An Attentive Interpretable Approach for Identifying and Quantifying Malware-Infected Internet-Scale IoT Bots Behind a NAT

Christelle Nader and Elias Bou-Harb
christelle.nader@my.utsa.edu,elias.bouharb@utsa.edu
The Cyber Center For Security and Analytics, University of Texas at San Antonio
San Antonio, Texas, USA

ABSTRACT

The explosive growth of the Internet-of-Things (IoT) paradigm has brought the rise of malicious activity targeting the Internet. Indeed, the lack of basic security protocols and measures in IoT devices is allowing attackers to use exploited Internet-scale IoT devices to organize malicious botnets, and cause significant damage to the Internet through Denial of Service (DoS) attacks, illicit scraping, and cryptojacking attacks. Such IoT botnets can be Internet-facing, or can also be deployed behind Network Address Translation (NAT) gateways that provide anonymity to the exploited bots. In this paper, we aim at detecting compromised IoT bots behind NAT gateways which could possibly generate malicious activities towards the Internet by leveraging large-scale macroscopic one-way darknet data. To the best of our knowledge, we are among the first to explore the capabilities of attentive interpretable tabular transformers to capture the nature of such nodes operating on one-way network traffic. Our results, which employed 2.6GB of darknet data, show that our approach was able to classify malware-infected NATed IoT bots with an accuracy of 93%, outperforming the state-of-the-art machine learning (ML) approaches. Additionally, we were able to infer around 4 million Internet-scale Mirai-infected NATed IoT bots and 16,871 unique NATed IP addresses. Results from this work put forward interesting future work in the area of network traffic analysis of NATed IoT bots for better Internet security, while highlighting the need for addressing the notions of attention and interpretability.

CCS CONCEPTS

• **Security and privacy** → **Network security**; • **Computing methodologies** → **Machine learning approaches**; • **Computer systems organization** → *Embedded and cyber-physical systems.*

KEYWORDS

Network traffic analysis, Internet of Things, IoT fingerprinting, Network Address Translation, Network Telescope, Machine Learning, Transformers, Attention

CF'22, May 17–19, 2022, Torino, Italy

ACM ISBN 978-1-4503-9338-6/22/05…$15.00
https://doi.org/10.1145/3528416.3530995

ACM Reference Format:
Christelle Nader and Elias Bou-Harb. 2022. An Attentive Interpretable Approach for Identifying and Quantifying Malware-Infected Internet-Scale IoT Bots Behind a NAT. In *19th ACM International Conference on Computing Frontiers (CF'22), May 17–19, 2022, Torino, Italy*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3528416.3530995

1 INTRODUCTION

The Internet serves billions of users every day, running a plethora of clients, while connecting billions of devices, including IoT devices. By 2023, the number of networked devices will reach 29.3 billion, up from 18.4 billion in 2018 [14]. Thus, assessing any element pertaining to the Internet on a global scale is challenging. With the explosive growth of deployed IoT devices over the past couple of years, comes the rise of malicious activity targeting the Internet. Due to the lack of basic security protocols and measures, IoT devices have become easy targets for exploitations and recruitment within coordinated IoT botnets [9], causing significant damage to the Internet and its related infrastructure. Such IoT botnets could perform malicious activities such as malware attacks, social engineering attacks, Distributed Denial of Service (DDoS) attacks, illicit scraping, and cryptojacking attacks [10, 11, 16, 22, 27].

Particularly, the Mirai botnet has attracted the attention of the research community after it infected more than 200,000 devices and performed the DDoS attacks of late 2016 [7]. Since then, we have seen an increase in the sophistication of malware targeting IoT systems such as the rapid proliferation of Mirai-based malware; Mirai being one of the most active IoT botnets to date [15]. In addition, with the recent exploitation of the Log4j2 web vulnerability, which enables threat actors to send a specially crafted request to launch a remote code execution attack [1], several botnet families (including Mirai) are recruiting IoT bots to exploit this web vulnerability. Consequently, detecting malicious activities generated by IoT bots becomes of paramount importance.

However, such IoT bots are not only Internet-facing, but can also be deployed behind NAT gateways. Indeed, the usage of NAT has grown exponentially over the last few decades as it allows several devices to share a limited number of public IP addresses in addition to providing Internet-wide services via port mapping. Consequently, the anonymity that NAT provides have induced the problem of identifying the nature of the NATed IoT bots. Indeed, NAT introduces numerous security issues and technical challenges in the IoT realm, including, but not limited to (1) under quantification or overestimation of the number of vulnerable devices found behind a NAT [18], which hinders IoT-centric botnet characterization and attribution, (2) the issue of legitimate IoT device/type/vendor identification and

characterization, and (3) the sound and comprehensive analysis of IoT malware evolution residing on NATed IoT devices. Broadly, fingerprinting IoT bots behind a NAT would aid in network and security provisioning, and cyber forensic triage.

Many researchers have leveraged ML and deep learning approaches in order to fingerprint such nodes due to their high efficiency. However, deep learning solutions lack in interpretability because of their black-box approaches. To this end, a new line of deep learning models "Transformers" came to light. Such models are widely used in the field of natural language processing (NLP) for various tasks, such as document classification, document entanglement, sentiment analysis, sentence similarity, etc. Such models are known for their interpretable feature provided by their "Self-Attention" mechanism which differentially weighs the significance of each part of the input data. Indeed, they have achieved a degree of performance competitive with popular, shallow and deep learning techniques such as Random Forest, gradient boosting, and Recurrent neural networks (RNNs).

Broadly, there still exist several challenges that need to be overcome to enable effective fingerprinting of exploited IoT bots residing behind a NAT: (1) There is a lack of visibility related to Mirai-infected NATed IoT bots on a macroscopic level. To this end, we leverage network telescopes (darknet data), i.e., a passive traffic monitoring system built on a globally routed set of unused IP addresses, which are able to capture over 1 million of packets per second to infer and characterize such nodes. (2) There is a lack of research work that have explored transformers with tabular data. Indeed, deep neural networks (DNNs) have shown significant success with images, text, and audio data types [6, 19, 24]. However, one data type that has yet to see such success is tabular data. Despite being the most common data type in the real-world, deep learning for tabular data remains under-explored, with ensemble decision trees (DTs) dominating most applications due to their plethora of benefits. (3) There is a lack of Internet-scale measurement techniques to provide near real-time continuous situational awareness for Mirai-infected NATed IoT bots as well as a lack of existing ground truth in this context. To this end, we innovate a ground truth dataset in order to train machine learning algorithms to identify such nodes.

To motivate empirical IoT cyber security initiatives as well as aid in reproducibility of the obtained results, we make the datasets and source codes of all the developed methods available to the research community at [2].

Contributions. Motivated by this research direction and the aforementioned challenges, we make the following contributions in this work:

- We build a ground truth dataset by exploiting a weakness in the packet generation algorithm and random number generation of the Mirai IoT malware. We leverage more than 113K Mirai scanning events obtained from darknet traffic over a total period of 6 days.
- We propose and evaluate a new approach to infer, characterize, and attribute Internet-scale NATed IoT bots by leveraging passive empirical measurements and transformers. To the best of our knowledge, we are among the first to explore such an approach. Our results show that our implementation outperforms the state-of-the-art shallow learning approaches. We limit our comparison to shallow learning approaches due to the fact that most popular deep learning approaches are either not applicable in our case or cannot be implemented due to the tabular nature of our data.
- We report on close to 4 million Internet-scale Mirai-infected NATed IoT bots and 16,871 unique NATed IP addresses. We also generate amalgamated statistics related to these inferred and exploited IoT bots, including but not limited to their country of origin and their organization type.

Organization. The rest of this paper is organized as follows. In the next section, we detail our approach and rationale. In Section 3, we empirically evaluate it using darknet data, while reporting on its accuracy metrics. In Section 4, we infer, measure, and characterize the Internet-Scale NATed IoT devices. In Section 5, we elaborate on the related work. Finally, in Section 6, we draw some conclusions and pinpoint a few endeavors which aim at paving the way for future work.

2 PROPOSED APPROACH

In this section, we present our approach and its related components. In Section 2.1, we build a ground truth in order to infer and characterize Mirai-infected NATed IoT bots on a macroscopic level. In Section 2.2, we leverage this ground truth as well as transformers to classify such nodes.

2.1 Building the ground truth

The idea behind building the ground truth is leveraging a weakness in the packet generation algorithm and random number generation of the Mirai IoT malware in order to label darknet packets as IoT bots behind a NAT. We particularly focus on Mirai-based malware due to its rapid proliferation as well as Mirai being one of the most active IoT botnets to date. In addition, as Mirai appears to have spurred hundreds of variants in the wild [15], by leveraging a weakness in Mirai's source code, we would be able to not only label Mirai-infected IoT bots, but also some of its variants. To this end, we use the methodology described by Griffioen and Doerr [18] on two separate network telescopes data captures to obtain a list of NATed Mirai packets.

Algorithm 1: Building the Ground Truth

```
Data: Nated_df, mirai_dump
Result: NATed Infected Packets
mirai_dump ⇐ darknet data
for packet1 in mirai_dump do
    for packet2 in mirai_dump do
        if packet1.ipsrc = packet2.ipsrc then
            if packet1.tcp_win ≠ packet2.tcp_win or
            packet1.srcPort ≠ packet2.srcPort or
            packet1.tcp_seq ≠ packet2.tcp_seq or
            packet1.dstPort ≠ packet2.dstPort then
                Nated_df ⇐ packet1, packet2
            end
        end
    end
end
```

After the Mirai source code was posted on the Internet, many copycats entered the scene by recycling Mirai's source code and

introducing minor changes to create their own IoT botnets. Despite such alterations (e.g., the passwords used or ports targeted), the scan and probe packets remained unchanged from the original Mirai. Thus, many IoT malware share behavioral characteristics that we leverage to build the ground truth.

The way Mirai generates its scan and attack packets exhibits some particularities. Initially, after start-up, the malware instantiates a custom-built random number generator (RNG). In addition, the source port and window size are randomly generated from the RNG but are fixed throughout the entire execution of the malware, i.e., these header values are the same until the device is cleaned up or rebooted.

Thus, if a device is assigned a new IP address, we are able to link it to a previous IP address due to the same configuration values. However, the session-permanent source port and window size are not sufficient to conclusively link packets to a specific infection. Therefore, we would also need other packet features such as the TCP sequence number and destination ports. The logic behind this algorithm is shown in Algorithm 1.

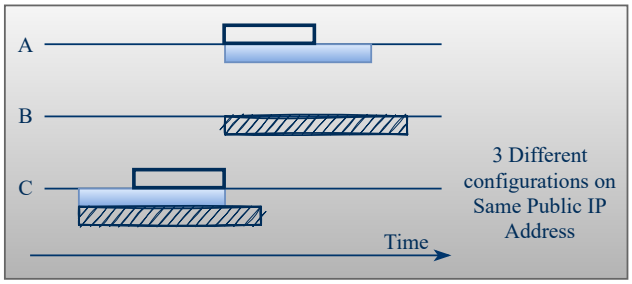


Figure 1: Scenario where IP churn is combined with a NAT

Figure 1 explains the scenario where we consider multiple continuous infections on different IP addresses in combination with a NAT. Consider three infections at hosts behind a NAT with the public IP address C (which are identifiable due to the different configuration values). Although possible, multiple Mirai infections originating from one IP are unlikely. Thus, several infections on one IP alone are not enough to verify the use of a NAT. However, if the infections churn to several IP addresses rather than a single IP, these infections have to be located at different devices behind a single IP address. In our example, if the infections from C churn to two separate IP addresses (i.e., IP address A and B), this would allow us to identify an IP address in a NAT.

2.2 Classifying NATed IoT Bots

We leverage the ground truth and a new canonical DNN architecture for tabular data, TabNet [8] in order to classify NATed IoT bots. We specifically use this ML model because of its ability to extract interesting insights more efficiently when compared to the labeling function described in Algorithm 1. Indeed, TabNet is trained using gradient descent-based optimization, which enables flexible integration into end-to-end learning. It also enables interpretability and better learning since its learning capacity is only used for important features. Figure 2 shows the general idea behind TabNet.

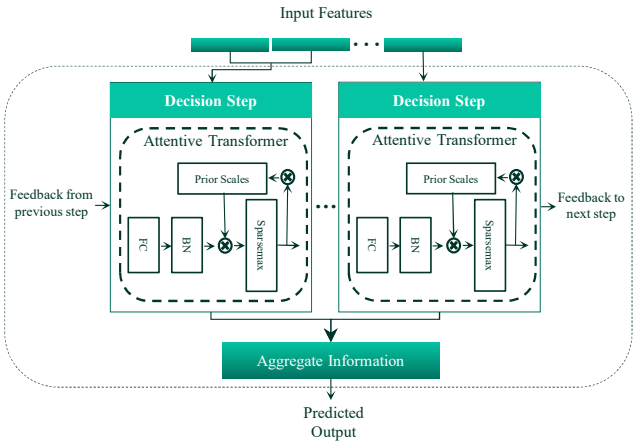


Figure 2: TabNet Transformer

In addition, TabNet employs a single deep learning architecture for feature selection and reasoning. Initially, batch normalization (BN) is applied to the raw numerical features instead of considering any global feature normalization. The same D-dimensional features $\mathbf{f} \in \mathfrak{R}^{B \times D}$ are passed to each decision step, where B is the batch size. This model also leverages the mapping of categorical features with trainable embeddings. Moreover, TabNet's encoding is based on a sequential multi-step processing with N_{steps} decision steps. The i^{th} step inputs the processed information from the $(i-1)^{th}$ step in order to decide which features should be used and outputs the processed feature representation to be aggregated into the overall decision.

Moreover, TabNet employs a learnable multiplicative mask **M[i]** $\in \mathfrak{R}^{B \times D}$ for soft selection of the important features. The attentive transformer is used to obtain such masks by using the processed features from the preceding step, **a[i-1]**: **M[i]** = sparsemax(**P[i-1]**.h$_i$(**a[i-1]**)), where **P** denotes the prior scale term, i.e., how much a specific feature has been previously used, and $\mathbf{h}_i$ is a trainable function that uses a single fully connected layer (FC) followed by BN. The latter FC employs a sparsemax normalization due to its superior performance and ability to select sparse features for explainability. The filtered features are then split for the decision step output and information for the subsequent step, **[d[i], a[i]]** = f$_i$(**M[i]** . **f**), where **d[i]** $\in \mathfrak{R}^{B \times N_d}$ and **a[i]** $\in \mathfrak{R}^{B \times N_a}$.

3 IMPLEMENTATION AND EVALUATION

This section details our implementation as well as its experimental results. In Section 3.1, we present the datasets used in our approach. In Section 3.2, we detail the preprocessing steps taken. In Section 3.3, we assess TabNet according to several accuracy metrics and compare its performance with known ML algorithms. Lastly, in Section 3.4, we assess the soundness of TabNet's output according to a heuristic methodology that we devise. We implement our algorithms using Python's Scikit-learn libraries. Additionally, we implement TabNet according to its pyTorch implementation [3]. Moreover, we evaluate our approach on a Ubuntu 18.04.5 LTS machine with a 62.6GB memory, an Intel Xeon(R) W-2145 CPU @3.70GHz

x 16, and a hard disk of 251GB. The datasets and source codes of all the developed methods are made available to the research community at [2].

3.1 Datasets and Findings

We leverage two separate network telescopes data captures to obtain the list of NATed Mirai packets. The first dataset comprises of 33,404 Mirai scanning events (around 800MB of data) collected during a three-hour period on October 20, 2021. The second dataset comprises of 80,000 Mirai scanning events (around 1.8GB of data) collected during a 5-day period from November 20, 2021 till November 24, 2021. Our findings are two-fold:

- **October 20, 2021 Data.** We label 831,482 packets (33.22%) as NATed Mirai packets. Out of these packets, we classify 2,513 unique addresses behind a NAT.
- **November 2021 Data.** We label 4,001,644 packets (20%) as NATed Mirai packets. Out of these packets, we classify 17,315 unique addresses behind a NAT.

3.2 Data Prepocessing

For consistency, completeness and soundness purposes, we perform a data preprocessing step on the captured traffic which includes omitting the source and destination IP addresses that have no contribution to classification. The remaining data consists of 21 features (e.g., $dstPort$, tcp_win, $srcPort$, etc. [12]).

The data that we collected is highly imbalanced, e.g., the NATed Mirai packets only constitute 33.22% and 20% of the total packets for the October 20, 2021 data and November 2021 data respectively. Seeing as Mirai generated packets are generally rare like other threats that are measured in the wild, and having only approximately 830K of NATed Mirai samples, we create two new balanced datasets, that we use in our implementation, comprising of 830K random samples of NATed Mirai packets and 830K random samples of not NATed Mirai packets for each dataset. In addition, we divide the October 20, 2021 data into three sets, namely, a training set consisting of 80%, a validation set consisting of 10%, and a testing set consisting of 10%.

For the other ML algorithms that we implement, each training set consists of 80%, and each testing set consists of 20%. In addition, we scale all the numerical features such that all feature inputs would be in the range of [0,1].

3.3 Experimental Results

To validate the effectiveness of TabNet, we assign accuracy as the evaluation metric in the $fit()$ function of the model. Additionally, we monitor the training time. We first train the model on the October 20, 2021 dataset and then test it on two datasets: (1) the October 20, 2021 dataset, and (2) the November 2021 dataset to validate the resiliency of our model against classification decay. For the second dataset, we divide the October 20, 2021 Data into two sets, namely, a training set consisting of 80%, and a validation set consisting of 20%. Additionally, the testing set consists of all the data from the November 2021 dataset. We then compare the obtained results with other known machine learning algorithms, namely, Logistic Regression with Restricted Boltzman Machine (RBM), Logistic Regression, Light Gradient Boosting Machine (LGBM), Linear Support Vector Classifier (LinearSVC), Random Forest (RF), Gaussian Naive Bayes, and Multi-layer Perceptron (MLP).

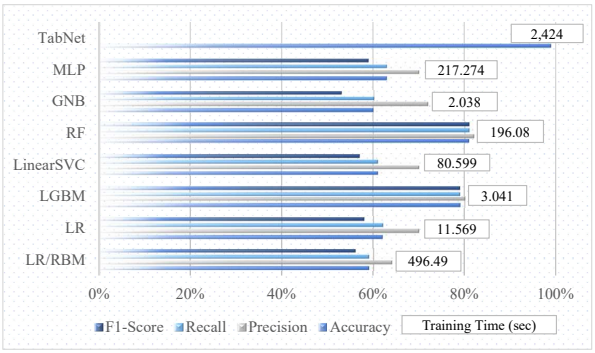
Particularly, the Logistic Regression with RBM is a semi-supervised ML algorithm that is based on Logistic Regression with Bernoulli RBM. Indeed, Bernoulli RBM estimates its parameters by using a Stochastic Maximum Likelihood (SML). In addition, we employ the "newton-cg" solver because it yielded the best results. Initially, we instantiate an RBM features' classifier which consists of pipelining an RBM model with a Logistic Regression model. We then implement the GridSearchCV function which converges towards the best parameter values for these models. As a result, the best parameter values are obtained as: Logistic C (10,000), RBM Learning Rate (0.01), and RBM Number of Components (200). Moreover, for the MLP classifier, we employ a "tanh" activation and a Stochastic Gradient Descent "sgd" solver because they yielded the best results.

We evaluate these deployed ML algorithms according to several metrics such as the accuracy, the training time (in sec), and the average Precision, Recall, and F1-Score. Figure 3 demonstrates the obtained results.

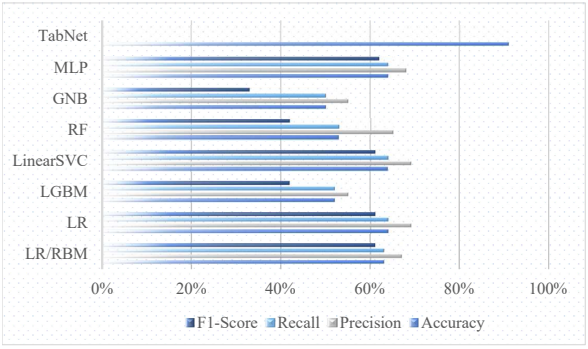
By comparing Figure 3a with Figure 3b, we show that Tabnet outperforms all the other ML algorithms implemented, with an accuracy of 99% and 91% for the October 20, 2021 data and November 2021 data respectively. In addition, the results show that TabNet has a training time of approximately 40 minutes, which is an acceptable result. Although the RF model achieved a high overall accuracy with the October 20, 2021 dataset (93%), due to classification decay, when testing the trained model on the November 2021 data, it performed poorly with an accuracy of 53%. Indeed, in a real-life scenario, it is unwise to assume that the relation between the input data and the target variable of an ML model will remain unchanged over time [17]. Thus, it is most likely that the model will become obsolete when the incoming data at testing time is different from that of training time. However, classification decay typically occurs when the training time and testing time are months or years apart [20, 28]. Therefore, it is surprising to witness this phenomenon in only a one month period; an observation that can only reinforce the ever-evolving malware threat landscape.

Features' importance. To better understand why the RF model displayed such a drop in accuracy, we look into the features' scores from the RF model. We discover five features that have the most impact: $dstPort$, tcp_win, $srcPort$, tcp_seq, and ip_id. Four of these features are directly related to the ones used in Algorithm 1. Thus, such a drop in accuracy can possibly be explained by the constant change in the chosen session-permanent source and window size, as well as the TCP sequence number and destination ports. We also look into the features that provide little to no impact on the model. We find eight features: TCP_OPT_SACK, $tcp\_reserve$, tcp_urp, $tcpdatalen$, tcp_ack_seq, $timestamp$, $prtcl$, and tcp_flag. We decide to drop these features and re-train the model on the October 20, 2021 data and test it on the November 2021 data. Upon further evaluation, we discover that although the RF model achieved a higher accuracy (62% versus 53% before dropping the features), it still performs poorly compared to TabNet.

We then shed light on the features that were most decisive in the learning process of TabNet. Based on the features' scores derived from the model, we find that five features have the most impact



(a) October 20, 2021 Data



(b) November, 2021 Data

Figure 3: Experimental Results

on TabNet: *TCP_OPT_TIMESTAMP*, *prtcl*, *ip_id*, *tcp_off*, and *TCP_OPT_SACK*.

Table 1: Summary of the most important features used by TabNet

Feature	Description
TCP_OPT_TIMESTAMP	Current roundtrip time (RTT) of the network between endpoints
prtcl	IP protocol
ip_id	ID used to re-associate fragmented packets
tcp_off	Total size of a TCP header in multiples of four bytes
TCP_OPT_SACK	The left and right edges of data that has been received beyond the packet's acknowledgment number

These features are summarized in Table 1. The *ip_id* and the *TCP_OPT_TIMESTAMP* can indicate an automatic and systematic scanning generator, whereas the *prtcl*, *tcp_off*, and *TCP_OPT_SACK* can be an indicator of a broad probing behavior. Therefore, such behaviors can be directly correlated to malware machinery. Although the RF model and TabNet have one feature in common (*ip_id*), TabNet's features are more indicative of the malware that is generating the probing behavior.

We finally look into TabNet's architecture to try and explain why it outperformed all the other models. Indeed, with a specific design, conventional DNN can be leveraged to implement a DT-like output manifold based on a linear combination of features where coefficients determine the proportion of each feature. TabNet is based on such a functionality to reap the benefits of DT-based algorithms while outperforming them. This is not only due to the use of sparse instance-wise feature selection learned from data but also by the learning capacity via non-linear processing of the selected features [8]. Consequently, it is no surprise that TabNet outperformed other DT-based algorithms such as the Random Forest.

3.4 Vetting Methodology

We design a heuristic vetting process in order to ensure that TabNet's output is sound. Our methodology depicts a two-step process. In step 1, we look up the obtained NATed addresses in the Shodan search engine to check what kind of results we obtain. Indeed, Shodan is the world's first search engine for Internet-connected devices that allows users to search for various types of servers, routers, or IoT devices that are connected to the Internet using a variety of filters [5]. We devise a binary scoring mechanism that we apply on the obtained results. We assign the binary number 1 to an address that has a higher likelihood of being behind a NAT and a binary number 0 for an address that has a lower likelihood of being behind a NAT. We summarize the results and assign this binary number to each IP address according to three use cases:

- **No Banners.** In the case where we obtain no results, there is a high probability that the IP address is behind a NAT. Thus, we assign this IP address the binary number 1.
- **Many Banners / Many Devices.** In the case where we obtain several banners originating from different kinds of devices, there is a high probability that the IP address is behind a NAT. Thus, we assign this IP address the binary number 1.
- **Few Banners / One Device.** In the case where we obtain one or few banners originating from one type of device, there is a need for further exploration. If the protocol UPnP is found, this indicates that the IP address is behind a NAT but used port forwarding to become accessible. Thus, in this specific case, we assign this IP address the binary number 1. Aside from this case, no conclusive result can be acquired, and therefore we assign this IP address the binary number 0.

In step 2, we filter the IP addresses based on their score. If the score is 1, then we can assume that the IP address is in fact behind a NAT and that our approach did not yield a false positive. We perform this vetting process on the results of the classifier after testing it on the November 2021 dataset. Initially, we pass a list of the unique IP addresses that were classified as NATed to the Shodan search engine. Out of the 4,001,644 NATed Mirai packets, we find 17,315 unique addresses behind a NAT. After looking them up in the

Shodan search engine, only 510 addresses gave us back results. We further explore these addresses according to the aforementioned use cases. Finally, we obtain 16,871 IP addresses that are deemed behind a NAT. Therefore, we only dropped 444 IP addresses (i.e., only 2.564% of the unique NATed IP addresses), which were considered false positives.

4 QUANTIFICATION AND CHARACTERIZATION

Our goal is to infer and measure Internet-scale malware-infected NATed IoT bots. To this end, we leverage the output of TabNet and index the obtained insights, including near-real-time information related to Internet-scale malware-infected IoT bots behind a NAT coupled with their geolocation information. To characterize the hosting environments of such nodes, we executed geo-location procedures by employing the MaxMind GeoIP2 database [4] and leveraging the list of IP addresses that remained after the vetting process.

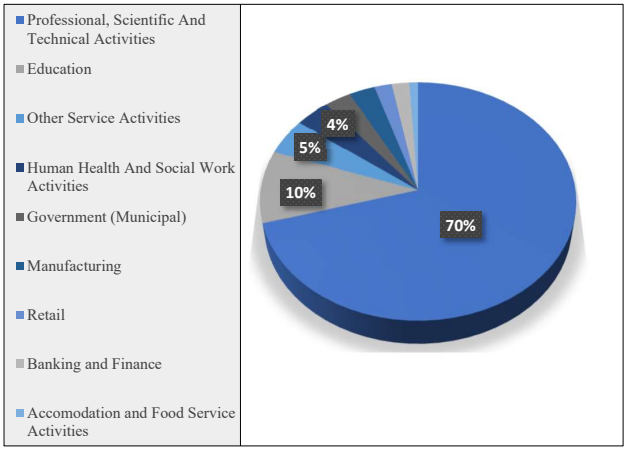


Figure 4: Distribution of NATed exploited IP addresses by organization type

In total, after analyzing the darknet data from November 20 till November 24 2021, we were able to find 3,879,426 malware-infected IoT devices behind a NAT, with an average of 230 IoT devices per IP address. We discovered that most of the IoT bots are targeting web services on ports 80, 81, 8080, and 8181. Other ports targeting SSH were also observed. We then investigated the distribution of exploited IP addresses behind a NAT by organization type as illustrated in Figure 4. We initially found that the majority of those IP addresses belong to the Information and Telecommunication sector (mainly Internet Service Providers), which makes sense as ISPs host the majority of NATed infrastructures such as home NATs. Consequently, we were more interested in the distribution of the remaining IP addresses as shown in Figure 4. We found that these remaining IP addresses belong to Professional, Scientific, and Technical activities (70%), Education (10%), Other Service activities such as Business Conglomerate (5%), Health (4%), Government such as municipal administrative activities (3%), and Manufacturing such as manufacture of basic pharmaceutical products and pharmaceutical preparations (3%), to name a few. Indeed, professional, scientific, and technical activities include, but are not limited to, private or public testing and research facilities (e.g., university labs, technical testing and analysis), or professional and private services (e.g., specialized design activities). Additionally, the health sector includes, but is not limited to, residential care activities or any activity related to human health and social work.

Table 2: Distribution of NATed exploited IoT bots by ISP

ISP	Country	# Bots	%
China Unicom Liaoning	China	1,327,332	35%
China Telecom	China	412,128	11%
National Telecom. Corp. HQ	Pakistan	166,168	4%
Hathway	India	113,372	3%
BSNL	India	107,476	3%
MTS PJSC	Russia	97,678	3%
Rostelecom	Russia	64,550	2%
China Mobile Guangdong	China	60,826	2%
Paulino Perreira Dos Santos ME	Brazil	56,396	1%
Wantel Tecnologia Ltda. Epp	Brazil	54,724	1%
Spectrum	U.S.	52,590	1%
VNPT	Vietnam	43,386	1%
HiNet	Taiwan	41,690	1%

Since the majority of the IP addresses belonged to Internet Service Providers, we proceeded by characterizing the distribution of the exploited IoT bots behind a NAT by ISP. The results are shown in Table 2. We find that the top 2 ISPs, namely, China Unicom Liaoning, and China Telecom, comprise 35% and 11% of the total number of NATed malware-infected IoT bots respectively. These ISPs are then followed by an ISP originating from Pakistan (4%) and two Indian ISPs (3% for both). We note that the United States falls in eleventh place with its ISP (Spectrum) only comprising of 1% of the total number of malware-infected IoT bots behind a NAT.

We finally proceeded by illustrating the worldwide distribution of top countries hosting exploited IoT bots behind a NAT as depicted in Figure 5. Intuitively, this outcome is affected by the investigated IP addresses, the specific darknet data sample that has been utilized, and the timeframe of the executed analysis. We find that China takes the lead by hosting 48% of the malware-infected IoT bots behind a NAT, followed by India (8%) and Russia (7%). We note that the Unites States and Brazil each hosts 5% of those bots.

5 RELATED WORK

In this section, we elaborate on three related topics that are relevant to our work herein, namely, IoT device fingerprinting, device identification behind a NAT, and application-based transformers for tabular data.

IoT Fingerprinting. Perdisci et al. [29] developed IoTFinder, an independent system for large-scale IoT device identification. Indeed, their proposed approach leveraged distributed passive DNS data which is comprised of more than 40 million clients. This collected data was then passed on to a multi-label machine learning-based classifier that only uses DNS fingerprints to classify IoT devices. Additionally, Kumar et al. [23] performed an in depth comparative

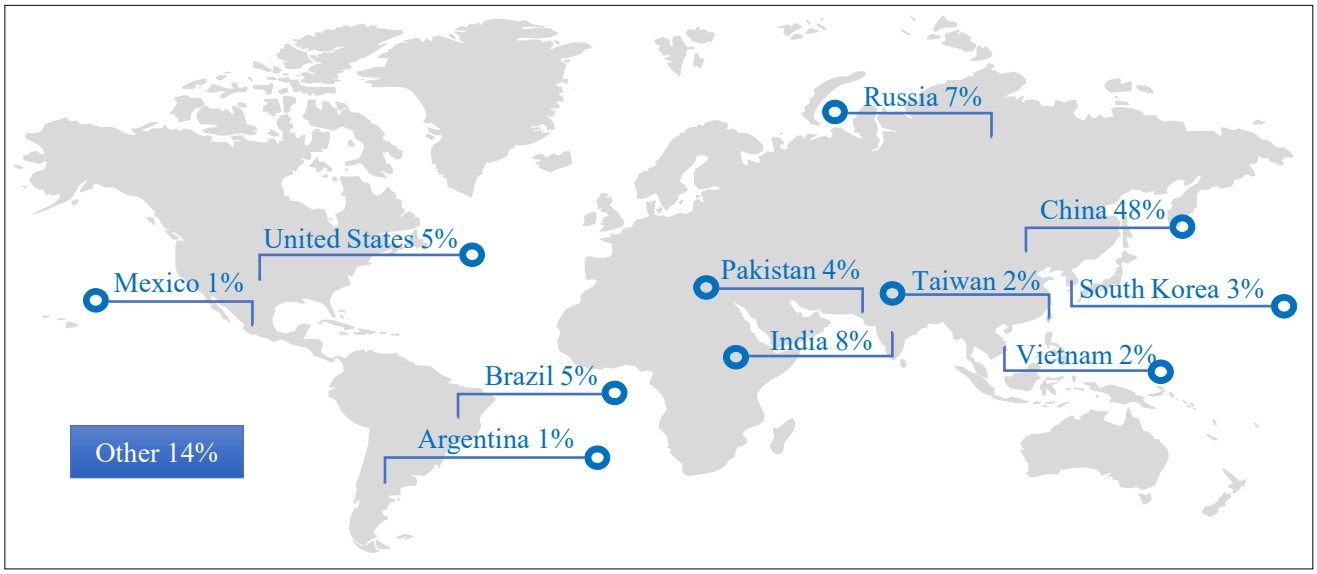


Figure 5: Global distribution of NATed exploited IoT bots

analysis of several machine learning algorithms by using different effective features extracted from IoT network traffic. Indeed, they leveraged 20 days of network traces generated from 20 popular IoT devices in order to evaluate the performance of those ML algorithms. Moreover, Pour el al. [30] leveraged network telescopes to classify compromised IoT devices from one way-network traffic by developing a multi-window convolutional neural network. By analyzing 3.6 TB of darknet traffic, their approach effectively uncovered 440,000 compromised IoT devices and 350 IoT botnets in the wild. Further, Cabana et al. [13] investigated the threat landscape of Industrial Control Systems (ICSs) devices by leveraging network telescope traffic and ML. Indeed, their tool generates threat intelligence by using Deep Packet Inspection (DPI) techniques on scanning campaigns that target ICSs. By analyzing 12.8 TB of darknet traffic, they were able to classify the sources behind the campaigns as well as their threat actors.

Device Identification behind a NAT. Khatouni et al. [21] proposed a passive supervised machine learning methodology to detect hosts behind NAT devices by using flow level statistics without any application layer information. Indeed, the authors captured a large dataset and performed an extensive evaluation with four existing approaches from the literature. Their results showed that their methodology can identify NAT behaviors and hosts with high accuracy. In addition, Yang et al. [32] proposed a methodology to identify NATs for online IoT devices based on Tri-Net; a semi-supervised DNN, by learning features on three layers, namely network, transport, and application layer in a small labeled data set. After evaluating this approach on a real-world dataset with more than 8 million online IoT devices, the authors were able to efficiently find 2,511,499 NATed IoT devices. Moreover, Meidan et al. [26] proposed a supervised machine learning-based method that can detect specific vulnerable IoT device models that are connected behind a domestic NAT. After evaluation, the authors showed that their flow-based method is robust and can detect IoT devices behind a home NAT with high accuracy.

Application-based transformers for tabular data. Yin et al. [33] proposed TABERT, a pretrained language model that jointly learns representation from both textual and tabular data. In addition, Yoon et al. [34] presented VIME, a novel tabular data augmentation method for self- and semi-supervised learning frameworks for the genomics and clinical data domains. Moreover, Somepalli et al. [31] proposed SAINT, a neural network methodology for tabular data via row attention and contrastive pre-training. Their findings showed that SAINT improves the performance over previous deep learning models and outperforms gradient boosting methods on average over several benchmark tasks. Further, Arik and Pfister [8] presented TabNet, a novel high-performance and interpretable tabular learning architecture that yields feature attributions and insights into its global behavior.

This paper contributes to IoT device identification behind a NAT, but focuses instead on the niche problem of fingerprinting Internet-scale NATed malware-infected IoT bots. Further, it explores a novel learning method (i.e., tabular transformers) while reporting on its performance and resiliency against classification decay, with an attempt to interpret the obtained results while also demonstrating its superior results compared to the state-of-the-art machine learning algorithms.

6 CONCLUDING REMARKS

This paper complements current device classification methods behind a NAT by leveraging an attentive interpretable tabular transformer and darknet data to detect Internet-scale malware-infected IoT bots behind NAT. This paper initially builds a ground truth based on Christian and Doerr's methodology [18] and implements TabNet [8] to efficiently classify NATed malware-infected IoT bots.

As a result, we found that our approach outperforms known ML methodologies and was proven resilient against classification decay with an accuracy of 91%. Moreover, we were able to identify 3,879,426 Internet-scale malware-infected IoT bots behind a NAT where 48% of them originated from China.

As for future work, we will continue to explore the notions behind interpretability and attention as applied on network traffic analysis. In addition, with the rise of the ZHtrap [25] botnet, which is a Mirai-based botnet that uses Tor for communications, it would be interesting to examine if a flavor of our developed method could be applied to the Tor network in order to detect compromised IoT bots behind Tor proxies. Finally, we will be attempting to infer malware-infected IoT bots behind a NAT on a real network traffic. As our approach only leverages data from network telescopes, and therefore likely malicious data, it would be interesting to see if our model would work on real network traffic given that benign traffic outweighs malicious data.

ACKNOWLEDGMENTS

This work was supported by a grant from the National Science Foundation, Office of Advanced CyberInfrastructure #1907821.

REFERENCES

[1] 2021. https://blog.netlab.360.com/ten-families-of-malicious-samples-are-spreading-using-the-log4j2-vulnerability-now/
[2] 2021. https://github.com/NATedIoTFingerprinting/NATedIoTFingerprinting
[3] 2021. https://github.com/dreamquark-ai/tabnet
[4] 2021. https://www.maxmind.com/en/geoip2-databases
[5] 2022. https://www.shodan.io/
[6] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*. PMLR, 173–182.
[7] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. 2017. Understanding the mirai botnet. In *26th {USENIX} security symposium ({USENIX} Security 17)*. 1093–1110.
[8] Sercan O Arık and Tomas Pfister. 2020. Tabnet: Attentive interpretable tabular learning. *arXiv* (2020).
[9] Elisa Bertino and Nayeem Islam. 2017. Botnets and internet of things security. *Computer* 50, 2 (2017), 76–79.
[10] Elias Bou-Harb, Mourad Debbabi, and Chadi Assi. 2014. Behavioral analytics for inferring large-scale orchestrated probing events. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 506–511.
[11] Elias Bou-Harb, Mourad Debbabi, and Chadi Assi. 2016. Big data behavioral analytics meet graph theory: on effective botnet takedowns. *IEEE Network* 31, 1 (2016), 18–26.
[12] Amine Boukhtouta, Serguei A Mokhov, Nour-Eddine Lakhdari, Mourad Debbabi, and Joey Paquet. 2016. Network malware classification comparison using DPI and flow packet headers. *Journal of Computer Virology and Hacking Techniques* 12, 2 (2016), 69–100.
[13] Olivier Cabana, Amr M Youssef, Mourad Debbabi, Bernard Lebel, Marthe Kassouf, Ribal Atallah, and Basile L Agba. 2021. Threat Intelligence Generation Using Network Telescope Data for Industrial Control Systems. *IEEE Transactions on Information Forensics and Security* 16 (2021), 3355–3370.
[14] U Cisco. 2020. Cisco annual internet report (2018–2023) white paper.
[15] Emanuele Cozzi, Pierre-Antoine Vervier, Matteo Dell'Amico, Yun Shen, Leyla Bilge, and Davide Balzarotti. 2020. The tangled genealogy of IoT malware. In *Annual Computer Security Applications Conference*. 1–16.
[16] Shayan Eskandari, Andreas Leoutsarakos, Troy Mursch, and Jeremy Clark. 2018. A first look at browser-based cryptojacking. In *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 58–66.
[17] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)* 46, 4 (2014), 1–37.
[18] Harm Griffioen and Christian Doerr. 2020. Quantifying autonomous system IP churn using attack traffic of botnets. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*. 1–10.
[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
[20] Roberto Jordaney, Kumar Sharad, Santanu K Dash, Zhi Wang, Davide Papini, Ilia Nouretdinov, and Lorenzo Cavallaro. 2017. Transcend: Detecting concept drift in malware classification models. In *26th USENIX Security Symposium (USENIX Security 17)*. 625–642.
[21] Ali Safari Khatouni, Lan Zhang, Khurram Aziz, Ibrahim Zincir, and Nur Zincir-Heywood. 2019. Exploring nat detection and host identification using machine learning. In *2019 15th International Conference on Network and Service Management (CNSM)*. IEEE, 1–8.
[22] CU Om Kumar and Ponsy RK Sathia Bhama. 2019. Detecting and confronting flash attacks from IoT botnets. *The Journal of Supercomputing* 75, 12 (2019), 8312–8338.
[23] Rakesh Kumar, Mayank Swarnkar, Gaurav Singal, and Neeraj Kumar. 2021. IoT Network Traffic Classification using Machine Learning Algorithms: An Experimental Analysis. *IEEE Internet of Things Journal* (2021).
[24] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.
[25] Ravie Lakshmanan. 2021. New Mirai Variant and ZHtrap Botnet Malware Emerge in the Wild. https://amp.thehackernews.com/thn/2021/03/new-mirai-variant-and-zhtrap-botnet.html
[26] Yair Meidan, Vinay Sachidananda, Hongyi Peng, Racheli Sagron, Yuval Elovici, and Asaf Shabtai. 2020. A novel approach for detecting vulnerable IoT devices connected behind a home NAT. *Computers & Security* 97 (2020), 101968.
[27] Marius Musch, Christian Wressnegger, Martin Johns, and Konrad Rieck. 2019. Thieves in the Browser: Web-based Cryptojacking in the Wild. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*. 1–10.
[28] Feargus Pendlebury, Fabio Pierazzi, Roberto Jordaney, Johannes Kinder, and Lorenzo Cavallaro. 2019. {TESSERACT}: Eliminating experimental bias in malware classification across space and time. In *28th USENIX Security Symposium (USENIX Security 19)*. 729–746.
[29] Roberto Perdisci, Thomas Papastergiou, Omar Alrawi, and Manos Antonakakis. 2020. IoTFinder: Efficient Large-Scale Identification of IoT Devices via Passive DNS Traffic Analysis. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 474–489.
[30] Morteza Safaei Pour, Antonio Mangino, Kurt Friday, Matthias Rathbun, Elias Bou-Harb, Farkhund Iqbal, Khaled Shaban, and Abdelkarim Erradi. 2019. Data-driven curation, learning and analysis for inferring evolving IoT botnets in the wild. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*. 1–10.
[31] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. 2021. SAINT: Improved Neural Networks for Tabular Data via Row Attention and Contrastive Pre-Training. *arXiv preprint arXiv:2106.01342* (2021).
[32] Zhaoteng Yan, Nan Yu, Hui Wen, Zhi Li, Hongsong Zhu, and Limin Sun. 2020. Detecting Internet-Scale NATs for IoT Devices Based on Tri-Net. In *International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 602–614.
[33] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. *arXiv preprint arXiv:2005.08314* (2020).
[34] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. 2020. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in Neural Information Processing Systems* 33 (2020).