The Impact of Different Botnet Flow Feature Subsets on Prediction Accuracy Using Supervised and Unsupervised Learning Methods

Sean Miller, Curtis Busby-Earle
Department of Computing
The University of the West Indies Mona

Abstract

Over the past ten to fifteen years botnets have gained the attention of researchers worldwide. A great deal of effort has been given to developing systems that would efficiently and effectively detect the presence of a botnet. This unique problem saw researchers applying machine learning (ML) to solve this problem. In this paper, a brief overview of the varied machine learning methods (ML) and their utility in relation to botnet detection is provided. The main aim of this paper is to clearly define the role different ML methods play in Botnet detection. We also examine different flow level feature subsets and the resulting impact on detection accuracy given the machine learning method used. A clear understanding of these various roles are critical for developing effective and efficient real-time online-detection approaches and ultimately, more robust models. In conclusion, it was found that, the features selected must compliment the machine learning method chosen.

1. Introduction

Individuals and businesses have become more depen-dent on Internet services and information systems to ef-ficiently and effectively carry out their everyday tasks. Consequently, the need for protecting the confidential-ity, integrity and availability of these systems has never been greater. Botmasters can use the aggregated power of many bots to exponentially intensify the impact of ma-licious activities. A single bot may not be a danger for the Internet, but a network of bots will definitely be able to create a huge disturbance [18]. We have reviewed sev-eral proposed ML-Based botnet-detection systems. Tak-ing into consideration the ML Method, we explore the role of each.

This paper is arranged as follows: Section 2 highlights work related to botnet detection systems. Section 3 gives an overview of the botnet phenomenon, describing the way in which bots work, different types of bots and the botnet life cycle. Section 4, botnet-detection, highlights the varied botnet detection techniques and approaches and their significance as it relates to the different machine learning methods. Section 5 provides a brief description of what machine learning entails and the features used to create theses models. Section 6 evaluates the role of the ML method embodied in each botnet detection approach. Section 7 describes the manner in which the dataset was obtained and prepared and details processes by which the classes are distributed. Additionally, this section presents the results from each model and feature subset combination. In conclusion section 8 evaluates both what has been established in the literature and the results as highlighted in the previous section.

2. Related Work

Several authors have written reviews of botnet-detection approaches as well as detection techniques. Maryam Feily [2] conducted a survey of botnets and bot detection, explaining the way in which bots operate; examining dif-ferent botnet-detection approaches placing botnets in one of three classes, namely anomaly-based, DNS based or Mining based detection. The paper also surveyed botnets and botnet detection, its aim was to explain the botnet phenomena and explore different botnet detection tech-niques. The paper classified botnet detection into four classes, namely: anomaly-based, signature-based, DNS-based, and mining- based. Along with the summarization of each class, detection techniques were compared.

Thomas Hyslip et al [3] surveyed botnet detection techniques based on their command and control infrastructure. With a focus on bot detection technique, this review examined various detection techniques and their impact on different botnet architectures.

Michael Bailey et al surveyed bot technology and defenses [4]. This survey scrutinized different detection methods in light of the data source which provide the bases for detection, examples of such sources are DNS, net flow, packet tap etc. They moreover examined different techniques used by detection methods inclusive of detection based on group behavior and detection based on signature. This survey paper also examined existing botnet research, the evolution and the future of botnets.

Stevanovic [1] surveyed machine learning based botnet detection approaches. His paper proffered a review of current machine learning based botnet detection methods for identifying botnet-related traffic. An overview on this area of study was given by summarizing the recent scientific efforts in this arena. His paper also examined each method and their susceptibility to various resilience techniques which may be employed by botmasters. Also examined was the result generated from each technique, the algorithm used and features chosen. Akin to this paper our focus is on ML-based botnet detection methods. The difference is that our paper explores how distinct researchers used ML- Methods to overcome specific challenges.

The main direction of this paper is a refocusing on ML-based approaches introduced by various researchers, using different ML methods to detect botnet activity. The main aim of this study is to determine which combination of ML method features and techniques is optimal when tackling the botnet detection dilemma.

3. The Botnet Phenomenon

A botnet is a network of infected computers, (referred to as zombies or bots), enslaved by an attacker to carry out their bidding. The users whose computers make up the bots in a botnet are usually ignorant of the fact that their systems have been compromised and are potentially taking part in malicious activities.

The resources of the infected computer - (bot), are under the control of the attacker - (botmaster), who uses these resources for his own agenda. Commands are given and received through communication between the enslaved computers and their botmaster via what is known as a command and control server, (C&C). The botnet life-cycle is described in four stages by Leonard et al. [19]

1. Formation (also called infection)
2. Command and Control environment
3. Attack
4. Post Attack

At the formation stage, the bot is installed on the users machine by exploiting vulnerabilities, an aspect of this malicious code is responsible for connecting the bot to its command and control server (C&C). After establishing C&C connection, the botmaster is able to send commands to the newly-added bot. This then transitions into the attack phase, whereupon instructions from the botmaster to the bots in the botnet through the C&C channel, attack commands are issued. Following an attack, bots may become exposed and cured, that is, the vulnerabilities which were exploited may be patched. The goal then of the botmaster is to recruit more bots - (post attack), and thus, cycle continues.

3.1. Command and Control architectures

The command and control channel (C&C) is the back-bone of a botnet. The C&C channel is that link be-tween the botmaster and the bot. Unique to bot mal-ware, the C&C channel is its defining characteristic [1]. The C&C channel is used to send instructions to bots as well as receive information from them. C&C chan-nels are usually one of three architectures: centralized, decentralized or hybrid (see fig 1). In a centralized bot-net, all the bots in the network connect to the same C&C server(s), controlled by a single botmaster. In contrast to other architectures this has the lowest latency of commu-nication. Decentralized C&C architectures are designed with resilience in mind. Void of a central point of failure, like that of centralized C&C architectures, these botnets posses multiple paths for sending instructions. The decentralized C&C makes use of peer to peer (P2P) communication protocols as the means for connecting to the infected machines. Hybrid approaches seek to combine the principles of the other two architectures by using hybrid P2P protocols alongside the low network delay found in centralized architectures.

4. Botnet Detection

As botnets become more threatening, researchers and security experts employ different approaches and tech-niques to solve the problem. The detection approach defines how the solution operates, such as detection by behavior [13] or signature [6]. Different techniques are based on different approaches. ML-based detection tech-niques are capable of using both approaches. Other tech-niques used in bot detection include anomaly and DNS.

4.1. Detection Approaches

Signature-based approaches require detailed knowledge of what a bot or bot-related characteristics, (e.g. traf-fic), may look like. This approach is used to target spe-cific characteristics such as a particular protocol [26] or service. This type of approach tends to be very precise and specific. Anything outside the specified scope will

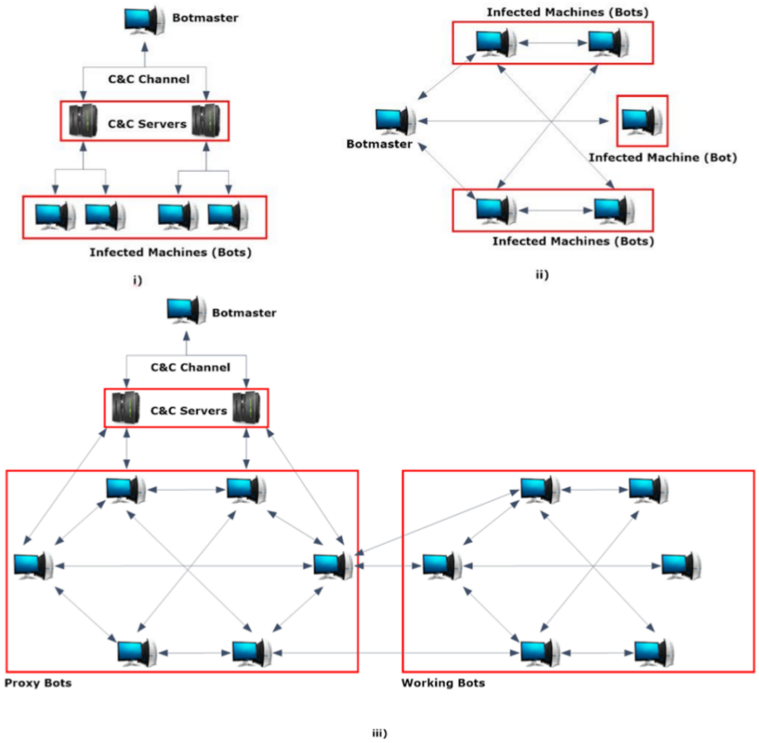

Figure 1. Botnet architectures: i) Centralized ii) Decentralized iii) Hybrid

go undetected. This approach is very effective against known botnets, but on the other hand are not very useful for unknown bots and are in actuality more susceptible to evasion techniques.

Detection based on bot behavior involves describing a model of how botnets generally operate. The generality of this approach makes it possible to capture new or unseen bots. However, if the model becomes too general the false positive rate may become high. In behavioral approaches, researchers make assumptions based on observations about core behaviors of botnets. For botnet detection, the main assumption across approaches is that bots operate in a cooperative manner, engaging in some form of group activity at varying stages of the botnet life cycle [7, 8, 13]. Whereas specific knowledge of a particular bot drives signature-based approaches, a clear definition of bots behavior is at the core of behavioral approaches.

4.2. Detection techniques

Anomaly-based detection techniques aims to detect bots based on abnormal network activities, such as ab-normally high traffic, high latency and unusual port activities. This is a definite decisive limitation for anomaly-based techniques considering that bots are inclined to employ normal protocols for C&C communi-cation. Anomaly-based techniques applies a behavioral approach to bot detection, thus, it is able to detect abnor-mal activities or behavior for unknown bots.

DNS-based detection techniques operate based on DNS information produced by botnets [8]. C&C communication channels are unique to bot malware; bots interact with C&C servers through these channels. To gain access to these servers, bots perform DNS queries. The aim of DNS-based approaches is to capture unusual DNS traffic in order to identify bots.

ML-based detection techniques have been considered to be the most effective at detecting botnets [1, 2, 3]. The basis for its effectiveness lies within its ability to identify bot related traffic within normal traffic [2]. This poses a challenge for other techniques as bots utilize normal protocols to mask C&C communication. However, ML-detection requires a sufficient amount of training examples and well-defined features to be optimally effective. As the focus of this paper, machine learning will be discussed in more detail in the following sections.

4.3. Scope of detection

Among the botnet detection approaches observed in the relevant literature, each had one goal regarding scope of detection, group activity or individual hosts.

The re-lationship between the scope of detection and the ML method chosen, demonstrate a great affinity for using un-supervised learning methods when the goal is group de-tection, and supervised when targeting individual hosts.

Detecting bots based on group activity assumes coordinated activities by bots in the same botnet. Distinguished by similar traffic patterns, the aim is to identify all the bots in the network based on their collective action rather that their individual operations. Unlike Group based detection, individual hosts are classified based on their individual actions and characteristics irrespective of the activity of the group they might be a part of.

5. Machine Learning

Machine learning (ML) is a branch of artificial intelli-gence that aims to develop systems with the ability to learn from past experience. In machine learning, data,(past experience), is given as input to a ML algorithm to derive patterns probable in order to create a model that represents the data. The main concern in this field is, How do we develop computer programs that auto-matically improve with experience? At the core of ML are statistical and computational principles derived from concepts that exist in many disciplines such as artifi-cial intelligence, philosophy, information theory, biol-ogy, cognitive science, computational complexity and control theory [16].

The aim of ML is to create a model based on the data given. This model describes the patterns that exist in the data, which should be able to make informed decisions given new (unseen) data.

5.1. Machine learning Features

For any machine learning task, the two most important decisions to address are deciding what features to use and which ML-Method (supervised, unsupervised) to select. The features selected will shape the type of model that is formed. Features are able to represent behaviors or tar-get specific characteristics. The ML-method chosen will impact how the model behaves, one method may create a model whose main concern is how different bots interact with each other while another model concerns itself with how individual bots operate.

Feature selection is the process of extracting the best subset of variables from all possible variables that most accurately represent the data. In botnet detection, the aim of the feature selection process is to select a subset of features that will best describe the behavior of bots or the specific bot being targeted. The features selected will depend on the type of data being used. Number of query lookup may be a feature from DNS data[8], Source and destination IP [15] for net flow data, checksum are features from packet top data. For ML-based detection most researchers chose net flow, (Traffic Flow) data. Traffic flow is an artificial logical equivalent to a call or connection as a sequence of packets sent from a particular unicast or multicast destination [20]. From this, flow-level features are derived. These features describe how each node on a particular network interact with other nodes. Examples of flow-level features are: flow duration, average byte per packet per flow, who indicated the connection (client or server). The features selected will support a particular approach. Flow-level features will support a behavioral approach, while packet level features that capture specific characteristics will support a signature-based approach.

The underlying hypothesis for ML-based botnet detection is, bots produce unique patterns hidden in network traffic or client machine activities. It is opinoined that on implementing some form of ML method, one may be able to uncover these patterns to successfully detect malicious activity.

6. Machine Learning Methods Used in Botnet Detection

In this section we will evaluate the role of each technique in bot detection based on how each has been used. Eval-uations of each technique will be separated as follows: firstly, an overview of the technique, and secondly, a brief look at how the technique has been used in the literature.

6.1. Supervised Learning (SL)

In supervised ML, models are built from labeled training data. The aim is to create a model (function h) that rep-resents the data, described by a function (h) that maps input variables x to their appropriate target y (fig 2). This function is sometimes referred to as the hypothesis $h(x)$.

There is also a distinction among supervised learning methods based on how the data is labeled. Supervised learning problems may be categorized as regression or classification. For regression problems, the labeled target values represent a range of values.

For classification problems, input variables are assigned to classes based on patterns represented in the data. Classification algorithms are concerned with the relationship between class label and input variables. Botnet detection is an example of such a problem, where we are trying to determine what class a packet or sequence of packet may be assigned to, i.e. Botnet or Not botnet traffic.

Table 1: Table I - Role of ML-Method in Botnet detection Systems

Detection System	ML Method	Scope of detection	Detection approach	True Positive Rate (TPR)	False Positive Rate (FPR)
David Zhao et al [15]	Supervised	Individual	Signature (P2P)	98%	2.3%
Carl Livadas et al [6]	Supervised	Individual	Signature (IRC)	NA	10% — 20%
Leyla Bilge et al [11]	Supervised	Individual	Signature (C&C Server)	87%	20%
F Sanchez et al [9]	Supervised	Individual	Signature	91%	0.56%
W. T Strayer et al [5]	Supervised	Individual	Signature (IRC)	NA	30%
Guofei Gu et al [7]	Unsupervised	Group	Behavior	99%	1%
Yu et al [13]	Unsupervised	Group	Behavior	100%	20%
Hyunsang Choi et al [8]	Unsupervised	Group	Behavior	95%	4%
Lei Zhang et al [14]	Unsupervised	Group	Signature	100%	0.2%
Wei Lu et al [12]	Unsupervised	Group	Signature	95%	NA

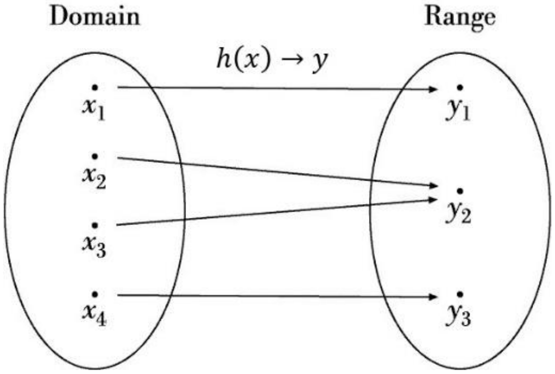

Figure 2. Showing relationship between ML domain *x* and targets *y*

6.2. The Role of Supervised Learning (SL)

In 2006, Livadas et al presented a network-based botnet detection approach based on supervised machine learn-ing techniques. The authors conducted an evaluation of three different machine learning methods for identifying IRC Botnets. Detection was carried out in two phases: the first phase classifies traffic based on IRC traffic and the second phase classifies IRC chat flows as bot-net or real chat flows.
Features Used:

1. Flow Duration
2. Maximum initial congestion window
3. Indicator of whether or not client or server initiated flow
4. Average byte per packet for flow
5. Average bits per second for flow
6. Average packets per second for flow
7. Percent of packets pushed in flow
8. Percent of packets in one of eight packet size bins
9. Variance of packet inter arrival time.
10. Variance of bytes per packet for flow

Strayer et al introduced an approach that targets IRC bots. This approach is broken into four stages. In the first stage, flows that are most likely to not have C&C data are filtered out based on knowledge of IRC bots, behavioral patterns and characteristics in flow. The second stage uses supervised learning to identify suspicious traffic flows. The third stage groups flows based on similar predefined characteristics. The groups then advance to the fourth stage which uses topological analysis to determine flows with the same controller. The flows with the same controller are finally examined to see if they are a part of a botnet or not.
Features Used:

1. Flow Start time
2. Flow end time
3. Flow protocol
4. Summary of TCP flags
5. Total number of packets exchanged in flow
6. Total number of bytes exchanged in flow
7. Total number of packets pushed in flow
8. Flow duration
9. Maximum congestion window size
10. Indicator of whether or not client or server initiated flow
11. Average bits per second for flow
12. Average packets per second for flow
13. Percent of packets pushed in flow
14. Percent of packets in one of eight packet size bins
15. Variance of packet inter arrival time.
16. Variance of bytes per packet for flow

Liao et al proposed a method that uses supervised learning techniques to identify P2P bots. The first stage of this two-stage approach involves feature extraction. In this stage, specific features which may be used to characterize P2P bots are extracted from the traffic flows. The features of these flows are passed to the second stage where supervised learning algorithm is used to classify each flow.
Features Used:

1. Indicator of synchronous session
2. Average number of bytes per flow
3. Average length of packet in flow
4. Standard deviation of number of bytes per flow.
5. Standard deviation of number of packets in flow.
6. Number of small sessions.
7. Percent of small packets
8. Average size of packets
9. Standard deviation of packet size
10. Average number of packets per flow
11. Percent of packets per flow
12. Number of small packets
13. Number of null packets

Shin et al introduced a bot detection system that classifies bots based on activities both on the network and the client's computer. This method has five modules (M1-M5) that correlate bot-related activities on the network and individual clients. The first module M1 is the human-process-network correlation analysis module. This module detects malicious processes by monitoring human process on the host relating to the keyboard and mouse and correlating them with network activity. The system checks the time difference between a process produced by a mouse click or keyboard event, the source of the event, and whether or not the process is running in the foreground at that time, is also taken into consideration. A small time difference may indicate that the process was generated by a human otherwise this process will be marked as suspicious and forwarded to the M2, M3 and M4. M2 and M3 uses supervised learning to classify queried domains names as malicious or benign and classify malicious behavior on host computers respectively. M4 monitors traffic generated by the suspicious process on the hosts network interface. Incoming packets and exchange rate between process and the remote site are compared. Once the exchange ratio is smaller than a predefined value, bot behavior is suspected. Finally, after each module makes its decision, the correlation engine - M5, combines the results to determine the final decision using a weighted voting scheme.
Features Used:

1. Time difference between current date and domain expiration date
2. Time difference between domain
3. Number of domain registration
4. Indicator of whether a target domain can be found on blacklist or not
5. Indicator if the domain names are contacted by the process
6. Indicator of the domain queried by the process frequently

6.3. Unsupervised Learning (UL)

Unsupervised Learning is the area of machine learning concerned with developing systems which can learn how to represent patterns in a data set based solely on input variables. The main aim of such a learner is to estab-lish a function to describe hidden patterns in unlabeled data. The absence of target values (y), or external en-vironmental evaluation, is what distinguishes unsuper-vised learning from supervised and reinforcement learn-ing [17]. The most common form of unsupervised learn-ing is called clustering. This is an unsupervised learn-ing technique used to find similarity in unlabeled data by grouping them in sections called clusters.

Given that all data points looks the same (unlabeled), the aim of a clustering algorithm is to understand the relationship between each data points and group them accordingly. In the same way, as it relates to botnet detection, clustering algorithms have been used to group traffic of similar characteristics in an effort to single out and identify traffic with malicious intent. The Botminer detection system [7] clusters similar communication traffic and similar malicious traffic and performs cross cluster correlation to identify the hosts that share both similar communication patterns and similar malicious activity patterns.

6.4. The Role of Unsupervised Learning (UL)

Yu et al. proposed a method for online detection using the k-means clustering algorithm to group bot related traffic. The approach uses network flow features [20] in prede-fined time windows. The aim is to group traffic based on similarity. The cluster with greater similarity than a pre-defined threshold will be classed as suspicious, thus, the host related to these flow will be flagged.
Features Used:

1. Number of bytes per packet for flow
2. Average bits per second for flow
3. Average packets per second per flow
4. Number of packets per flow

Chioi et al proposed a method for detecting bots based on how different host use DNS services. Bots use DNS to look up C&C servers and victims. The assumption by the researchers is that, bots which belong to the same botnet will use DNS services similarly. This method uses the X-means clustering algorithm to group domains that may be related to a botnet.
Features Used:

1. Number of domain tokens
2. Average length of domain tokens
3. Black listed 2nd level domains

4. Number of quires sent
5. Number of distinct sender IPs
6. Number of distinct sender autonomous system numbers (ASNs)
7. Query Type
8. Estimated similarity of a domain
9. Number of distinct resolved IPs
10. Number of distinct ASNs of resolved IPs
11. Number of distinct countries of resolved IPs
12. TTL Value in DNS answer

Zhang et al introduced a system for detecting botnets that identifies P2P botnets

in spite of the botnet being currently engaged in malicious activity. The emphasis of this method is to detect P2P bots by identifying C&C communication patterns that characterize P2P bots. The system first identifies P2P hosts then P2P bots among those hosts. This approach uses flow level features, the system presumes that P2P nodes create many failed outgoing flows. For each cluster of flows their destination IP is checked and for each IP their BGP prefix are checked. If the number of distinct BGP prefixes are smaller than a predefined amount, they are ignored. To differentiate legitimate P2P traffic from bot P2P connections, the authors assume that bots of the same botnet uses similar P2P protocol and network. Also they assume that pairs connect by two bots that have longer overlaps than that of legitimate P2P traffic.

1. Number of packets sent
2. Number of packets received
3. Number of bytes sent
4. Number of bytes received

Gu et al like most, assumes bot exhibit similar patterns in their traffic flows. Using the X-means clustering algorithm, the authors group flow with similar communication patterns. This detection approach has five components with three levels. The first level has the A and C-Plain monitors that monitors outgoing and internal traffic flows respectively. The second level is made up of the A and C-Plain clustering that clusters traffic, filtered by their respective monitors of the previous level. The results from these clusters are then passed to the third level, the cross- plain correlator, which makes the final decision about hosts that may be a part of a botnet. By combining the results from the A and C plain clusters.

W.Lu et al proposed a method that clusters flows based on similarities in payload. This method is split up into three sections, the first stage analyses feature, the second, clusters flows and the third, botnet decision. In the first stage, features are extracted from the flow payload in the time intervals as a 256-dimentional vector. In the second stage, flows are clustered using k-means and x-means clustering algorithm. These clusters are then passed to the third phase where the cluster with the lowest standard deviation is marked as botnet.

7. Data Preparation

The botnet dataset used was provided by the Canadian Institute for Cybersecurity [10]. The data came in the form of a pcap file the distribution of bots in the training set are as seen below.

7.1. Features

Features were selected for the creation of this modle based on the review of the literature, these features cap-ture the core of botnet activity from this particular per-spective (i.e flow) and therefore model botnet activity well. Top level features or characteristic were extracted using tshark commands such features include:

1. Source IP (ip.src)
2. Destination IP (ip.dst)
3. IP Length (Bytes) (ip.len)
4. TCP Push Flag (tcp.flags.push)
5. Protocol (ws.col.Protocol)
6. TCP Source Port (tcp.srcport)
7. TCP Destination Port (tcp.dstport)
8. UDP Source Port (udp.srcport)
9. UDP Destination Port (udp.dstport)

From these characteristics the following features were derived:

1. Average byte per packet per flow
2. Variance of bytes per packet per flow
3. Flow Protocol
4. Total number of packets exchanged in flow
5. Total number of bytes exchanged in flow
6. Total number of packets pushed in flow
7. % of packets pushed in flow
8. Source port
9. Destination port
10. Standard deviation of number of bytes per packet in flow

Feature : Average byte per packet per flow
Assumption : Bots usually act as a group
Category : Group Activity
ML Method : Unsupervised
This particular feature is geared towards modeling such behavior. Each packet has a particular amount of bytes, thus knowing the avg byte for each flow one will be able to group flows with similar this similarity.

Feature : Variance of bytes per packet per flow
Assumption : Bots usually act as a group
Category : Group Activity
ML Method : Unsupervised
This particular feature is geared towards modeling such behavior. Variance is a measure of how far a set of numbers are spread out from their mean. Thus a high variance will indicate normal activity and not a synchronized

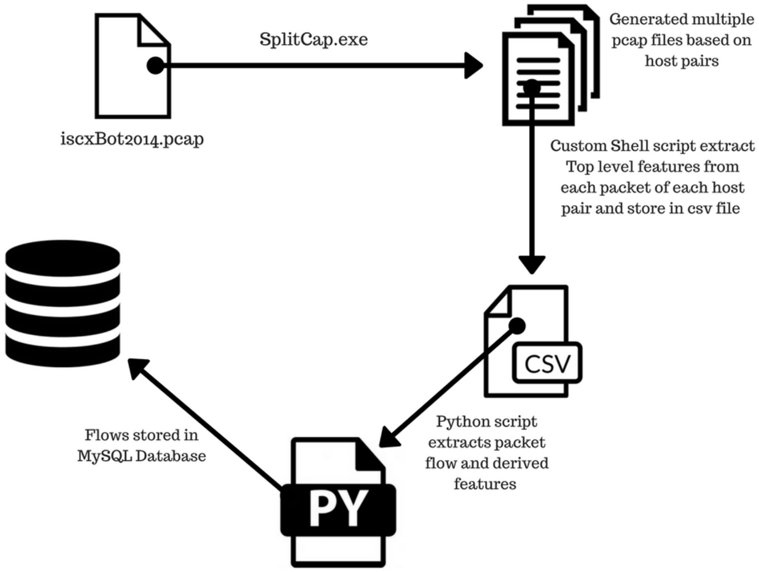
SplitCap.exe
Generated multiple pcap files based on host pairs
iscxBot2014.pcap
Custom Shell script extract Top level features from each packet of each host pair and store in csv file
CSV
Python script extracts packet flow and derived features
Flows stored in MySQL Database
PY

Figure 3. Showing Data Preparation steps

group bot activity but a low variance will indicate the opposite.

Feature : Flow Protocol
Assumption : bots use a particular protocol
Category : Individual Activity
ML Method : Supervised
This feature is used to target bots based on communication protocol, using this feature will improve the accuracy of a model if only bots that use a certain protocol is targeted, otherwise this may disturb the model.

Feature : Total number of packets exchanged in flow
Assumption : Bots usually act as a group
Category : Group Activity
ML Method : Unsupervised
During the attack phase or when bots receive instruction from the bot master their uniform activities may result in the number of packets sent in each flow being similar. The aim of this feature is to capture such behavior.

Feature : Total number of bytes exchanged in flow
Assumption : Bots usually act as a group
Category : Group Activity
ML Method : Unsupervised
similar to the feature above during the attack phase or when bots receive instruction from the bot master their uniform activities may result in the number of packets sent in each flow being similar.

Feature : Total number of packets pushed in flow
Assumption : bots use a particular protocol
Category : Individual Activity
ML Method : Supervised
This feature is specific to the transmission control protocol TCP. The socket that TCP makes available at the session level can be written to by the application with the option of ”pushing” data out immediately, rather than waiting for additional data to enter the buffer. This is specific to certain type of bot.

Feature : Percent of packets pushed in flow
Assumption : bots use a particular protocol
Category : Individual Activity
ML Method : Supervised
This is a supporting feature to the one above given a particular bot that uses TCP push packets. A high percentage of pushed packets will indicate a strong possibility of bot activity.

Feature : Source port
Assumption : bots use a particular protocol
Category : Individual Activity
ML Method : Supervised
Given knowledge of how bots operate, one my know what ports bots tend to use. With this information this feature would be quite significant to such a model as seen by the average drop in accuracy when this feature is removed when using supervised learning.

Feature : Destination port
Assumption : bots use a particular protocol
Category : Individual Activity
ML Method : Supervised

Given knowledge of how bots operate, one my know what ports bots tend to use. With this information this feature would be quite significant to such a model as seen by the average drop in accuracy when this feature is removed when using supervised learning.

Feature : Standard deviation of number of bytes per packet in flow
Assumption : bots use a particular protocol
Category : Individual Activity
ML Method : Supervised
This is a quantity calculated to indicate the extent of deviation for a group as a whole. A low deviation may signify bot activity.

Features were separated in three different sets as seen below:

1. Feature Set_1 — All features
2. Feature Set_2 — Features labeled Group Activity
3. Feature Set_3 — Features labeled Individual Activity

Group Activity:

1. Average byte per packet per flow
2. Variance of bytes per packet per flow
3. Total number of packets exchanged in flow
4. Total number of bytes exchanged in flow
5. Standard deviation of number of bytes per packet in flow

Individual Activity:

1. Flow Protocol
2. Total number of packets pushed in flow
3. % of packets pushed in flow
4. Source port
5. Destination port

1. Neris — IRC — 21159 (12%)
2. Rbot — IRC — 39316 (22%)
3. Virut — HTTP — 1638 (0.94 %)
4. NSIS — P2P — 4336 (2.48%)
5. SMTP Spam — P2P — 11296 (6.48%)
6. Zeus — P2P — 31 (0.01%)
7. Zeus control (C & C) — P2P — 20 (0.01%)

With the pcap file provided (isxBot2014.pcap) the following steps were taken to extract the flows and derive the necessary flow features. The splitCap program was used to split the original file into multiple pcap files based on ip host pairs and stored in a separate folder (capFiles). Each packet from each host pair was analyzed by a custom shell script (extract.sh) using tshark commands to to extract the top level features and then stored in a csv file. This csv file then served as input for a python script where flows were extracted, where a flow was defined as unique source destination ip and source destination ports. The data was then stored in a MYSQL database.

```
#!/bin/bash
capFiles='PcapFolder/*.pcap'

final='outFolder/Data.csv'

fields='-n -T fields -E separator=,
      -e ip.src
      -e ip.dst
      -e ip.len
      -e tcp.flags.push
      -e _ws.col.Protocol
      -e tcp.srcport
      -e tcp.dstport
      -e udp.srcport
      -e udp.dstport '

command='tshark '

for file in $capFiles
do
   echo ”processing file: $file”
   $command -r $file $fields >> $final
done
```

Above is the extract.sh scrip used to pull the top level features from the pcap files. from these features the selected features were calculated and used in the final dataset. For the python code below, First a list of distinct (source destination ip source destination port) was obtained. The first loop iterates through the list of flows. All packets in the file that is a part of the current flow is then put in another list the second loop iterates through this list. The flow features are calculated in the second loop and that flow is stored as a single instance in the database. Not all packets were used from the original pcap file, in all 500,803 packets were processed and 30,000 flows obtained for the taring set. Class distribution is shown in fig 4.

```
#Get List of all flows
#Initialize placeholders

# for each packet in list
for row in cur:

      source_ip = get_src_ip(row)
      dest_ip = get_dst_ip(row)
      src_port = get_src_port(row)
      dst_port = get_dst_port(row)

      #Reset place holders

      #Select all packets that are from
      the current flow
```

```
    #for each packet in current flow
    for row2 in cur2:
        #calculate features
        proto = get_protocol(row2)
        byte.append(get_ip_len(row2))
        pushed.append(get_tcp_push(row2))
        s_port = get_src_port_flow(row2)
        d_port = get_dst_port_flow(row2)
        flow_src_ip =
        source_ip = get_src_ip(row2)
        flow_dest_ip = get_dst_ip(row2)
        pack_count+=1

        avg_byte = np.mean(byte)
        var_byte = np.var(byte)
        flow_proto = proto
        pack_exc = pack_count
        byte_exc = sum(byte)
        pack_push = sum(pushed)
        flow_src_port = s_port
        flow_dst_port = d_port
        std_byte = np.std(byte)
        class_type =
        get_class(flow_src_ip,
        flow_dest_ip)

        if pack_count > 0:
          percent_push =
          (float(pack_push)
          /float(pack_count)) * 100.00
          #calculate percent_push

      #Store calculated features in DB

cur.close()
conn.close()
```

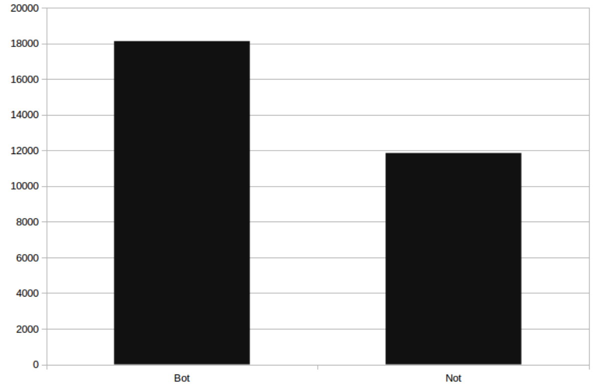

Figure 4. Showing Instance distribution in dataset

Table 2: Shows detection accuracy on ML Algorithms using different feature sets.

ML Method	Unsupervised	Supervised
ML Algorithm	K-Means	Naive Bayes
Feature *Set*$_1$	90%	90.775%
Feature *Set*$_2$	99%	60.205%
Feature *Set*$_3$	90%	93.735%

8. Conclusion

As bots became more threatening, research efforts in the area intensified, producing various methods of detecting and defending against botnets. To date, ML-based detec-tion methods have proven to be quite effective, though not without their limitations. Timely detection, real-time monitoring and adaptability to new threats are issues still to be solved. The different ML methods have different strengths and weaknesses as seen in the role they play in bot detection. The statistical foundation of SL meth-ods (i.e. the hypothesis representation), concerns its self with the relationship between the features (x) and target (y). In order to accurately represent the behavior of bots using SL, this must be defined by the features selected and thus assumes some detailed knowledge about what this behavior looks like.

Based on the characteristics of SL researchers in this field has made use of the precision of SL methods to accurately identify bots based on some known and specific characteristics (features used in SL). The precision of SL can be quite effective against bot traffic that seek to camouflage itself among legitimate traffic, given some specific characteristics of the malicious traffic. In our survey of the SL techniques we have observed a common trend. Apart from specific insights about bot traffic revealed in the feature space, SL methods perform very poorly. SL methods may overcome the camouflaged nature of bots. As seen in Table I, supervised learning methods are employed for cases where some specific characteristic is known.

Unsupervised learning methods are mainly used to target behavioral patterns not unique to any type of bot. The aim of research that used unsupervised methods is to capture group activity by bots in a botnet. Unsupervised learning in contrast to supervised learning has as its main concern, the relationship between samples. For this reason, it is able to recognize patterns that appears. Being so concerned with the similarity between samples, may cause a high rate of false positives as bots try to camouflage their activities. This issue however has been dealt with by some researchers [7], by representing specific characteristics in the features space that shape how groups are formed.

Supervised learning algorithms are more focused and the relationship between each feature attribute and the class from a particular instance. Therefore the more precise the features are in describing the class the more accurate the predictions. On the other hand given that unsupervised learning methods do not have class labels, their focus is on the relation between features themselves. Therefore given precise attributes that may represent a small subset of the dataset may disrupt the grouping process and throw off the clusters. However given the attributes that describes the general behavior of the dataset (in this case bots) the clusters will thrive. Botnet detection is a multifaceted problem with multiple ways of detection at different stages of the botnet life-cycle. Understanding what tools to use at the appropriate time with the right set of features are the key to developing robust detection systems. Given the multi-perspective nature of bots our future work will be to use the multi-view based machine learning ensemble method Multi-perspective machine learning (MPML) [25] to develop an accurate and robust botnet detection system.

9. References

[1] Stevanovic, Matija, and Jens Myrup Pedersen. ”Machine learning for identifying botnet network traffic.” (2013).

[2] Maryam Feily , Alireza Shahrestani , Sureswaran Ramadass, A Survey of Botnet and Botnet Detection, Proceedings of the 2009 Third International Conference on Emerging Security Information, Systems and Technologies, p.268-273, June 18-23, 2009

[3] Thomas S. Hyslip,Jason M. Pittman,A Survey of Botnet Detection Techniques by Command and Control Infrastructure. 2015.

[4] M. Bailey, E. Cooke, F. Jahanian, Y. Xu, and M. Karir, A Survey of Botnet Technology and Defenses, Cybersecurity Applications & Technology Conference for Homeland Security, IEEE Computer Society, Los Alamitos, CA, 2009, pp. 299-304.

[5] W. T. Strayer, D. Lapsely, R. Walsh, C. Livadas, Botnet detection basedon network behaviour, in: W. Lee, C. Wang, D. Dagon (Eds.), Botnet Detection, Vol. 36 of Advances in Information Security, Springer, 2008, pp. 124.

[6] C. Livadas, R. Walsh, D. Lapsley, W. Strayer, Usilng machine learning technliques to identify botnet traffic, in: Local Computer Networks, Proceedings 2006 31st IEEE Conference on, 2006, pp. 967 974. doi:10.1109/LCN.2006.322210.

[7] G. Gu, R. Perdisci, J. Zhang, W. Lee, Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection,in: Proceedings of the 17th conference on Security symposium, 2008, pp. 139154.

[8] H. Choi, H. Lee, Identifying botnets by capturing group activities in DNS traffic, Journal of Computer Networks 56 (2011) 2033.

[9] F. Sanchez, Z. Duan, Y. Dong, Blocking spam by separating end-user machines from legitimate mail server machines, in: Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference, CEAS 11, ACM, New York, NY, USA, 2011, pp. 116124. doi:10.1145/2030376.2030390.

[10] Beigi, E.B., Jazi, H.H., Stakhanova, N. and Ghorbani, A.A., 2014, October. Towards effective feature selection in machine learning-based botnet detection approaches. In Communications and Network Security (CNS), 2014 IEEE Conference on (pp. 247-255). IEEE.

[11] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, C. Kruegel, Disclosure: detecting botnet command and control servers through large-scale netflow analysis, in: Proceedings of the 28th Annual Computer Security Applications Conference, ACSAC 12, ACM, New York, NY, USA, 2012, pp. 129138. doi:10.1145/2420950.2420969

[12] W. Lu, G. Rammidi, A. A. Ghorbani, Clustering botnet communication traffic based on n-gram feature selection, Computer Communications 34 (2011) 502514.

[13] X. Yu, X. Dong, G. Yu, Y. Qin, D. Yue, Data-adaptive clustering analysis for online botnet detection, in: Computational Science and Optimization (CSO), 2010 Third International Joint Conference on, Vol. 1, 2010, pp. 456 460. doi:10.1109/CSO.2010.214.

[14] J. Zhang, R. Perdisci, W. Lee, U. Sarfraz, X. Luo, Detecting stealthy P2P botnets using statistical traffic fingerprints, in: 2011 IEEE/IFIP 41st International Conference on Dependable Systems and Networks (DSN), Hong Kong, IEEE/IFIP, 2011, pp. 121132.

[15] Zhao, David, Issa Traore, Ali Ghorbani, Bassam Sayed, Sherif Saad, and Wei Lu. ”Peer to peer botnet detection based on flow intervals.” In Information Security and Privacy Research, pp. 87-102. Springer Berlin Heidelberg, 2012.

[16] T. M. Mitchell, Machine Learning, 1st Edition, McGraw-Hill, Inc., New York, NY, USA, 1997.

[17] Dayan, Peter. ”Unsupervised learning.” The MIT encyclopedia of the cognitive sciences (1999).

[18] Vania, Jignesh, Arvind Meniya, and H. B. Jethva. ”A review on botnet and detection technique.” Int J Comput Trends Technol 4, no. 1 (2013): 23-29.

[19] Leonard, Justin, Shouhuai Xu, and Ravi Sandhu. ”A framework for understanding botnets.” In Availability, Reliability and Security, 2009. ARES’09. International Conference on, pp. 917-922. IEEE, 2009.

[20] Brownlee, N., C. Mills, and G. Ruth. ”RFC2722.” Traffic Flow Measurement: Architecture (1999).