Journal Pre-proof

Autonomous machine learning for early bot detection in the internet of things

Alex Medeiros Araujo, Anderson Bergamini de Neira, Michele Nogueira

PII: S2352-8648(22)00102-X

DOI: https://doi.org/10.1016/j.dcan.2022.05.011

Reference: DCAN 436

To appear in: *Digital Communications and Networks*

Received Date: 13 February 2021

Revised Date: 12 April 2022

Accepted Date: 13 May 2022

Please cite this article as: A.M. Araujo, A. Bergamini de Neira, M. Nogueira, Autonomous machine learning for early bot detection in the internet of things, *Digital Communications and Networks* (2022), doi: https://doi.org/10.1016/j.dcan.2022.05.011.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2022 Chongqing University of Posts and Telecommunications. Production and hosting by Elsevier B.V. on behalf of KeAi Communications Co. Ltd.

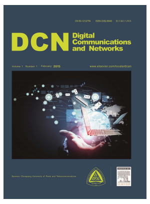
Journal Pre-proof

Digital Communications and Networks(DCN)

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/dcan



Autonomous machine learning for early bot detection in the internet of things

Alex Medeiros Araujo[a], Anderson Bergamini de Neira[a], Michele Nogueira[ab*]

[a]**Department of Informatics**
Federal University of Paraná (UFPR), Curitiba, Brazil
Emails: {amaraujo, abneira}@inf.ufpr.br
[b]**Department of Computer Science**
Federal University of Minas Gerais (UFMG), Belo Horizonte, Brazil
Email: michele@dcc.ufmg.br

Journal Pre-proof

Abstract

The high costs incurred due to attacks and the increasing number of different devices in the Internet of Things (IoT) highlight the necessity of the early detection of botnets (i.e., a network of infected devices) to gain an advantage against attacks. However, early botnet detection is challenging because of continuous malware mutations, the adoption of sophisticated obfuscation techniques, and the massive volume of data. The literature addresses botnet detection by modeling the behavior of malware spread, the classification of malicious traffic, and the analysis of traffic anomalies. This article details ANTE, a system for ANTicipating botnEt signals based on machine learning algorithms. The system adapts itself to different scenarios and detects different types of botnets. It autonomously selects the most appropriate Machine Learning (ML) pipeline for each botnet and improves the classification before an attack effectively begins. The system evaluation follows trace-driven experiments and compares ANTE results to other relevant results from the literature over four representative datasets: ISOT HTTP Botnet, CTU-13, CICDDoS2019, and BoT-IoT. Results show an average detection accuracy of 99.06% and an average bot detection precision of 100%.

KEYWORDS: Network security, Bot early detection, Autonomous machine learning, Network traffic analysis.

1. Introduction

The Internet of Things (IoT) is prominent in academia and industry with multiple applications. IoT devices present resource constraints in bandwidth, processing, memory, and energy, making these devices easy targets for attacks in the face of security vulnerabilities in hardware and software. Attackers exploit security vulnerabilities to build massive and damaging attacks, benefiting from the lack of privacy in sensitive data. Despite these limitations, the number of IoT devices and Machine-to-Machine (M2M) has shown a tendency to grow. In 2019, CISCO reported the existence of 7.4 billion M2M connections, and the forecast is 14.7 billion connections by 2023. The number of connected devices is expected to increase, reaching 29.3 billion by 2023. Moreover, CISCO predicts that 396 exabytes per month will be generated worldwide by 2022 [1]. Such growth tends to improve the destructive potential of botnets when exploited by attackers.

Botnets are one of the main issues in cybersecurity. A bot is a malware-infected device connected to the Internet. A botnet comprises bots that are able to execute commands from botmasters (i.e., devices controlled directly by an attacker) [2, 3]. Using botnets, attackers easily launch different types of attacks. Mainly, botnets serve as a way to (*i*) send bulk electronic correspondence (spam), (*ii*) capture sensitive information, and (*iii*) launch Distributed Denial of Service (DDoS) attacks. In addition to the classic use of botnets, botmasters are always in search of new ways to exploit botnets [4] and new

*Corresponding author:
Michele Nogueira (michele@dcc.ufmg.br).

Journal Pre-proof

types of attacks [5].

Technological diversity supports the creation of botnets with unique characteristics, making their detection a challenging task [6]. Geographical distribution and device mobility make it difficult to neutralize botnets and their effects [3]. Karim et al. [7] claimed that malware could hide malicious code so efficiently that several signature-based detection approaches are unable to detect it. With the advent of new technologies and sophisticated attacks, mainly due to the advent of IoT, botmasters avoid detection techniques. Gupta and Badve [8] pointed out that attackers flood servers with malicious network packets similar to real ones, making it challenging to differentiate malicious flow from legitimate flow.

Chang et al. [9] found that attackers alternate between sending packets and remaining temporarily inactive to make bot identification and botnet detection more difficult. Another severe limitation is that it is nontrivial to select a proper detection technique and its best configuration due to attack diversity, including the unknown (zero-day) attacks. Given the variety of botnets and the difficulty in identifying bots, it is essential to design new solutions for detecting botnets considering the new requirements. New solutions need to anticipate botnet formation to prevent further damage to services caused by attacks and adapt themselves to mitigate each attack [3].

This work details a system for ANTicipating botnEt detection (ANTE), which maps the behavior of bots with ML to create models that anticipate botnet signals. Unlike other works reported in the literature, such as [10], ANTE allows self-adaptation to different scenarios, learning to detect different types of botnets throughout its execution. Therefore, ANTE autonomously selects the most appropriate ML pipeline for each situation to improve the performance of bot classification before an attack begins.

An ML pipeline comprises three parts: (i) data collection and preprocessing, (ii) feature preprocessing, and (iii) estimator. ML algorithms support the identification of botnets, offering adaptation and treatment for massive and high-dimensional data. However, each ML algorithm has its particularities and does not always perform well against every botnet. Thus, the ANTE system decides how to deal with different botnets by autonomously selecting the most appropriate ML technique.

The ANTE system is evaluated on four datasets highly relevant and used in the literature: (i) Information Security and Object Technology (ISOT) Hypertext Transfer Protocol (HTTP) botnet dataset [11], (ii) Scenario 10 (capture number 51) of the Czech Technical University datasets (CTU-13) [12], (iii) DDoS evaluation dataset (CICDDoS2019) [13], and (iv) BoT-IoT dataset [14]. The experiments automatically select the ML pipeline following the ANTE design. The results from the four scenarios show an average accuracy of 99.06% and an average bot detection precision of 100%. These results demonstrate how the ANTE system can effectively choose an algorithm and the feature preprocessing steps to identify bots. Moreover, they show diversity in the proposed solutions, with a different ML pipeline for each scenario, demonstrating that the ANTE system handles different botnets. These results complement the previously conducted analysis [15].

Following are the main contributions of this article: (i) present design and details of a system to identify bots before an attack begins, (ii) application of adaptive strategies to determine the most appropriate ML pipeline for specific network conditions, and (iii) comparison of the ANTE system using three representative frameworks available in the literature. For the tested scenarios, this study shows that it is possible to identify the actions of botnets before an attack begins. The ANTE system chooses the appropriate ML pipeline for a specific scenario. Thus, having a mechanism to select the proper ML pipeline automatically is beneficial. Finally, this study compares three variations of the ANTE system, each based on relevant Autonomous Machine Learning (AutoML) frameworks from the literature, i.e., (i) Auto-Sklearn [16], (ii) H2OAutoML [17], and (iii) AutoGluon [18].

This article proceeds as follows. Section 2 presents the related works. Section 3 details the ANTE system. Section 4 describes the evaluation method and results. Section 5 discusses the results and the limitations of this study. Finally, Section 6 concludes the work and highlights future directions.

2. Related Works

Several studies reported in the literature have investigated the detection of bots and botnets [19, 20]. These studies have addressed this problem at both system and network levels. In brief, the main approaches for detecting botnets are as follows: (i) anomaly-based detection, (ii) traffic signature-based detection, (iii) graphs, (iv) supervised and unsupervised ML. Anomaly-based detection aims to identify misbehavior based on the observed characteristics, such as ports numbers, network latency, or traffic volume. In signature-based detection, solutions already know the botnet behavior and they identify the infected devices using the known behavior. Graph-based approaches based on mathematical models show the relationship between different network entities and then detect bots. Entropy maps the degree of randomness of the network and the interception of communication between malicious devices.

Journal Pre-proof

This section focuses mainly on the ML-based approaches for bot detection. They handle a large amount of data; additionally, they are quick and assertive in automating classification for different scenarios. This article assists in understanding their effectiveness for the early detection of various ML algorithms.

The most common ML-based approaches focus on training models for distinguishing legitimate data flow from malicious data flows [21]. These approaches have been applied to anticipate attacks based on bot activities [22, 23]. Supervised ML trains models because it uses a set of labeled data before suggesting to the model how a new record would be classified. There are several supervised ML algorithms. Literature often uses convolutional neural network and random forest [23, 24, 25]. Another approach groups devices with similar data flow to distinguish legitimate devices from bots [26, 27]. Unsupervised algorithms do not require training before classifying data flow.

In [22], the authors highlighted different stages of botnet behavior and created a model to predict attacks. The main limitation of the model lies in the number of mapped states. The solution does not work correctly if a botnet has a no-mapped form. Unlike [22], for the ANTE system, the infection behavior does not influence the early identification of bots. In [23], the authors focused on identifying botnet Command and Control (C&C) sessions. They analyzed a vector of characteristics using the Random Forest algorithm to identify C&C. However, the proposal does not identify decentralized botnets, such as Peer-to-Peer (P2P) ones.

The Autonomous ML (AutoML) study field aims at democratizing the use of ML. AutoML automates a part of the work with ML by automating the selection of techniques and hyperparameters [28]. Different studies [29, 30] have investigated the problem of selecting the best hyperparameters. Recently, efforts have been made to unify the choice of ML techniques with the best choice of hyperparameters [16, 31]. This problem is called the Combined Algorithm Selection and Hyperparameter (CASH) problem. One of the first frameworks in this area is Auto-Weka (2013) [32]. Subsequently, other approaches were proposed, such as Auto-Sklearn, Auto-Keras [33], H2OAutoML, and AutoGluon.

In addition to the efforts presented in this section, this study contributes to the literature by suggesting a new way of using ML techniques for early bot detection. Instead of using a method that is effective in specific cases, this study takes advantage of the variety of ML techniques to combat the high number of attacks by botnets. Thus, this study presents a system to select the appropriate ML pipeline for the network state. The system adapts itself against different botnets integrating different ML pipelines. Moreover, using robust ML pipelines can identify bots even before the attack starts, reducing the response time against the attack.

3. ANTE System

This section details the system (ANTE) for anticipating botnet signals based on ML algorithms. The ANTE system captures network traffic and searches for bots to notify network administrators before the effects of the botnet are irreversible, such as server exhaustion due to a DDoS attack. After capturing network traffic, the system extracts the network host behavior, and using AutoML, the ANTE system chooses the preprocessing strategy, classifier, and hyperparameters for the classifier. Based on this information, the system creates an ML pipeline, applies preprocessing, and trains the chosen technique to anticipate the beginning of an attack or botnet activities.

The system follows five steps: (i) capture of network traffic, (ii) extraction of host behavior, (iii) identification of the best model (AutoML), (iv) anticipation of bots and (v) administrator notification. Figure 1 shows the positioning of the ANTE system in a network. As illustrated, the system runs in a router or collaboration with a router. The goal is to allow the ANTE system to access a representative amount of traffic from the analyzed network. The router must have a port mirroring function so that the network administrator can configure it to forward a copy of the network traffic to the device that hosts the system. Thus, the ANTE system can analyze the network traffic and extract the features that represent the behavior of each active device.

The ANTE system applies AutoML to identify the best ML pipeline for the current conditions of the network. The ML pipeline comprises data and features preprocessing that optimize data distribution and choose the most appropriate ML model. Defining the ML pipeline, the system analyzes the behavior of the devices on a network and notifies the administrator when a bot is identified. The following subsections describe these steps and their relationship.

3.1. Traffic Capture

In the first step, the ANTE system uses a network monitor to collect network traffic. A physical or a virtual machine implements the network monitor in a real network. In situations where a massive traffic is present, the system can use a fraction of the total amount of captured network traffic instead of processing the entire capture. Such action occurs without a considerable impact on the overall system performance.

Journal Pre-proof

Journal Pre-proof

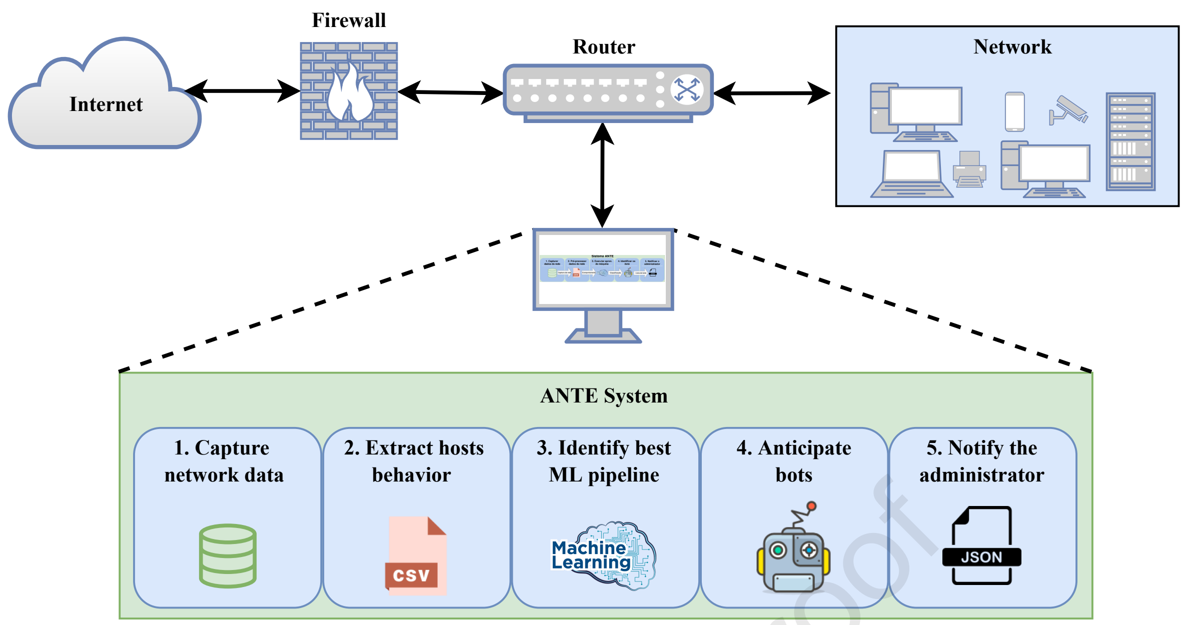


Fig. 1: ANTE Architecture

The ANTE network monitor runs in a streaming mode for real-time data processing and an offline mode using historical data as input. This first step of the ANTE system collects network traffic and processes it following a sliding-window approach. Two parameters are critical when using this approach, window length and sliding interval. The window width defines the length of the analyzed segments. The sliding interval is the arrival frequency for new network packets and the expiration of old packets, which ANTE excludes from the computation. Choosing a small window length reduces computational costs. However, this decreases accuracy because a small window cannot reliably capture the host behavior. The system improves the processing rate by increasing the sliding interval; however, this results in inaccuracy. These issues mean that the system design must focus on the trade-off between computational costs and performance.

3.2. *Behavior Extraction*

Taking the collected network traffic as input, the system groups the network packets in flows enable the extraction of device behavior. Exists different ways to group network packets based on fields from the packets header. Examples include the combination of fields such as (i) packet source Internet Protocol (IP) address; (ii) packet source IP address and MAC address; (iii) source and destination IP addresses and source and destination ports; and (iv) MAC address, source and destination IP address, and source and destination ports. Each combination has its advantages and disadvantages. The basic approach involves grouping packets via source IP address; however, this may be less effective in some attack types. Therefore, the ANTE system is flexible enough to group packets based on different fields. If the ANTE design concerns user privacy, then the system uses only the header fields and never uses the packet payload.

After grouping the network traffic using time windows, the ANTE system extracts a set of relevant features. These features must be expressive enough to represent the behavior of each device in the window. It extracts flow features that are particularly related to outgoing or incoming traffic from a device. Thus, the system seeks to infer the amount of traffic from/to devices and the frequency of sending data, among other relevant features. Moreover, the system extracts several graph-based features, which involves converting the grouped data to an entirely different data structure. Based on graph-based features, the system can infer relationships between devices, which is not possible using only the flow features. Despite a increase in the time taken to extract these features, ANTE improves the representation of the device's behavior. Finally, regardless of the type of feature set, the features must be representative so that ANTE can use them to determine the device behavior.

3.3. *Identifying the Best ML Pipeline*

The third step of the ANTE system is to identify the best ML pipeline. The ML pipeline comprises data preprocessing, feature preprocessing, and estimator. Some preprocessor options improve data distribution by reducing dimensionality, modifying features by joining low-discriminating features, and selecting the most representative features. The estimator is an ML technique chosen and

Journal Pre-proof

configured by the ANTE system to identify abnormal and benign devices and indicate possible bots. The ANTE system uses supervised, unsupervised, and semi-supervised ML techniques, thus providing even more ways to adapt to different scenarios. Irrespective of whether it is possible to label a part or the entire data for training, the system must achieve high accuracy in diverse scenarios and against distinct types of attacks. When the labeling process is costly, the system chooses an unsupervised technique that best adapts to the situations for autonomous bot identification.

This step employs AutoML methods to select the best ML pipeline and analyze data in the training window. For constructing the pipeline, we introduce the AutoML engine component. The AutoML engine selects and configures ML techniques from an initial dataset. In our study, we design and evaluate a collection of different engines. Each engine has its particularity; however, they generally have a set of preconfigured ML techniques optimized for the best use of the training data. Designing a system to select an ML technique among the available ones is the minimum requirement to consider a system as an AutoML engine. Some engines include methods for data preprocessing; however, this is not a requirement. In such cases, the ANTE ML pipeline would also work without techniques for data preprocessing. However, the preprocessing techniques usually improve the results.

3.4. Anticipating Bot Detection

Once the ML pipeline is defined, the ANTE system analyzing the grouped network traffic per window. As the system collects new data, it classifies the detected devices. The system notifies the network administrators of any indication that a device may be a bot. Thus, the bot does not need to initiate the attack or cause unrecoverable damage to the network before being detected. Hence, the system anticipates actions from bots and predicts attacks, even if they have not yet started.

3.5. Notify the Administrator

Through the output of the ML pipeline, the ANTE system classifies each device on the network into two lists. The first list comprises the devices classified as benign, and the second list includes bots. If the list of bots has records, the system takes an action to avoid denying the available services. Exists different options for these actions, such as limiting or blocking access to the device, which can be performed manually or automatically at the firewall. In this study, the envisaged action is to communicate with the network administrators through an email providing a list of IP addresses of possible bots. An option for automating incident response is to send a JavaScript Object Notation (JSON) message containing the occurrence data, such as the list of bots and the probability of the device being a bot. It serves as an entry to automate a firewall or an alert system.

4. Performance Evaluation

This section details the performance evaluation of the ANTE system, which involves the following four steps: (i) scenario selection, (ii) behavior extraction, (iii) AutoML execution, and (iv) host classification using AutoML pipeline. Figure 2 illustrates the ANTE evaluation methodology, and each step is detailed subsequently. All experiments were run on a computer with an i5 processor, 1-TB hard disk drive, and 8-GB random-access memory.

4.1. Scenario Selection

Evaluations employ four datasets as input that are highly used by studies reported in the literature containing captures from data flows of infected (bots) and noninfected (not bots) devices. The four datasets are the (i) ISOT HTTP botnet dataset, (ii) the capture number 51 of the Czech Technical University datasets (CTU-13), (iii) DDoS evaluation dataset (CICDDoS2019), and (iv) BoT-IoT dataset (see Section1). These datasets have malicious and legitimate network traffic. They comprise traffic from attacks of different botnets. Finally, a key characteristic of these datasets is their easy-to-follow documentation regarding labeling bots and the precise moment of the beginning of an attack.

4.2. Behavior Extraction

As shown in Figure 2, in the experiments, the ANTE system extracts the behavior of the network (item 2) from the datasets (item 1). Extraction is necessary for defining the analysis center, i.e., when the attack begins. The dataset documentation provides details regarding the actions of the botnets in the network traffic captures, indicating the time at which the attack begins. After defining this analysis center, in the experiments, the ANTE system sets the training, anticipation, and testing windows (illustrated in Figure 2, item 2).

The ANTE system extracts 3 min of capture immediately before the identified actions of the botnets (analysis center) begin. Then, it splits these 3 min of captures into two parts. The first part has 2 min 30 s of dataset capture, and it is called the training window. The second part comprises 30 s of dataset capture, and it is called the anticipation window. After the analysis center, the system extracts an additional 2 min of capture for the identification tests during the attack. The system defines the anticipation and test windows to analyze different stages of the botnet life cycle. These intervals test the system with the

Journal Pre-proof

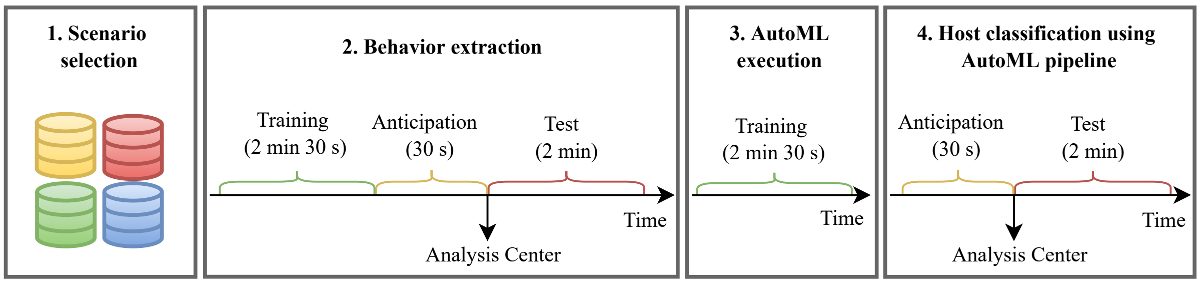


Fig. 2: The ANTE System Evaluation Steps

windows susceptible to reproduction in online environments. Furthermore, they allow us to explore different moments of an attack without necessarily using attack data in the ANTE training. However, other intervals can be investigated and analyzed in the future.

The experiments generate Comma-Separated Value (CSV) extension files for all time windows. These CSVs contain the data related to devices and their packet exchange on the network representing the behavior of the devices grouped via IP address. An alternative is to use a combination of IP and Medium Access Control (MAC) addresses. This behavior includes a combination of 18 flow features [23] and 4 graph features [34], presented in Table 1. Therefore, each CSV line represents a different device using the IP and label attributes. The ANTE system extracts all the features present in the CSVs (Table 1) from all datasets. Thus, the system does not favor features and ensures that it can use all features for all analyses. The experiments provide the malicious device identifier (label) to the system, known from the documentation of the respective databases to ensure the training of the supervised learning. Furthermore, the system employs device IP and label to only characterize the behavior and does not use them as a flow feature.

Table 1: Employed Features

Features	Description
Tsr/Tss	Sum of received/sent packets size (bytes)
Rpc/Spc	Amount of packets received/sent
Savg	Average sent packet size
Smin	Minimum sent packet size (bytes)
Smax	Maximum sent packet size (bytes)
Svar	Variance of sent packets (bytes)
Ravg	Average received packet size
Rmin	Minimum received packet size
Rmax	Maximum received packet size (bytes)
Rvar	Variance of received packets (bytes)
SITmin	Minimum interval time in sent packages
SITmax	Maximum interval time in sent packages
SITavg	Average interval time of sent packages
RITmin	Minimum interval time in received packages
RITmax	Maximum interval time in received packages
RITavg	Average interval time of received packages
OutDgr	Number of edges pointing out of the node
InDgr	Number of edges pointing in to the node
BC	Number of shortest paths passing through a node
EG	Centrality degree of a node

4.3. AutoML Execution

The experiments employ three instances of the ANTE system. Each instance employs a specific AutoML framework. The three frameworks are Auto-Sklearn, H2OAutoML, and AutoGluon. The subsequent paragraphs describe the particularities of each framework.

Auto-Sklearn [16] is an AutoML framework based on Scikit-learn, which comprises of 14 preprocessing methods, four feature preprocessing techniques, and 15 classification algorithms. Two key principles characterize Auto-Sklearn. First, the use of meta-learning to start the optimization process. Auto-Sklearn compares every new dataset against an internal database of past executions. It first explores the solutions that performed well on similar datasets. Second, it does not discard the intermediary models found during the optimization process. Auto-Sklearn automatically builds an ensemble using all the best classifiers found during the optimization process.

H2O is an ML and predictive analytics platform written in Java, with bindings for R (H2O-R), Python (H2O-Python), and Scala. The H2O AutoML framework [17] uses a combination of fast random search and stacked ensembles. The stacked ensembles include diverse base models as Gradient Boosting Machines (GBMs), Generalized Linear Models (GLMs), Random Forests, and Deep Neural Nets. After training the base models, H2O stacked ensemble algorithm builds the final ensemble by determining the optimal combination of the base learners. Currently, the H2OAutoML solution does not offer any preprocessing or feature preprocessing methods applicable to our work. Thus, the ML pipeline only comprises an estimator element.

Distinct from the other frameworks, AutoGluon [18] has specific modules for performing different tasks. The modules include data analysis using tables, image classification, object detection, and text prediction. We support the data analysis using tables. AutoGluon always applies the same preprocessing methods; hence, the preprocessing component is the same for every ML pipeline. AutoGluon works by testing algorithms in a predefined order. The ordering prioritizes reliable models, i.e., more expensive and less reliable models are

Journal Pre-proof

Journal Pre-proof

the last models to be explored. AutoGluon introduces a novel multilayer stack grouping to build an ensemble using the models trained during the optimization process.

The AutoML engine operates only with the data from the training window (shown in Figure 2, item 3). For defining the training data, ANTE system uses a sliding window approach where the system extracts the features at evenly spaced fixed-size windows over the training segment. The sliding window size is 30 s, and the start of every window is offset by 500 ms. This procedure is a variant of the walk forward optimization technique, which aims to avoid overfitting using an out-of-sample testing approach. The AutoML engine selects only the best ML model under a time constraint of 5 min. Finally, to further avoid overfitting and benefit models that can classify samples from bots and legitimate devices correctly, F1-Score is used to enable Auto-Sklearn and AutoGluon prioritize equitable models. H2OAutoML does not have the option to choose F1-Score as a metric to evaluate the best ML model.

The computational complexity of the ANTE system is directly related to that of the feature extraction algorithms and ML pipeline. This study uses 22 features to identify possible bots. In real scenarios, the extraction of so many features would be harrowing in real-time owind to the high consumption of computational resources. Removing the less-used features can reduce the load in this problem. However, feature selection is a nontrivial problem still addressed in the literature [35, 36]. To identify the best ML pipeline, Auto-Sklearn uses 5 min of training. Although Auto-Sklearn uses 3 data preprocessing techniques, 5 min of processing can negatively impact the ANTE accuracy.

4.4. Host Classification using AutoML Pipeline

ANTE defines the ML pipeline for each scenario after the analysis of the training data is complete. The ANTE ML pipeline involves three strategies to improve training data and a strategy for classifying devices. If the AutoML engine is unable to provide all strategies of the ANTE ML pipeline, this strategies will be unspecified. The first strategy solves the missing data problem, and it is called the imputation strategy. Missing data are the information of certain features lost by some devices because of different reasons, such as transmission issues and failure in collecting data. In this evaluation, all the analyzed devices always have the attributes properly estimated. Therefore, there are no missing data in the dataset. However, ANTE proposes an imputation strategy for dealing with missing data, if data are missed while using the system. The second strategy for improving training data is rescaling. Some ML algorithms show better results if the data are represented in certain scales, such as representing the values of the attributes in the range between zero and one. Thus, ANTE selects a rescaling strategy for all scenarios. The third strategy for improving training data is preprocessing features. Attribute selection seeks to remove low-discriminating attributes and improve the overall functioning of ANTE. Finally, the third strategy in the pipeline is the estimator.

For evaluations shown in Figure 2, item 4, the ANTE system uses the preprocessing technique, ML model and hyperparameters selected by the AutoML engine to classify hosts before an attack effectively begins (anticipation window) and throughout the attack (test window). For creating the test data, ANTE uses a sliding window size of 30 s and offsets the start of every window by 30 s. To measure the quality of the ANTE decisions, results compare the classification with the original labels using metrics such as accuracy, precision, recall, and the F1-Score, as detailed in the following text. Additionally, metrics evaluate the diversity in the choices of the preprocessors, ML models, and hyperparameters.

To simplify results presentation, we have defined acronyms for ANTE executions. Thus, the term ANTE-AS refers to the execution of ANTE using the Auto-Sklearn engine. The term ANTE-H2 refers to the execution of ANTE using the H2OAutoML engine. Finally, ANTE-AG refers to the execution of ANTE using the AutoGluon engine. Additionally, complementary evaluations include three Scikit-learn models created using the default parameters, Support Vector Machines (SVM), Naïve Bayes (NB), and Multilayer Perceptron (MLP). This comparison analyzes the importance of the correct choice and configuration of ML techniques and to examine whether other techniques would be assertive. Finally, evaluation scenarios also compare ANTE results to the results presented in [37].

The first metric employed is accuracy (Eq. 1), which is the ratio of the total number of correct classifications to the total number of classified items. In cases where the distribution of classes is unbalanced, this metric may be misinterpreted. If the classifier is misled by the majority of classes, it can make mistakes for all projections to the minority class. However, it still has an accuracy of >90%. Precision (Eq. 2), recall (Eq. 3), and F1-Score (Eq. 4) are the metrics based on the number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN). It is possible to obtain these metrics for the class of nonbots and bots. Thus, for this study, the bot collection is the positive class and presents the precision, recall, and F1-Score for the bot class. All information is available in the project repository[1].

[1]https://github.com/ccsc-cssg/dcn-neira-medeiros-nogueira

Journal Pre-proof

Journal Pre-proof

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \tag{1}$$

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$F1-Score = 2.\frac{precision.recall}{precision+recall} \tag{4}$$

4.5. ISOT Scenario

The ISOT scenario is a combination of several malicious and nonmalicious datasets maintained by the University of Victoria, Canada[2]. The scenario used in our evaluation comprises malicious and benign Domain Name System (DNS) traffic, totaling 780 MB of network traffic data generated by 35 devices, 9 of them are infected. The analysis center was defined as 21:39:31 on May 30, 2017, Coordinated Universal Time (UTC), because the attack aimed to steal the information theft. ANTE trains in the presence of eight bots among the 35 devices that exchanged packets before the attack.

Table 2 illustrates the entire configuration of the ML pipeline chosen by ANTE-AS after analyzing the training data for the ISOT scenario. Further information regarding the configurations is publicly available in the repository[3]. ANTE-AS chooses the mean as the most appropriate imputation strategy for this scenario. Thus, if the feature extraction component cannot calculate any attribute for any device during the system execution, ANTE-AS completes the attribute value using the mean of the same attribute of the other devices. Moreover, ANTE-AS has identified that changing the data scale by applying attribute normalization would be beneficial for the device classification. Thus, ANTE-AS scales the data based on the quantile range. The last strategy to improve training data is to apply the Select Percentile approach. Select Percentile is an approach for selecting features based on univariate statistical tests. Select Percentile removes less discriminating features using false discovery rate, false positive rate, or family wise error[4]. Finally, ANTE-AS selects the AdaBoost estimator as the most appropriate ML technique for the scenario.

As mentioned in Subsection 4.3, the ANTE-H2 and ANTE-AG ML pipelines vary only for the estimator between the scenarios. Table 2 shows the ML pipelines defined during the execution of ANTE-H2 and ANTE-AG. ANTE-H2 has selected and configured an estimator different from ANTE-AS. In this case, the chosen algorithm was the GBM. The third instance, ANTE-AG, has selected a different estimator than the other two instances; the one chosen here was the LightGBMClassifier.

Table 3 shows the result of applying the entire ML pipeline selected by ANTE during the three executions. Using the ANTE-AS and ANTE-H2 pipelines, we obtained the correct classification for all devices in the step before the attack. During the test window, ANTE-H2 misclassified the bot twice by marking it as a regular device. ANTE-AS misclassified two classifications. However, one of the occurrences classified a normal device as a bot and a bot as a normal device. Because ANTE-AG fails to classify some bot instances, the system exhibits slightly lower performance than previous ANTE executions. ANTE-AG managed to obtain expressive results, reaching an accuracy of 96% before and 97.69% after the attack started. Further, Table 3 shows the difficulty faced by SVM, NB, and MLP to correctly classify devices because they were not as effective as the ANTE approach.

4.6. CTU-13 Scenario

The CTU-13 scenario uses the capture-51 belonging to a set of 13 captures, called CTU-13 [12]. This set of 13 captures of botnet traffic was recorded at the Czech Technical University and made available through the Stratosphere Project[5]. This dataset has a size of approximately 66 GB and reports two attacks. In this scenario, we analyze the second attack that began on August 18, 2011, at 14:43:27, Central European Summer Time. This scenario contains an ICMP flood attack using 10 bots. During the training, ANTE identified 30 devices of which 10 were bots.

Table 2 presents the configuration of the ML pipeline chosen by ANTE-AS. Similar to the previous scenario, ANTE-AS chooses the mean as the imputation strategy. Unlike the previous scenario the Standardize approach is used as the rescaling strategy. The Standardize strategy normalizes features by applying a process where ANTE-AS calculates the average and standard deviation for each feature across the training base. For each new observation (new device to be classified), ANTE-AS removes the average value and divides it by the previously calculated standard deviation for each feature. Especially for this scenario, ANTE-AS is unable to select an approach for selecting features. All pipeline configurations are available in the repository[3]. ANTE-AS chooses an estimator different from the previous

[2]https://www.uvic.ca/engineering/ece/isot/datasets/botnet-ransomware Accessed in: 08/2020

[3]https://github.com/ccsc-cssg/dcn-neira-medeiros-nogueira

[4]scikit-learn.org/stable/modules/feature_selection.html#univariate-feature-selection Accessed in: 02/2021

[5]www.stratosphereips.org Accessed in: 08/2020

Journal Pre-proof

Journal Pre-proof

Table 2: The Best Machine Learning Pipeline per Scenario Identified by ANTE

	ANTE-AS				**ANTE-H2**				**ANTE-AG**			
	Data Preprocessor		**Features Preprocessor**	**Estimator**	**Data Preprocessor**		**Features Preprocessor**	**Estimator**	**Data Preprocessor**		**Features Preprocessor**	**Estimator**
ISOT Pipeline	Imputation: Mean	Rescaling: RobustScaler	Select Percentile	AdaBoost	Imputation: Not Applicable	Rescaling: Not Applicable	Not Applicable	GBM	Imputation: Median	Rescaling: Z-score	Not Applicable	LightGBMClassifier
CTU-13 Pipeline	Imputation: Mean	Rescaling: Standardize	None	Random Forest	Imputation: Not Applicable	Rescaling: Not Applicable	Not Applicable	DRF	Imputation: Median	Rescaling: Z-score	Not Applicable	LightGBMClassifierXT
CICDDoS Pipeline	Imputation: Most Frequent	Rescaling: Min-Max	Polynomial Features	Random Forest	Imputation: Not Applicable	Rescaling: Not Applicable	Not Applicable	GBM	Imputation: Median	Rescaling: Z-score	Not Applicable	LightGBMClassifierXT
BoT-IoT Pipeline	Imputation: Mean	Rescaling: Min-Max	FastICA	SVM	Imputation: Not Applicable	Rescaling: Not Applicable	Not Applicable	DRF	Imputation: Median	Rescaling: Z-score	Not Applicable	LightGBMClassifier

Table 3: Results for the ISOT Scenario

Approach	**Window**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
ANTE-AS	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	98.46%	95.83%	95.83%	95.83%
ANTE-H2	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	98.46%	100.0%	91.67%	95.65%
ANTE-AG	Anticipation	96.0%	100.0%	85.71%	92.31%
	Test	97.69%	100.0%	87.5%	93.33%
SVM	Anticipation	88.0%	100.0%	57.14%	72.73%
	Test	90.77%	100.0%	50.0%	66.67%
Naïve Bayes	Anticipation	72.0%	50.0%	100.0%	66.67%
	Test	73.85%	40.0%	83.33%	54.05%
MLP	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	95.38%	90.91%	83.33%	86.96%

scenario. In this case, ANTE-AS identifies the Random Forest estimator as the best option.

Unlike the ISOT scenario, in which ANTE-AS and ANTE-H2 chose different estimators, for the CTU-13 scenario, ANTE-H2 completed its ML pipeline using an estimator similar to that of ANTE-AS. This time, ANTE-H2 selected the Distributed Random Forest (DRF) algorithm, which is the same as the Random Forest selected by ANTE-AS. Meanwhile, ANTE-AG selected LightGBMClassifierXT as an estimator, which was not previously not used.

We apply all the three ML pipelines in the CTU-13 data to classify the devices present in the anticipation and test windows. Table 4 shows that the ANTE-AS, ANTE-H2, and ANTE-AG ML pipelines were able to classify all devices correctly in the two windows. A similar characteristic to the previous scenario was the difficulty faced by the estimators SVM, NB, and MLP to correctly classify the hosts.

Table 4: Results for the CTU-13 Scenario

Approach	**Window**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
ANTE-AS	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	100.0%	100.0%	100.0%	100.0%
ANTE-H2	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	100.0%	100.0%	100.0%	100.0%
ANTE-AG	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	100.0%	100.0%	100.0%	100.0%
SVM	Anticipation	92.31%	50.0%	100.0%	66.67%
	Test	73.91%	68.97%	100.0%	81.63%
Naïve Bayes	Anticipation	92.31%	0.0%	0.0%	0.0%
	Test	100.0%	100.0%	100.0%	100.0%
MLP	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	97.1%	95.24%	100.0%	97.56%

4.7. CICDDoS Scenario

The CICDDoS scenario uses the CICDDoS2019 dataset [38], made available by the University of New Brunswick (UNB)[6]. We chose this dataset because it was published in 2019, has 20 GB of network traffic, and has different types of attacks collected in two days [38]. Of the 12 attacks reported in this collection, this scenario analyzes the Network Basic Input Output System (NetBIOS) attack conducted on the first day of the experiment. According to the documentation, this attack occurred between 10:21 and 10:30 UTC on the first day. However, by analyzing the available data, it is impossible to find network flow at this time. Thus, the tests have investigated the peak traffic that occurred on December 1, 2018, at 15:06:10 UTC as the analysis center. Examining the data of this scenario, a peak of 110 devices was obtained; however, it has only one active bot.

The analysis of the training data by ANTE-AS shows that the selected ML pipeline has different approaches when compared with the previous scenarios. Table 2 indicates the choices made by ANTE-AS to compose the pipeline. The full set of settings is available in the repository[3]. The Most Frequent data processor is chosen for this scenario. If ANTE-AS identifies missing values, it completes them using the most common value of the feature. The Min–Max strategy used for rescaling is different from the three scenarios. In practice, this Min–Max strategy scales the values of the features between the lowest and highest values found for the feature. ANTE-AS chose a feature preprocessing technique different from the previous scenarios. Polynomial features reduce the dimensionality of features (number of features) by combining the less discriminating features. Finally, ANTE-AS chose Random Forest as the estimator, which is the same as in previous scenarios.

To complete their ML pipelines, ANTE-H2 and ANTE-AG did not introduce new estimators. Conversely, they selected the estimators already mentioned in previous ML pipelines. ANTE-H2 selected GBM, similar to in the first ML pipeline. ANTE-AG selected the same estimator used in the previous scenario, the LightGBMClassifierXT. While in the CTU-13 scenario, ANTE-AS and ANTE-H2 selected Random Forest; in the CICDDoS scenario, only ANTE-AS again selected the Random Forest estimator.

Table 5 shows the results obtained using the pipeline defined by ANTE. It is possible to verify if

[6]www.unb.ca/cic/datasets/ddos-2019.html Accessed in: 08/2020

Journal Pre-proof

all the proposed ML pipelines correctly classify all the hosts for the anticipation and test windows. As observed in other scenarios, the SVM, NB, and MLP estimators were not successful in classifying the devices found in this scenario.

Table 5: Results for the CICDDoS Scenario

Approach	Window	Accuracy	Precision	Recall	F1-Score
ANTE-AS	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	100.0%	100.0%	100.0%	100.0%
ANTE-H2	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	100.0%	100.0%	100.0%	100.0%
ANTE-AG	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	100.0%	100.0%	100.0%	100.0%
SVM	Anticipation	97.87%	0.0%	0.0%	0.0%
	Test	98.75%	0.0%	0.0%	0.0%
Naïve Bayes	Anticipation	97.87%	50.0%	100.0%	66.67%
	Test	100.0%	100.0%	100.0%	100.0%
MLP	Anticipation	93.62%	25.0%	100.0%	40.0%
	Test	99.07%	57.14%	100.0%	72.73%

4.8. *BoT-IoT Scenario*

The Cyber Range Lab of the center of UNSW Canberra Cyber created the BoT-IoT[7] dataset, the fourth scenario analyzed in this work. The dataset has 72,000,000, records and more than 69 GB of data was made available via pcap files. Traffic aggregates data from benign devices and bots. The dataset reports various types of attacks, such as DoS, DDoS, and OS Scan. We chose to analyze the DDoS attack by four bots that happened on June 4, 2018, at 06:03:34 UTC.

Table 2 shows that ANTE-AS has selected a different ML pipeline for the BoT–IoT scenario. ANTE-AS had already selected mean for imputation and Min–Max for rescaling, but they were not used together in the same ML pipeline. FastICA is a preprocessor feature technique that is different from the others because FastICA seeks to identify the independent features to verify those that best represent the dataset. SVM is the estimator chosen to complete the ANTE-AS ML pipeline. SVM is one of the most well-known and used ML techniques reported in the literature, i.e., to an extent that we use Scikit-Learn [39] standard SVM as a baseline. Both ANTE-H2 and ANTE-AG employed previously used estimators. ANTE-H2 selected the DFR, an estimator originally selected in the CTU-13 scenario. ANTE-AG selected the LightGBMClassifier, which was also selected in the ISOT scenario.

ANTE-AS results in an ML pipeline that can identity bots in the anticipation window before the attack. However, after the attack begins, the model's efficiency decreases considerably. Despite this, using the processing techniques and the configuration of the hyperparameters configured by ANTE-AS, results are considerably improved for the two windows when compared with SVM with the standard configurations and without the data preprocessing phase (row five of Table 6). ANTE-H2 was the ANTE instance that best identified an ML pipeline for this scenario, reaching an accuracy of 100% before the attack and 96.69% after the attach began. Similar to ANTE-AS, ANTE-AG has managed to classify all devices before the attack began; however, soon its efficiency decreased dramatically. CICDDoS was the scenario in which SVM, NB, and MLP had more difficulty in correctly identifying the devices on the network, reaching a minimum accuracy of 31.72% for one case.

[7]https://ieee-dataport.org/documents/bot-iot-dataset Accessed in: Aug 2020

Table 6: Results for the BoT-IoT Scenario

Approach	Window	Accuracy	Precision	Recall	F1-Score
ANTE-AS	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	81.27%	56.77%	96.77%	71.56%
ANTE-H2	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	96.69%	88.02%	100.0%	93.63%
ANTE-AG	Anticipation	100.0%	100.0%	100.0%	100.0%
	Test	79.2%	53.92%	100.0%	70.06%
SVM	Anticipation	77.13%	54.4%	100.0%	70.46%
	Test	31.72%	26.28%	100.0%	41.62%
Naïve Bayes	Anticipation	64.05%	41.98%	83.33%	55.84%
	Test	62.63%	38.14%	86.04%	52.85%
MLP	Anticipation	98.48%	100.0%	94.44%	97.14%
	Test	70.55%	43.82%	74.39%	55.15%

4.9. *Comparison with the Literature*

For comparison we have reproduced the results from [37] using the CTU-13 dataset. Moreover, we add the analysis of the ISOT HTTP and CICDDoS2019 datasets. The first comparison uses the obtained results by the technique proposed in [37] on the ISOT HTTP botnet dataset. For this dataset, Pektaş and Acarman's solution got all the classifications right as well as the ANTE-AS and ANTE-H2 system (anticipation window). For the CTU-13 dataset, all the ANTE approaches are superior to the Pektaş and Acarman's one. ANTE-AS and ANTE-H2 can classify all hosts correctly, while Pektaş and Acarman's approach reached an accuracy of 99.3%. The last comparison is to the CICDDOS2019 dataset. Pektaş and Acarman's approach obtains an accuracy of 96.87%. The issue with this result is that throughout the tests, the proposed method fails to identify the bots. All the ANTE instances classify all the devices correctly. Table 7 provides all the data referring to the comparison.

5. Discussion

While a score of 100% in any evaluation metric can generally be a sign of overfitting, in our work, this is a consequence of the highly imbalanced nature of botnet datasets. For most of the datasets, majority of devices belong to the nonbot class. In some datasets, there is only one instance of the bot class during testing, resulting in a precision of 0% or 100%. ANTE-AS is a complete AutoML engine in

Journal Pre-proof

Journal Pre-proof

Table 7: ANTE vs. Result from [37]

Dataset	System	Accuracy	Precision	Recall	F1
ISOT	ANTE-AS	100%	100%	100%	100%
	ANTE-H2	100%	100%	100%	100%
	ANTE-AG	96%	100%	85.7%	92.3%
	Ref. [37]	100%	100%	100%	100%
CTU-13	ANTE-AS	100%	100%	100%	100%
	ANTE-H2	100%	100%	100%	100%
	ANTE-AG	100%	100%	100%	100%
	Ref. [37]	99.3%	99.1%	99.2%	99.1%
CICDDoS	ANTE-AS	100%	100%	100%	100%
	ANTE-H2	100%	100%	100%	100%
	ANTE-AG	100%	100%	100%	100%
	Ref. [37]	96.8%	0%	0%	0%

terms of data preprocessing techniques. The ANTE-AS engine autonomously determine the best combination of preprocessing, feature selection, and classification methods. In contrast, the other engines found the best classification model and used the same data preprocessing steps on every pipeline. Furthermore, ANTE-AS selected the widest variety of estimators with three unique choices being AdaBoost, Random Forest (twice), and SVM. Finally, with the results obtained for the BoT–IoT scenario, the preprocessing combined with the correct configuration of the estimator makes a difference in identifying malicious devices. Despite this evidence, future works can further analyze such statements.

The results show the efficiency of the ML pipelines selected by ANTE. In the ISOT scenario, only ANTE-AG proposed an ML pipeline that unable to classify all the devices correctly before the attack. In the BoT–IoT scenario, the efficiency of the ML pipelines selected by ANTE ranged from 79.2% to 96.69%. Thus, for the analyzed scenarios, ANTE-H2 was the best adapted to cybersecurity, mainly for bot identification.

The ANTE system analyzes the behavior of devices in a network, including their interactions. Thus, the proposed system may not work if it cannot capture the traffic between devices. For proper operation, the system needs to determine the device behavior for features, including those based on graphs. Future works will focus on the feasibility of the proposed system in cases where it is impossible to observe the interactions between the network devices.

6. Conclusion

Given the destructive potential of bots and botnets intensified by the advent of the IoT and mobile devices, detecting bots and botnets as early as possible is necessary to combat the diversity of cyberattacks. This article detailed the ANTE system, which aims at early detection of bots using AutoML. The proposed system effectively classified the devices of the evaluated botnets. For the system to achieve this result, AutoML selects the best ML pipeline for each scenario. This indicates that ANTE considers the particularities of each situation, including different types of attacks and botnets, while selecting the most suitable techniques for each scenario. Thus, ANTE has highlighted the importance of the correct choice for ML techniques to ensure high precision in botnet identification. Another important finding is that bots are generally present and start exchanging packets before the effective initiation of the attacks. Even before the effective beginning of an attack, ANTE was able to identify bots. Future works include the reduction in the dependence on labels and they involve improving the processing time of ANTE to make the solution feasible online.

Acknowledgment

This work was supported by National Council for Scientific and Technological Development (CNPq/Brazil), grants #309129/2017-6 and #432204/2018-0, by São Paulo Research Foundation (FAPESP), grant #2018/23098-0, by the Coordination for the Improvement of Higher Education Personnel (CAPES/Brazil), grants #88887.501287/2020-00 and #88887.509309/2020-00, and by the National Teaching and Research Network (RNP) by the GT-Periscope project.

References

[1] T. Barnett, S. Jain, U. Andra, T. Khurana, Cisco visual networking index (vni), complete forecast update, 2017–2022, https://www.cisco.com/c/dam/m/en_us/network-intelligence/service-provider/digital-transformation/knowledge-network-webinars/pdfs/1211_BUSINESS_SERVICES_CKN_PDF.pdf (accessed 23 March 2020).

[2] Y. D. Mane, Detect and deactivate p2p zeus bot, in: Proceedings of the 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), IEEE, 2017, pp. 1–7.

[3] A. A. Santos, M. Nogueira, J. M. F. Moura, A stochastic adaptive model to explore mobile botnet dynamics, IEEE Commun. Lett. 21 (4) (2017) 753–756.

[4] S. Venkatesan, M. Albanese, A. Shah, R. Ganesan, S. Jajodia, Detecting stealthy botnets in a resource-constrained environment using reinforcement learning, in: Proceedings of the 2017 Workshop on Moving Target Defense, ACM, 2017, p. 75–85.

[5] D. Y. Huang, D. McCoy, H. Dharmdasani, S. Meiklejohn, V. Dave, C. Grier, S. Savage, N. Weaver, A. C. Snoeren, K. Levchenko, Botcoin: Monetizing stolen cycles, in: Proceedings of the 2014 Network and Distributed System Security Symposium, Internet Society, 2014, pp. 1–16.

[6] S. Ruano Rincón, S. Vaton, A. Beugnard, S. Garlatti, Semantics based analysis of botnet activity from heterogeneous data sources, in: Proceedings of the 2015 International Wireless Communications and Mobile Computing Conference (IWCMC), IEEE, 2015, pp. 391–396.

[7] A. Karim, R. B. Salleh, M. Shiraz, S. A. A. Shah, I. Awan, N. B. Anuar, Botnet detection techniques: review, future trends, and issues, Journal of Zhejiang University SCIENCE C 15 (11) (2014) 943–983.

[8] B. Gupta, O. P. Badve, Taxonomy of DoS and DDoS attacks and desirable defense mechanism in a cloud computing environment, Neural Computing and Applications 28 (12) (2017) 3655–3682.

Journal Pre-proof

Journal Pre-proof

[9] W. Chang, A. Mohaisen, A. Wang, S. Chen, Understanding adversarial strategies from bot recruitment to scheduling, in: Proceedings of the 2018 Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Springer, Cham, 2018, pp. 397–417.

[10] B. M. Rahal, A. Santos, M. Nogueira, A distributed architecture for ddos prediction and bot detection, IEEE Access 8 (-) (2020) 159756–159772.

[11] A. Alenazi, I. Traore, K. Ganame, I. Woungang, Holistic model for http botnet detection based on dns traffic analysis, in: Proceedings of the International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments, Springer, 2017, pp. 1–18.

[12] S. García, M. Grill, J. Stiborek, A. Zunino, An empirical comparison of botnet detection methods, Computers & Security 45 (-) (2014) 100 – 123.

[13] I. Sharafaldin, A. H. Lashkari, S. Hakak, A. A. Ghorbani, Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy, in: Proceedings of the 2019 International Carnahan Conference on Security Technology (ICCST), IEEE, 2019, pp. 1–8.

[14] N. Moustafa, The bot-iot dataset, https://ieee-dataport.org/documents/bot-iot-dataset (accessed 23 March 2020).

[15] A. B. de Neira, A. M. Araujo, M. Nogueira, Early botnet detection for the internet and the internet of things by autonomous machine learning, in: Proceedings of the 2020 16th International Conference on Mobility, Sensing and Networking (MSN), IEEE, 2020, pp. 516–523.

[16] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, F. Hutter, Efficient and robust automated machine learning, in: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, MIT Press, 2015, p. 2755–2763.

[17] E. LeDell, S. Poirier, H2o automl: Scalable automatic machine learning, in: Proceedings of the 7th ICML Workshop on Automated Machine Learning (AutoML), PRML, 2020, pp. 1–16.

[18] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, A. Smola, Autogluon-tabular: Robust and accurate automl for structured data, arXiv preprint - (-) (2020) 1–28.

[19] Y. Xing, H. Shu, H. Zhao, D. Li, L. Guo, Survey on botnet detection techniques: Classification, methods, and evaluation, Mathematical Problems in Engineering 2021 (-) (2021) 1–24.

[20] K. Shinan, K. Alsubhi, A. Alzahrani, M. U. Ashraf, Machine learning-based botnet detection in software-defined network: A systematic review, Symmetry 13 (5) (2021) 866.

[21] S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, P. Hakimian, Detecting P2P botnets through network behavior analysis and machine learning, in: Proceedings of the 2011 Ninth Annual International Conference on Privacy, Security and Trust, IEEE, 2011, pp. 174–180.

[22] Z. Abaid, D. Sarkar, M. A. Kaafar, S. Jha, The early bird gets the botnet: A markov chain based early warning system for botnet attacks, in: Proceedings of the 2016 IEEE 41st Conference on Local Computer Networks (LCN), IEEE, 2016, pp. 61–68.

[23] L. Lu, Y. Feng, K. Sakurai, C&C session detection using random forest, in: Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication, ACM, 2017, pp. 34:1–34:6.

[24] A. Bansal, S. Mahapatra, A comparative analysis of machine learning techniques for botnet detection, in: Proceedings of the 10th International Conference on Security of Information and Networks, ACM, 2017, pp. 91–98.

[25] S. Chen, Y. Chen, W. Tzeng, Effective botnet detection through neural networks on convolutional features, in: Proceedings of the 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), IEEE, 2018, pp. 372–378.

[26] S.-H. Li, Y.-C. Kao, Z.-C. Zhang, Y.-P. Chuang, D. C. Yen, A network behavior-based botnet detection mechanism using PSO and K-means, ACM Transactions on Management Information Systems 6 (1) (2015) 3:1–3:30.

[27] C.-Y. Wang, C.-L. Ou, Y.-E. Zhang, F.-M. Cho, P.-H. Chen, J.-B. Chang, C.-K. Shieh, Botcluster: A session-based P2P botnet clustering system on netflow, Computer Networks 145 (2018) 175 – 189.

[28] J. G. Madrid, H. J. Escalante, E. F. Morales, W.-W. Tu, Y. Yu, L. Sun-Hosoya, I. Guyon, M. Sebag, Towards automl in the presence of drift: first results, arXiv preprint - (-) (2019) 1–14.

[29] N. Lu, J. Zhou, Y. He, Y. Liu, Particle swarm optimization for parameter optimization of support vector machine model, in: Proceedings of the 2009 Second International Conference on Intelligent Computation Technology and Automation, IEEE, 2009, pp. 283–286.

[30] J. Peng, S. Wang, Parameter selection of support vector machine based on chaotic particle swarm optimization algorithm, in: Proceedings of the 2010 8th World Congress on Intelligent Control and Automation, IEEE, 2010, pp. 3271–3274.

[31] C. Thornton, F. Hutter, H. H. Hoos, K. Leyton-Brown, Auto-weka: Combined selection and hyperparameter optimization of classification algorithms, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13, ACM, 2013, p. 847–855.

[32] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, K. Leyton-Brown, Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka, The Journal of Machine Learning Research 18 (1) (2017) 826–830.

[33] H. Jin, Q. Song, X. Hu, Auto-keras: An efficient neural architecture search system, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2019, p. 1946–1956.

[34] S. Chowdhury, M. Khanzadeh, R. Akula, F. Zhang, S. Zhang, H. Medal, M. Marufuzzaman, L. Bian, Botnet detection using graph-based feature clustering, Journal of Big Data 4 (1) (2017) 1–14.

[35] E. Biglar Beigi, H. Hadian Jazi, N. Stakhanova, A. A. Ghorbani, Towards effective feature selection in machine learning-based botnet detection approaches, in: Proceedings of the 2014 IEEE Conference on Communications and Network Security, IEEE, 2014, pp. 247–255.

[36] M. I. Hossain, S. Eshrak, M. J. Auvik, S. F. Nasim, R. Rab, A. Rahman, Efficient feature selection for detecting botnets based on network traffic and behavior analysis, in: Proceedings of the 7th International Conference on Networking, Systems and Security, ACM, 2020, p. 56–62.

[37] A. Pektaş, T. Acarman, Deep learning to detect botnet via network flow summaries, Neural Computing and Applications 31 (11) (2019) 8021–8033.

[38] I. Sharafaldin, A. H. Lashkari, S. Hakak, A. A. Ghorbani, Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy, in: Proceedings of the 2019 International Carnahan Conference on Security Technology (ICCST), IEEE, 2019, pp. 1–8.

[39] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, Journal of machine Learning research 12 (2011) 2825–2830.

Journal Pre-proof

Journal Pre-proof

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Journal Pre-proof