434 for Amazon Echo and UDP 56700 for LiFX lightbulb, as shown by the top flow in Figures 2(a) and 2(b). Finally, we observe that Amazon Echo accesses a number of domain names including `softwareupdates.amazon.com`, `device-metrics-su.amazon.com`, `example.org`, `pindorama.amazon.com` and `pool.ntp.org`. However, LiFX lightbulb communicates with only two domains, i.e. `v2.broker.lifx.co` and `pool.ntp.org`.

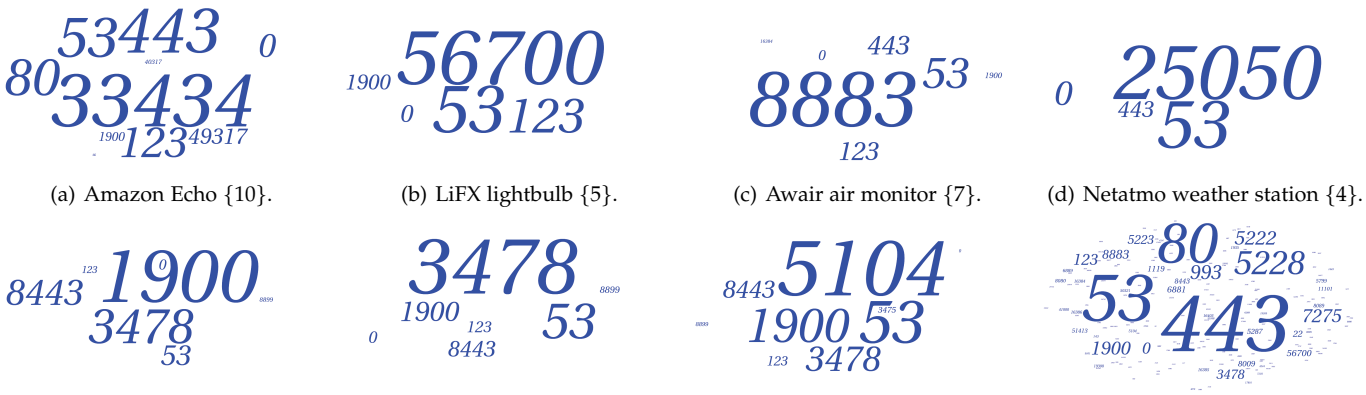
### 4.1 IoT Activity and Volume Pattern

We start with the *activity* pattern of IoT devices that is defined by the properties of their traffic flows. We define four key attributes at a per-flow level to characterize IoT devices based on their network activity: **flow volume** (i.e. sum total of download and upload bytes), **flow duration** (i.e. time between the first and the last packet in a flow), **average flow rate** (i.e. flow volume divided by the flow duration), and **device sleep time** (i.e. time interval over which the IoT device has no active flow).

We plot in Fig. 3 the probability distribution of the above four attributes for a chosen set of IoT devices using the trace data collected over 26 weeks. It can be observed from Fig. 3(a) that each IoT device tends to exchange a small amount of data per flow. For the case of the LiFX lightbulb (depicted by red bars), 26% of flows transfer between [130, 140] bytes and 20% between [120, 130] bytes. The flow volume for the Belkin motion sensor (depicted by green bars) is slightly higher; over 35% of flows transfer between [2800, 3800] bytes. For the Amazon Echo (depicted by blue bars), over 95% of flows transfer less than 1000 bytes. Though we present the flow volume histogram for only a few devices, most of our IoT devices exhibit a similar predictable pattern.

A similar pattern emerges for the flow duration as well. Referring to Fig. 3(b), we note that the flow duration of 53 seconds is seen in more than 40% of flows for Amazon Echo, while a duration of 60 seconds is seen for the LiFX lightbulb and Belkin motion sensor with a probability of 50% and 21% respectively.

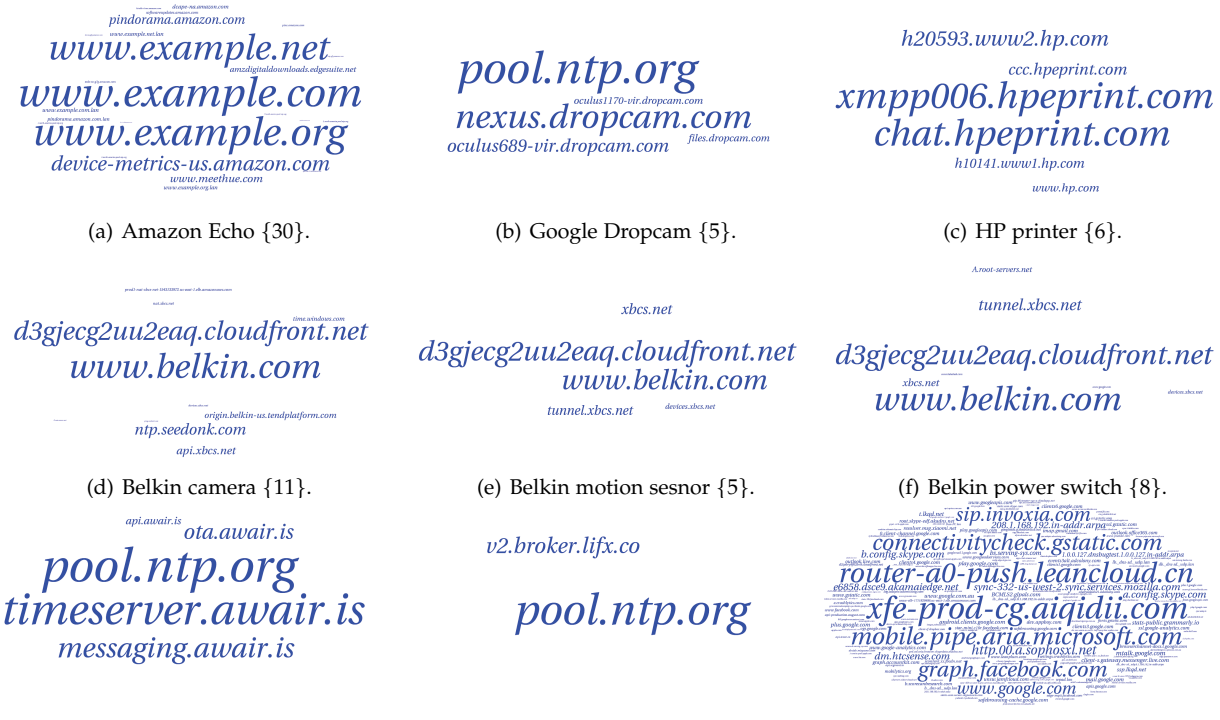
For the average flow rate attribute, Fig. 3(c) shows that the mean rate is rather small, in the bits-per-second range as one would qualitatively expect. Quantitatively, the figure shows that the LiFX lightbulb has an average flow rate of 18




(a) Amazon Echo {10}. (b) LiFX lightbulb {5}. (c) Awair air monitor {7}. (d) Netatmo weather station {4}.

(e) Belkin motion sensor {7}. (f) Belkin power switch {7}. (g) Belkin camera {9}. (h) Non-IoT {2382}.

Fig. 4. Word-cloud of server ports (total count of unique ports is shown in {sub-captions} next to the device name).




(a) Amazon Echo {30}. (b) Google Dropcam {5}. (c) HP printer {6}.

(d) Belkin camera {11}. (e) Belkin motion sesnor {5}. (f) Belkin power switch {8}.

(g) Awair air quality monitor {5}. (h) LiFX lightbulb {2}. (i) Non-IoT {11927}.

Fig. 5. Word-cloud of domain names (total count of unique domains is shown in {sub-captions} next to the device name).

bits-per-second nearly 60% of the time. Nearly 30% of Belkin flows have a bit rate in the range 59 to 60 bits-per-second while nearly 40% Amazon Echo flows have a bit range in the range 70 to 71 bits-per-second.

Lastly, in terms of the sleep time for the devices Fig. 3(d) shows that the Belkin motion sensor and the LiFX lightbulb exhibit a distinct sleep pattern. The duration is 1 second and 60 seconds with probability 73% and 48% respectively. However, multiple sleep times with small probabilities are observed for the Amazon Echo. This is because Amazon Echo keeps its TCP connections alive and goes to sleep only when it disconnects from the Internet. Other devices in our test-bed also perform like the Echo and do not seem to have a dominant sleep pattern.

### 4.2 IoT Signaling Pattern

We now focus on the application layer protocols, inferred using the port numbers, that IoT devices mostly use to communicate locally in the LAN and/or externally with servers on the public Internet.

#### 4.2.1 *Server port numbers*

Fig. 4 shows the word cloud of server-side port numbers of all flows initiated from a variety of IoT devices. For each device, if a port is used more frequently then it is shown by a larger font-size in the respective word cloud. Sub-captions (i.e. numbers within {}) report the number of unique server ports for each device. It can be seen that IoT devices each uniquely communicate with a handful of server ports whereas non-IoT devices use a much wider range of services (i.e. 2382 unique ports are shown in Fig. 4(h) and many of them are very infrequent). We observe that non-standard ports 33434, 56700, 8883, and 25050 are prominently seen in traffic originating from Amazon Echo, LiFX lightbulb, Awair air quality monitor, and Netatmo weather station respectively, as shown in the top row of Fig. 4. Further, we note devices from the same manufacturer share certain ports. For example, port numbers 8443 and 3478 are common between Belkin's motion sensor, power switch, and camera, as shown in Figures 4(e)-4(g). We also

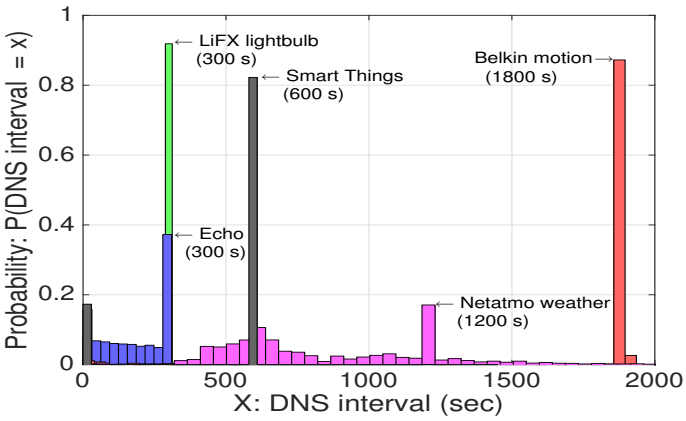



Fig. 6. Histogram of DNS interval.

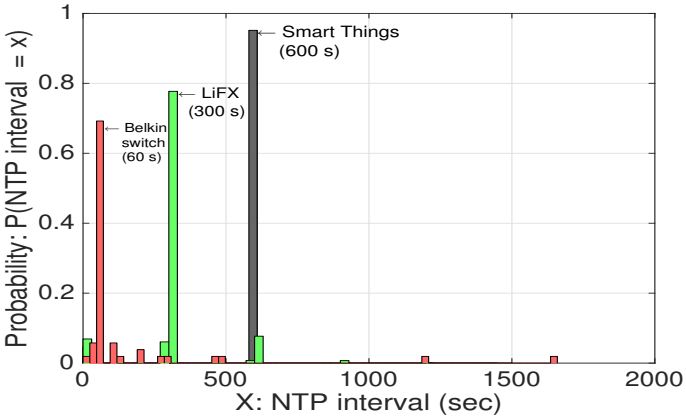



Fig. 7. Histogram of NTP interval.

note that well-known standard port numbers such as 53 (DNS), 123 (NTP), 0 (ICMP) and 1900 (SSDP) are used by many of the IoT devices as well as the non-IoTs with various frequencies, as shown in Fig. 4. Moreover, the server-side port number of 443 (TLS/SSL) is also used by many of the IoT devices.

#### 4.2.2 DNS queries

DNS is a common application used by almost all networked devices. Since IoT devices are custom-designed for specific purposes, they access a limited number of domains corresponding to their vendor-specific end-point servers. We plot in Fig. 5 the word cloud of domain names accessed by several IoT devices as well as non-IoTs. It is seen that IoT devices are fairly distinguishable by the domain names they communicate with. For example, as depicted in Figures 5(a)-5(c), domains such as `example.com`, `example.net`, and `example.org` are frequently requested by Amazon Echo; sub-domains of `hp.com` and `hpeprint.com` are seen in DNS queries from the HP printer. However, we also see that some prominent domain names are shared between the different devices. For example, `belkin.com` and `d3gjecg2uu2faq.cloudfront.net` are commonly used by Belkin devices (i.e. camera, motion sensor and power switch) as shown in Figures 5(d)-5(f); or `pool.ntp.org` is prominent in traffic flows generated from Google Dropcam, Awair air quality monitor and LiFX lightbulb, as shown in Figures 5(b)-5(h). Again considering non-IoTs in Fig. 5(i), we see about 12000 unique domains visited which is far diverse compared to IoT devices with only a handful of domains accessed repeatedly.


```
{"ciphersuite":["c014", "c00a", "0039", "0038", "0037", "0036", "0088", "0087", "0086",
"0085", "c00f", "c005", "0035", "0084", "c013", "c009", "0033", "0032", "0031", "0030",
"009a", "0099", "0098", "0097", "0045", "0044", "0043", "0042", "c00e", "c004", "002f",
"0096", "0041", "0007", "c011", "c007", "c00c", "c002", "0005", "0004", "c012", "c008",
"0016", "0013", "0010", "000d", "c00d", "c003", "000a", "0015", "0012", "000f", "000c",
"0009", "00ff"] ,  "negotiated cipher":"002f"}
```


(a) cs1 of Amazon Echo.


```
{"ciphersuite":["c030", "c02c", "c028", "c024", "c014", "c00a", "00a5", "00a3", "00a1",
"009f", "006b", "006a", "0069", "0068", "0039", "0038", "0037", "0036", "0088", "0087",
"0086", "0085", "c032", "c02e", "c02a", "c026", "c00f", "c005", "009d", "003d", "0035",
"0084", "c02f", "c02b", "c027", "c023", "c013", "c009", "00a4", "00a2", "00a0", "009e",
"0067", "0040", "003f", "003e", "0033", "0032", "0031", "0030", "009a", "0099", "0098",
"0097", "0045", "0044", "0043", "0042", "c031", "c02d", "c029", "c025", "c00e", "c004",
"009c", "003c", "002f", "0096", "0041", "0007", "c011", "c007", "c00c", "c002", "0005",
"0004", "c012", "c008", "0016", "0013", "0010", "000d", "c00d", "c003", "000a", "0015",
"0012", "000f", "000c", "0009", "00ff"] , "negotiated cipher": "c02f"}
```


(b) cs2 of Amazon Echo.

Fig. 8. Signature of cipher suite.

We also found that IoT devices differ from one other in how often the DNS protocol is used. We have observed from our traffic traces that IoT devices generate DNS queries during different stages of its operation; for example only during the boot-up phase (e.g. Google Dropcamp) or when interacting with a user (e.g. Hello Barbie) or periodically (e.g. Amazon Echo). As shown in Fig. 6, certain IoT devices exhibit a characteristic signature in the frequency of their DNS queries. The LiFX lightbulb and Amazon Echo send DNS queries very frequently (i.e. every 5 minutes) but a device like the Belkin motion sensor requests domain names only once every 30 minutes.

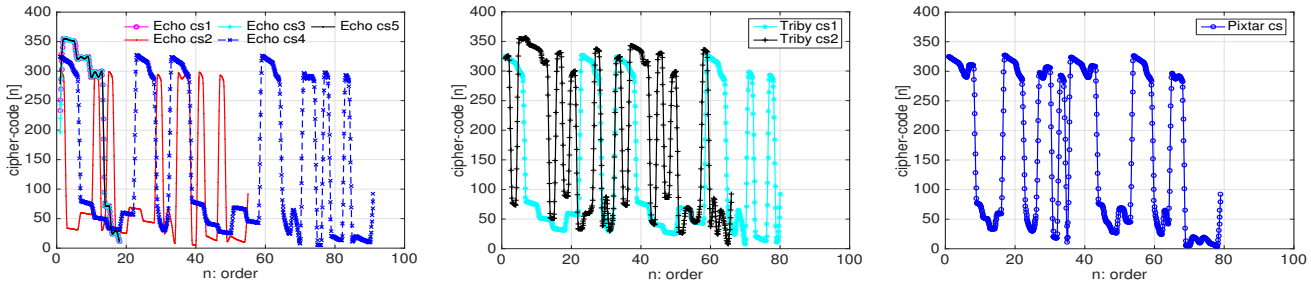
#### 4.2.3 NTP queries

As mentioned earlier, NTP is another popular protocol used by IoT devices because precise and verifiable timing is crucial for IoT operations [34]. Many IoT devices tend to use NTP protocol (UDP port 123) in a periodic manner in order to synchronize their time with publicly available NTP servers. For example, Awair air quality monitor, LiFX lightbulb and Google Dropcam obtain the IP address of time servers from `pool.ntp.org`. We also find that time synchronization occurs repeatedly in our test-bed and many IoT devices exhibit a recognizable pattern in the use of NTP. For example, the Belkin power switch, LiFX lightbulb and SmartThings hub send NTP requests every 60, 300 and 600 seconds respectively, as shown in histogram plot of Fig. 7.

#### 4.2.4 Cipher suite

A number of IoT devices use TLS/SSL protocol (port number 443) to communicate with their respective servers on the Internet [30]. In order to initiate the TLS connection and negotiate the security algorithms with servers, devices start handshaking by sending a "Client Hello" packet with a list of "cipher suites" that they can support, in the order of their preference. For example, Figures 8(a) and 8(b) depict cipher suites that Amazon Echo offers to two different Amazon servers. Each cipher suite (i.e. 4-digit code) can take one of 380 possible values and represents algorithms for key exchange, bulk encryption and message authentication code (MAC). For example, the cipher `002f` negotiated by an Amazon server uses RSA, AES_128_CBC, and SHA protocols for key exchange, bulk encryption and message authentication, respectively.

We find that 17 out of the 28 IoT devices in our setup, including the Amazon Echo, August Doorbell Cam, Awair air quality monitor, Belkin Camera, Canary Camera, Dropcam, Google Chromecast, Hello Barbie, HP ENVY Printer,




(a) Amazon Echo. (b) Triby. (c) Pixtar photoframe.

Fig. 9. Signature of cipher suite.

iHome, Netatmo Welcome camera, Philips Hue lightbulb, Pixtar photoframe, Ring Door Bell, Triby, Withings Aura smart sleep sensor and Withings Scale, use TLS/SSL for communication. We find that Amazon Echo uses total of five different cipher suite strings when communicating SSL to different servers, Triby speaker uses two strings, while the Pixtar photoframe uses only one string for all of its SSL communications. We plot unique cipher suite strings from these three devices in Fig. 9 as discrete signals: x-axis is the order of 4-digit cipher codes that appear in the offered suite, and y-axis is the index of the individual cipher codes (i.e. a value from $\{1, 2, ..., 380\}$). It is seen that the collection of cipher suite signals enunciates a unique signature for each IoT device. Exceptionally, we found that Pixtar photoframe shares its single cipher suite with one of 18 suites that are used by August door-bell – we will see in §5.2 that relying only on cipher suite attribute would not be effective in classifying Pixtar photo-frame traffic.

There are however many devices that rarely exchange cipher suites but instead prefer to keep their TLS connections alive for a long period. For example, Google Dropcam establishes a TLS connection to its own server whenever it boots up and maintains this connection as long as it has network connectivity, while Amazon Echo and Pixstar photoframe initiate on average 1 and 2 TLS connections respectively every hour.

**Summary:** In this section, we have identified 8 key attributes based on the underlying network traffic characteristics of IoT devices. They are flow volume, flow duration, average flow rate, device sleep time, server port numbers, DNS queries, NTP queries and cipher suites. Although, some devices (e.g. Amazon Echo, or LiFX lightbulb) can be uniquely identified by considering just one or two traffic attributes such as the list of domain-names, port-numbers, or cipher suites, these come with challenges. For example, a strong attribute like the list of cipher-suites is observed very infrequently in the traffic (e.g. only once a day). As another example, different types of devices from the same vendor visit similar domains and use the same port numbers to access cloud servers. Capturing aspects such as the number of occurrences for these attributes (e.g. number of times a domain is accessed or number of streams that use the port), in combination with other attributes, vastly improves the prediction capability to distinguish between devices from the same manufacturer. In the next section, we develop a multi-stage machine learning based algorithm using combinations of these attributes to help classify IoT devices with high accuracy.

## 5 Machine Learning Based Classification

In order to synthesize the attributes from our trace data, we first convert the raw pcap files into flows on an hourly basis using the Joy tool [31]. Then, for a given IoT device, we compute the traffic activity and signalling attributes defined in the previous section over the hourly instances. The number of instances for each device obtained from the trace spanning 26 weeks varies depending on factors such as the duration for which a device is online, or how a device generates traffic (autonomously or interactively). For example, there were only 13 hourly instances for the Blipcare BP monitor since it generates traffic only when the device is used by a user. On the other hand, we collected 4177 instances for Google Dropcam.

### 5.1 Multi-Stage Device Classification Architecture

We note that three of our attributes namely "set of domain names", "set of remote port numbers" and "set of cipher suites" are nominal (i.e. are not treated as numeric values) and multi-valued (for example, {"53":3, "123":1, "443":2} represents a set of remote port numbers with three occurrences of port number 53, two occurrences of port 123, and one occurrence of port number 443). Our remaining attributes including flow volume/duration, flow rate, sleep time, and DNS/NTP intervals contain single quantitative and continuous values. We therefore employ a two-stage hierarchical architecture for our IoT classifier as shown in Fig. 10.

In this architecture, we first feed each multi-valued attribute to its corresponding stage-0 classifier in the form of a "bag of words". A bag of words is a matrix whose rows represent labeled instances, and columns represent unique words. This matrix has $M$ rows (i.e. total number of instances) and $N$ columns (i.e. number of unique words). We observed 356, 421 and 54 unique words for domain-names, remote port numbers and cipher suite strings, as shown in Fig. 10. In addition to these unique words, we aggregated all corresponding words for non-IoT devices as "others" - a column called "others" in each Stage-0 matrix represents words not seen in IoT traffic. Each cell of this matrix is the number of occurrences of such unique words in a given instance.

As shown in Fig. 10, each classifier of Stage-0 generates two outputs, namely a tentative class and a confidence level, which together with other single-valued quantitative attributes (i.e. flow volume, duration, rate, sleep time, DNS, NTP intervals) are fed into a Stage-1 classifier that produces

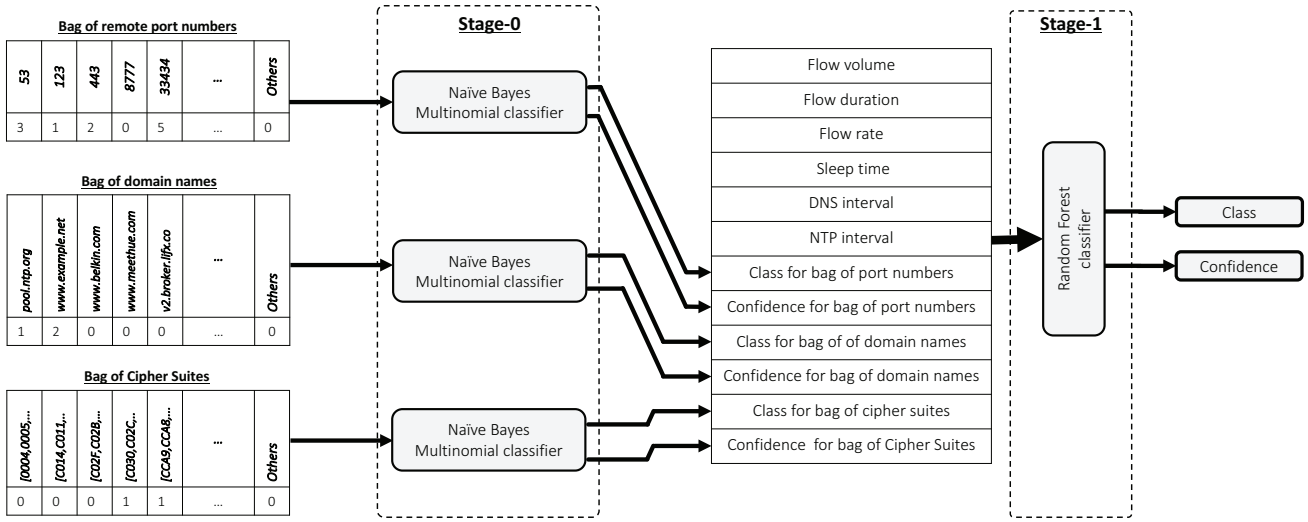



Fig. 10. System architecture of the multi-stage classifier.

the final output (i.e. the device identification with a confidence level).

#### 5.1.1 Stage-0: Bag-of-words Classifiers

We employ a Naive Bayes Multinomial classifier to analyze each bag of words in the stage-0 of our machine. It has been shown [35] that this classifier performs well in text classification when dealing with a large number of unique words. During the training phase, the classifier takes the distribution of words, e.g. individual unique domain names, and computes the probability of each word given a class using:

$$\Pr(w_j^{train}|c_i) = \frac{1 + \sum_{l=1}^{D} {n_{l,c_i,w_j}}^{train}}{N + \sum_{k=1}^{N}\sum_{l=1}^{D} {n_{l,c_i,w_k}}^{train}} \quad (1)$$

where $w_j$ is a unique word in the training dataset (e.g. port number 56700); $c_i$ is a class label (e.g. LiFX lightbulb); $D$ is the total number of instances; ${n_{l,c_i,w_j}}^{train}$ is the number of $w_j$ occurrences in each of instances with class label of $c_i$; $N$ is the total number of unique words (e.g. we have $N = 421$ unique port numbers in our dataset).

During the testing phase, the classifier needs to compute the following probability for all possible classes:

$$\Pr(c_i|W^{test}) = \Pr(c_i^{train}) \prod_{j=1}^{N} \Pr\left({w_j}^{train}|c_i\right)^{n_j^{test}} \quad (2)$$

where $W^{test}$ is a set represented by $\{w_1 : n_1^{test}, w_2 : n_2^{test}, ..., w_N : n_N^{test}\}$; $n_j^{test}$ is the occurrence number of individual unique words $w_j$ in a given test instance; $\Pr(c_i^{train})$ is the presence probability of a class $c_i$ in the whole training dataset (i.e. number of $c_i$ training instances divided by total number of all training instances). The classifier finally chooses the class that gives the maximum probability in (2) for a given set of words along with their occurrences. Note that a Naive Bayes Multinomial classifier performs well if training instances are fairly distributed among various classes [35].

#### 5.1.2 Stage-1 Classifier

We have a stage-1 classifier that takes all quantitative attributes along with the pair of outputs from each stage-0 classifier. Since the stage-1 attributes are not linearly separable and the outputs of stage-0 classifiers are nominal values, we use a Random Forest based stage-1 classifier. Another reason for selecting the Random Forest is its high tolerance to over-fitting compared to other decision tree classifiers.

### 5.2 Performance Evaluation

We use the Weka [36] tool for our IoT device classification. We have collected a total of 50,378 labeled instances from our traffic traces. As mentioned earlier, we have a number of instances from different devices – those that generate traffic when triggered by user interaction have small number of instances (e.g. 13 for Blipcare BP monitor, 21 for Google Chromecast) and those that autonomously generate traffic have a fairly large number of instances (e.g. 2,868 for Samsung Smart Things or 2,247 for Amazon Echo). We have randomly split instances into two groups, one containing 70% of the instances for "training" and another containing 30% of the instances for "testing".

Table 2 shows the performance of our classifier under various scenarios, each captured by a pair of columns. For a given scenario, we measure the true positive rate (i.e. fraction of test instances that are correctly classified) and false positive rate (i.e. fraction of test instances that are incorrectly classified) for every device corresponding to the rows in Table 2. We also obtain the average confidence level (i.e. a number between 0 and 1 depicted within square brackets in each cell) of our classifier for correctly classified and incorrectly classified instances. In addition, we aggregate the performance of individual classes and compute the overall accuracy (i.e. total true positive rate) along with the overall root relative squared error (RRSE) as measures of performance for our classifier. These measures are reported in the top row of each scenario in Table 2. Note that our objective is to achieve a high accuracy (close to 100%) with a fairly low error (close to zero).

TABLE 2
Performance of the proposed IoT device classifier under different sets of attributes.

| Devices | Port Numbers | | Domain Names | | Cipher Suite | | Combined stage-0 | | Final | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Accuracy: 92.13% RRSE: 39.93% | | Accuracy: 79.48% RRSE: 57.56% | | Accuracy: 36.15% RRSE: 86.73% | | Accuracy: 97.39% RRSE: 18.24% | | Accuracy: 99.88% RRSE: 5.06% | |
| | Correctly Classified | Incorrectly Classified | Correctly Classified | Incorrectly Classified | Correctly Classified | Incorrectly Classified | Correctly Classified | Incorrectly Classified | Correctly Classified | Incorrectly Classified |
| Amazon Echo | **100.0%** [1.00] | | **99.6%** [1.00] | Dropcam: **0.4%** [0.08] | **100.0%** [1.00] | | **99.9%** [1.00] | HP Printer: **0.1%** [0.52] | **99.7%** [1.00] | NonIoT: **0.1%** [0.37] Dropcam: **0.1%** [0.43] |
| August Doorbell | **99.0%** [1.00] | iHome Plug: **0.6%** [1.00] Others: **0.4%** [0.65] | **100.0%** [1.00] | | **78.8%** [1.00] | Pixstar Photo.: **21.2%** [1.00] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Awair air quality | **97.6%** [1.00] | NonIoT: **2.0%** [0.32] Amazon Echo: **0.4%** [0.53] | **99.2%** [1.00] | Smart Things: **0.4%** [0.49] Dropcam: **0.4%** [0.08] | **99.2%** [0.63] | Dropcam: **0.8%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Belkin Camera | **95.5%** [1.00] | Belkin Motion: **3.0%** [0.94] Others: **1.5%** [0.67] | **39.4%** [0.99] | Belkin Motion: **59.8%** [0.62] NonIoT: **0.8%** [1.00] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **97.7%** [0.99] | NonIoT: **1.5%** [0.74] Dropcam: **0.8%** [1.00] | **97.7%** [0.99] | NonIoT: **1.5%** [0.60] Netat. Camera: **0.8%** [0.57] |
| Belkin Motion | **99.8%** [1.00] | NonIoT: **0.2%** [1.00] | **0.0%** [-] | Belkin Switch: **100.0%** [0.57] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **99.5%** [1.00] | Samsung Cam: **0.3%** [0.79] NonIoT: **0.2%** [1.00] | **99.8%** [1.00] | NonIoT: **0.2%** [0.97] |
| Belkin Switch | **99.5%** [1.00] | Belkin Motion: **0.2%** [0.77] Others: **0.3%** [0.75] | **99.7%** [0.57] | Dropcam: **0.2%** [0.08] Blipcare BP Meter: **0.1%** [0.79] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **99.8%** [1.00] | Belkin Motion: **0.2%** [1.00] | **99.8%** [1.00] | Belkin Motion: **0.2%** [0.93] |
| Canary Camera | **100.0%** [1.00] | | **100.0%** [1.00] | | **100.0%** [1.00] | | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Dropcam | **98.1%** [0.33] | Smart Things: **0.6%** [0.99] Others: **1.3%** [0.59] | **100.0%** [0.09] | | **100.0%** [0.09] | | **74.0%** [0.96] | HP Printer: **25.7%** [0.52] Others: **0.3%** [0.41] | **100.0%** [1.00] | |
| LiFX Bulb | **100.0%** [1.00] | | **99.7%** [0.70] | Smart Things: **0.1%** [0.42] Dropcam: **0.1%** [0.08] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| NEST Smoke | **100.0%** [1.00] | | **100.0%** [1.00] | | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Netat. Weather | **99.8%** [1.00] | Dropcam: **0.2%** [0.08] | **99.9%** [1.00] | Dropcam: **0.1%** [0.08] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Netat. Camera | **95.4%** [1.00] | Dropcam: **2.0%** [0.97] Others: **2.6%** [0.70] | **97.8%** [1.00] | Dropcam: **2.2%** [0.08] | **99.7%** [0.92] | Dropcam: **0.3%** [0.08] | **99.8%** [1.00] | Pixstar Photo.: **0.1%** [0.54] Dropcam: **0.1%** [0.60] | **99.9%** [1.00] | Hue Bulb: **0.1%** [0.37] |
| Pixstar Photo. | **99.7%** [1.00] | Dropcam: **0.3%** [0.08] | **99.3%** [1.00] | Dropcam: **0.7%** [0.08] | **0.0%** [-] | August Doorbell: **99.7%** [0.71] Dropcam: **0.3%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Samsung Cam | **99.4%** [1.00] | Belkin Motion: **0.6%** [1.00] | **14.5%** [1.00] | Dropcam: **73.4%** [0.10] Smart Things: **12.0%** [0.43] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Smart Things | **97.5%** [1.00] | LiFX Bulb: **1.9%** [0.99] Others: **0.5%** [0.68] | **79.9%** [0.50] | LiFX Bulb: **20.1%** [0.50] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **99.8%** [1.00] | LiFX Bulb: **0.1%** [0.88] Dropcam: **0.1%** [0.97] | **99.8%** [1.00] | LiFX Bulb: **0.1%** [0.71] Dropcam: **0.1%** [0.67] |
| TP-Link Cam. | **100.0%** [1.00] | | **99.7%** [1.00] | Dropcam: **0.3%** [0.08] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| TP-Link Plug | **99.7%** [1.00] | Dropcam: **0.3%** [0.08] | **99.7%** [0.99] | Dropcam: **0.3%** [0.08] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Triby Speaker | **98.0%** [1.00] | Netat. Weather: **1.2%** [0.37] Others: **0.8%** [0.49] | **100.0%** [1.00] | | **41.2%** [0.99] | Dropcam: **54.8%** [0.08] Netat. Weather: **4.0%** [0.16] | **99.9%** [1.00] | NonIoT: **0.1%** [1.00] | **99.9%** [1.00] | NonIoT: **0.1%** [0.84] |
| Withings Sleep. | **96.8%** [1.00] | NonIoT: **1.9%** [0.99] Others: **1.2%** [0.55] | **99.6%** [1.00] | Dropcam: **0.4%** [0.08] | **23.5%** [1.00] | Dropcam: **76.5%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Hue Bulb | **88.8%** [1.00] | Samsung Cam: **11.1%** [0.45] Belkin Motion: **0.1%** [1.00] | **89.0%** [1.00] | Dropcam: **11.0%** [0.08] | **0.8%** [0.71] | Dropcam: **99.2%** [0.08] | **99.9%** [1.00] | NonIoT: **0.1%** [0.57] | **99.9%** [1.00] | NonIoT: **0.1%** [0.47] |
| Google Chromecast | **62.5%** [1.00] | Amazon Echo: **25.0%** [0.52] NonIoT: **12.5%** [0.60] | **100.0%** [1.00] | | **100.0%** [0.98] | | **87.5%** [1.00] | Dropcam: **12.5%** [0.69] | **87.5%** [0.98] | Dropcam: **12.5%** [0.57] |
| HP Printer | **61.5%** [0.99] | Dropcam: **38.0%** [0.16] Others: **0.6%** [0.86] | **3.8%** [1.00] | Dropcam: **96.2%** [0.08] | **2.5%** [0.45] | Dropcam: **97.1%** [0.08] Others: **0.4%** [0.75] | **99.3%** [0.82] | Dropcam: **0.4%** [0.85] Others: **0.2%** [0.39] | **99.8%** [0.99] | NonIoT: **0.1%** [0.67] Dropcam: **0.1%** [0.28] |
| iHome Plug | **79.2%** [0.90] | Dropcam: **10.2%** [0.34] Others: **10.6%** [0.42] | **87.5%** [0.97] | Dropcam: **12.5%** [0.08] | **18.0%** [0.57] | Dropcam: **82.0%** [0.08] | **89.8%** [1.00] | Dropcam: **9.8%** [0.96] HP Printer: **0.4%** [0.52] | **100.0%** [0.99] | |
| Withings Baby Mon. | **58.2%** [1.00] | NonIoT: **41.8%** [1.00] | **100.0%** [1.00] | | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Withings Scale | **74.8%** [0.98] | NonIoT: **15.3%** [0.56] Others: **9.9%** [0.19] | **41.4%** [0.79] | Withings Sleep.: **56.8%** [0.96] Dropcam: **1.8%** [0.08] | **42.3%** [0.33] | Dropcam: **57.7%** [0.08] | **99.1%** [1.00] | Dropcam: **0.9%** [0.54] | **100.0%** [1.00] | |
| Ring Door Bell | **0.6%** [0.98] | Netat. Weather: **95.8%** [0.18] Others: **3.6%** [0.60] | **100.0%** [0.98] | | **7.8%** [1.00] | Dropcam: **92.2%** [0.08] | **100.0%** [1.00] | | **100.0%** [1.00] | |
| Blipcare BP Meter | **20.0%** [0.54] | Ring Door Bell: **80.0%** [0.41] | **40.0%** [0.79] | HP Printer: **60.0%** [0.44] | **0.0%** [-] | Dropcam: **100.0%** [0.08] | **100.0%** [0.90] | | **100.0%** [0.85] | |
| Hello Barbie | **0.0%** [-] | Dropcam: **71.4%** [0.08] Others: **28.6%** [0.50] | **21.4%** [1.00] | Dropcam: **71.4%** [0.08] HP Printer: **7.1%** [0.45] | **21.4%** [0.99] | Dropcam: **78.6%** [0.08] | **14.3%** [0.97] | HP Printer: **78.6%** [0.52] Dropcam: **7.1%** [0.61] | **92.9%** [0.99] | Hue Bulb: **7.1%** [0.35] |
| NonIoT | **74.2%** [0.98] | Triby Speaker: **16.6%** [0.90] Others: **9.2%** [0.69] | **66.9%** [0.97] | Dropcam: **29.7%** [0.08] Others: **3.4%** [0.73] | **59.5%** [0.79] | Dropcam: **36.3%** [0.08] Others: **4.2%** [0.73] | **98.8%** [1.00] | HP Printer: **1.1%** [0.56] Dropcam: **0.2%** [0.75] | **99.7%** [0.99] | HP Printer: **0.3%** [0.55] |

### 5.2.1 *Performance of Stage-0: Port Numbers Attribute*

The first three columns correspond to those cases in which we consider only nominal attributes of stage-0 (i.e. bag of words corresponding to port numbers, domain names and cipher suites). The first column shows that when we only use a list of server-side port numbers for device classification, a reasonable accuracy of 92.13% is achieved, but RRSE is poor (at 39.93%). Inspecting the individual classes, we observe that certain classes highlighted by yellow or light-green (e.g. Ring door bell, Blipcare BP monitor, Hello Barbie, and Google chromecast) are poorly classified. We explain the reason behind this misclassification next.

**Ring door bell:** Out of 486 instances, 465 contain a single occurrence of the DNS query (i.e. remote port number 53). We see that 95.8% of test instances are incorrectly classified as Netatmo weather station. This is because of two reasons: (i) there are 2451 training instances of Netatmo compared to 323 of Ring door bell, which makes $\Pr(c_i^{train})$ of Netatmo larger than that of Ring door bell, and (ii) many Netatmo instances contain several (on average 4 times) occurrences of port 53 as opposed to only one for Ring Door bell, which also contributes to $\Pr(w_j|c_i)$ of Netatmo being greater than that for Ring door bell in (1). Thus, Ring door bell instances get classified as Netatmo weather station, warranting a second stage of classification with additional attributes for improved accuracy.

**Blipcare BP monitor:** It uses only two remote port numbers, namely 8777 and 53, in a total of 13 instances - the port numbers appear only once or twice in each instance. Surprisingly, we see that 80% of Blipcare test

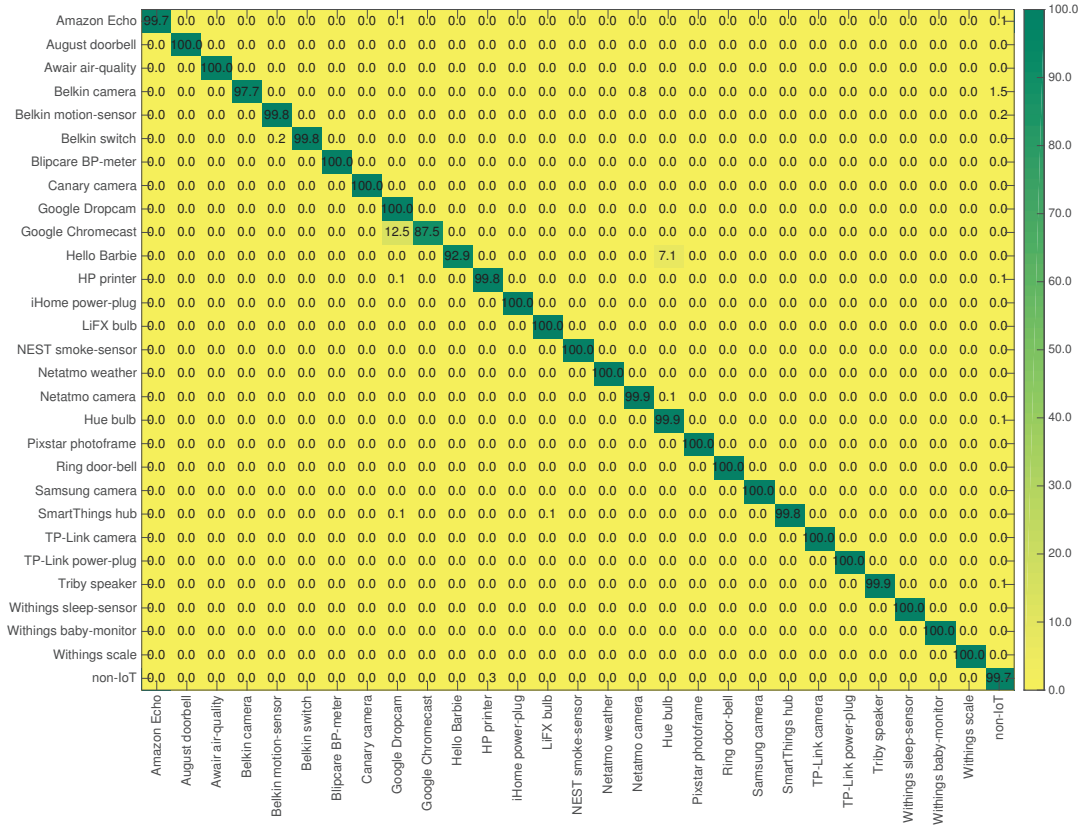



Fig. 11. Confusion matrix of our IoT device classification using all attributes (accuracy: 99.88%, RRSE: 5.06%).

instances are incorrectly classified as Ring Door Bell though the remote port number of 8777 is unique to the Blipcare BP monitor. This is because there are only a very small number of Blipcare instances in our dataset, which results in a fairly small value of $\Pr(\text{“53”}|Blipcare) = 0.0203$ and $\Pr(\text{“8777”}|Blipcare) = 0.0294$ in (1), and a negligible value of $\Pr(Blipcare^{train}) = 0.0003$ in (2). On the other hand, $\Pr(\text{“8777”}|Ring)$ becomes very small as the remote port number 8777 is never used by the Ring Door Bell in our dataset. However, the probability of $\Pr(\text{“8777”}|Ring) = 0.0011$ in (1) is sufficient enough to maximize the classifier probability $\Pr(Ring|\{\text{“53”} : 1, \text{“8777”} : 1\})$ in (2), given $\Pr(Ring^{train}) = 0.0097$.

**Other devices:** Server-side port numbers are empty in 72% of instances for Hello Barbie, since it communicates with local devices instead of Internet-based end-points. Similarly for HP printer (38%) and iHome power plug (10%). The lack of server-side port number information explains why these devices are classified as Dropcam, which has the highest value of $\Pr(Dropcam^{train}) = 0.0828$ in (2). We note that the confidence level of our stage-0 classifier is fairly low (i.e. less than $0.4$) in these cases, suggesting that the classifier chooses the most probable class given empty attribute (i.e. all $n_j^{test}$ are zero).

### 5.2.2 Performance of Stage-0: Domain Names Attribute

We now focus on the stage-0 machine that uses only a bag of domain-names, which yields an accuracy of $79.48\%$ with a fairly high RRSE value of $57.56\%$, as shown in the second column in Table 2. In this scenario, more classes suffer from misclassification (i.e. those with yellow coloured cells) compared to the previous scenario where only remote port numbers were considered. The reasons behind the misclassification are threefold: (i) since devices from the same manufacturer share a collection of domain names, as discussed in §4.2.2, $59.8\%$ of Belkin camera test instances are misclassified as Belkin Motion sensor and $100\%$ Belkin Motion sensor instances are misclassified as Belkin switch. Similarly, $56.8\%$ of Withings scale instances are incorrectly classified as Withings sleep sensor, and $12\%$ of Samsung smart cam are misclassified as Samsung Smartthings. (ii) a significant number of instances from select devices contain no DNS query entries (e.g. $96.2\%$ of HP printer, $73.4\%$ of Samsung Smart Cam, $71.4\%$ of Hello Barbie, $12.5\%$ of iHome power plug, $11\%$ of Hue bulb) and are thus incorrectly classified as a Dropcam, which also rarely generates DNS packets. (iii) the low number of training instances with domain names leads to poor performance (e.g. Blipcare BP meter and Hello Barbie).

### 5.2.3 Performance of Stage-0: Cipher Suite Attribute

Considering only the cipher suite attribute, this stage-0 classifier results in a fairly low accuracy of $36.15\%$ with a high RRSE of $86.73\%$, as shown in the third column in Table 2. Again, the main reason for such poor performance is the scarcity of cipher suite attribute in the training instances, though this attribute carries a very strong signature to uniquely identify an IoT device. Note that many of the IoT devices do not use secure communication at all and are thus devoid of this attribute (i.e. have an empty field for it). Unsurprisingly, instances of devices that exchange cipher suite fairly frequently including Amazon Echo, Awiar air quality monitor, Canary camera, Google Chromecast and Netatmo camera are correctly classified, as shown by the dark-green color cells in the corresponding column in Table 2. In addition, we find that August doorbell cam is

TABLE 3
Impact of attributes combination on performance of classifier.

| | Accuracy | RRSE |
|---|---|---|
| all attributes | 99.88% | 5.06% |
| low- and medium-cost attributes | 99.68% | 7.70% |
| only low-cost attributes | 97.85% | 18.63% |

sharing one of its cipher suite strings (out of total 18) with Pixstar photoframe, which has a single cipher suite string. Thus, 21.2% of August door bell instances are misclassified as Pixstar photoframe and almost all instances of Pixstar photoframe are classified as August doorbell.

#### 5.2.4 Performance of Stage-0: Combination of Attributes

We expect the combination of the three bags of words (port numbers, domain names, and cipher suites) to significantly enhances the accuracy of our classifier, as indeed shown by the fourth column titled "Combined stage-0" in Table 2. The overall accuracy reaches to 97.39% with RRSE of 18.24%. It can be seen that the majority of test instances are correctly classified, except for Hello Barbie. This is because most of the Hello Barbie attributes are empty in stage-0 and thus it is classified as Dropcam, as mentioned earlier.

Interestingly, we see that all test instances of Blipcare BP monitor are classified correctly though the accuracy of individual stage-0 was fairly poor. This is because our decision-tree-based classifier in stage-1 sees a strong correlation between the outputs of stage-0 classifiers and the actual class of training instance, even though those outputs (tentative class) are incorrect – e.g. having the tentative output from remote port number classifier as Ring door bell, having the tentative output from cipher suite classifier as Dropcam, and having the confidence level from domain name classifier less than 0.66 collectively is a strong indication of Blipcare instance.

#### 5.2.5 Overall Performance

As the last step, we incorporate the outputs from the stage-0 classifiers into stage-1 (without the latter having any notion of the quantitative attributes from the former), and additionally include quantitative attributes (flow volume, duration, rate, sleep time, DNS and NTP intervals). The last column of Table 2 shows the overall performance of the classification framework. In this case, the accuracy reaches a remarkably high value of 99.88%, with almost all classes labeled correctly with a very small value of RRSE at 5.06%. Fig. 11 shows the full confusion matrix of our classification when all the attributes are used in conjunction, and corroborates that the diagonal entries (corresponding to correct classification) are all at or very close to 100%, with just two exceptions – the Google Chromecast and the Hello Barbie. As explained earlier, the Chromecast gets classified as the Dropcam in some instances, while the Hello Barbie gets classified as a Hue bulb.

## 6 Real-Time Operation in a Network

Thus far, we have examined the performance of our multi-stage classifier using off-line analysis on captured traffic traces (i.e. pcap files). In this section, we discuss how one can realize a real-time implementation of our system taking into account the various stages involved in the analysis, namely attribute collection, machine training, and interpreting the classifier's output.

### 6.1 Computing Attributes

Extracting the attributes on-the-fly requires infrastructure that has sufficient visibility into the traffic flowing on the network. Flow related attributes such as flow volume, flow duration and flow rate can be extracted relatively easily using network switches that are instrumented with special hardware-accelerated flow-level analyzers, e.g. NetFlow capable devices [37]. We therefore deem the extraction cost of flow related attributes to be fairly low, and show them via blue color bars in Fig. 12(c) that depicts the relative costs and merits of the various attributes.

Attributes including bag of port numbers, sleep-time, and frequency of DNS/NTP requests can be extracted using flow-aware network switches with extra computation and state management. For example, remote port numbers of all flows associated with a given IoT device need to be recorded for the bag of port numbers. However, this specific state is not captured by default in commodity switches. Similarly, time intervals between successive UDP packets of NTP/DNS should be recorded, which requires additional computation. We therefore associate these attributes with medium cost, and shown as yellow color bars in Fig. 12(c).

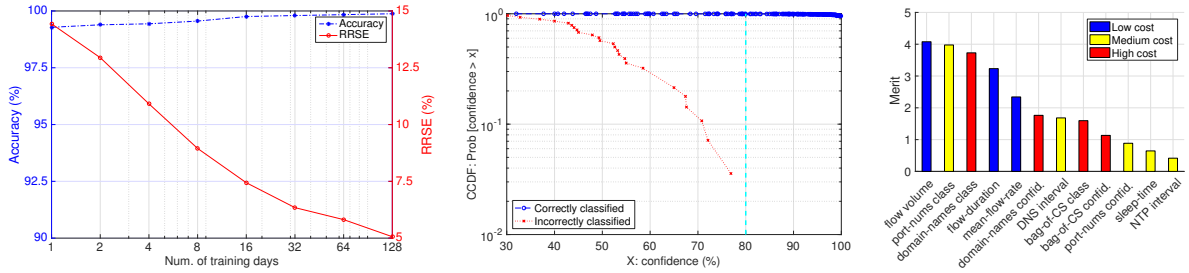
Lastly, two of our attributes, namely bag of domain names and bag of cipher suite strings, can only be extracted by looking inside the payload of the appropriate packets, which imposes considerable cost on processing. Thus, we associate these attributes with high collection cost, and shown them via red color bars in Fig. 12(c).

Having understood the extraction cost of various attributes, let us now examine the relative importance of the attributes in classifying the IoT devices. We quantify the importance of each attribute by employing the *select attributes* tool in Weka with *InfoGain* attribute evaluator and *Ranker* search method. Fig. 12(c) shows the attributes in decreasing order of merit score. A high merit score translates to superior strength in identifying the class of an instance. We can see that the "flow-volume" is the most important attribute, followed by "bag of remote port numbers", "bag of domain names" and "flow duration" respectively. The sleep-time and NTP interval are the attributes with the lowest merit.

Knowing the relative cost and merit of each attribute allows us to evaluate the performance of our classifier using: (a) only low cost attributes, (b) combination of low and medium cost attributes, and (c) all attributes. The classifier accuracy and RRSE are shown in Table 3. It is seen that using only low-cost attributes results in 97.85% accuracy with an RRSE value of 18.63%; the additional use of medium-cost attributes increases accuracy to 99.68% and significantly reduces the RRSE error to 7.7%; while including all attributes yields an overall accuracy of 99.88% and RRSE of 5.06%. The method can therefore be tuned to achieve appropriate balance between attribute collection cost and accuracy/error of classification.

### 6.2 Training the Machine

The duration of the training data set is another source of cost incurred by our classification. In Fig. 12(a), we plot the




(a) Impact of training data. (b) Confidence of classifier. (c) Merit of attributes.

Fig. 12. Operational insights for real-time implementation of our device classifier: (a) impact of training, (b) confidence-level for correct/incorrect classification, and (c) importance of attributes.

accuracy of the classifier on the left y-axis and the RRSE on the right y-axis as a function of the number of days involved in collecting the training data set. Note that the x-axis is in log-scale and each day represents 24 instances.

It can be seen that the classifier achieves an overall accuracy is 99.28% with only one day of training and saturates at 99.76% when trained over 16 days. On the other hand, RRSE drops from 14.43% to 7.5% when the training duration is increased from 1 day to 16 days. It further falls to 5.82% when we train using 70% of all instances from 128 days. As mentioned in §5, the RRSE value is sensitive to the accuracy of individual classes. We therefore believe that if there is a balanced number of instances from various classes, our classifier would perform better in terms of RRSE.

### 6.3 Interpreting the Output of Classifier

As discussed in §5.1, our classifier generates a confidence level during the testing phase. This can be used as a measure of reliability for our classifier. If adequate information is not provided by a test instance then the classifier will choose a random class (as discussed in §5.2.1) with a low confidence level - this can be interpreted as an "unknown" class. For example, given instances with an empty value for the cipher suite attribute, the corresponding stage-0 classifier will output Dropcam class with a confidence value of less than 10% - even for Dropcam instances that are classified correctly the confidence level is low within the same range.

We plot the CCDF of confidence level of our stage-1 classifier in Fig. 12(b) for instances classified as correct and incorrect. It is clearly seen that the confidence level is always below 80% when an instance is incorrectly classified, as shown by the red dotted line - the average confidence level for incorrectly classified instances is 54.22%. On the other hand, our classifier has an average 99.74% confidence level for instances that are correctly classified. We note that for only a negligible fraction of correctly classified instances (i.e. 0.37%) the confidence level is less than 80% as shown by the blue dashed line. This suggests that we can comfortably rely on our classifier's output for a device if it results in a confidence level of greater than 80%, otherwise we need to collect more traffic (and richer instances) from that device in order to increase the confidence level.

To demonstrate the ability of our classifier in detecting changes of normal behavior, we have launched UDP reflection and TCP SYN attacks of varying rates on the Samsung camera. When our classifier is fed these attributes during the attack, it incorrectly identifies the device, but its confidence-level drops to less than 50%. We note that the confidence level is 100% for normal traffic from Samsung camera, as shown in the last column of Table 2. This is taken as a sign of anomalous behavior that warrants further investigation by the network operator.

## 7 Conclusion

Despite the proliferation of IoT devices in smart homes, enterprises, campuses, and cities around the world, operators of such environments lack visibility into what IoT devices are connected to their networks, what their traffic characteristics are, and whether the devices are functioning appropriately free from security compromises. This work is the first to systematically characterize and classify IoT devices at run-time. We instrumented a smart environment with 28 unique IoT devices and collected traffic traces continuously over 26 weeks. We then statistically characterized the traffic in terms of activity cycles, signalling patterns, communication protocols and cipher suites. We developed a multi-stage machine learning based classification framework that uniquely identifies IoT devices with over 99% accuracy. Finally, we evaluated the real-time operational cost, speed, and accuracy trade-offs of our classification method. This paper shows that IoT devices can be identified with high accuracy based on their network behavior, and sets the stage for future work in detecting misbehaviors resulting from security breaches in teh smart environment.

## References

[1] I. Spectrum. (Last accessed July 2017.) Popular Internet of Things forecast of 50 billion devices by 2020 Is outdated. https://goo.gl/6wSUkk.

[2] Cisco, "Cisco 2017 Midyear Cybersecurity Report," Tech. Rep., 2017.

[3] A. Schiffer. (2017) How a fish tank helped hack a casino. https://goo.gl/SAHxCX.

[4] Ms. Smith. (2017) University attacked by its own vending machines, smart light bulbs & 5,000 IoT devices. https://goo.gl/cdNJnE.

[5] S. Alexander and R. Droms, "DHCP Options and BOOTP Vendor Extensions," Internet Requests for Comments, RFC Editor, RFC 2132, March 1997. [Online]. Available: https://tools.ietf.org/rfc/rfc2132.txt

[6] A. Sivanathan *et al.*, "Characterizing and Classifying IoT Traffic in Smart Cities and Campuses," in *Proc. IEEE Infocom Workshop on Smart Cities and Urban Computing*, Atlanta, USA, May 2017.

[7] S. Notra *et al.*, "An Experimental Study of Security and Privacy Risks with Emerging Household Appliances," in *Proc. M2MSec*, Oct 2014.

[8] F. Loi *et al.*, "Systematically Evaluating Security and Privacy for Consumer IoT Devices," in *Proc. ACM CCS workshop on IoT Security and Privacy (IoT S&P)*, Texas, USA, Nov 2017.

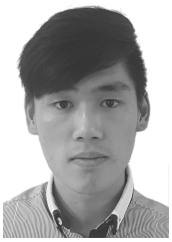
[9] I. Andrea *et al.*, "Internet of Things: Security vulnerabilities and challenges," in *2015 IEEE Symposium on Computers and Communication (ISCC)*, July 2015.

[10] K. Moskvitch, "Securing IoT: In your Smart Home and your Connected Enterprise," *Engineering Technology*, vol. 12, April 2017.

[11] N. Dhanjani, *Abusing the Internet of Things: Blackouts, Freakouts, and Stakeouts*. O'Reilly Media, 2015.
[12] E. Fernandes *et al.*, "Security Analysis of Emerging Smart Home Applications," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, may 2016.
[13] T. guardian. (2016) Why the internet of things is the new magic ingredient for cyber criminals. https://goo.gl/MuH8XS.
[14] T. Yu *et al.*, "Handling a Trillion (Unfixable) Flaws on a Billion Devices: Rethinking Network Security for the Internet-of-Things," in *Proc. ACM HotNets*, Nov 2015.
[15] A. Sivanathan *et al.*, "Low-Cost Flow-Based Security Solutions for Smart-Home IoT Devices," in *Proc. IEEE ANTS*, Nov 2016.
[16] A. Moore and D. Zuev, "Internet Traffic Classification Using Bayesian Analysis Techniques," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 1, pp. 50–60, Jun. 2005.
[17] M. Iliofotou *et al.*, "Exploiting Dynamicity in Graph-based Traffic Analysis: Techniques and Applications," in *Proc. ACM CoNEXT*, Rome, Italy, Dec 2009.
[18] D. Bonfiglio *et al.*, "Revealing Skype Traffic: When Randomness Plays with You," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 37–48, Aug. 2007.
[19] R. Ferdous *et al.*, "On the Use of SVMs to Detect Anomalies in a Stream of SIP Messages," in *Proc. IEEE ICMLA*, Boca Raton, Florida, USA, Dec 2012.
[20] M. Z. Shafiq *et al.*, "A First Look at Cellular Machine-to-Machine Traffic: Large Scale Measurement and Characterization," in *Proc. ACM Sigmetrics*, England, Jun 2012.
[21] N. Nikaein *et al.*, "Simple Traffic Modeling Framework for Machine Type Communication," in *Proc. ISWCS*, Germany, Aug 2013.
[22] M. Jadoul. The IoT: The Network Can Make It or Break It. https://insight.nokia.com/iot-network-can-make-it-or-break-it.
[23] M. Simon and Alcatel-Lucent. Architecting Networks: Supporting IoT.
[24] M. Laner *et al.*, "Traffic Models for Machine Type Communications," in *Proc. ISWCS*, Germany, Aug 2013.
[25] L. Markus *et al.*, *Traffic models for machine-to-machine (M2M) communications: types and applications*, 12 2015.
[26] A. Orrevad. M2M Traffic Characteristics: When Machines Participate in Communication.
[27] M. Lopez-Martin *et al.*, "Network Traffic Classifier with Convolutional and Recurrent Neural Networks for Internet of Things," *IEEE Access*, vol. 5, 2017.
[28] D. Tegeler *et al.*, "BotFinder: Finding Bots in Network Traffic Without Deep Packet Inspection," in *Proc. ACM CoNEXT*, Nice, France, Dec 2012.
[29] D. McGrew and B. Anderson, "Enhanced Telemetry for Encrypted Threat Analytics," in *Proc. IEEE ICNP*, Singapore, Nov 2016.
[30] B. Anderson and D. McGrew, "Identifying Encrypted Malware Traffic with Contextual Flow Data," in *Proc. ACM AISec*, Vienna, Austria, Oct 2016.
[31] Cisco. (2017) joy. https://github.com/cisco/joy.
[32] Y. Meidan *et al.*, "Detection of Unauthorized IoT Devices Using Machine Learning Techniques," *arXiv*, 2017. [Online]. Available: http://arxiv.org/abs/1709.04647
[33] (2016) OpenWrt. https://openwrt.org/.
[34] M. Weiss *et al.* (2015) Time-Aware Applications, Computers, and Communication Systems. http://dx.doi.org/10.6028/NIST.TN.1867.
[35] A. McCallum and K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification," *AAAI/ICML-98 Workshop on Learning for Text Categorization*, pp. 41–48, 1998.
[36] E. Frank *et al.*, *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*, fourth edition ed. Morgan Kaufmann, 2016.
[37] E. Vyncke and C. Paggen, *LAN switch security: what hackers know about your switches*. Cisco Press, 2008.

**Arunan Sivanathan** is currently pursuing his PhD in the School of Electrical and Telecommunication Engineering at the University of New South Wales (UNSW Sydney). He obtained his bachelor's degree from the University of Peradeniya, Sri Lanka in 2012. He later joined the University of Jaffna, Sri Lanka, as a Lecturer from 2013 to 2016. His primary research interests include security of Internet of Things and data analytics on machine-to-machine communication.

**Hassan Habibi Gharakheili** received his B.Sc. and M.Sc. degrees of Electrical Engineering from the Sharif University of Technology in Tehran, Iran in 2001 and 2004 respectively, and his Ph.D. in Electrical Engineering and Telecommunications from UNSW in Sydney, Australia in 2015. He is currently a postdoctoral researcher at UNSW Sydney. His research interests include network architectures, software-defined networking and Internet of Things.



**Franco Loi** is currently studying his Bachelor's degree in Electrical Engineering and Computer Science at the University of New South Wales in Sydney. His research interests are on the network security of IoT.

**Adam Radford** is a Distinguished Systems Engineer at Cisco Systems in Sydney, Australia. His background is software and automation, having spent 10 years building and automating campus networks. He then joined Cisco and has focused on a variety of technologies including voice, wireless and data center. In recent times his focus has been Enterprise networks, specifically automation and programmability. Adam has a first class honors degree in Science, majoring in Computer Science from UNSW.

**Chamith Wijenayake** is currently a Lecturer with the school of Electrical Engineering and Telecommunications at UNSW. He obtained his PhD (Electrical Engineering) from the University of Akron, Ohio, USA in 2014 and B.Sc. (Electronic and Telecom Engineering) from the University of Moratuwa, Sri Lanka in 2007. His research interests include multidimensional space-time signal processing for electronically scanned smart antenna arrays, light field signal processing, local signal approximations, and FPGA based system design for DSP applications.

**Arun Vishwanath** (SM '15, M '11) is a lead research scientist at IBM Research in Melbourne, Australia working in the area of IoT for energy optimization in smart buildings and IoT security. He received the Ph.D. degree in Electrical Engineering from the University of New South Wales, Sydney, Australia in 2011 and was a visiting Ph.D. scholar in the Department of Computer Science, North Carolina State University, USA in 2008. His research interests span the areas of IoT applications, cybersecurity and software defined networking. Arun has received several awards from IBM for outstanding technical accomplishments. He is the recipient of the Best Paper Award at the ACM e-Energy 2018 conference, is appointed Distinguished Speaker of ACM and is a Senior Member of IEEE.

**Vijay Sivaraman** received his B. Tech. from the Indian Institute of Technology in Delhi, India, in 1994, his M.S. from North Carolina State University in 1996, and his Ph.D. from the University of California at Los Angeles in 2000. He has worked at Bell-Labs as a student Fellow, in a silicon valley start-up manufacturing optical switch-routers, and as a Senior Research Engineer at the CSIRO in Australia. He is now a Professor at the University of New South Wales in Sydney, Australia. His research interests include Software Defined Networking, network architectures, and cyber-security particularly for IoT networks.