Early Detection of In-the-Wild Botnet Attacks by Exploiting Network Communication Uniformity: An Empirical Study

Zainab Abaid*†, Mohamed Ali Kaafar† and Sanjay Jha* †
*School of Computer Science and Engineering, University of New South Wales, Australia
{zainaba,sanjay}@cse.unsw.edu.au
†CSIRO Data61, Australia
dali.kaafar@data61.csiro.au

***Abstract*—Distributed attacks originating from botnet-infected machines (bots) such as large-scale malware propagation campaigns orchestrated via spam emails can quickly affect other network infrastructures. As these attacks are made successful only by the fact that hundreds of infected machines engage in them collectively, their damage can be avoided if machines infected with a common botnet can be detected early rather than after an attack is launched. Prior studies have suggested that outgoing bot attacks are often preceded by other "tell-tale" malicious behaviour, such as communication with botnet controllers (C&C servers) that command botnets to carry out attacks. We postulate that observing similar behaviour occuring in a synchronised manner across multiple machines is an early indicator of a widespread infection of a single botnet, leading potentially to a large-scale, distributed attack. Intuitively, if we can detect such synchronised behaviour early enough on a few machines in the network, we can quickly contain the threat before an attack does any serious damage. In this work we present a measurement-driven analysis to validate this intuition. We empirically analyse the various stages of malicious behaviour that are observed in real botnet traffic, and carry out the first systematic study of the network behaviour that typically precedes outgoing bot attacks and is synchronised across multiple infected machines. We then implement as a proof-of-concept a set of analysers that monitor synchronisation in botnet communication to generate early infection and attack alerts. We show that with this approach, we can quickly detect nearly 80% of real-world spamming and port scanning attacks, and even demonstrate a novel capability of preventing these attacks altogether by predicting them before they are launched.**

I. Introduction

Network administrators often become aware of botnet infections only when large-scale attacks, such as high-volume spam email, are observed. Prior work [9], [13] has suggested that these attacks are only the final blow in a multi-stage botnet infection sequence, and are usually preceded by an inbound exploit, e.g. drive-by download or SSH exploit, malicious binary download and communication with a command-and-control (C&C) server. Our intuition is that if these "pre-attack" stages of a botnet infection occur in a synchronised manner on multiple hosts, it is highly likely that the hosts are part of a botnet and may therefore engage in a large-scale attack. At this point it is possible to raise an early alert for network attacks. Thus, to achieve this early detection capability, it is first of all important to empirically investigate the temporal relationship among botnet infection stages to understand which behaviour typically precedes attacks. One would then monitor hosts for *synchronised* patterns in this behaviour, for example, multiple hosts contacting the same C&C server. Once such patterns are discovered, it is possible to generate an alert for attacks that members of a botnet may already have launched or are soon to launch.

ISBN 978-3-901882-94-4

The idea of observing the network for uniformity in communication patterns, for example similar HTTP requests to C&C servers, exists in earlier malware detection research [14], [12]. However such behavioural similarity has been used only in a limited manner, either as added evidence of an already suspected infection or to merely discover members of the same botnet family [10]. To the best of our knowledge it has not been investigated as an *early attack indicator*. Furthermore, there has been little effort to empirically analyse the correlation and temporal relationship between synchronised malicious activity and outgoing attacks, or to comprehensively study whether this approach can aid in early detection of attacks.

In this paper, we carry out a systematic study to investigate the important temporal relationship between attacks and synchronised malicious behaviour, and to investigate the possibility of quickly detecting attacks based on this relationship. Our investigation has been conducted on real network traces containing wide-area-network (WAN) traffic of 380 home and office networks which show a rich variety of "in-the-wild" botnet activity, comprising at least 13 known botnet families including HTTP, peer-to-peer(P2P) and IRC-based botnets. We first carry out statistical measurements on the dataset to determine the malicious behaviour that typically precedes attacks, and verify that the result of our analysis is in agreement with the behavioural sequence suggested in early botnet studies. We then build a proof-of-concept set of *analysers*, based on network traffic and protocol analysis, that monitor hosts for synchronisation in the behaviour typically preceding attacks.

We study how quickly our analysers are able to identify botnet infections based on synchronised network activity compared to a reputable, commercial blacklisting service, and whether they are able to raise timely alerts of attacks. We even investigate the novel notion of predicting attacks *before* they are launched and present insights into the limitations that must be overcome to make such predictions possible.

Overall, the key contributions of our work are as follows:

(a) An empirical analysis of the infection sequence of botnet malware pertaining to 13 distinct, real-world botnet families;
(b) A measurement of the communication uniformity observed in botnet infections in our dataset, and a systematic investigation of the correlation and temporal relationship between such uniformity and outgoing bot attacks;
(c) A novel approach for early detection of infections and attacks and experimental evaluation of its accuracy and usefulness compared to a commercial blacklisting service.

We demonstrate that the vast majority (up to 96%) of the infected hosts in our dataset demonstrate some level of synchronisation in various kinds of malicious behaviour, and that this synchronisation is correlated with attacking behaviour for at least 83% of the hosts. We also show that with this approach, at least 91% of the time, we were able identify infections no later than a commercial blacklisting service. Most importantly, we empirically validate the early attack detection potential of our approach by detecting 77% of attacks almost at launch-time, and show that even with our proof-of-concept implementation, it is sometimes possible to predict attacks *before* they occur; we present also our insights into the prediction problem and possible future directions for making it possible. As a further contribution, the scripts used for the analysis of synchronised behaviour will be released as a proof-of-concept open-source implementation to foster future research in the area by using and extending our work.

The remainder of this paper is organised as follows. Section II presents related work in this area; Section III presents an empirical analysis of temporal relationships among various infection stages; Section IV outlines our approach and algorithms; Section V presents the experimental setup and the issues we want to investigate; Section VI discusses the results of the investigation, and Section VII concludes the paper.

II. Related Work

Given the damaging potential of botnet attacks if left unchecked, the problem of quick detection and mitigation has received attention from security researchers. A common solution has been to improve the speed of network analysis tools and algorithms, for example by deploying parallel and distributed detection systems [20], exploiting parallelisation at the hardware level [18], or designing speedier algorithms for existing techniques such as faster pattern or rule matching [22]. Our work takes an approach that is orthogonal to these efforts; instead of trying to speed up existing techniques, we explore a different approach to early attack detection by using the uniformity expected from botnet communications as an indicator of a widespread infection and possible attacks.

Botnet detection research has frequently exploited botnets' behavioural uniformity in the past. A common detection approach is to train machine learning classifiers to identify typical patterns of botnet communication. Hosts are declared malicious if their communications match the learned patterns, which may be defined in terms of packet header attributes (for example protocol or size) [25], DNS activity [21], or P2P traffic characteristics [23]. Such solutions are only successful if future botnets follow past behavioural patterns that the classifiers were trained on. In this work, we overcome this limitation by relying on the tendency of botnets to engage in similar malicious communications, an invariant property that is not likely to change as the botnets evolve. In this regard, our work is inspired by prior research that considers traffic similarity itself as an indicator of infection. Gu et. al.'s BotMiner [12] clusters hosts based on similar communication traffic as well as similar malicious traffic; cross-cluster correlation identifies as bots the hosts that share both similar malicious activities and similar communication patterns. As similarity can be coincidental, some approaches tighten the criteria and look for *synchronisation* in network behaviour, which implies the presence of common entities (e.g. DNS queries, connection endpoints, attack victims etc.) in different hosts' communication. An example is BotSniffer [14] which detects C&C communication channels in network traffic by monitoring the command-response patterns of hosts that connect to with the *same* IRC and HTTP servers.

However, while these approaches have inspired our work, no prior research has proposed using synchronised behaviour as an indicator of upcoming attacks, nor empirically studied the relationship between botnet attacks and synchronised behaviour on real-world data. We present the first such study, and in addition, investigate the novel capability of using the uniformity of botnet communication to not only *detect* but also to *predict* attacks. To the best of our knowledge, the only recent work on attack prediction [15] was limited to monitoring network links for the coordination traffic that precedes distributed denial of service (DDoS) attacks. In this work, we investigate a general solution rather than focusing on a specific attack.

III. Investigating the Temporal Relationship Among Infection Stages

A. Dataset Description

We carry out our study on a real user dataset obtained from a large Internet Service Provider that serves both business and home clients, shared with us by the SysNet research lab [5]. It is a high-volume trace 500 GB in size, and was collected for approximately 8 hours (439 minutes) on 18 September 2012, on a full working day, at the B-RAS (Broadband Remote Access Server) of the ISP. It represents wide-area (WAN) traffic from 380 home or small enterprise networks, each assigned a different public IP address. The trace is in the form of PCAP files, containing full packets flowing to and from each

TABLE I: Description of botnet families in our dataset and number of hosts infected by each.

Family	Number of Hosts	Possible infection vector	Description
blackenergy	8	Distributed via drive by download or spam (no self propagation)	Used for DDoS, information stealing and malware distribution capability; commanded by HTTP-based centralized C&C.
dirtjumper	3	Wide variety – exploit, drive by download or distributed by malware	It is a DDoS toolkit with large number of variants (e.g. Pandora and Di botnet spawned from it), mostly with HTTP-based C&C.
graybird	2	Various Windows exploits depending on version; e.g. Hupigon uses IE vulnerability when compromised website accessed	Spyware Trojan with a centralised IRC or custom protocol based C&C.
gumblar	59	Various exploits, especially Acrobat or Flash from websites hosting infected Javascript	Centrally commanded for information stealing and delivering other malware; uses stolen FTP credentials to infect websites.
haxdoor	8	Bundled download through other applications, or various browser exploits triggered from spam or blog links	IRC-based keylogging Trojan to steal banking information
hitpop	1	Often downloaded by Trojans as payload bundled with other malware	HTTP-based C&C server usually commands DDoS attacks
mpack	15	Varies widely	Malicious toolkit from which a variety of botnets are spawned (e.g. Srizbi botnet; generally hosted on infected websites
optima	4	Drive by download among other possibilities	Centralized HTTP-based C&C commands DDoS attacks
pincher	21	Trojan possibly distributed as bundled download	Runs C&C server on TCP port 81 and is used for Information stealing; also possibly used as malware dropper (installs other malicious files)
torpig	1	Drive by download; exploits Adobe/Java vulnerabilities and installs Mebroot rootkit	Used for man in the browser phishing attacks for financial data stealing
tsunami	4	Not associated with any particular infection vector	IRC based Trojan with DDoS capability
zonebac	27	No particular vector; drive by download possible	Backdoor Trojan; lowers browser security settings; uploads sensitive information about host; could install additional malware

network. The ground truth information was provided as a one-time sample by the reputable security company Team Cymru [7]. The company maintains a proprietary threat repository of known C&C servers identified with the help of a chain of globally-deployed sensors. They provided a blacklist in the form of an hourly updated list of C&C servers that were being actively contacted by hosts from the monitored region during the data collection period. All hosts in the trace that perform bidirectional communication with an IP in that C&C list were flagged as bots. Based on this ground truth, 107 (out of 380) hosts were infected. The 107 infected hosts represent at least thirteen different botnet families. Table I (taken from our earlier work [11]) shows the names of these families and the number of hosts infected by each. As some ‘’hosts” (identified by public IP addresses) in fact represent more than one machine behind a Network Address Translation device, they may be infected by more than one kind of botnet. The total of Table I does not therefore add to 107.

We acknowledge that the dataset, collected in 2012, is a few years old and does not necessarily represent modern botnets, especially mobile malware. However, owing to policy, ethical and privacy restrictions in obtaining a real user dataset, we leave for future work repeating our experiments on more recent data.

B. Empirical Analysis of Behaviour Leading to Attacks

Prior studies [9], [13] have suggested that bots follow a well-defined sequence, starting with an incoming exploit, which if successful will result in a malicious binary download, followed by communication with a C&C server and then an outgoing attack, such as malware propagation, spam, or port scans. One of the earliest analyses of this sequence, presented in [9], is shown in Fig. 1. Another analysis of this sequence was presented in the seminal work by Gu et. al [13] and is shown in Fig. 2. We observe from the figure that the typical behaviour immediately preceding outgoing attacks is communication with a C&C server and a binary download.

The key idea of our analysis is to investigate, based on a new dataset of botnet infections, whether real-world botnet attacks are indeed preceded by these behaviours. For this study, we defined four distinct high-level behavioural stages described in Table II to represent a complete footprint of a botnet infection. To detect which of these stages were exhibited by infected hosts in our dataset, we used Snort [19], an open-source intrusion detection system. Snort applies a set of rules and signatures to network traffic to detect behaviour pertaining to malware infections. It generates an alert for each rule that is triggered by a malicious event observed in the traffic. We used well-tested rules from EmergingThreats [1] and Talos [6], as well as from the industry-standard commercial botnet detection tool, BotHunter [13]. The accuracy of BotHunter’s rules has been validated over multiple datasets by other researchers [13], [16]. We were therefore able to detect each stage of infection using a comprehensive set of signatures and heuristics, and find this labelling reliable for the purpose of this study.

We ran Snort over the traces of the infected machines in the dataset and mapped the alerts in the output log to one of the four defined malicious stages. We thus converted the output logs to sequences of the four stages, with one sequence generated against each host and each sequence covering the duration of eight hours. Table III presents a sample of the generated sequences. While botnets carry out a range of attacks, in this work we restrict our scope to spam campaigns and port scanning, as present rulesets are not capable of detecting any

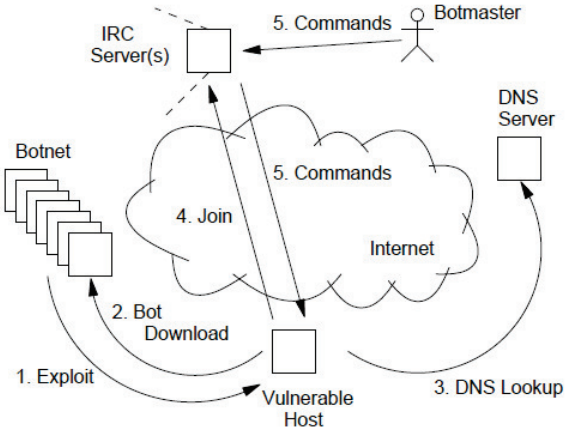

Fig. 1: Rajab et. al botnet infection sequence.

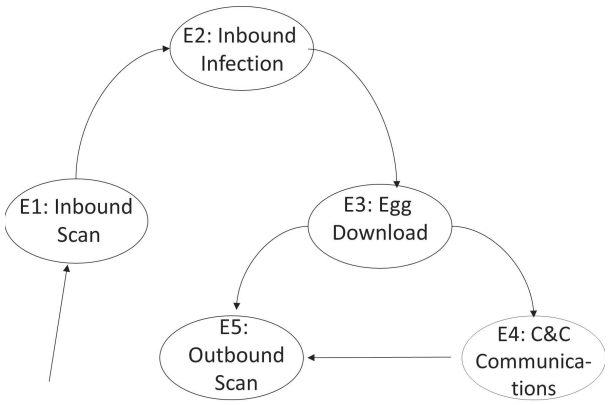

Fig. 2: Gu et. al botnet infection sequence.

TABLE II: Description of stages of malicious behaviour defined for our empirical analysis.

State	**Description of Behaviour**
Inbound Exploit	Includes the following: an incoming port scan aimed at discovering a vulnerable service; an exploit targeting application or OS vulnerabilities for gaining "backdoor" access or remote execution privileges; social engineering where users are tricked into compromising on usual security practices, e.g. clicking on an intriguing spam link; and drive-by downloads.
Binary Download	Download of a malicious piece of code or an executable, which may be disguised.
C&C Discovery and Communication	Attempts to contact the Botnet's C&C server and exchanges with the discovered C&C server.
Outgoing Attack	Any attack on other entities, e.g. phishing, malware propagation, DDoS or spam.

other attacks. However, these attacks are important to capture and mitigate as early as possible as they are the most likely to cause malware propagation and hence growth of the botnet.

We answer the question of which behaviour is most likely to precede attacks in real botnet infections by observing the occurrence of each behavioural stage i in the data, and calculating the frequency $f_{i\to j}$ with which it led to every other stage j as follows:

$$f_{i\to j} = \frac{\sum N_{i\to j}}{\sum N_i} \quad (1)$$

where:
$\sum N_{i\to j}$ is the number of times a stage i led immediately to a stage j;
$\sum N_i$ is the number of times stage i occurred in total, regardless of which stage it led to.
We calculate the value $f_{i\to j}$ for all possible pair-wise sequences of the behavioural stages, or *behavioural transitions*. The objective is to investigate which stages lead to the attack stage most frequently. Table IV shows the frequencies of occurrence we obtained for each behavioural transition, with each cell $[i, j]$ representing the value $f_{i\to j}$, or the frequency with which stage j was seen immediately after stage i. We highlight the top-3 values in the table showing the most likely transitions. The key result from this investigation is that the outbound attack stage (the column headed "Attack") is preceded with the highest probabilities by the C&C communication stage, followed by the Binary Download stage, which is in agreement with prior studies.

IV. Detection Approach and Analysers

A. Overview of Approach

Our approach is based on monitoring hosts for synchronisation in the behavioural stages known to precede botnet attacks. The empirical evidence presented in Section III suggests that these are most likely to be the C&C communication and binary download stages respectively. In our current work, however, as we present only a proof-of-concept implementation of this approach, we focus on detecting only synchronised C&C communication. It may be argued that it is sufficient to observe a single host engaging in C&C communication in order to raise an early attack alarm. However, the additional constraint of observing the communication from multiple machines would reduce false alarms, as synchronised malicious activity is a much stronger indicator that a coordinated attack will originate from the botnet. Our goal now is to investigate whether bots show a significant level of synchronisation in their C&C communication preceding attacks, and whether such synchronisation can be used to detect the bots at an early stage of infection, to detect attacks quickly, or even to predict attacks before they occur.

We define synchronised behaviour as an occurrence where two or more hosts communicate with common entities, e.g. query the same URL, and define three different indicators

TABLE III: Example of sequences of behavioural stages observed in dataset.

IP	Event Sequence
a.a.a.a	2:CNC, 3:CNC, 4:ATTACK, 7:EXPLOIT, 9:CNC, 13:BINARY, 16:EXPLOIT, 18:ATTACK, 23:CNC, 25:ATTACK, 27: BINARY...
b.b.b.b	0:EXPLOIT, 6:EXPLOIT, 7:BINARY, 19:ATTACK, 23:CNC, 26: BINARY, 27:ATTACK, 29:EXPLOIT, 30:BINARY, 36:CNC...
c.c.c.c	1:ATTACK, 2:ATTACK, 6:EXPLOIT, 7:BINARY, 9:CNC, 10:ATTACK, 11:CNC, 14:EXPLOIT, 18:BINARY, 19:CNC, 21:ATTACK...

of synchronised C&C Communication in terms of directly observable network attributes. We implement analysers to monitor the network and generate an alarm when each indicator is observed. We then analyse the temporal relationship between the alarms indicating synchronised behaviour and attacks (with which we labelled our dataset using Snort as discussed in Section III).

B. Description of Analysers

a) Domain Generation Algorithm (DGA) Analyser: This analyser aims to discover botnets running domain generation algorithms (DGA) to discover their C&C servers. DGA is a common technique used by botnets to hide their C&C servers, in which the server hides behind an ever-changing domain name. Bots algorithmically generate and try to resolve a number of domains, only one of which is registered as the C&C server. This mapping changes every day or few hours. Thus DGA behaviour is characterised by many, often repeating, failed DNS queries. The DGA analyser looks for instances of the same DNS query generating an NXDomain (non-existent domain) response on multiple machines. Ordinarily, an NXDomain response would follow a user error such as in misspelling a web address, so it is highly unlikely that the same NXDomain is seen on multiple machines. It is therefore a strong indicator of synchronised malicious behaviour. The DGA analyser generates an alarm for each host whose failed DNS queries also failed on other machines.

b) Failed Connection Analyser: Botnet malware sometimes ships with a hard-coded list of IP addresses that it tries to connect to; only one of those addresses is the actual C&C server. Thus, only one of the connections is successful. This analyser looks for multiple machines experiencing the same TCP connection failures. To reduce false positives, we filter connections to private (local) IP addresses (e.g. those starting with `192.168`), on the assumption that a C&C server will need to be globally accessible and will usually not be running behind a private IP address.

c) Blacklisted Host Contact Analyser: This analyser pulls publicly available URL and IP blacklists from well-reputed security companies and monitors for multiple machines contacting the same blacklisted host. It monitors both outgoing TCP connections to blacklisted IP addresses and HTTP connections to as well as DNS queries for blacklisted URLs. Instead of relying on a single blacklist, for increased sensitivity we combine blacklists from the following services: PalevoTracker [3], ZeusTracker [8], SpyeEyeTracker [4], and MalwareDomainList [2]. We additionally use blacklists included with BotHunter[13] as it is a well-reputed botnet detection software. Blacklists change continuously as some malicious servers and domains are taken down and others added every day. As our dataset was collected in September 2012, in our current implementation we used blacklists that were downloaded in the same month for maximum accuracy in detecting communication with servers that were blacklisted at the time.

TABLE IV: Proportions of all behavioural transitions observed in the dataset.

	Exploit	Binary	C&C	Attack
Exploit	0	0.219	0.313	0.469
Binary	0.013	0	0.257	0.730
C&C	0.007	0.129	0	0.864
Attack	0.012	0.308	0.680	0

We implemented the analysers using Java code on top of the Bro network analysis framework [17]. Bro ships with many scripts for protocol analysis and allows for adding custom scripts. We used all the base Bro functionality, along with a DNS failure analysis script from BotFlex, an open source tool that uses heuristics to detect the different stages of botnet behaviour [16]. We then implemented each analyser as a Java program that post-processed the event logs generated by Bro. As a contribution to the research community, we will release our analysis code as a guideline for implementing other synchronised behaviour analysers.

V. Experimental Setup

We now present three research considerations, each pertaining to a different aspect of our approach. The first is concerned with investigating the correlation between a bot infection and synchronised behaviour, the second sets a minimum criteria for the usefulness of our approach, and the third deals with the temporal relationship between synchronised behaviour and attacks.

A. Correlation between behavioural synchronisation and infection of hosts

We evaluate the correlation between infection and synchronised behaviour by taking the alarms raised by our analysers as an infection declaration. That is, when an analyser raises an alarm for synchronised behaviour being observed on a group of hosts, we label each of the hosts as infected, and then evaluate the accuracy of the labelling. Our main concern is whether our analysers generate alarms for only infected hosts or also falsely raise alarms for benign hosts. If this approach results in a high rate of true positives (TPR), it will indicate a strong correlation between infection and synchronisation, as most hosts engaging in synchronised behaviour are infected.

Although we are effectively using – and evaluating – the analysers as *detection algorithms*, it is important to note that the aim is not to propose this approach as a botnet detection solution, but merely to investigate this important correlation.

B. Minimum usefulness criterion: early infection detection potential

We test the early infection detection capability of our analysers compared to a commercial C&C blacklist, i.e., can the analysers detect an infection before a blacklist service detects C&C communication? Note that *infection detection* refers only to identifying that a host is infected with botnet malware, regardless of whether or not it is engaging in attacks; thus, it is different from *attack detection*. This is important because C&C communication is certainly not the first stage in a bot's infection sequence, being preceded instead by inbound exploit and malware download, and it may be followed very quickly by an outgoing attack. An analyser that is not able to raise an alarm until after the C&C Communication stage cannot aid in effectively apprehending attacks.

We use the same Team Cymru blacklist that was used to label our dataset, and for each host, we record as the *blacklist detection time* the timestamp of the first instance of bidirectional communication with a server on the blacklist. We denote this blacklist detection time as t_B, and the time that each analyser A_i first raised an alarm as t_{A_i}. The *detection delay* of the analyser is then computed as

$$Del_{A_i} = t_{A_i} - t_B \tag{2}$$

We declare an analyser to have good early infection detection potential if $Del_{A_i} \leq 0$ for most hosts, i.e., the analyser raises alarms no later than the blacklist detection time. Note that this only forms a *minimum criterion* for an analyser's usefulness; ideally, an analyser should be able to raise an alarm *before* a blacklist can. Thus, we also investigate strictly early alarm generation potential of our analysers in Section VI-C.

C. Correlation and temporal relationship between attacks and behavioural synchronisation

We investigate two aspects: (a) the *correlation* between attacks and behavioural synchronisation, i.e., how often hosts that launch attacks also show synchronised behaviour; and (b) the *temporal relationship* between attacks and behavioural synchronisation, i.e., whether synchronised behaviour occurs before, at the same time as, or after attacks are observed. Specifically, we ask ourselves the following questions:

1) In what proportion of hosts do we observe synchronised behaviour but no attacks? *(Question 1)*
2) In what proportion of hosts do we observe synchronised behaviour as well as attacks at any time before or after the synchronised behaviour? *(Question 2)*
3) In what proportion of hosts do we see some form of synchronised behaviour that occurs strictly *before* an attack? *(Question 3)*
4) Which analysers are the most (and least) effective at generating early warnings? *(Question 4)*

The first two questions aim to investigate whether this approach could be misleading by generating warnings of attacks that never materialise. The third question aims to investigate whether this approach can actually help us generate *early* warnings for attacks, and the fourth question aims to point out the best behavioural indicator of upcoming attacks.

VI. Results and Discussion

We now present the results from the three experiment categories described in Section V, i.e., pertaining to the correlation between behavioural synchronisation, infection and attacks, and the early infection detection potential of our analysers.

A. Correlation between behavioural synchronisation and infection

As discussed in Section V-A, we investigate this correlation by presenting an analysis of the accuracy of our approach in detecting infections, irrespective of the presence or time of attacks. The idea is to get a measure of how frequently each type of synchronised behaviour that we analyse is associated with infections. We measure accuracy in terms of a confusion matrix, declaring a true positive if an analyser generates at least one alarm for a host that is actually infected (according to the ground truth) and a false positive if the alarm is for a host that is actually benign. Similarly, a true negative is declared if an analyser never raised an alarm for a benign host, and a false negative if an alarm was never raised for an infected host. We calculate these metrics individually for each detector and also for a combination of all analysers - i.e. we consider an alarm to have been raised for a host if any one analyser raised an alarm.

Tables V, VI, VII and VIII show the confusion matrices with and detection rates obtained. The confusion matrices show the absolute number of hosts that were classified malicious or benign - for example, the "Malicious, Malicious" cell in the matrices represents the number of actually malicious hosts that were also classified malicious. We observe that the highest true positive rate (TPR) of 96% is obtained by the Failed Connection analyser, indicating that failed connections to the same destination is a behaviour commonly found in diverse botnet types, as our dataset contains a variety of IRC, P2P and HTTP botnets. The TPR for the blacklist analyser follows closely at 93%, but the DGA analyser lags behind at 85% TPR, which suggests that many botnets may not show synchronised failed DNS queries. This is not surprising because the DGA analyser is relevant only for botnets that use domain generation algorithms. Monitoring for synchronised behaviour using all analysers can yield up to 97% TPR, but at the cost of a high 27% false positive rate (FPR). FPR for the individual detectors is close to 15%, which is fairly high. These results demonstrate that synchronised behaviour is often, but not necessarily, correlated with botnet infections; some legitimate hosts may also coincidentally show such synchronisation. For example, this may happen when a server commonly accessed by legitimate hosts (such as an internal file server) is down.

TABLE V: NX Domain Analyser detection accuracy.

		Predicted: Malicious	Predicted: Benign
Actual	Malicious	92	15
	Benign	34	239

Detection Rates	
TPR	0.85
TNR	0.87
FPR	0.14
FNR	0.14
Accuracy	0.87

TABLE VI: Blacklist Analyser detection accuracy.

		Predicted: Malicious	Predicted: Benign
Actual	Malicious	99	8
	Benign	40	233

Detection Rates	
TPR	0.93
TNR	0.85
FPR	0.14
FNR	0.07
Accuracy	0.87

TABLE VII: Failed Connection Analyser detection accuracy.

		Predicted: Malicious	Predicted: Benign
Actual	Malicious	103	4
	Benign	42	231

Detection Rates	
TPR	0.96
TNR	0.85
FPR	0.15
FNR	0.03
Accuracy	0.88

TABLE VIII: All analysers combined detection accuracy.

		Predicted: Malicious	Predicted: Benign
Actual	Malicious	105	2
	Benign	74	199

Detection Rates	
TPR	0.97
TNR	0.73
FPR	0.27
FNR	0.02
Accuracy	0.80

B. Minimum Usefulness: Early infection detection potential

For this analysis, we divided the data into short time windows of thirty minutes and ran our analysers over each host's traffic in each window. For each host, we marked the detection time t_{A_i} for each analyser as the window in which it first generated an alarm for the host. We calculate the blacklist detection time t_B by mapping the exact time for the first detected C&C communication to the window it falls in. As specified in Section V-B, detection delay is then computed as $Del_{A_i} = t_{A_i} - t_B$. The *Early Alarms* bar in Fig. 3 shows the percentage of early alarms for each analyser A_i, which represents the percentage of hosts for which $Del_{A_i} \leq 0$, i.e. where the time of the first alarm from the analyser at least matches the blacklist detection time. In the figure, *All Combined* represents the percentage of hosts where t_{A_i} of any one analyser was less than the blacklist detection time. We observe that for the majority of hosts, our analysers were able to match blacklist detection time. The Blacklist Analyser matched the blacklist detection time for 78% of hosts, and the Failed Connection analyser followed closely at 75%; even the lowest performing DGA analyser matched blacklist detection times for 71% of hosts. When all analysers were run together, the number of hosts showing early infection detections climbed to 91%. We reiterate that this part of our analysis serves only to verify that a *minimum usefulness criterion* is met; the next section shows the results for strictly early alarms. Given that the current implementation is merely a proof-of-concept and analyses only three kinds of synchronised behaviour, the results show that this approach has good early infection detection potential and only rarely exceeds the blacklist detection time.

C. Correlation and temporal relationship between attacks and behavioural synchronisation

We now consider the core investigation of our empirical study: the extent to which attacks are correlated with synchronised behaviour, and the temporal relationship they bear. The temporal relationship is important to study because it determines whether observing synchronised behaviour can enable generating *early alerts* of attacks that are taking place, as well as *predictions* of attacks before they occur. Similar to the previous experiment, we again divide the trace into short time windows (30 minutes). We obtain the time of attacks from the Snort-labelled event sequences used for the empirical analysis presented in Section III, and for each host, mark the window in which we see the first instance of attack as the time of attack t_{att}. We then compare this time with the first-alarm time t_{A_i} of each analyser A_i and compute a detection delay as: $Del_{A_i} = t_{A_i} - t_{att}$ Good early attack detection potential of our approach is indicated if $Del_{A_i} \leq 0$ for most hosts; this would mean that we are able to generate alerts for attacks either before the attack or very soon after it occurs. On the other hand, a potential for attack *prediction* is only indicated if $Del_{A_i} < 0$ for most hosts, as the alarm has to come strictly before an attack for it to qualify as a prediction.

We now answer the questions from Section V-C as follows:

- *Question* 1*:* The *False Alarms* bar in Fig. 3 shows the percentage of infected hosts for which attack alarms are generated by an analyser but attacks do not materialise. The analysers show similar performance with false alarms between 12 and 16 percent. The maximum false alarm rate is 16% when using all analysers combined; while this does represent a rather large number of false alarms, it is offset by the potential gain of apprehending attacks early and can be handled in practice by defining a suitable policy for responding to alarms. For example, instead of blocking hosts as a result of attack alarms, which would lead to the falsely identified legitimate hosts being blocked as well, an ISP or network administrator could rate-limit the hosts and generate a warning to disinfect the system.
- *Question* 2*:* This concerns the true alarm rate, i.e. instances where we see alarms from our analysers for hosts that are actually carrying out attacks. The *True Alarms* bar in Fig. 3 shows that in the majority ($> 83\%$) of the cases, alarms are generated for hosts that are engaging in attacks. The DGA analyser generates the maximum true alarms (more than 86%) with the Failed Connection and

Blacklist analysers following closely at 85% and 84%. We conclude from the answers to questions (1) and (2) that a significant correlation exists between attacks and synchronised behaviour.

- *Question* 3*:* This question addresses the issue of how often alarms occur strictly before attacks, at almost the same time as attacks, and after attacks have been launched. The *Early or Same Time Alarm* bar in Fig. 4 shows that when using all analysers we are able to generate alarms before or very close to the beginning of the attack (i.e. in the same thirty-minute window) for 77.1% of the hosts, while there is no host for which the alarm comes strictly after the attack. The DGA and Failed Connection analysers, when running alone, generate same-time alarms for about 63% of the hosts, while for the Blacklist Analyser this value is limited to 58%. Unfortunately, the *Early Alarm* bar in the same figure shows that all three analysers generate strictly early attack alarms for only 15% of the hosts, and when we use all analysers together, we are able to generate strictly early alarms for 20% of the hosts. The proof-of-concept implementation, therefore, has very limited attack prediction capability.
- *Question* 4*:* Regarding the best and worst analysers for generating *early* warnings, as we observe exactly the same performance across analysers (15% of attacks predicted accurately before they occur), there is no difference in early warning capability. Regarding attack warnings that are generated very close to but not before the time of the attack, the Blacklist analyser performs slightly worse; it is also the only analyser which generates a small percentage (2%) of late alarms.

Overall, our approach appears to be promising for detecting attacks quickly after they are launched; even the proof-of-concept implementation, which is limited to analysing behavioural synchronisation only for the C&C communication stage, is able to detect 77% of attacks at the same time or very close to the time that they are launched. However, for *predicting* attacks, a significant improvement is needed, as currently we are only predicting 20% of attacks well before they occur. We believe that an improvement in this capability can be achieved by monitoring more kinds of behavioural synchronisation at all stages of infection, and particularly at the earliest inbound exploit and binary download stages.

D. Summary of Findings

This empirical study is an effort towards understanding the behavioural synchronisation observed in botnets and its correlation with attacks. We have not presented a comprehensive botnet detection or attack prediction solution; rather the contribution of this work lies in our measurement-driven analysis of real-world botnet activity and the investigation of the potential of our proposed approach. Nevertheless, our findings and conclusions summarised below can be of practical benefit in fostering future research in this area.

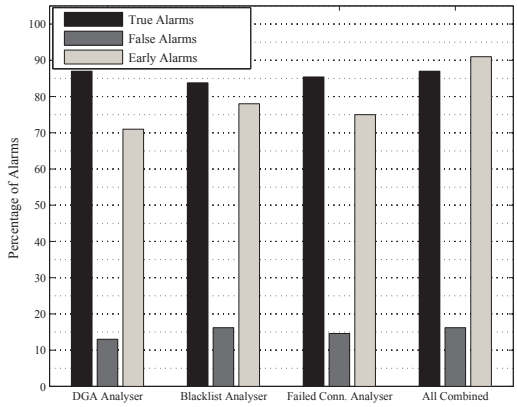

Fig. 3: *Early Alarms*: percentage of hosts on which infections were detected no later than a commercial blacklist. *True Alarms*: percentage of hosts showing synchronised behaviour as well as attacks. *False Alarms*: percentage of hosts showing synchronised behaviour but no attacks.

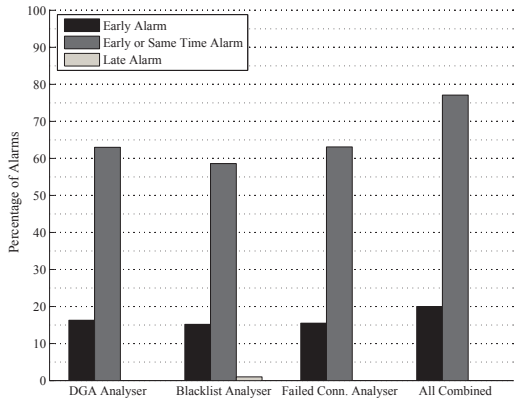

Fig. 4: Percentage of hosts showing "true" alarms that come before, with and after attacks

1) Based on a dataset of 380 hosts including 107 machines infected with a diverse range of botnets, we verified the popular understanding that C&C communication is the behaviour that most commonly precedes outgoing attacks by bots; 86.4% instances of C&C communication were followed by attacks in our dataset.
2) The majority of bots engage in some form of synchronised behaviour; 96% bots in our dataset showed synchronisation in failed connections. However, synchronisation in C&C communication cannot be used alone as a measure of detecting botnets as it is highly susceptible to false alarms; 27% of hosts showing at least one form of synchronised behaviour in our dataset were not infected.
3) Monitoring a network for behavioural synchronisation can prove useful in identification of infections; we were able to identify 91% of infections at the same time or earlier than a commercial blacklisting service.

4) The majority of bots that launch attacks also show some form of synchronised behaviour (83% in our dataset); however, generating attack warnings on the sole basis of synchronised C&C communication can yield some false alarms (16% in our dataset).
5) Finally, while we were able to detect 77% of attacks very close to the time they are launched, predicting attacks strictly before they occur requires further work. For the present, our proof-of-concept implementation only achieves a prediction capability of 20%.

VII. Conclusion and Future Directions

In this work we presented an empirical analysis of the synchronised behaviour that is observed from the members of a botnet and studied the temporal relationship between synchronised behaviour and attacks. As a proof-of-concept implementation, we designed a number of analysers that monitor a network for synchronised C&C communication, and investigated the potential of using synchronised behaviour as an indicator of infection as well as of upcoming attacks. Using traffic from 107 botnet infections representing 13 different botnet families, we compared the detection accuracy of our approach with a commercial blacklist and showed that it achieved at least comparable accuracy. We also showed that 83% of hosts that engaged in attacks also showed some form of synchronisation in their C&C communication, and that with this approach, we were able to quickly detect 77% of attacks. However, as our current implementation detects synchronised behaviour within a very limited scope, we were only able to predict 20% of attacks strictly before they occurred.

This brings us to a discussion of open problems for future research. The main limitation of our current work is the limited scope of synchronised behaviour; instead of monitoring synchronisation in only C&C communication, considering earlier infection stages such as incoming exploits and binary downloads, which are usually expected to occur before C&C communication, should assist in generation earlier warning of attacks. This would require a dataset that captures the complete infection footprint and the full duration of botnet infections, a property that is rarely found in datasets such as ours capturing in-the-wild botnet activity, as some hosts may already be infected when the trace capture is started. For example, one possible direction is to investigate whether multiple hosts compromised by the same exploits (e.g. a particular OS vulnerability) engage in attacks in future. It would also be worthwhile to use a more recent dataset, perhaps including mobile botnets, and analyse whether the same behavioural patterns are observed. Furthermore, we leave as future work a comparison of the accuracy and timeliness of detection of synchronised behaviour using our analysers with other similar solutions, such as BotMiner and BotSniffer [12], [14], which also detect synchronised C&C communication patterns as one indicator of a widespread infection, because these solutions are not available publicly and require significant implementation effort. In addition, comparing the running time of our analysers with these solutions remains as future work.

References

[1] Emerging Threats. https://rules.emergingthreats.net/. [Online; accessed 18-Jan-2017].
[2] Malware Domain List's IP Blocklist. http://www.malwaredomainlist.com/hostslist/ip.txt. [Online; accessed 27-Sep-2012].
[3] PalevoTracker's IP Blocklist. https://palevotracker.abuse.ch/blocklists.php?download=ipblocklist. [Online; accessed 27-Sep-2012].
[4] SpyEye's IP Blocklist. https://spyeyetracker.abuse.ch/blocklist.php?download=ipblocklist. [Online; accessed 27-Sep-2012].
[5] Systems and Networks Research Lab. https://sysnet.lums.edu.pk/. [Online; accessed 18-Jan-2017].
[6] Talos Ruleset. https://snort.org/talos. [Online; accessed 18-Jan-2017].
[7] Team Cymru. http://www.team-cymru.org/. [Online; accessed 18-Jan-2017].
[8] ZeusTracker's IP Blocklist. https://zeustracker.abuse.ch/blocklist.php?download=ipblocklist. [Online; accessed 27-Sep-2012].
[9] Moheeb Abu Rajab, Jay Zarfoss, Fabian Monrose, and Andreas Terzis. A multifaceted approach to understanding the botnet phenomenon. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 41–52. ACM, 2006.
[10] Manos Antonakakis, Roberto Perdisci, Yacin Nadji, Nikolaos Vasiloglou II, Saeed Abu-Nimeh, Wenke Lee, and David Dagon. From throw-away traffic to bots: Detecting the rise of dga-based malware. In *USENIX security symposium*, volume 12, 2012.
[11] Ayesha Binte Ashfaq, Zainab Abaid, Maliha Ismail, Muhammad Umar Aslam, Affan A. Syed, and Syed Ali Khayam. Diagnosing bot infections using bayesian inference. *Journal of Computer Virology and Hacking Techniques*, pages 1–18, 2016.
[12] Guofei Gu, Roberto Perdisci, Junjie Zhang, Wenke Lee, et al. Botminer: Clustering analysis of network traffic for protocol-and structure-independent botnet detection. In *USENIX Security Symposium*, volume 5, pages 139–154, 2008.
[13] Guofei Gu, Phillip A Porras, Vinod Yegneswaran, Martin W Fong, and Wenke Lee. BotHunter: Detecting malware infection through IDS-driven dialog correlation. In *USENIX Security*, volume 7, pages 1–16, 2007.
[14] Guofei Gu, Junjie Zhang, and Wenke Lee. Botsniffer: Detecting botnet command and control channels in network traffic. 2008.
[15] Takayuki Hirayama, Kentaroh Toyoda, and Iwao Sasase. Fast target link flooding attack detection scheme by analyzing traceroute packets flow. In *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*, pages 1–6. IEEE, 2015.
[16] Sheharbano Khattak, Zaafar Ahmed, Affan A Syed, and Syed Ali Khayam. Botflex: A community-driven tool for botnet detection. *Journal of Network and Computer Applications*, 58:144–154, 2015.
[17] Vern Paxson. Bro: a system for detecting network intruders in real-time. *Computer networks*, 31(23):2435–2463, 1999.
[18] Salvatore Pontarelli, Giuseppe Bianchi, and Simone Teofili. Traffic-aware design of a high-speed fpga network intrusion detection system. *IEEE Transactions on Computers*, 62(11):2322–2334, 2013.
[19] Martin Roesch et al. Snort: Lightweight intrusion detection for networks. In *LISA*, volume 99, pages 229–238, 1999.
[20] Matthias Vallentin, Robin Sommer, Jason Lee, Craig Leres, Vern Paxson, and Brian Tierney. The nids cluster: Scalable, stateful network intrusion detection on commodity hardware. In *International Workshop on Recent Advances in Intrusion Detection*, pages 107–126. Springer, 2007.
[21] Ricardo Villamarín-Salomón and José Carlos Brustoloni. Bayesian bot detection based on dns traffic similarity. In *Proceedings of the 2009 ACM symposium on Applied Computing*, pages 2035–2041. ACM, 2009.
[22] Xiaofei Wang, Yang Xu, Junchen Jiang, Olga Ormond, Bin Liu, and Xiaojun Wang. Strifa: Stride finite automata for high-speed regular expression matching in network intrusion detection systems. *IEEE Systems Journal*, 7(3):374–384, 2013.
[23] Junjie Zhang, Roberto Perdisci, Wenke Lee, Unum Sarfraz, and Xiapu Luo. Detecting stealthy p2p botnets using statistical traffic fingerprints. In *Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference on*, pages 121–132. IEEE, 2011.
[24] David Zhao, Issa Traore, Ali Ghorbani, Bassam Sayed, Sherif Saad, and Wei Lu. Peer to peer botnet detection based on flow intervals. In *IFIP International Information Security Conference*, pages 87–102. Springer, 2012.
[25] David Zhao, Issa Traore, Bassam Sayed, Wei Lu, Sherif Saad, Ali Ghorbani, and Dan Garant. Botnet detection based on traffic behavior analysis and flow intervals. *Computers & Security*, 39:2–16, 2013.