Available online at www.sciencedirect.com

ScienceDirect

00 (2016) 1–19

Journal Logo

Survey of Approaches and Features for the Identification of HTTP-Based Botnet Traffic

Dilara Acarali[a], Muttukrishnan Rajarajan[a], Nikos Komninos[a], Ian Herwono[b]

[a]*School of Engineering and Mathematical Science, City University London, London, United Kingdom.*
[b]*Security Futures Practice, Research & Innovation, British Telecom, Ipswich IP5 3RE, United Kingdom.*

Abstract

Botnet use is on the rise, with a growing number of botmasters now switching to the HTTP-based C&C infrastructure. This offers them more stealth by allowing them to blend in with benign web traffic. Several works have been carried out aimed at characterising or detecting HTTP-based bots, many of which use network communication features as identifiers of botnet behaviour. In this paper, we present a survey of these approaches and the network features they use in order to highlight how botnet traffic is currently differentiated from normal traffic. We classify papers by traffic types, and provide a breakdown of features by protocol. In doing so, we hope to highlight the relationships between features at the application, transport and network layers.

Keywords: Bot, botnet traffic, network analysis, feature analysis, network-based detection

1. Introduction

The digital age has brought many benefits to society, with the Internet acting as an enabler for growth and development across practically all sectors of business and industry. Unsurprisingly, it has also become an attractive and fertile environment for criminal activity. Malware programs, which may take many forms, are now frequently used for financial theft, identity theft, espionage, and disruption of services. A particularly troublesome type of malware is a bot, an exploited system which acts as a remote tool for an attacker to control and use the resources of a target system. Typically, the attacker (called the botmaster) will do the same for multiple systems and then use them collectively, in what is known as a botnet, to launch attack campaigns.

Botnet classification is based on the type of communication protocol used. The main classes currently defined by the research community are P2P-based, IRC-based, and HTTP-based. P2P-based botnets use a decentralised architecture, in which each node can act as both a client and a server. In contrast, both IRC and HTTP-based botnets operate with a centralised architecture, where bots are required to connect to a command and control (C&C) server in order to upload data or receive commands. For this study, we have chosen to focus on HTTP-based botnets. The wide-spread use of web-based services has pushed the adoption of HTTP as an advantageous communication protocol thanks to the fact that it is usually permitted by firewalls, and it allows bot traffic to blend in with vast volumes of benign activity. The benefits of using HTTP rather than IRC are demonstrated and discussed by Farina et al. (2016) and Gu et al. (2008). Additionally, McAfee (2015) listed at least 5 HTTP-based botnets as top spammers in their 2014 Q4 Threat Report. This was also echoed by Symantec (2014), where HTTP-based botnets made up over half of their 2014 top 10 spamming botnets.

The difficulty of detecting botnet traffic (especially for HTTP-based botnets) amongst web activity is a current research challenge. Rostami et al. (2014) compared clean network traces to those collected from HTTP-based botnets. They focused on HTTP data in client-side PCAP files to highlight differences in clean and malicious traffic at the application layer. However, they do not consider how this traffic may manifest at other layers, nor how it may be measured. Both Haddadi & Zincir-Heywood (2014) and

Email addresses: dilara.acarali@city.ac.uk (Dilara Acarali), r.muttukrishnan@city.ac.uk (Muttukrishnan Rajarajan), nikos.komninos1@city.ac.uk (Nikos Komninos), ian.herwono@bt.com (Ian Herwono)

Beigi et al. (2014) study the effectiveness of flow-based features in combination with machine learning techniques for botnet detection. The former analyses the use of flow exporters (used to extract and aggregate flows), whilst the latter considers the use of different flow-based features as inputs to machine learning algorithms. Whilst they provide interesting insights, these works consider neither features from the application layer to help to characterise the malware, nor the behaviours which may underpin them. Meanwhile, Garcia et al. (2014) conduct a general survey on network-based detection methods, including a categorisation of algorithms and in-depth analysis of key works. However, they do not identify the specific features those approaches use.

The aim of this work is to provide a study of the characteristics of HTTP-based botnet traffic and the types of features used to measure or detect them. Such features or identifiers are the inputs to many detection mechanisms, regardless of the methodology used. Therefore, it is important to have a clear view of both what these identifiers may be as well as how they fit together. To achieve this, we conduct a survey of recent network-based detection methods, with specific focus on the types of traffic they observe and the parameters they define for the distinction of benign and malicious network data. Identified features are abstracted into sets, and classified by protocols and the associated layers of the OSI model. Our goal is to enable a better understanding of the nature of HTTP-based botnet traffic, including the underlying behaviours which cause it and the implementation of different protocols. (Note that the term "HTTP-based botnet" refers to those that communicate with C&Cs over HTTP, but they may use other protocols for other activities).

The main contributions of this paper are as outlined below:

- A survey of recent network-based detection approaches for HTTP-based botnets
- A survey of traffic-based features used to detect bot traffic
- An abstraction of the main types of features in relation to protocols and OSI layers

The paper is organised as follows: Section 2 will provide a background on the observed behaviours of HTTP-based botnets and the traffic which is generated as a result of those behaviours. In Section 3, we will examine the works that attempt to characterise those different types of traffic. Section 4 is a look at sets of measurable features used by existing works, broken down by protocol. In Section 5 we discuss the implications of our findings and possibilities for future work. Section 6 provides conclusions.

2. Behaviour of HTTP-Based Botnets

In order to interpret the traffic generated by bots, we first need to understand the activities that they may be engaging in. In this section, we consider sequences of events in a typical botnet's lifecycle, and split them into groups based on the aims behind those events. We then consider what kind of traffic may be observed for each of these groups.

2.1. Lifecycle

Botnets inherently exhibit similar behaviours over the course of their active lifespan. These behaviours can be abstracted into a number of sequential stages to form a model of the botnet lifecycle. The stages may be slightly different depending on whether behaviours are viewed from the perspective of the botmaster or of the defenders. Rodriguez-Gomez et al. (2013) consider the former approach, and define the lifecycle as Conception, Recruitment, Interaction, Marketing, Attack Execution and Attack Success. Conception covers the initial design and development of the malware, including the botmaster's motivations. The Recruitment stage is the infection process through which new bot victims are acquired. Next, the Interaction stage is where communication channels are established, including those between the bots and C&C servers. Marketing is (as the name suggests) the advertisement of the botnet or its services. Attack Execution will then depend on the chosen campaign, and the end of this stage will be marked by Attack Success.

Meanwhile, the stages as defined by Silva et al. (2013) are Initial Infection, Secondary Injection, Rally, Malicious Activities, and Maintenance. Initial Infection is the stage where the bot malware first comes into contact with a vulnerable node, and roughly aligns with the previous model's Recruitment stage. Then, a Secondary Injection is triggered, where the executed malware downloads additional configuration files. Rally denotes the initial connection attempts made by bots to C&Cs from which they will receive further instructions. This aligns with Interaction in the previous model. The commands of the botmaster will dictate the botnet's behaviours during Malicious Activities (similarly to Attack Execution and Attack Success). Maintenance covers the process of updating or patching the botnet.

The lifecycle conceptually encapsulates botnet behaviours in stages, based on the motivation behind those behaviours. This means that the traffic which manifests as a result can be similarly compartmentalised, as defined in Figure 1. In this study, the stages from the two models which are most likely to generate traffic are combined as Propagation, Rally, Interaction, and Attack. Propagation encompasses the proliferation and infection of the malware (covering the Recruitment, Initial Infection, and Secondary Injection stages). Then, the establishment of

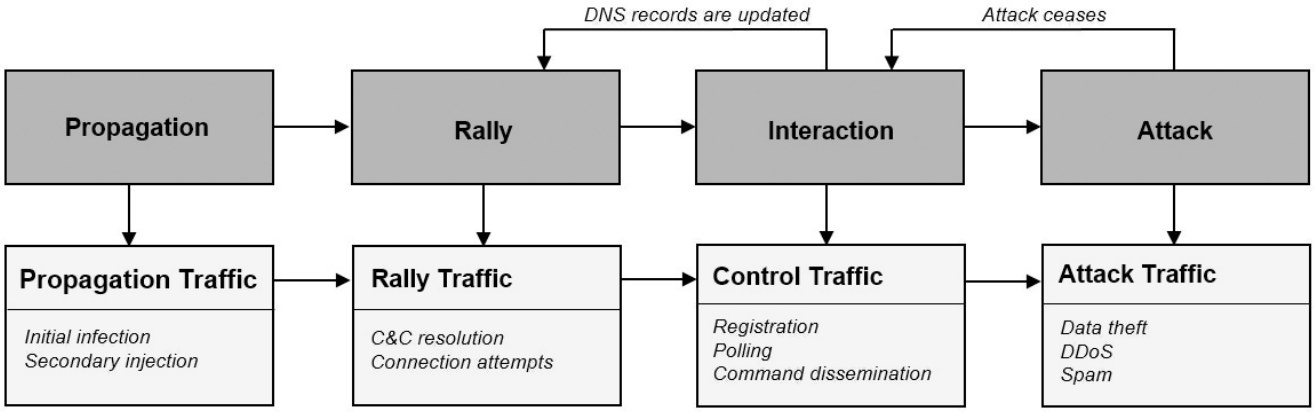

Figure 1. Lifecycle model adapted from Rodriguez-Gomez et al. (2013) and Silva et al. (2013), with definitions of traffic types for each stage.

control channels is covered by the Rally stage. Interaction denotes control traffic between bots and the C&Cs (including the dissemination of commands, as well as updates), followed by Attack to cover Malicious Activities and related traffic.

2.2. Behaviours & Observed Traffic

As a bot progresses through the lifecycle, it will execute different tasks. Traffic should be generated at each step, with characteristics reflective of what the bot is trying to accomplish. This section provides a close-up of possible traffic that may be observed at each stage of the lifecycle.

2.2.1. Propagation Behaviours

A botnet accumulates resources through the infection of vulnerable systems. HTTP-based bot binaries are spread to victims through propagation vectors including drive-bys (e.g. Asprox, observed by Borgaonkar (2010)), spam and social engineering (e.g. KoobFace, reported by Thomas & Nicol (2010)), and pay-per-download campaigns (e.g. DirtJumper, reported by Andrade & Vlajic (2012)). An executed bot binary will begin to make changes on the host system to obfuscate itself. Some may trigger a secondary injection, contacting an embedded target from which additional configuration files are downloaded (Silva et al., 2013). The specific dynamics of the infection process vary between malwares, delivery vectors, and campaigns. These complexities make it difficult to separate bot infections from other types of malware. Therefore, propagation behaviours are considered outside of our scope for this work. We believe that this is a larger topic which warrants a separate study.

2.2.2. Rally Behaviours

Bots which have successfully infected a victim will try to rally to the relevant C&C servers. This process is likely to be repeated on multiple occasions in a bot's lifetime because of rebooted hosts (Silva et al., 2013) or servers with changing domains or IPs. In order to make the initial connection, some bot binaries are hardcoded with a list of domains, which they will attempt to resolve (e.g. Citadel, studied by Rahimian et al. (2014)). Alternatively, a domain generation algorithm (DGA) may be implemented (as seen in Zeus, Citadel, and ICE1X, reported by Sood et al. (2014)). DGAs are given a seed value and automatically produce sets of "pseudo-random" domain names. Both the bots and the botmaster must use the same seed for their results to match (Sood et al., 2014). This adds a layer of obfuscation by providing a kind of camouflage for the registered domains (of which there will be only a few). DGAs also enable the use of another obfuscation technique known as domain fluxing, where C&Cs frequently switch their domain names (Sood et al., 2014). Alternatively, botnets may use fast-fluxing, where the IP addresses of servers are switched instead.

2.2.3. Interaction Behaviours

Once a successful connection with a C&C is established, bots can share and receive data from the botmasters. In some botnets, new members are required to register themselves with the C&Cs (e.g. DirtJumper bots include a 15-digit identification number in their first HTTP request, reported by Andrade & Vlajic (2012)). For command dissemination, HTTP-based botnets utilise a pull-based approach, where members must initiate their own connections and continuously poll the C&Cs for updates. Asprox (Borgaonkar, 2010), DirtJumper (Andrade & Vlajic, 2012) and Zeus (Binsalleeh et al., 2010) all display such polling behaviour. The advantage in doing so is that perimeter controls such as firewalls are circumvented. Most firewalls allow outgoing connections on HTTP ports 80 or 443 for standard web traffic. Data that bots receive from the C&Cs may include configuration files, updates (for maintenance purposes), or attack instructions (e.g. Asprox bots

receive JavaScript files for drive-by-downloads, reported by Borgaonkar (2010)). Bots may also send data back to the server, including log information on running attacks (Farina et al., 2016) or users' personal details harvested from the victim node.

2.2.4. Attack Behaviours

Bots will then engage in attacks depending on the nature of the instructions that they receive. HTTP-based botnets have been observed conducting a variety of attack campaigns, including DDoS (e.g. BlackEnergy, reported by Shiaeles et al. (2012)), web injects and man-in-the-browser attacks (e.g. Zeus and Citadel (Sood et al., 2014)), SQL injections (e.g. Asprox, studied by Borgaonkar (2010)), and many others. The kinds of traffic which are generated will naturally depend on the nature of the attack.

Table 1 provides a summary of some reported bot behaviours and the corresponding traffic which is generated by them.

3. Characterisation & Identification of HTTP-Based Botnet Traffic

The traffic generated by HTTP-based botnets is characteristic of their underlying behaviours. Existing network-based detection approaches are therefore built on this understanding in order to distinguish between benign and malicious traffic. In this section, current detection approaches tailored for (or applicable to) HTTP-based botnets are surveyed according to the types of traffic they aim to identify.

3.1. Identification of Rally Traffic

Sharifnya & Abadi (2013) study rally behaviours related to domain-fluxing botnets like Kraken and propose a reputation system to identify the use of DGAs. Bots are expected to generate a number of domain names, each of which will be queried until one can be resolved. The system groups DNS queries by similarity, and then examines them for domains which appear to be algorithmically generated. For a series of observed time windows, they extract the hosts who a) query one of those domains and b) generate a high number of query failures. These two properties are then combined to calculate the host's negative reputation score. Hosts who appear in multiple time windows, or behave as part of a group are considered more suspicious and given higher scores.

Domain-fluxing is also the focus of Schiavoni et al. (2014), who propose a system called Phoenix. They model benign domains by their adherence to language rules (described as being pronounceable). Malicious domains in DNS traffic are identified using blacklists, and the queries related to those domains are extracted. Both the domains and their traffic are used in a semi-supervised learning approach to identify the ones which do not match the benign model. The expectation is that domains generated by DGAs will be random and hence not made up of real words. Phoenix then clusters malicious domains by their IP-mappings. Each cluster is assumed to represent a single algorithm, and is therefore used to generate a fingerprint of that underlying DGA. The authors use the fingerprints for detection, and report good results when testing on DNS traffic from botnets such as Torpig and SpyEye (Schiavoni et al., 2014). However, they note that this approach is ineffective against unregistered domains, and is also language specific. Hence, domain names from another language may cause false positives (Schiavoni et al., 2014).

3.2. Identification of Control Traffic

The periodic nature of communication between bots and C&Cs is used heavily in the detection of interaction behaviours. Wang et al. (2010) claim that bots should be automated and systematic, repeating sets of behaviours with regular intervals. Based on this, they characterise C&C traffic as a series of similar periodic HTTP messages exchanged between bots and their servers. Clustering is used to find statistically similar flows within TCP sessions, excluding those with incomplete handshakes or empty payloads. Clusters containing patterns of periodic behaviour are then compressed into signatures to be used in future detections. When testing on Kraken, Zeus, and BlackEnergy, the authors found that each malware family had similar periodic patterns, suggesting that it may even be possible to differentiate between traffic originating from different botnets. The main drawback of this approach is that botnets displaying random communication patterns may be overlooked.

Soniya & Wilscy (2013) describe normal HTTP traffic as bursty, meaning that a large number of connections are made to the same destinations over a short period of time. According to their research, botnet traffic (generated in a scheduled and automated manner) should be more evenly distributed over time. They propose a detection scheme which identifies C&C communication by the regularity of flow timings and sizes. Flows are extracted from both normal and botnet traffic (collected from samples of BlackEnergy and Zeus), and aggregated by target destination. Then the entropy of flow sizes and flow timings are calculated and used to train a neural network classifier. Despite only using flow-level analysis the authors report good detection results (97.4% detection rate, with false positive rate of 2.5%). They suggest that the false positives (caused mainly by software updates) can be minimised using whitelisting.

Etemad & Vahdani (2012) take a similar approach for centralised botnet detection. Like Wang et al. (2010), they identify C&C traffic by the presence of periodic HTTP messages. They highlight that no connections are maintained from one transaction to another, so bots contin-

Table 1. Summary of some observed bot behaviours, and the traffic which is generated as a result.

Reported Behaviours	**Traffic Expected**
Rally	
Attempts to resolve C&C domains	Bots will generate DNS queries, with failed queries due to inactive C&Cs
Use of fluxing technology	Bots will generate DNS queries for new C&Cs every time fluxing occurs
Control	
Authentication with the C&C	Similar initial requests sent by bots from the same botnet
Command dissemination	Specially-crafted packets containing specific command codes and information
Provision of attack resources	Similar responses/payloads sent to bots by C&Cs
Constant polling of C&C servers	Repeated POST queries sent by bots on HTTP ports
Attack	
Data exfiltration	Similar outbound packets, with possible use of encryption
Distributed denial of service (DDoS)	Sudden flux in outgoing (for bots) or incoming (for servers) HTTP requests

uously make new connections in order to retrieve commands. The system extracts HTTP traffic from network data using a protocol analyser, and then identifies exchanges where hosts have made GET, POST, or HEAD requests, and the servers have responded. A HTTP analyser is then used to determine the level of periodic repeatability in similar messages. Rather than examining packets in their entirety, message similarity is determined by observing only the beginning of each packet (i.e. the first sequence of bytes) for patterns or keywords. However, as noted by the authors, this work also suffers from the fact that random HTTP bot traffic will likely go undetected.

Another issue is the number of false positives caused by traffic which is both benign and periodic. Eslahi et al. (2013) propose a method to address this as well as the detection of random patterns of behaviour. Based on the typical characteristics of botnet activities, they define a HAR (High Access Rate) and LAR (Low Access Rate) filter. In effect, these filters are designed to provide upper and lower bounds of suspicious periodicity. When applied to web traffic, the HAR filter removes HTTP connections which are considered too frequent and therefore likely to be generated by automated software updates. Meanwhile, the LAR filter removes periodic HTTP connections which happen too sporadically to be generated by bots (Eslahi et al., 2013). The result is a distinction between benign and malicious types of periodicity. The authors report that this method was successful in reducing the amount of data (removing up to 99.6% of the original test packets), but false positives were still caused by repetitive user behaviours. They propose to overcome this by incorporating the use of User-Agent fields as a feature (Eslahi et al., 2013), though this will be ineffective if botnets use HTTPS.

Levels of periodicity are explored again by Eslahi et al. (2015). In this work, the authors aim to better characterise the types of periodic traffic most likely to be associated with botnets. They extract HTTP traffic featuring GET or POST requests and group by message similarity. Botnet traffic is then identified using three metrics: periodic factor (PF) which captures repetition, range of absolute frequencies (RF) which captures the frequency of events, and time sequence factor (TF), which captures event distribution. According to the authors, events in botnet traffic should repeat across multiple time windows, be generated a uniform number of times, and be observed at equal time intervals. This demonstrates the automated, systematic nature of bots. The values from each metric are then combined using decision trees to give traffic a final periodicity classification. This work improves upon others by giving a clearer definition of periodic botnet traffic. However, false positives were reported once again, with the authors concluding that periodicity should be complimented by other measures for better results.

In an alternative approach, Lu & Brooks (2011) propose the use of network traffic analysis combined with hidden Markov models to differentiate the C&C behaviours of Zeus from normal traffic. As before, the defining feature is considered to be periodicity, this time captured in the form of inter-packet delays. These measurements are used to generate a hidden Markov model to reveal patterns in packet sequences. The authors expect members of the same botnet to have similar behavioural patterns. Therefore, detection is achieved by mapping data to the model to test transition probabilities. This work demonstrates the flexibility of using network traffic, elements of which can be used as inputs for different detection methodologies or systems. However, aside from only being tested with Zeus, the obvious limitation is that random behaviours may once again be unaccounted for.

Instead of focusing on observing low level flows or HTTP packets, Venkatesh & Nadarajan (2012) propose the use of the TCP protocol to characterise control information at the transport layer. TCP sessions underpin the exchange

of HTTP data at the application layer. Therefore, the same patterns of repetition should be observable. Additionally, bots attempting to make connections to C&Cs won't always receive responses. Therefore, the authors suggest that there should be a large number of half-open connections and a larger number of outgoing TCP sessions. They extract TCP information from both normal and botnet traffic, which is then used to train a neural network classifier. This method is reported to achieve accurate results when applied to traffic samples for Zeus and SpyEye, showing that TCP features provide a plausible alternative to other approaches. The main drawback is that without packet-level analysis, individual botnets cannot be characterised.

Li et al. (2014) suggest that botnets implementing the HTTP protocol often use poorly formed headers. Therefore, deviations from protocol standards can be used for detection. Network data is first clustered into groups of similar flows. According to the authors, the packets exchanged at the start of C&C communications often contain sensitive details (e.g. IPs, bot IDs, and process names). Therefore, they reduce the volume of data by extracting only the first request and response packets from HTTP flows. The flows are analysed for periodicity, whilst packet headers are examined for missing fields and malicious keywords. The system combines the results of each check to calculate a suspiciousness score for that flow. If scores exceed a threshold, the captured packets are used to generate detection signatures. Whilst this is a good example of how flow and packet-level inspection can be combined, such an examination of packet headers may not be possible if data is encrypted.

Grill & Rehak (2014) study HTTP header integrity as well, specifically looking at User-Agent fields for their proposed system. This field provides information on the application that created the HTTP request, but its inclusion is not enforced by the protocol. User-Agent strings are also generated at run-time using details of the host OS environment. The authors suggest that this is difficult for botmasters to accurately replicate, so that User-Agent will either be omitted completely or will use the same values each time. For network data collected from web proxy logs, they separate messages into 3 groups: those with no User-Agent field (Empty), those which use strings based on browser types (Specific), and those with strings not based on browser types (Discrepant). They note that spoofed strings will not be detectable with this method. Each group is then analysed individually to find suspicious hosts. For the Empty and Specific types, the system looks for the least popular domains or the least frequent strings and considers those anomalous. For the Discrepant group, the host is checked for recent updates. If none are found, this is also considered anomalous. According to the authors, the most frequently seen type of fake User-Agent field was Discrepant. The advantage of this method is that it can be applied to proxy logs without needing flow-level analysis. However, it will be ineffective against encrypted traffic.

Meanwhile, Cai & Zou (2012) aim to use a combination of features in their proposed detection system, making a number of observations about the difference between normal and malicious traffic. They characterise bot requests by the similarity of the target services, URIs, files, and queries. HTTP traffic is clustered by request similarity, with the least popular request strings being considered more suspicious. A list of suspect servers is extracted from those clusters, and analysed for additional suspicious traits, including the similarity of HTTP headers, the size of responses from C&C servers, and the location of source IPs. Like Li et al. (2014), the authors suggest that HTTP headers generated by bots will be shorter in length due to non-mandatory fields being omitted by the malware. Also, C&C responses (consisting of configuration files or commands) should be shorter than web pages delivered by legitimate servers. Lastly, they report that bot source IPs tend to concentrate within certain subnets due to local propagation. The output of the system is a collection of suspected hosts and servers. Despite using a combination of features, they still report false positives caused by error messages and the use of the HTTP protocol for other purposes. They suggest the use of whitelisting and additional features to overcome this. Overall, the system is considered effective at finding unknown botnets.

3.3. Identification of Attack Traffic

Malware attacks cover a very large range of activities. A central botnet attack function is data exfiltration, which is studied by Al-Bataineh & White (2012). They observe Zeus, whose members are known to use keylogging, screen captures, and web injects to collect sensitive data on their victims (Sood et al., 2014) which is then sent to the C&C servers. Whilst this may appear similar to control traffic, the focus here is not on the way bots acquire updates or commands. Therefore, measures such as periodicity and consistent packet sizes may not be applicable. HTTP POST messages originating from hosts within the observed network are extracted from web proxy logs, and checked for encrypted payloads. Unlike Cai & Zou (2012), Li et al. (2014), and Grill & Rehak (2014), the authors report that in their observations botnet HTTP headers did not appear anomalous. Instead, they consider encrypted payloads in outbound HTTP POST messages to be suspicious. Based on this, a decision tree classifier is used for detection. Despite being modelled on Zeus, this approach may also be applicable to others which operate similarly, especially successors like Citadel. However, botnets which do not use encryption may go undetected.

Meanwhile, Shiaeles et al. (2012) propose a method to detect DDoS attacks, such as those implemented by BlackEnergy. They consider DDoS attacks to develop quickly in

stages until they reach a peak level of disruption. Therefore, the aim is to detect both the early stages and the participating bots in order to mitigate damage. The approach is based on the finding that user-generated traffic in a very short time frame (of up to a few minutes) follows a Poisson distribution. They extract TCP packets (with HTTP payloads) and examine packet arrival times for each time frame. Then, using fuzzy estimators, the distribution of incoming packets is compared to the historical Poisson model to detect anomalies. After a DDoS attack is identified, the same process is repeated for the packet source IPs to find which hosts are generating the largest amount of traffic. The authors report good performance when tested on hPing and BlackEnergy. They note that spoofed IPs or hosts behind perimeter defences may generate false positives, but prioritise the detection of DDoS in small time frames, and suggest the use of integrated anti-spoofing approaches and additional features for increased accuracy.

Xiang et al. (2011) aim to detect low-rate DDoS, a stealthier attack that uses a lower traffic rate and slower pace to stay hidden. For their router-based approach, they propose two new metrics called generalised entropy and information distance for finding uniform behaviour, focusing on similar packet sizes and similar source IP ranges. Traffic through the routers is sampled in 300 second intervals and the probability distribution (of either packet sizes or source IPs) is computed for each LAN. Generalised entropy seeks to improve on Shannon's entropy by adding an adjustable parameter to influence whether higher or lower probability events will have greater impact on the final result. Meanwhile, information distance compares the probability distributions across LANs, with the aim of detecting the DDoS in the minimum number of hops. If the results are above given thresholds, an attack is detected, which then triggers an IP traceback algorithm to uncover the bot sources behind the attack. The authors tested on clean traffic from the MIT Lincoln Laboratory Scenario and low-rate DDoS traffic from CAIDA, and report fewer false positives with an earlier detection capability compared to Shannon's entropy and Kullback-Leibler's divergence. The advantage of this approach is its ability to pick out attack traffic using only network layer features, as well as an integrated method to identify malicious source nodes. However, this scheme requires control of all routers in the network which may not be possible in shared or larger networks.

In another related work, Xiang et al. (2009) discuss a deterministic IP traceback system called FDPM (Flexible Deterministic Packet Marking) for identifying DDoS packets sources. The system is designed to be deployed at the local router, and utilises certain bits in the IP header (within the Type of Service, Fragment ID and Reserved Flag fields) to 'mark' a stream of packets with the original source IP address. Marks may be of variable lengths, making the approach adaptable to varying network loads and the use of different protocol combinations. If router load increases above a threshold, the system switches to flow-based marking, focusing on flows with close time proximity, that are going to the same destination, and consuming a larger proportion of the available bandwidth. To reduce the number of false positives, a low pass filter removes instances of benign but significant "short-term fluctuations" (Xiang et al., 2009). This work provides a positive approach to deal with IP spoofing. However, once again this scheme cannot be implemented if the analyst does not have control over all of the routers in the network.

3.4. Core Principles of Botnet Detection

The analysis of existing detection methods reveals some common ideas about bot behaviour. These principles appear to underpin most traffic-based approaches, and can be summarised as follows:

- **Automation**: Bots are automated programs designed to conduct specific scheduled activities or to respond to commands in particular ways. Therefore, there should always be a discernible structure (and possibly timing) to their behaviour and the traffic they generate.
- **Similarity**: Whether large or small, a botnet is always made up of multiple bot nodes, and its success depends on them working together. Assuming the presence of a single controlling entity (i.e. the botmaster), bots who are members of the same botnet (or who are at least participating in the same campaign) should behave similarly.
- **Divergence**: Normal user behaviour is highly random and difficult to predict. This is a result of the myriad of online services and resources available for use. This should contrast highly with the systematic and consistent behaviours of bots. Whilst it is possible to profile general user activities for anomaly detection, we would still expect to see less uniformity at a granular level than compared to an automated bot.

Table 2 shows which of the 3 principles the surveyed methods align with. Note that this is based on the explicit approaches and the nature of the features used. Otherwise, all botnet approaches implicitly touch upon all 3 principles simply due to the nature of bot behaviour.

4. Features of HTTP-Based Botnet Traffic

Network-based botnet detection revolves around the use of features which are extracted from traffic data. A feature is roughly defined as an indicator or measurement that captures an attribute of traffic. Comparison of instances of the same feature taken from both benign and malicious traffic should reveal differences related to the nature of the

Table 2. Summary of surveyed papers and which of the 3 principles they based their approaches on.

Paper	Automation	Similarity	Divergence
Sharifnya & Abadi (2013)	Y	Y	Y
Schiavoni et al. (2014)	Y	Y	Y
Wang et al. (2010)	Y	Y	N
Soniya & Wilscy (2013)	Y	N	Y
Etemad & Vahdani (2012)	Y	Y	Y
Eslahi et al. (2013)	Y	Y	Y
Eslahi et al. (2015)	Y	Y	N
Lu & Brooks (2011)	Y	Y	Y
Venkatesh & Nadarajan (2012)	Y	Y	Y
Li et al. (2014)	Y	Y	Y
Grill & Rehak (2014)	N	Y	Y
Cai & Zou (2012)	Y	Y	Y
Al-Bataineh & White (2012)	N	Y	Y
Shiaeles et al. (2012)	Y	Y	Y
Xiang et al. (2011)	Y	Y	N
Xiang et al. (2009)	Y	Y	N

activities which generate them. In this section, we survey the features used in current detection approaches related to HTTP-based botnets, specifically considering the IP, TCP, HTTP and DNS protocols.

The IP protocol and its header information is generally useful for defining connection endpoints and finding high-level statistical patterns. Though lacking in granularity, this type of flow-based analysis can be automated and used to filter out unnecessary traffic. Meanwhile, observing the implementation of the TCP protocol (which commonly underpins HTTP communication) and related features adds granularity to flow-based analysis by allowing end-to-end connections to be split down into individual sessions.

HTTP headers provide the most qualitative information on botnet traffic, enabling not just detection but also the characterisation of individual malware. However, additional processing may be necessary to extract HTTP messages from network data in the absence of web proxy logs. Meanwhile, HTTPS (which adds an SSL/TLS encryption layer to standard HTTP) may prevent this level of analysis altogether. Alongside HTTP, we also considered the DNS protocol. DNS is classified as an application layer protocol although it acts more like a network layer utility (Odom, 2011a), bridging IP addresses with domain names. Unlike the other protocols, the use of DNS during the rally phase makes it possible to detect bots before they begin exchanging data with the C&Cs.

4.1. Size and Frequency-Based Features

Size and volume are one simple way to measure traffic. This may include the sizes of packets in terms of bytes, or the sizes of flows in terms of bytes or packets, as well as the volume of certain types of either packets or flows.

4.1.1. Network Layer/IP

Type: Similar Flow Sizes
Normal Traffic: Should be of varied sizes to reflect random nature of online user activity.
Bot Traffic: Should be consistent, especially for the same activities or for members of the same botnet.
Features: Entropy of packet counts for similar flows (Soniya & Wilscy, 2013), generalised entropy of IP packet sizes (Xiang et al., 2011), information distance of IP packet sizes (Xiang et al., 2011)

Based on the principle of automation, bots are programmed to behave a certain way without direct individual control. In terms of flow sizes, this results in a level of consistency. As a hypothetical example, a bot configured to send a certain type of message when triggered will likely generate a flow of the same size for each instance of that message. This will also apply to other bots from the same botnet, as well as to responses from the C&Cs. Soniya & Wilscy (2013) capture this as the entropy of packet counts across a number of similar flows, where a lower level of uncertainty denotes a higher probability of bots. Similarly, Xiang et al. (2011) study packet size to detect DDoS attacks. They sample traffic in 300 second intervals, and then find the generalised entropy of packet sizes, where a smaller result means that packet sizes are more uniform and hence, more likely to be part of an attack. They also calculate the information distance between the probability distribution of packet sizes across consecutive LANs. If the distance is greater than the threshold, an attack is detected (Xiang et al., 2011).

Type: Flow Bandwidth Consumption
Normal Traffic: Apart from occasional random fluctuations (possibly tied to a known benign event or network error), there should generally be no sustained and unexplained excessive consumption of bandwidth.

Bot Traffic: During a DDoS attack, packet flooding will result in flows with the same endpoints occupying a larger portion of the bandwidth, making it difficult to run other services.
Features: Bandwidth consumed by flows (Xiang et al., 2009)

The size of flows may also be considered in terms of the amount of bandwidth they use. Xiang et al. (2009) consider the proportion of the available bandwidth consumed by a flow when selecting it for source IP marking. For flows (passing through the local router) which are targeting the same destination (see Traffic Destination IP Addresses in Section 4.3.1), the greater the bandwidth consumed the higher the likelihood of those packets belonging to a DDoS attack and therefore being marked.

4.1.2. Transport Layer/TCP

Type: Similar Session Sizes
Normal Traffic: Randomly generated TCP sessions, containing packets of various sizes.
Bot Traffic: Comparable TCP sessions of similar length, with similar sized packets
Features: Number of request bytes per flow (Wang et al., 2010), number of response bytes per flow (Wang et al., 2010), number of useful packets per flow (Wang et al., 2010)

Similar flow sizes identified using IP-based features will manifest at the transport layer as well. Bots generating connections or responding automatically to triggers will create TCP sessions for specific purposes. This behaviour should be repeated over time, and by other members of the same botnet. To measure this, Wang et al. (2010) reassemble TCP sessions and filter out packets with empty payloads or that are duplicated or retransmitted. Then they count the number of packets in the flows, as well as the number of bytes in request packets and response packets.

Type: Proportion of TCP Packets
Normal Traffic: Random distribution and number of TCP packets, both incoming and outgoing.
Bot Traffic: Large number of TCP packets, with more of them part of outgoing connection attempts.
Features: Ratio of incoming to outgoing TCP packets (per time window) (Venkatesh & Nadarajan, 2012), ratio of TCP packets to overall packets (per time window) (Venkatesh & Nadarajan, 2012)

HTTP-based botnets do not maintain connections to the C&C servers (Venkatesh & Nadarajan, 2012). Instead, the session is terminated, and then re-established for each new transaction. Given that bots are automated and that they behave similarly to one another, we can assume that this process will be repeated many times by multiple bots, which will result in a large number of TCP sessions. In addition to this, pull-based bots must poll C&Cs regularly, causing a large number of outgoing TCP connection attempts. To detect this, Venkatesh & Nadarajan (2012) find the ratio of TCP packets to the total number of packets per time interval, as well as the ratio of incoming to outgoing TCP packets per time interval.

4.2. Temporal Features

Timing is obviously a key element in any type of traffic analysis, both for sequencing events and understanding event density. Temporal features may therefore be considered in terms of when an event took place, the period of time between events, or the frequency of events for a given interval.

4.2.1. Network Layer/IP

Type: Periodic Flows
Normal Traffic: Flows should be of varied lengths, with random intervals.
Bot Traffic: Should show patterns of periodicity, and be evenly distributed across time.
Features: Periodic requests flows (Li et al., 2014), entropy of time gap between similar flows (Soniya & Wilscy, 2013), inter-packet delays (Lu & Brooks, 2011)

As with flow sizes, flow timings should show consistency, both across multiple traffic events and across bot instances. As outlined by the similarity principle, commands tend to be disseminated from a central entity, so each bot receiving the same set of commands should work to a similar schedule. For example, activities such as C&C polling take place periodically, with the length of that period defined by the botmaster (Binsalleeh et al., 2010). To capture this, Li et al. (2014) look for patterns of periodicity in clustered requests within a single time window. Soniya & Wilscy (2013) measure the entropy of the time intervals between similar flows, considering a higher entropy score to indicate random, bursty user behaviours. Scores that fall below a threshold are considered malicious. Lu & Brooks (2011) capture the delays between successive packets (specifically related to higher level HTTP POST requests and responses) in Zeus bot traffic to generate a hidden Markov model of botnet traffic patterns.

4.2.2. Transport Layer/TCP

Type: Periodic TCP Sessions
Normal Traffic: Randomly generated TCP sessions, with no fixed timing.
Bot Traffic: Similar TCP sessions initiated repeatedly.
Features: Periodicity of similar TCP connections (Wang et al., 2010), TCP packet arrival times (Shiaeles et al., 2012)

As with traffic at the network and application layer, periodicity should also be observed at the transport layer.

Repeated TCP connections initiated by the bots are the necessary foundation for HTTP-based polling activities. For clusters of similar TCP sessions, Wang et al. (2010) look for the presence of those sessions over a periodic point sequence. This results in a binary vector, denoting if and how often sessions from a given cluster were observed. The greater the level of periodicity, the more suspicious that traffic is considered. Meanwhile, Shiaeles et al. (2012) use the arrival time of TCP packets (over short time intervals) to identify developing DDoS attacks. Human-generated TCP packets are expected to fit a Poisson model. Using a fuzzy estimator, they model the standard mean arrival time, with values falling below this mean classed as DDoS events.

4.2.3. Application Layer/HTTP

Type: Periodic HTTP Requests
Normal Traffic: Various types of HTTP request, generated with random intervals.
Bot Traffic: Repetitive HTTP requests generated at regular intervals.
Features: Degree of periodic repeatability (Etemad & Vahdani, 2012), periodic factor (Eslahi et al., 2015), HTTP messages range of absolute frequencies (Eslahi et al., 2015), HTTP messages time sequence (Eslahi et al., 2015), HTTP request density (Cai & Zou, 2012), HTTP request periodicity (Eslahi et al., 2013)

Bots communicating with C&Cs over HTTP need to poll them continuously to receive updates. To achieve this, they will repeatedly generate similar HTTP requests to the server at regular intervals. The actual period length will be configured by the botmaster, and may be different across different instances of the same malware. Etemad & Vahdani (2012) measure this using the degree of periodic repeatability (and repeatability standard deviation), where a low value means that communication takes place at regular intervals and is therefore suspicious. Eslahi et al. (2013) split their observation window into equal time intervals and find the distribution of group events across those intervals. Events related to bots are expected to appear in all intervals. In a separate work, Eslahi et al. (2015) actually define 3 features. Periodic factor measures how often a group of similar HTTP messages appear over a series of time windows. The range of absolute frequencies is used to measure how many times that group appears in each window. Lastly, the time sequence factor measures the time intervals between groups of messages. Meanwhile, Cai & Zou (2012) measure periodicity as the number of requests between a client and a server in a time period, defined as the request density. Bot channels are expected to have a high density, as the same types of traffic are repeatedly generated.

If the botnet utilises HTTPS rather than HTTP, than periodicity of web traffic would have to be measured at the lower levels, using destination ports to identify traffic instances, and IP addresses to identify the endpoints. This is because HTTPS encrypts headers as well as payloads, making it difficult to identify what types of request are being made.

4.2.4. Application Layer/DNS

Type: Group Activity
Normal Traffic: Various queries for different domains, with large distribution of source hosts.
Bot Traffic: Suspicious DNS queries for the same domains, coming from the same group of hosts.
Features: Suspicious group activity across time (Sharifnya & Abadi, 2013), participant similarity (Sharifnya & Abadi, 2013)

Based on the similarity principle, we expect members of the same botnet to generate comparable DNS traffic with similar timing. In other words, a large number of similar queries will be issued by the same set of hosts. Sharifnya & Abadi (2013) define a group DNS activity as one where multiple hosts query domains using the same TLD/SLDs or which map to the same IPs. A group activity is considered suspicious if those domains are identified as DGA-generated (see Suspicious Domain Names). Suspicious group activity events are then mapped across a series of time periods. As with suspicious failures, if a host is found to have taken part in suspicious group activities over multiple periods, this contributes to a higher negative reputation score. Additionally, for a given host, Jaccard index is used to measure the similarity of its participants in those group activities. Bots are expected to have a lower diversity of participants as the same nodes in the same botnet work in unison.

4.3. Content-Based Features

Where available, the contents of messages at the different layers provide details of the traffic which help to flesh out the underlying activities. This may include various header fields as well as the actual payloads.

4.3.1. Network Layer/IP

Type: Traffic Source IP Addresses
Normal Traffic: Source IPs belonging to users should be randomly distributed across address space.
Bot Traffic: Source IPs belonging to bots should be concentrated within certain ranges, and may generate large amounts of traffic in response to certain events (e.g. command dissemination).
Features: Source IP location (Cai & Zou, 2012), DDoS participant host IPs (Shiaeles et al., 2012), generalised entropy of source IP addresses (Xiang et al., 2011), information distance of source IP addresses (Xiang et al., 2011)

Bots which manage to infiltrate a vulnerable network are likely to propagate more quickly to local hosts than

to remote ones. Based on the assumption that members of the same botnet will behave similarly, in certain cases this may result in a number of comparable connections being made from similar address ranges to the same C&C servers. For example, if the botmaster issues a general update command, multiple bot hosts may attempt to contact a server and download a similar binary file. Cai & Zou (2012) capture this by observing the source IPs in flows targeting suspicious servers, and sorting them into /24 subnets. To deal with instances where legitimate users make repeated connections to the same websites, they combine this with their measure of periodicity to capture request density (see Periodic HTTP Requests in Section 4.2.3). Similarly, Shiaeles et al. (2012) aim to identify the hosts participating in DDoS attacks by calculating the density of traffic coming from each unique IP address in suspicious TCP streams. This is defined as the number of packets generated by that host within the time period. They then find the average inter-packet arrival time and compare to a fuzzy estimator value (see Periodic TCP Sessions in Section 4.2.2). Xiang et al. (2011) utilise the distribution of source IPs by sampling traffic in intervals of 300 seconds and finding the generalised entropy, considering a smaller result to be more suspicious. They also use information distance to compute the difference between the probability distributions of source IPs between two adjacent LANs, and detect an attack when the result is above a threshold.

Type: Traffic Destination IP Addresses
Normal Traffic: Destination IPs in benign traffic should be randomly distributed to reflect the variety of services accessed by users.
Bot Traffic: DDoS attacks, such as those launched by bots, should target the same specific destination IPs.
Features: Flows with the same destination IP (Xiang et al., 2009)

The previous set of features focused on attack sources via source IP addresses. However, in the case of DDoS-style attacks, we may want to consider attack targets as well. In the flow-based marking portion of their work, Xiang et al. (2009) aim to identify DDoS flows by observing destination IP addresses. Specifically, they count the number of packets going to the same destination IP, and use this to derive a marking probability. This is combined with a measure of the bandwidth consumed (see Flow Bandwidth Consumption in Section 4.1.1) so that the packets marked as DDoS are those which add the heaviest load to the network and who are targeting the same destination nodes.

4.3.2. Transport Layer/TCP

Type: Broken Handshakes
Normal Traffic: Standard use of the TCP protocol, with reasonable levels of error.
Bot Traffic: Large number of failed outgoing TCP handshakes.
Features: Ratio of one-way TCP connections (per time window) (Venkatesh & Nadarajan, 2012)

Bots initiating TCP handshakes with C&C servers may not receive a response if the server is offline, migrated, or using fluxing technology. If there are multiple members of the same botnet in the observed network, these failed connections may happen within the same interval of time. This will result in a larger-than-usual number of failed or half-open connections. Venkatesh & Nadarajan (2012) measure this as the ratio of one-way TCP connections to the total number of TCP packets per time window.

Type: Flags
Normal Traffic: Standard use of the TCP protocol, with random number of each type of flag.
Bot Traffic: Large (and consistent) number of TCP packets with SYN flag, as well as PSH and FIN.
Features: Number of SYN flags (per time window) (Venkatesh & Nadarajan, 2012), number of FIN flags (per time window) (Venkatesh & Nadarajan, 2012), no of PSH flags (per time window) (Venkatesh & Nadarajan, 2012)

A bot which is persistently attempting to initiate TCP connections will naturally generate a large number of SYN flags. Assuming that they are members of the same botnet, this behaviour will be observed across a number of bots if the target C&C is offline. We also know that there will be multiple instances of bot-to-C&C sessions, and that each one will be torn down once transactions are complete. Therefore, bot traffic should generate many PSH and FIN flags as well. Venkatesh & Nadarajan (2012) count the number of SYN, FIN, and PSH flags in a single time interval, where a large but consistent number of each flag over multiple time intervals is considered suspicious.

4.3.3. Application Layer/HTTP

Type: Similar HTTP Requests
Normal Traffic: Randomly-timed, diverse range of requests for various services, URIs, and targets.
Bot Traffic: Persistent requests, repeatedly asking for the same services or resources.
Features: Similarity of request strings (Cai & Zou, 2012), entropy of HTTP POST payloads (Al-Bataineh & White, 2012), byte distribution of HTTP POST payloads (Al-Bataineh & White, 2012)

The periodic request generated by bots will be similar in context and aim. Unlike those coming from normal users, these requests are part of a bot's recurring tasks. Therefore, the services, files and URIs identified in requests should be consistent across both request and bot instances. Cai & Zou (2012) use Levenshtein Distance (LD) to measure the similarity of two HTTP request strings in HTTP flows. Specifically, requests are checked for the similarity

of their methods (either GET or POST), their file paths and any additional query parameters. Flows grouped by destination servers are then clustered by request similarity. Meanwhile, Al-Bataineh & White (2012) measure the similarity of POST requests by the entropy and byte distribution of their payloads. They aim to identify similar, encrypted payloads in outgoing connections which are considered to be indicative of data exfiltration. This may also be applicable to instances of HTTPS traffic. Whilst it is not possible to pick out specific request types, comparing entropy and byte distributions may reveal patterns in repeated traffic.

Type: Similar HTTP Responses
Normal Traffic: Diverse range of services or files provided to clients, which may be large in size.
Bot Traffic: Responses made up of small-sized packets designed to avoid drawing attention.
Features: HTTP response packet payload length (Cai & Zou, 2012)

Files returned to bots from the C&C servers may vary in size depending on nature of the commands or updates they contain. However, botmasters are generally likely to avoid large bulk transfers which will put strain on the network and draw unwanted attention. Cai & Zou (2012) suggest that the payloads received for legitimate requests tend to be larger in contrast. They measure the length of HTTP reply packet payloads for a set of clustered flows. They consider total payloads below a threshold of 2KB to be suspicious. This feature won't be useful for HTTPS traffic, as the payload size will be obscured by encryption.

Type: Poor HTTP Header Integrity
Normal Traffic: Standard HTTP header fields and values, correctly formatted.
Bot Traffic: Short HTTP headers with missing fields, or fields filled out with custom strings.
Features: Deviation from header standards (Li et al., 2014), malicious keywords in initial requests/responses (Li et al., 2014), incorrect User-Agent fields (Grill & Rehak, 2014), HTTP header similarity (Cai & Zou, 2012)

Legitimate services generally adhere to protocol standards, which includes the construction of well-formed headers. However, malware will sometimes omit non-mandatory fields or use their own custom strings, resulting in a divergence from normal protocol implementation. To capture this attribute, Li et al. (2014) check HTTP headers for deviations from a predefined profile of normal protocol use. They also inspect the headers of the first request and response packets for a collection of known malware strings. Similarly, Grill & Rehak (2014) check User-Agent fields for custom strings which are not actually representative of the host's software environment. Meanwhile, Cai & Zou (2012) compare the similarity of request headers, where a close match is considered to indicate a bot presence. However, this information won't be observable for HTTPS traffic due to header encryptions.

4.3.4. Application Layer/DNS

Type: Domain-to-IP Mapping
Normal Traffic: Random mapping, or sets of IPs for known legitimate web services.
Bot Traffic: Similar IPs mapping to suspicious domains.
Features: Similarity of IP mappings (Schiavoni et al., 2014)

A botmaster who uses multiple C&Cs with fluxing technology is likely to have ownership over multiple domain names and IPs. Therefore, it may be reasonable to assume that there are pools of IPs which map to similar domain names making up sections of the C&C network. From logs of DNS query results, Schiavoni et al. (2014) collect a set of domains believed to have been generated by DGAs, and analyse the similarity of the IPs they map to. If domains are generated by the same DGA and map to similar IPs (i.e. in the same subnet), then this is considered to be indicative of botnet behaviour. If a botnet implements DNSSEC to avoid hijacking by researchers, the DNS responses should be digitally signed. This may provide an extra parameter for comparison, as instances of similar domain-to-IP mapping may feature DNS responses containing identical signatures.

Type: Suspicious Domain Names
Normal Traffic: Readable domain names, containing real words and following formatting rules.
Bot Traffic: Domain names which are made up of random strings, possibly with similar top (TLD) or second level domains (SLD).
Features: Deviation from benign names model (Schiavoni et al., 2014), pronounceability of domain names (Schiavoni et al., 2014), meaningful character ratio (Schiavoni et al., 2014), n-gram distribution of domain names (Sharifnya & Abadi, 2013)

Human-generated domains will typically use titles or follow certain formats. Botnet domains don't necessarily adhere to these characteristics, especially when DGAs are used. A DGA-generated domain name will be alphanumerically random, and in a set, there may be patterns of similar TLDs or SLDs. Schiavoni et al. (2014) measure the meaningful character ratio to capture the proportion of a domain name that passes for real English. They then measure pronouncability using n-gram normality score (counting the frequency of domain n-grams as they appear in a dictionary). These results are combined into an average value for benign domain names. Mahalanobis distance is then used to compare a domain to the centroid or mean value of benign domains, with a threshold to determine when the domain is deemed suspicious. Meanwhile, Sharifnya & Abadi (2013) find the n-gram distribution of observed domains names and compare them to the distribu-

tions of benign domains to identify those which are algorithmically generated. This is achieved using Kullback-Leibler (K-L) divergence to measure the distance between probability distributions and Spearman's rank coefficient to measure correlation between n-gram frequency values.

Type: Query Success Rate
Normal Traffic: Normal query traffic, with reasonable rate of error.
Bot Traffic: Large number of failed DNS queries, possibly for the same domain names.
Features: Quantity of DNS query failures (Sharifnya & Abadi, 2013), suspicious query failures across time (Sharifnya & Abadi, 2013)

A bot trying to rally to its C&C server will generate DNS queries in bursts as it works sequentially through a list of hardcoded or DGA-generated domains in order to find a connectable C&C. The assumption is that in doing so they will generate failures consistently. Sharifnya & Abadi (2013) observe failed DNS queries for hosts, defining a threshold for the number of failures which are considered suspicious. These events are then mapped across a series of time periods for that host, where a large number of suspicious failures over multiple periods results in a higher negative reputation score.

Table 3 provides a summary of the features, organised by bot behaviours (as identified in Section 2). Domain resolution activities revolve around DNS-based features. The similarity principle, highlighted as Group Activity, can also be measured. The automation principle is captured in periodic polling activities, whilst the divergence principle can be seen in the features under Pull-Down Activities (which cover protocol usage that is characteristic of bots). Additionally, the blank cells of the table serve as suggestions for areas of future research and possible opportunities to find new features.

To provide an alternative view, Figures 2 - 5 collect features for each protocol, and break them down by the context, and then by set. For example, TCP features are made up of Sessions and Time. Under Time, Flags is a set of features, listed as the number of different flag types. Meanwhile, IP-based features are split into Flows and Hosts, which are the two contexts in which measurements are being made. The former covers features related to end-to-end communication, whilst the latter is for the characteristics of those endpoints. HTTP features are broken down into Requests and Responses (for the characteristics of those messages), and Headers to account for features relating to the specific way that bots handle this protocol. Lastly, DNS features are broken down into Mappings, Domains and Activity. The Domains set is for features which deal with the make-up of domain names. Activity covers those which capture similar, synchronised DNS behaviours, and finally, Mappings cover the relationships between domains and IPs.

Figure 6 demonstrates how the features relate to each other in a simplified example of the traffic flows coming from a single bot host. Initially, the host is expected to query domains (where domain A and domain B may be DGA-generated). Some DNS queries are expected to fail and if there are multiple bots, group DNS activity may also be observed. When an IP is resolved, the bot will then attempt to contact that C&C server, setting up a TCP session. Some C&C servers may not respond resulting in incomplete handshakes. This may impact the overall number of TCP packets generated and usage of flags. Once TCP session are successfully established, bots will send similar HTTP requests, and likely receive similar responses.

5. Discussion

So far, we have considered the known behaviours of HTTP-based botnets, how this impacts the traffic they generate and how that traffic may be identified. In this section, we discuss the implications of our findings and suggest possibilities for the future of HTTP-based botnet detection.

5.1. Current Research

As we have outlined in this survey there have been many research efforts in response to the HTTP-based malware threat. Discussion of these works is split into two parts. Firstly, we look at the general approaches taken towards identifying malicious traffic, including the areas in which research was most focused. Secondly, we consider the effectiveness of the feature sets used based on the categorisations provided in Section 4.

5.1.1. Characterisation & Identification

The survey demonstrates the significance of traffic-based analysis in not only detecting botnets but also characterising and understanding them. Given the nature of bots, they are highly likely to generate some form of traffic in order to achieve their goals. When that traffic is considered in terms of underlying activities, it is possible to reveal patterns of behaviour. We noted that all the approaches we studied are underpinned by common principles of botnet behaviour, defined as automation, similarity (to other bots), and divergence (from normal users). This shows that botnet detection is being approached in the same way, and is hence developing along the same lines.

These basic characteristics are easier to observe when the type of traffic is narrowed down. Overall, many works appear to focus on a single stage (of the lifecycle) and a specific type of traffic. This may be sufficient for detection, but doesn't provide detail on the sequence of bot activities. With no knowledge of the events before the point of detection, potential vulnerabilities may be missed. Additionally, bots which slip through detection at that stage

Table 3. Summary of features sorted by source behaviours against protocol types.

	Botnet behaviours				
	Resolving of C&C Domains	**Periodic Polling of C&Cs**	**Pull-Down of Commands & Updates**	**Group Activity**	**Data Exfiltration**
DNS	Similarity of IP mappings between domains, used for clustering (Schiavoni et al., 2014)	-	-	Suspicious collective queries to DGA-generated domains (Sharifnya & Abadi, 2013)	-
	Deviation of domains from model, measured with Mahalanobis distance (Schiavoni et al., 2014)	-	-	Participant similarity of hosts across time, measured with Jaccard distance (Sharifnya & Abadi, 2013)	-
	Meaningful characters ratio of domain prefix split into substrings of 3 symbols (Schiavoni et al., 2014)	-	-	-	-
	N-gram distribution of domain names (Sharifnya & Abadi, 2013)	-	-	-	-
	No. of DNS query failures per host which exceeds a threshold (Sharifnya & Abadi, 2013)	-	-	-	-
	Suspicious query failures generated across time (Sharifnya & Abadi, 2013)	-	-	-	-
IP	-	Packet count entropy for flow set, where value closer to 0 denotes bots (Soniya & Wilscy, 2013)	-	Source IPs concentrated in similar /24 subnets (Cai & Zou, 2012)	-
	-	Periodic request flows for similar flows, mined per time window (Li et al., 2014)	-	Unique DDoS participants, measured by no. of packets generated compared to an estimator (Shiaeles et al., 2012)	-
	-	Inter-packet delays between consecutive packets for inferring Hidden Markov model (Lu & Brooks, 2011)	-	Proportion of bandwidth consumed by flows with the same destination IP, indicating possible DDoS attack (Xiang et al., 2009)	-
	-	Entropy of time gap between similar flows, with upper bot threshold of 0.5 (Soniya & Wilscy, 2013)	-	-	-
	-	Generalised entropy of IP packet sizes/source IP address distribution, compared to thresholds (Xiang et al., 2011)	-	-	-
	-	Information distance of IP packet sizes/source IP address distribution compared to thresholds (Xiang et al., 2011)	-	-	-
TCP	-	No of request bytes, used for clustering (Wang et al., 2010)	Ratio of incoming to outgoing TCP packets, sampled in 5s intervals (Venkatesh & Nadarajan, 2012)	-	-
	-	No of response bytes, used for clustering (Wang et al., 2010)	Ratio of TCP packets to overall packets, sampled in 5s intervals (Venkatesh & Nadarajan, 2012)	-	-
	-	No of useful packets, used for clustering (Wang et al., 2010)	Ratio of one-way TCP connections, sampled in 5s intervals (Venkatesh & Nadarajan, 2012)	-	-
	-	Periodicity of similar TCP connections, where connections form a periodic point sequence (Wang et al., 2010)	No of SYN, PSH & FIN flags, sampled in 5s intervals (Venkatesh & Nadarajan, 2012)	-	-
	-	TCP packet arrival time, in 1-2s intervals, compared to a mean estimator (Shiaeles et al., 2012)	-	-	-
HTTP	-	Degree of periodic repeatability of requests, where low value denotes bots (Etemad & Vahdani, 2012)	Similarity of request strings, measured with Levenshtein distance & threshold between 0-0.3 (Cai & Zou, 2012)	-	Byte entropy of POST payloads; encrypted content denoted by Shannon entropy value greater than 6.503545 (Al-Bataineh & White, 2012)
	-	Periodic factor, a binary value of whether an event was detected across periods or not (Eslahi et al., 2015)	Deviation from header standards, missing fields like User-Agent or Referrer (Li et al., 2014)	-	Byte frequency distribution of POST payloads, finds those which are most constant (Al-Bataineh & White, 2012)
	-	Range of absolute frequencies of HTTP messages, where value greater than 1 denotes non-uniform requests (Eslahi et al., 2015)	Keywords in initial requests/ responses, like "os=" or "id=" (Li et al., 2014)	-	Response packet payload length, with total bot context length threshold of 2KB (Cai & Zou, 2012)
	-	Messages time sequence, checks if intervals of requests follow an arithmetic sequence (Eslahi et al., 2015)	Incorrect User-Agent fields, defined as unknown, well-known, or empty (Grill & Rehak, 2014)	-	-
	-	Request density, defined as no. of requests between given endpoints, with bot threshold between 0.004-0.1 (Cai & Zou, 2012)	-	-	-
	-	Request periodicity for 1hr intervals, finds hosts with activities in each interval (Eslahi et al., 2013)	-	-	-

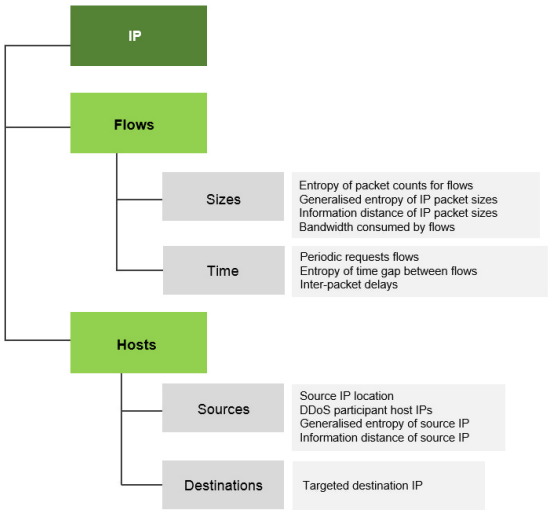

Figure 2. IP-based features, from (Soniya & Wilscy, 2013), (Li et al., 2014), (Lu & Brooks, 2011), (Cai & Zou, 2012), (Shiaeles et al., 2012), (Xiang et al., 2011) & (Xiang et al., 2009).

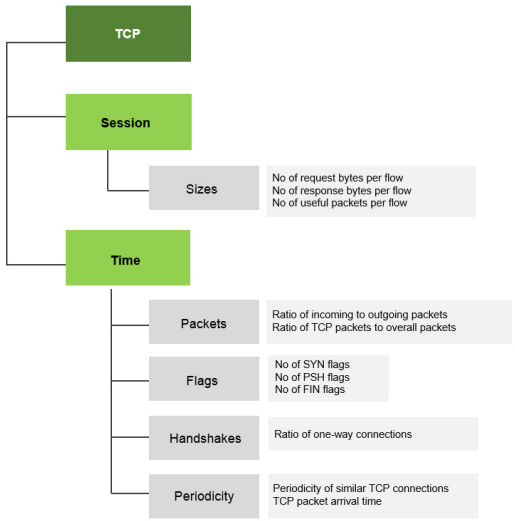

Figure 3. TCP-based features, from (Wang et al., 2010), (Shiaeles et al., 2012), & (Venkatesh & Nadarajan, 2012).

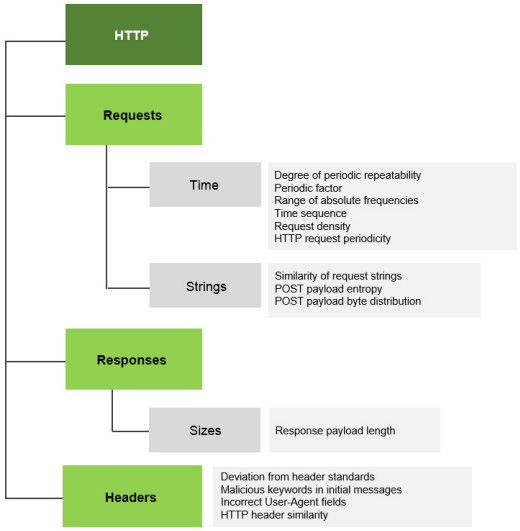

Figure 4. HTTP-based features, from (Etemad & Vahdani, 2012), (Eslahi et al., 2013), (Eslahi et al., 2015), (Cai & Zou, 2012), (Al-Bataineh & White, 2012), (Li et al., 2014), & (Grill & Rehak, 2014).

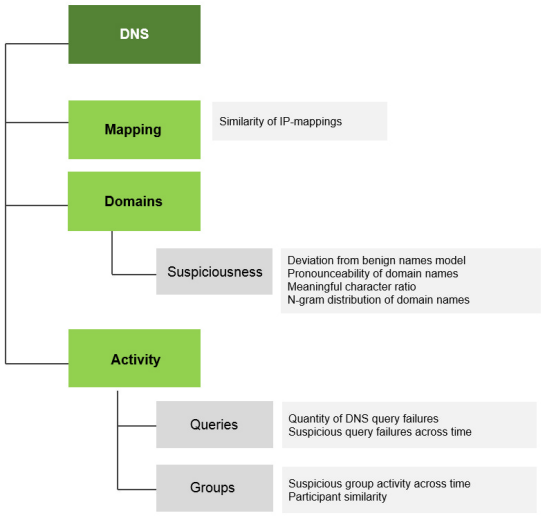

Figure 5. DNS-based features, from (Schiavoni et al., 2014) & (Sharifnya & Abadi, 2013).

cannot be identified at the next stage. Therefore, it is important to be able to detect various types of botnet traffic in order to uncover the methods of the botmaster. Understanding of chains of events may also be significant when HTTP-based network activity is too close to benign traffic.

In our survey, we found that a large number of works targeted control traffic. This is a logical approach, since this communication is central to the functioning of the botnet. However, in doing so, these methods can only detect C&C traffic as it manifests. In other words, bots can only be detected after they are within the observed network. For the works who rely on the similarity principle, there needs to be multiple bot instances in order to make a reliable detection. Even approaches that propose host-based traffic analysis will need the same bot to engage in activities multiple times. Therefore, bots should ideally be detected in the minimum time in order to mitigate potential damage caused. A promising angle is the observation of DNS traf-

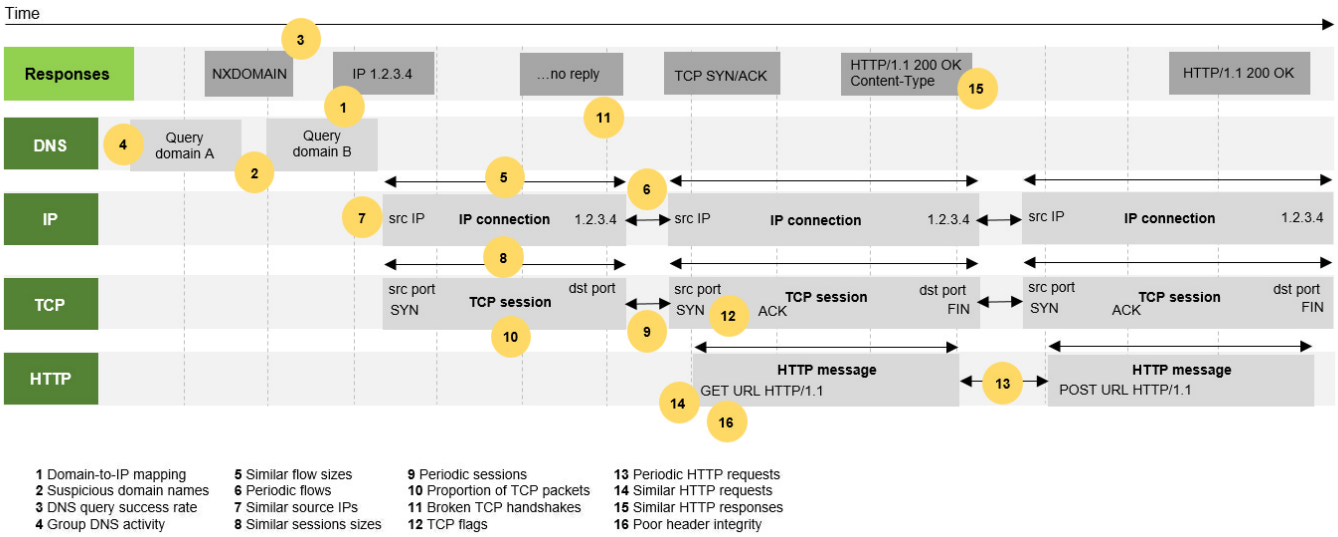

Figure 6. Overview of the features in the context of network traffic.

fic in the rally stage. Since a bot needs to resolve domains, this provides a window of opportunity to make detections before bots are able to establish connections to their C&C servers. Where fluxing is used, even after connections are made bots will eventually need to send out another round of queries.

We also learnt that (similarly to propagation traffic), attack traffic is difficult to characterise, due to the variety of attack methods which may be utilised by botmasters. This contrasts highly with control traffic, as campaigns may start and end at any time and exfiltrated data may be of variable length. It is therefore more challenging to pick out instances of uniform behaviour. A potential solution to this may be to detect the command messages specifically containing attack instructions to try and pre-empt these activities.

5.1.2. Features

In Section 4 we chose to organise features by protocol for a number of reasons. Our first aim was to attempt a high-level characterisation of botnet traffic based on the implementation of those protocols. Whilst works like Li et al. (2014) and Soniya & Wilscy (2013) make use of benign and malicious traffic profiles, we suggest that for the benefit of the community, considering how different researchers have captured similar bot traffic and combining these perspectives provides a better overview of bot behaviour for a given protocol. For example, we learnt that HTTP headers may use similar keywords (Li et al., 2014), use wrong field entries (Grill & Rehak, 2014), omit fields altogether and show similarity across instances (Cai & Zou, 2012). The results of this characterisation are summarised in Figures 2 - 5.

Secondly, we were also able to gain an understanding of how the same traffic may look at different layers. This may be of particular benefit when considering whether to use flow-based or packet-based analysis, or working with restricted datasets. For example, for polling activities, we have seen that periodicity can be measured at the network, transport and application layers. This shows that researchers and defenders have a number of options which can be adapted to their specific circumstances.

Similarly, different characteristics can be observed at each layer at the same time. An example of this based on the surveyed papers is when researchers see a number of flows with the same source and destination IPs, with similar timings and lengths, and deem them suspicious. This may then be supported by the observation of multiple TCP sessions continuously being set up and torn down within those flows, increasing the level of suspiciousness. This in turn may be supported by the presence of repeated HTTP requests with the same anomalies in their headers. Hence, despite only representing a single time period, each layer provides additional detail which increases the likelihood of the source host being a bot.

If we take a number of those instantaneous snapshots of the traffic over time, this theoretically starts to reveal the sequence of bot activities. As demonstrated in Figure 6, we have features that are designed to capture the progress of a bot from its rally stage through to attacks, suggesting both a number of expected anomalies, as well as the order in which they should appear. In this paper, we consider all of the surveyed works to be observing (at a high-level) the same series of events, focusing on different moments in the botnet lifecycle and different layers of the protocol stack. Hence, providing a consolidated view of those approaches can help to define the expected sequence of bot behaviours.

For HTTPS-based botnet communications, this be-

comes more difficult as application layer details are hidden behind SSL or TLS encryption. If it is not possible to decrypt captured packets, then network analysts may need to focus their attention on statistically identifying anomalies at the lower layers. For example, Venkatesh & Nadarajan (2012), who use transport layer features, were able to detect encrypted bot traffic. The presence of web activity (inferred from the use of SSL or TLS) may also still be used as a marker for the bot's current status in its lifecycle.

We also briefly touched upon the possible use of DNSSEC in Section 4.3.4. As mentioned, a botmaster may choose to implement DNSSEC to avoid instances of bot hijacking by confirming to bots that they are indeed connecting to a real C&C server. However, DNSSEC does not encrypt responses (ICANN, 2016), so under these circumstances application layer analysis is still available, unlike with HTTPS. Furthermore, a botnet using DNSSEC may attract attention due to larger packet sizes (Hands et al., 2015).

5.2. Challenges & Future Work

Current detection approaches focus on traffic generated at one stage of the botnet lifecycle. We feel it is beneficial to be able to detect bots across the stages, as this provides both a wider view of bot activities (which can aid predictive approaches) and a wider window of opportunity for detection. Furthermore, positive detection made in one phase can potentially be supported or confirmed in the next, improving accuracy.

False positives also remain a problem. This appears to be inevitable to some extent as services operating on the same protocols through the same infrastructure are going to share some characteristics. Hence, attempting to differentiate traffic based on only one or two of those characteristics is likely to fail at some point. The most obvious solution is to consider a wider range of features, from different protocols. We predict that the best results can be achieved if the chosen features combine measurements of automation, similarity and divergence.

For both multi-phase detection and mitigation of false positives, a tradeoff exists between the amount of data analysed and the level of processing required, so it is impractical to attempt to accommodate all of the features mentioned. Therefore, in the future we would like to investigate combinations of features (across layers and stages) to identify those which best complement each other.

Another challenge is the general lack of uniform metrics within the research community. This makes measurements of traffic attributes or anomalies difficult to compare across different works. Any common definitions will need to clearly encompass or enumerate given characteristics, whilst being flexible enough to be used in multiple contexts. One potential approach may be a specially-designed language like CybOX (GitHub, 2016).

In the future, we would like to expand this work by identifying more features. We believe that incorporating other methods for measuring anomalies will provide more breadth and hence improve our understanding of bot processes. This can also help to uncover new characteristics which are currently being overlooked. Other protocols which are used by HTTP-based botnets should be included as well. For example, UDP may provide new insights into bot DNS traffic, whilst SMTP could help to characterise attacks related to spam mail.

There is also potential benefit in considering the level of processing needed to acquire different features. In this paper, we tried to consolidate similar features into sets, denoted by the characteristics being measured. However, features can be sorted as primary and secondary, where the former are 'raw' and taken directly from the data, whilst the latter are derived from those raw features. An example may be timestamps and time intervals. This will provide an opportunity to understand dependency relationships and may help uncover more efficient handling methods.

5.3. Impact of Next-Generation Protocols

As technology progresses, newer versions of existing protocols are being introduced. Differences in the formatting and implementation of those protocols may have an impact on the operations of malware, and how related features are measured. An example is IPv6, which will be more widely adopted as Internet coverage continues to grow. The obvious difference between IPv6 and IPv4 is the increased address range (Odom, 2011b). This will remove the need for private IPs and functions like NAT (Odom, 2011b). Instead of bots or C&Cs who share similar subnets, we may see a much wider distribution. IPv6 headers are also longer (due to longer IP addresses (Odom, 2011b)) which may impact processing times of current detection methods which rely on addresses to separate flows.

Another new protocol version is HTTP/2, which is designed to provide general performance improvements (O'Reilly.com, 2013) (GitHub). This new standard compresses HTTP headers (from ASCII to binary) and allows multiple HTTP requests and responses to take place within the same TCP connection (comparable to an advanced version of current HTTP pipelining) (O'Reilly.com, 2013) (GitHub). HTTP/2 also features server push, whereby the server doesn't have to wait for the browser to parse HTML pages before pushing related embedded assets (O'Reilly.com, 2013) (GitHub). These changes may have an impact on expected TCP statistics. For instance, if bots try to hide themselves by making multiple, small connections, they may stand out more against larger, longer TCP connections. On the other hand, malware could exploit the server push feature to infect user systems without needing user interaction with webpage elements. Meanwhile, compressed binary HTTP headers will

potentially make automated analysis of application layer information much simpler for researchers, by removing the need to parse plain text messages.

6. Conclusion

HTTP-based botnets aim to carry out their activities by hiding in plain sight, amongst the vast quantities of web traffic generated by benign user activities. To assist in the fight against this threat, we have provided a study of the general tasks that bots must complete at different stages of their lifecycle, along with the traffic events which may be observed as a result. We have surveyed a collection of recent papers focused on HTTP-based communication detection, and categorised and evaluated the traffic-based features they have used. We have presented our findings with a hierarchical structure based on feature types and protocols arranged by OSI layers, and touched upon how secure versions and future iterations of these protocols may impact analysis capability.

Based on the wealth of traffic-based detection techniques presented, it is clear that this is a promising and effective approach in defending against botnets. We have seen that with a clear understanding of correct implementation of network protocols and processes, it is possible to pick out instances of unusual behaviour. However, it is important to be able to observe across multiple layers of the network wherever possible, and to tie sequences of activities together to put specific instances of bot traffic into context. Network analysts should therefore carefully consider the types of devices, users and services which they would expect to see on their networks, as well as what types of capture or log data will be available to them.

We briefly discussed how analysis techniques may need to be adapted for new protocol versions like HTTP/v2 and IPv6. As trends in the digital landscape shift and develop, so too will the approaches of malware. An example of a recent trend among users has been identity anonymisation with the use of tools like the Tor network (Project) to maintain privacy on the web. A botnet which runs on the Tor network would be difficult to track without gaining control of multiple relays points (Project), making it difficult to trace end-to-end communication. Similarly, as HTTPS and encryption becomes more widely implemented, this can diminish the depth of the analyses that can be made. However, works like Venkatesh & Nadarajan (2012) show that it is possible to overcome these obstacles and detect patterns in traffic. Future botnet-detection (and intrusion-detection approaches in general) should consider and work around these limitations.

Acknowledgements

This work was completed with sponsorship from BT.

References

Al-Bataineh, A., & White, G. (2012). Analysis and detection of malicious data exfiltration in web traffic. In *Malicious and Unwanted Software (MALWARE), 2012 7th International Conference on* (pp. 26–31). doi:10.1109/MALWARE.2012.6461004.

Andrade, M., & Vlajic, N. (2012). Dirt jumper: A key player in today's botnet-for-ddos market. In *Internet Security (WorldCIS), 2012 World Congress on* (pp. 239–244).

Beigi, E., Jazi, H., Stakhanova, N., & Ghorbani, A. (2014). Towards effective feature selection in machine learning-based botnet detection approaches. In *Communications and Network Security (CNS), 2014 IEEE Conference on* (pp. 247–255). doi:10.1109/CNS.2014.6997492.

Binsalleeh, H., Ormerod, T., Boukhtouta, A., Sinha, P., Youssef, A., Debbabi, M., & Wang, L. (2010). On the analysis of the zeus botnet crimeware toolkit. In *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference on* (pp. 31–38). doi:10.1109/PST.2010.5593240.

Borgaonkar, R. (2010). An analysis of the asprox botnet. In *Emerging Security Information Systems and Technologies (SECURWARE), 2010 Fourth International Conference on* (pp. 148–153). doi:10.1109/SECURWARE.2010.32.

Cai, T., & Zou, F. (2012). Detecting http botnet with clustering network traffic. In *Wireless Communications, Networking and Mobile Computing (WiCOM), 2012 8th International Conference on* (pp. 1–7). doi:10.1109/WiCOM.2012.6478491.

Eslahi, M., Hashim, H., & Tahir, N. (2013). An efficient false alarm reduction approach in http-based botnet detection. In *Computers Informatics (ISCI), 2013 IEEE Symposium on* (pp. 201–205). doi:10.1109/ISCI.2013.6612403.

Eslahi, M., Rohmad, M., Nilsaz, H., Naseri, M., Tahir, N., & Hashim, H. (2015). Periodicity classification of http traffic to detect http botnets. In *Computer Applications Industrial Electronics (ISCAIE), 2015 IEEE Symposium on* (pp. 119–123). doi:10.1109/ISCAIE.2015.7298339.

Etemad, F., & Vahdani, P. (2012). Real-time botnet command and control characterization at the host level. In *Telecommunications (IST), 2012 Sixth International Symposium on* (pp. 1005–1009). doi:10.1109/ISTEL.2012.6483133.

Farina, P., Cambiaso, E., Papaleo, G., & Aiello, M. (2016). Are mobile botnets a possible threat? the case of slowbot net. *Computers & Security*, *58*, 268–283.

Garcia, S., Zunino, A., & Campo, M. (2014). Survey on network-based botnet detection methods. *Security and Communication Networks*, *7*, 878–903. URL: http://dx.doi.org/10.1002/sec.800. doi:10.1002/sec.800.

GitHub (). Http/2 frequently asked questions. URL: https://http2.github.io/faq/ last accessed 20th January 2016.

GitHub (2016). Cyber observable expression (cybox). URL: https://cyboxproject.github.io/ last accessed 29th January 2016.

Grill, M., & Rehak, M. (2014). Malware detection using http user-agent discrepancy identification. In *Information Forensics and Security (WIFS), 2014 IEEE International Workshop on* (pp. 221–226). doi:10.1109/WIFS.2014.7084331.

Gu, G., Zhang, J., & Lee, W. (2008). Botsniffer: Detecting botnet command and control channels in network traffic, .

Haddadi, F., & Zincir-Heywood, A. (2014). Benchmarking the effect of flow exporters and protocol filters on botnet traffic classification. *Systems Journal, IEEE*, *PP*, 1–12. doi:10.1109/JSYST.2014.2364743.

Hands, N. M., Yang, B., & Hansen, R. A. (2015). A study on botnets utilizing dns. In *Proceedings of the 4th Annual ACM Conference on Research in Information Technology* (pp. 23–28). ACM.

ICANN (2016). Dnssec. URL: https://www.icann.org/resources/pages/dnssec-qaa-2014-01-29-en last accessed 22nd August 2016.

Li, K., Liu, C., & Cui, X. (2014). Poster: A lightweight unknown http botnets detecting and characterizing system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security* CCS '14 (pp. 1454–1456). New York,

NY, USA: ACM. URL: http://doi.acm.org/10.1145/2660267.2662375. doi:10.1145/2660267.2662375.

Lu, C., & Brooks, R. (2011). Botnet traffic detection using hidden markov models. In *Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research* CSIIRW '11 (pp. 31:1–31:1). New York, NY, USA: ACM. URL: http://doi.acm.org/10.1145/2179298.2179332. doi:10.1145/2179298.2179332.

McAfee (2015). Threats report february 2015. URL: http://www.mcafee.com/uk/resources/reports/rp-quarterly-threat-q4-2014.pdf last accessed 22nd July 2015.

Odom, W. (2011a). *CCENT/CCNA ICND1 640-822 Official Cert Guide (3rd Edition)*. Cisco Press.

Odom, W. (2011b). *CCNA ICND2 640-816 Official Cert Guide (3rd Edition)*. Cisco Press.

O'Reilly.com, C. L. (2013). Chapter 12. http/2. URL: http://chimera.labs.oreilly.com/books/1230000000545/ch12.html last accessed 20th January 2016.

Project, T. T. (). Tor: Overview. URL: https://www.torproject.org/about/overview.html last accessed 22nd August 2016.

Rahimian, A., Ziarati, R., Preda, S., & Debbabi, M. (2014). Foundations and practice of security: 6th international symposium, fps 2013, la rochelle, france, october 21-22, 2013, revised selected papers. chapter On the Reverse Engineering of the Citadel Botnet. (pp. 408–425"). Springer International Publishing. URL: http://dx.doi.org/10.1007/978-3-319-05302-8_25. doi:10.1007/978-3-319-05302-8_25.

Rodriguez-Gomez, R. A., Macia-Fernandez, G., & Garcia-Teodoro, P. (2013). Survey and taxonomy of botnet research through life-cycle. *ACM Comput. Surv.*, *45*, 45:1–45:33. URL: http://doi.acm.org/10.1145/2501654.2501659. doi:10.1145/2501654.2501659.

Rostami, M., Eslahi, M., Shanmugam, B., & Ismail, Z. (2014). Botnet evolution: Network traffic indicators. In *Biometrics and Security Technologies (ISBAST), 2014 International Symposium on* (pp. 274–279). doi:10.1109/ISBAST.2014.7013134.

Schiavoni, S., Maggi, F., Cavallaro, L., & Zanero, S. (2014). Detection of intrusions and malware, and vulnerability assessment: 11th international conference, dimva 2014, egham, uk, july 10-11, 2014. proceedings. chapter Phoenix: DGA-Based Botnet Tracking and Intelligence. (pp. 192–211). Springer International Publishing. URL: http://dx.doi.org/10.1007/978-3-319-08509-8_11. doi:10.1007/978-3-319-08509-8_11.

Sharifnya, R., & Abadi, M. (2013). A novel reputation system to detect dga-based botnets. In *Computer and Knowledge Engineering (ICCKE), 2013 3th International eConference on* (pp. 417–423). doi:10.1109/ICCKE.2013.6682860.

Shiaeles, S. N., Katos, V., Karakos, A. S., & Papadopoulos, B. K. (2012). Real time ddos detection using fuzzy estimators. *Computers & Security*, *31*, 782–790. URL: http://www.sciencedirect.com/science/article/pii/S0167404812000922. doi:http://dx.doi.org/10.1016/j.cose.2012.06.002.

Silva, S. S. C., Silva, R. M. P., Pinto, R. C. G., & Salles, R. M. (2013). Botnets: A survey. *Comput. Netw.*, *57*, 378–403. URL: http://dx.doi.org/10.1016/j.comnet.2012.07.021. doi:10.1016/j.comnet.2012.07.021.

Soniya, B., & Wilscy, M. (2013). Using entropy of traffic features to identify bot infected hosts. In *Intelligent Computational Systems (RAICS), 2013 IEEE Recent Advances in* (pp. 13–18). doi:10.1109/RAICS.2013.6745439.

Sood, A., Zeadally, S., & Enbody, R. (2014). An empirical study of http-based financial botnets. *Dependable and Secure Computing, IEEE Transactions on*, *PP*, 1–1. doi:10.1109/TDSC.2014.2382590.

Symantec (2014). Internet security threat report. URL: https://www4.symantec.com/mktginfo/whitepaper/ISTR/21347932_GA-internet-security-threat-report-volume-20-2015-social_v2.pdf last accessed 22nd July 2015.

Thomas, K., & Nicol, D. (2010). The koobface botnet and the rise of social malware. In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on* (pp. 63–70). doi:10.1109/MALWARE.2010.5665793.

Venkatesh, G., & Nadarajan, R. (2012). Information security theory and practice. security, privacy and trust in computing systems and ambient intelligent ecosystems: 6th ifip wg 11.2 international workshop, wistp 2012, egham, uk, june 20-22, 2012. proceedings. chapter HTTP Botnet Detection Using Adaptive Learning Rate Multilayer Feed-Forward Neural Network. (pp. 38–48). Berlin, Heidelberg: Springer Berlin Heidelberg. URL: http://dx.doi.org/10.1007/978-3-642-30955-7_5. doi:10.1007/978-3-642-30955-7_5.

Wang, B., Li, Z., Li, D., Liu, F., & Chen, H. (2010). Modeling connections behavior for web-based bots detection. In *e-Business and Information System Security (EBISS), 2010 2nd International Conference on* (pp. 1–4). doi:10.1109/EBISS.2010.5473532.

Xiang, Y., Li, K., & Zhou, W. (2011). Low-rate ddos attacks detection and traceback by using new information metrics. *IEEE Transactions on Information Forensics and Security*, *6*, 426–437.

Xiang, Y., Zhou, W., & Guo, M. (2009). Flexible deterministic packet marking: An ip traceback system to find the real source of attacks. *IEEE Transactions on Parallel and Distributed Systems*, *20*, 567–580.