A Statistical P2P Botnet Detection Resilient to Mimicry Attacks

Fateme Faraji Daneshgar, Atiye Mohammadkhani, and Maghsoud Abbaspour
(Corresponding author: Maghsoud Abbaspour)

Faculty of Computer Science and Engineering, Shahid Beheshti Univercity
GC, Evin, Tehran 1983969411, Iran
Email: maghsoud@sbu.ac.ir

(Received Mar. 7, 2021; Revised and Accepted Mar. 3, 2022; First Online Apr. 23, 2022)

Abstract

Botnet technology has continued to evolve rapidly, making detection a very challenging problem. P2P botnets are more dangerous and resistant than all emerged botnets due to their distributed architecture. The most proposed P2P botnet detection schemes are designed, relying on the statistical behavior of bots. However, considering the adversarial nature of the botnet detection problem, the bots can be designed to mimic normal behavior and fly under the radar. Thereupon, designing a P2P botnet detection system resilient to the mimicry attack is paramount. In this paper, we implement a mimicry P2P botnet to investigate the resiliency of existing P2P botnet detection schemes. Furthermore, a statistical feature set is proposed to leverage botnets' inherent properties. Our experimental results showed that the proposed feature set is resilient to mimicry attacks and can detect P2P bots with high accuracy.

Keywords: Mimicry Botnet Detection; P2P Botnet Detection; Resilient P2P Botnet Detection

1 Introduction

A 'botnet' is a set of compromised devices as 'bot' (zombie), which is infected by malware instances. It is managed remotely by 'botmaster' through a command and control (C&C) channel. Botnets are becoming increasingly prevalent as the primary enabling technology in a variety of malicious campaigns such as email spam, click fraud, distributed denial-of-service (DDoS) attacks, and Cryptocurrency mining.

To eliminate P2P botnets many P2P botnet detection schemes are proposed. However, despite the impressive efforts of security researchers, they do not achieve much success in the cybersecurity arms race. Malware authors continuously utilize advanced technologies to harden the process of detection [3, 19]. P2P bots are designed to mimic legitimate cyber behavior to fly under the radar and disguise their malicious actions [16,25,26]. For example, Matta *et al.* [16] proposed a botnet with the ability to emulate normal behavior by continually learning admissible patterns from the environment. The bots mimic normal patterns by picking messages from an emulation dictionary, which is learned continually, to ensure that a reasonable innovation rate can be sustained.

The botnet detection mechanisms are based on the distinguishing characteristics of malicious and benign traffic, which are utilized as botnet footprints. Although the proposed approaches are finely tuned to distinguish between benign and malicious behavior, their resiliency against mimicry attacks does not receive much attention. In our previous study [5], we analyzed P2P botnet detection footprints utilized in detection systems in terms of their resilience against the existence of legitimate P2P traffic and mimicry attacks. We observed that the most proposed footprints could be disrupted using mimicry attacks. However, the effect of mimicry attacks on the accuracy of proposed P2P botnet detection schemes is not investigated.

One of the most common approaches to detect botnets is statistical botnet detection, in which a detection model is trained using a machine learning method and some statistical features (like the average of "packet length" and "inter-arrival time between packets" in a flow). The inspiration behind these approaches is that the behavior of bots is statistically different from benign hosts. For example, since the bots communicate with each other to conserve their connectivity and request the botmaster's commands, the length of the packets in malicious traffic is different from that of the benign traffic in which the network flows are utilized to exchange data. However, P2P bots can be designed to mimic normal behavior in terms of these features and subvert the detection mechanisms based on these statistical behaviors.

The resiliency of detection systems is highly dependent on the security of utilized machine learning methods and features. The security of machine learning approaches applied in botnet detection systems is investigated in previous studies [8]. However, the resilience of proposed approaches from the utilized features point of view is not considered. Consequently, investigating the proposed statistical detection approaches in terms of their resilience against the mimicry attack could shed light on the drawbacks of existing schemes and help to design the more resilient approaches.

In this paper, we implemented a mimicry P2P botnet in which the length of the packets is manipulated to mimic normal behavior. The "length of packets" is selected since the most important features used in most proposed approaches with high "information gain" are based on the "packet length" properties. Then, we utilize this botnet to investigate the resilience of the proposed statistical P2P botnet detection systems using this botnet. Furthermore, a feature set is proposed to build a statistical detection model. The proposed feature set is not only resilient to mimicry attacks but also results in botnet detection with high precision.

The rest of this paper is organized as follows. Related work is discussed in Section 2, followed by the implementation details of the mimicry P2P botnet described in Section 3. Section 4 explains our resilient feature set. The evaluation and the experimental results are given in Section 5, and Section 6 concludes the paper.

2 Related Work

The most proposed P2P botnet detection schemes are based on the models built from the network behavior of bots using some machine learning and data mining approaches. However, the problem of botnet detection is adversarial, as the botmaster can change the behavior of bots or mimics the normal behavior to subvert the detection systems. Therefore, in this section, we review the related works in two Subsections. In Subsection 2.1, the previous studies aim at analyzing and evaluating the security of botnet detection proposals are introduced. Then, the proposed P2P botnet detection schemes based on machine learning approaches are reviewed in Subsection 2.2.

2.1 Security Evaluation of Botnet Detection Schemes

Although the security of botnet detection poses critical challenges, it has been addressed in a few studies [5,8,13,22,23]. As mentioned earlier, machine learning techniques are applied in many botnet detection proposals to build the detection model. Therefore, the security flaws of machine learning approaches can be misused by the adversary to subvert the detection system. Thereupon, Gardiner *et al.* [8] systematically investigated attacks against the ML components used in these approaches using existing models from the adversarial machine learning literature.

Stinson and Mitchell [22] introduced a systematic framework for evaluating the evasion ability of bot/botnet detection schemes in terms of the cost of evasion techniques against bot/botnet detection methods. Two costs are considered to evaluate the efficiency of a technique: implementation complexity and the effect of the attack on botnet utility. Implementation complexity is based on the ease with which bot writers can incrementally modify current bots to evade detection. To analyze the impact of an evasion tactic on botnet utility, they introduced five aspects of botnet utility as the diversity of attacks, lead time required to launch an attack, botnet size, attack rate, and synchronization level. They analyzed different evasion tactics in terms of the implementation cost and its impact on botnet utility using the five mentioned criteria.

Knysz *et al.* [13] presented several novel mimicry attack techniques that allow botmaster to avoid fast-flux based detection. In this study, they first analyzed the state-of-the-art fast-flux detectors and their effectiveness against the current botnet threat, demonstrating how botmasters can thwart detection strategies.

Su *et al.* [23] introduced the forward-backward feature that is robust against the variation over payload length, the inter-arrival of packets, and the number of packets in the flow. In this approach, each flow is translated into a string of ¡in¿,¡out¿ representing the direction of the packets in the flow. Then, the corresponding directionless string is computed using the xor operation. Afterward, the produced forward-backward strings with any arbitrary length are mapped to the same dimension feature using n-gram. They showed that adding this feature to traditional feature sets can improve the accuracy of proposed schemes by around 5%, and it solely can achieve an accuracy of 90%. The authors showed that the resilience of previous approaches against the noise could be improved by adding the forward-backward string feature to the traditional feature sets. However, the forward-backward string feature also is not resilient against mimicry attacks as the adversary can insert junk packets between the main packets to disrupt the order of ¡in¿ ¡out¿ strings.

In our previous work [5], we investigated the resilience of proposed P2P botnet detection footprints against three scenarios, including 1) the coexistence of malicious and benign traffic, 2) parasite P2P botnets, and 3) mimicry attacks. We defined the *resilience* using two criteria of distortion of the footprint and the ability of the footprint to distinguish the malicious traffic. If a footprint does not be distorted and could detect the malicious traffic in the evaluation scenario, the footprint is resilient to that scenario. We observed that the most proposed P2P botnet footprints are not resilient to the evaluation scenarios. Therefore, designing the more resilient botnet detection schemes is desirable.

2.2 Machine Learning Based P2P Botnet Detection Schemes

The studies introduced in this Subsection aim at finding the most relative statistical features and most efficient ML approach to train the detection model. We considered these studies to evaluate the resiliency of their proposed statistical feature set against mimicry attacks. As a consequence, the detection rate and results of the papers are not reported.

Garg *et al.* [9] proposed a statistical P2P botnet detection method based on the Random Forest classifier. The authors introduced 12 features related to some flow statistics like duration, number of bytes, and number of packets. They also tackled the problem of imbalanced data using some techniques like downsampling and cost-sensitive learning.

Liao *et al.* [14] used a methodology based on packet size to distinguish between P2P Botnet traffic and legitimate P2P traffic. The authors showed that the size of P2P Botnet packets is smaller than that of any other P2P application. They utilized a feature set based on the statistical characteristics of flows and sessions to detect the bot traffic. Bayesian networks, Naive Bayes and J48, are used to classify network traffic.

Saad *et al.* [18] studied the ability of five different commonly used machine learning techniques to meet online botnet detection requirements, namely adaptability, novelty detection, and early detection. The authors utilized a set of seven flow-based and four host-based features to characterize the malicious p2P traffic. They selected the most common machine learning classification techniques that were used in the literature to detect botnet as Nearest Neighbors Classifier, Linear Support Vector Machine, Artificial Neural Network, Gaussian Based Classifier, and Naive Bayes classifier. However, they found that none of the studied techniques can address all the above requirements at once.

Zhao *et al.* [28] proposed an approach to detect botnet activity by classifying network traffic behavior. The authors selected a set of attributes based on the behavior of various well-known protocols as well as the behavior of known botnets such as Storm, Nugache, and Waledac. For example, the bot queries for updates or instructions on the network continuously, resulting in many uniform-sized, small packets that continuously occur. They utilized a set of seven flow-based and one host-based attributes to train the detection model based on the Bayesian Network and a Decision Tree algorithm.

Yu *et al.* [27] proposed a data mining-based approach for botnet detection base on similarity search and incremental discrete Fourier transform (DFT). To represent raw traffic flows, they captured network flows and converted these flows into a feature stream consisting of some flow-based attributes such as average bytes-per-packet, average packets-per-second, and flow duration. Then, the feature streams were clustered based on the average Euclidean distance. To deal with the challenge of computing the similarities among huge feature streams, the authors used the DFT as the method of similarity search.

Barthakur *et al.* [4] claimed a novel approach to set small rules using Fuzzy logic for P2P botnet detection with the intuition that fuzzy rules with soft boundaries might improve the detection accuracy. They generated fuzzy rules using the Fuzzy Unordered Rule Induction Algorithm (FURIA) from a dataset of four statistical flow-based features extracted from C&C traffic. Ullah Khan *et al.* [12] proposed a P2P botnet detection scheme based on a two-stage traffic classification method. At the first stage, the non-P2P traffic is filtered to reduce the amount of network traffic through well-known ports, DNS queries, and flow counting. To reveal the similarity of C&C communication between zombie hosts in the second stage, they introduced 9 features extracted from network conversation. They utilized three Machine Learning Classifiers of Naive Bayes, Decision Tree, and ANN to train the detection model.

Alauthman *et al.* [2] proposed a P2P Bot detection based on an adaptive multilayer feed-forward neural network in cooperation with decision trees. In this study, 29 features are extracted based on the definition of a connection as a group of packets exchanged between two different hosts, which are identified by the 4-tuple (source IP address, destination IP address, source port, and destination port). Most of the features are related to TCP control packets like SYN, ACK, FIN, etc.

In a more recent study, Alauthman *et al.* [1] developed a reinforcement learning-based P2P botnet detection system, which comprises a traffic reduction method, to deal with a high volume of network traffic. The authors claimed that their proposed system is capable of detecting the bots before the bot launches any malicious activity. In this study, 43 flow-level features and 16 host-level features are collected. Then, a classification and regression tree (CART) [15] is used as the feature reduction technique, and 22 features are selected as significant features. Finally, a detection model is trained using the Reinforcement Learning method.

Gadelrab *et al.* [7] proposed BotCap, a botnet detection system based on statistical characteristics. The authors analyzed several botware samples of known botnets and collected a set of 52 statistical features (mostly introduced in previous studies) to distinguish between benign and malicious traffic. Then, they conducted some experiments to test the suitability of ML techniques and also to pick a minimal subset of the identified features that provide the best detection. However, their testbed includes IRC and HTTP-based botnets lacking the P2P botnets.

Homayoun *et al.* [10] proposed a deep learning-based botnet traffic analyzer called Botnet Traffic Shark (BoT-Shark). To detect malicious botnet traffics, the authors adopted two deep learning techniques, namely Autoencoders and Convolutional Neural Networks (CNNs), to eliminate the dependency of detection systems to primary features achieved by NetFlow extractor tools. Botshark has the capability of detecting malicious traffics from botnets of two common topologies, namely centralized and decentralized botnets. Furthermore, it does not pre-filter any primary extracted features and does not need expert knowledge in selecting proper features to extract features automatically.

Wang *et al.* [24] proposed BotMark, an automated model that detects botnets with hybrid analysis of flow-based and graph-based network traffic behaviors. They utilized 15 statistical flow-based traffic features as well as 3 graph-based features in building the detection model. For flow-based detection, they consider the similarity and stability of C-flow as measurements in the detection. In particular, they employ k-means to measure the similarity of C-flows and assign similarity scores and calculate the stability score of C-flows through the distribution of packet length within a C-flow. The flows that share the same protocol, source IP, destination IP and port within an epoch are defined as a C-flow. The graph-based detection is based on the observation that the neighborhoods of anomalous nodes significantly differ from those of normal nodes in communication graphs. The least-square technique and Local Outlier Factor (LOF) is utilized to calculate anomaly scores that measure the differences of their neighborhoods. BotMark performs automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors by an ensemble of the detection results based on similarity scores, stability scores, and anomaly scores.

3 A Mimicry P2P Botnet

To evaluate the resilience of proposed statistical P2P botnet detection schemes against the mimicry attacks, we designed and implemented a mimicry P2P botnet in which the bots mimic the normal behavior in terms of "packet size." Our mimicry botnet is based on a proof of concept P2P botnet written in Python [11]. This botnet is resistant to targeted takedown attempts and protects the identity of the botnet owner. These characteristics are achieved by using a Kademlia distributed hash table (DHT) and three basic components:

- Bots- The compromised nodes that join the network, query a specific hash in the DHT, post their unique ID, and then wait for an acknowledgment from a commander.
- Commander - A commander that continuously checks the login location for new bots, and sends out ACKs when new bots join the network. Commands are sent via specific query locations that are unique to each bot. Every bot has a unique command location; therefore, it is easy to send commands to individual clients while still being easy to send global commands to all clients. The commander never has direct contact with a bot. All communication is performed through DHT queries, and it protects the commander from being compromised by a rogue client.
- Server - a Kademlia server for clients to bootstrap into the network

There are many tools to manipulate the packet length alive [20] such as Hping, Ostinato, Scapy, and NetfilterQueue Scapy. We utilized NetfilterQueue Scapy, which provides access to the packets that are matched by an iptables rule in Linux. The packets can be accepted, dropped, altered, or given a mark. A NetfilterQueue object represents a single queue. Iptables is an application that allows users to configure specific rules that will be enforced by the kernel's netfilter framework. It acts as a packet filter and firewall that examines and directs traffic based on the port, protocol, and other criteria. To capture the C&C traffic of the bots, we defined the Iptables rule as:
Iptables -A OUTPUT -p udp –dport 8468 -j NFQUEUE –queue-num 1
Then a NetfilterQueue is created and bound to that rule. The packets are matching the rule wait in the queue for manipulation.

Algorithm 1 shows the mimicking process in which the packet size is justified using the normal frequency distribution. For each packet in the queue, the length of its payload is computed, and a random length is selected from F. This length should be greater than the payload length; otherwise, the random selection is repeated. Afterward, some junk bytes are inserted to the payload to reach the Mimicrylen. Finally, the mimicry packet is accepted and sent.

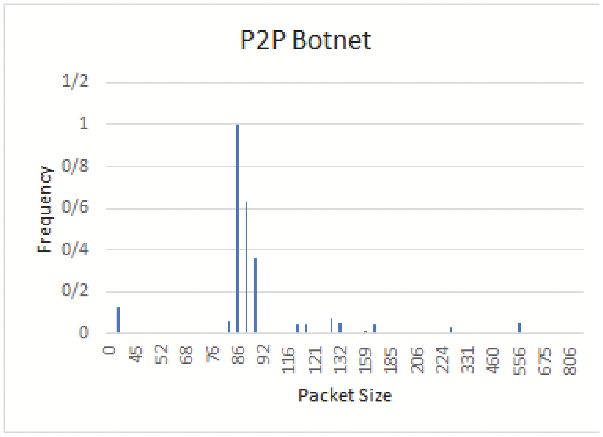
Algorithm 1 Pseudo code of packet manipulation

```
Input: Q: The C&C packet queue, F: The frequency distribution of normal packet size
Output: The mimicry C&C packet
1: for each packet, p, in Q do
2:     packetLen = The payload length of p
3:     Mimicrylen = a randomly selected number from F
4:     while Mimicrylen¡packetLen do
5:         Mimicrylen = a randomly selected number from F
6:     end while
7:     insert $Mimicrylen - packetLen$ junk bytes to the payload of p
8:     accept p
9: end for
```

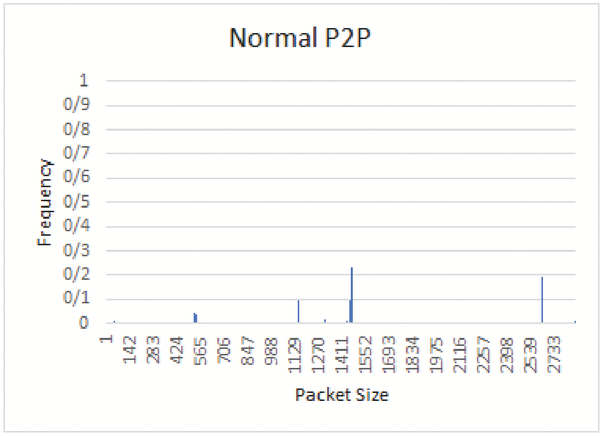
Fig. 1 shows the frequency of the packet size distribution related to the traffic of a P2P bot, an eMule peer, and a mimicry bot. As it is illustrated in the figure, the distribution of packet size in mimicry bot and eMule peer is very similar.

4 A Statistical Feature Set to Characterize P2P Bots Resilient to Mimicry Attack

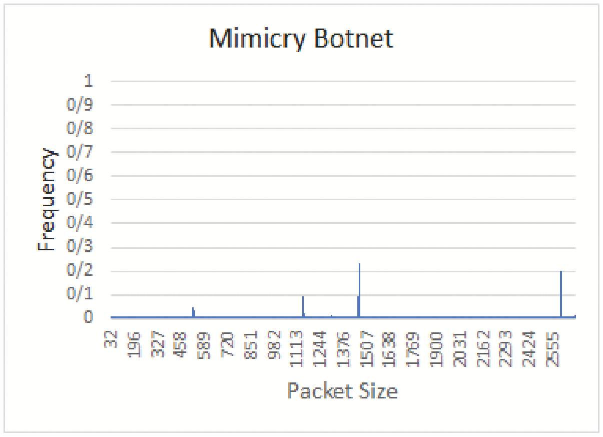
As mentioned before, most statistical botnet detection frameworks are relying on the correlation of flows based on the *Packet Size* and *Timing* behavior of bots to distinguish between malicious and benign traffic. We showed in the previous section that the adversary could mimic



(a) P2P botnet



(b) Normal P2P application



(c) Mimicry botnet

Figure 1: Packet Length Distribution in P2P botnet, Normal P2P application and Mimicry botnet

the normal behavior in terms of the ”packet size” characteristics by selecting a random packet size from normal packet size distribution and injecting the required number of junk bytes into the bot packet.

On the other hand, since bots are predetermined programs, there exists some regularity or periodicity in terms of timing behavior, while no regularity can be seen in human-driven activities. Consequently, to mimic the normal behavior and to fly beneath the radar, the adversary should perturb the timing regularity. This can be done by injecting a bit of random delay between packets sending by bots. Therefore, the main question arises; could we build a detection model based on the statistical characteristics being resilient to mimicry attack?

To define the resilient features, we should focus on the fundamental characteristics of the botnets that are essential to the connectivity of bots and the utility of botnet. In our previous work [5], we analyzed the botnet detection footprints introduced in proposed botnet detection schemes in terms of their resilience against the coexistence of benign P2P traffic, parasite P2P botnets, and mimicry efforts. Based on these analyses and some experimental investigations, we introduced some resilient characteristics. One of the main resilient characteristics of P2P bots is that they keep up persistent communications with each other for efficiency reasons. On the other hand, to conserve stealthiness, they maintain a list of known peers to bootstrap into the network aiming at limiting the number of active connections. Thereupon, P2P bots contact a smaller set of peers in comparison to the benign P2P nodes. Therefore, two resilient characteristics of P2P bots in comparison to benign P2P nodes are:

- long-lived flows
- small set of contact

Thereupon, a statistical feature set defined based on these characteristics would be resilient to mimicry attacks. To capture the ”long-lived flows” attribute, the ”flow duration” feature is used, which is one of the most informative features in most proposed statistical botnet detection schemes. The network flows are generated using 5-tuple <source address, destination address, source port, destination port, protocol >, and the *Flowgap* threshold denoted as W_f. The *Flowgap* threshold indicates the maximum allowed time between the packets in a flow.

However, the network flows in some legitimate P2P applications (like skype) also are long-lived. Nevertheless, our analyses show that the network traffic between two skype peers and two P2P bots is distinguishable using some other statistical characteristics like ”the number of flows in a time window”. As a consequence, to extract these features, the network flows of every two hosts are group into a *Flowgroup* using the *Convgap* threshold denoted as W_c. In other words, these features are computed from network conversations.

The second fundamental attribute of P2P bots is the ”small set of contacts”. To capture this behavior, the net-

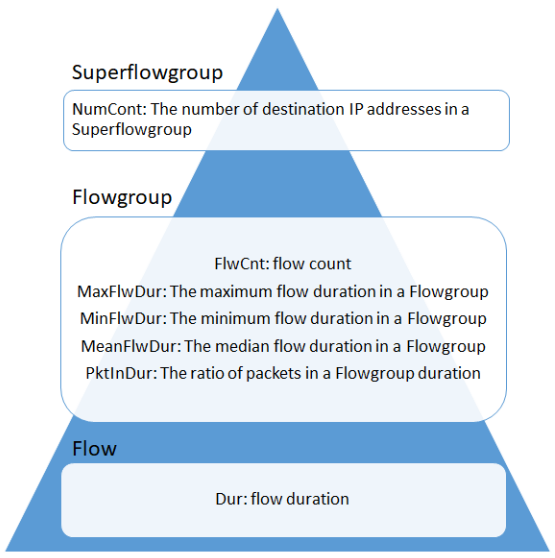

Superflowgroup
NumCont: The number of destination IP addresses in a Superflowgroup
Flowgroup
FlwCnt: flow count
MaxFlwDur: The maximum flow duration in a Flowgroup
MinFlwDur: The minimum flow duration in a Flowgroup
MeanFlwDur: The median flow duration in a Flowgroup
PktInDur: The ratio of packets in a Flowgroup duration
Flow
Dur: flow duration

Figure 2: Proposed resilient feature set

work flows of each host, are aggregated into a *Superflowgroup* using the *Hostgap* threshold denoted as W_h. Then the number of destination IP addresses is computed as "the number of contacts" feature. Fig. 2 shows the proposed statistical features.

Therefore, our proposed resilient feature set is computed from three different flow granularity of *Flow*, *Flowgroup*, and *Superflowgroup*. The statistical features based on the fundamental characteristics of P2P bots Computed from different levels of traffic aggregation could result in resiliency against the mimicry attacks.

5 Evaluation

To evaluate the performance and resilience of the proposed approach, a series of experiments are conducted. The detailed description of evaluation datasets is described in Subsection 5.1. The efficiency of our proposed statistical feature set is evaluated in Subsection 5.2. Finally, in Subsection 5.3, we examine the resilience of the proposed feature set against the mimicry attacks.

5.1 Evaluation Datasets

To experiment with the efficiency and resilience of the proposed statistical feature set, we need two types of datasets. The first one includes the real-world P2P botnet traffic utilized in most proposed P2P botnet detection schemes to evaluate the detection accuracy along with the legitimate P2P traffic and the network background traffic. On the other hand, the second dataset includes the laboratory P2P botnet and the mimicry botnet to investigate the resilience of the proposed feature set in comparison to the previously proposed feature sets. In continue, the evaluation datasets are described in more detail.

5.1.1 Performance Evaluation Datasets

There are two datasets of network traces related to the real-world botnets [17, 18] that are utilized for performance evaluation in most proposed P2P botnet detection schemes. Therefore, to evaluate the detection rate of our proposed feature set along with the previous approaches, we utilized two datasets, D1 and D2, each containing three types of network traces:

- **P2P Botnet network traces**: The ISOT dataset [18] contains the malicious traffic of two Storm bots for 1 hour and a Waledac bot for near one hour. This traffic is utilized as the P2P botnet traffic in D1. The dataset obtained from [17] contains network traces of three P2P botnets, Storm, Waledac, and Zeus. This dataset includes the network traffic of 13 Storm bots for 7 days and 3 Waledac bots for 3 days and a Zeus bot for 34 days. The randomly selected one hour of network traces from three Storm bots and a Waledac bot and a Zeus bot is considered as P2P botnet traffic in the D2 dataset.

- **Legitimate P2P network traces**: The dataset obtained from [17] contains the network traffic of five benign P2P applications, namely uTorrent, eMule, Vuze, Skype, and FrostWire, for several contentious days. We randomly selected one hour of each application's network traces as legitimate P2P traffic for both D1 and D2 datasets.

- **Network background traffic**: The ISCX IDS dataset [21] has been generated in a physical testbed implementation using real devices that generate real (e.g., SSH, HTTP, and SMTP) traffic. We utilize the randomly selected three hours of this traffic as network background traces for both D1 and D2 datasets.

5.1.2 Resilience Evaluation Dataset

As it is described earlier, we implement a mimicry P2P botnet to evaluate the resilience of proposed statistical P2P botnet detection schemes against the mimicry attack. The mimicry botnet is based on a proof of concept P2P botnet [11], which utilizes Kademlia DHT as its C&C channel. We set up both P2P botnet and mimicry botnet in our college lab in a controlled environment consisting of 15 virtual machines as P2P bots, and two virtual machines as bootstrap and commander servers. The C&C traffic of bots from both botnets is captured for one day using Wireshark. Table 1 shows the statistics of the D3 dataset consisting of the randomly selected 6 hours of the collected C&C traffic of the P2P bots ($D3_1$) and the mimicry P2P bots ($D3_2$). It should be noted that the legitimate P2P traffic and the network background traffic in $D3_1$ and $D3_2$ datasets are the same as the D1 and D2 datasets.

Table 1: Statistics of mimicry and P2P traces

C&C traffic	#Packets	#Flows
P2P Botnet ($D3_1$)	771264	1082
Mimicry Botnet ($D3_2$)	737635	1051

5.2 Performance Evaluation and Results

The first step to evaluate the efficiency of the proposed statistical feature set is parameter tuning. Our proposed features are extracted from three levels of Flow, Flowgroup, and Superflowgroup, which are defined using Flowgap (W_f), Convgap (W_c), and Hostgap (W_h) parameters. To find the best values of these parameters, we conducted several experiments with W_f ranging from 20 to 80, and W_c ranging from 120 to 600, and W_h ranging from 900 to 1800 seconds. The J48 decision tree is selected as the classification approach, and 10-fold cross-validation is used to estimate the error rate of the classifier. Tables 2 and 3 show the results of these experiments for D1 and D2 datasets, respectively.

The values of *TPR (True Positive Rate)* and *FPR (False Negative Rate)* are reported as the measures of detection accuracy, and the *Treesize* values (the size of the J48 tree) are reported as the measure of model complexity. The best result is highlighted in each row. The results of this experiment indicate that the accuracy of the detection model built using our proposed resilient feature set is reasonable for a massive range of parameters. Our detection model achieves the TPR= 100% and FPR=0 for some parameter settings.

It can be seen from the tables that the best results are obtained for $W_h = 1500$s and $W_h = 1800$s. In other words, it needs at least 1500 seconds for the host-level behavior of bots to be revealed. However, to select distinct parameter values for remainder of evaluation experiments, we choose $W_f = 60$, $W_c = 600$, and $W_h = 1800$.

To estimate the effect of different machine learning approaches on our proposed detection model, we utilize four commonly used classifiers, namely, Bayesian network(BN), J48 decision tree, Random forest (RF), and Support Vector Machine (SVM) to build the P2P botnet detection model. Table 4 shows the results of this experiment in terms of TPR, FPR, and F-Measure criteria for D1 and D2 datasets. It is evidenced that the detection models built using the proposed feature set and J48 and RF as classification approaches achieve the high TPR of 100% and low FPR of 0%. Despite the lower performance of the detection model built based on SVM and BN, they are yet acceptable and comparable with other published detection schemes.

Table 5 shows the detection performance of our proposed P2P botnet detection model along with the best-reported results of some published P2P botnet detection schemes. It is observed that our proposed approach achieves the highest TPR and lowest FPR. It should be noted that this significant performance is achieved with the cost of computation complexity as our proposed feature set is computed from three different levels of Flow, FlowGroup, and Superflowgroup. Nevertheless, considering the fact that the detection model is learned once in offline mode, this overhead is rational. Furthermore, using this 3-level feature set, we aim at achieving a resilient P2P botnet detection scheme that is investigated in the next subsection.

5.3 Resilience Evaluation of Proposed Statistical Feature Set

To investigate the resilience of our proposed Resilient Feature Set (RFS) in comparison with some other statistical feature sets introduced in the literature, we conducted two experiments. In the first experiment, a P2P botnet detection model is built for each statistical feature set using the J48 decision tree classification approach. These models are trained using the randomly selected 60% of the $D3_1$ dataset (the network traces of a P2P botnet) and then are tested using the remaining 40% of this dataset. The first experiment aims at evaluating our proposed and existing statistical feature set in the same testbed including the same dataset and model training approach.

However, in the second experiment, we consider the resilience evaluation of our proposed feature set in comparison with other statistical feature sets. To this end, we conducted a mimicry attack to the trained P2P botnet detection models trained in the first experiment. In other words, the P2P botnet detection models trained in the first experiments are verified using the randomly selected 40% of the $D3_2$ (the network traces of the mimicry P2P botnet).

The statistical feature sets introduced in other P2P botnet detection proposals are listed in table 6. The descriptions of these features are described Table 7.

To build the detection models, we utilize the J48 decision tree classification implemented in Weka [6]. The process of aggregating the network packets into flows and extracting the features is implemented in python.

Table 8 shows the results of these experiments. The second and third columns show the TPR and FPR of the first experiment and the fourth and fifth columns show the results of the second experiment. It can be seen that the P2P botnet detection model trained using the RFS achieves the best results with the TPR of 99.4% and FPR of 0%.

Furthermore, the P2P botnet detection models trained using the previous statistical feature sets can not resist against the mimicry attack. Fig. 3 shows the reduction in TPR of P2P botnet detection models due on the mimicry attack. The TPR and FPR of 0% related to these models indicate that the whole test samples are detected as normal and no mimicry bot sample is detected. However, the P2P botnet detection model trained using the RFS can detect the mimicry P2P bots with the TPR of 92.9% with a 6.5% decrease compared to the P2P bots.

Table 2: Detection Performance with Different Parameters for D1 dataset

W_f	W_c	$W_h = 900$			$W_h = 1200$			$W_h = 1500$			$W_h = 1800$		
		TPR	**FPR**	**TreeSize**	**TPR**	**FPR**	**TreeSize**	**TPR**	**FPR**	**TreeSize**	**TPR**	**FPR**	**TreeSize**
20	120	0.997	0.002	363	**1.000**	**0.000**	**65**	1.000	0.000	75	1.000	0.000	87
	200	0998	0.002	303	0.999	0.000	61	1.000	0.000	81	**1.000**	**0.000**	**73**
	400	0.998	0.001	293	0.999	0.000	63	**0.999**	**0.000**	**45**	0.998	0.000	117
	600	0.996	0.002	407	0.998	0.001	243	0.999	0.001	91	**1.000**	**0.000**	**51**
40	120	0.998	0.002	299	0.998	0.001	153	**0.999**	**0.000**	**47**	0.998	0.000	181
	200	0.997	0.002	323	0.998	0.001	137	**0.999**	**0.000**	**47**	0.998	0.000	113
	400	0.997	0.002	317	0.998	0.001	179	**0.999**	**0.000**	**105**	0.999	0.001	81
	600	0.997	0.002	341	0.998	0.001	255	**0.999**	**0.000**	**103**	0.998	0.001	27
60	120	0.997	0.002	241	0.999	0.001	167	**0.999**	**0.000**	**133**	0.998	0.001	163
	200	0.997	0.002	239	0.998	0.001	149	**0.999**	**0.001**	**67**	0.998	0.001	181
	400	0.997	0.002	259	0.999	0.001	93	**0.999**	**0.000**	**33**	0.998	0.001	185
	600	0.997	0.003	351	0.998	0.001	185	0.999	0.000	37	**1.000**	**0.000**	**17**
80	120	0.998	0.002	165	**1.000**	**0.000**	**35**	1.000	0.000	51	0.998	0.000	67
	200	0.997	0.002	245	0.998	0.002	131	**1.000**	**0.000**	**47**	0.998	0.001	119
	400	0.998	0.002	173	**0.999**	**0.000**	**33**	0.999	0.000	43	0.998	0.001	19
	600	0.998	0.002	295	0.999	0.000	109	0.998	0.000	115	**1.000**	**0.000**	**21**

Table 3: Detection Performance with Different Parameters for D2 dataset

W_f	W_c	$W_h = 900$			$W_h = 1200$			$W_h = 1500$			$W_h = 1800$		
		TPR	**FPR**	**TreeSize**	**TPR**	**FPR**	**TreeSize**	**TPR**	**FPR**	**TreeSize**	**TPR**	**FPR**	**TreeSize**
20	120	1.000	0.000	85	1.000	0.000	35	**1.000**	**0.000**	**15**	1.000	0.000	23
	200	1.000	0.000	35	1.000	0.000	49	**1.000**	**0.000**	**15**	1.000	0.000	23
	400	1.000	0.000	39	1.000	0.000	51	**1.000**	**0.000**	**15**	1.000	0.000	27
	600	0.999	0.002	265	1.000	0.000	59	1.000	0.000	31	**1.000**	**0.000**	**19**
40	120	0.999	0.003	171	1.000	0.000	87	**1.000**	**0.000**	**15**	1.000	0.000	19
	200	0.999	0.002	171	1.000	0.000	39	**1.000**	**0.000**	**15**	1.000	0.000	19
	400	0.996	0.006	541	1.000	0.000	47	**1.000**	**0.000**	**15**	1.000	0.000	23
	600	0.999	0.002	461	1.000	0.000	47	1.000	0.000	31	**1.000**	**0.000**	**23**
60	120	0.998	0.009	213	1.000	0.000	35	**1.000**	**0.000**	**19**	**1.000**	**0.000**	**19**
	200	0.999	0.003	225	1.000	0.000	43	**1.000**	**0.000**	**19**	1.000	0.000	23
	400	0.997	0.003	193	0.999	0.000	109	**1.000**	**0.000**	**19**	1.000	0.000	23
	600	0.997	0.008	741	1.000	0.000	43	**1.000**	**0.000**	**31**	**1.000**	**0.000**	**31**
80	120	0.999	0.003	199	1.000	0.000	77	1.000	0.002	73	**1.000**	**0.000**	**27**
	200	0.999	0.002	221	1.000	0.000	69	0.997	0.011	143	**1.000**	**0.000**	**27**
	400	0.997	0.004	101	1.000	0.000	59	**1.000**	**0.000**	**39**	0.998	0.001	105
	600	0.996	0.002	123	0.999	0.006	287	**1.000**	**0.000**	**47**	1.000	0.000	49

It can be concluded that the statistical feature set without using the features related to the packet length and timing characteristics of P2P bots is not only resilient to the mimicry attacks but also can achieve a high accuracy

Table 4: Detection Performance with Different Machine Learning Methods

Dataset	Method	TPR	FPR	F-Measure
D1	BN	98.3%	1.3%	98.3%
	J48	100%	0%	100%
	RF	100%	0%	100%
	SVM	93.2%	3.8%	94.1%
D2	BN	99.6%	1%	99.8%
	J48	100%	0%	100%
	RF	100%	0%	100%
	SVM	99.8%	3.4%	91.3%

Table 5: Comparison with other published P2P botnet detection approaches

P2P Botnet Detection Scheme	TPR	FPR
Garg *et al.* [9]	97.93%	0.38%
Saad *et al.* [18]	97.9%	5.1%
Zhao *et al.* [28]	98.1%	2.1%
Yu *et al.* [27]	100%	12.5%
Barthakur *et al.* [4]	99.7%	0.6%
Alauthman *et al.* [2]	99.20%	0.75%
Alauthman *et al.* [1]	99.10%	0.01%
Homayoun *et al.* [10]	91%	13%
Wang *et al.* [24]	98.3%	0.5%
Proposed Approach	100%	0%

to detect the P2P bots.

6 Conclusion

P2P botnets are one of the most serious threats to Internet security. Numerous studies have been done to eliminate P2P botnets. However, malware authors continuously utilize advanced technologies to harden the process of detection. Mimicry attack is one of these efforts in which the P2P bots are designed to mimic cyber behavior to fly under the radar and disguise their malicious actions. Therefore, designing a P2P botnet detection scheme resilient to mimicry attacks is of paramount importance.

Many P2P botnet detection proposals are based on the packet length and timing behavior of the network traffic of P2P bots. Nevertheless, these characteristics can be mimic by P2P bots to subvert these detection systems.

As a consequence, in this paper, we propose a statistical feature set without using the features related to the packet length and timing behavior of bots. The proposed feature set is extracted from the intrinsic characteristics of P2P bots, naming, long-lived flows, and the small set of contacts. These behaviors can not be modified without losing the functionality and efficiency of botnets.

To evaluate the resilience of the proposed feature set we implement a P2P botnet and a mimicry P2P botnet in a controlled environment in our lab. The evaluation experiments show that our proposed statistical feature set is not only resilient to the mimicry attack but also can detect P2P bots with high accuracy.

Table 6: Feature sets used in other published P2P botnet detection approaches

Number	Features	Reference
1	IOP-byte, APL, TBT, PPS, BPS, PL, IOP-frame, DUR, Fromframe, Toframe, Tobyte , Frombyte	[9]
2	IOP, APL, FPL, TPC, TBT, DPL, PL	[18]
3	FPL, APL, PV, TPC, PPS, TBP, NR	[28]
4	TPC, DUR, TBT, ABPP, BitPS, PPS	[27]
5	TPC, LSP, TBLSP, TBT, PLSP, VIT, APL, PV, RPD, RTD	[4]
6	TCPC, TCPR,TCPT, ATCPL, ARCPL, ACPL, TFC, TBT, RAOAC, ATBC	[2]
7	TCPC, TCPT, TCPR, TSYN, RSYN, TACK, RACK, TFC, TSYNACK, RSYNACK, TSYN-RSYNACK, TFINACK, RFINACK, TRST, RRST, TRSTACK, RRSTACK, SYN-ACKTime, SYN-RSTTime, SYNRST-ACKTime, DUR	[1]
8	DUR, TPC, Fromframe, Toframe, SIntPkt, SIntPktIdl, DIntPkt, DIntPktIdl, TBT, Tobyte, Frombyte, SIntPktAct, DIntPktAct, sMeanPktSz, dMeanPktSz, sMinPktSz, dMinPktSz, sMaxPktSz, dMaxPktSz, BitPS, SrcLoad, DstLoad, PPS, SrcLoad, DstLoad	[10]
9	DUR, TPC, NSP, AIT, TBT, APL, PV, FPL, DPL, LSP, MP, TBLSP, BPS, PPS, FPH	[24]

References

[1] M. Alauthman, N. Aslam, M. Al-Kasassbeh, S. Khan, A. Al-Qerem, and K.-K. R. Choo, “An efficient reinforcement learning-based botnet detection

Table 7: Description of statistical features used in published approaches

Feature Name	Description	Feature Name	Description
FPL	Length of the first packet in the flow	TPC	Total number of packets per flow
TBT	Total number of bytes per flow	APL	Average packet length per flow
IOP	Ratio of the number of incoming packets over the number of outgoing packets	DPL	Total number of subsets of packets of the same length over the total number of packets in the same flow
PV	Variance of payload packet length	DUR	Flow duration
PPS	Number of packets exchanged per second	BPS	Number of bytes exchanged per second
BitPS	Average bits-per-second for flow	Tobyte	Number of outgoing bytes
Frombyte	Number of incoming bytes	IOP-byte	Ratio of incoming over output bytes in the flow
IOP-frame	Ratio of incoming over outgoing number of frames in the flow	Fromframe	Number of Incoming frames in the flow
Toframe	Number of outgoing frames in the flow	LSP	Size of the packet carrying maximum bytes in a flow
TBLSP	Total bytes transferred with largest sized packets	PLSP	Portion of largest sized packet
VIT	Variance of inter-arrival time	RPD	Difference in number of packets between two responding flows
RTD	Difference in time of last packet received for two responding flows	ABPP	Average Byte-per packet
PL	Total number of bytes of all the packets over the total number of packets in the flow	TBP	Average time between packets
NR	Number of reconnects	TCPC	Number of control packets in the flow
TCPT	Number of transmitted control packets in the flow	TCPR	Number of received control packets in the flow
TSYN	Number of transmitted SYN packets in the flow	RSYN	Number of received SYN packets in the flow
TACK	Number of transmitted ACK packets in the flow	RACK	Number of received ACK packets in the flow
TFC	Number of transmitted failed connection in the flow	TSYNACK	Number of transmitted SYN-ACK packets in the flow
RSYNACK	Number of received SYN-ACK packets in the flow	TSYN-RSYNACK	Transmitted SYN-Received SYN-ACK
TFINACK	Number of transmitted FIN-ACK packets	RFINACK	Number of received FIN-ACK packets
TRST	Number of transmitted RST packets	RRST	Number of received RST packets
TRSTACK	Number of transmitted RST-ACK packets	RRSTACK	Number of received RST-ACK packets
SYN-ACKTime	Inter-arrival time between SYN and ACK	SYN-RSTTime	Inter-arrival time between SYN and RST
SYNRST-ACKTime	Inter-arrival time between SYN and RST-ACK	ATCPL	Average length of transmitted control packets in the flow
ARCPL	Average length of received control packets in the flow	ACPL	Average length of control packets in the flow
RAOAC	Ratio of average length of outgoing packets over the average length of control packets	ATBC	Average time between an attempt to create connection
SIntPkt	Source inter-packet arrival time	SIntPktIdl	Source idle inter-packet arrival time
DIntPkt	Destination inter-packet arrival time	DIntPktIdl	Destination idle inter-packet arrival time
SIntPktAct	Source active inter packet arrival time	DIntPktAct	Destination active inter packet arrival time
sMeanPktSz	Average of transmitting bytes	dMeanPktSz	Average of received bytes
sMinPktSz	Minimum transmitted packet size	dMinPktSz	Minimum received packet size
sMaxPktSz	Minimum transmitted packet size	dMaxPktSz	Minimum received packet size
SrcLoad	Transmitted bits per second	DstLoad	Received bits per second
SrcRate	Transmitted packets per second	DstRate	Received packets per second
NSP	Number of small packets	AIT	Average arrival time of packets
MP	The number of maximum packets	FPH	The number of C-flows per hour

Table 8: TPR and FPR of Resilience evaluation experiments

Feature set	TPR-P2P	FPR-P2P	TPR-mimicry	FPR-mimicry
1	95.4%	0.1%	0	0
2	98.5%	0%	0	0
3	98.7%	0%	0	0
4	95.4%	2.7%	0	0
5	96.5%	0.1%	0	0
6	97.5%	0.1%	-	-
7	97.1%	0.1%	-	-
8	92.3%	7.9%	0	0
9	98.3%	0.5%	0	0
RFS	99.4%	0%	92.9%	0

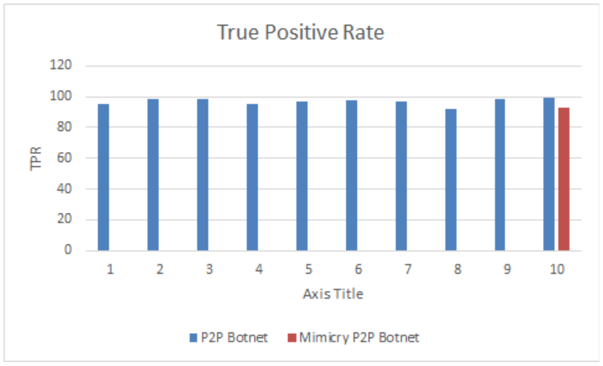


Figure 3: True Positive Rate of P2P botnet detection models for P2P and mimicry P2P botnets traces

approach," *Journal of Network and Computer Applications*, vol. 150, p. 102479, 2020.

[2] M. Alauthaman, N. Aslam, L. Zhang, R. Alasem, and M. A. Hossain, "A p2p botnet detection scheme based on decision tree and adaptive multilayer neural networks," *Neural Computing and Applications*, vol. 29, no. 11, pp. 991–1004, 2018.

[3] S. T. Ali, P. McCorry, P. H. J. Lee, and F. Hao, "Zombiecoin 2.0: managing next-generation botnets using bitcoin," *International Journal of Information Security*, vol. 17, no. 4, pp. 411–422, 2018.

[4] P. Barthakur, M. Dahal, and M. K. Ghose, "Adoption of a fuzzy based classification model for p2p botnet detection." *IJ Network Security*, vol. 17, no. 5, pp. 522–534, 2015.

[5] F. F. Daneshgar and M. Abbaspour, "On the resilience of p2p botnet footprints in the presence of legitimate p2p traffic," *International Journal of Communication Systems*, vol. 32, no. 13, p. e3973, 2019.

[6] E. Frank, M. A. Hall, and I. Witten, "The weka workbench. online appendix," *Data mining: practical machine learning tools and techniques*, 2016.

[7] M. S. Gadelrab, M. ElSheikh, M. A. Ghoneim, and M. Rashwan, "Botcap: Machine learning approach for botnet detection based on statistical features," *International Journal of Communitation Networks and Information Security*, vol. 10, no. 3, p. 563, 2018.

[8] J. Gardiner and S. Nagaraja, "On the security of machine learning in malware c&c detection: A survey," *ACM Computing Surveys*, vol. 49, no. 3, pp. 1–39, 2016.

[9] S. Garg, A. K. Sarje, and S. K. Peddoju, "Improved detection of p2p botnets through network behavior analysis," in *International Conference on Security in Computer Networks and Distributed Systems*. Springer, 2014, pp. 334–345.

[10] S. Homayoun, M. Ahmadzadeh, S. Hashemi, A. Dehghantanha, and R. Khayami, "Botshark: A deep learning approach for botnet traffic detection," in *Cyber Threat Intelligence*. Springer, 2018, pp. 137–153.

[11] J. Howard, "Pythonp2pbotnet," Tech. Rep. https://github.com/jhoward321/PythonP2PBotnet, April 2018.

[12] R. U. Khan, R. Kumar, M. Alazab, and X. Zhang, "A hybrid technique to detect botnets, based on p2p traffic similarity," in *2019 Cybersecurity and Cyberforensics Conference (CCC)*. IEEE, 2019, pp. 136–142.

[13] M. Knysz, X. Hu, K. G. Shin, and M. S. Hwang, "Good guys vs. bot guise: Mimicry attacks against fast-flux detection systems," in *Proceedings of IEEE INFOCOM*, April 2011, pp. 1844–1852.

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

[15] W. Y. Loh, "Classification and regression trees," *Wiley interdisciplinary reviews: data mining and knowledge discovery*, vol. 1, no. 1, pp. 14–23, 2011.

[16] V. Matta, M. D. Mauro, P. H. J. Lee, and M. Longo, "Ddos attacks with randomized traffic innovation: Botnet identification challenges and strategies," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1844–1859, 2017.

[17] B. Rahbarinia, R. Perdisci, A. Lanzi, and K. Li, "Peerrush: Mining for unwanted p2p traffic," in *International conference on detection of intrusions and malware, and vulnerability assessment*. Springer, 2013, pp. 62–82.

[18] 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 *2011 Ninth annual international conference on privacy, security and trust*. IEEE, 2011, pp. 174–180.

[19] SECTOR, "Building botnets on the blockchain," Tech. Rep., Nov. 2017. (https://sector.ca/building-botnets-on-the-blockchain/)

[20] P. Shankdhar, "15 best free packet crafting tool," Mar. 4, 2018. (https://resources.infosecinstitute.com/15-best-free-packet-crafting-tools)

[21] 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.

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

[23] Y. H. Su, A. Rezapour, and W. G. Tzeng, "The forward-backward string: A new robust feature for botnet detection," in *2017 IEEE Conference on Dependable and Secure Computing*. IEEE, 2017, pp. 485–492.

[24] W. Wang, Y. Shang, Y. He, Y. Li, and J. Liu, "Botmark: Automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors," *Information Sciences*, vol. 511, pp. 284–296, 2020.

[25] Z. Wang, M. Qin, M. Chen, C. Jia, and Y. Ma, "A learning evasive email-based p2p-like botnet," *China Communications*, vol. 15, no. 2, pp. 15–24, 2018.

[26] S. Yu, S. Guo, I. Chen, and I. Stojmenovic, "Fool me if you can: Mimicking attacks and anti-attacks in cyberspace," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 139–151, 2013.

[27] X. Yu, X. Dong, G. Yu, Y. Qin, D. Yue, and Y. Zhao, "Online botnet detection based on incremental discrete fourier transform," *Journal of Networks*, vol. 5, no. 5, p. 568, 2010.

[28] 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*, vol. 39, pp. 2–16, 2013.

Biography

Fateme Faraji Daneshgar received her bachelor's degree in software engineering from Tarbiat Moallem University in Tehran in 2004. Immediately after graduation, she was accepted for a master's degree in software engineering at Tarbiat Modares University. Having completed his master's degree in 2007, she worked for about 2 years as a research associate at Bank Mellat Research Center and the National Library of Iran. He also worked as a software designer on some IT projects for 3 years before starting his Ph.D. Program at Shahid Beheshti University, Iran in 2012. She graduated in 2021 with a doctorate. Her main research interest is "network security" and "data mining" and "machine learning".

Atiyeh MohammadKhani received her B.S degree in Software Engineering from Dr. Shariati Technical and Vocational College of Tehran, Iran in 2013. She finished her Master in Information Technology Engineering from Shahid Beheshti University, Tehran, Iran in 2019. Her main research interests are network security, data mining, malware detection, social network analysis and IoT security.

Maghsoud Abbaspour received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Tehran, Tehran, Iran, in 1992, 1995 and 2003, respectively. He is an associate professor of Faculty of Computer Science and Engineering and director of Computer Networking and Network Security Laboratory in Shahid Beheshti University, Tehran, Iran since 2005. He is interested in Wireless Sensor Networks, peer to peer and ad hoc networks, network security and Internet of Things.