Towards Effective Feature Selection in Machine Learning-Based Botnet Detection Approaches

Elaheh Biglar Beigi, Hossein Hadian Jazi, Natalia Stakhanova and Ali A. Ghorbani
Information Security Center of Excellence
Faculty of Computer Science, University of New Brunswick, Fredericton, Canada
Email: {elaheh.b, h.hadian, natalia.stakhanova, ghorbani}@unb.ca

***Abstract*—Botnets, as one of the most formidable cyber security threats, are becoming more sophisticated and resistant to detection. In spite of specific behaviors each botnet has, there exist adequate similarities inside each botnet that separate its behavior from benign traffic. Several botnet detection systems have been proposed based on these similarities. However, offering a solution for differentiating botnet traffic (even those using same protocol, e.g. IRC) from normal traffic is not trivial. Extraction of features in either host or network level to model a botnet has been one of the most popular methods in botnet detection. A subset of features, usually selected based on some intuitive understanding of botnets, is used by the machine learning algorithms to classify/ cluster botnet traffic. These approaches, tested against two or three botnet traces, have mostly showed satisfactory detection results. Even though, their effectiveness in detection of other botnets or real traffic remains in doubt. Additionally, effectiveness of different combination of features in terms of providing more detection coverage has not been fully studied. In this paper we revisit flow-based features employed in the existing botnet detection studies and evaluate their relative effectiveness. To ensure a proper evaluation we create a dataset containing a diverse set of botnet traces and background traffic.**

I. Introduction

The recent growth of botnet activity in cyberspace has attracted in a significant way the attention of research community. Botnets have become one of the biggest security threats, responsible for a large volume of malicious activities from distributed-denial-of-service (DDoS) attacks to spamming and phishing.

The concept of botnet refers to a collection of infected computers, bots, that are controlled by an attacker, (i.e., botmaster) through various command and control (C&C) channels. These channels can operate on different communication protocols (e.g. HTTP, IRC) and use various botnet topologies: centralized, distributed (e.g., P2P) or randomized.

A number of techniques aim at enhancing botnet detection were proposed and deployed over the last ten years [1], [2]. The increasingly prevalent approach among these studies is machine learning based detection. One of the critical characteristics that defines the performance of machine learning-based detection is the set of features employed for the classification of malicious activities. The primary motivation to the selection of proper features lies in an inherit trade-off between detection accuracy and computational cost of classification. On the one hand, the use of all possible features inevitably leads to a significant detection overhead. Thus reducing this set of features poses a problem of removing relevant features. On the other hand, inability of the features to capture the diversity of malicious behavior correctly results in low classification accuracy of the detection approach.

For example, packet content-based features have been used widely to classify IRC botnets from normal traffic. However, widespread use of encrypted traffic has frequently highlighted the inefficiency of these features for botnet detection [3], [4]. Employing more features or implementing more complex detection algorithms on the other hand, can cripple timely detection and optimal defense in large scale environments [5], [6]. These challenges drive researchers to utilize some statistical network flow features (regardless of packet content) that help distinguishing botnet traffic from benign traffic in a timely and efficient manner.

In spite of the importance of feature selection, the majority of the existing studies undermine the impact of the feature set on the detection in the real-life deployment environment. Many of the studies resort to selection of features based on intuitive understanding of the problem. For example, *average bytes-per-packets* has been widely used in many P2P botnet detection approaches as it is expected to demonstrate the similarity between flows generated in the same botnet [7], [8].

A number of studies leverage popular feature selection methods such as CFS (Correlation Feature Selection) [9], PCA (Principal Component Analysis) [10], and Minimum-redundancy-maximum-relevance (mRMR) [11] to remove less effective features from other candidate features. The main goal of these algorithms is choosing features (attributes) with high discriminatory power and low overlap with other features in terms of information they can provide. Specific characteristics of the detection models can also be used to eliminate less discriminative features, e.g. features seen on the higher level of a decision tree model have higher discriminatory power [12]. However, these techniques mostly rely on the underlying dataset and detection model to select features which may not necessarily be useful when applied to other datasets or models [12]. Even for the same dataset, using features selection algorithms does not always result in the best performance [13].

Only a small number of the existing studies experimentally

978-1-4799-5890-0/14/$31.00 ©2014 IEEE

evaluate different combinations of features to decide on a feature subset that leads to the highest performance [12], [14], [15].

In addition to the lack of comprehensive evaluation of feature sets, the majority of the existing studies resort to a limited number of botnet traces in their testbed datasets. Although these approaches mostly report a high detection and low false positive rates, obtaining these highly accurate results on a more broad dataset is questionable [16].

In this paper we aim to address these problems and focus on the proper selection and experimental assessment of features for accurate detection of botnets. To understand the impact of commonly employed features on the detection accuracy, we review the rationale behind their selection, and discuss their relative resilience and degree of effectiveness against the potential development of botnets. Since the majority of machine learning-based detection approaches focus on the network flows, in this study we narrow our attention to the flow-based features. In addition to the ability of handling encrypted traffic, flow-based features are less susceptible to privacy concerns as no packet payload information is examined. They are also seen as less computationally expensive compared to the packet-level features.

To properly assess flow-based features, we design a validation dataset that incorporates a wide variety of botnets (centralized and decentralized) using different protocols (IRC, HTTP, P2P). To build a more realistic environment we include traces from 16 both short and long lived (up to several months) botnets.

As a part of our experimental study, we compare the results of our study with the approach recently developed by Zhao et al. [13]. Our results show that the reported study is limited to a few specific botnets used for experimentation. Since feature subset employed by Zhao et al. can represent limited botnet range, the approach fails to detect other types of botnets. The feature set selected by our feature selection algorithm, on the other hand, shows higher detection rate in the more general environment. Our contributions can be summarized as follows:

- We review the flow-based features commonly employed in the botnet detection systems and examine their theoretical applicability to detection of diverse botnets.
- We experimentally analyze their respective impact on classification accuracy of botnet detection.
- We conduct our experiments on a comprehensive set of diverse botnet traffic. To this end we developed a comprehensive dataset consisting of 16 different botnet traces to examine suitability of features for detection of both centralized and decentralized botnets. Among the existing data sets, our set provides the most diverse and comprehensive collection of botnet traces.

To the best of our knowledge, our study is the first attempt to systematically evaluate a large set of features for their applicability to the detection of variety of botnets.

II. Related Work

A significant amount of botnet detection literature have been proposed based on machine learning techniques. Stevanovic and Myrup gave a survey of some existing machine learning-based (network and client based) approaches [17]. Based on the scope of detection, proposed approaches can be categorized into: *centralized, decentralized, and topology independent*.

a) ***Centralized****:* A considerable amount of centralized botnet detection literature emphasized using packet-based features [3], [4]. In addition to the inability of handling encrypted data, packet-based features have a limited application in real time or large scale environments due to the large processing overhead they cause. The IRC botnet detection approach proposed by Livadas et al. relies on supervised machine learning algorithm [12]. Based on the experiments, two features were chosen as the most useful in classifying IRC flows and non IRC flows, namely *average bytes per packet*, and *variance of bytes per packet*. The same features were employed in other IRC detection approaches [14], [18]. Having a single point of failure makes IRC-based botnets less resilient to detection. As a result, botnets with more robust topology and protocols (e.g., HTTP or P2P) became more popular. Garant and Lu designed and implemented a centralized HTTP-based botnet called Weasel [19]. Investigating features that can be applied to other web-based botnets, the authors selected: *length in bytes*, *number of packets*, *flow duration*, *TCP flags*,and *length of flow in bytes*. *Protocol* was also employed as a filtering parameter, and *flow direction*, in conjunction with other features helped diagnosing bot and botmaster sides of the flow.

b) ***Decentralized****:* The majority of existing P2P botnet detection techniques consider botnet detection as a two-step process: (1) identification of P2P traffic/nodes, and (2) detection of malicious P2P connections. Among them is a detection approach proposed by Yen and Reiter [20]. Assuming that P2P nodes are characterized by a large volume of outgoing flows to probe their peers continuously, the authors employed a number of statistical features. Other features used in the first step were flow activity time characterizing persistence of bot activity and a number of distinct destination IP addresses, as malicious P2P nodes were assumed to connect to diverse networks. In the second step several additional features were selected based on two assumptions: 1) clients in the same botnet use the same protocol and network, and (2) there is a significant overlap in peers that P2P bots connect to. The experimental results showed high detection accuracy on a dataset containing traces of Storm and Waledac botnets.

Saad et al. [6] compared five machine learning algorithms to outline their applicability to online P2P botnet detection. They applied 17 features in two groups of *flow-based* and *host-based* features. Flow-based features (*average packet length, total number of bytes, etc.*) help to classify P2P traffic and non-P2P traffic while host-based features (*number of connections over the number of destination IP addresses, ratio of source port to destination ports, etc.*) identify hosts with similar C&C

communication patterns. The same feature set was employed by Zhao et. al [21] in a study that compared two machine learning techniques, decision tree and Naive Bayes for P2P botnet detection. The experiments were conducted using traces of three botnets: Storm, Nugache and Waledac. The study was based on two assumptions: 1) the initial packet of any botnet communication exhibits a unique behavior (e.g., first packet size), 2) bots might randomly connect and disconnect to evade detection. The authors claim that the selected features can cover some of the evasion techniques (e.g., flow perturbation by packet injection or making reconnection). They later improved the list of features used in the classification stage to support detection of novel botnets [13]. P2P botnet detection approach introduced by Noh et al. [22] employed the following features: *protocol* (TCP/UDP), *port* (indicator of port numbers and number of connections created), and *traffic* (indicator of being one/two way connection and the ratio of incoming traffic to outgoing) categories. The experiments were conducted on a set of three botnets: SpamThru, Storm, and Nugache.

*c) **Protocol/topology independent**:* There have been several attempts to provide a protocol/topology free botnet detection mechanism. The basic idea is that bots within the same botnet show similar activities or communication patterns. BotMiner [5] was introduced to identify botnet behavior independent from the employed protocol. BotMiner leveraged the similarities in communication within a botnet and employed the following features to detect them: number of flows per hour, number of packets per flow, average number of bytes per packets, and average number of bytes per second. Although powerful, BotMiner efficiency depends on the malicious traffic detection module that can be evaded using more sophisticated evasion techniques [23]. Yu et al. suggest extracting four basic (packet, byte, and bit statistics) features of the flow in each sliding window to cluster botnet flows in an online and adaptive fashion [7]. Although this approach was presented as protocol independent, the experiments were only conducted on a data set containing Rbot traces. Most recently, Disclosure, as a protocol independent botnet detection system [24] was introduced to detect C&C servers (defined as a pair of IP and port). The employed features were categorized into the three groups: *Flow Size-Based*, *Client Access Patterns-Based*, and *Temporal* features.

III. Flow-based Features

Although a large number of flow-based features were proposed and employed for the detection of various types of botnet, the true value of these features for recognition of botnets is still not clear. In this section we review the features introduced in the previous works and discuss the rationale behind their usage. Summary of the features is given in Table I. It should be noted that due to repetitive use of some features in the existing studies, only unique features are considered. For example, due to redundant information that *number of exchanged bytes per second* and *number of exchanged bits per second* convey, only one feature was retained for the analysis.

Source and Destination IP addresses Source and destination IP addresses have been widely used in the intrusion detection domain. In botnet detection context, their primary value comes from a possibility to quantify number of distinct connections. While as a feature it does not provide a definitive conclusion, it is complementary in assessment of botnet communication patterns. For example, in general the intervals between legitimate connections to distinct IP addresses are longer than the intervals in botnet communication [26]. In practice however, a simple blacklisting of known IP addresses of C&C controllers have been widely employed.

Source and Destination port At the peak of centralized botnets' popularity, source and destination ports were often employed as a definitive feature in recognition of botnet traffic. However with the shift to more resilient technologies, the use of this feature became inefficient. For example, due to periodic change of source-destination port combination, detection of Nugache botnet simply based on this pair became infeasible [27].

Protocol Similar to source-destination port pair, protocol was widely used for filtering out non related traffic (e.g., non-IRC traffic in the centralized detectors) thus reducing the volume of flows requiring further processing [19]. Sometimes the sheer presence of a specific protocol in traffic raises suspicion. For example, IRC traffic is relatively rarely used for legitimate purposes to the extend that in certain networks there is no use for this protocol at all. Similarly since hosts with pure UDP communication are rare, UDP botnets stand out in the network traffic. Lately protocol feature found its place in detectors for grouping similar traffic generated inside a botnet [6], [21].

Duration Duration is one of the most versatile parameters used in detection of botnets. As such initial connection attempts in the majority of botnets are known to be unidirectional and very short followed by much longer communication sessions. Although during the life time of a botnet, duration of communication may vary, certain types of botnets are known to be 'chatty' (e.g., Palevo botnet, IRC botnets). These characteristics have been widely applied in many botnet detection approaches [12], [14], [19], [25].

First packet length (FPS) can be seen as a variation of duration feature. First packet transferred in the flow can reveal some characteristic of the underlying protocol and as such can be useful in detecting specific types of botnets [6], [13], [21].

Flow size features Features characterizing the length of flow (packets) (TBT, APL, DPL, and PV) are mostly intended to represent similar communication patterns. These features have been used with the purpose of both traffic classification (to distinguish specific protocols, especially P2P) and botnet detection. The main assumption is that botnet traffic generated by bots (mostly predefined actions) is more uniform than traffic generated by legitimate users exhibiting a very diverse behavior. Using these features detecting botnets using fixed length commands (e.g., Weasel) is possible [19].

Ratio between the number of incoming packets over the

TABLE I
Summary of flow features used in machine learning based detection approaches

Feature	Description	Type[a]	Reference
Source IP		Independent	[6] [21]
Destination IP		Independent	[6] [21]
Source port		Independent	[6] [21]
Destination port		Independent	[6] [21]
Protocol		Independent	[6] [21] [19]
PX (total number of packets exchanged)	Used to separate normal and P2P traffic and group hosts with similar activities	Independent	[6], [5] [13] [21]
NNP (number of null packets exchanged[b])		Independent	[8] [21]
NSP (number of small packets[c] exchanged)		Decentralized	[8]
PSP (percentage of small packets exchanged)		Decentralized	[8]
IOPR (ratio between the number of incoming packets over the number of outgoing packets)	Used to identify similar communications	Independent	[6]
Reconnect (number of reconnects)	To capture bots that perform frequent reconnections to evade detection	Independent	[21]
Duration (flow duration)	Used to classifying traffic into IRC chat and non IRC. In some botnets e.g. weasel that establish brief connections, this feature can also be used to identify similar malicious communications	Centralized	[19] [12] [14] [25]
FPS (length of the first packet)	Many protocols show identical behavior when exchanging the first packet	Independent	[6] [13] [21]
TBT (total number of bytes)	This feature is used to extract similarity in botnets traffic, e.g. fixed-length commands	Independent	[6] [8] [19]
APL (average payload packet length)	This feature is used to extract similarity in botnets traffic	Independent	[21] [6], [8]
DPL (total number of packets with the same length over the total number of packets)	This feature is used to extract similarity in botnets traffic	Independent	[6]
PV (Standard deviation of payload packet length)	This feature is used to extract similarity in botnets traffic and also classify IRC traffic from non IRC traffic	Independent	[13] [12] [14] [25] [21]
BS (average bits-per-second)	This feature is used to extract similarity in botnets traffic and also to classify IRC traffic from non IRC traffic	Independent	[7] [12] [14] [25]
PS (average packets-per-second in a time window)	This feature is used to extract similarity in botnet traffic and also to classify IRC traffic and non IRC traffic	Independent	[7] [12] [14] [25]
AIT (average inter arrival time of packets)		Independent	[21]
PPS (average packets-per-second)		Independent	[7] [12] [14] [25]

[a]Features applied for both centralized and decentralized botnet detection were considered as $Independent$.
[b]Packets with zero size payload
[c]length of 63-400 bytes

number of outgoing packets (IOPR) A number of studies aiming to assess the breakdown of network traffic on the Internet have emphasized even distribution between inbound and outbound traffic for various protocols [28]. Further analysis specifically into distribution of malicious traffic (including botnet communication) showed that this equality does not hold [29]. Although there is no conclusive evidence that this feature have discriminatory power, these studies give insight into the use of ratio between incoming and outgoing traffic in the context of botnet behavior detection.

Packet exchange Based on the assumption that bots try to keep their connections alive, they usually send a large number of packets. Number of packets exchanged helps to identify this behavior.

Reconnect In an attempt to disguise their behavior, botnet hosts often employ various simple strategies to prevent a detector from analyzing related communication. As such many of the commonly employed features depend on accurate capture of related flows and packets (e.g., number of packets per flow, source-destination port pair, total number of bytes exchanged). Randomly reconnecting an established flow will change the flow-based statistics thus breaking detection based on their similarity [30]. This can be controlled to some extend with a carefully set number of reconnects allowed within a specific time window [21].

Number and percentage of small/null packets exchanged The use of small packets in botnet hosts communications have been widely known. For example, hosts in decentralized botnets tend to exchange small packets to probe their peers, while IRC hosts exchange small chat packets with the C&C servers. The use of null packets in practice however was not seen. Both features though were tested in the recent studies [8], [21].

Average packet/bits per second, average inter arrival time of packets Similar to flow size features, features in this group (BS, PS, PPS and AIT) aim to characterize the similarity of network communication.

IV. Feature Evaluation

Feature selection is a common problem generally viewed in data mining field from two perspectives: filtering and wrapping. Filtering methods use evaluation functions and mostly depend on the properties of underlying data. The 'wrapper' approach, often seen as superior, finds appropriate features through repetitive application of classification algorithm with different sets of features. The best set is then determined by a selected measure of performance. Since the focus of our study is to analyze relative importance of the existing flow-based features and discover the most effective subset of features that will yield the best classification accuracy, we follow the wrapper strategy.

Detection algorithm: We chose the decision tree classifier, a widely used learning approach that demonstrated good accuracy in botnet detection studies [13], [14], [21], [31]. For our flow-based detection we employed a C4.5 version of decision tree with reduced error pruning (REP) that allows to improve the overall detection accuracy of a classifier with a reduced decision tree.

Classification accuracy metric: To evaluate the effectiveness of the considered features we employ two traditional metrics for assessing classification accuracy: detection rate and false positive rate.

Feature selection algorithm: Among wrapper methods there are a number of various heuristic methods, however, the greedy approaches have gained wide popularity. A common greedy procedure is a stepwise selection that adds or removes features at each step of selection. There are two variations of this process: *backward elimination*, i.e, removal operation, or *forward selection*, i.e., addition of features one by one. In our study we combine both techniques. The developed procedure consisting of two steps: *Group Exclusion* and *Feature Inclusion* is given in Figure 1. Group exclusion step aims to evaluate groups of features removing the groups that contribute the least into the overall accuracy. The following Feature inclusion step analyzes each of the features in the worst performing groups aiming to select individual features that might increase the accuracy.

Considering all possible combination of features, calculated by $\sum_{i=1}^{n} C(n,i)$ where n is the total number of features (in our case with 16 features is around 65000 different combinations), would load a significant computational cost specially when dealing with large datasets. To reduce this number, we categorized all features in Table I into four groups[1]: *Byte-based* (features based on byte), *Packet-based* (features based on packets statistics), *Time* (features depend on time), and *Behavior-based* (features representing specific flow behavior). Features in each group are as follows:

- **Byte-based**: TBT, APL, DPL, PV
- **Time-based**: BS, PS, PPS, AIT
- **Behavior-based**: Reconnect, Duration, FPS
- **Packet-based**: PX, NNP, NSP, IOPR, PSP

Input: initialize all groups, final feature set={}, initial detection rate is calculated by applying all features
Output: final feature set

1: **while** detection rate is increasing **do**
2: **for** each group $g_i \in (g_1, g_2, ..., g_n)$ **do**
3: * Group exclusion step *\
4: exclude features in g_i
5: Do experiment using all remaining features (including final feature set), calculate accuracy (a_i);
6: **end for**
7: Find maximum accuracy a_m achieved in the previous step $(a_1, ..., a_n)$ g_m, considered as the worse group (because its exclusion has contributed in the best result), is selected as the candidate group for exclusion;
8: **for** each feature $f_j \in g_m$={$f_1, ..., fk$} **do**
9: * Member inclusion step *\
10: Addf_j and exclude remaining features in g_m;
11: Do experiment and calculate accuracy (a_j);
12: **end for**
13: Find maximum accuracya_m, achieved in the previous step $(a_1, ..., a_k)$; f_m, considered as the best feature in group g_m (because its inclusion has contributed in the best result), is added to the final feature set;
14: **end while**

Fig. 1: Feature Selection procedure

This algorithm starts by calculating the detection rate using all features and considering it as the baseline. In each iteration we exclude one group (Group exclusion step) and see how it affects the final detection rate (increasing or decreasing). Based on the observation, relative importance of each group is revealed. At this step instead of completely removing the worst group, we individually include each member (and remove other features in that group) and evaluate the performance again. In this step the best feature in the worst group is selected and added to the final feature set. The process continues until termination condition is met (examining all groups and features or decreasing the maximum detection rate). Detection result obtained in each experiment is kept to give a ranked list of features. Using mentioned algorithm, we evaluated all features in Table I using our validation dataset.

V. Data Set

Assessing performance of any detection approach requires experimentation with data that is heterogeneous enough to simulate real traffic to an acceptable level. The lack of such data sets available for evaluating botnet detection approaches is well known in the field mostly due to a number of challenges that have been repeatedly emphasized in the literature [16], [32]. In this study, to help us properly access the contribution of various features to the detection of botnet traces, we construct such data set paying a close attention to the following challenges:

[1]We ignore source and destination IP addresses and ports, mostly due to their known inefficiency in universal botnet detection. Although we acknowledge a possible value of these features in particular cases (e.g., blacklisting), in classification context they will mislead the model potentially preventing an accurate analysis of the rest of the considered features.

Generality. Unfortunately, most of the existing botnet datasets have *generality* issue, i.e, they mostly include data from a few botnets (usually two or three samples). Limited in nature (detectors developed in these environments only reflect a small number of characteristics describing a very specific botnet behavior), these approaches are impractical and ineffective in a face of novel threats.

Realism. The effectiveness of the developed approach in practice is highly dependent on realistic botnet traffic traces used for its evaluation. Botnet traffic is usually generated/captured in a controlled environment. Providing a resilient environment (not detectable by the botnet) in which a botnet performs all of its intended malicious functionality is not trivial. In addition to resiliency, collection period must be long enough to allow dormant bots to exhibit their functionality.

Representativeness. Another problem with generating botnet data is an ability of collected network traffic traces to reflect real environment a detector will face during deployment. Due to privacy concerns gathering background data in a real production environment is not feasible in most cases, as a result traffic is either simulated or gathered in a controlled environment. To overcome these challenges, we create an evaluation set combining non overlapping subsets of the following data:

- *ISOT dataset* [13] that has been created by merging different available datasets: French chapter of the Honeynet project [33], Ericsson Research in Hungray [34], and Lawrence Berkeley National Laboratory [35]. It contains both malicious (traces of Storm and Zeus botnets) and non malicious traffic (gaming packets, HTTP traffic and P2P application such as bittorrent). We used 15% and 25% of ISOT dataset in our training and test datasets respectively.
- *ISCX 2012 IDS dataset* [36] has been generated in a physical testbed implementation using real devices that generate real (e.g. SSH, HTTP, and SMTP) traffic that mimics users' behavior. We included a subset of their normal traces in our training dataset. We also included a subset of their normal and IRC botnet traffic in our test dataset.
- *Botnet traffic generated by the Malware Capture Facility Project* [37], a research project with the purpose of generating and capturing botnet traces in long term. From this data we extracted four botnet traces (Neris, Rbot, Virut, and NSIS) for our training dataset and nine botnet traces (Neris, Rbot, Virut, NSIS, Menti, Sogou, and Murlo) for our test dataset [2].

To merge these data traces in one unified data set we employed so called *overlay methodology* [16], one of the most popular methods for creating synthetic datasets. Malicious data is usually captured by honeypots or through infecting computers with a given bot binary in a controlled environment [17]. Botnet traces can be merged with benign data by mapping malicious data to either machines existing in the home network or machines outside of the current network [16]. Considering the wide range of IP addresses in the traces, we mapped botnet IPs to the hosts outside of the current network using Bit-Twist packet generator [38]. Malicious and benign traffic were then replayed using TCPReplay [39] and captured by TCPdump [40] as a single dataset.

TABLE II
Distribution of botnet types in the training data-set

Botnet name	Type	Portion of flows in data-set
Neris	IRC	21159 (12%)
Rbot	IRC	39316 (22%)
Virut	HTTP	1638 (0.94 %)
NSIS	P2P	4336 (2.48%)
SMTP Spam	P2P	11296 (6.48%)
Zeus	P2P	31 (0.01%)
Zeus control (C & C)	P2P	20 (0.01%)

The resulting set was divided into training and test datasets that included 7 and 16 types of botnets, respectively. Tables II and III depict distribution and type of botnets in each dataset. Our training dataset is 5.3 GB in size of which 43.92% is malicious and the reminder contains normal flows. Test dataset is 8.5 GB of which 44.97% is malicious flows. We added more diversity of botnet traces in the test dataset than the training dataset in order to evaluate the novelty detection a feature subset can provide[3].

TABLE III
Distribution of botnet types in the test data-set

Botnet name	Type	Portion of flows in data-set
Neris	IRC	25967 (5.67%)
Rbot	IRC	83 (0.018%)
Menti	IRC	2878(0.62%)
Sogou	HTTP	89 (0.019%)
Murlo	IRC	4881 (1.06%)
Virut	HTTP	58576 (12.80%)
NSIS	P2P	757 (0.165%)
Zeus	P2P	502 (0.109%)
SMTP Spam	P2P	21633 (4.72%)
UDP Storm	P2P	44062 (9.63%)
Tbot	IRC	1296 (0.283%)
Zero Access	P2P	1011 (0.221%)
Weasel	P2P	42313 (9.25%)
Smoke Bot	P2P	78 (0.017%)
Zeus Control(C& C)	P2P	31 (0.006%)
ISCX IRC bot [36]	P2P	1816 (0.387%)

VI. Evaluation study

Our framework has been implemented in C# and utilizes Microsoft Network Monitor to capture packets from a network interface or a pcap file. Feature vectors were extracted in the 60 second time windows. While we experimented with different time window settings, the 60 second time window showed the best accuracy at considerably low computational complexity. The

[2]There are more than one trace for some of these botnet samples

[3]Complete information about datasets is available at http://iscx.ca/botnet-dataset

feature vectors were then passed to the detection module where the decision tree model was applied.

TABLE IV
Iteration1- Group feature exclusion (Exp: Experiment)

Exp	Byte based	Packet based	Time based	Behavior based	Detection Rate
Exp1		✓	✓	✓	68.82%
Exp2	✓		✓	✓	68.58%
Exp3	✓	✓		✓	65.44%
Exp4	✓	✓	✓		66.01%

TABLE V
Iteration1-Feature inclusion step (Exp: Experiment)

Exp	TBT	APL	DPL	PV	Packet,Time, Behavior based	Detection Rate
Exp5	✓				✓	69.03%
Exp6		✓			✓	69.92%
Exp7			✓		✓	66.35%
Exp8				✓	✓	63.49%

A. Feature Selection Algorithm

We evaluated the efficiency of the considered features using feature selection algorithm presented in Section IV in four iterations. Different number of experiments have been done in each iteration, depending on the number of features in the eliminated group. Table IV and V detail *Group exclusion* and *Feature inclusion* steps of the first iteration.

In the first iteration, four experiments were performed. In the first group exclusion step the highest performance was obtained using Packet, Time, and Behavior-based features (i.e. excluding Byte-based features). As a result, the Byte-based group was nominated as the worst group and passed to the Feature inclusion step. Similarly in the second step four experiments were performed. As Table V shows, APL (average payload packet length) contributed the most, increasing the final detection rate from 63% to 69%. Adding PV (standard deviation of payload packet length) showed no improvement in the resulting detection rate (63%). Accordingly, only APL feature from the Byte-based group remained and the rest of the features were excluded.

The rest of the iterations were performed in similar manner. Figure 2 presents the summary of all four iterations. As Figure 2 shows in the last iteration, four features: APL (average payload packet length), IOPR (ratio between the number of incoming packets over the number of outgoing packets), BS (average bits-per-second), and Duration, have been selected as the best final feature set that provide the highest overall detection rate.

Figure 3 shows detection rates achieved in each experiment. Dotted line gives the flow path of the feature selection algorithm. Considering feature groups exclusion order, Byte-based group has less effect on the final detection rate while Packet-based features has the highest impact (by excluding this group, detection rate drops sharply). False positive rate of the experiments ranging from 2%-3% is shown in Figure 4.

The feature selection algorithm terminated at the maximum detection rate of 75% and minimum false positive of 2.3%.

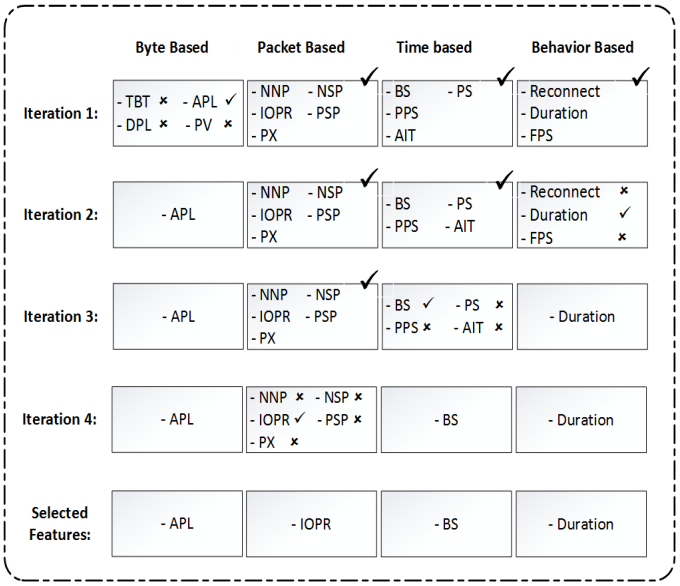

Fig. 2: Iterations summary (check mark indicates the feature selected and passed to the next iteration)

This result was not anticipated as the majority of the previous studies reported very high accuracy. Analyzing the potential reasons behind it, we note that a machine learning algorithm itself can considerably affect the final accuracy, implying that using other detection algorithm might potentially bring a higher detection rate. However, our result to some extend reflects the true performance that perhaps one could expect from solutions deployed in real production environment. A considerable portion of our test dataset contains botnet samples not presented in the training dataset, which means that the detector model is able to detect not seen before cases (a common scenario for an explosion of botnets today) with 75% accuracy.

The nature of analyzed features should also be considered here. Using different evasion techniques, botnets are able to generate network flows very similar to benign applications. For example, *flow duration, bytes per packet, bytes per second, packets per flow, packets per second*, and *number of exchanged packets* can easily be evaded by injecting junk bytes/packets. *Flows per address* or *flows per hour* can also confuse the detection model if bots keep reconnecting at random time intervals. Bots can be commanded to disseminate information with some delays to circumvent time-based correlations. All these challenges make using statistical flow features exclusively inefficient for detection of rapidly growing type of botnets. One possible remedy would be considering multi dimensional snapshots of network traffic, e.g., combining flow level features with pair-level features (a pair of source and destination, no matter which port and protocol used) or conversation level features (a pair of source and destination with the same protocol) to alleviate some of flow features weaknesses.

B. Comparative Evaluation

Since our result is different from the results reported in the majority of the existing studies tested with much less diverse data

and set of features, we conducted a set of experiments to evaluate the recently proposed detection technique on our dataset.

We experimented with the detection method proposed by Zhao et al. [13] which similarly employs a C4.5 decision tree algorithm with feature vectors extracted from the flows using a sliding window technique. For the experiments, we employed the same algorithm settings and parameters (e.g., time window of 60 second) reported in the study.

In the first rounds, we replicated the original experiment using the feature set and dataset (with traces of only three botnets: Storm, Nugache and Waledac) employed in the study [13]. Not surprisingly, we obtained a very similar result to what has been reported in the paper (detection rate of 99% and false positive of 0.001%).

In the second round, we employed the original feature set but tested it using our newly developed training and test datasets. This time the detection rate dropped sharply to 68%, while the false positive increased to 3%.

The high detection rate of the first experiment can be explained by 1) the nature of their dataset, i.e., the dominant botnet portion of their test dataset is SMTP Spam and UDP Storm, which can be easily detected by the selected features, 2) the overlap of training and test datasets created a situation where the decision tree was tested against same data that has been used for training. As a result, the model (not surprisingly) simply retrieved the recorded behavior. At the same time, low detection rate gained in the second round clearly shows the original features employed in the study do not have enough discriminatory power to detect new botnet types (compared to 75% detection rate we obtained using our feature set).

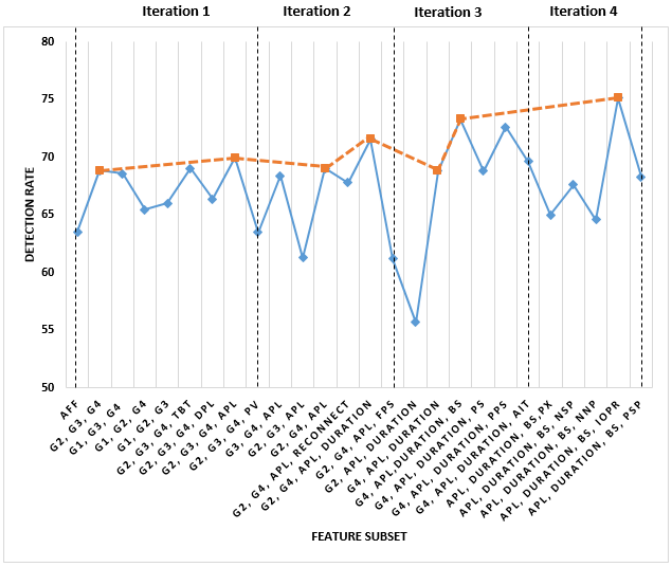

Fig. 3: Detection rate in each experiment (**AFF**: All flow features, **G1**: Byte-based group, **G2**: Time-based group, **G3**: Behavior-based group, **G4**: Packet-based group)

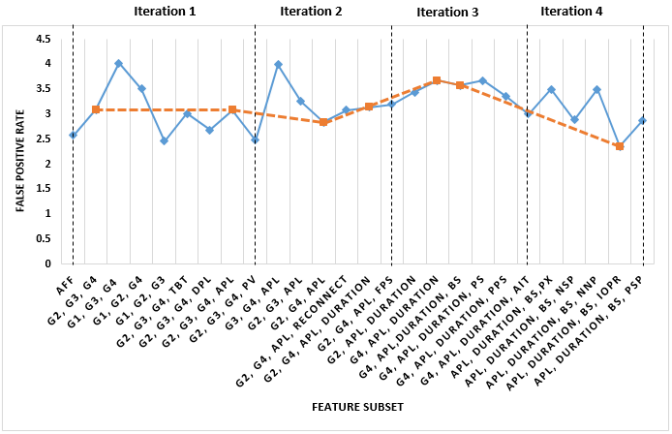

Fig. 4: False positive rate in each experiment (**AFF**: All flow features, **G1**: Byte-based group, **G2**: Time-based group, **G3**: Behavior-based group, **G4**: Packet-based group)

VII. CONCLUSION

The explosion of botnets brought solutions for accurate botnet detection to the fore front of security. The reliance on network traffic has showed the most promise in this regards mostly due to simplicity of modelization, feasibility for online and large scale detection, and detection in all phases of bot's life cycle. To assist this detection machine learning algorithms based on features extracted from network flows have been widely employed. Flow-based features are often seen as the most powerful in identification of specialized botnets. However, their universal effectiveness for detection of various types of botnets has not been fully studied. In this paper, we evaluated flow-based features employed in the previous botnet detection studies using a developed feature selection algorithm. For our experiments we developed and employed a diverse dataset (consisting 16 botnets) to fully test the effectiveness of features for accurate detection.

Although our final feature set showed a high detection rate of 99% on a biased dataset containing limited number of botnets, the more truthful detection ability of these features was discovered on a much more diverse set of botnet traces (75%). Apart from the involved machine learning-based factors, we believe this decrease in performance reflects the inherit nature of the statistical flow features in modeling different botnets. The main reason can be induced from a rich set of evasion techniques used to obscure botnets traffic.

In the context of the mentioned challenges, the necessity to make the botnet detection infrastructure more intelligent is obvious. One possible future work would be considering multidimensional snapshots of network traffic, e.g., combining flow level features with pair level features (a pair of source and destination, no matter which port and protocol used) or conversation level features (a pair of source and destination with the same protocol). Fortifying machine learning-based detection techniques with the complementary security devices (e.g. firewalls or IDSs) could be another solution.

REFERENCES

[1] R. S. Abdullah, M. F. Abdollah, Z. A. M. Noh, M. Z. Mas' ud, S. R. Selamat, R. Yusof, and U. T. M. Melaka, "Revealing the criterion on botnet detection technique," *IJCSI International Journal of Computer Science Issues*, vol. 10, no. 2, pp. 208–215, 2013.

[2] M. Feily, A. Shahrestani, and S. Ramadass, "A survey of botnet and botnet detection," in *Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on*. IEEE, 2009, pp. 268–273.

[3] G. Gu, J. Zhang, and W. Lee, "BotSniffer: Detecting botnet command and control channels in network traffic," in *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, 2008.

[4] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "Bothunter: Detecting malware infection through ids-driven dialog correlation," in *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*. USENIX Association, 2007, p. 12.

[5] G. Gu, R. Perdisci, J. Zhang, W. Lee *et al.*, "Botminer: Clustering analysis of network traffic for protocol-and structure-independent botnet detection." in *USENIX Security Symposium*, 2008, pp. 139–154.

[6] S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, and P. Hakimian, "Detecting p2p botnets through network behavior analysis and machine learning," in *Privacy, Security and Trust (PST), 2011 Ninth Annual International Conference on*. IEEE, 2011, pp. 174–180.

[7] X. Yu, X. Dong, G. Yu, Y. Qin, and D. Yue, "Data-adaptive clustering analysis for online botnet detection," in *Computational Science and Optimization (CSO), 2010 Third International Joint Conference on*, vol. 1. IEEE, 2010, pp. 456–460.

[8] W.-H. Liao and C.-C. Chang, "Peer to peer botnet detection using data mining scheme," in *Internet Technology and Applications, 2010 International Conference on*. IEEE, 2010, pp. 1–4.

[9] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, The University of Waikato, 1999.

[10] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2005.

[11] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1226–1238, 2005.

[12] C. Livadas, R. Walsh, D. Lapsley, and W. T. Strayer, "Using machine learning technliques to identify botnet traffic," in *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*. IEEE, 2006, pp. 967–974.

[13] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, and D. Garant, "Botnet detection based on traffic behavior analysis and flow intervals," *Computers & Security*, 2013.

[14] W. T. Strayer, D. Lapsely, R. Walsh, and C. Livadas, "Botnet detection based on network behavior," in *Botnet Detection*. Springer, 2008, pp. 1–24.

[15] P. Narang, J. M. Reddy, and C. Hota, "Feature selection for detection of peer-to-peer botnet traffic," in *Proceedings of the 6th ACM India Computing Convention*. ACM, 2013, p. 16.

[16] A. J. Aviv and A. Haeberlen, "Challenges in experimenting with botnet detection systems," in *USENIX 4th CSET Workshop, San Francisco, CA*, 2011.

[17] M. Stevanovic and J. M. Pedersen, "Machine learning for identifying botnet network traffic," Networking and Security Section, Department of Electronic Systems, Aalborg University, Tech. Rep., 2013.

[18] X. Zang, A. Tangpong, G. Kesidis, and D. J. Miller, "Botnet detection through fine flow classification," *Departments of CS&E and EE, The Pennsylvania State University, University Park, PA*, vol. 16802, 2011.

[19] D. Garant and W. Lu, "Mining botnet behaviors on the large-scale web application community," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. IEEE, 2013, pp. 185–190.

[20] T.-F. Yen and M. K. Reiter, "Are your hosts trading or plotting? telling p2p file-sharing and bots apart," in *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*. IEEE, 2010, pp. 241–252.

[21] D. Zhao, I. Traore, A. Ghorbani, B. Sayed, S. Saad, and W. Lu, "Peer to peer botnet detection based on flow intervals," in *Information Security and Privacy Research*. Springer, 2012, pp. 87–102.

[22] S.-K. Noh, J.-H. Oh, J.-S. Lee, B.-N. Noh, and H.-C. Jeong, "Detecting p2p botnets using a multi-phased flow model," in *Digital Society, 2009. ICDS'09. Third International Conference on*. IEEE, 2009, pp. 247–253.

[23] E. Stinson and J. C. Mitchell, "Towards systematic evaluation of the evadability of bot/botnet detection methods." *WOOT*, vol. 8, pp. 1–9, 2008.

[24] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel, "Disclosure: detecting botnet command and control servers through large-scale netflow analysis," in *Proceedings of the 28th Annual Computer Security Applications Conference*. ACM, 2012, pp. 129–138.

[25] W. T. Strayer, R. Walsh, C. Livadas, and D. Lapsley, "Detecting botnets with tight command and control," in *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*. IEEE, 2006, pp. 195–202.

[26] Microsoft, "Security intelligence report: Detecting botnets."

[27] H. Hang, X. Wei, M. Faloutsos, and T. Eliassi-Rad, "Entelecheia: Detecting p2p botnets in their waiting stage," in *IFIP Networking Conference, 2013*. IEEE, 2013, pp. 1–9.

[28] W. John and S. Tafvelin, "Differences between in - and outbound internet backbone traffic." in *In TERENA Networking Conference (TNC)*, 2007.

[29] M. Almgren and W. John, "Tracking malicious hosts on a 10gbps backbone link," in *Proceedings of the 15th Nordic Conference on Information Security Technology for Applications*, ser. NordSec'10. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 104–120.

[30] E. Stinson and J. C. Mitchell, "Towards systematic evaluation of the evadability of bot/botnet detection methods." in *WOOT*. USENIX Association, 2008.

[31] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches," *Computer Communications*, vol. 34, no. 18, pp. 2227–2235, 2011.

[32] M. Tavallaee, N. Stakhanova, and A. A. Ghorbani, "Toward credible evaluation of anomaly-based intrusion-detection methods," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 40, no. 5, pp. 516–524, 2010.

[33] "The honeynet project, french chapter," http://www.honeynet.org/chapters/france.

[34] G. Szabó, D. Orincsay, S. Malomsoky, and I. Szabó, "On the validation of traffic classification algorithms," in *Passive and Active Network Measurement*. Springer, 2008, pp. 72–81.

[35] "Lawrence berkeley national laboratory and icsi, lbnl/icsi enterprise tracing project. lbnl enterprise trace repository. 2005," http://www.icir.org/enterprise-tracing.

[36] 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.

[37] S. Garcia, "Malware capture facility project, cvut university," https://mcfp.agents.fel.cvut.cz, retrieved July 03, 2013.

[38] Bit-Twist, "Libpcap-based ethernet packet generator," http://bittwist.sourceforge.net/, retrieved July 10, 2013.

[39] A. Turner and M. Bing, "Tcpreplay: Pcap editing and replay tools for* nix," *online], http://tcpreplay. sourceforge. net*, 2005.

[40] "Tcpdump and libpcap," http://www.tcpdump.org, retrieved July 23, 2013.