Descriptor: *Deakin IoT Traffic*

Aleksandar Pasquini[1], Rajesh Vasa[1], Irini Logothetis [1], Hassan Habibi Gharakheili[2], Alexander Chambers[3], and Minh Tran[3]

[1]A2I2, Deakin University, Geelong, VIC 3220, Australia
[2]School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, NSW 2052, Australia
[3]Information Sciences Division, Defence Science & Technology Group, Edinburgh, SA, Australia

Corresponding author: Aleksandar Pasquini (email: aleksandar.pasquini@deakin.edu.au).

This research is supported by the Commonwealth of Australia as represented by the Defence Science and Technology Group of the Department of Defence.

ABSTRACT This work presents the Deakin IoT Traffic dataset comprising of packets emitted by Internet of Things (IoT) devices, each having different functions and behaviors. In total, 112 million packets were collected over 119 days from 24 IoT devices. Of the 24 IoT devices, 19 of them are distinct. The collected packets are stored in PCAP (Packet Capture) files, preserving a wide array of network protocols such as DHCP, ARP, DNS, HTTP, and other TCP/UDP-based applications, which enable a detailed analysis of communication patterns. Alongside the IoT PCAP files, we provide the full PCAP files which contain traffic from 36 non-IoT devices. In addition to the PCAP files, the dataset includes a CSV file that maps each IoT device to its unique MAC address, simplifying device-specific analysis. To assist researchers, Python scripts are provided for extracting and processing packets, demonstrating ways to leverage the data for various applications. This dataset is valuable for a wide range of research purposes: it enables researchers to study and differentiate network behaviors based on device function, supports behavior-based profiling for anomaly detection, and provides a foundation for designing IoT-tailored network policies.

IEEE SOCIETY/COUNCIL Computer Society (CS)

DATA DOI/PID $10.26187/deakin.28013234$

DATA TYPE/LOCATION PCAP files; Deakin University Burwood Campus, Melbourne, Australia

INDEX TERMS IoT, Packet Captures, Network Traffic

BACKGROUND

With the growth and heterogeneity of IoT devices, managing and understanding their behavior is a challenge. A behavior profile describes the expected communication patterns and operational characteristics of a device. They serve as a reference point for what constitutes normal device activity (a baseline). By monitoring device behavior against the baseline, it becomes easier to manage devices and detect anomalies.

To create behavioral profiles, historical traffic data is needed. Our IoT dataset[1] contains packets that are emitted by the IoT devices. Many datasets on IoT traffic exist, as seen in Table 1; however, the type of packets collected differs between them. Two main aspects affect the traffic data. The first is the location where traffic is measured and recorded. Packets collected from within a local network before undergoing network address translation function (pre-NAT) differ from packets collected post-NAT. In typical home network settings, routers perform NAT by changing the IP headers of packets and replacing internal (private) addresses with public IP addresses on the Internet. Only a few datasets collect traffic after the gateway, such as HomeMole [4], IoT-deNat [12], and MON(IOT)R [2] datasets. Some of these datasets also include virtual private network (VPN) traffic, which further changes the packets by encrypting them. The majority, however, collect packets in pre-NAT environments, as network administrators typically have access to this information.

The second aspect is the device's operational phase. Previous studies [13] have categorized three possible phases. The first is the setup phase. This phase is the shortest and begins when the device first connects to a network. A device

[1]Our full traffic capture includes packets from both IoT and non-IoT devices on our testbed, with the IoT dataset representing a subset of this capture.

TABLE 1. Overview of benign IoT PCAP datasets. "Filtered" indicates that the traffic has been transformed in some way, such as by excluding traffic of non-IoT devices in Deakin IoT Traffic. Filtering methods vary across datasets.

Dataset Name	Number of Devices		Device Behaviors			Private or Public	Filtered or Raw	Number of Packets
	IoT	Non-IoT	Setup	Idle	Interaction			
Deakin IoT Traffic	24	0	✓	✓	✓	Public	Filtered	112.84m
Active CICIoT-22 [1]	24	0	-	✓	✓	Public	Filtered	63.57m
MON(IOT)R [2]	81	0	-	✓	✓	Public	Filtered	40.80m
LSIF [3]	22	0	-	✓	✓	Public	Filtered	32.43m
HomeMole-Ind [4]	7	0	-	✓	✓	Public	Filtered	2.75m
IoTFinder [5]	53	0	-	✓	✓	Public	Filtered	2.05m
Benign IoT-23 [6]	3	0	-	✓	✓	Public	Filtered	428k
IoTSentinel [7]	31	0	✓	-	-	Public	Filtered	193k
SHIoT [8]	36	0	-	✓	✓	Private	Raw	555.50m
YourThings [9]	45	9	-	✓	✓	Public	Raw	451.53m
Deakin Full Traffic	24	36	✓	✓	✓	Public	Raw	257.67m
UNSW [10]	28	3	-	✓	✓	Public	Raw	23.81m
Benign ACI IoT [11]	30	21	-	✓	✓	Public	Raw	8.06m

will communicate with its company servers to check for updates and configure itself. After this configuration ends, the device transitions into either idle or interaction phases. If the device senses an event, then that device executes its task and the device's state changes to its interaction phase. This interaction increases the amount of traffic the device generates, and potentially more informative traffic patterns can be found. If there are no events, then the device is in its idle phase. Idle phases produce a minimal amount of traffic. Certain datasets are limited to specific phases of device activity. For example, IoTSentinel [7] exclusively contains device setup traffic, amounting to just 193 thousand packets captured. Similarly, IoTFinder [5] is restricted to DNS traffic alone. This selective focus on particular device phases limits the ability to generate comprehensive profiles of device behavior.

The main limitation of previous datasets is that they do not fully capture all aspects of device behavior. Short capture durations miss rare behaviors, and the datasets do not capture how devices change over time. Both aspects are useful for building behavioral profiles. To address this gap, the Deakin IoT Traffic dataset was created. It only contains pre-NAT, benign IoT traffic captured over 120 days. IoT devices generally have a limited set of functions, which makes it easier to build behavioral profiles. In contrast, non-IoT behavioral profiles are more challenging to construct. To support further research, we provide the full PCAP files for those interested in analyzing non-IoT traffic. Moreover, no malicious (attack) traffic was generated as our primary intent was to capture benign behaviors, which can be well-defined and constrained, unlike attacks. Creating a comprehensive device profile is more useful as any type of malicious attack can be detected as an anomaly. Additionally, capturing packets from all three phases, namely setting up, performing actions, or being idle, is essential, as real-world devices may exhibit distinct patterns during each phase. This comprehensive approach: (a) yields richer device profiles, and (b) enables more diverse applications.

COLLECTION METHODS AND DESIGN

To generate this dataset, we constructed a network designed to collect and monitor traffic. Packet capture was performed on the LAN bridge of the network gateway, enabling the use of packet MAC addresses as unique identifiers. No protocols were excluded, ensuring all types of traffic were recorded. The captured traffic is stored in PCAP files. A PCAP file, short for "packet capture" file, is a data format used to record raw network traffic at the packet level. It contains a sequential log of network packets transmitted over a network during a specific time frame. Each packet in a PCAP file includes detailed information such as source and destination addresses, protocol types, headers, and payload data.

To capture traffic from the devices' setup phase, we added and removed devices from the network over time. To generate interaction traffic, we actively and passively engaged with devices to perform specific tasks. The devices were also left on when no one was in the lab to ensure that idle phase traffic was collected.

A. Network Setup

Fig. 1 shows the network setup in our Cyberlab at Deakin University, and all IoT and non-IoT devices were connected via WiFi. To simulate a realistic network environment, devices were dynamically added and removed, reflecting typical usage patterns. Thus, despite having 24 IoT devices, not all of them were always generating traffic.

Most consumer routers (home gateways) have limited functionality for network data collection by default. To overcome this limitation, we installed OpenWrt [14] on the router. OpenWrt is an open-source firmware that provides control options for routers, transforming standard consumer-grade routers into powerful tools for network monitoring

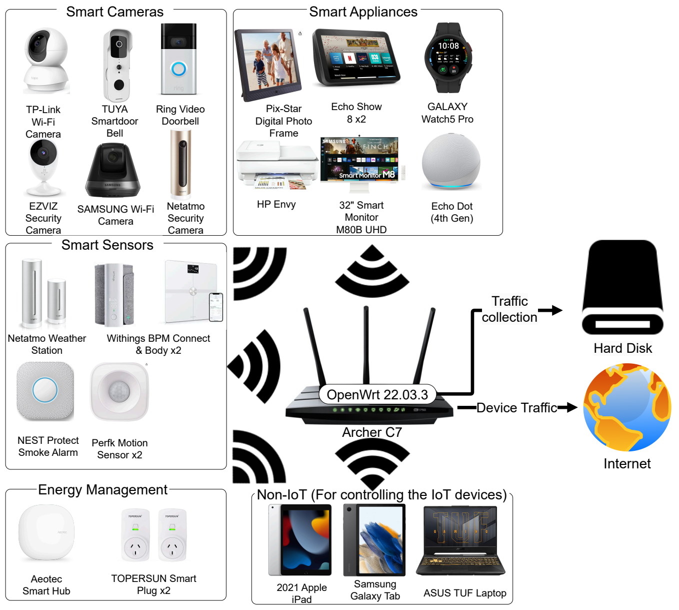


FIGURE 1. **How the Deakin Cyberlab was set up. The non-IoT device traffic is only found in the Full PCAP files.**

Algorithm 1: Daily Packet Capture Script.

```
Set SAVEDIR ← "/mnt/sda1/traffic"
Create directory SAVEDIR if it does not exist
Set DURATION ← $60 \times 60 \times 24$
Compute SECONDSLEFT ← seconds until today's
  23:59:59
Set REPEAT ← $29 \lor 30 \lor 7$
for $i \leftarrow 0$ to REPEAT do
    Get current date components: YYYY, MM, DD
    Set FILENAME ← "YYYY-MM-DD.pcap"
    Run tcpdump with parameters:
        Interface: any
        Output file: SAVEDIR/FILENAME
        Duration: SECONDSLEFT
        File rotation limit: 1
    Display message: "Packet capture for YYYY-MM-DD
      completed."
    Set SECONDSLEFT ←DURATION
end
```

and control. Since OpenWrt is not supported by all routers, we selected the Archer AC1750 for its compatibility. This setup provides complete control over network interfaces and supports packet-capturing tools like `tcpdump`.

B. Traffic Collection

To capture the packets from the devices, we executed the shell script shown in Algorithm 1, which automates the daily process network packet capture using `tcpdump`. The script starts by specifying the directory where the packet capture files will be stored, setting the variable `SAVEDIR` to `/mnt/sda1/traffic`. It ensures the directory exists by running `mkdir -p "$SAVEDIR"`.

To set the duration of each capture, the script calculates the total number of seconds in a day and stores it in the variable `DURATION`. This is done by multiplying 60 seconds by 60 minutes, and then by 24 hours, resulting in 86,400 seconds.

Next, the script calculates the remaining seconds until midnight (23:59:59) by subtracting the current epoch time from the epoch time for the end of the day. This result is stored in the variable `SECONDSLEFT` and is used to determine the duration of the first day's capture, which runs only for the remaining time in the current day. Once the first day concludes, subsequent captures use the predefined duration stored in `DURATION`.

The main functionality of the script is implemented within a loop that iterates `REPEAT` times, enabling daily packet captures for the current day and the subsequent `REPEAT-1` days. During the data collection period, `REPEAT` was manually configured to 30, 29 or 7 depending on the anticipated frequency of our presence in the lab and interactions with the IoT devices. After completing `REPEAT` days, the script must be restarted manually, and the captured packets on the hard drive are moved to backup locations. During each iteration, the script updates the date variables (`YYYY`, `MM`, and `DD`) to reflect the current date at execution. It then generates a unique filename (`FILENAME`) in the `YYYY-MM-DD` format to identify each capture file by its corresponding date.

The packet capture is initiated with the `tcpdump` command, configured with the following options:

- `-i any`: Captures packets on all network interfaces.
- `-w "$SAVEDIR/$FILENAME"`: Specifies the output file path for the captured packets.
- `-G "$SECONDSLEFT"`: Sets the duration in seconds for the capture. For the first iteration, this value indicates the remaining seconds in the current day; for subsequent iterations, it is a full day (86,400 seconds).
- `-W 1`: Limits the output to one file per capture session.

Upon completion of each capture, the script outputs a message using `echo` to indicate that the capture for the specific day is complete. After the first capture, the `SECONDSLEFT` variable is reset to `DURATION`, ensuring subsequent captures run for a full 24-hour period. This script was repeated until 119 PCAP files were generated. During the collection process, devices were set up and interacted with. This metadata was recorded in CSV files.

C. Postprocessing

We stored the MAC address of IoT devices in a CSV file named `macAddresses.csv`. These addresses are listed in the second column of Table 2, along with their total packet count (sorted in descending order), their commercial name, and the date they sent their first packet ("first seen" date). From the table, it can be seen that the Samsung camera sent the

TABLE 2. The list of IoT devices in the Deakin IoT Traffic dataset and basic statistics.

First-Seen date (YYYY-MM-DD)	Device MAC address	IoT device commercial name	# Packets sent
2023-07-11	00:16:6c:d7:d5:f9	SAMSUNG Pan/Tilt 1080P Wi-Fi Camera	20.8m
2023-05-17	18:48:be:31:4b:49	Echo Show 8	18.9m
2023-05-18	70:ee:50:96:bb:dc	Netatmo Weather Station	11.3m
2023-05-17	10:5a:17:b8:a2:0b	TOPERSUN Smart Plug	10.8m
2023-07-07	70:ee:50:57:95:29	Netatmo Smart Indoor Security Camera	9.6m
2023-05-17	10:5a:17:b8:9f:70	TOPERSUN Smart Plug	9.5m
2023-05-15	40:f6:bc:bc:89:7b	Echo Dot (4th Gen)	7.5m
2024-03-27	74:d4:23:32:a2:d7	Echo Show 8	5.2m
2024-03-27	68:3a:48:0d:d4:1c	Aeotec Smart Hub	5.3m
2023-07-18	84:69:93:27:ad:35	HP Envy	3.3m
2023-05-15	70:09:71:9d:ad:10	32” Smart Monitor M80B UHD	2.6m
2023-05-17	90:48:6c:08:da:8a	Ring Video Doorbell	2.3m
2023-05-22	54:af:97:bb:8d:8f	TP-Link Tapo Pan/Tilt Wi-Fi Camera	1.2m
2023-05-22	40:ac:bf:29:04:d4	EZVIZ Security Camera	1.1m
2023-07-13	b0:02:47:6f:63:37	Pix-Star Easy Digital Photo Frame	1.1m
2023-05-18	70:3a:2d:4a:48:e2	TUYA Smart Doorbell	1.0m
2024-04-18	6e:fe:2f:5a:d7:7e	GALAXY Watch5 Pro	303.1k
2023-05-29	1c:90:ff:bf:89:46	Perfk Motion Sensor	92.8k
2023-05-29	fc:67:1f:53:fa:6e	Perfk Motion Sensor	79.6k
2023-05-17	cc:a7:c1:6a:b5:78	NEST Protect Smoke Alarm	12.3k
2024-03-27	00:24:e4:e4:55:26	Withings Body+ (Scales)	7947
2024-03-27	00:24:e4:f7:ee:ac	Withings Connect (Blood Pressure)	2702
2023-05-15	00:24:e4:e3:15:6e	Withings Body+ (Scales)	2324
2023-05-15	00:24:e4:f6:91:38	Withings Connect (Blood Pressure)	510

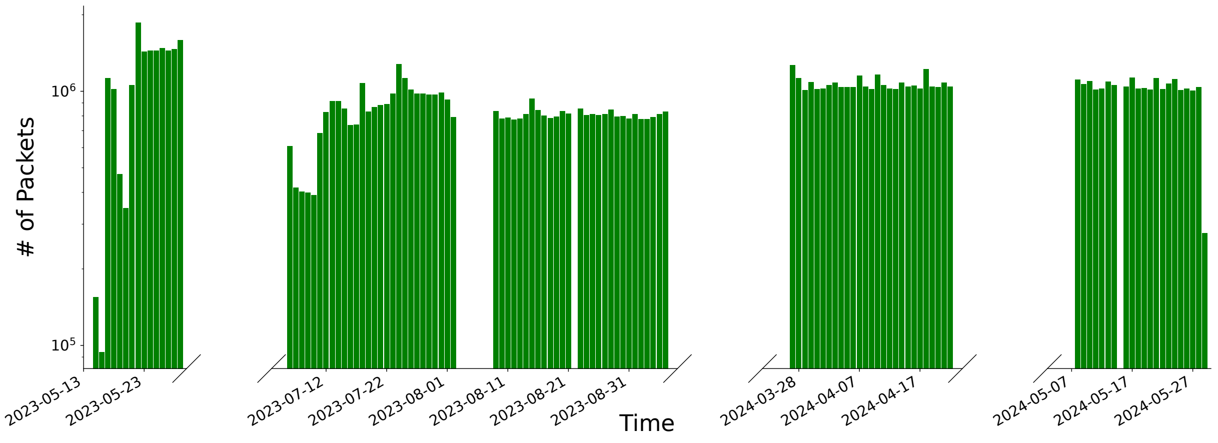


FIGURE 2. Packet count per daily IoT PCAP file.

highest number of packets, which is double the amount sent by the Netatmo camera. Also, despite one of the Withings scales being added almost a year earlier, it sent fewer packets than the newer one. This indicates that the newer scale was interacted with more than the older one.

We then separated the non-IoT devices from the IoT by creating a pure IoT traffic dataset. Any packets originating from MAC addresses not listed in `macAddresses.csv` were removed from the IoT dataset, as they are assumed to come from non-IoT devices. Packets with matching source MAC addresses were saved to a new file, prefixed with “IoT_”.

Once this processing was complete, the Deakin IoT Traffic dataset was finalized. The time trace of packets collected is shown in Fig. 2. Initially, the network contained a few devices, resulting in a lower volume of captured packets. After 22 July 2023, the network maintained a minimum of 12 IoT devices, which resulted in a consistent level of traffic being recorded in the PCAP files.

TABLE 3. **Active interaction times. These time periods indicate the device was interacted with to achieve its main function, *e.g.,* stream a video to the Smart Monitor M80B UHD or press the doorbell on the Ring Video Doorbell.**

IoT device	Date	Start	End
32” Smart Monitor M80B UHD	2023-07-06	13:27	14:05
Echo Dot (4th Gen)	2023-07-06	13:20	14:05
32” Smart Monitor M80B UHD	2023-07-11	13:31	16:15
Echo Show 8	2023-07-11	13:39	13:40
Netatmo Smart Indoor Security Camera	2023-07-11	13:31	16:15
Samsung Smart Cam	2023-07-11	14:14	14:25
Ring Video Doorbell	2023-07-11	16:03	16:05
Netatmo Smart Indoor Security Camera	2023-07-12	14:19	16:03
32” Smart Monitor M80B UHD	2023-07-12	14:25	16:03
TP-Link Tapo Pan/Tilt Wi-Fi Camera	2023-07-12	14:52	14:53
Perfk Motion Sensor	2023-07-12	15:20	15:25
32” Smart Monitor M80B UHD	2023-07-13	15:12	16:00
32” Smart Monitor M80B UHD	2023-07-18	14:04	16:20
32” Smart Monitor M80B UHD	2023-07-24	13:42	16:02
Pix-Star Easy Digital Photo Frame	2023-07-24	15:35	15:40
Echo Show 8	2023-07-25	14:41	16:25

Three additional CSV files were created to store the generated metadata. `activeInteractions.csv` contains the times and dates when devices were actively interacted with. `passiveInteractions.csv` logs the times and dates when human users were present in the lab and possibly engaging with the environment and devices in a passive manner. Finally, `setupTimes.csv` contains the times and dates when the devices were first added to the network.

VALIDATION AND QUALITY

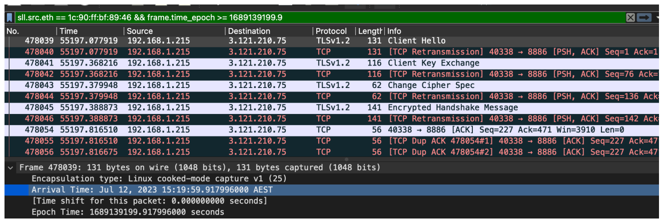
To demonstrate the value of the Deakin IoT Traffic dataset, we conducted three experiments. Each experiment uses data from different parts of the dataset.

Interaction Traffic Analysis

Table 3 shows the recorded times when the devices were interacted with. Note that it is not a full of list interactions (See Missing Annotation and Data for more details). Nevertheless, our traffic traces data can be used to identify other periods of device interactions.

For example, the Perfk Motion Sensor (highlighted by a yellow shade) recorded a user interaction at 15:20 on the 12th of July 2023, lasting approximately five minutes. Analyzing the `IoT_2023-07-12.pcap` file during that time reveals that the Perfk Motion Sensor sent TLSv1.2 packets to `3.121.210.75`, as shown in Fig. 3. This IP address belongs to Tuya Smart, the manufacturer of the motion sensor. This specific traffic likely correlates with motion detection, as this sensor does not send TLSv1.2 packets to `3.121.210.75` at other times of the day. From this observation, we constructed the following command (pattern) to identify other occasions when the Motion Sensor was interacted with:

```
sll.src.eth==1c:90:ff:bf:89:46 && ip.dst==3.121.210.75
```



```
sll.src.eth == 1c:90:ff:bf:89:46 && frame.time_epoch >= 1689139199.9
No.     Time          Source         Destination   Protocol  Length Info
 478039 55197.077919  192.168.1.215  3.121.210.75  TLSv1.2   131 Client Hello
 478040 55197.077919  192.168.1.215  3.121.210.75  TCP       131 [TCP Retransmission] 40338 → 8886 [PSH, ACK] Seq=1 Ack=1
 478041 55197.368216  192.168.1.215  3.121.210.75  TLSv1.2   116 Client Key Exchange
 478042 55197.368216  192.168.1.215  3.121.210.75  TCP       116 [TCP Retransmission] 40338 → 8886 [PSH, ACK] Seq=76 Ack=
 478043 55197.379948  192.168.1.215  3.121.210.75  TLSv1.2    62 Change Cipher Spec
 478044 55197.379948  192.168.1.215  3.121.210.75  TCP        62 [TCP Retransmission] 40338 → 8886 [PSH, ACK] Seq=136 Ack
 478045 55197.388873  192.168.1.215  3.121.210.75  TLSv1.2   141 Encrypted Handshake Message
 478046 55197.388873  192.168.1.215  3.121.210.75  TCP       141 [TCP Retransmission] 40338 → 8886 [PSH, ACK] Seq=142 Ack
 478054 55197.816510  192.168.1.215  3.121.210.75  TCP        56 40338 → 8886 [ACK] Seq=227 Ack=471 Win=3910 Len=0
 478055 55197.816510  192.168.1.215  3.121.210.75  TCP        56 [TCP Dup ACK 478054#1] 40338 → 8886 [ACK] Seq=227 Ack=47
 478056 55197.816675  192.168.1.215  3.121.210.75  TCP        56 [TCP Dup ACK 478054#2] 40338 → 8886 [ACK] Seq=227 Ack=47
⌄ Frame 478039: 131 bytes on wire (1048 bits), 131 bytes captured (1048 bits)
    Encapsulation type: Linux cooked-mode capture v1 (25)
    Arrival Time: Jul 12, 2023 15:19:59.917996000 AEST
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1689139199.917996000 seconds
```

FIGURE 3. **Packets sent by the Perfk Motion Sensor after interaction.**

We tested this pattern by applying it to a traffic trace from Sunday (`IoT_2023-07-16.pcap`). It yielded no packets, as there was no motion in our lab on that day. However, when applied to `IoT_2023-07-18.pcap` (Tuesday), it returns packets starting at 15:32. This timing aligns with Table 4, which records the presence of a human in the lab at that time. Future work could build upon this by developing a model to detect transitions between idle and interaction states of a device.

Behavioral Profiles with PCAP Traffic

For our second experiment, a filtered PCAP file was used to create a behavior profile for three representative IoT devices. We choose an Echo Show 8, a Netatmo Smart Indoor Security Camera, and the 32” Smart Monitor M80B UHD. These devices have distinct functions that will make the process of generating profiles easier.

To create these profiles, we employed two different machine learning algorithms implemented with scikit-learn [15]: an unsupervised Isolation Forest and a supervised Random Forest. For Isolation Forest, we trained a separate model for each device type and then combined the three models into an ensemble for classification. All hyperparameters were left at their default settings, except for the contamination parameter for the Isolation Forest models, which was set to 0.001. This was chosen because the dataset only contains benign traffic, ensuring that the models classify the training data with 99.9% accuracy. Five traffic features were extracted to prepare the data for the models.

- f_1 **Packet Length:** The total length of the packet (headers and payload) in bytes.
- f_2 **Protocol Type:** If the packet contains an IP layer, the protocol number is extracted and stored. If the IP layer is absent, a value of -1 is assigned to this feature.
- f_3 **Source and Destination Ports:** If the packet contains a TCP or UDP layer, the source port (`sport`) and destination port (`dport`) are extracted and stored. If neither TCP nor UDP layers are present, values of -1 are assigned for both source and destination ports.
- f_4 **Payload Length:** The length of the Cooked Linux payload of each packet in bytes.

This feature extraction ensures that the feature vector maintains a consistent length for each packet, which is required by machine learning algorithms for their input. We trained the models on 70% of the dataset and applied

TABLE 4. Records of passive interaction times during which the presence of human users in the lab caused an "Environment Change", leading to passive interactions with devices like cameras and motion sensors.

Date (YYYY-MM-DD)	Start	End
2023-07-13	13:00	16:00
2023-07-18	13:43	16:25
2023-07-24	12:59	16:02
2023-07-25	13:03	16:25
2023-08-10	12:50	16:00
2023-08-15	13:00	16:00
2023-08-16	13:04	14:54
2023-08-17	13:44	15:59
2023-08-21	12:21	15:56
2023-08-22	13:13	16:17
2023-08-23	11:52	16:09
2023-08-28	12:43	15:57
2023-08-29	13:13	16:13
2023-08-30	14:00	16:15
2023-09-05	13:14	16:12
2023-09-06	12:13	16:00
2023-09-07	15:10	16:30

them to the remaining 30%. The Isolation Forest models were ensembled by assigning a score from each model's decision function to each instance, and the class with the highest score was assigned to the instance. The resulting test accuracies were 98% for the Random Forest and 83% for the ensembled Isolation Forest. This shows that the PCAP files provide sufficient information for the models to build effective behavioral profiles. This experiment can be extended by adding more devices and testing how well the profiles perform for malicious packet detection.

Equations to Distinguish IoT from non-IoT Devices

Our final experiment examined the full PCAP files. We chose to use a Kolmogorov-Arnold Networks (KANs) [16] to distinguish between the IoT and non-IoT devices. KANs are neural networks that utilize the Kolmogorov-Arnold representation theorem, which states that any continuous multivariate function can be represented as a finite sum of continuous univariate functions. In these networks, the traditional weights are replaced with learnable spline functions on edges. This allows each connection to adapt not only in magnitude but the form of the transformation it applies, enabling non-linear mappings with fewer parameters. We can then extract the univariate equation and use it to approximate complex functions, such as whether a device is IoT or not.

For this experiment, we used a single PCAP file (*i.e.,* `2023-08-30.pcap`). The packets needed to be transformed into a form compatible with the KAN. To achieve this, we again created feature vectors but with different features specific to this experiment. Also, to reduce the amount of data, we selected only packets with an IP layer. From each IP packet, we extracted the following:

TABLE 5. Device Setup times. Echo Dot (4th Gen) was set up twice.

IoT device	Date	Start	End
Echo Dot (4th Gen)	2023-05-15	15:30	15:33
Withings Body+ (Scales)	2023-05-15	15:45	15:47
Withings Connect (Blood Pressure)	2023-05-15	15:49	15:50
Netatmo Weather Station	2023-07-05	15:42	15:45
32" Smart Monitor M80B UHD	2023-07-05	16:19	16:28
Echo Dot (4th Gen)	2023-07-05	16:33	16:36
Samsung Smart Cam	2023-07-11	14:11	14:14
Ring Video Doorbell	2023-07-11	15:54	16:03
TP-Link Tapo Pan/Tilt Wi-Fi Camera	2023-07-12	14:47	14:51
Perfk Motion Sensor	2023-07-12	15:20	15:23
TOPERSUN Smart Plug	2023-07-12	15:19	15:22
Pix-Star Easy Digital Photo Frame	2023-07-13	15:26	15:38
HP Envy	2023-07-18	16:10	16:30

x_1 **Packet Length**: The total length of the packet in bytes. It indicates whether the packet carries a large payload or is simply a control message.

x_2 **Protocol Number**: The protocol number specifies the protocol used at the transport layer, such as TCP (protocol number 6) or UDP (protocol number 17). If it is not available, it gets set to zero.

x_3 **Time-to-Live (TTL)**: The TTL value indicates the maximum number of hops a packet can make on a network before it is discarded.

x_4 **Window Size**: The window size is a TCP header field that specifies the size of the sender's receive window. It indicates how much data (in bytes) the sender is willing to receive before expecting an acknowledgment. If it is not available, it gets set to zero.

x_5 **Destination Port**: The destination port number identifies the specific application or service the packet is intended for, such as HTTP (port 80), HTTPS (port 443), or DNS (port 53).

x_6 **Payload Hash**: If the packet had a payload, we computed the MD5 hash of the payload bytes, converted the hash to an integer, and normalized it to a float between 0 and 1. This value was rounded to 12 decimal places. If there was no payload, this feature was set to zero.

After training the KAN on a 80% random split of the PCAP file (comprising 52% IoT packets and 48% non-IoT packets), we derived the following equations:

Non-IoT equation:

$$-0.0441x_1 - 0.0519x_2 - 0.0669x_3 - 0.0933x_4 + 0.1752x_5 + \frac{2.1409}{9.979 - 3.392x_6} - 0.1914$$

IoT equation:

$$-0.043x_1 - 0.0519x_2 - 0.0597x_3 - 0.0932x_4 + 0.1751x_5 - 0.4885\cos(0.5922x_6 + 0.4976) + 0.4297$$

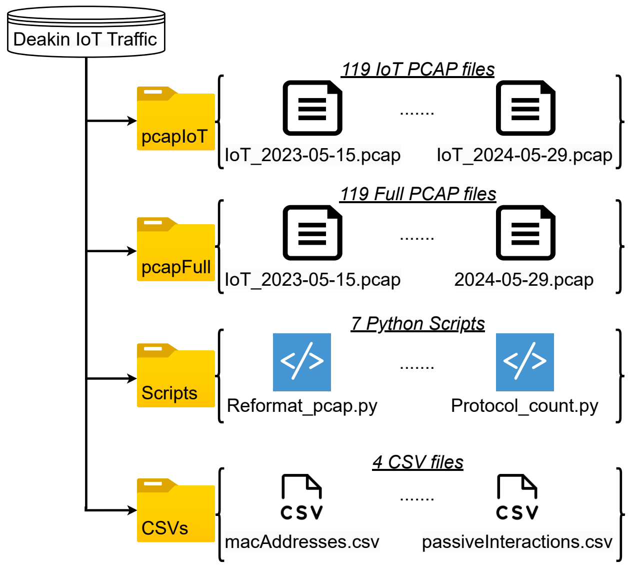


FIGURE 4. The structure of the Deakin IoT Traffic dataset.

From these equations, we observe that for the first five features (*i.e.,* x_1, ... , x_5), there is almost no difference between an IoT packet and a non-IoT one. The main differentiator is the Payload Hash (*i.e.,* x_6). IoT and non-IoT devices produce different hashes because the data structures and communication protocols found in the payload are different.

Applying these equations to the test dataset resulted in an accuracy of 80%. For each packet, both equations are evaluated, and the one producing the higher numerical output determines the packet's class. Specifically, 75% of the non-IoT packets were correctly classified, while the IoT equation correctly identified 85% of the IoT packets. This shows that while the payloads of IoT and non-IoT devices differ, IoT payloads closely resemble other IoT payloads, and non-IoT payloads similarly resemble other non-IoT payloads. Future work could expand on this by developing equations for different classes of IoT devices or creating behavioral profiles for non-IoT devices.

RECORDS AND STORAGE

Fig. 4 shows the structure of the Deakin IoT Traffic dataset comprising four components. The first component is the `pcapIoT` folder. The PCAP files with "IoT" in their filename have been filtered to only include packets that have been emitted by an IoT device.

The second component is the `pcapFull` directory. It contains the full PCAP files without any filtering. This means that these files also contain traffic traces of non-IoT devices. However, the first 32 days of the Full PCAP dataset only contain IoT traffic. This is because the testbed was configured to focus solely on IoT traffic during the first month. After that period, all traffic (IoT and non-IoT) was recorded.

The third component is the Scripts directory. This folder contains seven Python scripts that perform different tasks. The scripts are as follows:

1) `Stats.py`: Generates the high-level statistics (*e.g.,* total packet counts - Fig. 2, average number of packets per file) for the dataset.
2) `removeNonIoT.py`: Removes non-IoT packets from the PCAP files and creates the filtered IoT PCAP files.
3) `reformatPcap.py`: Reformats the PCAP files from date-based to device-based. A new directory will be created for each device and it will include PCAP files containing packets exclusively from that device.
4) `protocolCount.py`: Counts the number of protocols used in the dataset and was used to help generate Fig. 5.
5) `packetCount.py`: Counts the number of packets used in the dataset and was used to help generate Table. 1.
6) `oneClassVsMultiClass.py`: Runs the ensemble isolation forest and the random forest models and outputs their respective accuracies for 3 classes.
7) `IoTOrNonIoT.py`: Runs the KAN model and outputs symbolic equations and the equations' accuracies.

The final component consists of four CSV files that provide metadata for the Deakin IoT Traffic dataset. `macAddresses.csv` contains two columns: one for the MAC address and another for the associated device, as shown in Table 2. The `activeInteractions.csv` file contains the start and end times during which a device was actively interacted with. Its structure is illustrated in Table 3. Similarly, the `setupTimes.csv` and `passiveInteractions.csv` files share a similar layout, but they capture information related to device setup events and passive interactions, respectively. Their information can be seen in Tables 4 and 5.

INSIGHTS AND NOTES

The following notes need to be considered if this dataset is used.

Daylight Savings

Our data-collection script treated all days as 24-hour periods. However, researchers using our data should note the occurrence of daylight saving time in Australia during the collection period: on Sunday, 1 October 2023, clocks moved forward one hour at 2am, and on Sunday, 7 April 2024, clocks moved backward one hour at 3am. These daylight saving time changes may impact how certain temporal patterns in the traffic traces are interpreted.

Missing Annotation and Data

We have ground-truth records for a subset of the interactions that human users had with the IoT devices and/or the lab environment. However, since our lab space was shared with other researchers, it is highly likely that additional interactions occurred but were not recorded. Furthermore, some devices were added to the testbed without recording their setup times. Additionally, it is important to note that data collection was intermittently disrupted when the router was turned off during both scheduled and unscheduled power

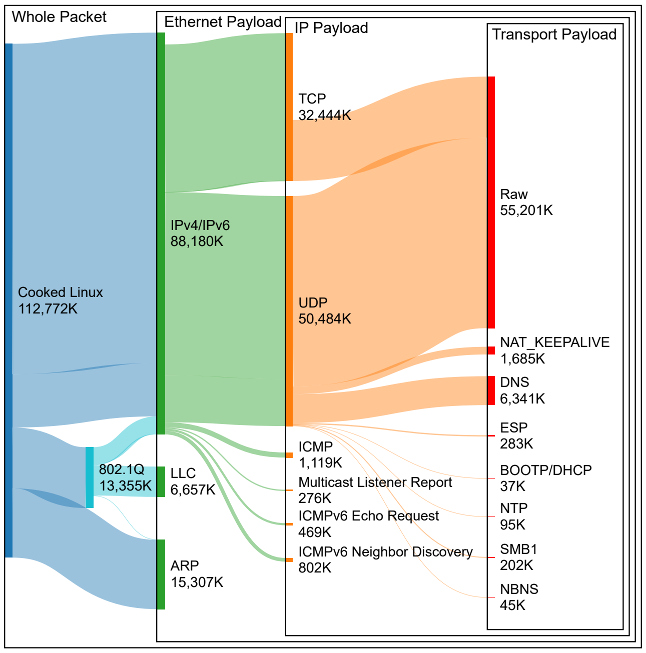


FIGURE 5. Protocol distribution in the Deakin IoT Traffic dataset, with protocols appearing fewer than 10,000 times omitted.

outages, requiring a system reboot manually to resume operations. Consequently, this resulted in gaps of hours in the traffic data, and thus, not all daily PCAP files conclude at 23:59.

Protocols

A breakdown of the protocols used in the dataset can be seen in Fig. 5. Note that the usual Ethernet layer is replaced with "Cooked Linux" layer (SLL). This layer is added to every packet by `tcpdump` when used with the `-i any` option. The SLL header provides a standardized format for packet metadata in cases where Ethernet headers are absent, such as with loopback interfaces (lo) and certain virtual interfaces. This format allows for the capture of all traffic across heterogeneous interface types. The SSL structure includes a pseudo-header that specifies the packet origin and type but omits the destination MAC address.

Despite these differences, key information, such as the source MAC address, remains accessible within the Cooked Linux capture layer, allowing downstream tasks such as anomaly detection, network behavior analysis and machine learning benchmarking to proceed without adaptation. The absence of the destination MAC address, however, may affect applications relying directly on the complete Ethernet header structure and adjustment might be necessary.

SOURCE CODE AND SCRIPTS

The following third-party software was used to create the Deakin IoT Traffic dataset:

1) OpenWrt 22.03.3 [14]
2) Tcpdump 4.9.3 [17]
3) Scapy 2.6.1 [18]
4) Python 3.10 [19]

All our Python scripts and PCAP files can be downloaded at this link: Deakin IoT Traffic. The only requirement for usage is that you cite this paper.

ACKNOWLEDGEMENTS AND INTERESTS

A. Pasquini curated and analyzed the data, performed the experiments, and wrote parts of the manuscript. R. Vasa, I. Logothetis and H. Habibi Gharakheili reviewed the curation and wrote parts of the manuscript. All authors reviewed the manuscript.
This research is supported by the Commonwealth of Australia as represented by the Defence Science and Technology Group of the Department of Defence.
The authors have declared no conflicts of interest.

REFERENCES

[1] S. Dadkhah *et al.*, "Towards the Development of a Realistic Multidimensional IoT Profiling Dataset," in *Proc. IEEE PST*, Fredericton, NB, Canada, Aug 2022.
[2] J. Ren *et al.*, "Information Exposure for Consumer IoT Devices: A Multidimensional, Network-Informed Measurement Approach," in *Proc. IMC*, Amsterdam, Netherlands, Oct 2019.
[3] B. Charyyev and M. H. Gunes, "Locality-Sensitive IoT Network Traffic Fingerprinting for Device Identification," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1272–1281, 2020.
[4] S. Dong *et al.*, "Your Smart Home Can't Keep a Secret: Towards Automated Fingerprinting of IoT Traffic," in *Proc. ACM Asia-CCS*, Taipei, Taiwan, Oct 2020.
[5] R. Perdisci *et al.*, "Iotfinder: Efficient Large-Scale Identification of IoT Devices via Passive DNS Traffic Analysis," in *Proc. IEEE EuroS&P*, Genoa, Italy, Sep 2020.
[6] Stratosphere, "Stratosphere laboratory datasets," 2015, retrieved March 13, 2020, from https://www.stratosphereips.org/datasets-overview.
[7] M. Miettinen *et al.*, "IoT SENTINEL: Automated Device-Type Identification for Security Enforcement in IoT," in *Proc. IEEE ICDS*, Atlanta, GA, USA, Jul 2017.
[8] I. Cvitić *et al.*, "Ensemble Machine Learning Approach for Classification of IoT Devices in Smart Home," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 11, pp. 3179–3202, 2021.
[9] O. Alrawi *et al.*, "Sok: Security Evaluation of Home-Based IoT Deployments," in *Proc. IEEE S&P*, San Francisco, USA, May 2019.
[10] A. Sivanathan *et al.*, "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, Aug 2019.
[11] N. Bastian *et al.*, "ACI IoT Network Traffic Dataset 2023," 2023. [Online]. Available: https://dx.doi.org/10.21227/qacj-3x32
[12] Y. Meidan *et al.*, "A Novel Approach for Detecting Vulnerable IoT Devices Connected Behind a Home NAT," *Computers & Security*, vol. 97, p. 101968, 2020.
[13] H. Jmila *et al.*, "A Survey of Smart Home IoT Device Classification Using Machine Learning-Based Network Traffic Analysis," *IEEE Access*, vol. 10, pp. 97 117–97 141, 2022.
[14] Open source project. (2023) OpenWrt. [Online]. Available: https://openwrt.org
[15] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[16] Z. Liu *et al.*, "KAN: Kolmogorov-Arnold Networks," *arXiv preprint arXiv:2404.19756*, 2024.
[17] Tcpdump Team. (2023) Tcpdump. [Online]. Available: https://www.tcpdump.org
[18] P. Biondi *et al.* (2023) Scapy. [Online]. Available: https://scapy.net
[19] Open source project. (2023) Python. [Online]. Available: https://www.python.org