Detecting bots using multi-level traffic analysis

Matija Stevanovic and Jens Myrup Pedersen

Department of Electronic Systems, Aalborg University
Aalborg, Denmark

ABSTRACT

Botnets, as networks of compromised “zombie” computers, represent one of the most serious security threats on the Internet today. This paper explores how machines compromised with bot malware can be identified at local and enterprise networks in accurate and time-efficient manner. The paper introduces a novel multi-level botnet detection approach that performs network traffic analysis of three protocols widely considered as the main carriers of botnet Command and Control (C&C) and attack traffic, i.e. TCP, UDP and DNS. The proposed method relies on supervised machine learning for identifying patterns of botnet network traffic. The method has been evaluated through a series of experiments using traffic traces originating from 40 different bot samples and diverse benign applications. The evaluation indicates accurate and time-efficient classification of botnet traffic for all the three protocols as well as promising performance of identifying potentially compromised machines. The future work will be devoted to the optimization of traffic analysis and correlation of findings from three analysis levels in order to increase the accuracy of identifying compromised clients within the network.

Keywords: Botnet, Botnet Detection, Traffic Analysis, Traffic Classification, MLAs, Random Forests, Client analysis.

1. INTRODUCTION

Botnets are one of the most serious threats to Internet security and one of the most challenging topics within the field of network security today. Botnets represent a usually large collections of computers compromised with a sophisticated malware that puts them under the control of a remote attacker (Hogben et al., 2011). The compromised computers are often referred to as “bots” while the attacker is referred to as the “botmaster”. Contrary to other more conventional malware types, such as viruses, trojans and worms, botnet malware has an advantage of being able to communicate with an attacker through a Command and Control (C&C) communication channel. Botnets deploy C&C channel using a variety of communication protocols, such as: IRC, HTTP/HTTPS and P2P protocols. Additionally, modern botnets use many resilience techniques that make C&C channel more resilient to detection such as encryption, protocol obfuscation, Fast-flux and DGA (Domain Generation Algorithm) (Silva et al., 2013). Using the C&C channel, the botmaster can remotely control the behaviour of bots, turning them into highly distributed platform for the implementation of a wide range of malicious and illegal activities, such as: sending SPAM e-mails, Distributed Denial of Service (DDoS) attacks, information theft and malware distribution.

As botmasters are relying on network traffic for the communication with bots and the implementation of attack campaigns, many detection approaches are targeting botnets using network traffic analysis. During the last decade an abundance of detection approaches have been proposed relying on diverse principles of network traffic analysis (Silva et al., 2013; García et al., 2014a). One of the latest classes of detection approaches employs machine learning algorithms (MLAs) for identifying anomalous botnet traffic (Stevanovic et al., 2016). These approaches are often seen as the state-of-the-art detection approaches as they promise accurate and automatized detection of botnet traffic patterns. The contemporary machine learning-based detection approaches are using different MLAs most commonly supervised MLAs for classifying network traffic as malicious or benign. These approaches target botnets at different points in the network, they are based on different principles of traffic analysis and they are developed and evaluated using diverse traffic data sets (Stevanovic et al., 2016). In this paper we extend our previous work on network traffic classification for botnet detection (Stevanovic et al., 2015) in order to develop accurate and robust detection of compromised clients based on network traffic classification. Our goal is to develop a detection method that will provide identification of potentially compromised client machines while minimizing the number of false positives thus limiting the need for extensive operators’ involvement in the process of evaluating raised alarms.

We propose a novel multi-level botnet detection method by relying on three traffic classification methods targeting three protocols widely considered as the carriers of botnet network activity namely TCP, UDP and DNS. The proposed method is developed to address some of the pitfalls of using network traffic classification for botnet detection. First, we use supervised machine learning as the algorithm of traffic analysis that can provide automatized detection of botnet traffic by inferring the knowledge about the botnet traffic patterns from already available network traces. We rely on Random Forests classifier for providing accurate classification of botnet traffic. Second, the proposed methods target TCP, UDP and DNS as the main carriers of botnet C&C communication and attack traffic. Contrary to some of the existing work we develop a classifier for each of the protocols in order to obtain more precise analysis and ultimately more accurate classification. Third, we propose the use of novel feature sets for representing traffic instances within the classifiers. The traffic features are carefully chosen in order to capture the main traits of botnet network activity. Fourth, we evaluate the proposed method using one of the most comprehensive data sets of botnet network traces thus providing a thorough evaluation of classification performance and the capabilities of identifying compromised clients. Finally, we target bots at local and enterprise networks as we are able to obtain reliable training data on botnet traffic by relying on honeypots and malware testing environments.

The rest of the paper is organized as follows. Section 2 presents an overview of related work. Section 3 introduces multi-level botnet detection method based on TCP, UDP and DNS traffic analysis. Section 4 presents the results of performance evaluation for the proposed botnet detection method. Section 5 discusses presented results and possibilities for future work. Finally, Section 6 concludes the paper.

2. BACKGROUND

Botnet detection based on network traffic classification is one of the latest and the most promising classes of botnet detection approaches. The main assumption behind these approaches is that botnets create distinguishable traffic patterns that could be accurately detected using supervised MLAs. Over the last couple of years, a number of detection approaches that rely on traffic classification have been proposed (Stevanovic et al., 2016). Some of the most prominent approaches were proposed by Strayer et al. (2008), Masud et al. (2008), Saad et al. (2011), Zhao et al. (2013), Shin et al. (2012), Bilge et al. (2012, 2014), Perdisci et al. (2012), Haddadi et al. (2014) and Antonakakis et al. (2011).

Based on the point of traffic monitoring contemporary detection methods can be coarsely classified as ones implemented closer to client machines (Strayer et al., 2008; Masud et al., 2008; Saad et al., 2011; Zhao et al., 2013; Shin et al., 2012; Haddadi et al., 2014) and ones implemented further away from clients in higher network tiers (Bilge et al., 2012; Bilge et al., 2014;

Antonakakis et al., 2011; Perdisci et al., 2012). Detection approaches that analyse traffic further away from clients are able to capture some of the fundamental properties of botnet operation such as group behaviour and synchronicity of compromised machines. However, this scenario also has a number of limitations such as difficulty of processing high volume of traffic and identifying compromised clients due to the use of NAT (Network Address Translation). As a result, approaches implemented in the higher network tiers usually target either sampled traffic such as NetFlow (Bilge et al., 2012) or DNS traffic that represent only a fraction of total traffic (Bilge et al., 2014; Antonakakis et al., 2011; Perdisci et al., 2012). In contrast detection approaches implemented closer to client machines commonly process smaller amount of traffic often providing more detailed traffic analysis that can capture finite patterns of botnet network activity.

The contemporary detection approaches employ diverse principles of traffic analysis thus having different detection scope and capabilities. Network traffic is commonly analysed by observing traffic “flows” that encompass both TCP and UDP traffic (Strayer et al., 2008; Masud et al., 2008; Saad et al., 2011; Zhao et al., 2013; Haddadi et al., 2014). Other approaches target DNS traffic by analysing it either between local client and resolver (Shin et al., 2012) or above the resolver in upper DNS hierarchies (Bilge et al., 2014; Antonakakis et al., 2011; Perdisci et al., 2012). DNS traffic is analysed using different perspectives where some approaches classify Fully Qualified Domain Names (FQDNs) based on the features extracted from DNS queries and responses (Bilge et al., 2014; Antonakakis et al., 2011) while others classify domain clusters (Perdisci et al., 2012). Regarding the traffic features used for representing TCP and UDP flows some authors such as Masud et al. (2008) use features dependent on content of packet payload thus violating integrity of end users’ data and being vulnerable to payload encryption. Other approaches (Masud et al., 2008 and Shin et al., 2012) use features extracted from the client machines thus requiring the access to client machines under monitoring. Some authors (Saad et al., 2011; Zhao et al., 2013) consider using IP addresses as features thus violating end users’ privacy and potentially introducing bias in the data set which could lead to over-optimistic performance. Different approaches target different botnet network activities where some approaches (Strayer et al., 2008; Masud et al., 2008; Saad et al., 2011; Bilge et al., 2012; Bilge et al., 2014; Antonakakis et al., 2011; Perdisci et al., 2012; Haddadi et al., 2014) identify C&C communication, while others cover all botnet network activity. Finally, the approaches used various supervised MLAs for the classification task, while for the majority of the approaches tree classifiers have shown the best performance (Zhao et al., 2013; Bilge et al., 2012; Bilge et al., 2014; Antonakakis et al., 2011; Perdisci et al., 2012; Stevanovic et al., 2014, Haddadi et al., 2014).

The existing botnet detection methods are developed and evaluated using various data sets of botnet and benign traffic (Stevanovic et al., 2016). The used data sets are often sparse consisting of only a handful of botnet traces that are obtained in not-transparent way. Furthermore, the approaches often rely on data sets that are artificially formed by overlaying and merging data sets recorded at different monitoring points in network. Finally, some approaches use third party labelling solutions for forming the “ground truth” on botnet traffic thus putting the reliability of the training data in question.

In this paper we introduce a novel multi-level botnet detection approach that is able to identify compromised machines at local and enterprise networks. The method builds on our previous work on botnet traffic classification (Stevanovic et al., 2015) by further developing TCP, UDP and DNS traffic classifiers and introducing client analysis entity that is able to pinpoint malicious clients based on results of the three classifiers. The proposed approach analyses TCP, UDP and DNS traffic separately in order to provide more accurate detection. We believe that due to the different nature of TCP and UDP (first connection-oriented and second connectionless) they should be classified using separate classifiers where additional traffic features for TCP traffic would be used. Furthermore, we believe that DNS traffic analysis is crucial as many botnets rely on it for discovering addresses of C&C infrastructure or victims of attack campaigns. The three methods are based on Random Forests classifier as a capable ensemble classifier. We choose Random Forests classifier based on good performance in classifying botnet traffic reported by several studies (Antonakakis et al., 2011; Perdisci et al., 2012) and confirmed by our previous work on traffic classification for botnet detection (Stevanovic et al., 2015). For the realization of the classifiers we defined traffic features that should successfully capture the traits of botnet network activity. For TCP and UDP we rely only on features extracted from the packets headers without using IP addresses as features. This way we avoid violating privacy of the end users and over optimistic classification performance due to the bias introduced by using IP addresses as features. For DNS traffic analysis we classify FQDNs based on the features extracted from DNS queries and responses. The proposed methods do not make any assumptions about the botnet traffic, observing both C&C and attack traffic. Finally, for the development and evaluation of the detection methods we use data obtained by several honeypots and malware testing environments thus having greater confidence in obtained data sets, and the ground truth on botnet traffic used in our work. For the evaluation we use traces from 40 botnets and numerous benign applications. Comparing to other methods (Stevanovic et al., 2016) we use one of the most extensive botnet data sets which should contribute to higher confidence in reported classification performance.

3. DETECTION METHOD

In order to identify bots at local and enterprise networks we propose a multi-level detection approach that identifies compromised client machines by classifying network traffic as malicious or benign using supervised machine learning, as illustrated in Figure 1.

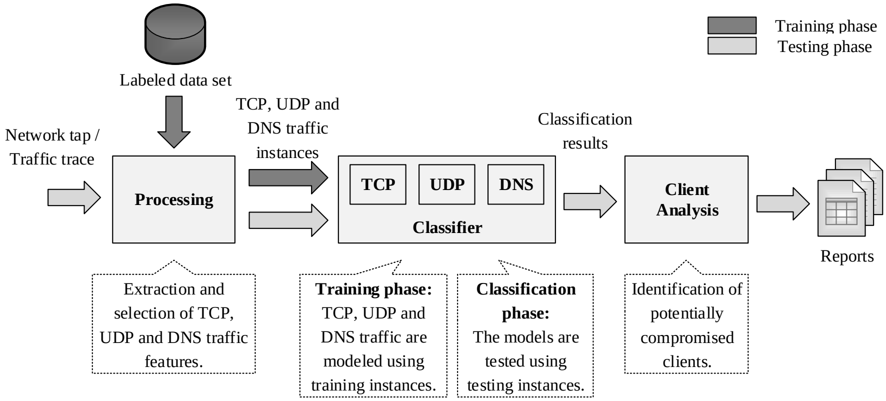
Training phase
Testing phase
Labeled data set
Network tap / Traffic trace
Processing
TCP, UDP and DNS traffic instances
TCP
UDP
DNS
Classifier
Classification results
Client Analysis
Reports
Extraction and selection of TCP, UDP and DNS traffic features.
Training phase: TCP, UDP and DNS traffic are modeled using training instances.
Classification phase: The models are tested using testing instances.
Identification of potentially compromised clients.

Figure 1. A botnet detection method based on multi-level traffic analysis

The system analyses network traffic on three levels by analysing TCP, UDP and DNS traffic produced by monitored clients. The system consists of three main components: Processing entity, Classifier entity and Client Analysis entity. The first entity performs processing of network traffic observed from either live network or existing traffic trace. This entity processes traffic so traffic instances for three analysis levels are extracted and characterized with a set of statistical features. The extracted traffic instances are then enriched using GEO and WHOIS information. The second entity is the Classifier that is in charge of building the model of malicious and benign traffic using training data and the classification of newly observed traffic instances. The third and the final entity of the system is Client Analysis entity that performs client analysis by correlating classification results from three level of analysis in order to generate report on potentially compromised clients within the monitored network. As the detection approach relies on supervised machine learning it operates in two phases i.e. training and test phase. During the training phase traffic models are trained using a labelled training data while in the test phase the previously generated models are tested by unlabelled test data.

The proposed method processes traffic in time windows, where at the end of each consequent time window traffic instances for TCP, UDP and DNS traffic are extracted. This way the normalization of traffic is performed by taking “snapshots” of traffic which results in the possibility of using diverse network traffic traces for development and evaluation of the method. The method observes traffic between local and remote clients (R2L, L2R) as well as local to local traffic (L2L), while all multicast and link-logical traffic are discarded.

3.1. TCP and UDP traffic analysis

TCP and UDP traffic are analysed from the perspective of bidirectional transport layer conversations that are defined as traffic exchanged between source and destination IP addresses on certain source and destination ports. For each TCP and UDP conversation we extract a set of statistical features that capture botnet traffic heuristics. Furthermore, we perform enrichment of the extracted features using external GEO location services. The features extracted for TCP and UDP conversations are presented by Table 1. It should be noted that UDP traffic analysis is realized by omitting UDP conversation that facilitate DNS traffic (i.e. UDP port 53). This is done as both malicious and benign DNS query-response pairs correspond to UDP conversations with similar characteristics.

For UDP conversations we extract a series of traffic features that can be divided in four groups i.e. Basic conversation features, Geographical features, Time-based features and Bidirectional features (25 features in total). For TCP conversation in addition to the four groups of features we also extract TCP specific features (18 features in total). Basic conversation features cover the basic statistics of TCP/UDP conversations. These features are able to capture the traffic that uses unusual ports associated with P2P communication. Furthermore, they can capture heavy traffic and brute force attacks by considering the number of packets and their size. Geographical features indicate geographical locations for remote IPs contacted by local machines. This features should capture tendencies of some countries to be more often associated with cyber-criminal than others. Time-based features describe the rate of transferring the data thus being able to describe the brute force attacks as well as the periodicity of botnet traffic. Bidirectional features take in consideration differences in communication between the two directions of communication indicating any unbalanced communication that can usually be associated with the attacks and communication between the botmaster and bots. TCP specific features capture events in regards to establishing and maintaining TCP conversations. Keeping the track of these events can indicate suspiciously high number of unsuccessful TCP attempts that usually characterize botnets communication due to often interrupted and unavailable botnet infrastructure. Also these features are able to capture TCP-based brute force attacks such as SYN floods, SYN ACK floods, ACK floods, ACK PUSH floods and RST and FIN floods by keeping track of TCP flags distribution.

Table 1. TCP/UDP traffic analysis: the list of features extracted for TCP and UDP conversations.

Feature	Type	Number[1]
Basic conversation features		
Port number	Numerical	2
Layer 7 protocol	Categorical	1
Duration (last pkt - first pkt)	Numerical	1
Total number of packets	Numerical	2
Total number of Bytes	Numerical	2
Mean of the number of Bytes per packet	Numerical	2
Std of the number of Bytes per packet	Numerical	2
Geographical features		
Remote IP country	Categorical	1
Remote IP continent	Categorical	1
Time-based features		
Number of packets per second	Numerical	2
Number of Bytes per second	Numerical	2
Mean of packets inter-arrival time	Numerical	2
Std of packets inter-arrival time	Numerical	2
Bidirectional features		
Ratio of number of packets OUT/IN	Numerical	1
Ratio of number of Bytes OUT/IN	Numerical	1
Ratio of inter-arrival times OUT/IN	Numerical	1
TCP specific features		
Number of three way handshakes	Numerical	1
Number of connection tear downs	Numerical	1
Number of complete conversation	Numerical	1
Average conversation duration	Numerical	1
TCP Flags	Categorical	2
Percentage of TCP SYN packets	Numerical	2
Percentage of TCP SYN ACK packets	Numerical	2
Percentage of TCP ACK packets	Numerical	2
Percentage of TCP ACK PUSH packets	Numerical	2
Percentage of TCP FIN packets	Numerical	2
Percentage of TCP RST packets	Numerical	2

3.2. DNS traffic analysis

DNS traffic analysis is implemented by observing DNS query-response pairs for queried FQDNs. For each of the queried FQDN we extract a number of statistical features and after the enriching process using GEO and WHOIS services 37 features are selected, as presented in Table 2. The selected features belong to four groups i.e. FQDN-based features, Query-based features, Response-based features and Geographical location features.

[1] *Some features are calculated for both directions of the conversation while others are unique for the particular conversation.*

Table 2. DNS traffic analysis: features extracted for DNS query-response pairs.

Feature	**Type**
FQDN-based features	
Number of tokens	Numerical
Avg length of token	Numerical
Length of SLD (Second Level Domain)	Numerical
Length of Domain	Numerical
Language of SLD	Categorical
Entropy (range of characters) for SLD	Numerical
Distance from n-grams of legitimate domains (`alexa.com`) for SLD	Numerical
Distance from n-grams of dictionary words domains for SLD	Numerical
Number of dictionary words in SLD	Numerical
Ratio of numerical characters in SLD	Numerical
Ratio of vowels in SLD	Numerical
Ratio of consonants in SLD	Numerical
Number of dictionary words in domain	Numerical
Ratio of numerical characters in domain	Numerical
Ratio of vowels in domain	Numerical
Ratio of consonants in domain	Numerical
Query-based features	
Type of query	Categorical
Number of queries	Numerical
Mean of query length	Numerical
Std of query length	Numerical
Mean of queries inter-arrival time	Numerical
Std of queries inter-arrival time	Numerical
Response-based features	
Number of query responses	Numerical
Mean of query response length	Numerical
Std of query response length	Numerical
Mean of query responses inter-arrival time	Numerical
Std of query response inter-arrival time	Numerical
Number of NOERROR responses	Numerical
Number of NXDOMAIN responses	Numerical
Avg number of answers	Numerical
Avg number of authority answers	Numerical
Avg number of additional answers	Numerical
Avg number of resolved IPs	Numerical
Mean of the value of TTL (Time-To-Live) field	Numerical
Std of the value of TTL field	Numerical
Geographical features	
Number of countries resolved IPs belong to	Numerical
Number of ASs resolved IPs belong to	Numerical

FQDN-based features quantify lexical properties of domain names in order to differentiate between human-memorable domains and “unusual” domains such as pseudorandom domains that commonly characterize Domain-Flux (DGA). Query-based features describe the way how the FQDN was queried capturing any irregularities such as high number of queries and periodicity of querying certain domain. Response-based features capture characteristics of the query responses. These features cover a number of botnet DNS characteristics such as the number of NXDOMAIN responses that can indicate botnet domains that have been taken down, the value of TTL field that can characterize Fast-flux and etc. Finally, Geographical location features capture the characteristics of IPs resolved for queried FQDNs. These features can indicate if the IPs are hosted over a high number of countries or Autonomous Systems (ASs) which is often associated with malicious hosting strategies such as Fast-flux.

3.3. Classifier entity

For the task of classifying traffic within all three traffic analysis methods we use Random Forests classifier (Breiman, 2001). Random Forests represent an ensemble learning method used for classification, that operate by constructing a multitude of decision trees at training time and outputting the class that is the result of majority vote from the individual trees. Random Forests are developed in order to correct overfitting as the common drawback of decision trees. The method combines “bagging” concept and the random selection of features, in order to construct a collection of decision trees with controlled variance. For the implementation of the classifiers within the proposed traffic analysis methods we use 10 trees where at each node $log_2^n + 1$ features, where n is the total number of features, that are used for growing the tree.

3.4. Client analysis entity

Client analysis entity has a goal of identifying potentially compromised computers based on the results of the three classifiers. This entity correlates findings of the three classifiers in order to provide a report on potentially compromised clients within the monitored network. The client analysis operates as illustrated in the Figure 2. Traffic instances identified as malicious by TCP, UDP and DNS classifiers i.e. alerts are filtered in order to eliminate false positives. The filtering is done by analysing characteristics of alerts based on IP and domain whitelisting and geographical analysis. Filtered alerts are then forwarded to a simple decision making process where client is deemed malicious if there is at least one alert for it.

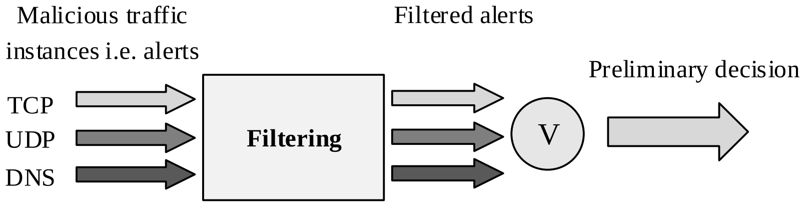

Figure 2. Client analysis: Whitelisting and Geolocation analysis.

For TCP and UDP traffic, the filtering of alerts is based on the Autonomous Systems (AS) to which remote IPs belong to. The implemented whitelisting excludes all conversations that involve remote IPs that are hosted by trustworthy providers such as Microsoft, Oracle, Apple, etc., as these providers often have tough hosting policies and there is much smaller chance that they would be used for hosting C&C infrastructure. After performing the filtering based on AS, we filter clients for which alerts include conversations to remote IPs hosted in less than 2 counties. This is done as C&C infrastructure is commonly hosted using IP addresses located in different countries, in order to achieve resiliency.

For DNS traffic we rely on similar process of filtering alerts. First, DNS alerts are filtered using domain whitelist. For domain whitelisting we use the first 10,000 most popular domains according to `alexa.com`. Remaining alerts are then scrutinized based on the number of countries hosting the domain and the number of NXDOMAIN responses. We filter out clients whose alerts include domains that are on average hosted by less than 2 countries or for which at all responses are of NOERROR type. The reasoning behind this decision process is that cyber-criminals often rely on Fast-flux hosting strategies that use many IP addresses typically distributed over the world. Furthermore, botnets are commonly associated with high number of unsuccessful queries due to C&C infrastructure often being taken down.

As already mentioned the proposed method will mark client malicious if there is at least one alert after the filtering. Thereby, in order for client to be identified as malicious the approach needs to generate at least one alert irrelevant of type. This is because not all client would produce TCP, UDP and DNS traffic. The preliminary decision formed this way is presented to operator together with a report for each of the potentially compromised clients indicating a number of TCP and UDP conversations and DNS query-response pairs that are classified as malicious for the particular client machine. In addition, the method also provides overview of the basic characteristics of traffic instances marked as malicious. The generated reports are presented to the operator in order to make a final evaluation of the preliminary conclusions generated by the approach.

4. EXPERIMENTS AND DETECTION RESULTS

In this section, we evaluate the performance of the proposed detection method by analysing the performance of classifying TCP, UDP and DNS traffic and capabilities of identifying potentially compromised machines. The presented analysis methods are fully implemented in Python, by relying on **`scikit-learn`** Python machine learning library for implementing Random Forests classifiers. The experiments are done off-line using pre-recorded data sets consisting of a number of malicious and benign traffic traces in the form of **`pcap`** files. All experiments were done using an off-the-shelf computer with Intel Core i7 at 3.4 GHz and 16GB of RAM memory. It should be noted that during the operation the method did not use more than 8GB of RAM. Finally, current implementation assumes IPv4 traffic but it should be noted that there are no obstacles in using the method on IPv6 traffic.

4.1. Data sets

For the evaluation, we use several malicious and benign traffic data sets. Benign data sets present traffic traces recorded at several LAN environments, while malicious data sets include traffic traces recorded by several honeypots and malware testing environments.

Benign data sets used for evaluation:

- **UPC data set** (Bujlow et al., 2013) - represents benign traffic generated for the purpose of evaluating DPI tools. The traffic is recorded at small local network consisting of 3 machines over the course of 2.5 months. Data set includes traffic from various benign applications such as web browsing, torrent clients, FTP clients etc.

- **ISCX data set** (Shiravi et al., 2013) - represents data set that is generated in order to evaluate intrusion detection systems (IDS). The data set represent 7 days of trace from specially deployed network environment with 20 client machines. From this data set we use benign traffic from the first, the fifth and the sixth day of the trace.

Malicious data sets used for evaluation:

- **ISOT data set** (Saad et al., 2013) - 5 network traces obtained from 3 different P2P botnets. These traces were obtained by French chapter of Honeypot project and they include traffic that covers both C&C communication and attack campaigns.

- **ISCX data set** (Shiravi et al., 2013) - From this data set we used trace of traffic produced by ISCX IRC botnet. The trace includes both C&C communication and attack phase of botnet operation.
- **MCFP data set** (García et al., 2014a and García et al., 2016) - 15 traces. The data set covers traffic produced by executing different malware for an extended period of time. The traces cover both C&C communication and botnet attack campaigns.
- **Contagio data set** (Parkour, 2015) - 14 traces. The data set covers traffic produced by executing different bot malware in a malware testing environment. The traces primarily cover initial bootstrapping procedure, while some of them also include other phases of the botnet operation.
- **HoneyJar data set** - 5 traces generated using HoneyJar (Pedersen et al., 2015) - a malware testing environment deployed by researchers at Aalborg University with a goal of secure, automated and contained experimenting with malware. The traces primarily cover initial bootstrapping procedure performed by bots.

The benign and malicious data sets are summarized in Tables 3 and 4, respectively. The presented numbers of TCP, UDP conversations and DNS query-response pairs are taken over the total duration of the traces.

Table 3. The summary of benign traffic data sets.

Traffic trace	**Number of packets**	**TCP conversations**	**UDP conversations**	**DNS queries**	**Duration**
UPC trace #1	11M	62k	56k	7k	~ 60 days
UPC trace #2	21M	107k	65k	12k	~ 30 days
UPC trace #3	7M	74k	146k	8k	~ 60 days
ISCX trace #1	22M	218k	59k	24k	1 day
ISCX trace #2	25M	257k	74k	25k	1 day
ISCX trace #3	25M	271k	61k	27k	1 day
Total	111M	989k	461k	103k	

Table 4. The summary of botnet traffic data sets.

Traffic trace[2]	**Number of packets**	**Number of TCP conversations**	**Number of UDP conversations**	**Number of DNS queries**	**Duration (seconds)**
Storm SMTP (i)	156,980	4,667	0	7,285	3,115
Storm UDP (i)	368,776	0	6,923	22,572	1,851
Waledac (i)	213,095	4,927	3,144	6,602	760
Zeus (i)	1,215	10	0	0	38,809
ZeusCnC (i)	1,632	18	0	0	310
IRC botnet (is)	9,945,235	22,501	0	1	85,818
Botnet 42 (m)	321,481	11,686	15	7,612	17,269
Botnet 43 (m)	175,491	19,447	24	196	12,342
Botnet 44 (m)	476,404	31,364	157	8	241,875
Botnet 45 (m)	256,492	153	13	2	3,971
Botnet 46 (m)	45,760	852	4	29	1,216
Botnet 47 (m)	24,387	4,507	8	5	7,218
Botnet 48 (m)	20,604	40	4	17	946
Botnet 49 (m)	80,951	8,381	287	22	69,732
Botnet 51 (m)	2,123,888	75,936	92	75,224	10,003
Botnet 52 (m)	1,217	1,217	219	9	532
Botnet 53 (m)	351,211	370	5,678	190	3,824
Botnet 54 (m)	437,217	30,320	40	2,343	58,686
Botnet 90 (m)	167,302	18,833	3	3	40,350
Botnet 91 (m)	197,370	21,161	265	3,469	1,236
Botnet 92 (m)	144,937	17,313	4	5	36,693
Blackhole (c)	6,251	63	56	12	663
Cutwail (c)	8,515	125	4	88	75
Pushdo (c)	24,180	1,958	3	183	1,991
Dirtjumper (c)	19,063	2,760	2	4	241
Styx (c)	10,658	33	0	6	1,145
Kelihos (c)	13,926	1,835	2	0	1,092
Kuluoz (c)	179,577	15,509	19	1,356	1,197
Zeus (c)	1,739	21	72	3	275
Purplehaze (c)	324,709	7,090	2	786	2,194
Sality (c)	233,635	8,291	29	6,344	127,626
Tbot (c)	12,870	210	1	2	5,400
ZeroAccess (c)	2,514	47	278	26	388
Sirefef (c)	19,528	434	639	224	1,001
Gameover (c)	7,443	166	34	14	2,370
Agobot (h)	3,659	0	5	1	3,600
Batimal (h)	1,625	169	0	5	10,800
Mybot (h)	3,614	1,821	4	2	3,600
Palevo (h)	4,252	468	0	5	10,800
Shiz (h)	1,562	124	0	223	10,800
Total	~ 16M	~ 310M	~ 18k	~ 134k	

[2] *(m) - MCFP trace, (c) - Contagio trace, (h) - HoneyJar trace, (i) - ISOT trace, (is) - ISCX trace*

4.2. Evaluation procedure

As already elaborated in the previous section, the main goal of the experiments is to evaluate the performance of traffic classification performed by different levels of traffic analysis and to evaluate the capabilities of identifying compromised clients using the proposed method.

4.2.1. Evaluation of traffic classification on different levels of traffic analysis

We evaluate performance of classifying malicious and benign traffic using “batch” analysis where we use all available data sets and evaluate the performance of classification using 10-fold cross validation scheme. For the classification of TCP and UDP conversations we vary the length of time window and the number of packets processed per conversation, in order to find out the “optimal” value of the two parameters. Similarly, for the DNS traffic analysis we vary the length of time window examining influence of it on classification performance. We do not vary the number of processed DNS queries and responses within the time window as we would like to fully capture time relations of DNS queries.

The classification performance is characterized by performance metrics that described both accuracy and time requirements of traffic classification. The accuracy is expressed by following metrics:

- **Precision:** $precision = \frac{TP}{TP+FP}$
- **Recall:** $recall = \frac{TP}{TP+FN}$

Where TP, FP and FN are number of true positives, false positives and false negatives, respectively. Time requirements of classification are quantified by time used for the training and the testing i.e. classification.

4.2.2. Evaluation of capabilities of identifying potentially compromised clients

This scenario is evaluating performance of identifying compromised clients by the proposed detection system. The goal of this scenario is to illustrate the capabilities of the proposed approach to identify compromised machines. In order to do so we train the proposed approach with one part of the data set and test it against another disjoint data set. For training we used all data sets except for the Botnet 90 trace and ISCX day #1 data sets, as they will be used for testing. Botnet 90 data set is chosen as it is the trace that contains the highest number of compromised clients from all malicious traces considered i.e. 11 in total. ISCX day #1 on the other hand represent a significant non-malicious data set that captures traffic from more than 20 non-malicious clients. This evaluation scenario is realized for the optimal values of time window and the number of packets per conversation inferred from the evaluation of the three classifiers.

The performance of identifying malicious clients is characterized by number of true positives, false positives, true negatives, false negatives as well as false positive rate (FPR), where FPR is defined as follows:

- **False positive rate:** $FPR = \frac{FP}{FP+TN}$

4.3. Results of experiments

The results of the evaluation are presented by Figures 3-8 that illustrate the results of classifying TCP, UDP and DNS traffic and Tables 5-7 that illustrate the capabilities of identifying compromised machines.

4.3.1. TCP, UDP and DNS traffic classification

Figures 3 and 4 show the performance of TCP traffic classification while Figures 5 and 6 show the performance of UDP traffic classification. The results are produced by varying the length of time window and the number of packets processed per TCP/UDP conversation. Figures 7 and 8 illustrate the performance of DNS traffic classification, for various lengths of time window. Within the experiments we used 4 values of the time window length i.e. 300, 600, 1800 and 3600 seconds and 4 different values of the maximum number of packets per TCP/UDP conversation i.e. 10, 100, 1,000, 10,000 packets.

The results presented in Figure 3 indicate that for TCP traffic classification increasing number of packets processed per conversation and increasing the length of the time window enhance the classification performance. While increasing the length of time window brings modest improvements in classification performance, increase in the number of observed packets has a much bigger influence. Therefore, the number of packets analysed per conversation is crucial for improving the performance of TCP classification. Furthermore, from the results show that for more than 1000 packets and the length of time window of more than 300 seconds performance peak. Under these conditions the classification of malicious traffic is characterized with precision and recall with values greater than 0.995 and 0.982 respectively, while classification of benign traffic has even better performance with precision and recall higher than 0.995. Results presented in Figure 4 indicate that classifier requires a little bit less time to be trained when a longer time window is used. This can be explained due to the fact that shorter time window brings more training and test instances, which require more time to be processed. However, it should be noted that the testing time is not influenced by this. Moreover, it should be noted that used Random Forests classifier with 10 trees performed very well in the sense of time requirements, taking less than 120 seconds to be trained and to perform the classification.

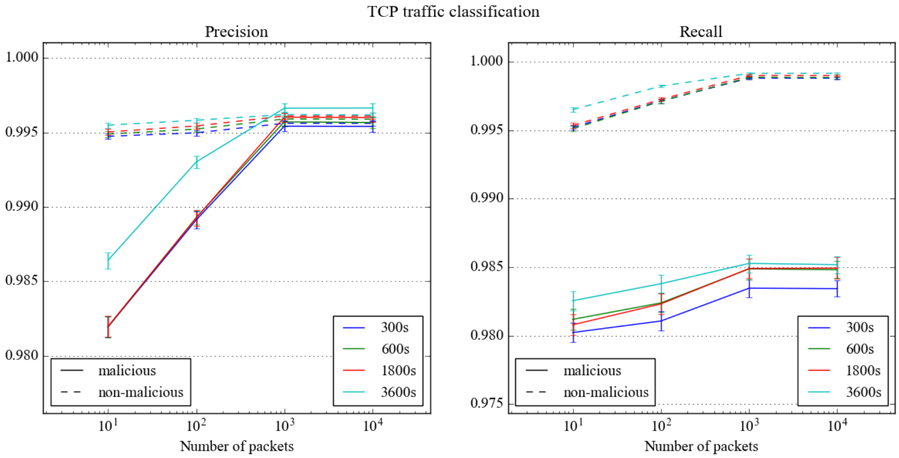

Figure 3. Classification results for TCP traffic.

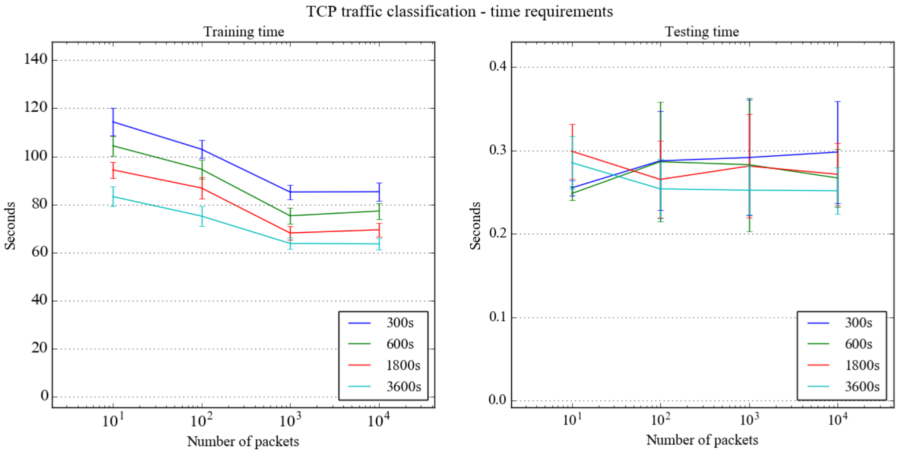

Figure 4. Time requirements for TCP traffic classification: Training and Testing time.

The results of UDP traffic classification are illustrated in Figure 5, showing constant classification performance of UDP traffic for different number of packets per conversation. Furthermore, the length of time window does not have significant influence on classification results. For time window equal to 3600 seconds and 10 packets per conversation precision and recall have values of above 0.995 and 0.985 while for benign traffic we have nearly perfect classification with precision and recall with values higher than 0.998 and 0.999 respectively. Figure 6 shows time requirements of UDP traffic classification. The results follow the same trends as in the case of TCP classification. The time needed to perform training and classification is less than 5 seconds, which is significantly lower than in case of TCP traffic due to the smaller number of UDP instances within our data set and smaller feature set used for representing them.

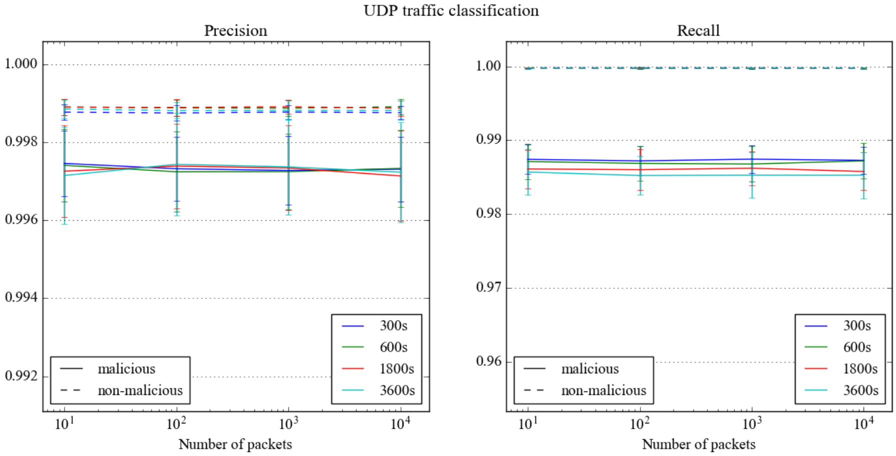

Figure 5. Classification results for UDP traffic.

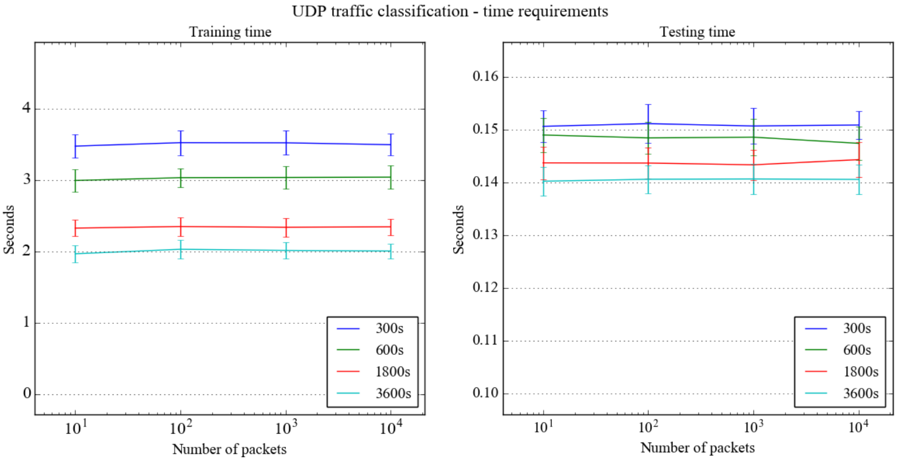

Figure 6. Time requirements for UDP traffic classification: Training and Testing time.

The results for DNS traffic are presented in Figures 7 and 8. The figures show that the performance of classification slightly degrade with increasing the size of time window. Overall, DNS classification has shown the performance comparable with TCP and UDP classifiers having the precision and recall for both malicious and benign traffic higher than 0.985 and 0.975 respectively. Time requirements follow the same trend as in cases of TCP and UDP traffic, where for training and classification the classifier requires less than 17 seconds.

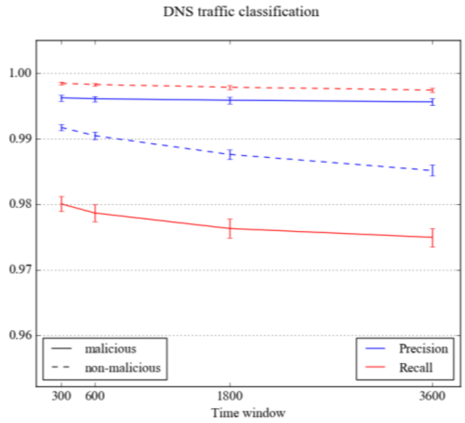

Figure 7. Classification results for DNS traffic.

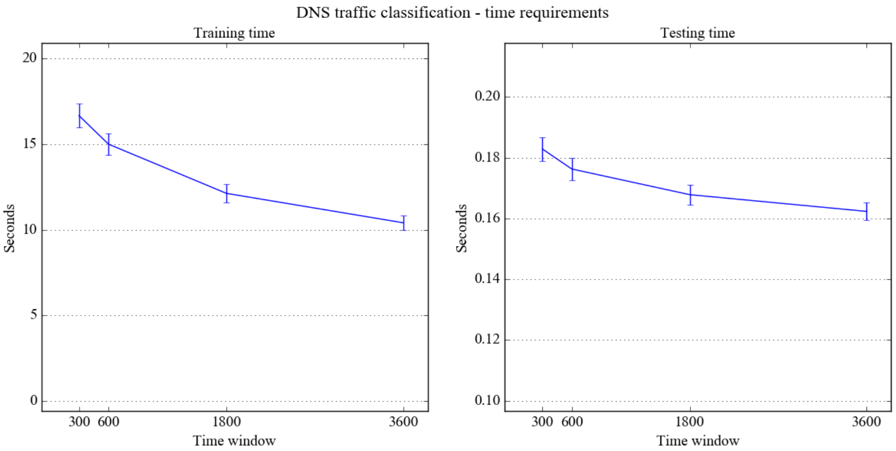

Figure 8. Time requirements for DNS traffic classification: Training and Testing time.

4.3.2. Identifying compromised clients

Tables 5, 6 and 7 illustrate the performance of the proposed system in identifying potentially compromised clients i.e. bots based on the traffic classification. Table 5 shows the results of identifying compromised client based on TCP, UDP or DNS traffic analysis. Table 6 illustrates results when all three levels of analysis are used. Finally, Table 7 represent an example of report produced for a client in the network. The results were obtained for using analysis window of 600 seconds and maximally 10,000 packets per conversation as for these parameters all three classifiers perform well.

The results of identifying compromised clients using only one of the traffic analysis levels are presented in Table 5. The table illustrates low FP and FN for each analysis levels. The results also show that different levels of traffic analysis are able to discover different number of clients indicating the potential of correlating findings from different levels of analysis. Table 6 on the other hand illustrates results of identifying compromised clients when all three traffic analysis levels are used. In this case all malicious client machines were identified while only 2 benign clients are deemed malicious, accounting for false positive rate (FPR) of 0.0435. The number of falsely identified machines is low and can be easily filtered out by a network operator.

Table 5. Results of identifying malicious clients based on TCP, UDP and DNS analysis.

TCP				UDP				DNS			
TP	**FP**	**TN**	**FN**	**TP**	**FP**	**TN**	**FN**	**TP**	**FP**	**TN**	**FN**
10	**1**	**32**	**1**	**4**	**1**	**41**	**7**	**7**	**1**	**27**	**4**

Table 6. Results of identifying malicious clients based on all three levels of traffic analysis.

ACTUAL		**PREDICTED**	
		Malicious	**Benign**
	Malicious	**11**	**0**
	Benign	**2**	**44**

For each client evaluated by the system i.e. for both malicious and benign machines the proposed approach outputs a report. The report provides a preliminary decision on the nature of client machine (i.e. Malicious or Benign), the number of TCP, UDP and DNS traffic instances indicated as malicious as well as a brief description of each traffic instances marked as malicious by the three classifiers. Table 7 illustrates example of report for a malicious client machine from the testing trace. The report provides the operator with a sufficient amount of information so a qualified decision on the nature of client can be made.

Table 7. An example of report provided by the method.

192.168.1.236 **Malicious**	**Classification results**		
	TCP	**UDP**	**DNS**
	Detected: 534 Total: 572	Detected: 33 Total: 89	Detected: 98 Total: 167

Overview							
	Nr.	**Source IP**	**Destination IP**	**Src port**	**Dst port**	**Protocol**	**Country**
TCP	1.	192.168.1.236	65.55.88.22	4599	25	SMTP	US
	2.	192.168.1.236	212.135.6.24	4773	25	SMTP	GB
	3.	192.168.1.236	217.12.11.64	4936	25	SMTP	GB
	…	…	…	…	…	…	…
	534.	192.168.1.236	212.108.64.65	1143	25	SMTP	GB
	Nr.	**Source IP**	**Destination IP**	**Src port**	**Dst port**	**Protocol**	**Country**
UDP	1.	192.168.1.236	124.83.81.49	4536	5320	-	PH
	2.	192.168.1.236	82.79.244.40	4439	5396	-	RO
	3.	192.168.1.236	201.250.155.47	4482	6945	-	AR
	…	…	…	…	…	…	…
	33.	192.168.1.236	77.77.16.211	4521	5904	-	BG
	Nr.	**Domain**	**Response type**	**Number of IPs**	**Average TTL**	**Hosting countries**	**Hosting ASs**
DNS	1.	bvznq.cc	NXDOMIAN	0	-	-	-
	2.	xnkuvnmui.ws	NOERROR	7	42	2	7
	3.	trlvwluyjtu.cc	NXDOMIAN	0	-	-	-
	…	…	…	…	…	…	…
	98.	dgghem.org	NOERROR	7	42	2	7

5. DISCUSSION

This section discusses the characteristics of the proposed detection method outlining its capabilities and limitations and elaborating on possibilities for future work.

5.1. Principles of operation

The proposed method targets machines compromised with bot malware at local and enterprise networks by detecting if they are associated with any malicious TCP, UDP and DNS traffic. In order to identify malicious traffic, we develop three classifiers that capture characteristics of malicious botnet network activity. The classifiers are based on a capable Random Forests classifier. Results of the classification are scrutinized by the client analysis entity in order to identify any malicious clients in the network.

The proposed method is able to identify compromised clients if they are producing traffic on any of the three protocols. Furthermore, as the proposed method is trained using data sets that include all phases of botnet operation i.e. C&C communication and attack traffic, the proposed system is independent from bot operational phase. Finally, we argue that the proposed approach can be even more reliable if additional data sets of malicious and benign traffic are used for training the classifiers.

For the three levels of traffic analysis we rely on feature sets that are specially developed to encompass characteristics of botnet network activity. We argue that chosen feature representation successfully captures a large subset of botnet traffic characteristics, thus avoiding common pitfall of tailoring feature representations that do not generalize well.

Traffic classifiers used for classifying of TCP, UDP and DNS traffic are based on Random Forests classifier with only 10 trees. This opens the potential of experimenting with much larger forests in order to further improve classification performance. The use of additional trees leads to approximately linear increase in training time but based on the results presented in Section 4 there is space for increasing time consumption for the sake of improving accuracy.

For the realization of classifiers, we have considered different lengths of analysis window and the number of packets per conversation. The length of analysis window directly affects how promptly can alerts be raised and compromised clients can be identified. Therefore, for the client identification evaluation we chose time window of 600 seconds as it provides balance in the performance of the three classifiers. Future work could consider experimenting with other sizes of time windows depending on the requirements of the actual detection system.

Finally, one of the crucial elements of the proposed method is the client analysis entity i.e. the way in which the approach implements the correlation of findings from the three levels of traffic analysis. The current method is based on filtering false alarms based on the fundamental traits of botnet operation. We believe that the used client analysis entity succeeds at automatizing a number of validation steps that would be performed by a network operator, thus significantly reducing efforts needed for scrutinizing the results of the approach. However, we acknowledge that the proposed client analysis should be further developed to cover more advance characteristics of botnet operation. Also the client analysis should be more thoroughly evaluated using additional malicious data sets that originate from botnets with different C&C communication mechanisms and attack strategies.

5.2. Detection performance

The results presented in the previous section illustrate a great potential of using the proposed method for identifying compromised client machines at local and enterprise networks.

The three traffic classifiers are characterized with high performance for both malicious and benign traffic. We are seeing a low number of false positives for all three classifiers which give us confidence in relying on them for identifying compromised clients based on TCP/UDP conversations and DNS queries that are flagged as malicious. Furthermore, the classification results are on par or better than the ones reported by the existing work (Stevanovic et al., 2016). Finally, as we used one of the most comprehensive botnet traffic data sets for development and evaluation of the proposed classifiers, we are confident that we would see similar results on new, previously unseen, traffic traces.

The analysis of the number of packets per TCP/UDP conversation and the length of time window brings interesting results as well. Varying the number of packets per TCP/UDP conversation it is obvious that the best performance of TCP traffic classification is obtained for more than 1000 packets per conversation while the number of packets does not have an effect on UDP classifier. It should also be mentioned that even for 10 packets per TCP conversation we are seeing good classification results and low number of false positives. This is quite promising as not needing to trace higher number of packets would save resources when the system is analysing heavy traffic flows. Regarding the size of time window used when analysing the traffic, TCP and UDP classification has better detection performance for longer time window. On the other hand, the length on the time-window causes slight degradation of the performance of DNS traffic classification. Therefore, we can conclude that balanced classification performance can be obtained for time window of length 600 seconds and 1000 packets per TCP/UDP conversation.

The results of identifying compromised clients are promising as well. For the used evaluation scenario, we have seen only 2 falsely identified clients which accounts for FPR of 0.0435. However, we would like to stress that further evaluation is needed in order to make more conclusive results about the proposed client analysis entity.

Although promising the classification performance of the three classifiers should be further improved so the number of false positives would be minimized, which would consequently lead to more accurate identification of compromised clients by the client analysis entity. This could be achieved by optimizing feature sets used for representing traffic instances so they would better capture the heuristics of botnet traffic. This requires further analysis of malicious botnet traffic, as well as benign traffic that have been misclassified as malicious. Furthermore, we could obtain a certain improvement of classification performance by using a higher number of classification trees within the Random Forests classifier. These improvements would come at the cost of the time-efficiency of classification but are still worth considering. Finally, the results of traffic analysis could be improved by using additional traffic data sets of higher quality for training the classifiers.

5.3. Operational deployment

The proposed method is developed with operational deployment in mind and we believe that it represents a good candidate for being deployed in real-world operational networks. The proposed detection method could be potentially deployed as a cloud-based solution where classification and client analysis entities could be placed in the cloud, while traffic processing would be implemented in routers that connect home or enterprise network to the Internet. This application would unlock possibilities for deploying more advanced classifiers and client analysis entities that could leverage the computational power of the cloud. The operational deployment should also be coupled with existing malware testing environments that would continuously update the pool of training data with traffic traces originating from the latest malware samples.

5.4. Future work

The future work will be devoted to further optimization of the traffic classification so a number of false positives would be even further reduced. This can be done through future feature optimization and feature engineering. Furthermore, we will place special focus on optimizing client analysis entity so compromised clients would be identified more precisely. Evaluation of proposed method using novel traffic traces will also be performed. Finally, we will work on operational deployment of the proposed method.

6. CONCLUSION

In this paper we proposed a novel approach for detection of bots at local and enterprise networks. The proposed approach employed multi-level traffic analysis by analysing TCP, UDP and DNS traffic using supervised machine learning. The proposed method integrated three traffic classifiers based on Random Forests classifier and novel sets of features designed to better capture the characteristics of botnet network activity. We evaluated the proposed method within one of the most extensive evaluation campaigns using traffic traces from 40 bot samples and diverse benign applications. The results of evaluation indicate the possibility of obtaining high accuracy of botnet traffic classification for all three classification methods. The proposed TCP classifier is characterized by precision and recall higher than 0.98 for analysing only 10 packets per conversation. The UDP classifier is less sensitive to the number of analysed packets having precision and recall higher than 0.995 and 0.985, respectively. The DNS classifier has also shown overall stable performance with precision and recall higher than 0.995 and 0.976, respectively. The presented results are in the most cases better than the results reported by the existing work thus indicating a great potential of using the proposed classifiers. Furthermore, the proposed method has shown the ability of identifying compromised machines with high accuracy while producing only a small number of false positives. The future work will be devoted to further performance evaluation and the optimization of the traffic analysis. Finally, special emphasis will be placed on improving the client analysis used by the method in order to ensure reliable identification of compromised clients.

7. REFERENCES

Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou II, N., & Dagon, D. (2011). Detecting Malware Domains at the Upper DNS Hierarchy. In USENIX Security Symposium (p. 16).

Bilge, L., Balzarotti, D., Robertson, W., Kirda, E., & Kruegel, C. (2012). Disclosure: detecting botnet command and control servers through large-scale netflow analysis. In Proceedings of the 28th Annual Computer Security Applications Conference (pp. 129-138). ACM.

Bilge, L., Sen, S., Balzarotti, D., Kirda, E., & Kruegel, C. (2014). EXPOSURE: a passive DNS analysis service to detect and report malicious domains. ACM Transactions on Information and System Security (TISSEC), 16(4), 14.

Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

Bujlow, T., Carela-Español, V., & Barlet-Ros, P. (2013). Comparison of Deep Packet Inspection (DPI) Tools for Traffic Classification. Universitat Politècnica de Catalunya.

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

García, Sebastián, Alejandro Zunino, and Marcelo Campo (2014b). "Survey on network-based botnet detection methods." Security and Communication Networks 7.5 (2014): 878-903.

García S., and Uhlir V. (2016). Malware capture facility project. Retrieved August 1, 2016, from http://mcfp.weebly.com.

Haddadi, F., Runkel, D., Zincir-Heywood, A. N., & Heywood, M. I. (2014, July). On botnet behaviour analysis using GP and C4. 5. In Proceedings of the

Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation (pp. 1253-1260). ACM

Hogben, G., Plohmann, D., Gerhards-Padilla, E., & Leder, F. (2011). Botnets: Detection, measurement, disinfection and defence. European Network and Information Security Agency.

Masud, M. M., Al-Khateeb, T., Khan, L., Thuraisingham, B., & Hamlen, K. W. (2008). Flow-based identification of botnet traffic by mining multiple log files. In Distributed Framework and Applications, 2008. DFmA 2008. First International Conference on (pp. 200-206). IEEE.

Parkour M. (2016). Contagio malware dump. Retrieved February 25, 2015, from http://contagiodump.blogspot.dk/

Pedersen, J. M., & Stevanovic, M. (2015). AAU-Star and AAU Honeyjar: Malware Analysis Platforms Developed by Students. In R. S. Choraś (Ed.), Image Processing and Communications Challenges 7. (pp. 281-287). Springer. (Advances in Intelligent Systems and Computing, Vol. 389). DOI: 10.1007/978-3-319-23814-2_32

Perdisci, R., Corona, I., & Giacinto, G. (2012). Early detection of malicious flux networks via large-scale passive DNS traffic analysis. Dependable and Secure Computing, IEEE Transactions on, 9(5), 714-726.

Saad, S., Traore, I., Ghorbani, A., Sayed, B., Zhao, D., Lu, W., ... & Hakimian, P. (2011). Detecting P2P botnets through network behavior analysis and machine learning. In Privacy, Security and Trust (PST), 2011 Ninth Annual International Conference on (pp. 174-180). IEEE.

Shin, S., Xu, Z., & Gu, G. (2012). EFFORT: Efficient and effective bot malware detection. In INFOCOM, 2012 Proceedings IEEE (pp. 2846-2850). IEEE.

Shiravi, A., Shiravi, H., Tavallaee, M., & Ghorbani, A. A. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. computers & security, 31(3), 357-374.

Silva, S. S., Silva, R. M., Pinto, R. C., & Salles, R. M. (2013). Botnets: A survey. Computer Networks, 57(2), 378-403.

Strayer, W. T., Lapsely, D., Walsh, R., & Livadas, C. (2008). Botnet detection based on network behavior. In Botnet Detection (pp. 1-24). Springer US.

Stevanovic, M., & Pedersen, J. M. (2014). An efficient flow-based botnet detection using supervised machine learning. In Computing, Networking and Communications (ICNC), 2014 International Conference on (pp. 797-801). IEEE.

Stevanovic, M., & Pedersen, J. M. (2015). An analysis of network traffic classification for botnet detection. In The proceedings of International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), 2015. IEEE Press. (International Conference on Cyber Situational Awareness, Data Analytics and Assessment Proceedings. (cyberSA)). DOI: 10.1109/CyberSA.2015.7361120

Stevanovic, M., & Pedersen, J. M. (2016). On the Use of Machine Learning for Identifying Botnet Network Traffic. Journal of Cyber Security and Mobility, 4(2 & 3), 1-32. DOI: 10.13052/jcsm2245-1439.421

Zhao, D., Traore, I., Sayed, B., Lu, W., Saad, S., Ghorbani, A., & Garant, D. (2013). Botnet detection based on traffic behavior analysis and flow intervals. Computers & Security, 39, 2-16.

KEY TERMS

- ***Botnet* -** a network of computers compromised with sophisticated bot malware.

- ***Botnet Detection*** **-** detection of botnets based on either behavioural or network traffic analysis.
- ***Traffic Analysis*** **-** the analysis of network traffic produced by malicious or benign applications.
- ***Traffic Classification*** – the classification of network traffic instances.
- ***MLAs*** – machine learning algorithms (MLAs) represent a set of methods used for classification, regression or clustering of observations.
- ***Random Forests*** – a class of ensemble machine learning algorithms based on "bagging" of decision tree classifiers.
- ***Client Analysis*** – the analysis of network-based alerts with a goal of identifying bot malware infections.

BIOGRAPHICAL NOTES

Matija Stevanovic received the Ph.D. in Electrical Engineering in 2016 from Aalborg University, Denmark and M.Sc. in Electrical Engineering in 2011, from the Faculty of Electrical Engineering, Belgrade University. He is currently a postdoctoral researcher in the Wireless Communication Section, Department of Electronic Systems, Aalborg University. His research interests include network security, traffic anomaly detection and malware detection based on network traffic analysis.

Jens Myrup Pedersen received the M.Sc. in Mathematics and Computer Science in 2002, and the Ph.D. in Electrical Engineering in 2005 from Aalborg University, Denmark. He is currently Associate Professor at the Wireless Communication Section, Department of Electronic Systems, Aalborg University. His research interests include network planning, traffic monitoring, and network security. He is author/co-author of more than 100 publications in international conferences and journals, and has participated in Danish, Nordic and European funded research projects. He is also board member of a number of companies within technology and innovation.

Reference to this paper should be made as follows: Stevanovic, M. & Pedersen JM. (2016). Detecting bots using multi-level traffic analysis. *International Journal on Cyber Situational Awareness*, Vol. 1, No. 1, pp182-209.