Network Analytics for Streaming Traffic Analysis

Sara Khanchi
Dalhousie University
Halifax, Nova Scotia, Canada
Email: s.khanchi@dal.ca

Nur Zincir-Heywood
Dalhousie University
Halifax, Nova Scotia, Canada
Email: zincir@cs.dal.ca

Malcolm Heywood
Dalhousie University
Halifax, Nova Scotia, Canada
Email: mheywood@cs.dal.ca

***Abstract*—Network traffic analytics has always been an important field for cybersecurity and network operations. The analysis could be done on the stream of data (network traffic) in an on-line mode for real-time network operations or on the captured traffic data in an off-line mode for forensic analysis. In this paper, we aim to shed light into using machine learning based network analytics on traffic flows both for network operations and forensic analysis purposes. In doing so, our goal is to design and develop systems that could assist the human expert in analyzing "unknown background" traffic.**

I. Introduction

Network traffic analysis is a fundamental part of network security and operations. The more we know about the traffic we are dealing with, the more we could provide better services and protect the network from malicious activities. Classification is one way of performing network traffic analysis. Previously, the off-line classification methods were mostly used in this area. Over time, network traffic has become more and more sophisticated with huge volumes of data, which makes the feasibility of any analysis method challenging. This growth in traffic volume along with the ever changing nature of normal and attack behaviours, leads to the study of data analytics methods for network analytics both for streaming as well as forensic modes. In the streaming mode, models are built incrementally over time, so only a small part of data is sampled to represent the ground truth, or the *label budget*, which is the percentage of ground truth used to guide model development [1]. Champion model (solution) is available at any point to label the data. However, the champion itself can be replaced/updated at any point in the stream.

In real-world scenarios, not all network traffic is known to the network expert who labels the traffic. This unknown traffic is like a gray area that is full of mysteries to be uncovered. In this paper, our focus is to explore this gray area and extract the known set of behaviours for further analysis and exploitation. To achieve this, we employ the benchmarking CTU-13 botnet datasets, [2]. This dataset includes four classes: Background, Normal, Botnet and Botnet C&C. The last three classes are the minor classes and make up about 4% of the dataset. Conversely, the first class, *Background*, is the gray area that is labeled as "background" by the human experts [2]. We combined all CTU-13 datasets in order to analyze their unknown "background" traffic. We call this CTU13-mixed hereafter. In doing so, our aim is to detect the known behaviours of the CTU-13 datasets, i.e.normal, botnet, C&C, in the background traffic and differentiate those from the "unknown unknowns".

In the rest of the paper, Section II summarizes the related works. Section III describes the off-line and streaming methods that are used. Section IV introduces the datasets and the evaluations employed, whereas Section V presents the results. Finally, conclusions are drawn and the future work is discussed in Section VI.

II. Related works

Ryu *et al*. [3] perform a study of different machine learning algorithms and their ensembles to detect botnet behaviours in the network traffic. For this purpose, they choose three widely used methods: Naive Bayes, Decision Tree and Neural network. Then they investigate if the ensemble of these methods improve the performance. They utilize CTU-13 botnet dataset for evaluation purposes.

Lagraa *et al*. [4] propose *BotGM* to detect botnet behaviours in network traffic utilizing a passive behavioural graph-based model. Firstly, they aggregate the network flows into sequences sharing the same Source and Destination IP addresses. Then, they build graphs on top of the sequences and detect outliers based on unsupervised learning technique which are referred as botnet behaviours. They evaluate the system under CTU-13 botnet dataset. Their results show that their system works better in terms of the accuracy in comparison with the evaluations published on the same dataset for three systems, namely BClus, CAMNEP and BotHunter [2].

Masud *et al*. [5] use an ensemble method to detect botnets in the network data stream. They use a multi-partition multi-chunk method to analyze data in the streaming mode. They generate a botnet dataset based on Nugache bot and assume that all data labels are available.

Gupta *et al*. [6] benchmark on-line machine learning algorithms available in MLib library of Apache Spark[1], big data mining tool. They employ NSL-KDD dataset [7] to measure the performance.

Kato *et al*. [8] propose an anomaly based intrusion detection system. Their system uses Apache Spark to handle the huge volume of data for packet based analysis. UNB ISCX 2012 dataset [9] is used in their case study. They reduce the feature dimension by Principal Component Analysis (PCA) and apply

[1] http://spark.apache.org/

978-3-903176-15-7 © 2019 IFIP

Gaussian Mixture Model (GMM) to cluster traffic into normal or attack classes.

Hsiesh *et al.* [10] propose a Distributed Denial of Service (DDoS) detection system which is based on Neural Networks (NN) implemented in Apache Spark cluster. They use DARPA LLDOS 1.0 dataset [11] for training and apply the model to a real network environment.

Vahdat *et al.* [12] propose a streaming Genetic Programming (GP) classifier which deals with changes in the stream under the constraint of label budget. They demonstrate that GP teaming model is more effective than the single GP model.

Khanchi *et al.* [1] introduce a set of requirements for streaming operation on network data for botnet detection. They assume that the stream of data is imbalanced and subjected to shift/drift and introduce a GP teaming framework for operation under these constraints. They use CTU-13 botnet dataset to evaluate their proposed method. More recently, an analysis on the mixture of the CTU-13 datasets was performed to study the performance of the GP classifier in presence of different botnet behaviours [13].

In summary, previous network analytics related works focus on classifying the "known" traffic behaviours in different traffic datasets using different machine learning methods. In this work, we focus on using streaming and off-line learning algorithms to shed light into the "unknown" network traffic instead.

III. Machine Learning Methods

In this paper, the background traffic behaviour is analyzed by two machine learning methodologies: Stream-GP (streaming) and Random Forest (off-line). Both methods are based on ensemble solutions. Although Random Forest is not a streaming solution in this paper, using Apache Spark framework enables us to apply the off-line trained Random Forest model on a streaming scenario.

Fig. 1 illustrates the overall differences between off-line and streaming (active learning) classifiers. In off-line classification, Fig. 1a, a fixed training partition is used to build a machine learning model. Then the model is applied on the unseen test data. On the other hand, in streaming classification, Fig. 1b, the stream is continuously going. At any time, the classifier has access to the current window of the stream. The classifier learns the stream based on the data determined by sampling and archiving policies, [14] [15]. The champion is updated throughout the course of the stream to adapt to changes. The specifications of the selected methods are explained in the following.

A. Stream-GP Classifier

Stream-GP [1] provides a champion classifier that labels stream content using a 'team' of programs. The composition of the champion team evolves throughout the stream by adding or removing programs.

An active learning architecture is assumed (Fig. 1b), in which the learning process is based on a data subset, $DS(i)$. A data subset decouples the learning process from the size of the dataset. At each learning epoch, Gap(i) exemplars are selected based on Sampling Policy (S), replacing 'gap' exemplars from DS(i). Based on the label budget, β, number of exemplars would be sampled from the current non-overlapping window, sw(i). For example, if $\beta = 0.05$, it means only 5% of the whole data is asked for their true labels. During the sampling process, no label information is known. Thereafter, the sampled exemplars are asked for their true labels and placed in the data subset. An Archiving Policy (A) determines which exemplars to replace. By each data subset update, $\tau = 5$ training epochs are performed and a Champion GP team, $gp*$ is selected to make label predictions.

B. Random Forest Classifier

Random Forest is an ensemble classifier developed by Breiman [16]. It grows a number of binary trees which the final suggested label would be derived based on voting on outputs of the set of trees. Trees of the Random Forest classifier are learned based on different parts of the training set with the goal of decreasing the variance. Therefore, it uses the bagging method to train the tree models. Given $X = x_1, ..., x_n$ with $Y = y_1, ..., y_n$ as labels, bagging will repeatedly (B times) samples randomly with replacement from the training set and fits trees with these samples. Not only it splits the dataset into smaller data subsets, but also the similar process happens for the attribute selection when branching the tree nodes. At each internal node of the tree, a subset of attributes are selected randomly and decision on how to split is determined based on an entropy metric. After training, the predicted label is given based on the voting of trees' outcome.

In this work, Random Forest in Apache Spark is used specifically to deal with the huge volume of data. The implementation uses CART [17] algorithm for classification which Gini impurity algorithm is the metric to evaluate splits in the dataset, Eq. 1. It computes how often a randomly selected attribute would be incorrectly labeled if it was randomly labeled based on the distribution of labels in the subset.

$$i(t) = 1 - \sum_{t=0}^{1} P_t^2 \tag{1}$$

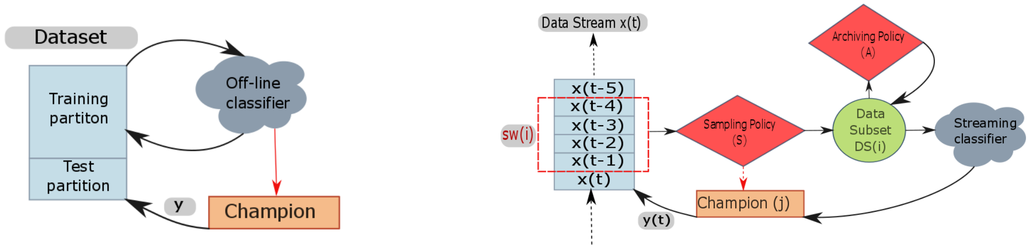
The smaller the Gini impurity index is, the better the candidate is to split. It would continue splitting the nodes until it could not create purer children or a stop criteria is reached.

IV. Empirical Methodology

A. Dataset

In this study, a mixture of CTU-13 botnet datasets is used. These thirteen datasets concatenated so that they become available as a single dataset, CTU13-mixed, which contains all different botnet attacks provided in this collection. Each dataset in this collection contains network flows extracted by Argus flow generator[2]. The flow attributes are listed in Table I with their descriptions. For data analysis purposes,

[2] http://qosient.com/argus/



(a) Off-line classifier

(b) Streaming classifier

Fig. 1. Overview of Streaming classifier versus Off-line classifier

TABLE I
ARGUS FLOW FEATURES FOR CTU-13 DATASET

Feature	Description
stime	Start time
ltime	End time
dur	Duration
saddr	Source IP address
sport	Source port number
dir	Direction of transaction
daddr	Destination IP address
dport	Destination port number
State	Transaction state
SToS	Source ToS byte value
pkts	Total transaction packet count
bytes	Total transaction bytes

IP addresses and port numbers are excluded. This decision is made given that IP addresses could be spoofed and port numbers could be used dynamically by many applications (both benign and malicious) such as VoIP, video streaming or gaming. Once the data is analyzed by ML tools, the excluded features are considered in final visual analysis based on predictions. Four classes of activities are available in this dataset: Background, Normal, Botnet and Botnet Command-and-Control (C&C). Garcia *et al.* [2] assigned these classes manually based on predefined filters. Firstly, all flows are labeled as background by default. Then, if the flow falls in the normal filter definition, it is labeled as normal. Finally, flows relative to the botnet attack designated systems are considered as botnet and botnet C&C. The CTU13-mixed is created by concatenating all CTU-13 datasets. Each dataset is refered as Capture**X**, where X denotes the number of the dataset. The concatenation happens in the following order: Capture5 (Virut), Capture6 (Menti), Capture3 (Rbot), Capture1 (Neris), Capture4 (Rbot), Capture7 (Sogou), Capture2 (Neris), Capture10 (Rbot), Capture8 (Murlo), Capture9 (Neris), Capture11 (Rbot), Capture12 (NSIS) and Capture13 (Virut). The order is prepared in a way to make distance between the same type of bot activity so we could test whether the streaming classifier could recall the previous learned behaviour.

Overall, the CTU13-mixed dataset consists of $19,175,568$ flows, distributed as the background (95.99%), normal (1.78%), botnet (2.2%) and botnet C&C (0.03%) classes.

B. Parameters

In Random Forest, number of trees is set to 50 and Gini impurity index is used. All other parameters are set to the default values in Apache Spark setting. According to the setting, fifty trees are created where nodes are split based on the information of a subset of features due to the Gini algorithm. In the training phase, all the known flows (labeled as normal, botnet or botnet C&C) from the whole dataset are used. Once the model is trained and saved, it is employed on the designated *'background'* traffic, which is not labeled as known behaviour by Stream-GP, for testing purposes.

Stream-GP parameters are taken from the previous work [1]. At each training epoch, $Gap = 20$ exemplars in data subset are replaced with new instances from the current window. In addition, $Tgap = 20$ teams are replaced. Then $\tau = 5$ training generations are performed. Sampling is done under $\beta = 0.05$ label budget which means 5% of the total stream is used for training. As this is a streaming approach, especially under label budget, only a limited amount of training data ($\beta = 5\%$) is available.

C. Tools

1) Apache Spark: Apache Spark is a clustering framework suitable for working on big data analysis and computation. Spark MLlib[3] is a distributed ML library on top of Spark core. In this research, Random Forest classification algorithm from the MLlib library of Apache Spark is used to analyze the background traffic.

2) Microsoft Power BI: Microsoft power BI[4] is a business analytics solution for visualization of data. Here, this tool is used to demonstrate the analysis that each algorithm provides.

D. Analysis Scenario

Stream-GP dynamically changes the champion classifier over the course of the stream, as guided by the 5% of the stream that is labeled. Moreover, it labels the flows as one of the known behaviours, *normal/botnet/botnet C&C*, or as unknown. Our main focus is the background traffic that Stream-GP is able to identify as one of the three known behaviours. It should be noted here that Stream-GP labels the background

[3] Apache Spark MLlib, http://spark.apache.org/docs/latest/mllib-guide.html
[4] https://powerbi.microsoft.com/en-us/what-is-power-bi/

traffic as *"unknown"* only when it is not recognized as one of the known behaviours. After the stream is all labeled by Stream-GP, those original background flows labeled as one of the known classes, are extracted. Then, the Random Forest model trained on the ground truth traffic is applied on the extracted background traffic and results are saved. Random Forest is only trained on the known behaviours so it would not label flows as unknown. The research concentrates on the relations hidden in the predictions of these two methods on the desired extracted background traffic.

V. Results

A. Network Topology

Fig. 2 is an overview of the local network topology for CTU-13 datasets. Normal and botnet systems along their IP addresses are demonstrated. The direct router's IP address, the first router that connects the local designated network to the rest of the world, is discovered based on tracking the Protocol Independent Multicast (PIM) and Internet Group Management Protocol (IGMP) traffic.

Fig. 3 shows an overview of CTU13-mixed dataset based on class distributions and in the order they occur in the stream.

B. Traffic Analysis

The CTU13-mixed background traffic is investigated by two means: Stream-GP and Random Forest. Stream-GP builds a model on-line over the course of exposure to the stream data. On the other hand, Random Forest is learned off-line on the data that ground truth (true label) is known and then applied to the unknown background traffic. In this section, some analysis on both the Ground Truth traffic and Background traffic is represented.

1) Ground truth traffic: refers to the portion of the CTU13-mixed dataset that is labeled manually based on the known behaviours [2]. For training Stream-GP, only a part of this information, based on label budget, is used for learning purposes as would be in a real-life on-line ML scenario. On the other hand, Random Forest uses all this information for learning purposes as would off-line ML methods do.

Some differences are evident between ground truth and background traffic. Some protocols and destination ports are available in background traffic that has no sign in ground truth traffic. For instance, destination port - 13363, takes a major part of background traffic but it does not exist in the ground truth traffic. Botnet traffic is more than Normal traffic in ground truth traffic which is mostly not the case in real life, Fig. 4.

By focusing on the malicious behaviours in the ground truth traffic, Fig. 5, the following observations could be made:

- Almost all Internet Control Message Protocol (ICMP), Simple Mail Transfer Protocol (SMTP) and Secure Shell (SSH) flows are given as malicious.
- Big portions of Secure Hyper Text Transfer Protocol (HTTPS) and Domain Name System (DNS) traffic are also malicious.
- A part of HTTP traffic is also malicious but when we only focus on C&C traffic, it is mostly HTTP rather than HTTPS. This corresponds to one of the well-known ways of establishing C&C connection, using HTTP open port 80 in the literature [18].

Botnets use omnipresent protocols like HTTP and DNS to hide their malicious activities within normal transactions [19]. DNS is essential for Internet transactions to happen so similarly HTTP is that important. By taking advantages of these well-known and widely used protocols, intruders can hide their malicious identity with a mask of normal behaviour and bypass the firewalls. They also use the encryption to completely hide their identities [20].

2) Background traffic: After reviewing the ground truth traffic and how it is composed, it is time to perform analysis on background traffic.

Stream-GP detects most of the known background traffic as normal but it also assigns a big part to malicious activities. Focusing on malicious activities labeled by Stream-GP, Fig. 6, the following analysis is made:

The malicious activities are mostly performed under these protocols: DNS (53), HTTP (80), HTTPS (443) and unassigned ports like 6881 (probable bittorrent) and 13363. The first three protocols are the well-known types of botnet communications [19]. The last one is an unknown port number which is a known peer-to-peer botnet type, bittorent [21]. Additionally, ICMP flows are considered malicious by Stream-GP, given that these could be the first step for port scanning [22]. Moreover, some DNS flows are detected as malicious by Stream-GP whereas Random Forest seesm to miss this. It is shown that one popular way of botnet attacks is through use of DNS protocol [23] [24]. Furthermore, Address Resolution Protocol (ARP) flows are considered normal in Random Forest but malicious by Stream-GP. In ground truth traffic, majority of ARP flows are labeled as malicious, so Stream-GP seems to generalize this known behaviour well. Finally, malicious TCP connections are mostly HTTP and HTTPS. All flows destined to port 8088, a HTTP proxy port, are also detected as botnet by both Stream-GP and Random Forest.

VI. Conclusion and Future Work

The background traffic in CTU-13 botnet dataset is investigated by means of a streaming ML classifier, Stream-GP, and off-line classifier, Random Forest. The outcome of the two different models are analyzed and compared to the known ground truth traffic. Observations demonstrate that background traffic contains various known and unknown behaviours that are not available in the ground truth traffic provided with the CTU-13 dataset [2]. Based on Stream-GP, CTU-13 background traffic is predicted as known behaviours which is distributed to Normal (13%), Botnet (5%) and C&C (4%). The rest of the background traffic (78%) is yet considered as "unknown". In summary, Stream-GP based traffic analysis not only shed light into unknown traffic but also seems to generalize what it learns given the ground truth traffic better. Thus, it could assist

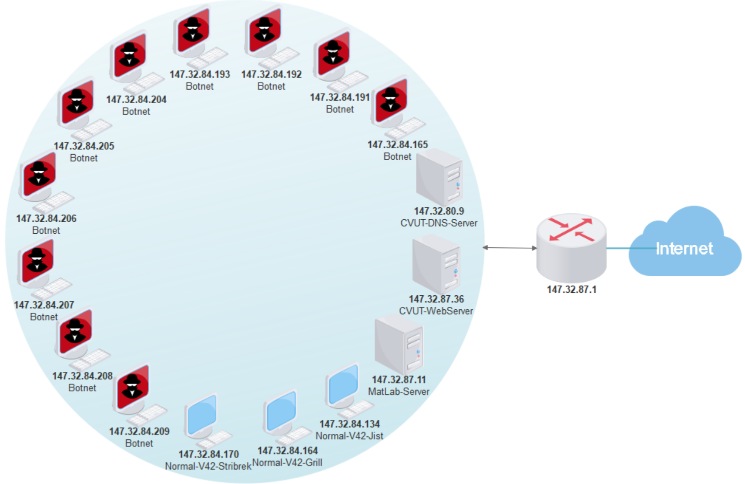


Fig. 2. CTU-13 network topology

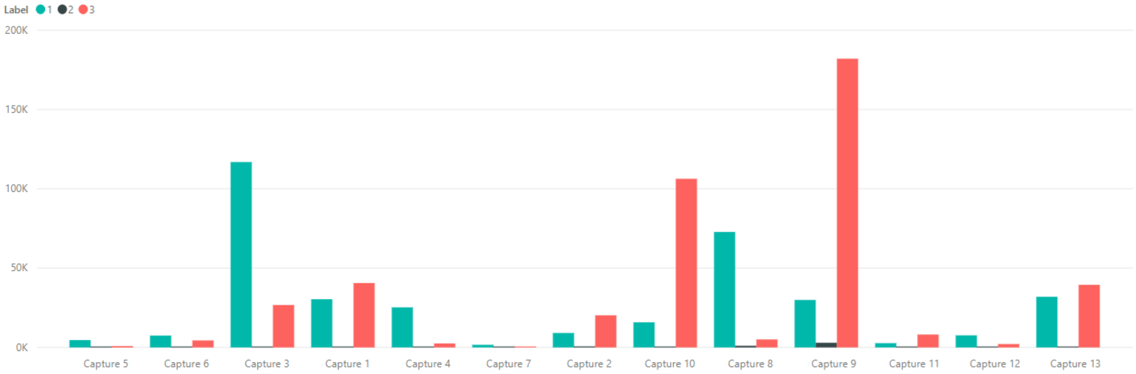


Fig. 3. CTU13-mixed dataset timeline, Labels: Normal (1), Botnet C&C (2) and Botnet (3)

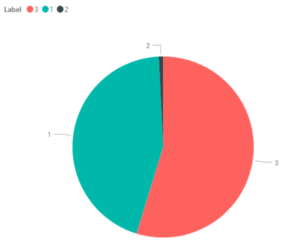


Fig. 4. Distribution of Ground Truth traffic, Labels: Normal (1), Botnet C&C (2) and Botnet (3)

the human analyst under more realistic scenarios. Future work will explore such unknown traffic analysis on other datasets for more benchmarking efforts.

Acknowledgments

This research is supported by Natural Science and Engineering Research Council of Canada (NSERC). The research is conducted as part of the Dalhousie NIMS Lab at: https://projects.cs.dal.ca/projectx/.

References

[1] S. Khanchi, A. Vahdat, M. I. Heywood, and A. N. Zincir-Heywood, "On botnet detection with genetic programming under streaming data label budgets and class imbalance," *Swarm and Evolutionary Computation*, vol. 39, pp. 123 – 140, 2018.

[2] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Computers & Security*, vol. 45, pp. 100–123, 2014.

[3] S. Ryu and B. Yang, "A comparative study of machine learning algorithms and their ensembles for botnet detection," *Journal of Computer and Communications*, vol. 6, no. 5, pp. 119–129, 2018.

[4] S. Lagraa, J. Franois, A. Lahmadi, M. Miner, C. Hammerschmidt, and R. State, "Botgm: Unsupervised graph mining to detect botnets in traffic flows," in *2017 1st Cyber Security in Networking Conference (CSNet)*, Oct 2017, pp. 1–8.

[5] M. Masud, J. Gao, L. Khan, J. Han, and B. Thuraisingham, "A multi-partition multi-chunk ensemble technique to classify concept-drifting data streams," in *Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, 2009, pp. 363–375.

[6] G. P. Gupta and M. Kulariya, "A framework for fast and efficient cyber security network intrusion detection using apache spark," *Procedia Computer Science*, vol. 93, pp. 824 – 831, 2016, proceedings of the 6th

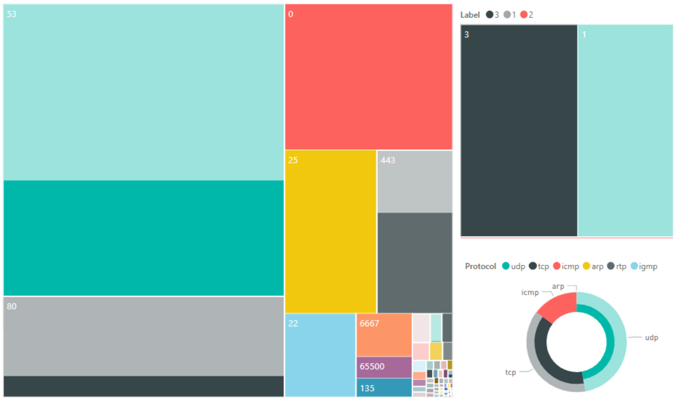
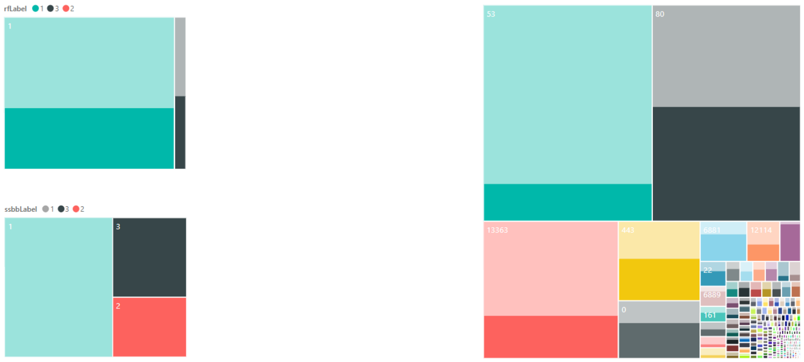


Fig. 5. Botnet activities in ground truth traffic, Labels: Normal (1), Botnet C&C (2) and Botnet (3)



(a) Class distribution

(b) DST port

Fig. 6. Stream GP analytics on malicious botnet activities, Labels: Normal (1), Botnet C&C (2) and Botnet (3)

International Conference on Advances in Computing and Communications.

[7] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, July 2009, pp. 1–6.

[8] K. Kato and V. Klyuev, "Development of a network intrusion detection system using apache hadoop and spark," in *2017 IEEE Conference on Dependable and Secure Computing*, Aug 2017, pp. 416–423.

[9] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, vol. 31, no. 3, pp. 357 – 374, 2012.

[10] C. Hsieh and T. Chan, "Detection ddos attacks based on neural-network using apache spark," in *2016 International Conference on Applied System Innovation (ICASI)*, May 2016, pp. 1–4.

[11] "Mit lincoln lab., darpa intrusion detection scenario specific datasets, 2002. available from:," https://www.ll.mit.edu/r-d/datasets/2000-darpa-intrusion-detection-scenario-specific-datasets.

[12] A. Vahdat, J. Morgan, A. R. McIntyre, M. I. Heywood, and A. N. Zincir-Heywood, "Evolving GP classifiers for streaming data tasks with concept change and label budgets: A benchmarking study," in *Handbook of Genetic Programming Applications*. Springer, 2015, ch. 18, pp. 451–480.

[13] S. Khanchi, N. Zincir-Heywood, and M. Heywood, "Streaming botnet traffic analysis using bio-inspired active learning," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, April 2018, pp. 1–6.

[14] S. Khanchi, M. I. Heywood, and A. N. Zincir-Heywood, "On the impact of class imbalance in GP streaming classification with label budgets," in *European Genetic Programming Conference*, 2016, pp. 35–50.

[15] I. Zliobaite, A. Bifet, B. Pfahringer, and G. Holmes, "Active learning with drifting streaming data," *IEEE Transactions on Neural Networks Learning Systems*, vol. 25, no. 1, pp. 27–39, 2014.

[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001.

[17] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "classification and regression trees," *CRC Press*, 1984.

[18] G. Fedynyshyn, M. C. Chuah, and G. Tan, "Detection and classification of different botnet c&c channels," in *Autonomic and Trusted Computing*. Springer Berlin Heidelberg, 2011, pp. 228–242.

[19] "Http-botnets: the dark side of an standard protocol." http://securityaffairs.co/wordpress/13747/cyber-crime/http-botnets-the-dark-side-of-an-standard-protocol.html.

[20] G. Vormayr, T. Zseby, and J. Fabini, "Botnet communication patterns," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2768–2796, 2017.

[21] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon, "Peer-to-peer botnets: Overview and case study," in *Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets*, ser. HotBots'07. USENIX Association, 2007.

[22] A. Dainotti, A. King, K. Claffy, F. Papale, and A. Pescap, "Analysis of a /0 stealth scan from a botnet," *IEEE/ACM Transactions on Networking*, vol. 23, no. 2, pp. 341–354, April 2015.

[23] Z. Zhu, G. Lu, Y. Chen, Z. J. Fu, P. Roberts, and K. Han, "Botnet research survey," in *2008 32nd Annual IEEE International Computer Software and Applications Conference*, July 2008, pp. 967–972.

[24] C. J. Dietrich, C. Rossow, F. C. Freiling, H. Bos, M. v. Steen, and N. Pohlmann, "On botnets that use dns for command and control," in *2011 Seventh European Conference on Computer Network Defense*, Sept 2011, pp. 9–16.