# Development of a Multimodal Testbed for Dataset Collection in Detecting DGA-based Botnet Attacks Using DNS Queries, Network Traffic, and CPU Power Consumption

Zul-Azri Ibrahim[1], Saiful Adli Ismail[2], Fiza Abdul Rahim[3], Salman Yussof[4], Muhammad Idris Khairul Anuar[5], Aiman Harith Azwan[6], Muhammad Hazim Abas[7]

[1,4,5,6,7]*College of Computing and Informatics, Universiti Tenaga Nasional, Malaysia*

[2,3]*Faculty of Artificial Intelligence, Universiti Teknologi Malaysia, Malaysia*

[1]*zulazri@uniten.edu.my*, [2]*saifuladli@utm.my*, [3]*fiza.abdulrahim@utm.my*, [4]*salman@uniten.edu.my*, [5]*cs01081958@student.uniten.edu.my*, [6]*cs01081907@student.uniten.edu.my*, [7]*cs01081966@student.uniten.edu.my*

**Article history**

Received: 5 April 2026

Received in revised form: 17 April 2026

Accepted: 25 May 2026

Published online: 15 June 2026

*Corresponding author zulazri@uniten.edu.my

***Abstract***

*The detection of Domain Generation Algorithm (DGA)-based botnets in Internet of Things (IoT) environments poses significant challenges due to the dynamic and evasive nature of these botnets. Traditional detection approaches that rely primarily on single data sources such as DNS logs or network traffic often fail to capture the complex, multi-stage behavior of modern botnets. This research presents the development of a dedicated IoT-focused testbed designed to capture a comprehensive multimodal dataset integrating DNS query logs, network traffic data, and CPU power consumption. The primary contribution of this work is the creation of a phase-labelled dataset that categorizes data into three distinct stages: Normal operation, Command and Control (C&C) communication, and Attack execution. This structured labelling provides valuable temporal insights into botnet behavior and supports the development of machine learning models with early-stage detection capability, potentially identifying threats during the C&C phase, before the attacks are launched. Initial analysis and visualizations of the collected data reveal distinct behavioral patterns across power consumption, DNS activity, and network traffic. Notable findings include the identification of high correlated features within the network traffic and DNS query datasets, together with observable phase-dependent variations in CPU power consumption corresponding to different botnet activity stages. These insights suggest that integrating diverse data modalities can significantly enhance the accuracy and robustness of botnet detection in IoT environments.*

***Keywords:*** *Dataset, DGA Botnet, Multimodal, IoT, Testbed*

* *Corresponding author. zulazri@uniten.edu.my*

## 1. Introduction

Domain Generation Algorithm (DGA)-based botnets are a type of malware that automatically generate large numbers of domain names to maintain communication with their command-and-control (C&C) server [1]. This technique enables these botnets to bypass traditional cybersecurity prevention mechanisms such as blacklists of malicious domains. Even if a subset of these domains is detected and blocked, the botnet can still adapt and continue its operations using newly generated domains. This adaptability makes the detection and mitigation of DGA botnets difficult, creating risks for both enterprise and Internet of Things (IoT) environments.

Despite progress in areas such as network monitoring, signature-based detection, and threat intelligence, distinguishing DGA traffic remains a challenge. These botnets often disguise their activity within legitimate DNS requests [2]. Existing methods that depend only on single data sources like DNS logs or network traffic can work in certain situations. However, they often miss the more subtle multi-stage communication patterns utilized by DGA botnets, especially when attacks target IoT environments that lack solid security foundations [3]. This limitation becomes apparent when considering resource consumption and device-level anomalies [4].

The proliferation of IoT devices has increased the complexity of mitigating botnet-related threats. Large-scale deployment of IoT devices has expanded the potential attack surface and has increased the risk of distributed attacks. DGA-based botnets targeting IoT ecosystems and critical infrastructure highlight the demand for more comprehensive detection approaches that can incorporate and analyze multimodal data [3]. The primary objective of this work is to produce a multimodal dataset that serves as a foundational resource for future research in IoT botnet detection. The dataset allows the combination of traffic analysis, DNS behavior, and power consumption monitoring, thereby providing a more holistic view of botnet behavior. Furthermore, the dataset supports the development of multimodal machine learning models capable of detecting subtle deviations that would be overlooked using single-source analysis. This comprehensive approach is expected to improve both accuracy and timeliness of DGA-based botnet detection, ultimately enhancing cybersecurity resilience in IoT ecosystems.

In this study, a testbed is developed to simulate DGA based botnet attacks on IoT devices and capture a comprehensive dataset that spans network traffic, DNS queries, and CPU power consumption. The testbed is designed to represent the full lifecycle of a DGA botnet, from initial infection and C&C communication to the attack phase, enabling phase-based labelling into Normal, C&C, and Attack categories. This phase-based labelling facilitates supervised machine learning training and supports the exploration of early detection techniques that can identify compromised devices before the attack is launched. By combining different modalities, the dataset offers a holistic view of botnet behavior, enabling the development of multimodal machine learning models capable of detecting subtle deviations that would be overlooked using single-source analysis.

The remainder of this paper is structured as follows. Section 2 reviews related studies and outlines the research gaps. Section 3 presents testbed design and architecture. Section 4 presents the experiment design and setup, while Section 5 explains the dataset composition together with the labelling strategy. Section 6

outlines the testbed results and analysis. Finally, Section 7 concludes the paper and discusses the future directions.

## 2. Background and Related Work

The detection of malware, particularly DGA-based botnets, remains a critical area of cybersecurity research. Traditional botnet detection approaches often rely on single-modal datasets, primarily consisting of network traffic, DNS logs, or system-based indicators such as CPU and memory usage. Several datasets have been developed to facilitate machine learning-based intrusion detection in this context. The MedBIoT dataset [5] focuses on network traffic in a controlled IoT environment of 83 devices, capturing early botnet activities including infection and Command and Control (C&C) communication using Mirai, BashLite, and Torii. Similarly, the dataset by [6] addresses existing dataset limitations by providing diverse attack scenarios, including DoS, DDoS, and data exfiltration, within a virtualized IoT testbed. The CICIoT2023 dataset [7] focuses on attack diversity, incorporating 33 attack types across seven categories generated in a 105 devices IoT network, enhancing support for robust intrusion detection models.

Stratosphere Lab's IoT-23 dataset [8] offers 23 labeled network traffic captures covering malware families like Mirai, Gafgyt, and Tsunami, but lacks extensive coverage of advanced malware using encrypted communications or sophisticated DGAs. In the DNS-focused domain, [9] presented the "10 Days DNS Network Traffic" dataset capturing real-world DNS behavior, although its partial availability limits comprehensive analysis. The UMUDGA dataset [10] provides over 30 million labeled algorithmically generated domains (AGDs) enriched with lexical and NLP features for DGA detection, while the UTL_DGA22 dataset [11] extends coverage to over 70 DGA families, although both datasets are limited by their static nature and potential language biases.

Beyond network and DNS analysis, [12] explored power-based anomaly detection using an 8-layer CNN to classify device states based on power consumption patterns, achieving promising accuracy but constrained by the limited range of botnet families tested. [13] proposed a multimodal approach integrating power consumption and network traffic data, demonstrating improved detection performance with Random Forest models, albeit requiring specialized hardware. Similarly, [14] adopted a multimodal strategy combining energy consumption and network behavior from six IoT devices infected with Mirai, Ufonet, and RouterSploit, achieving up to 99.91% accuracy, highlighting the efficacy of multimodal data fusion for enhanced IoT botnet detection but not DGA-based malware. As shown in Table 1, most existing datasets rely predominantly on single-modal data, particularly network traffic, as exemplified by the datasets of [5], [6], [7] and [8]. While these datasets provide valuable insights into communication patterns, they generally lack system-level context, which makes them susceptible to evasion through encryption and traffic obfuscation. Other datasets, such as those proposed by [9], [10], and [11], extend their focus to DNS logs, which are critical for detecting DGA-based botnets. However, these datasets still do not incorporate power consumption data, which has been shown to expose device-level anomalies associated with malware infections.

Datasets from [12], [13] and [14] are among the few that explore power consumption as a feature for detecting malicious activity. Notably, datasets [13] and [14] extend this approach by combining power consumption with network traffic, thereby demonstrating the potential of multimodal approaches to improve detection accuracy. This analysis highlights the importance of developing multimodal datasets that can combine information from multiple sources like network traffic, DNS logs, and power consumption data. This dataset provides a more complete view of malware behaviour across all phases of infection, communication, and attack execution.

**Table 1. Summary of Datasets for Botnet and Malware Detection**

| Dataset Works | Modalities Used | Malware Type | Phase Label | Environment |
|---|---|---|---|---|
| MedBIoT [5] | 1 - Network Traffic | Botnet | Infection, Propagation | IoT |
| Bot-IoT [6] | 1 - Network Traffic | Botnet | Attack | IoT |
| CICIoT2023 [7] | 1 - Network Traffic | Botnet, Malware | Attack | IoT |
| IoT-23 [8] | 1 - Network Traffic | Botnet, Malware | C&C, Attack, Scan | IoT |
| 10 Days [9] | 2 - Network Traffic, DNS Logs | Botnet | General Malicious | General |
| UMUDGA [10] | 1 - DNS Logs | Malware (DGA) | Malicious, Benign | General |
| UTL_DGA22 [11] | 1 - DNS Logs | Malware (DGA) | Malicious, Benign | General |
| Power-Based IoT Botnet Detection [12] | 1 - Power Consumption | Botnet | Infection | IoT |
| Malware Detection Using Power and Network Data [13] | 2 - Power Consumption, Network Traffic | Malware | Malicious, Benign | General |
| Energy-Based IoT Malware Detection [14] | 2 - Power Consumption, Network Traffic | Botnet | Malicious, Benign | IoT |

## 3. Testbed Development and Architecture

The hardware involved in the development of the testbed shown in consists of three Raspberry Pi 4 Model B [15] devices, each serving a different role. Figure 1 shows the devices used in testbed development with their respective functions. Additionally, three laptops running Windows 11 function as servers and controllers for the botnet and IoT services. Laptop 1 hosts virtual machines for the DNS server and the C&C server, both running Ubuntu with separate Wireless Network Interface Cards (WNICs) to ensure proper network segmentation.


**Figure 1. Testbed Equipment**

Table 2 shows a detailed inventory of the hardware components and software stack used in the testbed. Bind9 [16] serves as the DNS server to handle queries from IoT devices and hosts, while Oracle VirtualBox functions as a hypervisor for managing Ubuntu-based virtual machines. Network traffic is captured and analyzed using Wireshark [17], and CPU power consumption data is monitored using PowerJoular [18]. Because the testbed deployment is isolated from public networks, Chrony [19] is integrated into the testbed to synchronize timestamps across all devices to ensure precise correlation between different data sources. These tools act as an alternative to the Network Time Protocol (NTP) daemon that cannot be accessed from an isolated testbed.

**Table 2. Testbed devices and function**

| No | Device | Specification and Sensors | Software and Sensors | Function |
|---|---|---|---|---|
| 1 | IoT 1 (192.168.2.101) | Raspberry Pi 4 Model B 8Gb RAM, | Ubuntu, Temp & Humidity Sensor, PowerJoular | Collects environmental data and generates traffic |
| 2 | IoT 2 (192.168.2.102) | Raspberry Pi 4 Model B 8GB RAM, | Ubuntu, Webcam Sensor, PowerJoular | Captures video stream for testbed activity |
| 3 | IoT 3 (192.168.2.103) | Raspberry Pi 4 Model B 8GB RAM, | Ubuntu, Display Screen, PowerJoular | Streams video service for streaming simulation |
| 4 | Laptop 1 – DNS (192.168.2.251, C&C 192.168.2.254) | Lenove Thinkpad, Intel i7, 16GB RAM | Windows 11, Oracle VirtualBox, Ubuntu VM, Bind9, Chrony | Hosts DNS server and C&C server |
| 5 | Laptop 2 - Target Host (192.168.2.155) | Acer Aspire Vero, Intel i7, 8GB RAM | Windows 11, Wireshark | Target machine of DDoS attacks |
| 6 | Laptop 3 - Service Host (192.168.2.200) | HP Pavilion, Intel i5, 16GB RAM | Windows 11, IoT Service Controller | Interact with IoT devices for running services |
| 7 | Router / AP (192.168.2.1) | TP-Link TL-WR840N | Factory Firmware | Provides wireless connectivity |

The botnet in this testbed is implemented using Python-based malware scripts to replicate the behavior of a real Mirai-like botnet taken from [20] which can launch DoS attacks from IoT devices. The initial code was originally designed to enable infected IoT devices to establish communication with the C&C server using an Internet Protocol (IP) address only. To align it with the characteristics of a DGA-based botnet, the researchers refined the botnet's source code to allow it to communicate with the C&C server through dynamically generated domain addresses. Additionally, the researchers modified the malware to instruct infected IoT devices to send a series of random false DNS requests to effectively replicate the behavior of a DGA botnet, which communicates with the C&C server through dynamically generated domains, mimicking real-world DGA techniques. The attack phase primarily involves DDoS activities targeting the designated victim machine

## 4. Experimental Setup

The dataset generated in this study was gathered over multiple experimental sessions, each lasting between 60 and 120 minutes, with predefined attack intervals ranging from 5 to 10 minutes. The dataset consists of three primary components: network traffic data, DNS query logs, and CPU power consumption metrics. The DNS query logs contain records of domain resolution attempts made by both legitimate and botnet-infected devices. Network traffic data was captured at both packet and flow levels, providing insights into communication patterns between infected devices and the C&C server. CPU power consumption metrics were recorded to monitor the energy usage of infected and non-infected devices.

The testbed was run under six different scenarios, as shown in Table 3, each designed to analyze botnet behavior under varying conditions, including the number of bots, attack duration, active services, attack targets, and attack intervals.

**Table 3. DGA Botnet Attack Test Plan**

| Test | Bots | Duration (minutes) | Services | Target | Attack Interval (minutes) |
|---|---|---|---|---|---|
| T1 | 0 | 60 | ALL | None | None |
| T2 | 3 | 60 | NONE | 192.168.2.155 | 5-10 |
| T3 | 3 | 120 | ALL | 192.168.2.155 | 5-10 |
| T4 | 2 | 60 | Camera, Streaming | 192.168.2.103 | 5-10 |
| T5 | 2 | 60 | NONE | 192.168.2.103 | 5-10 |
| T6 | 3 | 60 | ALL | 192.168.2.155 | 10-30 |

Testing Plan 1 (T1) scenario served as a baseline to set the benchmarks to collect benign information. All IoT devices operated with their respective services enabled without performing any attacks. This allowed for the observation of normal traffic patterns without botnet activity. This test ran for one hour with all services active. Testing Plan 2 (T2) focused solely on attack behavior. During this test, all IoT services were disabled when a DDoS attack was launched against the target host on 192.168.2.155. This setup helped isolate traffic attacks from normal service activity, setting a benchmark on analysis of malicious network behavior.

Testing Plan 3 (T3) delivered the main dataset for this testbed. This test represented a full-scale attack scenario where three IoT devices launched a sustained DDoS attack on the target host (192.168.2.155) for two hours while all services remained active. This test was designed to replicate a real-world scenario, allowing an in-depth evaluation of attack impact, resource utilization, and behavior detection from the dataset collected under continuous botnet activity. Testing Plan 4 (T4) targeted the conventional host machine running Windows 11 OS, the attack targeted an IoT device. With all services running, two infected bots were simulated to execute a DDoS attack on 192.168.2.103, to simulate the impact of botnet activity on IoT-based victims.

Testing Plan 5 (T5) followed a similar approach but without any active services running on the infected IoT devices. The dataset collected from this testing should allow for a comparative analysis of attacks between service-heavy and idle IoT systems. The final test, Testing Plan 6 (T6), aimed to simulate real-world DGA-based botnet behavior by introducing latency in botnet execution, specifically delaying the infected IoT devices from initiating communication with the C&C server (192.168.2.251). All IoT devices ran their services while periodically attempting to resolve DGA-generated domains to establish a connection with the botmaster. Once communication was established, the devices executed DDoS attacks on 192.168.2.155. This test replicated real-world scenarios where botnets introduce delays before engaging with their C&C infrastructure to evade detection. These varied test scenarios provided valuable insights into botnet attack patterns, service disruption, and detection strategies, contributing to a better understanding of how DGA-based botnets interact with IoT environments.

Overall, these testing plans provided a structured evaluation of DGA-based botnet behavior, assessing the impact of attacks under different conditions. T3, the primary scenario, demonstrated the effects of a two-hour DDoS attack from three IoT devices with all services active, highlighting resource exhaustion and network disruption. T4 and T5 examined botnet attacks on IoT devices, showing how service availability influences attack impact. T6 introduced execution delays in botnet C&C communication, mimicking real-world evasion techniques. The datasets collected from these experiments offer valuable insights into botnet attack detection, network disruption patterns, and resource utilization, supporting the development of more effective botnet mitigation strategies in IoT environments.

## 5. Dataset Composition

The dataset collected from the testbed needs to provide a comprehensive representation of DGA-based botnet activities, capturing multiple data modalities to facilitate detection and analysis. Since botnet behavior varies significantly across its lifecycle, encompassing initial infection, command and control communication, attack execution, and evasion phases, specific labeling ensures a clear distinction between normal and malicious activities [2]. Figure 2 shows a schematic diagram of the testbed with DGA botnet attack. The first step involves the infected IoT device (192.168.2.101) generates multiple domain names using a Domain Generation Algorithm (DGA) to locate an active C&C server (192.168.2.251). It continuously queries domains until one is successfully resolved. In the second step, the DNS server (192.168.2.254) processes the IoT device's queries and responds once a botmaster-registered domain is found. The third step involves the IoT device

connects to the C&C server (192.168.2.251) and remains dormant awaiting instructions. In the fourth step, the C&C server issues DDoS attack instructions, specifying the target IP (192.168.2.155) and attack parameters. Lastly, in the fifth step, IoT device remains idle until activation to reduce detection risk. The infected device floods the target host (192.168.2.155) with traffic using SYN packets. If the C&C server is taken down, the IoT device repeats the first step again, generating new domains until it reconnects to a new active C&C.

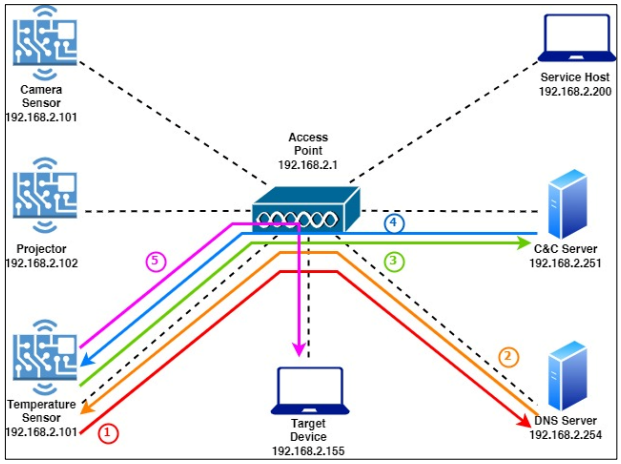


**Figure 2. Testbed Schematic Diagram**

Labeling the dataset is crucial for identifying different stages of botnet activity and enhancing early detection capabilities. The researchers have categorized the collected dataset into three labels Normal, C&C, and Attack. The Normal label represents benign traffic, such as normal DNS queries, HTTP requests, and non-malicious network communication. The C&C label represents data from the communication phase where the infected IoT device attempts to connect to the C&C server by sending DNS queries to resolve a valid domain before establishing a connection. The Attack label is assigned to the dataset when the infected device begins transmitting malicious traffic to the target host, indicating the execution of a DDoS attack.

### 5.1 Network Traffic Dataset Labeling

The network logging process is initiated using dumpcap, a packet capture tool within Wireshark. With the captured network traffic data in raw format, direct analysis will be difficult due to its unstructured nature. Raw packet captures contain low-level packet details but lack higher-level insights, requiring extensive manual inspection or custom scripts to extract relevant features. Identifying botnet-related activities within this raw data is particularly difficult, as it does not inherently provide structured information such as connection summaries, DNS resolutions, or communication patterns.

To address these limitations, Zeek processes the data, converting raw packets into structured logs [21]. Zeek's automated analysis can facilitate efficient extraction of key metadata, such as session details, domain lookups, and traffic details, significantly improving the detection and classification of botnet behavior. The data

preprocessing phase begins with feature extraction using Zeek, which transforms raw pcap files into structured, human-readable logs. Zeek processes the network traffic and generates multiple log files. Two important log files used for the labeling process, extracted from Zeek data, are the conn.log and dns.log. The conn.log provides metadata on network connections, including source and destination IPs, port numbers, connection duration, and data transfer volume, offering a high-level summary of all network activity. The dns.log records detailed information about DNS queries and responses, making it instrumental in identifying botnet-generated domains and distinguishing benign from malicious traffic. Then, the Flaber tool automatically labels Zeek's conn.log files based on predefined criteria stored in a CSV file [22]. This tool is specifically designed to label Zeek conn.log files efficiently. It automates the process of assigning labels to network connections based on predefined conditions, making it useful for labeling the Zeek log files based on IPs, ports, protocols and other metadata fields.

As shown in Figure 3, the labeling starts by processing conn.log using flaber.py, which extracts relevant network flow details, converts them into a structured JSON format (out.json), and applies initial labels based on predefined rules from labels.csv. The labelled JSON data is then transformed into a CSV file, followed by timestamp standardization to ensure uniformity, resulting in connlog.csv. The program will read the labelling rules from label.csv and prepare them for use in the log labeling process. It compares the current log entry against the rules' conditions from label.csv. If a match is found, then a corresponding label "Attack" or "C&C" is assigned to the log; if no rule matches, then a default label of "Normal" is applied.

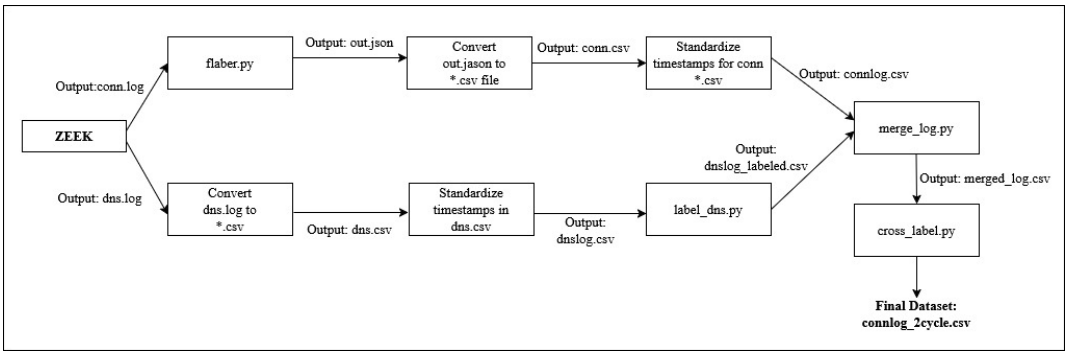


**Figure 3. Network Traffic Labelling Process Flows**

Figure 4 shows the list of rules created when the network traffic from infected IoT device 192.168.1.103 is used. The Field column specifies the network attribute being analyzed, such as source IP (`id.orig_h`), destination IP (`id.resp_h`), destination port (`id.resp_p`), or service type (`service`). The Bro field column maps these attributes to Zeek's network analysis framework. The Data column contains the IP addresses or port numbers to be compared against network logs using conditions specified in the Comparator column which in this case stands for 'equal to'. The Label column classifies traffic as "Attack," "C&C," or "Normal". Finally, the Connector column specifies the conditions for the rule set. The first rules (rows one and two) define attack detection. Here, the infected IoT device at IP 192.168.2.103 communicates with the victim host at 192.168.2.155, the traffic is labeled as "Attack" to capture malicious activity targeting the victim. Rules in rows

three and four identify command-and-control (C&C) communication. If the infected device sends traffic to the botnet's C&C server at 192.168.2.254 using port 8080, then the connection is labeled as "C&C."

| Id | Field | bro field number | Data | Comparator | Label | type | connector |
|---|---|---|---|---|---|---|---|
| 1 | id.orig_h | 3 | 192.168.2.103 | eq | Attack | Malicious | and 2 |
| 2 | id.resp_h | 5 | 192.168.2.155 | eq | Attack | Malicious | and 1 |
| 3 | id.resp_h | 5 | 192.168.2.254 | eq | C&C | Master | and 4 |
| 4 | id.resp_p | 6 | 8080 | eq | C&C | Master | and 3 |
| 5 | id.orig_h | 3 | 192.168.2.103 | eq | Normal | Benign | and 6 |
| 6 | id.resp_h | 5 | 192.168.2.200 | eq | Normal | Benign | and 5 |
| 7 | id.orig_h | 3 | 192.168.2.200 | eq | Normal | Benign | and 8 |
| 8 | id.resp_h | 5 | 192.168.2.103 | eq | Normal | Benign | and 7 |
| 9 | service | 8 | dns | eq | C&C | Master | - |

**Figure 4. Sample of Label Data for label.csv File**

Rules for benign communications are in rows five and six. They specify that when the infected IoT device exchanges traffic with the service host at 192.168.2.200, the connection is labeled as "Normal," to recognize legitimate interactions and prevent false positives. Similarly, rows seven and eight cover responses from the service host back to the infected device, which are also labelled as "Normal," to ensure everyday device operations are not mistakenly flagged as malicious.

Finally, rule in row nine establishes DNS-based C&C detection: when the infected device sends DNS queries that resolve to domains listed as malicious based on a predefined list, the traffic is labeled as "C&C." This rule is critical for catching botnets using domain generation algorithms to hide their infrastructure, since such queries often precede C&C communication. Together, these rules ensure accurate labeling of attack, C&C, and normal traffic, forming a robust foundation for reliable botnet detection.

### 5.2 DNS Queries Dataset Labeling

The labeling process in this work was inspired by the [23] study, which highlights the importance of structured DNS datasets for detecting malicious activities. Building upon this approach, our study focuses on constructing a labeled DNS dataset within a controlled testbed environment, ensuring a clear distinction between malicious and benign DNS queries. By systematically processing and labeling DNS logs, this dataset serves as a foundation for evaluating detection techniques based on DNS traffic patterns.

Figure 5 shows that the process of labeling and feature extraction for DNS query logs is essential for constructing a structured dataset that accurately distinguishes between malicious and benign DNS activities. The first step in this process involves trimming the DNS query logs to the required time range, ensuring that only relevant data within the intended collection period is retained. The trimmed log, saved as "correctedtimequeries.txt" file, is then opened in Excel, where an index column is added to preserve the original query sequence. This indexing step is necessary because subsequent feature extraction processes may record queries, and maintaining the original sequence is crucial for temporal analysis.

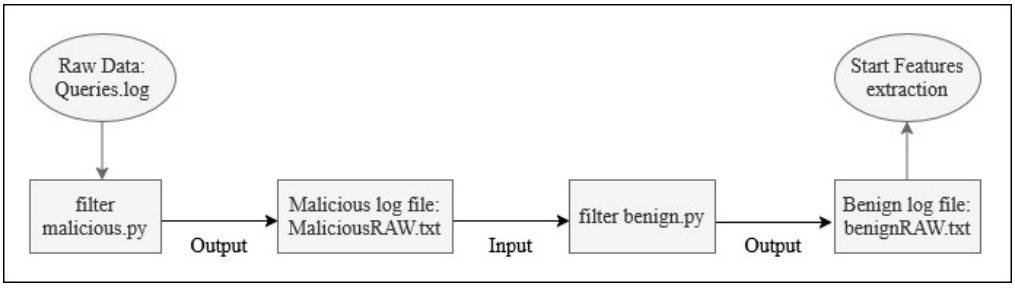


**Figure 5. DNS Dataset Labelling Process**

To classify queries, two filtering scripts filter_malicious.py and filter_benign.py were created. The filter_malicious.py script scans the dataset and extracts all DNS queries containing the domain "ripon", which was predefined as the C&C domain used by the botnet. The extracted queries are stored in "maliciousRAW.txt", forming a dedicated dataset of malicious queries. Next, the filter_benign.py script processes the original dataset, removing any queries previously identified as malicious. This results in "benignRAW.txt" file, a dataset containing only benign queries. This structured separation of malicious and benign queries is critical for ensuring that the dataset remains uncontaminated and suitable for feature extraction and classification. Once the queries are categorized, their format is standardized using extract_time_domain_recordtype.py, which extracts essential information such as `timestamp`, `queried domain`, and `DNS record type`. The reformatted data is saved as "benignFormatted.txt" and "maliciousFormatted.txt", preparing it for feature extraction. Additionally, a deduplicated list of benign domains, excluding any containing "ripon", is created and saved as a text file. This file is essential for feature extraction scripts that require only benign domains for analysis.

### 5.3 CPU Consumption Dataset Labelling

The labelling process of CPU power consumption consists of two stages: initial labelling using network log data and refinement using manually recorded event timestamps. In the first stage, power consumption data, which is stored in the "powlog.csv" file collected using PowerJoular, will be compared and cross-checked with the already labelled network log file "connlog_2cycle.csv". To automate the labelling process, two custom scripts cross_label_power.py and 2nd_label_power.py were created. These scripts were designed to systematically merge, synchronize, and refine the labels assigned to power consumption data.

From Figure 6, the first script, cross_label_power.py, was responsible for the initial integration of network activity logs with power consumption data. It processed power log file "powlog.csv" and labelled network log file "connlog_2cycle.csv", aligning their timestamps to a common format. Since power consumption logs were recorded at high frequencies and network activity timestamps varied in precision, rounding timestamps ensured that events occurring within the same second were consistently mapped. The script assigned labels to power consumption data based on the most frequent network event occurring within each second. If no network activity was detected for a given timestamp, the entry was labelled as "Normal". This process generated a preliminary labelled dataset named "powlog_labeled.csv".

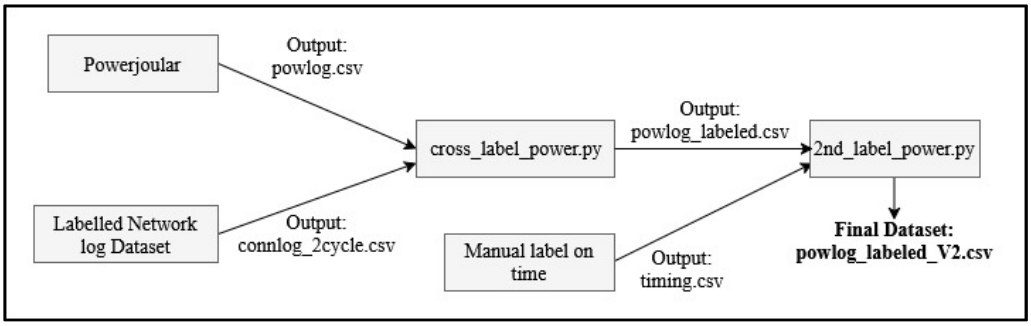


**Figure 6. CPU Power Labelling Process**

The second script, 2nd_label_power.py, further refined the labelling by incorporating manually annotated event timings from “timing.csv”. This script merged the preliminary dataset with the event timeline recorded when the botnet contacted the command-and-control server and initiated the DDoS attack on the target victim, ensuring a more precise classification of botnet activities. If the label column from “powlog_labeled.csv” has a missing value in any row, the script replaces it with the corresponding Event value from the “timing.csv” file. If both label and Event are missing, the script assigns a default label value of "Normal" because by default there was no attack or command-and-control communication happening at that time, as everything was logged by the network monitoring tool. Finally, the fully updated and labelled power log is saved as powlog_labeled_v2.csv, completing the process.

## 6. Result And Analysis

### 6.1 Network Log Dataset Overview and Visualization

The final network dataset was generated after data cleaning and labeling, containing 23 features that capture essential attributes of network connections. These features include timestamps (`ts`, `tsStandard`), connection identifiers (`uid`), source and destination IP addresses (`id.orig_h`, `id.resp_h`), ports (`id.orig_p`, `id.resp_p`), protocol type (`proto`), network service type (`service`), connection duration (`duration`), and various packet-related statistics such as byte counts (`orig_bytes`, `resp_bytes`) and packet counts (`orig_pkts`, `resp_pkts`). The dataset also includes three pieces of Zeek-specific metadata, connection state (`conn_state`), missed bytes (`missed_bytes`), and logging indicators (`local_orig`, `local_resp`). The label field categorizes each connection as malicious or benign, serving as the ground truth for attack detection.

For comparison, the well-known IoT-23 dataset [8], which is widely used for network security research, contains 21 features extracted using Zeek. Our dataset extends this feature set by adding `tunnel_parents` and `tsStandard`, enhancing metadata tracking and timestamp standardization. The dataset was collected across six distinct test plans as shown in Table 3, with three datasets generated per experiment, resulting in a total of 18 datasets. The dataset sizes vary based on network activity, with Test Plan 3 (device 192.168.2.101) recording over 4.3 million connection logs during an active attack scenario. Figure 7 shows a snippet of the cleaned network log dataset, representing structured network traffic data collected

after the raw capture was processed and refined. The columns include essential network metadata such as timestamps, source and destination IP addresses, protocol types, packet sizes, and flow duration. These features are crucial for analyzing traffic behavior and distinguishing between normal IoT activity and malicious botnet communication.

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ts | uid | id.orig_h | id.orig_p | id.resp_h | id.resp_p | proto | service | duration | orig_bytes | resp_byte | conn_stat | local_orig | local_resp | missed_b | history | orig_pkts | orig_ip_by | resp_pkts | resp_ip_b | tunnel_pa | label | tsStandar |
| 1714030578 | COxuJk47( | 192.168.2. | 5353 | 224.0.0.25 | 5353 | udp | dns | 0 | 0 | 0 | S0 | TRUE | TRUE | 0 | D | 1 | 73 | 0 | 0 | 0 | Normal | 15:36:18 |
| 1714030556 | CPRTlq42f | 192.168.2. | 38877 | 192.168.2. | 123 | udp | ntp | 0.00929 | 48 | 48 | SF | TRUE | TRUE | 0 | Dd | 1 | 76 | 1 | 76 | 0 | Normal | 15:35:56 |
| 1714030613 | CbJHSw4U | 192.168.2. | 5353 | 224.0.0.25 | 5353 | udp | dns | 0 | 0 | 0 | S0 | TRUE | TRUE | 0 | D | 1 | 73 | 0 | 0 | 0 | Normal | 15:36:53 |
| 1714030625 | C74AXS2V | 192.168.2. | 5353 | 224.0.0.25 | 5353 | udp | dns | 0 | 0 | 0 | S0 | TRUE | TRUE | 0 | D | 1 | 73 | 0 | 0 | 0 | Normal | 15:37:05 |
| 1714030642 | CRlV7nPs | 192.168.2. | 5353 | 224.0.0.25 | 5353 | udp | dns | 0.925202 | 86 | 0 | S0 | TRUE | TRUE | 0 | D | 2 | 142 | 0 | 0 | 0 | Normal | 15:37:22 |
| 1714030620 | CawPxy2R | 192.168.2. | 44969 | 192.168.2. | 123 | udp | ntp | 0.003254 | 48 | 48 | SF | TRUE | TRUE | 0 | Dd | 1 | 76 | 1 | 76 | 0 | Normal | 15:37:00 |
| 1714030721 | C8pNsB23 | 192.168.2. | 38331 | 192.168.2. | 53 | udp | dns | 0.058441 | 156 | 156 | SF | TRUE | TRUE | 0 | Dd | 4 | 268 | 4 | 268 | 0 | Normal | 15:38:41 |
| 1714030721 | CNjgYq1r( | 192.168.2. | 43696 | 192.168.2. | 53 | udp | dns | 0.011942 | 98 | 206 | SF | TRUE | TRUE | 0 | Dd | 2 | 154 | 2 | 262 | 0 | Normal | 15:38:41 |
| 1714030723 | CHms9o3I | 192.168.2. | 33870 | 192.168.2. | 53 | udp | dns | 0.021496 | 160 | 160 | SF | TRUE | TRUE | 0 | Dd | 4 | 272 | 4 | 272 | 0 | C&C | 15:38:43 |

**Figure 7. Network Log Dataset After Cleaning**

To analyze the dataset, data visualization and statistical analysis were performed using Jupyter Notebook. Since network connection logs contain both categorical and numerical features, Label Encoding was applied to transform categorical fields into numerical values. This step was essential for computing correlation coefficients, which measure the strength of relationships between features. A heatmap of feature correlations was generated to identify features that strongly influence classification.

The five highest correlated features with the label were identified. From Figure 8, the analysis revealed that destination IP address (`id.resp_h`) and network service type (`service`) were the most strongly correlated features, reflecting the importance of destination addresses and service types in attack classification. Additionally, protocol type (`proto`) exhibited strong correlation, which aligns with the dataset's attack patterns, as the botnet primarily utilized TCP SYN Flood attacks.

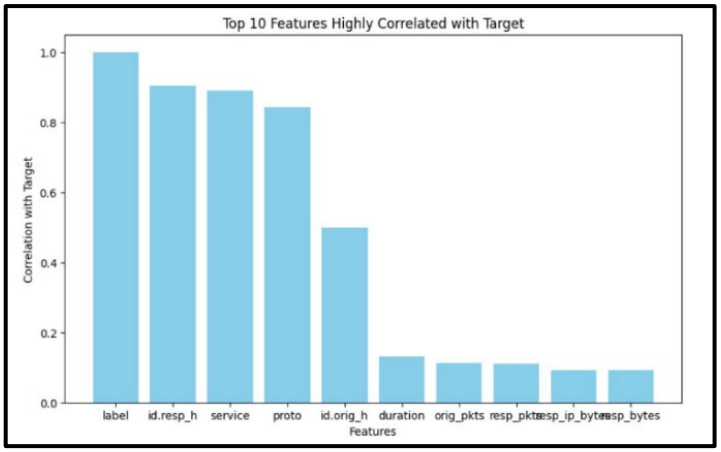


**Figure 8. Top 10 Highest Correlated Features with Label**

Conversely, features such as `tsStandard`, `uid`, and `ts` showed minimal correlation with the label, as timestamps and unique connection identifiers do not inherently indicate malicious behavior. By focusing on the most relevant features,

the dataset can be optimized for machine learning-based intrusion detection, reducing computational complexity while retaining key indicators of botnet activity.

### 6.3 DNS Queries Dataset Overview and Visualization

After data cleaning and extraction, the final dataset comprises 37 features. In comparison to the dataset from reference work by [23], which contains 34 features, this study's dataset includes similar attributes, with three additional features: `Timestamp`, `RequesterIP`, and `SecondLevelDomain`. The `RequesterIP` feature enables cross-referencing with other datasets. The dataset is stored as a CSV file with its descriptions and default values. Every parameter is labeled as "Normal" represents benign domains, and “C&C” denotes malicious domains. The dataset is inherently imbalanced, reflecting real-world conditions. Some rows contain NULL values in at least one feature. These NULL values should be addressed during data preparation to ensure optimal data handling.

Like network traffic dataset visualization, evaluating and visualize the relevance of the DNS queries dataset features, a correlation coefficient heatmap was generated from Test Plan 3 data. The results reveal important features that can be strongly associated with the dataset labels, as shown in Figure 9. The `DomainReputation` feature exhibited the strongest correlation with the label, as domains with high reputation scores were typically classified as Normal, while those with low scores were labeled Malicious. Similarly, `IPReputation` showed a strong correlation, as known malicious IPs were frequently associated with botnet activities. Additionally, `CreationDate` played a significant role in classification, as newly registered domains were more likely to be flagged as Malicious due to their frequent use in botnet-generated domain names. `RequesterIP` also contributed significantly, particularly when specific IP addresses were linked to malicious activities within the testbed environment. Lastly, `LastUpdate` influenced classification, as recently updated domains were often considered more suspicious.

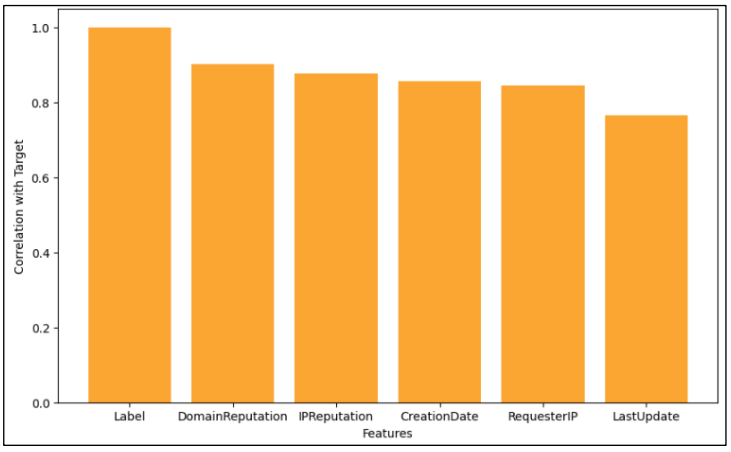


**Figure 9. DNS Queries Dataset Top 5 Most Highly Correlated Features**

By focusing on highly correlated features while removing less relevant attributes, the dataset was optimized for machine learning-based intrusion detection, reducing computational complexity while preserving key indicators of botnet activity. This structured dataset ensures high-quality network traffic analysis, making it a valuable

resource for evaluating botnet detection models, network anomaly detection, and real-time threat intelligence systems.

### 6.5 CPU Power Dataset Overview

The CPU power dataset overview sample originates from an infected IoT device 192.168.2.103 in Test Plan 3. It consists of 7508 entries with key attributes, including time, CPU utilization, CPU power consumption, and botnet phase labels which are labelled as Normal, C&C, and Attack as shown in Figure 10. The dataset captures power consumption variations across different botnet states, making it a potential candidate for machine learning-based detection. The CPU utilization values range from 0.07% to 0.52%, while CPU power consumption fluctuates between 2.98W and 4.65W, depending on the botnet activity phase.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | Time | CPU Utilization | CPU Power | Label |
| 2 | 16:15:31 | 0.471698113 | 4.356872114 | Attack |
| 3 | 17:05:46 | 0.520435967 | 4.659697323 | Attack |
| 4 | 16:39:35 | 0.115577889 | 3.03260287 | C&C |
| 5 | 16:38:53 | 0.225063939 | 3.231245376 | C&C |
| 6 | 16:44:22 | 0.080200501 | 2.992169594 | Normal |
| 7 | 18:08:55 | 0.07518797 | 2.98801937 | Normal |

**Figure 10. Snippet of Dataset for Tesplan 3 Device (192.168.2.103)**

Figure 11 shows label distribution analysis to view the dataset composition. The results show that 53.44% (4012 entries) of the dataset belongs to the Normal phase, representing periods when the device operates without botnet interference. The C&C phase accounts for 26.88% (2018 entries), capturing moments when the device attempts to establish communication with the botnet's C&C server. The Attack phase, comprising 19.68% (1478 entries), corresponds to instances when the botnet launches a DDoS attack, causing significant spikes in power consumption.

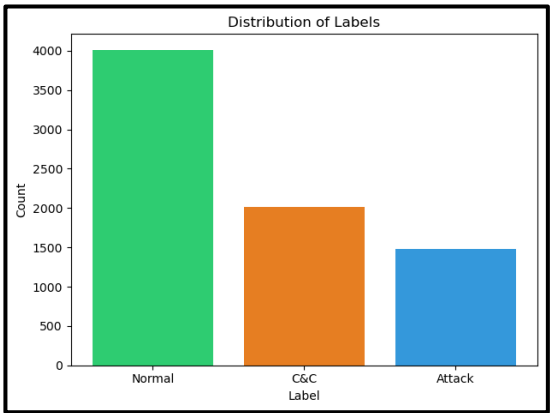


**Figure 11. CPU Power Daset Label Distribution**

A time series analysis was conducted to observe CPU utilization trends over time across different botnet phases. The visualization revealed distinct patterns, with the

Attack phase showing sharp spikes in CPU utilization, reaching up to 0.52%, indicating high computational demand during an active attack. The C&C phase displayed moderate fluctuations, reflecting periodic communication attempts with the botnet's control server. Interestingly, the Normal phase also exhibited occasional power increases when the IoT device streamed video from the service host, temporarily increasing power usage during normal operation. In one of separate work, the researcher used ANOVA statistical techniques and managed to show significant differences across three botnet activities using only the CPU power consumption dataset [24].

### 6.6 Summary

This work has managed to produce a multimodal, phase-labelled dataset that integrates CPU power, DNS queries and network traffic. Unlike single modal datasets from [5] and [7], which mainly focus on network behavior, this dataset also captures hardware level activity that can help identify hidden malicious processes. Although bi modal studies by [13] and [14] combine network and power data to improve detection, they do not include domain name system behavior, which is important for detecting modern algorithm based communication. The dataset provides a unique perspective across three lifecycle phases. In the Normal phase, it establishes a comprehensive baseline of legitimate service and power activity. During the C&C phase, the integration of DNS queries and processor fluctuations enables early detection of algorithmic threats, capturing a dynamic view missed by static datasets like [10]. In the Attack phase, the synchronization of network floods with hardware spikes offers dual layer ground truth superior to traditional monitoring. This integrated approach ensures more robust and proactive security for IoT environments.

## 7. Conclusion

In conclusion, the generated dataset from the testbed has been carefully labeled to reflect the Normal, C&C communication, and Attack phases, providing a phase-aware dataset for analyzing botnet behavior in a network that comprises IoT devices. The initial analysis reveals distinct behavioral patterns across the three data modalities, demonstrating that combining network, DNS, and power consumption data offers valuable insights into botnet activities. The top features were identified for network traffic and DNS queries, while CPU power consumption showed clear behavioral changes corresponding to different botnet phases. These findings provide a strong indication that integrating multiple data sources could enhance the effectiveness of botnet detection in resource-constrained IoT environments. Future work will focus on developing machine learning models that leverage this multimodal dataset for automated detection and phase classification of DGA-based botnets. Additionally, further analysis will explore the potential for early-stage detection, particularly during the C&C communication phase, to enhance pre-attack mitigation strategies. Expanding the testbed to include additional IoT device types, more sophisticated botnet variants, and encrypted communication scenarios will also be considered to ensure the proposed detection approaches remain robust against evolving threats.

## Acknowledgments

The research presented in this paper is a part of work of the broader project on the Responsible AI for a Secure and Trustworthy Energy Transition. This project has been partially funded by the Universiti Tenaga Nasional through Grant Project 202202001ETG

## Conflicts of Interest

The author declares that there is no conflict of interest regarding the publication of this paper.

## References

[1] M. Kührer, C. Rossow, and T. Holz, "Paint it black: Evaluating the effectiveness of malware blacklists," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8688 LNCS, pp. 1–21, 2014, doi: 10.1007/978-3-319-11379-1_1.

[2] B. C. Cebere, J. L. B. Flueren, S. Sebastián, D. Plohmann, and C. Rossow, "Down to earth! Guidelines for DGA-based Malware Detection," in *The 27th International Symposium on Research in Attacks, Intrusions and Defenses*, New York, NY, USA: ACM, Sep. 2024, pp. 147–165. doi: 10.1145/3678890.3678913.

[3] A. Shafee, "Botnets and their detection techniques," in *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, 2020, pp. 1–6. doi: 10.1109/ISNCC49221.2020.9297307.

[4] M. Wazzan, D. Algazzawi, O. Bamasaq, A. Albeshri, and L. Cheng, "Internet of things botnet detection approaches: Analysis and recommendations for future research," Jun. 02, 2021, *MDPI AG*. doi: 10.3390/app11125713.

[5] A. Guerra-Manzanares, J. Medina-Galindo, H. Bahsi, and S. Nomm, "MedBIoT: Generation of an IoT Botnet Dataset in a Medium-sized IoT Network," *Int. Conf. Inf. Syst. Secur. Priv.*, no. March, pp. 207–218, 2020, doi: 10.5220/0009187802070218.

[6] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Futur. Gener. Comput. Syst.*, vol. 100, pp. 779–796, 2019, doi: 10.1016/j.future.2019.05.041.

[7] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment," *Sensors*, vol. 23, no. 13, 2023, doi: 10.3390/s23135941.

[8] S. Garcia, A. Parmisano, and M. J. Erquiaga, "IoT-23: A labeled dataset with malicious andbenign IoT network traffic," Jan. 2020, *Zenodo*. doi: 10.5281/zenodo.4743746.

[9] M. Singh, M. Singh, and S. Kaur, "10 Days DNS Network Traffic from April-May, 2016," 2019, *Mendeley Data*. doi: 10.17632/zh3wnddzxy.2.

[10] M. Zago, M. Gil Pérez, and G. Martínez Pérez, "UMUDGA: A dataset for profiling DGA-based botnet," *Comput. Secur.*, vol. 92, 2020, doi: 10.1016/j.cose.2020.101719.

[11] T. A. Tuan, N. V. Anh, T. T. Luong, and H. V. Long, "UTL_DGA22 - a dataset for DGA botnet detection and classification," *Comput. Networks*, vol. 221, no. October 2022, p. 109508, 2023, doi: 10.1016/j.comnet.2022.109508.

[12] W. Jung, H. Zhao, M. Sun, and G. Zhou, "IoT botnet detection via power consumption modeling," *Smart Heal.*, vol. 15, no. December 2019, p. 100103, 2020, doi: 10.1016/j.smhl.2019.100103.

[13] J. Hernandez Jimenez, K. Goseva-Popstojanova, J. M. Hernández Jiménez, K. Goseva-Popstojanova, J. Hernandez Jimenez, and K. Goseva-Popstojanova, "Malware Detection Using Power Consumption and Network Traffic Data," *Proc. - 2019 2nd Int. Conf. Data Intell. Secur. ICDIS 2019*, pp. 53–59, 2019, doi: 10.1109/ICDIS.2019.00016.

[14] F. Jaafar, D. Ameyed, A. Barrak, and M. Cheriet, "Identification of Compromised IoT Devices: Combined Approach Based on Energy Consumption and Network Traffic Analysis," in *IEEE International Conference on Software Quality, Reliability and Security, QRS*, Institute of Electrical and Electronics Engineers, 2021, pp. 514–523. doi: 10.1109/QRS54544.2021.00062.

[15] Raspberry Pi Foundation, "Raspberry Pi 4 Model B." Accessed: Oct. 10, 2024. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

[16] Internet Systems Consortium, "BIND9." Accessed: Oct. 10, 2024. [Online]. Available: https://www.isc.org/bind/

[17] Wireshark Foundation, "Wireshark." Accessed: Oct. 10, 2024. [Online]. Available: https://www.wireshark.org

[18] Noureddine Adel, "PowerJoular and JoularJX: Multi-Platform Software Power Monitoring Tools," in *Proceedings of the 18th International Conference on Intelligent Environments*, IEEE, 2022.

[19] Chrony Project, "Chrony Documentation." Accessed: Oct. 10, 2024. [Online]. Available: https://chrony-project.org/documentation.html

[20] wodxgod, "PYbot." Accessed: Jul. 15, 2024. [Online]. Available: https://github.com/wodxgod/PYbot

[21] Corelight, "zeek." Accessed: Oct. 11, 2024. [Online]. Available: https://zeekdotorg.wpcomstaging.com/

[22] S. IPS, "Flaber," 2025. [Online]. Available: https://github.com/stratosphereips/flaber/blob/main/flaber.py

[23] C. Marques, S. Malta, and J. P. Magalhães, "DNS dataset for malicious domains detection," *Data Br.*, vol. 38, Oct. 2021, doi: 10.1016/j.dib.2021.107342.

[24] Z. A. Ibrahim, S. A. Ismail, F. A. Rahim, M. H. bin Abas, M. I. bin Khairul Anuar, and A. H. Bin Azwan, "Statistical Analysis of CPU Power Consumption for Detecting DGA Botnet Command and Control Communication," *2024 IEEE Int. Conf. Comput. ICOCO 2024*, pp. 60–65, 2024, doi: 10.1109/ICOCO62848.2024.10928271.