

Understanding IoT Domain Names: Analysis and Classification Using Machine Learning

| | | | |
|--|--|--|---|
| Ibrahim Ayoub <i>Afnic – Université Paris-Saclay</i> Yvelines, France ibrahim.ayoub@afnic.fr | Martine S. Lenders <i>Technische Universität Dresden</i> Dresden, Germany martine.lenders@tu-dresden.de | Benoît Ampeau <i>Afnic</i> Yvelines, France benoit.ampeau@afnic.fr | Sandoche Balakrichenan <i>Afnic</i> Yvelines, France sandoche.balakrichenan@afnic.fr |
| Kinda Khawam <i>Université de Versailles St-Quentin</i> Versailles, France kinda.khawam@uvsq.fr | Thomas C. Schmidt <i>HAW Hamburg</i> Hamburg, Germany t.schmidt@haw-hamburg.de | Matthias Wählisch <i>Technische Universität Dresden – Barkhausen Institut</i> Dresden, Germany m.waehlich@tu-dresden.de | |

Abstract—In this paper, we investigate the domain names of servers on the Internet that are accessed by IoT devices performing machine-to-machine communications. Using machine learning, we classify between them and domain names of servers contacted by other types of devices. By surveying past studies that used testbeds with real-world devices and using lists of top visited websites, we construct lists of domain names of both types of servers. We study the statistical properties of the domain name lists and train six machine learning models to perform the classification. The word embedding technique we use to get the real-value representation of the domain names is Word2vec. Among the models we train, Random Forest achieves the highest performance in classifying the domain names, yielding the highest accuracy, precision, recall, and F_1 score. Our work offers novel insights to IoT, potentially informing protocol design and aiding in network security and performance monitoring.

Index Terms—IoT, domain names, machine learning, security

I. INTRODUCTION

Domain name classification enables detecting both phishing and domain names generated by domain generation algorithms (DGAs) [1]–[3]. Phishing domain names are used by malicious servers that pose as legitimate ones and lure users into providing sensitive information and credentials. On the other hand, DGAs run on malware-infected devices and generate domain names to help the infected devices contact the Command & Control (C&C) servers. The domain name classification techniques could also be applied in the Internet of Things (IoT) environments. IoT devices often need to communicate with servers on the Internet to which they connect using their domain names [4].

In this paper, we study IoT from a different viewpoint by studying the domain names of the servers on the Internet that IoT devices interact with and classify between them and servers contacted by other types of devices. We are interested in IoT devices that perform strictly machine-to-machine (M2M) communications. The servers such devices contact might be IoT-specific backend servers to which they

relay information, receive commands and updates, or other generic servers not exclusive to IoT.

We compile two lists of domain names. Using packet captures of real devices, initially filtering the traffic of IoT M2M devices, we construct a list of domain names of servers that are contacted by these devices. This list we call *IoT M2M Names*. The remaining packet captures we use to construct a list of domain names of servers exclusively contacted by other types of devices, never by IoT M2M devices. This list we call *Other Names*.

For the rest of the paper, we will refer to IoT M2M Devices as *IoT M2M Devices* and IoT devices that are not M2M, generic devices, and human users as *Other Devices*.

The end goal is to study the domain names of servers contacted by *IoT M2M Devices* and classify between them and the domain names of servers that cater to *Other Devices*.

Previous works [2], [3], [5], [6] use lists of top visited websites as a negative class in domain name classification problems. We also use two top lists, namely Cisco Umbrella 1 Million [7] (hereafter referred to as *Cisco*) and Tranco Top 1M [8], [9] (hereafter referred to as *Tranco*) to evaluate the performance of the models with such lists and the validity of using them in IoT domain name classification.

First, we construct the two lists, *IoT M2M Names* and *Other Names*. We use the public datasets from 12 previous studies (See Appendix A). Filtering the traffic of *IoT M2M Devices*, we use it to construct the *IoT M2M Names* set. The remaining traffic, the traffic of *Other Devices*, is then used to construct the *Other Names* set. This is followed by a data pre-processing phase that includes several sanitization tests and a study of the statistical properties of the domain names. Finally, we train six machine learning models to classify between *IoT M2M Names* and *Others Names*, *Cisco*, *Tranco* and a list comprising the three sets, and evaluate the performance of the machine learning models.

Our work studies IoT from a new perspective by studying the domain names of the servers contacted by *IoT M2M Devices*.

We study the composition and the statistical properties of these domain names. Such insight helps shed light on domain names of IoT servers used by *IoT M2M Devices* as we present our findings about what the average domain name of such servers looks like and identify patterns and particularities if found. This could be helpful to model and generate M2M IoT domain names that align with the average domain name of that type—for instance, aiding protocol design such as name compression for constrained devices [10]. Furthermore, the machine learning models we trained successfully classified *IoT M2M Names* and *Others Names*, and the performance evaluation we performed indicated the ability of the models to be generalized to unseen data. This capability could aid in detecting outliers in M2M IoT networks. For example, non-M2M traffic in networks consisting solely of M2M IoT devices could be detected.

The remainder of the paper is organized as follows. Section II provides background information, while Section III outlines the methodology employed. Section IV presents the results obtained and Section V discusses the key takeaways. Finally, Section VI discusses related work, and, Section VII concludes the work.

II. BACKGROUND

A. *IoT M2M Names and Other Names*

IoT M2M Devices usually contact servers on the Internet to relay information about the physical world or from which they receive commands and firmware updates [4]. These devices rely on domain names as an indirection mechanism to connect to IP endpoints, simplifying maintenance. This allows, for example, a transparent change of server addresses, as only the mapping in the DNS would need to be changed. The *IoT M2M Devices* are typically pre-configured with the domain names of the servers they might need to contact, and they obtain the addresses of these servers by resolving the domain names via DNS. Beyond address resolution, future IoT devices might also use DNS to identify the service bindings of these servers, *e.g.*, whether they use the TCP-based HTTP/2, the QUIC-based HTTP/3, or other services such as CoAP, using SVCB resource records [11], [12].

The domain names of such servers exhibit distinct construction characteristics influenced by various factors. An IoT backend server, for example, might have a name that correlates with its high-level function. For example, a collection of IP cameras might have a backend server whose domain name is `cam.example.com`. Moreover, large IoT backend service providers tend to adopt a naming convention for their servers, which follows the pattern below [4]:

```
<subdomain>.<region>.<second-level-domain>
```

where `<subdomain>` could be the name of the IoT service or the protocol name, `<region>` refers to the location of the server, and `<second-level-domain>` could be the second-level domain of the service provider or a name related to the IoT service.

Last, being involved in M2M communications, some *IoT M2M Names* may contain machine-friendly character sequences that do not prioritize legibility or memorability and are challenging for humans to comprehend.

Meanwhile, *Other Names*, *i.e.*, domain names of servers catering *Other Devices* do not follow the same patterns. Since servers with such domain names serve a wide range of devices and human users, the legibility and memorability of their domain names are prioritized.

In this work, we visualize the differences between the two types of domain names by studying their statistical properties. We then move to classify them using machine learning.

B. *Word2vec for Word Embedding*

Before inputting them into machine learning models, the domain names have to be processed to obtain a real-valued vector representation of them. There are several options to achieve this. One way is through Natural Language Processing (NLP). NLP methods aim to obtain the real-valued vector representation of the textual data. This includes, for example, character level embedding [2], [5], [13], which gives each character a fixed-size real-valued vector representation. Another NLP method is the Term Frequency-Inverse Document Frequency (TF-IDF), which assigns an importance value to each text element based on its frequency of appearance [6]. A different way of processing textual data is the extraction of hand-crafted features, which studies the text, tries to extract properties, and uses these properties as a real-valued vector representation of the text [6], [14].

The method we use in this paper to obtain the real-valued vector representation of the domain names is Word2vec. In the context of NLP, this is also called word embedding. Word2vec [15], [16] is one of several word embedding techniques, and it uses a shallow neural network to convert each word to a vector of real numbers. Word2vec captures semantic relationships between words, and the resulting real-valued vectors depend on the context of each word within the text. Two possible architectures for Word2vec exist, Continuous Bag-of-Words (CBOW) and Skip-gram. CBOW estimates the vector representation of a target word based on the context (*i.e.*, surrounding words) of this word. The number of surrounding words considered within the context is specified by a *window size*. Two words that appear regularly together in a text would have vector representations geometrically close to each other in space. Consequently, words that do not appear together in the text are assigned vector representations that are distant geometrically. For Skip-gram, the model predicts the words before and after a target word based on the *window size*. Between CBOW and Skip-gram, we chose to go with CBOW as it is less expensive computationally and faster to train [16].

III. METHOD

The objective of our study is to gain a better understanding of *IoT M2M Names* and to evaluate the performance of common machine learning models in the classification between *IoT M2M Names* and *Other Names*. We construct a dataset

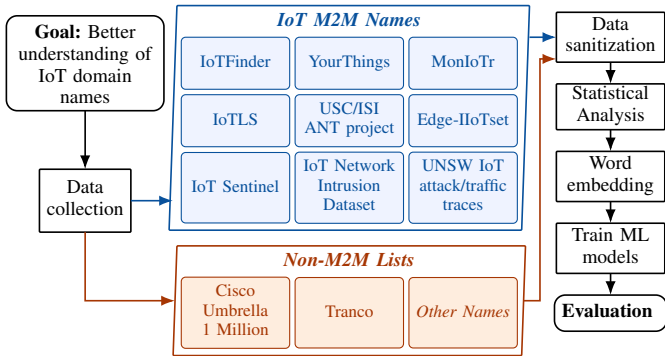


Fig. 1: Our method applied in this paper.

of domain names resolved by real *IoT M2M Devices* and a dataset of domain names resolved by real *Other Devices*. In addition, we use two top lists, namely *Cisco* and *Tranco*. We sanitize the datasets to ensure that only valid domain names are used. We then familiarize ourselves with the datasets through a statistical analysis.

To train the machine learning models, we need a real-valued vector representation of the domain names, which we generate using Word2vec. Last, we train six machine learning models to classify between *IoT M2M Names* and *Other Names*, *Cisco*, and *Tranco*. Figure 1 summarizes our steps.

A. Data Collection

In our analysis, we use two types of domain name lists. The first type includes a list that contains domain names of servers contacted by *IoT M2M Devices*, i.e., *IoT M2M Names*. The second type includes *Other Names*, a list of domain names of servers contacted by *Other Devices*, and two top lists, namely *Cisco* and *Tranco*. The following explains how we constructed the lists of domain names.

1) ***IoT M2M Names***: The *IoT M2M Names* list should contain domain names of servers on the Internet contacted by *IoT M2M Devices*, e.g., IoT backend servers that provide services to *IoT M2M Devices* such as a server that saves the footage from IP cameras or servers from which *IoT M2M Devices* receive commands and software updates. This list may also include domain names of servers that are not IoT-specific but are nevertheless contacted by *IoT M2M Devices*.

We use 12 public datasets that have been gathered in prior work. The datasets we used are IoTFinder [17], YourThings [18], MonIoTr [19], IoTLS [20], three datasets from the USC/ISI ANT project [21]–[23], Edge-IIoTset [24], IoT Sentinel [25], IoT Network Intrusion Dataset [26], UNSW IoT traffic traces [27], and UNSW IoT attack traces [28]. See Appendix A for a summary of the 12 datasets we used. These datasets contain packet captures collected in testbeds that included real devices, of which are *IoT M2M Devices*. Each is available as a set of PCAP files, including DNS messages sent from and received by the devices.

We filtered the PCAP files and extracted the unique DNS responses received by each device. Some datasets also contained captures from devices that were not strictly involved

TABLE I: Number of unique domain names after the resolver test.

| Dataset | <i>IoT M2M Names</i> | <i>Other Names</i> | <i>Cisco</i> | <i>Tranco</i> |
|--------------|----------------------|--------------------|--------------|---------------|
| Resolvable | 1 417 | 4 903 | 896 093 | 970 644 |
| Unresolvable | 1 134 | 1 477 | 103 907 | 29 356 |
| Total | 2 551 | 6 380 | 1 000 000 | 1 000 000 |

in M2M communication, such as desktop PCs, smartphones, gaming consoles, or smart TVs. We removed the captures of these devices but used them to construct *Other Names* (see below). Finally, we extracted the queried domain names from the resulting DNS responses. The resulting dataset, which we called *IoT M2M Names*, contained 2551 unique domain names.

2) ***Other Names and Top-visited Websites Lists***: The *Other Names* list should contain a list of domain names of servers used by devices not engaging in M2M communication or which are used by humans directly. We construct *Other Names* using traffic from the devices we excluded in the previous step. In addition, we use two top lists which are used as a negative class in domain name classification problems [2], [3], [5], [6]. There are several lists of this kind, each with criteria for calculating popularity.

The three lists we use are:

- ***Other Names***: While preparing the *IoT M2M Names* list, we filtered out the network traffic of devices that did not conform with our criteria for *IoT M2M Devices*, i.e., *Other Devices*. This allowed us to construct a dataset of real domain names of servers contacted by such devices. The dataset contained 6380 unique domain names.
- ***Cisco***: The Cisco Umbrella 1 Million [7] is a daily published list of one million websites. Any domain name could be included in the list. The ranking of each domain name is based on the number of unique client IPs that visited it [7]. The list for our evaluation was gathered on September 21, 2023.
- ***Tranco***: Tranco [8] is a research-oriented list of one million domain names. The ranking of each domain name is based on its average rank over the past 30 days from four other popular domain name lists [9]. The list for our evaluation was gathered on September 21, 2023, and thus covers the period from August 22 to September 20, 2023.

B. Data Sanitization

We conduct a data sanitization process, which includes several tests to ensure the validity of the domain names in each list. We perform the following tests:

Resolver Test. For each list of domain names, we try to resolve an *A record* for every domain name. We ensure that domain names that exist but do not have the requested *A record* are also counted as resolvable and account for empty non-terminal nodes. The results of the resolver test are presented in Table I.

Syntax Check. To ensure that all domain names in our lists respect the same syntax rules, we use the syntax checking used

TABLE II: Number of unique domain names after the syntax check.

| Dataset | <i>IoT M2M Names</i> | <i>Other Names</i> | <i>Cisco</i> | <i>Tranco</i> |
|-----------|----------------------|--------------------|--------------|---------------|
| Accepted | 1 415 | 4 895 | 888 297 | 970 644 |
| Discarded | 2 | 8 | 7 796 | 0 |
| Total | 1 417 | 4 903 | 896 093 | 970 644 |

TABLE III: Number of unique domain names after removing the IoT domain names from the non-IoT datasets.

| Dataset | <i>Other Names</i> | <i>Cisco</i> | <i>Tranco</i> |
|-------------------------|--------------------|--------------|---------------|
| Common with IoT Dataset | 979 | 940 | 14 |
| Remaining | 3 916 | 887 357 | 970 630 |
| Total | 4 895 | 888 297 | 970 644 |

by Zonemaster [29]. The process starts with a normalization procedure that replaces all the dots with the regular full stop of Unicode '\u002E' (or '.' as a character). The next step removes leading and trailing spaces. Next, a sequence of tests is conducted.

- Check if the domain name starts with a dot,
- Check if the domain name has consecutive dots,
- Remove trailing dots if found,
- Check if any label in the domain name is longer than 63 characters,
- Check if the total length of the domain name is more than 253 characters,
- Check if the domain name has only one label,
- Check if any label starts or ends with a hyphen ('-'), and, finally,
- Check if the domain name has double hyphen ('--') at positions 3 and 4 without it starting with 'xn'¹.

If one of the checks fails, the domain name is discarded. The results of the syntax check are presented in Table II.

Remove commons. Some domain names from the *IoT M2M Names* might appear in *Other Names* or in the top lists. Therefore, we remove the common domain names between *IoT M2M Names* and the other datasets from the other datasets. The resulting dataset sizes can be seen in Table III.

Final lists. After data sanitization, we obtain the final lists, which will be used in the following steps. The final lists can be seen in Table IV.

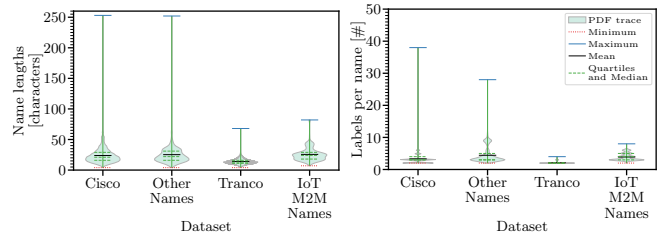
C. Statistical Study

We perform a statistical analysis of the domain name lengths and number of labels in each list, for which the results can be

¹*i.e.*, not an Internationalized Domain Name (IDN)

TABLE IV: Number of unique domain names in the final datasets.

| Dataset | <i>IoT M2M Names</i> | <i>Other Names</i> | <i>Cisco</i> | <i>Tranco</i> |
|------------------|----------------------|--------------------|--------------|---------------|
| Domain Names [#] | 1 415 | 3 916 | 887 357 | 970 630 |



(a) Name lengths.

(b) Number of labels.

Fig. 2: Violin plots for name properties found for each domain name in our datasets.

seen in the violin plots in Figure 2. Violin plots are similar to box plots, showing key statistical properties. However, they also estimate the probability density function (PDF) as a trace that forms the “body” of the “violins” around the properties.

The violin plots allow us to easily spot a similarity between domain names in *IoT M2M Names*, *Other Names*, and *Cisco* in terms of domain name length and number of labels per domain. This is due to the way each dataset is constructed. The three datasets contain domain names as observed in the DNS requests and are, therefore, more representative.

Tranco, on the other hand, is different from all the other lists we are using. The average *Tranco* domain name has fewer characters and labels than the average domain name from the other lists. *Tranco* mainly contains second-level domains in the form of *domain.tld*, while *IoT M2M Names*, *Other Names*, and *Cisco* do not have that limitation.

In addition, we plot the relative frequencies of the top labels in each list in Figure 3. The four lists have the same top two labels, which form at least 20% of the total labels in each list. Moreover, the top labels across all lists are predominantly Top-Level Domains (TLDs), such as “com”, “net”, and “org”. This implies that, when constructing an *IoT M2M* domain name, little emphasis is placed on selecting specialized *IoT*-only TLDs. Instead, the use of regular commercial TLDs seems a common practice. This indicates that the domain names of the different lists have, in general, similar TLDs and that the TLD of a domain name does not give a strong indication about the type of server that uses it as its domain name. We also see that the majority of labels—approximately 70%—take less than or equal to 4% each of the entire set.

The two studies above show that inspecting the length of a domain name in characters, its number of labels, or its TLD does not reveal information about whether this domain name is that of a server that caters to *IoT M2M Devices* or *Other Devices*. In the context of our work, these values do not seem to be distinctive features to reveal the list the domain name belongs to. The only exception could be the domain names from *Tranco*, which are second-level domains that tend to have limited number of characters.

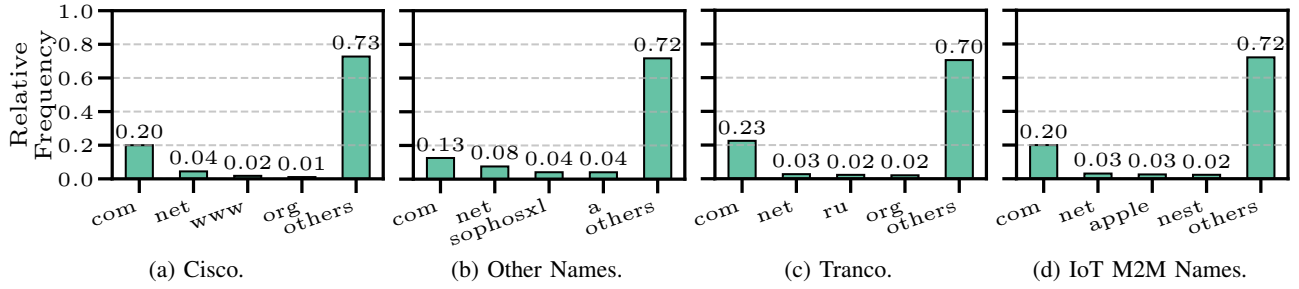


Fig. 3: Top 50 labels in the datasets. The top 4 of those are shown separately, while the remaining 46 are summarized to “others”.

D. Word2vec: Real-Valued Vector Representation of Domain Names

Word2vec expects prose text as input, *i.e.*, in the form of full documents with connected sentences and ideas where it can be used to capture the semantic relations. The challenge we face when using Word2vec with domain names is that these domain names do not form prose text. Instead, they are individual labels separated by periods. As such, each domain name is treated as a sentence, and each label is treated as a word. For example, `iot.backend.org` contains three labels and is transformed to “iot”, “backend”, and “org”. Another challenge is the limited size of domain names, which results in limited context and explains our choice of a *window size* of 3. After the Word2vec algorithm is done, we obtain a real-valued vector representation of each label. The dimensions of each vector are set in advance. Before applying Word2vec, and to have a consistent dataset in terms of size for training the machine learning models, we pad the domain names by adding ‘*’ as a dummy label on the left of each domain name. The longest domain name in our lists has 38 labels, so we pad all the domain names to have 40 labels.

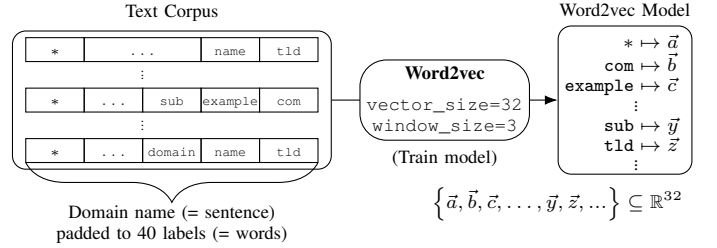
The parameters we used are as follows:

- **Padding:** To each domain name, we added ‘*’ on the left. Each ‘*’ was treated as a dummy label (*i.e.*, a word), and they were added until all the domain names were of length 40 labels (words).
- **Word2vec:** We used CBOW (Continuous Bag-of-Words Model) with a *window size* of 3.
- **Vectors:** Each word was represented by a vector $\in \mathbb{R}^{32}$.

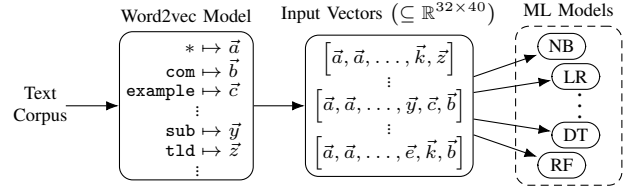
This vector representation can then be used to map each domain name to a 32×40 real-valued vector ($\in \mathbb{R}^{32 \times 40}$). The Word2vec process is depicted in Figure 4.

IV. RESULTS: DOMAIN NAME CLASSIFICATION

We train six machine learning models to classify *IoT M2M Names*, and *Other Names*, *Cisco* and *Tranco* domain names. We use the following models and refer to them using the acronyms in parentheses: Naïve Bayes (NB), Logistic Regression (LR), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree (DT), and Random Forest (RF).



(a) Step 1: Generate Word2vec model as label to real-valued vector mapping.



(b) Step 2: Use Word2vec model to generate input for machine learning models from text corpus.

Fig. 4: Word embedding: After prepending ‘*’ to each domain name until it has 40 labels, Word2vec is used to generate a real-valued vector representation of 32×40 real numbers of each domain name.

After the sanitization process, *IoT M2M Names* contains 1415 domain names. From *Other Names*, *Cisco*, and *Tranco*, we then pick 1415 domain names individually. We also create an additional list of 1415 domain names by uniformly sampling a *Mix* of *Other Names*, *Cisco*, and *Tranco*.

We select the 1415 domain names from *Other Names*, *Cisco*, and *Tranco* in two ways:

- We take the top 1415 domain names or
- randomly choose them, uniformly distributed, from the whole list.

After selecting 1415 domain names from each list, the domain names are labeled accordingly, and a combined list is constructed. The combined list is then processed via Word2vec to obtain the real-valued vector representation of each domain name. These real-valued vectors are used to train the machine learning models. The models are trained as binary classifiers between two classes, namely *IoT M2M Names* and *Other*

domain names where *Other* domain names belong to either one of *Other Names*, *Cisco*, or *Tranco*, or a *Mix* of them. To evaluate the performance of each of the models, we calculate the resulting accuracy, precision, recall, and the F_1 score in subsection IV-A. Moreover, we perform in subsection IV-B cross-validation to assess the robustness of the models and their ability to generalize to unseen data. Lastly, we perform in subsection IV-C an ablation test to analyze the impact of the different labels of the domain names on the performance of the models.

A. Performance Evaluation

In this section, we present our results after training several machine learning models to classify between *IoT M2M Names* and *Other* domain names. In each scenario, each list was processed with Word2vec to obtain the real-valued vector representation of each domain name of size 32×40 . We used an 80-20 train-test split.

We train the machine learning models NB, LR, KNN, SVM, DT, and RF. For each model, we calculate four parameters: Accuracy, precision, recall, and the F_1 score. We first calculate the confusion matrix to identify true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The values for our parameters are then produced using the following formulas:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (1)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

$$F_1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{\text{TP}}{\text{TP} + \frac{\text{FN} + \text{FP}}{2}} \quad (4)$$

Accuracy measures the ratio of positive predictions (TP and TN) to all the predictions made by the model, see Equation 1. Precision measures the ratio of true positive predictions (TP) to all the positive predictions (TP and FP) made by the model, see Equation 2. Recall measures the ratio of true positive predictions (TP) to all the actual positive instances in the dataset (TP and FN), see Equation 3. Finally, the F_1 score is the harmonic mean of precision and recall, see Equation 4.

The results for using the top 1415 domain names can be seen in Figure 5. For the random selection of domain names, see Figure 7.

1) **Results when using top domain names from *Other Names*, *Cisco*, and *Tranco*:** The results of training the models using the top 1415 domain names from *Other Names*, *Cisco*, and *Tranco* are presented in Figure 5. Each graph represents one of the four parameters obtained by the different models. The six models we trained exhibited the strongest performance when *Tranco* was used. The lowest performing model when *Tranco* was used was NB, but it still achieved values greater than 94% for the four parameters, while the rest achieved values between 98% and 100% for the four parameters. The lowest-performing model, regardless if *Other Names*,

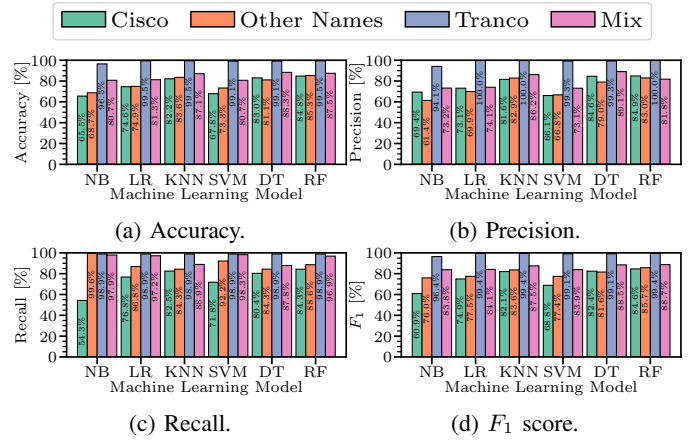


Fig. 5: Accuracy, precision, recall, and F_1 score of each ML model for the top 1415 domain names from *Other Names*, *Cisco*, and *Tranco*, plus a uniformly sampled *Mix* of 1415 domain names from the three lists, each vs. the 1415 domain names from *IoT M2M Names*.

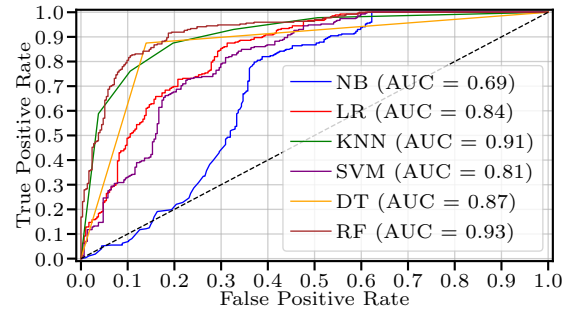


Fig. 6: Receiver Operation Characteristic (ROC) Curves for the top 1415 domain names from *Other Names*. The Area Under the Curve (AUC) is provided in the legend.

Cisco, or *Tranco* was used, is NB. NB achieved values below 70% for the four parameters when *Cisco* was used and low accuracy, precision, and F_1 scores when *Other Names* and *Mix* were used. NB, however, achieved high recall values of values $> 97.9\%$ when *Other Names*, *Tranco* and *Mix* were used. The best-performing overall model with all the lists is RF. RF achieved close to 99% for the four parameters when *Tranco* was used. Moreover, RF achieved slightly lower values for the four parameters for the rest of the lists, achieving values between 87% and 92% for the four parameters with *Cisco* and *Mix*. Even lower values were achieved for the four parameters when *Other Names* was used with values ranging between 79.3% and 82%. The performance of the models is also visualized in Figure 6, which shows the Receiver Operating Characteristic (ROC) curves plotted for every model when *Other Names* is used. ROC curves show the performance of the models at different classification thresholds. A comparison can be done between the performances of the models by comparing the Area Under the Curve (AUC) of each one. In our case, the ROC curves in Figure 6 further show the superiority of RF

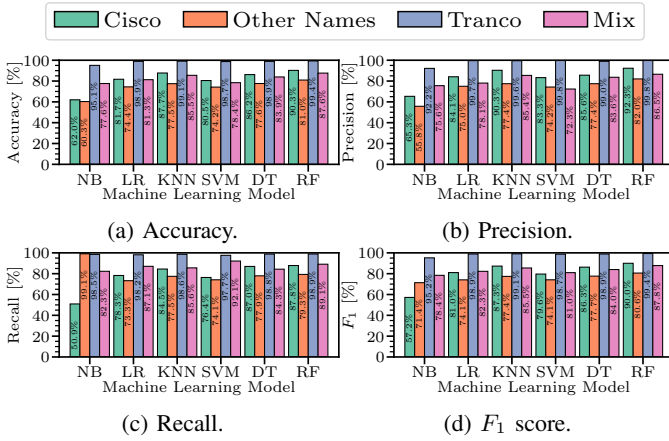


Fig. 7: Average accuracy, precision, recall and F_1 score of each ML model for 20 random picks of 1415 domain names from *Other Names*, *Cisco*, and *Tranco*, plus a uniformly sampled *Mix* of 1415 domain names from the three lists, each vs. the 1415 IoT domain names.

compared to the other DTs where its AUC = 0.93. NB, as expected from the previous measurements, has the lowest AUC of 0.69 and, therefore, has the lowest performance between the six models.

2) **Results when using random domain names from *Other Names*, *Cisco*, and *Tranco*:** The results of training the models when using random 1415 domain names from *Other Names*, *Cisco*, and *Tranco* are presented in Figure 7. Each graph represents one of the four parameters obtained by the different models. We trained the six models by randomly choosing 1415 domain names from each list to generalize our results further. This is particularly interesting when using *Cisco* and *Tranco*, each containing close to 1 million domain names. For each list, 100 random picks of 1415 domain names were made, and the results presented are the average of each of the four parameters over the 100 random picks.

The results over the 100 random picks are consistent with the previous results obtained when using the top domain names of *Other Names*, *Cisco*, and *Tranco*. The models performed best when *Tranco* was used, with NB having the lowest performance and the rest of the models achieving values > 98% for the four parameters. The lowest performing model, regardless of the list used is still NB, particularly when *Cisco* is used as a non-IoT dataset where it achieved values in the order of 60% for the four parameters. NB, however, achieved high recall values of values > 82.3% when the lists other than *Cisco* were used. The best-performing model, regardless of the list used is still RF. RF achieved values in the order of 99% for the four parameters when *Tranco* was used and had the lowest performance when *Other Names* was used with values in the order of 80% for the four parameters.

B. Cross Validation

We use cross-validation to assess the robustness of our models and their ability to generalize to unseen data. When

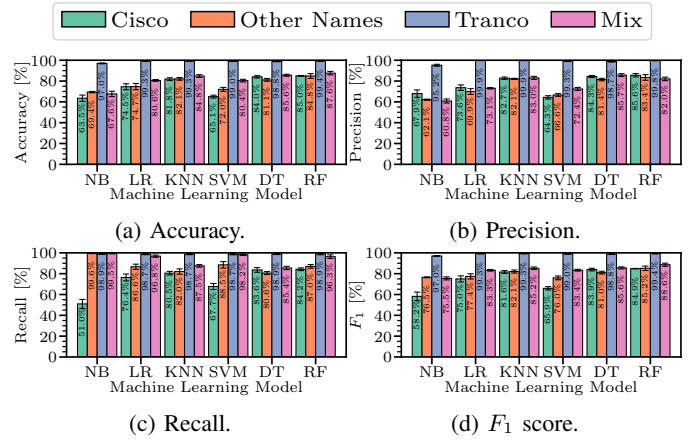


Fig. 8: Mean (colored bars) and standard deviation (error bars) of accuracy, precision, recall and F_1 score of the ML models over five folds for the top 1415 domain names from *Other Names*, *Cisco*, and *Tranco*, plus a uniformly sampled *Mix* of 1415 domain names from the three lists, vs. the 1415 IoT domain names.

assessing a model using cross-validation, the dataset containing all the classes is divided into K folds or subsets, and the model is trained K times. During every training instance, one of the K folds is used as a testing dataset, while the remaining $K - 1$ are used for training. We use Stratified K -fold cross-validation to ensure that the distribution of classes in the folds is similar to their distribution in the original dataset. Given the size of the *IoT M2M Names* list, we used $K = 5$ to ensure that each fold contains enough entries to provide a reliable estimate of performance. $K = 5$ allows each model to be trained five times. From *Other Names*, *Cisco*, and *Tranco*, we choose the top 1415 domain names, which are added to the 1415 domain names of *IoT M2M Names*. We show the results in Figure 8 as averages and standard deviation values of the evaluation parameters over the five folds.

The colored bars in Figure 8 represent the mean of the four parameters over the five folds, and the error bars at the top of each colored bar represent the standard deviation. We notice that the means of the four parameters over the five folds are consistent with the results from the performance evaluation we performed in Section IV-A while having a low standard deviation which indicates that the models are stable across the folds and that they are likely to generalize well to unseen data.

C. Ablation Test

An ablation test includes removing elements from the ML model or suppressing a subset of the features to study the effect they might have on the performance. We perform the ablation test by removing one label at a time by replacing the 32-dimensional vector representing the label with zeros. We perform 40 training and testing sessions, ablating one label in both the training and the testing datasets every time, training, and finally evaluating the models. The results for

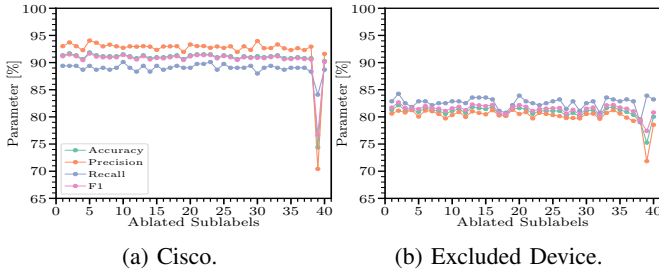


Fig. 9: Ablation Test with Random Forest (RF).

RF when used with *Other Names* and *Cisco* are presented in Figures 9a and 9b.

The stable performance observed when ablating the dummy labels ('*') demonstrate that the padding we added to the left of each domain name held no information and did not alter the performance of the models. The second-to-last label, which is the second-level domain (label 39), seems to have the highest impact on the performance as the values drastically dropped in both figures. For example, the accuracy and precision in Figure 9b dropped by around 15 and 25 percent, respectively. When the last label—the TLD of the domain name (label 40)—was ablated, however, values of the parameters did not experience the same drastic decrease, and the effect of ablating label 40 seemed equivalent to ablating the dummy labels. This shows that the second-level domain of a domain name is the most indicative of its class, while other labels including the TLD do not provide information about it. This is consistent with the results obtained in Figure 3, which showed that the TLDs are common between all the lists of domain names. Therefore, they are not distinctive about the class of the domain name.

V. DISCUSSION

The size of *IoT M2M Names*.

Despite the large amounts of raw data we started with, the size of *IoT M2M Names* remained relatively comparatively modest after removing duplicates and sanitizing the data. This is primarily due to the scope of our study which covers IoT devices that engage in M2M communications. Such devices exhibit limitations in their functionalities compared to IoT devices that are not strictly M2M which explains the low number of servers on the Internet these devices contact. Hence, we noticed a low number of frequently contacted servers in contrast to numerous servers that are less regularly or rarely contacted.

Usage and limitations of top-lists. We used 2 known top-lists, namely *Cisco* and *Tranco*, to evaluate the validity of using such lists as negative class in contexts similar to ours. We noticed that *Tranco* is the least valuable between the two as the majority of domains in it are second-level domains which does not reflect how domain names actually appear in DNS traffic. *Cisco*, on the other hand, is suitable as the entries in it are not limited to second-level domains and are included in the list as seen in DNS traffic. *Cisco* also resembles the domain

names in *IoT M2M Names* and *Other Names* statistically. The difference between *Cisco* and *Tranco* was most visible when training and testing the machine learning models. *Tranco* is easily distinguishable and so the models almost achieved perfect scores with it. The performance was different with *Cisco* which achieved a lower performance, but one which is comparable to when *Other Names*, the list that contains real domain names from real devices, was used. *Cisco* seems to be the better option to be used as a negative class in domain name classification problems with similar contexts.

Better sources of data. The domain names we used in this work, namely *IoT M2M Names* and *Other Names*, were extracted from packet captures of testbeds that had real devices. A better data source would undoubtedly require larger, more diverse, testbeds that have more devices conforming with our criteria for *IoT M2M Devices*. On the other hand, changing the scope to include devices other than strictly M2M ones, e.g., smart TVs, would certainly enlarge and diversify the list of domain names.

VI. RELATED WORK

The Identification of IoT devices is a popular topic in IoT research, primarily relying on machine learning techniques to accurately distinguish various features of IoT devices. The IoTFinder study [17] devised a multi-label classifier using machine learning techniques to identify IoT devices, generating a statistical fingerprint for each device using DNS traffic traces. The work in [25] presented a machine-learning-based model to identify the types of IoT devices connecting to a network and enforce security rules in vulnerable ones. The authors in [27] suggested a multi-stage machine learning classifier to identify IoT devices based on their network activity. The work in [30] evaluated four machine-learning-based approaches of IoT devices identification, concluding the need to retrain the models to avoid a performance drop. The work in [31] suggested an enhanced deep learning framework to identify IoT devices.

Domain name classification using machine learning, on the other hand, is also a recurring focus within research to detect, for example, phishing and DGA-generated domain names. The works in [1], [32] are recent systematic literature reviews about using Deep Learning (DL) techniques for phishing detection. The work in [13] suggests using DL techniques to detect phishing websites but using raw domain names and applying embedding to each character, conclude that using raw domain names is computationally less expensive than other techniques. The work in [14] also deals with raw URL data in what is referred to as a *lightweight* URL-based phishing detection and uses supervised machine learning techniques to extract features. The authors in [6] use supervised machine learning to detect phishing URLs based on features extracted from the URL, such as the hostname, full URL, and the Term Frequency-Inverse Document Frequency (TF-IDF), achieving an accuracy of up to 94%. The authors in [2] and [3] use Long Short-Term Memory (LSTM) networks to detect domain names generated by DGAs. Other techniques include using

Generative Adversarial Networks [33], full-convolutional systems [5], and semi-supervised learning [34].

VII. CONCLUSION AND OUTLOOK

In this paper, we studied the properties of domain names of servers contacted by *IoT M2M Devices* and trained several machine learning models to classify between *IoT M2M Names* and *Other Names*. We collected 12 public lists of domain names using past studies and two top lists, *Cisco* and *Tranco*. Our results showed that solely relying on the statistical properties of domain names does not indicate its type. We also observed that the TLDs of *IoT M2M Names* are common with *Other Names* and, therefore, are not indicative of their class. The machine learning models we trained, on the other hand, were successful in classifying *IoT M2M Names* and *Other Names*, with Random Forest having the best (overall) performance.

Looking forward, we aim to increase the size of the IoT dataset either by adding datasets from future works that have testbeds of real IoT devices or by setting up our own testbed. In addition, we aim to train our models against lists of known malicious domain names and domain names generated by DGAs to enhance the security aspect of our classifier. Future work may also include applying different word embedding techniques and feature extraction methods.

REFERENCES

- [1] N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions," *IEEE Access*, vol. 10, pp. 36429–36463, 2022.
- [2] Y. Qiao, B. Zhang, W. Zhang, A. Kumar, and H. Wu, "DGA Domain Name Classification Method Based on Long Short-Term Memory with Attention Mechanism," *Applied Sciences*, vol. 9, p. 4205, 10 2019.
- [3] J. Woodbridge, H. Anderson, A. Ahuja, and D. Grant, "Predicting Domain Generation Algorithms with Long Short-Term Memory Networks," *arXiv preprint arXiv:1611.00791*, 11 2016.
- [4] S. J. Saïdi, S. Matic, O. Gasser, G. Smaragdakis, and A. Feldmann, "Deep dive into the IoT backend ecosystem," in *Proceedings of the 22nd ACM Internet Measurement Conference*. ACM, Oct. 2022.
- [5] L. Yang, G. Liu, J. Wang, H. Bai, J. Zhai, and Y. Dai, "Fast3DS: A real-time full-convolutional malicious domain name detection system," *Journal of Information Security and Applications*, vol. 61, p. 102933, 2021.
- [6] R. Rao, T. Vaishnavi, and A. Pais, "CatchPhish: detection of phishing websites by inspecting URLs," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, 02 2020.
- [7] Cisco Umbrella, "Cisco Umbrella 1 Million." [Online]. Available: <https://s3-us-west-1.amazonaws.com/umbrella-static/index.html>
- [8] Tranco, "Tranco." [Online]. Available: <https://tranco-list.eu/list/K273W>
- [9] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoo, M. Korczyński, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," in *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, ser. NDSS 2019, Feb. 2019.
- [10] M. S. Lenders, C. Bormann, T. C. Schmidt, and M. Wählisch, "A Concise Binary Object Representation (CBOR) of DNS Messages," Internet Engineering Task Force, Internet-Draft draft-lenders-dns-cbor-06, Nov. 2023, work in Progress. [Online]. Available: <https://datatracker.ietf.org/doc/draft-lenders-dns-cbor/06/>
- [11] B. Schwartz, M. Bishop, and E. Nygren, "Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)," IETF, RFC 9460, November 2023. [Online]. Available: <https://doi.org/10.17487/RFC9460>
- [12] C. Amsüss and M. S. Lenders, "CoAP Protocol Indication," IETF, Internet-Draft – work in progress 04, March 2024. [Online]. Available: <https://datatracker.ietf.org/doc/html/draft-ietf-core-transport-indication-04>
- [13] S. Srinivasan, V. Ravi, A. Arunachalam, M. Alazab, and S. Kp, *DURLD: Malicious URL Detection Using Deep Learning-Based Character Level Representations*. Springer, 01 2021, pp. 535–554.
- [14] A. Butnaru, A. Mylonas, and N. Pitropakis, "Towards Lightweight URL-Based Phishing Detection," *Future Internet*, vol. 13, no. 6, p. 154, Jun 2021.
- [15] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting Similarities among Languages for Machine Translation," 2013.
- [16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *Proceedings of Workshop at ICLR*, vol. 2013, 01 2013.
- [17] R. Perdisci, T. Papastergiou, O. Alrawi, and M. Antonakakis, "IoTFinder: Efficient Large-Scale Identification of IoT Devices via Passive DNS Traffic Analysis," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2020, pp. 474–489.
- [18] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose, "SoK: Security Evaluation of Home-Based IoT Deployments," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 1362–1380.
- [19] J. Ren, D. J. Dubois, D. Choffnes, A. M. Mandalari, R. Kolcun, and H. Haddadi, "Information Exposure for Consumer IoT Devices: A Multidimensional, Network-Informed Measurement Approach," in *Proc. of the Internet Measurement Conference (IMC)*, 2019.
- [20] M. T. Paracha, D. J. Dubois, N. Vallina-Rodriguez, and D. Choffnes, "IoTLS: Understanding TLS Usage in Consumer IoT Devices," in *Proceedings of the 21st ACM Internet Measurement Conference*, ser. IMC '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 165–178.
- [21] ANT Lab, "IoT devices' First-Time Bootup Traces, PREDICT ID: USC-LANDER/IoT_Bootup_Traces-20161207. Provided by the USC/LANDER project." [Online]. Available: <http://www.isi.edu/ant/lander>
- [22] —, "IoT devices' First-Time Bootup Traces, PREDICT ID: USC-LANDER/IoT_Bootup_Traces-20181107. Provided by the USC/LANDER project." [Online]. Available: <http://www.isi.edu/ant/lander>
- [23] —, "10-day Operational IoT Traces, PREDICT ID: USC-LANDER/IoT_Operation_Traces-20200127. Provided by the USC/LANDER project." [Online]. Available: <http://www.isi.edu/ant/lander>
- [24] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications: Centralized and Federated Learning," 2022.
- [25] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "IoT SENTINEL: Automated device-type identification for security enforcement in IoT," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, jun 2017.
- [26] H. Kang, D. H. Ahn, G. M. Lee, J. D. Yoo, K. H. Park, and H. K. Kim, "IoT network intrusion dataset," 2019.
- [27] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2019.
- [28] A. Hamza, H. H. Gharakheili, T. A. Benson, and V. Sivaraman, "Detecting Volumetric Attacks on LoT Devices via SDN-Based Monitoring of MUD Activity," in *Proceedings of the 2019 ACM Symposium on SDN Research*, ser. SOSR '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 36–48.
- [29] Afnic and The Swedish Internet Foundation, "Zonemaster: Requirements and normalization of domain names in input." [Online]. Available: <https://github.com/zonemaster/zonemaster/blob/4ae8a6e/docs/specifications/tests/RequirementsAndNormalizationOfDomainNames.md>
- [30] R. Kolcun, D. A. Popescu, V. Safronov, P. Yadav, A. M. Mandalari, R. Mortier, and H. Haddadi, "Revisiting iot device identification," 2021.
- [31] Y. Liu, J. Wang, J. Li, H. Song, T. Yang, S. Niu, and Z. Ming, "Zero-bias deep learning for accurate identification of internet-of-things (iot) devices," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2627–2634, 2021.
- [32] C. Catal, G. Giray, B. Tekinerdogan, S. Kumar, and S. Shukla, "Applications of deep learning for phishing detection: a systematic literature review," *Knowledge and Information Systems*, vol. 64, no. 6, pp. 1457–1500, Jun. 2022.

- [33] J. Geng, S. Li, Z. Liu, Z. Cheng, and L. Fan, "Effective Malicious URL Detection by Using Generative Adversarial Networks," in *Web Engineering: 22nd International Conference, ICWE 2022, Bari, Italy, July 5–8, 2022, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2022, p. 341–356.
- [34] A. Faroughi, A. Morichetta, L. Vassio, F. Figueiredo, M. Mellia, and R. Javidan, "Towards website domain name classification using graph based semi-supervised learning," *Computer Networks*, vol. 188, p. 107865, 2021.

APPENDIX A

THE 12 DATASETS USED TO CONSTRUCT THE IoT M2M NAMES LIST

We used 12 datasets from previous works to construct *IoT M2M Names*. The datasets are: IoTFinder [17], YourThings [18], MonIoTr [19], IoTLS [20], three datasets from the USC/ISI ANT project [21]–[23], Edge-IIoTset [24], IoT Sentinel [25], IoT Network Intrusion Dataset [26], UNSW IoT traffic traces [27], and UNSW IoT attack traces [28]. Below is a summary of each one:

- **IoTFinder** [17]: IoTFinder is a multi-label classifier for detecting IoT devices by studying passively collected DNS traffic. The testbed contained 65 IoT devices. The data were collected between August 1, 2019, and September 30, 2019.
- **YourThings** [18]: A study of home-based IoT devices to assess their security properties. The testbed contained 65 IoT devices. The data were collected between April 10 and April 19, 2018.
- **MonIoTr** [19]: A study of information exposed in the traffic of consumer IoT devices. The testbed contained 81 IoT devices. The data were collected between March 28 and May 8, 2019, as well as on September 1, 2019.
- **IoTLS** [20]: A study about the use of TLS in consumer IoT devices. The testbed contained 40 IoT devices. The data were collected between January 2018 and March 2020.
- **USC/ISI ANT project** [21]–[23]: The ANT Lab is an Internet research group at the University of Southern California (USC) that has published several datasets related to various network topics e.g., traffic, outage, and DNS. We used three datasets from the USC/ISI ANT project. Two datasets contain the bootup traces of 6 and 11 IoT devices, respectively. The third dataset contains traffic observed in a network of 14 IoT devices over a period of 10 days.
- **Edge-IIoTset** [24]: An IoT traffic dataset that includes benign and attack traffic. The testbed contained 13 real IoT devices. The benign traffic, which we used in this paper, was collected between November 21, 2021, and January 10, 2022.
- **IoT SENTINEL** [25]: IoT SENTINEL is a security system that identifies devices present in the network and monitors traffic from vulnerable ones. The testbed contained 31 real IoT devices, and the traffic was collected during the setup of each device.
- **IoT Network Intrusion Dataset** [26]: An IoT traffic dataset that includes benign and attack traffic. The testbed

contained two real IoT devices. We used the benign traffic in this paper.

- **UNSW IoT traffic traces** [27]: A study about classification of IoT devices in Smart Home environments. The testbed contained 28 IoT devices, and the traffic was collected between October 2016 and April 2017.
- **UNSW IoT attack traces** [28]: A study about detecting volumetric attacks against IoT devices and the dataset includes benign and attack traffic. The testbed contained 10 IoT devices, and the traffic was collected for 16 days.

Ethical Considerations. This work does not raise any ethical issues.