A Taxonomy of Botnet Behavior, Detection and Defense

Sheharbano Khattak, Naurin Rasheed Ramay, Kamran Riaz Khan, Affan A. Syed, and Syed Ali Khayam,

Abstract—A number of detection and defense mechanisms have emerged in the last decade to tackle the botnet phenomenon. It is important to organize this knowledge to better understand the botnet problem and its solution space. In this paper, we structure existing botnet literature into three comprehensive taxonomies of botnet behavioral features, detection and defenses. This elevated view highlights opportunities for network defense by revealing shortcomings in existing approaches. We introduce the notion of a *dimension* to denote different criteria which can be used to classify botnet detection techniques. We demonstrate that classification by dimensions is particularly useful for evaluating botnet detection mechanisms through various metrics of interest. We also show how botnet behavioral features from the first taxonomy affect the accuracy of the detection approaches in the second taxonomy. This information can be used to devise integrated detection strategies by combining complementary approaches. To provide real-world context, we liberally augment our discussions with relevant examples from security research and products.

Index Terms—bot, botnet, botmaster, C&C, DNS flux, IP flux, spambot, stepping-stone, cyberwarfare, DDoS, spam, cyberfraud, fast flux service network, bot family, complex event processing

1 Introduction

Botnets pose an alarming–and arguably *the* most potent–threat to the security of Internet-connected users and systems. As a response to this persistent yet rapidly-evolving threat, hundreds of scientific reports have been published on botnet architectures, economics, detection and defense. Despite continuously growing interest in this domain, the vast body of literature on botnets remains largely unstructured. While some surveys and taxonomies of botnet behavior, detection and defense have been proposed [1], [2], [3], [4], [5], [6], these efforts only address a subset of the entire botnet phenomenon.

This paper presents, to the best of our knowledge, the first systematic analysis of the botnet threat from three aspects: botnet behaviors/architectures, detection mechanisms, and defense strategies. This approach provides a panoramic view of the problem as well as the solution space, highlighting both pitfalls and opportunities for a robust defensive solution. Hence we contend that our taxonomies will aid in visualizing the diversity in botnet research, and in making informed decisions when devising new detection and defense mechanisms.

Our first taxonomy classifies botnet features based on their behaviors. Different phases in the life cycle of a botnet, such as host infection, rallying and command and control (C&C) communication, provide a high-level behavioral view. This behavioral landscape is complicated by evasion techniques and topological choices of botnet creators. We provide an extensive overview of all of these behavioral aspects, and posit that most of the past, present and future botnets can be entirely described and categorized with the help of this taxonomy.

The second taxonomy classifies different approaches for botnet detection. To cater to the interests of readers with diverse goals, we

- *S. Khattak, N. R. Ramay, K. R. Khan and A. A. Syed are with SysNet, National University of Computer and Emerging Sciences, Islamabad, Pakistan.*
  E-mail: {sheharbano.k,naurin.ramay,krkhan,affan.syed}@sysnet .org.pk
- *S. A. Khayam is with PLUMgrid, Inc.440 North Wolfe Rd., Sunnyvale CA 94085*
  E-mail: akhayam@plumgrid.com

highlight different bases or 'dimensions' which can be used to classify botnet detection approaches. Consequently, this taxonomy reveals previously-unexplored botnet detection dimensions which can/should be pursued in future research.

Our third taxonomy classifies botnet defense strategies. We identify shortcomings and strengths of individual defense mechanisms to highlight areas where further research is needed.

The proposed taxonomies reveal an inherent connection between botnet behavioral features and detection approaches. We show that the selection of botnet behavioral features (from our first taxonomy) have a direct impact on the accuracy of the detection approaches (from the second taxonomy). Network security research and products can use this information to evaluate the efficacy of different detection approaches for specific threats. Furthermore, complementary detection approaches can be combined to devise an integrated botnet detection and defense solution.

We conclude by discussing some botnet trends that we expect to continue into future.

2 Taxonomy of Botnet Behavior

A botnet is a collection of compromised machines (*bots*) receiving and responding to commands from a server (the *C&C server*) that serves as a rendezvous mechanism for commands from a human controller (the *botmaster*) (Fig. 1). To evade detection, the botmaster can optionally employ a number of proxy machines, called *stepping-stones*, between the C&C server and itself. Machines are infected by means of a malicious executable program referred to as *bot binary*. Bots belonging to the same botnet form the *bot family*. The ultimate goal of a botnet is to carry out malicious activities or attacks on behalf of its controller.

To fully understand the botnet phenomenon, it is important to systematically explain different features related to botnet behavior[1]. We explain the typical botnet life cycle with reference to our proposed taxonomy of botnet features (Fig. 2). From the point of view of a bot, the infection starts with execution of the bot binary on the victim machine. Bot binary is transported to the victim machine using a *Propagation* mechanism. The next step is to contact the C&C server and announce its presence. This is called call-home mechanism or *Rallying*. Rallying marks the establishment of a *C&C* channel through which the bot receives updates and commands. Based on how C&C communication takes place, the botnet forms an overlay *Topology*. The newly recruited bot then waits for commands to serve the actual *Purpose* of the botnet and optionally spread the infection to other hosts using *Propagation* mechanisms. An important consideration through all the botnet operations is *Evasion*. Different mechanisms are employed to ensure that the bot binary, C&C communication, C&C server(s) and botmaster may not be trivially detected.

In this taxonomy, we broadly categorize botnet behavioral features as those concerning *Propagation*, *Rallying*, *C&C*, *Purpose* and *Evasion*. The rest of this section further classifies these high-level botnet features.

2.1 Propagation

One of the primary goals of a botnet is to continuously increase its footprint in terms of number of bots. Most bot binaries have inbuilt mechanisms to facilitate its propagation to other hosts. Depending on the degree of required human intervention, propagation mechanisms can be broadly classified as active and passive.

2.1.1 *Active*

In this mode of propagation, the botnet is capable of locating and infecting other hosts without any (human) user intervention.

A predominant active propagation mechanism is *Scanning*. A scanning bot probes other hosts in the network looking for one or more vulnerabilities to exploit. The vulnerability exploit helps the botnet in gaining administrative privilege to the victim machine which is typically followed by installation of the bot binary and eventually C&C communication ensues. Some botnets borrow their propagation tactics

[1] For the rest of this document, the term botnet features refers to botnet behavioral features.

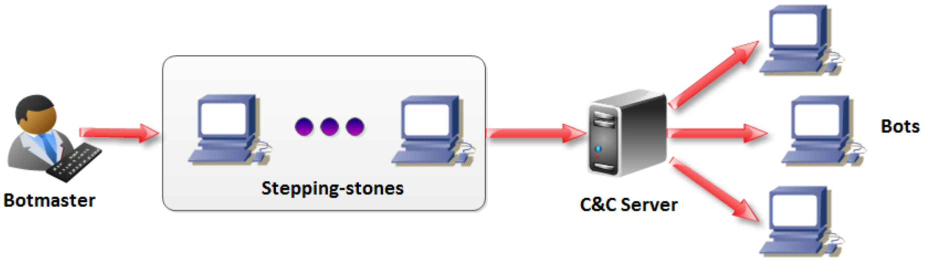

Fig. 1. Structure of a typical botnet.

from worms. They make copies of themselves and propagate automatically, aiming to infect as many hosts as possible. The worm may not necessarily include the main bot binary, however, it prepares ground for future bot binary installation. Both Storm [7] and Sinit [8] exhibited worm-like behavior for propagation. However, Sinit's use of random scanning for peer-discovery, instead of a well-defined bootstrap process, resulted in poor overall network connectivity. Sality [9] was found to conduct a horizontal, covert and coordinated scan of the entire IPv4 address space during a 12-day span using $\approx$ 3 million distinct IP addresses.

2.1.2 Passive

Passive propagation requires some level of user intervention. Next we describe the three most widely used passive propagation mechanisms.

Drive-by Download: Some websites have been either compromised, or specially crafted for automated installation of bot malware on machines of visitors. These websites contain cleverly crafted 'active content' (such as JavaScript or ActiveX controls) which automatically initiate download of the malware to the visitor's machine.

Infected Media: Botnets can also find new bots to recruit by sharing of infected media (e.g., USB hard drives). This is a powerful method of propagation as it can potentially spread the infection to private networks not connected directly to the Internet. Stuxnet [10] is a highly targeted botnet which allegedly hurt Iran's nuclear program by causing sensitive equipment to malfunction. It initially infected Iran's uranium enrichment infrastructure through this mode of propagation.

Social Engineering: An underestimated but powerful method of bot recruitment is through social engineering. Social engineering encompasses all methods that entice the user to willingly download the bot binary. Some botnets exploit the culture of trust prevalent in social networks by posting catchy messages from users' (hijacked) accounts. For example, Koobface [11] tricked users into clicking on a link that pointed to a fake YouTube website. The user was then asked to download specific executable file to watch the video which was actually malware that turned the machine into a bot. Another popular medium for social engineering is emails with interesting subjects and content, enticing users to download email attachments. Storm [7] sent spam emails with catchy subjects that contained malicious links to install the bot binary on victim machines.

2.2 Rallying Mechanism

Rallying is the process used by bots to discover their C&C servers. This marks the formal registration of a newly infected bot with the botnet. Some commonly used rallying methods are described next.

2.2.1 IP address

In this method, the IP address or some means to get the IP address of the C&C server is provided along with the bot binary. IP addresses can be hardcoded or dynamically assigned.

Static Hardcoded IP Addresses: The IP address of a C&C server can either be provided as part of the bot binary (binary hardcoding) or separately (seeding).

In binary hardcoding, the IP address of the C&C server is hardcoded into the bot binary.

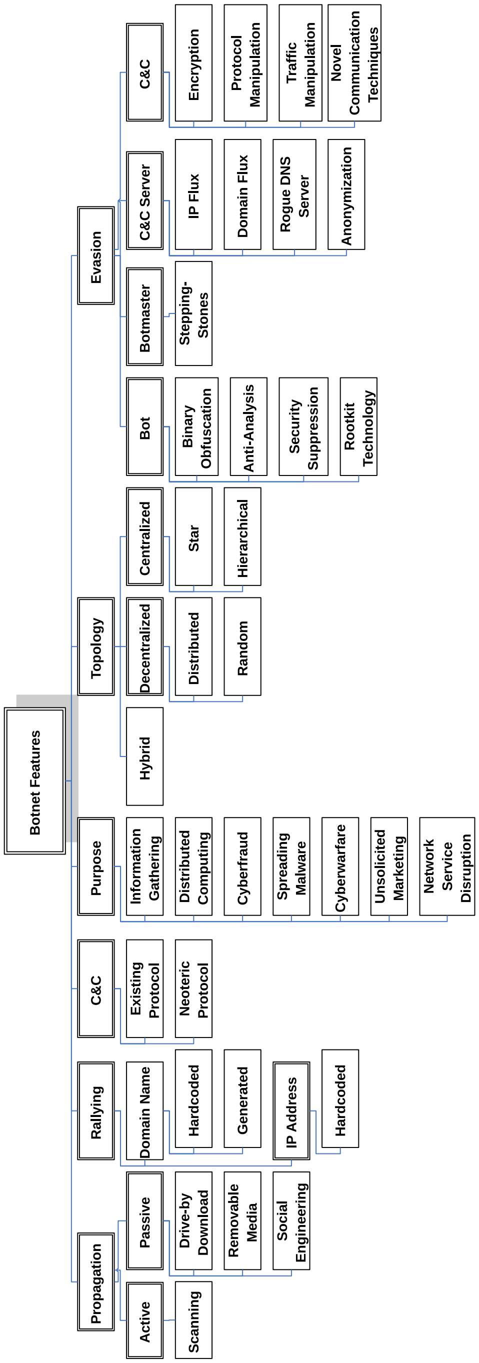

Fig. 2. Taxonomy of Botnet Behavior.

Botnets were tempted to gravitate towards binary hardcoding because it eliminates the use of DNS from the picture, making their activities stealthier. However, this is a rather primitive method of rallying. An obvious pitfall to this is that reverse engineering the bot binary may reveal the C&C server, potentially leading to C&C server hijack. Another disadvantage of this approach is that a network admininstrator can easily blacklist C&C IPs at a network gateway using an ACL, thereby severing call-back channels of all the bots.

Seeding is primarily used by p2p botnets. At the time of infection, the bot is provided with an initial list of peers. The list reflects a group of active peers in the botnet and is regularly updated. The peer list is separate than the bot binary and can be hidden anywhere on the infected machine with an elusive name. For example, Kelihos/Hlux, a p2p botnet, stored its peer list in the Windows registry under *HKEY_CURRENT_USER/Software/Google* together with other configuration details [12]. Reverse engineering the bot binary does not necessarily reveal the peer list.

Some botnets use methods that combine seeding with binary hardcoding. Nugache [13] provides a good representative of such a botnet. Initial seeding is done either by preseeding the victim machine's Windows Registry with a peer-list before actually running the malware, or by obtaining the list from a small set of default hosts hardcoded into the bot binary [14]. The former case falls under seeding while the latter is more typical of binary hardcoding.

2.2.2 Domain name

In this case, the bot is provided with domain names of potential C&C servers. In some cases, the domain name itself does not belong to the C&C server; rather it acts as a stepping-stone or a link to facilitate communication with the actual C&C server. A botnet may utilize additional services such as DDNS and rogue DNS server to maximize its lifetime and make its C&C structure more resilient.

Hardcoded: Like IP addresses, domain names belonging to C&C servers can be hardcoded into the bot binary. This is a better approach than IP address hardcoding from the botnet point-of-view. If the IP address associated with the domain name is taken down, the bot master can still carry out its malicious activities by mapping the domain name to a new IP address while requiring no updation on the bot end.

Generated: Botnets can dynamically generate domain names by using algorithm (Domain Generation Algorithm) known to the bot and the botmaster. This makes the job of law enforcement agencies difficult. Taking down a domain is a complicated process involving several formalities. By the time the older domain is taken down, the botnet has typically already moved to a new domain. This is called bot-herding.

2.3 C&C

Without C&C communication, a botnet is just an incoherent, random collection of infected machines. In other words, C&C communication forms the backbone of a botnet. Ideally, C&C communication should utilize a mechanism that involves minimum latency combined with simplicity, availability and stealth. C&C communication can either leverage existing communication protocols or use custom-made protocols for this purpose.

2.3.1 Existing Protocol

For a botnet, there are several incentives to use an existing protocols for carrying out its C&C communication. Existing protocols have been tried and tested and are less likely to have bugs compared to custom protocols. Also, using existing protocols enables C&C communication to mix with regular traffic making detection difficult.

In their infancy, IRC was the C&C medium of choice for botnets. IRC is widely deployed across the Internet and several public IRC networks are in existence. It has simple text based command syntax and provides almost real time communication between bots and C&C server. IRC remained dominant in botnet C&C for some time but is slowly being replaced by other protocols. The use of IRC is not common, particularly in enterprise networks. Also, the message format of the standard implementation of IRC is unique, making IRC traffic easily distinguishable from normal traffic. Agobot, Spybot, and Sdbot are some popular IRC based botnets [15].

After the relative success of law enforcement agencies and industry in tackling the issue of IRC botnets, the next step in botnet evolution was HTTP C&C communication. In HTTP-based botnets, bots contact C&C server periodically to fetch commands. Blocking of HTTP traffic is not a viable option for most organizations and corporate networks. Besides, HTTP is the most common protocol used on the Internet making it ideal for C&C communication. Use of HTTP as the C&C protocol results in a centralized botnet structure. In the context of larger botnets, some strategy must be adopted to keep the C&C server from being overwhelmed if all the bots happen to contact it simultaneously.

Peer-to-peer (p2p) networks, originally developed to facilitate file sharing among peer nodes, have been utilized for botnet C&C communication. Commands can be dispersed using any node in the p2p network, making detection of C&C servers very difficult. In addition, p2p traffic classification is a daunting task, which makes it hard for gateway security devices to detect and filter p2p traffic. Several protocols are available for p2p-based C&C communication, such as WASTE, BitTorrent, Kademlia, Direct Connect, Gnutella, and Overnet. Slapper [16] and Sinit [8] are the forerunners of the current breed of p2p botnets, followed by Phatbot [17], Storm [14] and Nugache [13].

Researchers have shown how Skype (proprietary protocol) [18] and VOIP (e.g. Session Initiation Protocol) [19] may be used in future for C&C communication.

2.3.2 Neoteric Protocol

Botnets can use proprietary application-level protocols for C&C communication. This helps to evade detection approaches that rely on traffic classification, as C&C traffic is not discernible. However, this very property can raise suspicion as C&C traffic stands out from regular traffic which can be easily recognized. Researchers [20], [21] have used automated protocol reverse engineering to understand custom C&C protocols.

Botnet C&C can also use existing applications for C&C in a way in which they were not intended to be used. The recent Web 2.0 explosion has resulted in a plethora of services focused on user generated content. In particular, social networks have generated an enormous web following. Because of their huge size and dynamic nature, it is impractical to monitor or inspect all the user generated content. A botmaster can use any fake profile on social networks such as Facebook or Twitter as its C&C server, where commands are published as ‘feed’ or ‘status’. The longevity of the botnet depends on its ability to generate fake profiles and convey this information to bots. Researchers have analyzed the feasibility and dynamics of social networks as C&C medium [22], [23].

Whitewell [24] used Facebook accounts as stepping-stones in establishing its C&C. The bot agent first accessed a Facebook account to retrieve configuration information including URLs pointing to C&C servers. Actual commands were received from the addresses pointed to by these URLs. Torpig used Twitter search trends in its domain generation algorithm while the actual C&C communication took place over HTTP [25]. Sninfs is another botnet that made use of Twitter to distribute C&C instructions to download information-stealing malware on bots [26].

Botnets appearing in the wild that purely rely on social networks for C&C have been shrugged off by security researchers as proof-of-concept efforts. Social networks are deemed infeasible for C&C communication because they present a central point of control which can be taken down easily [26]–in contrast to HTTP-based botnets where taking down C&C server(s)is an arduous task involving third-party cooperation and possibly legal intervention. The task is further facilitated by the cooperative attitude of owners of these social networks who have a common interest in protecting their users.

2.4 Purpose

The main motive of a botmaster in recruiting and managing hundreds and thousands of bots is to use their combined power to carry out malicious activities on its behalf. We call this the purpose of the botnet. In contrast to other malware, motives behind botnet operation are of much graver nature. The botmaster is able to derive large financial gains while remaining clandestine. The odds of being detected are also very low as the botmaster uses machines belonging to unwitting, innocent users for carrying out malicious activities. In addition to financial incentives, botnets are also driven by other goals such as intellectual property stealing, spying and cyberwarfare. We now discuss some of the prime purposes which are being served by contemporary botnets. Note that a botnet, or any subset of bots in it, can serve many different purposes at different times as long as the requisite C&C commands are understood by the bots.

2.4.1 Information Gathering

Information may be gathered for financial gains or reconnaissance purposes. Bots are equipped with a variety of tactics to steal sensitive data and credentials, such as credit card numbers and bank account numbers from infected machines. Botnets are also employed to gather information related to a rival party (a nation or a company), for the purpose of reconnaissance.

Aurora was a specialized botnet that came into limelight following Google China's public disclosure of being victimized by it. It was responsible for stealing intellectual property from several organizations located in different countries [27]. Ghostnet is another example of a botnet used for cyber espionage. It infiltrated high-value political, economic and media locations in several countries, including embassies, foreign ministries and other government offices [28].

Both Aurora and GhostNet are representatives of an emerging class of cyber threats called Advanced Persistent Threats (APT). APT is a category of cybercrime aimed at political and business assests. The attackers make use of all the latest developments in intrusion technologies. Overlooking immediate financial gains, they focus on a specific target. The attack is stealthy and continues over a long stretch of time, and may continue even after the key goals of the attacker have been achieved. The attacker usually has a greater objective than financial gain, is well funded, knowledgeable and well organized [27].

Information gathering can also be financially motivated. During the ten days when Torpig was hijacked, a goldmine of data was recovered including online bank accounts and credit and debit cards [25].

2.4.2 Distributed Computing

Desktop computers typically utilize only a small percentage of the available computing power. Botnets utilize maximum potential of the bots by using their storage and processing power to host and share files, perform distributed password cracking, or any other computational activity of distributed nature [29].

2.4.3 Cyberfraud

Cyberfraud refers to online activities related to deliberate deception for unfair gains. Botnets are used for carrying out cyberfrauds. Web-Phishing entices unsuspecting visitors into performing actions that they would not commit if informed about their consequences. Serving such content on authentic webservers runs the risk of being terminated by hosting authorities. Some botnets use bots to host pages of botmaster's choice by installing a stripped-down version of a webserver on the infected machine. Torpig, an information stealing botnet, used web-phishing to harvest senstive information from infected machines. When a user infected by Torpig visits a target website as specified in the bot configuration file, the original webpage to be displayed is replaced with a fake, but identical, page. The information entered by the user, such as passwords and credit card crdentials is conveyed to a drop-zone for the botmaster's benefit [25].

Botnets can be used to rig the results of online games and polls by ordering bots to act on the botmaster's behalf. Botnets also commit clickfrauds by directing bots to click on pay-per-click advertisements displayed on websites to yield biased click statistics. More recently, botnets have been found to manipulate search engine rankings for key search terms to attract traffic to fraudulent websites [30].

2.4.4 Spreading Malware

Some botnets are known to be used for launching other malware. The malware may be part of the botmaster's scheme or the botnet's services may have been rented for this purpose. ZeuS and Pushdo got installed on victim machines by piggybacking on other malware [31], [32].

2.4.5 Cyberwarfare

Cyberwarfare involves measures taken by a state to disrupt or damage another state's assets by penetrating its computers and networks. The last couple of years have seen botnets being increasingly used for cyberwarfare. Cyberspace is regarded as an important area for countries to gain strategic edge over each other. The concept gained momentum when a massive Distributed Denial of Service (DDoS) attack was launched against Estonian websites in 2007, allegedly by Russia [33]. Stuxnet, discovered in July 2010, took nation-sponsored cyberwarfare to the next level. It was a highly targeted and sophisticated piece of malware, which revolutionalized the cyberwarfare landscape. Stuxnet affected Iran's nuclear sites by manipulating the Programmanble Logic Controller (PLC) used to control uranium enriching centrifuges [10].

2.4.6 Unsolicited Marketing

Online Marketing has proved to be more effective than traditional marketing methods, being instant yet cheap. However, this has been abused by some marketers by subjecting users to unsolicited advertisements in the form of spam emails, pop up ads etc. Several steps have been taken in the last few years to discourage this trend. Blacklisting was used effectively against mail servers responsible for sending spam. However, spammers have found an alternative in the form of botnets. Botmasters supply email templates to bots which send spam emails based on these templates. Spreading the task of spam sending over hundred of bots has made detection very difficult as each bot is responsible for sending only a small number of spam emails.

Rustock, Pushdo, Bagle, Bobax and MegaD are examples of botnets whose names have become synonymous with spam. Researchers tried to estimate spam economy by studying a number of recovered C&C servers. The largest email list was estimated to be worth 10,000-20,000 dollars and profit of the botnet's operators for offering spam services was valued at 1.7-4.2 million dollars [34].

2.4.7 Network Service Disruption

The combined power of the bots, often running into thousands, can be used to bring down legitimate Internet services. Botnets were originally created in the context of IRC networks. A popular attack against IRC networks was the clone attack in which all the bots try to connect to an IRC network using clones. Clones are IRC clients controlled by programs/scripts. The resultant traffic overwhelms the IRC network by exhausting its resources. Resultantly, the network is brought down.

Botnets are also used to carry out DDoS attacks. Thousands of bots send requests to the victim service over a short period of time, causing the service to crash and consequently become unavailable to legitimate clients. The ability to launch DDoS attacks and make websites and critical services unavailable is also used for cyber-extortion. Large businesses and enterprises are willing to pay extortion money to botmasters rather than losing sales and credibility. Network service disruption can potentially transform into cyberwarfare if the attack is motivated by a state trying to disrupt another state's cyber infrastructure, as was evident in the case of cyber-attack against Estonia [33].

In a proof-of-concept experiment, researchers showed how DDoS can be trivially launched against a target by creating a specially crafted Facebook app. Using the app, users unwittingly generated traffic towards a victim [35].

2.5 Evasion

Botnets operate stealthily to evade detection and increase their probability and duration of survival. We can view the evasion strategies adopted by a botnet from the perspective of the bot, botmaster, C&C server and C&C communication.

2.5.1 Evasion tactics at Bots

For bots to remain available to the botmaster for an extended period of time, a number of mechanisms are employed to evade host-based detection. We discuss some of these mechanisms now.

Binary Obfuscation: Bot family expansion occurs by exploitation of vulnerabilities on machines that are subsequently infected by the bot-binary. The bot-binary incorporates mechanisms to coordinate with the botmaster to receive commands. To avoid being detected by host-based security applications, several evasion techniques are employed to conceal the bot-binary. Pattern-based detection approaches are defeated by the use of polymorphism. Polymorphism refers to the ability of the bot-binary to exist in several forms. One of the ways to achieve this is by using encryption. The same effect can also be achieved by packing the bot-binary. Packing refers to file condensation. In the context of malware, packing helps obfuscate the malicious code. Some packers are able to produce new binaries every time the original malicious executable is packed.

While code polymorphism succeeds in concealing the bot-binary from pattern-based security applications, it can still be detected by memory-based detection approaches. When executed, the bot-binary needs to be decrypted or unpacked resulting in the same code. This problem is taken care of by code metamorphism. It allows for the bot-binary to be rewritten into different, but semantically equivalent code to defeat memory-based detection approaches

Anti-Analysis: Researchers analyze botnet behavior by running bot binary on virtual machines or sandbox such as [36]. Another method for botnet analysis is to use honeypots that emulate known software and network vulnerabilities to be infected by botnet(s). Honeypots are designed to be self-contained and prevent the spread of botnet beyond the honeypot. To evade such analysis, some bot-binaries perform checks to determine the environment in which they are being executed. If the binary detects a virtual machine or sandbox, it can either refuse to run or it modifies its functionality to evade analysis. After an initial surge of VM-aware botnets, such as Conficker, Rbot, SDbot/Reptile, Mechbot, SpyBot and AgoBot, the trend for such evasion technique is going down due to two reasons [37]. First, legitimate programs rarely perform tests for the execution environment, and thus unnecessarily flags the binary as suspicious. Secondly, virtual machines are now popular among ordinary users who are a legitimate (and ever growing) target, rather than being restricted for security analysis.

Security Suppression: After successfully infecting a machine, a botnet may proceed and disable existing security software on the victim machine. If the host is already infected with other competing malware, those are also wiped out. For example, Conficker [38] disables several security related Windows services and registry keys upon installation. It includes a domain name blacklist which it uses to block access to certain security related websites and a process blacklist to terminate processes that may aid in its detection [25].

Rootkit Technology: A rootkit is a program that maintains a persistent and undetectable presence on the infected machine by subverting normal operating system behavior. Botnets may install rootkits on compromised machines to gain privileged access to them. This enables them to carry out malicious activities while bypassing typical authentication and authorization mechanisms. As a result, traditional anti-virus software fails to detect intrusion.

2.5.2 Evasion tactics at C&C Servers

Botnet differs from other malware in the ability of the botmaster to remotely control and coordinate all the infected machines via C&C server(s). This essentially means that the 'brain' of the botnet lies in the C&C server. The same can turn into its Achilles' heel. Therefore, botnets invest considerable resources to conceal the C&C server.

IP Flux: Botmasters utilize IP Flux to frequently change the IP address associated with the domain name of its C&C server. This helps in evading IP based blacklisting and blocking. This phenomenon is also known as Fast Flux and the resultant structure is called Fast Flux Service Network. IP Flux has been facilitated by Dynamic DNS (DDNS) service. Like DNS, DDNS performs domain name-to-IP mapping. In contrast with DNS which handles only static IP addresses, DDNS can operate with dynamic IP addresses too. Botnets make use of DDNS to keep the (C&C server) domain name to

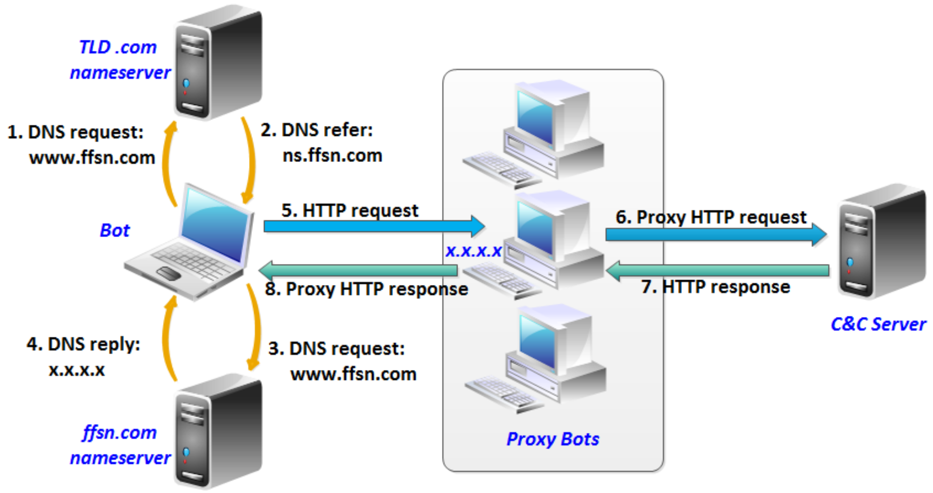

Fig. 3. Fast Flux Service Network (FFSN) (based on [39])

IP address mapping up to date in real-time. Fast flux comes in two flavors; single flux and double flux.

Considering that the bots are aware of the domain name associated with the C&C server, there are two basic steps involved in communicating with it:

i) Resolve the domain name to an IP address. This information will be provided by the nameserver responsible for the requested domain.
ii) Send request to the resolved IP address.

Single flux targets step (ii) in the method explained above. Bots do not communicate directly with the C&C server. There is an intermediate layer of machines that act as proxies and relay communication between bots and C&C server. These machines themselves are compromised by the botnet, and hence we call them 'proxy bots'. The resolved IP addresses correspond to the proxy bots. A different subset of the entire pool of proxy bot IP addresses is associated with the domain name after short intervals of time. The botmaster uses its collection of proxy bot IP addresses in a round robin fashion, so the same IP addresses may reappear in the domain's A record [2] after some time. To ensure that the frequent change in domain's record is seen by all bots, the TTL in the name server's reply is set to be very short, usually a few seconds. Botnets use a domain name for a short period of time before disposing it off and associating a new domain name with the same set of proxy bots.

Double flux takes IP Flux to the next level by extending the concept of flux to the nameserver responsible for resolving the C&C domain name. A request for the C&C domain name will be resolved by a nameserver under the botnet control (step (i)). The IP address corresponding to the nameserver changes frequently. The nameserver's response will include IP address associated with the requested domain name. The resolved IP address also changes frequently and corresponds to the proxy bots, which relay messages back and forth between bots and the C&C server (step (ii)).

Fig. 3 provides an example of IP flux. The bot knows the C&C server domain name, i.e., *www.ffsn.com*. The bot obtains information about the nameserver for *www.ffsn.com* from the Top-Level Domain (TLD) .com nameserver. Querying the authoritative nameserver *ns.ffsn.com* results in *www.ffsn.com* being mapped to the IP address *x.x.x.x*. The IP address *x.x.x.x* corresponds to one of the proxy bots. The relevant proxy bot then acts as a mediator between the bot and C&C server. In double flux, the nameserver *ns.ffsn.com* is also under botmaster's influence in addition to proxy bots.

[2] DNS A (Address) Record performs domain name to IP address mapping.

Domain Flux: Domain flux associates multiple domain names with the same IP address. It helps evade URL-based filtering and detection techniques. Domain flux can be achieved by either leveraging on the existing service of DNS to provide domain wildcarding or by using a domain name generation algorithm. Domain wildcarding allows a higher level domain prefixed with any random string to be associated with the same IP address. For example, consider that *nust.edu.pk* maps to *10.1.1.1*. Through domain wildcarding, **.nust.edu.pk* will also be mapped to *10.1.1.1* where * can be any prefix. Alternately, bots can use an algorithm for domain name generation. This algorithm periodically generates a list of domain names. The list is usually large and not all the generated domain names are active at a given time. The bots identify the domain name currently being used as C&C server by trying to contact all these domain names one by one. The domain name that positively resolves indicates successful C&C server identification. Contacting several random domain names results in a large number of DNS Non-Existent Domain (NXDomain) responses, a possible symptom of botnet infection [40]. The botmaster knows which domain names will be contacted by the bots at a given time. This is possible because the same algorithm is known to the botmaster too. He/she pre-registers the domain names which are expected to be contacted by the bots. Note that the botmaster can associate the same IP address with each domain name registered.

Conficker attracted worldwide attention by affecting millions of computers and reportedly infected key infrastructures and government and military offices. It uses a domain generation algorithm that generates 50,000 domain names per day, out of which 500 randomly selected domains tried to contact with the C&C server [41].

Rogue DNS Server: The attitude towards cyber crime differs greatly among different countries. In particular, local authorities in some countries are indifferent to requests for takedown of malicious servers. Botnets turn this to advantage by having their own DNS service (possibly distributed) hosted in such locations. The addresses of these DNS servers are provided to bots which use them to resolve the address of C&C servers. Rogue DNS servers not only provide a mechanism for carrying out C&C stealthily and effectively, but can also redirect a website's traffic to another bogus website (pharming) or a malicious website focused on stealing sensitive user information (phishing).

Anonymization: Anonymization hides the identity of the sender of a message such that a message cannot be traced back to its sender. Moreover, it cannot be confirmed if two messages were sent by the same sender. While this frustrates network surveillance and traffic analysis attempts, it is a boon for botnets which can carry out C&C communication using low latency anonymization networks, such as Tor [42], to conceal C&C server(s). Bots communicate with the C&C server without knowing its actual location. C&C servers can be hosted as Tor hidden services [43]. Hidden services are a facility provided by Tor [44] to make it possible to offer services, such as web publishing or instant messaging servers while hiding their location.

2.5.3 Hiding C&C Communication

Detection of C&C communication can have various implications. On one hand, detection of C&C can enumerate bots which can be subsequently disinfected. On the other hand, it can expose C&C servers. The latter can have more serious consequences, for example C&C server can be hijacked crippling the entire botnet. At worst, detection of C&C between botmaster and C&C server can reveal the main culprit, the botmaster. In addition to this, discovery of C&C communication can help defenders understand how the botnet operates and exploit this information to damage the botnet. For these reasons, botnets employ several mechanisms to obfuscate C&C communication.

Encryption: Botnets encrypt C&C communication to evade detection. Encryption causes content based analysis to fail, forcing researchers to rely on other traffic characteristics, such as packet arrival times and packet length. Moving from simple obfuscation techniques to elaborate encryption schemes, botnets have made C&C communication virtually impenetrable. Nugache, touted to be one of the most advanced botnets [45], uses a sophisticated scheme for C&C encryption. A variable bit length RSA key exchange is followed by seeding symmetric Rijndael-256 session keys for each peer connection. Keystroke log files are also encrypted using Rijndael with the help of a key derived from some peer-specific information [13].

Protocol Manipulation: Some botnets use protocol tunneling to disguise C&C communication. Typically, firewalls allow HTTP traffic. Leveraging this configuration, botnets have started to use HTTP tunnels for C&C communication. Another emerging trend is the use of IPv6 tunneling for C&C communication [46]. Most modern operating systems support IPv6 by default. However, many intermediate devices do not recognize IPv6 traffic. IPv6 tunneling allows transportation of IPv6 packets over incompatible intermediate devices that only support IPv4. Many firewalls and IDS either do not support IPv6 or are misconfigured, limiting their abilitiy to detect or filter IPv6 traffic. This can be exploited by botnets to carry out C&C communication while bypassing security measures.

Traffic Manipulation: A very active C&C communication may tip off security applications about the botnet. Botnets might purposely create low volume C&C traffic spread over relatively large periods of time to defeat statistical and volume based detection techniques.

Novel Communication Techniques: Botnets use novel communication techniques for C&C which cannot be trivially detected. Social networking websites, in particular Facebook and Twitter, are known to be used by botnets for C&C. Brazen [22], an information-stealing botnet, used Twitter to disseminate links that contained commands or executables to download. Bots subscribed to the malicious Twitter account using RSS to get status updates. Additionally, botnets may utilize any of the existing techniques for information hiding for C&C communication. Some possible candidates are the metadata in image files and least significant bit encoding in image files.

2.5.4 *Evasion by the Botmaster*

Botmaster is the most protected component of a botnet. Botmaster detection can lead to losing control of the botnet. Moreover, the botmaster can potentially face legal consequences in the form of a prison sentence and/or hefty fines. Therefore, botmasters utilize elaborate mechanisms to evade detection.

Stepping-Stones: Botmasters generally hide their true identity by setting up a number of intermediate hosts, called stepping-stones, between the C&C server and themselves. These stepping-stones can be network redirection services, for example proxies such as HTTP or SOCKS, and SSH servers. The stepping-stones themselves are hosts compromised by the bot master. Botnets prefer to set up stepping-stones in countries with lax cyber crime legislature. This complicates things for law enforcement agencies by necessitating the cooperation of organizations in other countries. Also, the trace back mechanisms are severely crippled because network redirection services operate at the application level and discard off all lower layer information before relaying messages to the next hop.

Botmaster can use an anonymization network as a stepping-stone. This offers the additional benefit of obscuring botmaster's IP address, making it very hard to traceback the botmaster. Defenders can leverage the inherent drawback of anonymity networks, i.e., traffic monitoring cannot be defeated at boundaries of the anonymity network. Detection measures deployed at Internet edges, before a botmaster's traffic enters the anonymity network, can yield promising results. However, it is extremely challenging because a botmaster generates very little traffic and this traffic is not easily distinguishable from legitimate traffic.

2.6 Topology

Another aspect in the taxonomy of botnet features is the topology of botnets based on how bots communicate with the C&C server(s). This topology is essentially an overlay network and is agnostic to the underlying physical topology. Our classification draws inspiration from a report by Damballa [5] on botnet communication topologies.

2.6.1 Centralized

In a centralized topology, all the bots report to and receive commands from a single C&C server. Thus its an easy implementation with minimum C&C overhead. However, it is also a single point-of-failure for the entire botnet. To deal with this problem, techniques such as IP flux and domain flux have been adopted by botnets over time. Botnets using HTTP and IRC as means of C&C communication are typical examples of centralized botnet topology.

Star: The basic star topology was adopted by many initial IRC-based botnets. It is a simple model in which the bots directly communicate with the botmaster. An obvious advantage to this is increase in the speed of communication between bots and the botmaster. However, this topology also suffers from the problem of central point of failure–if the central C&C server is taken down, the entire botnet gets disbanded.

Hierarchical: Driven by the desire to conceal the botmaster, botnets incorporate one or more layers of proxies between the bots and themselves. The proxies themselves are compromised machines serving the botmaster. This renders a hierarchical quality to the resultant topology, hence the name. There is little probability that analyzing activities of a bot will expose the C&C server. Even if one of the immediate bot proxies is taken down, the botnet will still be functional. The hierarchical nature also allows for portions of the botnet to be separately rented out to third parties. Because of the multiple layers of indirection, communication between the bots and the botmaster is bound to experience some latency.

2.6.2 Decentralized

In decentralized topology, no single entity is responsible for providing command and control to bots. Bot management is either distributed among multiple C&C servers or there is no obvious master-slave relationship between bots and C&C server.

Distributed: In a distributed topology, multiple servers control a subset of the bot family. The servers are able to communicate directly with each other. Typically, the servers are spread over different geographical locations. This allows for fast communication by allocating similarly located bots to the nearest C&C server. This also entails benefits of load distribution, availability and resilience. If one server is taken down, its bots can be distributed among the remaining servers. Besides, the issue of central point of failure has also been tackled as it is unlikely that all the countries hosting the C&C servers will simultaneously and positively entertain legal take down requests. The resulting trade-off, for the botmaster, is increased complexity of the botnet implementation.

Random: In a random topology, there is no clear master-slave relationship. Any bot can be used to issue commands to other bots in the botnet.

In peer-to-peer (p2p) networks, communication between bots and the botmaster forms unpredictable routes. The botmaster can use any peer node to float commands which will be broadcast to all the bots. The absence of centralized C&C makes it extremely difficult to locate the botmaster or hijack the botnet. In hierarchical topology, shutdown of a proxy server may result in a portion of the botnet becoming dysfunctional. In contrast, in p2p topology, 'cleaning' of a single bot will have no effect as alternate routes are always available. A disadvantage to the p2p based random toplogy is that the C&C communication will experience unpredictable delays making it unsuitable for carrying out coordinated, large-scale attacks. Furthermore, capture of a single bot reveals several other bots because each bot maintains a peer-list.

Cooke et al. [47] proposed a topology that has not been observed in real world botnets but gives insight into possible future botnet trends. In this topology, each bot knows about only one other bot. The botmaster can use any bot to issue a command which is encrypted and passed on to the next bot discovered through random scanning of the Internet. This demonstrates an extremely resilient model with high survivability.

2.6.3 Hybrid

An interesting topology for botnets would be a combination of centralized and decentralized topologies, for example, a botnet using centralized structure between C&C server and the front-end proxy bots but p2p as C&C for the bots under control of individual proxy bots. At the time of writing of this paper, we are not aware of any existing botnet that utilizes a hybrid topology. Further research is needed to investigate the pros and cons of such a structure.

2.7 Summary: Taxonomy of botnet behavior

As a first step to understand the botnet phenomenon, it is important to systematically explain botnet behavior. In this section, we presented a taxonomy of botnet behavior. Our high-level categorization of botnet behavior comprises of *Propagation*, *Rallying*, *C&C*, *Purpose*, *Evasion* and *Topology*.

Propagation mechanisms refer to the methods used by botnets to infect other machines. Based on the degree of (human) user intervention required, we classify propagation mechanisms as *Active* and *Passive*. In *Active* mode of propagation, the botnet is capable of locating and infecting other machines without requiring assistance from a (human) user. *Scanning* is the only representative of this type of propagation. In *Passive* propagation, infection cannot spread to other machines without user assistance. Subclasses of passive propagation are malware distribution through *drive-by download*, *infected media* and *social engineering*.

Rallying is the process used by bots to discover their C&C server(s). Bots can locate C&C server(s) by *IP address* or *domain name*. Both IP address and domain name of the C&C server can either be *hardcoded* into the bot binary or *generated* using some algorithm.

Once C&C server has been located, *command-and-control (C&C) communication* is used to issue instructions to the bots. These instructions could relate to update/modification of botnet malware, spreading the infection to other machines or other malicious activities. C&C communication can take place either through *existing protocols* or *custom/neoteric protocols*.

Purpose refers to the main motive of the botmaster in recruiting the bots. The most notable motives include *information gathering*, *distributed computing*, *cyberfraud*, *spreading malware*, *cyber warfare*, *unsolicited marketing* and *network service disruption*.

A primary consideration in the operation of a botnet is stealth or *evasion*. Botnets employ a number of mechanisms to evade detection and thus increase their probability and duration of survival. These mechanisms can be broadly categorized on the basis of the component of botnet infrastructure they are trying to obfuscate, i.e., *bots*, *C&C server(s)*, *C&C communication* or *botmaster*. *Evasion tactics at bots* strive to defeat host-based detection. These involve *binary obfuscation* of the bot binary and *rootkits*. Additionally, botnets may also include *security suppression* and *anti-analysis* techniques to evade detection by security software and research tools (such as virtual machines or honeypots) respectively. Being the control center of the botnet infrastructure, botnets set up a number of *evasion tactics at C&C servers* to enable them to function uninterrupted for the maximum possible duration. These tactics involve frequently changing the IP address (*IP flux*) or the domain name (*domain flux*) associated with the C&C server, *anonymization* of the C&C server and/or the use of DNS servers under the control of botmaster (*rogue DNS server*). Another important aspect of evasion is *hiding C&C communication*. This can be achieved through *encryption*, *protocol manipulation*, *traffic manipulation* and/or use of *novel communication techniques*. Finally, *evasion tactics by the botmaster* typically entail the use of network redirection services or anonymization (*stepping-stones*).

Another important aspect in the taxonomy of botnet behavior is the *topology* of botnets based on how bots communicate with the C&C server(s). The high-level classification of botnet topology comprises *centralized*, *decentralized* and *hybrid* topologies. In a *centralized* topology, all the bots report to and receive commands from a single C&C server. *Star* and *hierarchical* topologies are derivatives of the centralized botnet topology. In *decentralized* topology, no single entity is responsible for providing command and control to bots. *Distributed* and *random* topologies derive from the decentralized botnet topology. Finally, the *hybrid* topology represents a combination of centralized and decentralized topologies.

3 Taxonomy of Botnet Detection Mechanisms

Several mechanisms for botnet detection have emerged over time. To the best of our knowledge, none of these techniques unveil all the botnet components at once. Existing detection approaches try to identify a part of the big jigsaw puzzle; i.e., the botnet. One part can lead to the other, and different parts can be placed together to reveal a greater portion of the puzzle, however, there is no panacea. Based on which component is being targeted, we classify botnet detection into different facets; bot detection, C&C detection and botmaster detection. We have intentionally excluded C&C server detection from the aforementioned facets because detection of C&C communication typically reveals C&C servers too.

Our definition of botnet, a distributed malware with bots, C&C server(s), a botmaster, and the C&C communication between these components, is in harmony with the interpretation in existing literature. However, most previous literature refers to detection of single, bot-infected machines as bot detection, and detection of bot families as botnet detection. In view of the typical interpretation of the term 'botnet', this definition of botnet detection is paradoxical. We posit that what is commonly understood as botnet detection is still bot detection, with particular regard to bot families. Gu et al. [48] proposed a detection approach, BotMiner, that could detect bot families or sub-families within the monitored network. To elucidate the scope of their detection framework, they described a botnet as a coordinated group of malware infected machines with similar communication and activity patterns. This means that if the botmaster uses the bot family to carry out n non-overlapping malicious activities, BotMiner would detect n instances of the same bot family, hence the term bot sub-family detection. Another work [49] used the term bot family to refer to botnet infected machines with similar communication patterns. To provide a generalized frame of reference, we define the following terms:

Definition *1. Botnet Detection:* Detection of all components of a botnet, comprising the botmaster, C&C server(s), means of C&C, and (all or a subset of) bots.

Definition *2. Bot Detection:* Detection of botnet infected machines, with or without regard to bot families.

Definition *3. Bot Family Detection:* A class of *Bot Detection* focused on bot family detection.

In this taxonomy, we broadly categorize botnet detection mechanisms as *Bot Detection*, *C&C detection* and *Botmaster Detection*. These facets of botnet detection can be used in combination. For example, *C&C Detection* can be followed by *Bot Detection* and vice versa. Botnet detection approaches can be broadly categorized as *Active* and *Passive* (We later discuss other alternatives in Section 3.4). In active detection, the strategy is to take part in the botnet operation by impersonating as a component of the botnet instead of passively observing its activities. For example, active C&C detection involves online manipulation of network flows to deduce information about possible C&C communication. In contrast, passive detection approaches detect botnets by silently observing and analyzing botnet activities without making a conscious effort to participate in the proceedings.

Our high-level classification of botnet detection mechanisms into 'active' and 'passive' approaches is of practical importance for any researcher or network administrator. Active detection techniques raise several legal and ethical questions, and whether it can/cannot be performed will vary from organization to organization. For this reason, network policies typically prohibit active manipulation of network flows. Another issue that necessitates use of passive detection techniques is users' privacy concerns. Moreover, active detection methods can jeopardize the security of the very hosts they seek to protect. If active detection is discovered by the botmaster, there is a chance that he/she will direct attack traffic to the responsible host(s) as counter defense.

The rest of this section discusses further classification of these high level botnet detection facets; i.e., *Bot Detection*, *C&C detection* and *Botmaster Detection*.

3.1 Bot Detection

Bot detection can be performed with or without regard to bot families. Users and network administrators are usually indifferent to information about bot families. Their primary concern is to protect their systems and networks from infections, regardless of details about bot family. On the other hand, security researchers are particularly interested in identifying bot families. The degree of prevalence of different botnets, their geographical distribution, and the common characteristics of botnets are some of the plausible reasons for their heightened interest. Detection of bots indicates vulnerability of a host or network to botnet infection. This can be followed by remedial strategies aimed at recovering from the infection and preventive measures to avoid getting infected in future.

3.1.1 Active Detection

Active bot detection involves participating in the botnet operation. This typically involves impersonating as a component of the botnet. Active detection approaches involve *Infiltration* and *C&C Server Hijack*.

Infiltration: In infiltration, a defender-controlled machine masquerades as an actual bot and probes the C&C server or other peers in case of a p2p-based botnet to gain details about other bots. Nappa et al. [18] proposed a replay attack on a Skype-based botnet. The technique can be effective for other p2p botnets too. The defender progressively gains information about other bots by repeatedly issuing crafted, bogus messages to declare itself as a new bot and subsequently obtain new peer-list.

C&C Server Hijack: Bots can be actively detected by C&C server hijack. Bots report to and receive commands from C&C server. Taking control of the C&C server will reveal all the bots that contact it. This can be achieved by exploiting botnet rallying mechanism. A defender can use this information to his/her advantage to hijack the server. This approach also leverages knowledge of botnet topology. Centralized botnet structures are more amenable to C&C server hijack. In decentralized botnets, the C&C server can be any peer and will, at most, reveal information about bots in its peer list. To gain futher information, some other techniques need to be employed, such as active crawling of the p2p botnet. The seizure of C&C servers can be *Physical* or *Virtual*.

In a *Physical* hijack, law enforcement agencies physically seize the C&C servers. However, it is possible to take over the C&C servers without involving legal authorities by mutual cooperation. This is possible if the C&C servers in question are not in geographically diverse locations. With the help of service providers, researchers gained access to several C&C servers used by Pushdo/Cutwail botnet [34]. In addition to other interesting information, 24 databases containing details about the bots and spam operations were disclosed.

In a *Virtual* take over, defenders hijack the C&C servers by redirecting C&C communication to a machine under their control. This technique has been used by researchers to hijack botnets that use domain names for rallying. By virtue of DNS sinkholing, traffic sent by bots to known botnet domains can be forwarded to defender-controlled machine. The domain names of C&C servers can be learnt by analyzing the botnet behavior on infected machines. Researchers [38], [25] have recovered future rendezvous points by reverse engineering the domain generation algorithm used by botnets utilizing domain flux.

3.1.2 Passive Detection

Passive detection approaches detect botnets by silently observing and analyzing their activities without making a conscious effort to participate in the proceedings. Passive techniques can be *Syntactic* or *Semantic*.

Syntactic: Syntactic or signature-based approaches identify botnets by comparison with pre-determined patterns of botnet infection obtained from observed samples. Rishi [50] is a purely syntactic approach for detection of IRC

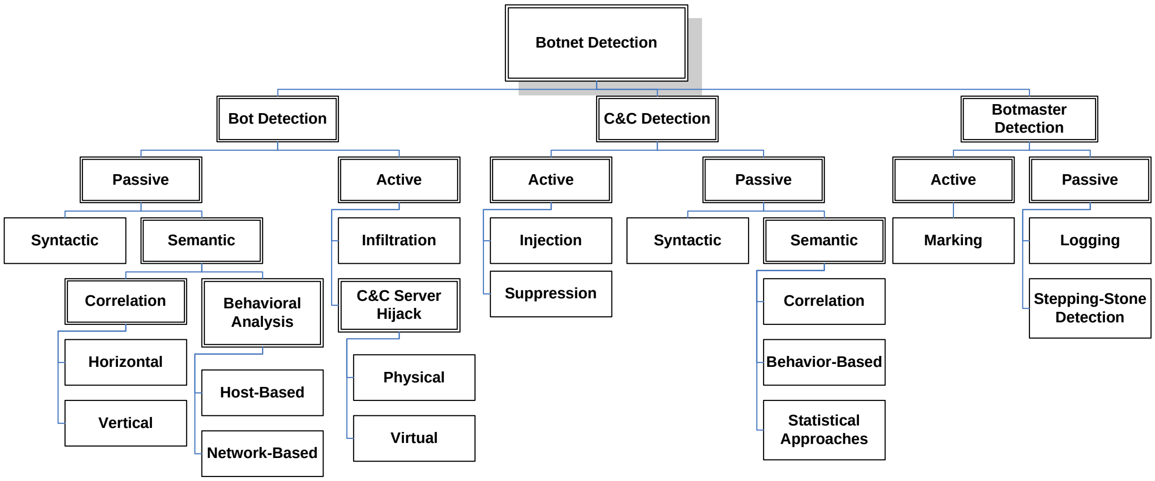

Fig. 4. Taxonomy of Botnet Detection Mechanisms.

based bots. It formulated regular expressions as signatures to identify bot-like IRC nicknames. Snort [51] has a rich signature database and is used by BotHunter [52] along with other anomaly detection components to feed alerts to a correlation module.

Signature-based detection degrades if strong evasion mechanisms such as encryption and bot binary obfuscation are in place. New threats for which signatures have not been developed go completely undetected. Hence, this method should be complemented with behavior-based detection approaches.

Semantic: Semantic detection methods use the context of events and protocol information to announce detection of malicious behavior. The process of bot detection entails careful analysis and is based on deviation from established benign behavior or similarity with bot behavior. Semantic detection techniques can be further classified as those based on *Correlation* and *Behavioral Analysis*.

Correlation techniques are used to identify bots as well as bot families. Botnet is a coordinated attack infrastructure. This idea has been used to cluster hosts that perform similar activities or communication. Correlation-based techniques have been shown to successfully detect bots utilizing centralized as well as decentralized topology.

Correlation can be performed on primary data as well as secondary data. Primary data encompasses all the activities vital to the sustenance of the botnet like egg download, propagation activities, and C&C communication. Attacks and malicious activities fall under secondary data. Secondary data analysis is an effect-to-cause approach and relates malicious behavior to bots. It cannot be used as a generic method to detect all kinds of botnets because it is restricted to the attack characteristics that the analyst is trying to target. Researchers have tried to detect and study different bots by correlating similar spam emails collected from records of popular email service providers [53]. Ramachandran et al. [54] detected botnets by monitoring and correlating DNSBL queries which botnets perform as a way of reconnaissance before launching a spam campaign. Researchers have also detected botnets by analyzing and correlating anomalous DDNS [55], DNS traffic [56] and queries to search engine [57].

Complex Event Processing (CEP) is an area that closely relates to correlation. It correlates events in real-time to detect a target complex event comprising of multiple simple or complex events. The concept has recently been applied to the area of information security [58], [59], [60]. Given its complex nature, botnet detection seems to be a suitable candidate to be mapped as a CEP problem. However, we are not aware of any research effort that attempts this mapping.

Correlation can be further divided into two main branches; *Vertical* and *Horizontal* correlation.

Horizontal correlation detects bots by observing similarities in host behavior and/or communication. BotMiner [48] clusters similar communication and malicious activity patterns and then performs cross-cluster correlation to list bots in the monitored network. BotSniffer [61] uses spatial-temporal similarities between bot families to detect bots. Several methods have been proposed that detect botnets by clustering flows with similar characteristics [62], [63], [64], [65]. Horizontal correlation techniques suffer from a setback; there must be more than one bot infected by the same botnet in the monitored network for successful detection.

Vertical correlation correlates activities of a single machine and compares it with a model of bot behavior. BotHunter [52] analyzes the sequence of communication exchanges between a host and the Internet. It models the infection as a loosely coupled sequence of five stages: inbound scanning, exploit usage, egg downloading, outbound bot coordination dialog, and outbound attack propagation. Suspicious outbound activity coupled with intrusion detection activity indicates a successful bot infection. BotTracer [66] uses virtual machine techniques to detect botnets based on three phases of botnet life cycle; automatic startup of bot with no user intervention, C&C establishment and attack.

Behavioral Analysis is another class of semantic detection techniques that analyzes botnets by observing deviations of machine/traffic behavior from an established normal pattern or its similarity with known botnet behavior. *Behavioral Analysis* can be *Host-based* or *Network-based*.

Host-based detection methods look for signs of bot-like behavior on a host. BotSwat [67] detects botnet infection by identifying command-response behavior. It tracks programs that use data received from unreliable network sources (tainted data) and looks for possibility of remote bot initiation. It has related a set of system calls with bot activity, which are called 'gate functions'. Botnet infection is suspected if tainted data is passed as argument(s) to gate functions.

Network-based methods use information derived from network traffic and services to detect bots. It typically entails classification of traffic into network applications and looking for bot-like behavior in individual application traffic. Classification of traffic into network applications is not trivial. Applications using dynamic and random ports have rendered port-based application classification futile. Developing payload based application signatures is defeated by the use of encrypted traffic and privacy issues. Lu et al. [68] used payload signature examination method to classify traffic and found that 40% network flows could not be classified into specific applications. A number of methods [69], [70], [71], [50] focus on detection of IRC-based botnets and hence require some means to divide traffic into IRC and non-IRC parts. P2p bots can be separated from p2p file sharing applications on the basis of flow characteristics, p2p churn, and difference between machine and human behavior [65]. Researchers have detected bots comprising FFSN's by observing DNS resource records [72], [73]. Similarly, bots can be detected by monitoring requests to rogue DNS servers known to support botnet activities. A vein of research focuses on identifying C&C communication in network traffic which can be used to reveal bots. More discussion can be found under the facet about C&C Detection.

3.2 C&C Detection

Detection of C&C channel is an important aspect of botnet detection. Identification of C&C and its subsequent analysis can help in understanding botnet behavior. This information can be leveraged to identify bots and possibly C&C servers.

3.2.1 Active Detection

Active C&C detection involves taking part in the botnet operation, for example, online manipulation of network flows to deduce information about possible C&C communication. Active C&C detection involves *Injection* or *Suppression*.

Injection: Injection entails injecting packets into suspicious network flows. The similarity of the reply to the injected packets with typical bot response indicates that the flow might be part of C&C communication. Injection can be performed either by infering the botnet C&C protocol, or by blindly replaying incoming packets in the suspicious flow with or without minor changes. The latter route can be taken for botnets using stateless C&C protocol. It will fail for botnets that are secured against replay attacks of this form by using timestamps or sequence numbers. C&C detection based on protocol inference requires reverse engineering the botnet C&C protocol. This allows initiation of an informed C&C dialog for forensic purposes. C&C detection based on protocol inference can be automated by feeding the detector with information about protocols of known botnets. Using this knowledge, packets can be injected into suspicious flows to compare the response with known botnet response [20]. BotProbe is a tool based on active injection techniques to identify chat-like botnet C&C communication [74] . BotProbe leverages two basic differences between human and bot responses. Unlike humans, bots respond deterministically to the same command and are intolerant to typographic mistakes.

Suppression: In suppression, incoming/outgoing packets in suspicious network flows are suppressed to elicit known response from any of the ends of the C&C communication. For example, consider a suspected bot that requests C&C server for some update and the corresponding response is dropped. After some retries, the bot will activate its back-up mechanism. If the events that are fired up as part of the back-up mechanism are already known, botnet infection can be confirmed. SQUEEZE [75] triggers C&C failover strategies (backup C&C servers and use of domain generation algorithm) by progressively blocking connections to C&C servers that a bot tries to contact.

3.2.2 Passive Detection

Active C&C detection is complemented by passive techniques. Active C&C detection receives more scrutiny and criticism because of policy restraints and greater penalty in case of false positives. This explains the prevalence of passive C&C detection methods. It involves silently observing network traffic, looking for cues of C&C communication. These mechanisms can be broadly categorized as *Syntactic* and *Semantic*.

Syntactic: Syntactic C&C detection works by developing signature-based models of C&C traffic. The signatures are obtained by observing frequently occurring strings or token sequences in malicious traffic. Manual signature development is less reliable and time-consuming. Methods for automated signature/model generation have been proposed recently [76], [49], [77].

Semantic: Semantic C&C detection approaches use some heuristic to associate certain behavior with C&C traffic. These can be further divided into *Statistical*, *Correlation* and *Behavior-based* approaches.

Statistical Approaches can be used to detect botnet C&C communication. Machine learning has been extensively used for network traffic classification [78]. However, its efficacy in detecting C&C has not been explored much. Machine learning, particularly supervised learning, has been used for C&C detection. It involves identification of features, such as range of packet lengths, inter-packet arrival times and flow duration. Using these features, a classifier is trained on relevant datasets. Subsequently, it develops rules which are fed to the Machine Learning algorithm for classifying network traffic as benign and C&C. Machine learning algorithms have been used to first classify network traffic into IRC and non-IRC traffic and then identified botnet and non-botnet traffic within the IRC traffic [79]. In another approach, graph-based models are used to represent malicious C&C connections [80]. The graph model for each network connection is based on the system calls that lead to this connection and the system calls that operate on data that is returned. Machine learning techniques are then used to automatically generate graph templates for C&C activity by training the classifier on a set of graphs that are associated with known C&C.

The main idea behind *Correlation* based methods is that similar communication patterns in network traffic can point to C&C traffic. This implies that correlation must be performed at a higher network element, such as a router, where network traffic from multiple hosts is visible and can be analyzed for similarity. Strayer et al. [64] used a method to first identify botnet-like traffic and then find C&C traffic by clustering flows with similar characteristics such as bandwidth, packet timing, and burst duration. Gu et al. [48] combined this approach with activity-based correlation of hosts and performed cross-cluster correlation to detect botnets.

In *Behavior-based* detection methods, C&C traffic is identified by observing its deviation from normal traffic or its similarity with established behavioral model of C&C traffic. In behavior-based detection methods, the scope of analysis is typically restricted to individual hosts (no comparison is made with behavior of other hosts). Divergence of a flow from typical network usage for a user can qualify it as possible C&C flow. For example, a connection made to Russia at midnight from a user's machine who does not use the network after evening is cause for concern. Regardless of which C&C protocol is being followed, C&C communication has some behavioral characteristics that can give it away. Wurzinger et al. [49] presented a system to automatically generate C&C models from botnet samples run in a controlled environment. These models were generated by attributing response behavior, such as sending spam emails and carrying out DoS attacks, to previously-issued commands. The command portion of the model was signature-based. However, response was detected by observing anomalies in behavior. These command-response models were then used to detect C&C and subsequently bots.

3.3 Botmaster Detection

Not many botnet detection techniques target the botmaster. Botmaster detection can have serious ramifications for a botnet. On the one hand, it can lead to legal prosecution and hefty fines for the botmaster. On the other hand, it can lead to disbandment of the entire botnet based on information provided by the botmaster. For the reasons cited above, botmaster is the most protected part of a botnet and hence difficult to detect. The botmaster issues only a few commands to the C&C server to be relayed to the bots, thus generating little traffic that might also be encrypted. Detection of the botmaster is further complicated by the presence of stepping-stones between the botmaster and the C&C server.

3.3.1 Active

Active botmaster detection involves manipulation of botnet activity. There are very few mechanisms that actively detect the botmaster. Active botmaster detection techniqes revolve around *Marking*.

Marking techniques have been used extensively to traceback culprits responsible for malicious activities over the Internet. There are various flavors of packet marking such as probabilistic packet marking [81], [82], [83], ICMP traceback [83], [84] and deterministic packet marking [85], [86]. In marking schemes, some information is written into packets by either the victim machine or intermediate routers to help locate the attacker. Ramsbrock et al. [87] presented a mechanism for live traceback of botmaster by injecting watermark in response packets to the botmaster from a rogue bot under the defender's control.

3.3.2 Passive

Passive detection of botmaster involves analysis of network traffic and other data without manipulating or modifying botnet operation. Passive mechanisms for botmaster detection involve *Logging* and *Stepping-stone Detection*.

In logging mechanisms, routers log information about packets passing through them. This information is used to verify whether or not suspected packets were forwarded by specific routers. Logging mechanisms incur heavy computational complexity in addition to scalability issues. Source Path Isolation Engine (SPIE) is a hash-based IP traceback mechanism [88]. SPIE used deterministic logging mechanism to reconstruct path to the attacker. Logging mechanisms have not been used for botmaster detection so far, mainly because routers on a packet's path cannot be dictated to support and maintain additional logs.

Another way to detect botmaster is through stepping-stone detection. The botmaster hides its identity behind one or more stepping-stones. While stepping-stone detection does not directly detect the botmaster, in certain cases it can be used recursively to identify the botmaster. There are two main difficulties in the detection of stepping-stone connections. Packets from the botmaster may arrive to the C&C server with random delays between them. The delay can be caused by network factors or the botmaster can intentionally introduce them to evade detection. The botmaster may also add additional packets, called chaff, to further confuse the detection process.

All existing stepping-stone detection methods work on the basis of correlation between connection content, host activity or packet timing. The methods based on host activity correlate user login information from different hosts part of the stepping-stone chain. As the stepping-stones are under the botmaster's control, detection can be easily evaded by manipulating and forging information on these hosts. Content-based detection methods such as *thumbprinting* [89] detect connections belonging to the same chain by observing similarity in their contents. Because these methods are based on content inspection, they are effective for unencrypted traffic only. The predominance of encryption for obscuring C&C traffic has rendered content-based detection futile.

Another branch of approaches for stepping-stone detection leverages information regarding packet arrival time. Timing and chaff perturbation are great challenges for the effectiveness of these methods. Both timing and chaff perturbation are traffic mainpulation techniques. Timing perturbation refers to random delays between packets while chaff refers to insertion of meaningless packets to frustrate analysis. Characteristics of interactive traffic such as packet size and timing have been used to detect encrypted stepping-stone connection [90]. Wang et al. [91] correlated inter-packet timing characteristics of both encrypted and unencrypted connections to detect stepping-stone connections. Donoho et al. [92] used wavelets and multiscale methods to separate short-term behavior of stepping-stone connections (delay and chaff perturbations) from their long-term behavior (which can be correlated). Blum [93] used techniques from computational learning theory and the analysis of random walks to detect and identify encrypted stepping-stone connections with polynomial upper bounds on the number of packets required for the analysis. Zhang et al. [94] proposed techniques to detect encrypted stepping-stone connections. Their method was agnostic to delay and chaff perturbations. Other timing based methods include [95], [96], [97].

The previously discussed techniques passively detect stepping-stones, while very few techniques actively detect stepping-stones [98], [95]. Wang et al. [98] coined the sleepy watermarking method to actively detect unencrypted stepping-stone connections. Sleepy Watermark Tracing (SWT) activates or 'wakes up' when an intrusion is detected. In such an event, it injects a watermark into backward connection of the intrusion and collaborates with intermediate routers to reveal all the hosts in the stepping-stone chain.

3.4 Discussion on the Taxonomy

In the preceding discussion, we broadly classified botnet detection mechanisms as 'active' and 'passive'. This classification was along the dimension *Level of Activity*. In this section, we identify a number of other dimensions that could be used to classify botnet detection mechanisms. We highlight classification based on the remaining dimensions as new venues for research. Furthermore, we investigate the effect of different botnet features on the accuracy of botnet detection approaches.

3.4.1 Dimensions of Botnet Detection

Depending on the interest of the reader, each facet of botnet detection can be further explored in the light of any of the applicable botnet detection dimensions as described in Fig. 5. For example, a security researcher interested in estimating footprint of different botnets would want to explore detection techniques with the dimension *Discernment*. On the other hand, researchers developing a botnet detection tool to be deployed at a large ISP would be interested

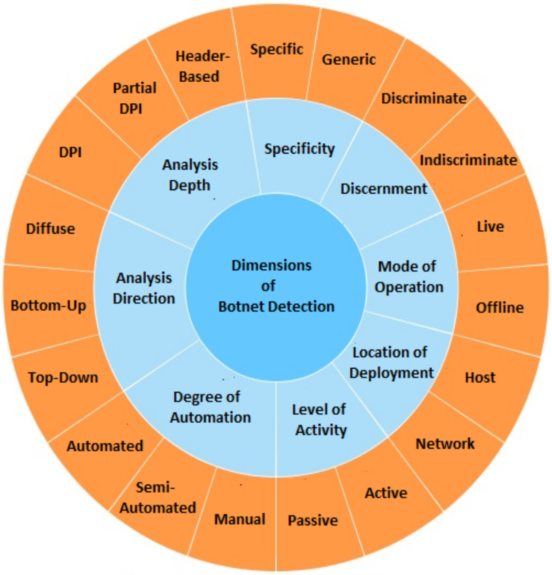

Fig. 5. Dimensions of Botnet Detection.

in the dimension *Analysis Depth*. We call these different classification criteria 'dimensions'.

Some dimensions may not be applicable to certain facets. For example, *Analysis Direction* and *Discernment* are more relevant to *Bot Detection*. *Specificity* is applicable to *Bot Detection* and *C&C Detection*. The remaining dimensions can be used with any of the three facets of botnet detection; i.e., *Bot Detection*, *C&C Detection* and *Botmaster Detection*.

Degree of Automation: Depending on the degree of human participation in the detection process, we can classify botnet detection methods as manual, semi-automated and automated.

Manual approaches that require significant human effort to detect botnets fall under this category. Typically, such techniques require manual acquisition and reverse engineering of bot binary for developing signatures that are fed into custom-made botnet detection software. Considerable human effort is required to reflect even the slightest change in botnet functionality. Many methods [99], [14] for p2p-based botnet detection are manual.

Semi-Automated methods for botnet detection require very little human intervention and most of the detection is performed in automated fashion. Typically, human involvement is required only when something changes in an existing botnet or new kinds of botnet have to be catered for in the detection engine. For example, [50], [52] heavily rely on manually developed signatures for botnet detection. If the botnet C&C style changes, or a new botnet appears, corresponding signatures have to be manually developed and incorporated in the detection framework. This is in contrast with recent work that automatically infers botnet C&C protocol [20] which may be used for signature development.

An *Automated* system should require no human intervention after initial development. Ideally, any detection method should be as generic and automated as possible. Methods that rely on behavioral characteristics of botnets such as [64] or automatic generation of detection models [49] have the potential to operate in a fully automated manner.

Analysis Direction: Botnet detection can be carried out in several directions, where direction means the sequence of analysis. Some detection methods start at the bots and perform correlation at an upstream network component, others take the opposite route and observe anomalies at upstream network components and trace the effect back to the cause.

Top-Down approaches observe phenomena at upstream network components such as spam [53], [100], [101] and anomalies in DDNS [55] and DNS traffic [56]. Clustering and correlation techniques are then applied to identify bots belonging to specific botnets.

In *Bottom-Up* approaches, behavior exhibited by individual bots is analyzed at an upstream network component to identify bot families based on similarity of behavior. BotMiner [48] detects bot families by clustering machines that display similar communication and activity patterns.

Diffuse methods of detection are more relevant to p2p botnets where there is no hierarchy in the botnet components. The detection process can start at any bot and other bots can be discovered by analyzing communication patterns or peer-lists.

Analysis Depth: Some detection methods base their analysis on compact and easily accessible data, thus involving minimum overhead. Other techniques perform more in-depth analysis of data. This factor plays a significant role in the effectiveness of network-based, real-time detection algorithms.

Deep Packet Inspection (DPI) based detection methods that perform fine-grained analysis of data. Deep Packet Inspection (DPI) has been used for signature-based detection of malicious payloads [52] or communication [61], [50] and to compute content-similarity for packet payloads [65]. DPI-based methods involve significant computational and operational complexity, particularly if large amount of network traffic is involved.

Partial DPI based methods perform DPI only for suspicious data instead of analyzing all data indiscriminately. Zhang et al. [102] proposed a sampling-based approach that identified bot flows which were then forwarded to fine-grained botnet detectors.

Header-Based methods operate on header/flow level data and incur minimum complexity, making them ideal for being deployed over large networks. Karasaridis et al. [70] compared flow records of suspicious hosts with IRC traffic models to identify botnet controller activity.

Specificity: Some detection methods are tailor-made to detect certain kinds of botnets. Other methods perform botnet detection on the basis of general botnet characteristics which do not vary among different instances. Specificity is the measure of a detection method's dependence on instance-specific features.

Specific detection methods are custom-made to detect certain kind of botnets. For example, Rishi [50] can only detect IRC-based botnets. These approaches fail to detect botnets using a different structure than the one targeted.

Generic approaches consider general characteristics of botnets instead of targeting specific botnet instances [64], [48]. Resultantly, the results are more promising and the scope of detection is broader.

Discernment: Discernment is the ability of a detection method to differentiate between different bot families. While a home user is less likely to be concerned about details of the other members of the bot family that infected his/her machine, this information is valuable to researchers who try to estimate the footprint of different botnets.

Discriminate methods for botnet detection not only identify the infected machine, but can also provide information about bot family. This typically requires an elevated view of the network. Bot families are determined by clustering hosts with similar communication or behavioral characteristics [48]. Host-based detection techniques can also identify bot family which the bot in question serves. Signature-based detection can possibly name the bot family to which the bot belongs. Use of system calls and library routines associated with known botnets can also reveal which bot family the bot is associated with.

Indiscriminate methods can detect botnet-infected machines but cannot make distinction between different bot families. A good example of this is BotHunter [52] which can discover infected machines in the monitored network but gives no information about bot families.

Mode of Operation: Mode of operation refers to the suitability of the detection method to operate in a live or offline environment.

Live methods can detect botnets in real-time while the monitored host or network carries out its normal operations. For example, Ramsbrock et al. [87] use watermarking for live detection of botmaster. The ability of a detection method to operate in live mode is closely related to how much it relies on DPI. A network based detection method performing full DPI is bound to falter in live environment with potentially huge and unpredictable traffic.

Offline methods for botnet detection provide promising results when run on log files or network traffic dumps. These methods are particularly helpful for performing network forensics or research purposes. BotHunter [52] can operate in both live and offline setting. In offline mode, it can operate on pcap file as well as Snort [51] logs. Some security companies [103], [104] offer services to investigate botnet-related incidents by analyzing logs and other evidence.

Location of Deployment: Based on location of deployment, botnet detection methods can be classified as host-based and network-based. These are not mutually exclusive and a distributed method can make use of both of them.

Host-Based methods that analyze host behavior or data to detect potential botnet infection. Obviously, these methods can only declare botnet infection on individual machines and give no information about other bots belonging to the same bot family. Both BotSwat [67] and BotTracer [66] are examples of botnet detection systems deployable on hosts only.

Network-Based methods analyze network traffic and depending on the scope of analysis, can be deployed anywhere in the network hierarchy. Possible deployment location can be proxy server, ISP and so forth. Karasaridis et al. [70] proposed a method to detect botnets on a large Tier-1 ISP network.

Degree of Activity: By degree of activity, we mean the extent to which the detection method interferes in the botnet operation. Some detection methods quietly observe ongoing botnet activity and base their decision on this information alone. Other detection techniques involve active participation in botnet C&C or infiltration.

In *Active* detection, the strategy is to take part in the botnet operation by impersonating as a component of the botnet instead of passively observing its activities. Active C&C detection involves online manipulation of network flows to deduce information about possible C&C communication.

Passive detection approaches detect botnets by silently observing and analyzing botnet activities without making a conscious effort to participate in the proceedings.

3.4.2 Effect of Botnet Features on Botnet Detection

Different botnet features positively or negatively affect the accuracy of botnet detection mechanisms. This is very important information from the point of view of a security researcher. Armed with knowledge of pros and cons of individual detection mechanisms, complementary approaches can be paired to achieve a synergistic effect. Additionally, informed decisions can be made in the choice of detection mechanisms for designing tailored strategies targeted at detection of specific threats. To this end, we present three tables,Table 1, 2 and 3, that investigate the effect of botnet features on the three facets, *Bot Detection*, *C&C detection* and *Botmaster Detection* of botnet detection mechanisms, respectively. We do not aim to explain each row in the tables. Instead, we highlight some interesting observations.

Binary Obfuscation and *Encryption* degrade the accuracy of detection approaches that rely on reverse engineering of bot binary or C&C. *Syntactic* detection approaches leverage on known malicious patterns in host or network data. *Infiltration* and *Injection* require understanding of the bot binary and/or C&C protocol. For *C&C Server Hijack*, the IP addresses of C&C servers for subsequent hijack can be retrieved by analyzing bot binary in which these can be possibly hardcoded. Another way for discovering such IP addresses is to monitor C&C communication. *Binary Obfuscation* complicates inference of the bot binary while *Encryption* renders C&C indecipherable.

The choice of botnet topology makes some detection approaches more effective. *C&C Server Hijack* is more effective for botnets utilizing *Centralized* structure. Taking down the central C&C server(s) incapacitates the entire botnet. *Horizontal Correlation* associates group activity and communication patterns with botnet behavior. Similarity in communication patterns is particularly evident in the case of *Centralized* botnets. *C&C Server Hijack* is not an effective detection approach for *Decentralized* botnets as any node can be used to float commands. At best, hijacking a supernode can reveal a small portion of the entire botnet.

All bots will ultimately carry out the *Purpose* for which they were employed by the botmaster. The same, particularly if carried out in an aggressive manner, can hint at botnet infection. *Correlation* based on host activity is more effective for botnets with aggressive *Purpose*. For example, botnets targeted at *Unsolicited Marketing* and *Network Service Disruption* are more likely to generate traffic that would be feasible to be detected by network traffic analysis than botnets focused on *Information Gathering*.

Detection approaches that rely on botnet command-response behavior or observe similarity in communication patterns and flow characteristics (e.g. packet size, inter-packet arrival times and flow duration) degrade if a

TABLE 1
An overview of how different botnet features affect bot detection mechanisms.

Bot Detection Mechanism	**Related Botnet Features**	
	Improve	**Degrade**
Infiltration		Encryption, Binary Obfuscation.
C&C Server Hijack	Centralized Topology, Rogue DNS Server.	Decentralized Topology, Bot Binary Obfuscation, Anonymization, IP Flux, Information Hiding.
Syntactic		Bot Binary Obfuscation, Information Hiding, Encryption.
Horizontal Correlation	Centralized topology, Purpose.	Traffic Manipulation, Information Hiding, Decentralized topology.
Vertical Correlation	Purpose, Propagation, Bot Evasion, Rogue DNS Server.	C&C Evasion, Binary Obfuscation.
Host-based		Traffic Manipulation.
Network-based	C&C, IP Flux, Rogue DNS Server.	Protocol Manipulation, Traffic Manipulation, Information Hiding.

botnet makes use of *Traffic Manipulation*. This includes some *Host-based* detection approaches, *Network-based* methods, in particular those that rely on traffic classification, and *Suppression* in case of C&C detection. Use of *Marking* for botmaster detection experiences loss in accuracy if *Traffic Manipulation*, especially timing perturbations and traffic padding, is employed for C&C evasion.

Protocol Manipulation affects detection techniques that are based on observation of traffic anomalies or possible C&C communication. *Information Hiding* techniques make it difficult to detect botnets by analyzing network traffic for the presence of C&C communication. *Behavioral Analysis*, in particular *Network-based* methods that rely on traffic classification, degrade considerably because of *Protocol Manipulation*.

IP addresses of C&C servers can be discovered by monitoring C&C communication. *Anonymization* and *IP Flux* make it difficult to trace C&C communication to C&C servers. *Logging* mechanisms use information stored by routers to trace attack packets back to the source. Logging mechanisms do not help much in the presence of *Anonymization* which hides the source of a network flow by bouncing packets through a network of volunteer servers before delivering them to the destination.

The choice of protocol for C&C communication is closely related to the effectiveness of some detection approaches. C&C based on *Neoteric* and unpopular protocols can be more conveniently detected than C&C that utilizes popular, *Existing* network protocols such as HTTP. Statistical approaches, particularly Machine Learning, have been used to classify network traffic and subsequently detect C&C by observing anomalies in individual application traffic. Their accuracy degrades if C&C com-

TABLE 2
An overview of how different botnet features affect C&C detection mechanisms.

C&C Detection Mechanism	**Related Botnet Features**	
	Improve	**Degrade**
Injection		Encryption.
Suppression		Traffic Manipulation.
Syntactic		Encryption.
Correlation	Centralized Topology.	Traffic Manipulation, Information Hiding, Protocol Manipulation.
Behavior-based	Rogue DNS.	Traffic Manipulation, Information Hiding.
Statistical Approaches	Neoteric Protocol.	Existing Protocol, Traffic Manipulation, Information Hiding, Protocol Manipulation, Encryption.

TABLE 3
An overview of how different botnet features affect botmaster detection mechanisms.

Botmaster Detection Mechanism	**Related Botnet Features**	
	Improve	**Degrade**
Marking		Traffic Manipulation.
Logging		Anonymization.
Stepping-stone Detection	Stepping-stone.	C&C Evasion.

munication employs popular *Existing* protocols. Additionally, the task of machine learning algorithms is made significantly difficult by the use of *Traffic Manipulation*, *Information Hiding*, *Protocol Manipulation* and *Encryption*. Use of *Existing Protocol* for C&C further complicates telling benign traffic and C&C apart. The aforementioned features have a similar effect on *Stepping-stone detection*.

Botnets employ *Rogue DNS Servers* to serve phishing content and to avoid blacklisting services. *Behavior-based* detection techniques reinforce confidence about botnet C&C by looking for suspicious behavior, such as the use of *Rogue DNS Server* for resolving domain names instead of the host's default DNS server. In the context of *IP Flux*, detection of FFSN by monitoring DNS responses to suspicious domain names offers two benefits. It can help in detecting the compromised machines being used as proxies to host malicious botnet services. Secondly, DNS queries to web services hosted by FFSN can indicate that the querying machine is also compromised. Moreover, DNS queries directed to known *Rogue DNS Servers* can reveal bots.

Some detection methods such as *Vertical Correlation* correlate events in a temporal fashion. Detection of malware *Propagation* and unusual behavior such as *Security Suppression*, *Anti-*

analysis, *Rootkit* installation and use of *Rogue DNS Server* hint at botnet infection. Observing *Purpose* of a botnet, in particular its attack behavior, adds to confidence of the detection approach. *C&C Evasion* techniques degrade vertical correlation if potential C&C traffic is one of the inputs to the correlation engine.

These tables can be best utilized by identifying complementary detection approaches that compensate each other's shortcomings and highlight their respective strengths. Development of a comprehensive strategy along these lines will help in generic and more effective botnet detection.

3.5 Summary: Taxonomy of botnet detection mechanisms

In the event that an intrusion attempt succeeds, the very least a robust security plan should address is its detection. With this objective in mind, in this section, we provided an extensive overview of botnet detection mechanisms. To highlight the component of botnet infrastructure that a detection technique is targeting, we classify botnet detection mechanisms as those concerning *bots*, *C&C communication* and *botmaster*.

Bot detection refers to detection of botnet compromised machines without regard to the larger superset of botnet population of which the compromised host is a member. Based on the degree of participation of the defender in botnet operation, bot detection strategies can be further classified as *active* and *passive*. *Active* bot detection requires the defender to impersonate as a component of the botnet. Subclasses include botnet *infiltration* and *C&C server hijack*. *Passive* detection approaches detect botnets by silently observing and analyzing their activities without making a conscious effort to participate in the proceedings. This analysis can leverage comparison with predetermined patterns of botnet infection derived from observed samples (*syntactic* analysis) or the context of events and protocol information (*semantic* analysis). The latter can further entail *correlation* and *behavioral analysis*.

A number of botnet detection mechanisms focus on *detection of C&C channel*. *Active detection* involves taking part in the botnet operation, for example, manipulation of network flows to deduce information about possible C&C communication. *Suppression* and *Injection* represent subclasses of active mode of C&C detection. *Passive detection* involves silently observing network traffic, looking for cues of C&C communication. This may entail comparison with predetermined C&C signatures (*syntactic* analysis) or behavioral characteristics of C&C traffic (*semantic* analysis). The latter can be further broken down into *statistical* and *behavior-based* approaches.

Some detection mechanisms target the botmaster. *Active detection* involves manipulation of botnet activity, for example, insertion of information into packets either by victim machine or intermediate routers to help locate the attacker (*marking* techniques). *Passive detection* involves analysis of network traffic without manipulation or modification of botnet operation. *Logging* and *stepping-stone detection* represent subclasses of passive botmaster detection.

We argued that the high-level classification of botnet detection mechanisms, i.e., bot, C&C and botmaster detection (referred to as facets of botnet detection) can be further categorized with different dimensions depending on interest of the reader. In our taxonomy, we classified all the facets of botnet detection with the dimension *level of activity*, i.e., *active* and *passive* detection. We identified a number of other dimensions for classification of botnet detection techniques; *degree of automation*, *analysis direction*, *analysis depth*, *specificity*, *discernment*, *mode of operation* and *location of deployment*. We left detailed discussion as an avenue for future research.

We also analyzed how the absence or presence of different botnet behavioral features affect the accuracy of botnet detection mechanisms. In this regard, we presented three tables that investigate the effect of botnet behavioral features on the three facets (bot detection, C&C detection and botmaster detection) of botnet detection mechanisms. This information can be utilized to devise a comprehensive detection strategy that combines complementary detection approaches.

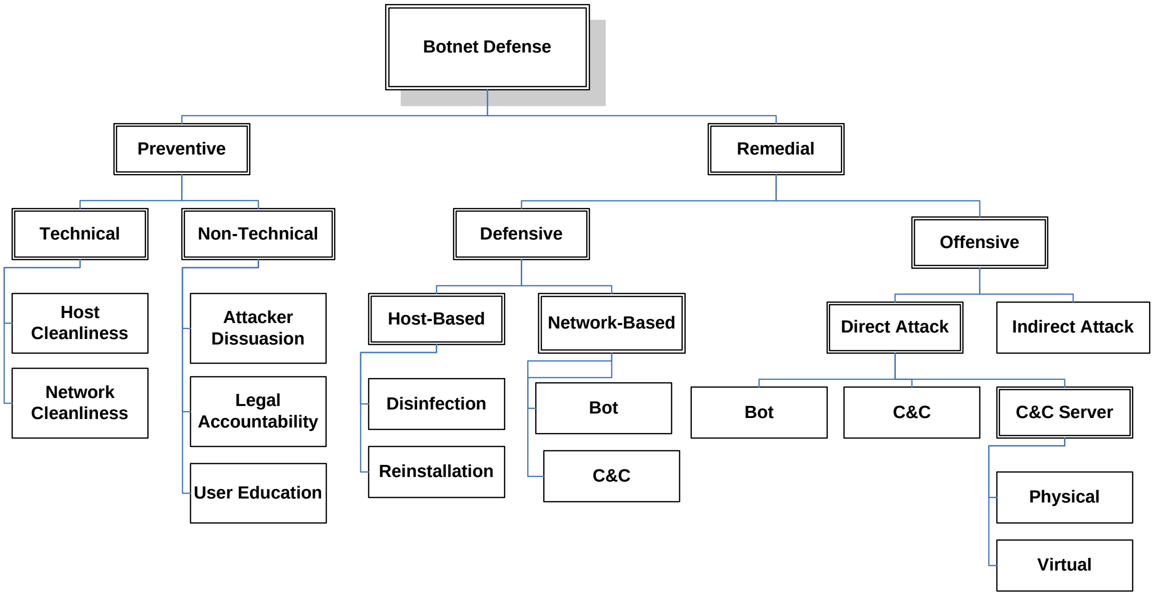

Fig. 6. Taxonomy of Botnet Defense Mechanisms.

4 Taxonomy of Botnet Defense Mechanisms

At present, defense against botnets is mostly preventive or defensive. Preventive defense includes proactive measures to avoid botnet infection. Defensive methods are reactive in nature and concern themselves with cleaning systems once they have been infected. There is an aching need for developing defense strategies that solve the problem at its root. While bot disinfection is important from user point-of-view, it does not hurt the botnet which finds other machines to infect and serve its purpose. Ideally, a defense method should incapacitate the whole botnet or significantly damage it.

We broadly categorize botnet defense mechanisms as *Preventive* and *Remedial*. The rest of this section further classifies these high level classes of botnet defense mechanisms.

4.1 Preventive

Hosts and networks can adopt preventive measures against botnets to raise the bar for possible botnet infection. These methods are effective before the botnet infection has taken place.

4.1.1 Technical

Technical approaches include botnet defense activities and measures that are related to computers and networks.

Host Cleanliness: Botnets spread by exploiting operating system or application vulnerabilities on victim machine or by social engineering. Measures for host cleanliness should be adopted to proactively defend against botnet infection. A reasonable patch management system should be in place to install the latest security patches for operating systems and applications. Operating systems and most applications now come with the auto-update feature to relieve users of the responsibility of manual updation. Infection can be warded off by avoiding opening emails from unknown sources, particularly if the email contains attachments with executable files or scripts. Similarly, one should exercise restraint in clicking links on untrusted websites. Security settings of web browsers should be adjusted optimally and automatic execution of scripts such as JavaScript, ActiveX and VBScript should be turned off. Programs and users should be given only basic permissions in accordance with the principle of least privilege and administrative rights should be exercised with discretion. Installation of security software such as anti-virus, anti-spyware, anti-trojan, rootkit detection packages and firewall can help fend off some infections. The former can be complemented by Host Intrusion Prevention System to protect from previously unknown threats by identifying anomalous behavior. Closing ports used by applications favored by botnets for C&C communication such as IRC and FTP can reduce the risk of botnet infection to some extent. We realize that the measures described above compromise user experience and are generally short-circuited by them. However, to date these remain the most effective and preventive mechanism and thus the focus should remain on informing and motivating users of their importance while reducing their hit on usability.

Network Cleanliness: While network administrators can benefit from the previous section to defend individual hosts in the network, there are some network level preventive steps that can further reduce the risk of botnet infection. The principle of separation of privilege dictates that multiple levels of security make it non-trivial for the attacker to circumvent it. Same is true of networks. In addition to host-based protection, servers, external connections, email gateways should all have optimum security in place. Network Anomaly Detection System and firewall are good options for proactive defense against the botnet threat. Access to known malicious domains should be blocked. A list of such domains (malicious/C&C/RBN domains) can be obtained from the Internet. Network traffic should be made to pass through a web proxy where it is possible to scan incoming content for malware presence and outbound content can be scrutinized for possible data leaks. Placing honeypot and/or darknet in the network can indicate vulnerability of the network to certain threats and can point to the presence of infected machines in the monitored network. Finally, a comprehensive network security policy should be developed and enforced.

4.1.2 Non-Technical

These approaches fall under domains which are not directly related to the digital world of computers.

Attacker Dissuasion: It has been proved time and again that there is increasing financial motivation behind botnet operation. Several studies [105], [106] suggest the lucrativeness of the business of spam, clickfrauds and other malicious activities carried out by botnets. Targeting the business model employed by botnets can reduce incentives for the bot master to run the botnet. MARK (the Multihost Adware Revenue Killer) was a distributed network of machines capable of controlling advertising impression numbers, click through rates and software package installs, to carry out effective attacks against malicious-code generated revenue streams [107]. The motivation was to make the botmaster give up maintenance of the botnet by reducing advertising revenue generated by adware and botnets. A number of anti-spam approaches have been tendered [108], [109], [110]. Detecting and subsequently filtering spam email before it reaches the intended recipients can hurt the profitability of this model of marketing to advertisers. In turn, the botmasters will lose business and give up the botnet. Current defense mechanisms against DDoS are not very effective [111]. Better preventive strategies against DDoS that stop the attack before it affects the targeted service will reduce the utility of the botnet to the botmaster.

Legal Accountability: It is important to complement existing defense mechanisms against botnets with a solid legal framework. As long as the botmaster enjoys relative impunity over the Internet, technical detection and mitigation efforts alone will not suffice. Microsoft's takedown of Rustock botnet [112] reinforces the frequently voiced concern of security organizations for more comprehensive legislature to tackle the botnet phenomenon [113]. USA, EU countries, China and Japan have strict penalties for cyber crimes [114]. Despite this, USA, Germany and France were among the top three countries hosting C&C servers in a survey conducted by security firm Damballa [115]. C&C servers tend to be scattered around the globe in countries with different legal attitude towards cyber crime, therefore taking down a server in one country will not decapitate the whole botnet. Botnets are a distributed phenomenon and therefore there is a strong need for collaborative legislation and cross-border cooperation to fight botnets. The legal consequences of cybercrime can potentially deter botmasters from operating botnets.

User Education: All the technical defenses will ultimately falter if the user has no motivation to follow them. Generally, users are not concerned about the security of their computers as long as they can perform their routine activities without hindrance. People are often the weakest link in an otherwise secure network. Botnets exploit this inherent weakness to infect systems by use of social engineering tactics. Some countries legally bind users to act responsibly by imposing privacy standards and fines for negligent or willful non-compliances [116]. Users should be educated about the gravity of the botnet threat and the measures they can take to avoid becoming bots.

4.2 Remedial

Remedial defense strategies help in partial or complete recovery from botnet infection. These mechanisms try to solve the problem either by removing the infection from infected machines and networks (defensive) or damaging the botnet infrastructure in a way that it either stops or significantly complicates perpetration of malicious activities (offensive).

4.2.1 Defensive Strategies

This class of methods is aimed at self-recovery in the event of botnet infection. Defensive strategies can be further classified as *Host-based* and *Network-based*.

Host-based: Host-based techniques help individual machines recover from botnet infection. Mitigation techniques that target C&C servers do nothing for the infected bots. Bots will continue to experience negative effects of the infection and can even unwittingly participate in a never-ending cycle of malicious activities if they fail to receive the stop command from the exanimate C&C server for a previously issued start command. Host-based defense seeks to restore individual bots to their clean state. Host-based defensive strategies can involve *Disinfection* or *Reinstallation*.

One of the defensive methods for recovering machines from botnet infection is *Disinfection*. There are off-the-shelf programs dedicated to disinfecting compromised systems, however their effectiveness in completely eliminating the botnet infection is uncertain. Determining all the services, files and registry keys that have been changed or installed does not guarantee total disinfection particularly if rootkit installation was involved. Master Boot Record (MBR) infections are notoriously difficult to get rid of and disinfection efforts may prove futile. In most cases, OS reinstallation is required.

Another option to tackle the problem of botnet infection is *Reinstallation*. In majority of cases, botnet infection cannot be entirely removed and users need to restore the operating system to an uncorrupted previous state. If a clean image is not available, complete reinstallation might be required.

Network-based: Network-based techniques aim to secure networks once it is known that one or more machines in the network are infected with botnet. These techniques can be categorized into those that *Block Bots* or *Block C&C Communication*.

Botnet-infected networks can be secured by blocking *Bots*. To contain the infection, the administrator can quarantine the infected machines in the administered network and fix them before bringing them back on the network. This is similar to the notion of walled-garden. Walled-garden is a state of isolation in which an ISP can place machines showing symptoms of botnet infection to reduce further damage to other machines inside or outside the network. The infected machine is denied all network communication except with a white-list of domain names helpful for remediation of the infection. The bot machine is not brought back on until its cleanliness is confirmed as per the ISP policy. The Messaging Anti-Abuse Working Group (MAAWG) described the concept of walled-garden as a three step process comprised of detection, notification and remediation [117].

Network-based defense strategies also involve blocking *C&C* communication. A network can be protected by identifying C&C communication and then blocking it. The task can be done in a pro-active fashion by making use of up-to-date URL and IP blacklists. However, identification of C&C traffic within the network traffic is not a trivial task. In the simplest form, traffic to and from known malicious ports should be blocked. For finer results, traffic analysis might be required. Alternately, all network communication can be blocked except a whitelist of known 'good' domains till the infected machines are cleaned.

4.2.2 Offensive

This class encompasses aggressive defense mechanisms that intimidate or completely paralyze a botnet. This can be done by carrying out an *Indirect* or a *Direct Attack* on botnets.

Indirect Attack: Indirect attack reduces usability of the botnet to the botmaster. Unlike malware of past whose purpose was to create mayhem on the Internet, there are greater financial incentives for botnets in carrying out their malicious activities. Attacking the underlying business model to significantly reduce profits has the potential to daunt the bot master to an extent it gives up maintenance of the botnet. Some botnets steal sensitive information, such as passwords and banking credentials. The stolen information is transferred from the victim's computer to a dropzone and ultimately sold to third parties. The botmaster's credibility can be hurt if fake information is injected into the dropzone. This can be done by reverse engineering the botnet's C&C protocol. This will damage the botmaster's reputation and affect his business prospectives. Ormerod et al. [118] suggested injecting fake but tracable credentials into the botnet dropzone so that the misfeasors could be identified and prosecuted.

Direct Attack: Direct attack entails mechanisms to directly hurt one or more components of the botnet. On the basis of target of the attack, this class of remedial mechanisms can involve attack on *Bots*, *C&C Communication* and *C&C Servers*.

A direct attack can be staged on *Bots*. Like other software, the bot binary and C&C protocol are likely to have bugs [119], [120]. These can be exploited by the defender to take control of bots and remotely disinfect them or trigger the C&C command related to removal of the bot from the botnet. There are legal and ethical issues involved in remotely disinfecting a user's computer without his/her consent and may be seen as privacy invasion.

Another option is to coordinate a direct attack on *C&C Communication*. The C&C communication can be poisoned by introducing bogus commands in the channel. In p2p botnets, bots search new commands by using bot command keys. Holz et al. [14] presented a way to disrupt botnet communication by injecting benign content with command keys same as those used by botnets so that the actual commands never reach the bots. Sybil attack [121] is an attack against p2p botnets that introduces into it peers under control of the defender. The sybils can be strategically positioned to gain control over a part of the p2p network. They can reroute queries to any non-bot node or a peer sybil. In this way, the sybils can disrupt a part of the overall C&C communication. Eclipse attack [122] is similar to Sybil attack except that it gains control over a much smaller portion of the p2p network. Inference of the C&C protocol enables the defender to alter and inject commands into C&C that help minimize the damage caused by botnet. The specific commands for carrying out attack such as spam and DDoS can be stopped from reaching bots and replaced by commands to remove bots from the botnet.

Direct attack can also be launched on *C&C Servers*. This involves *Physical* or *Virtual* removal of C&C servers.

Physical removal of C&C servers can potentially render the botnet dysfunctional. However, the task is easier said than done. It is applicable to centralized botnets only and requires prior knowledge of the location of the C&C servers, which itself is an arduous task. There are ethical and legal hurdles involved in taking down servers and requires cooperation from hosting providers. Most C&C servers are hosted by providers claiming bullet-proof hosting that are immune to the nature of content being hosted. This necessitates legal intervention.

It is also possible to perpetrate *Virtual* take over of C&C servers. Physical seizure is not always necessary and the desired effect can be achieved by other means. Border Gateway Protocol (BGP) blackholing is a technique to null-route traffic to and from known malicious servers. The same can be done through DNS

Sinkholing. For DNS queries for known C&C servers, the sinkhole returns a non-routable or any address except the actual address of the malicious server. Alternately, defender has the option to carry out DoS attack against known, centralized C&C server to degrade its service.

4.3 Summary: Taxonomy of botnet defense mechanisms

Botnet defense comprises mechanisms that help in prevention of botnet infection. This is not always a dependable option as botnets represent advanced threat and the intrusion may succeed despite proactive security measures. Botnet defense also covers reactive techniques such as recovery from botnet infection or offensive measures to disrupt or impair the botnet. This section presented a comprehensive taxonomy of botnet defense mechanisms. Botnet defense mechanisms can be broadly classified as *Preventive* and *Remedial*.

Preventive mechanisms discourage botnet infection proactively. These can be further classified as *technical* and *non-technical*. *Technical* mechanisms advocate hardening of computer and network security (*host cleanliness* and *network cleanliness* respectively) to ward off possible botnet infection. *Non-technical* mechanisms address botnet infection through other means, such as *attacker dissuasion*, *legal accountability* and *user education*.

Remedial defense mechanisms are reactive in nature. These can be further classified as *defensive* and *offensive*. *Defensive* strategies are concerned with recovery from botnet infection. This could involve *host-based defense measures*, such as host *disinfection* and software *reinstallation* or *network-based defensive measures*, such as *isolating bots* or *blocking C&C communication*. *Offensive* strategies comprise of aggressive defense mechanisms that aim to disrupt a botnet. This can be achieved by carrying out a *direct* or *indirect* attack on botnets. *Direct attack* entails mechanisms to directly hurt one or more components of a botnet. *Indirect attack* reduces the usability of the botnet to the botmaster so that there is little incentive to carry on its maintenance.

At present, defense against botnets is mostly preventive or defensive. Preventive defense includes proactive measures to avoid botnet infection. Defensive methods are reactive in nature and concern themselves with cleaning systems once they have been infected. There is an aching need for developing defense strategies that solve the problem at its root. While bot disinfection is important from user point-of-view, it does not hurt the botnet which finds other machines to infect and serve its purpose. Ideally, a defense method should incapacitate the whole botnet or significantly damage it.

5 Related Work

A great deal of previous work has focused on different aspects of botnets, their detection and defense. Cooke et al. [47] in their pioneer work explained C&C structures and proposed a novel structure which they called 'random'. A brief discussion of this structure can be found in section 2.6.2. A branch of research seeks to demystify certain aspects of botnet operation, such as evasion techniques [123], [124], C&C structures [5] and propagation. Nazario [4] highlighted the multifaceted nature of botnets and maintained that this should be reflected in related taxonomies. His taxonomy was based on factors such as network structure, language of bot binary, features (attacks ,server, proxy) and propagation strategies. This relates to *Purpose*, *Topology* and *Propagation* in our taxonomy of botnet features. Their work does not take into account C&C characteristics, botnet evasion techniques and rallying mechanisms. Dagon [1] emphasized that the aim of taxonomy should be identification of detection opportunities. The taxonomy classified C&C on the basis of nature of C&C resources (public/private), RFC compliance, and activity level (how often the bots contact the botmaster). The concept of protocol agnostic detection is related to secondary data *Correlation* and *Network-based* semantic bot detection in section 3.1.2. The taxonomy only highlights C&C channels and the remaining work provides a general discussion of rallying, detection and response strategies. Trendmicro and SANS published detailed reports [2], [125] explaining various botnet components. Dagon et al. [3] classified botnet structures into three models and analyzed each model from the perspective of response mechanisms.

A subset of botnet research revolves around botnet detection. Bailey et al. [126] provided a survey of existing botnet research with emphasis on botnet technology and defense. The survey examined botnet detection methods on the basis of an interesting dimension, data sources. The data sources included DNS, Netflow, packet tap, address allocation, honeypot and host data. The authors posited that this information would be a useful metric for choosing the detection method that works on the data readily available to the interested party. In our botnet detection taxonomy, we have discussed the aspect of data source in all the subsections and made clear demarcation where absolutely necessary. Classification of detection techniques on the basis of cooperative behavior, signature matching and attack behavior map to our *Correlation* and *Syntactic* detection (sections 3.1.2, 3.2.2) and secondary data *Correlation* in section 3.1.2. A similar effort classified botnet detection techniques as signature-based, anomaly-based, DNS-based and mining-based [127]. Both of these lack the depth and organization that is characteristic of a taxonomy and is a high level categorization to help readers understand existing work in this area. Zeidanloo et al. [128] presented a taxonomy of botnet detection. Honeypots have been shown as a class of botnet detection. Honeypots are functional to understanding the botnet operation and assessing vulnerability of a network to the botnet threat, yet it is important to understand that they do not identify bots or bot families per se. The other class, Intrusion Detection System (IDS), is along the lines of established taxonomy for IDS [129] with the additional notion of activity in the context of network monitoring. The taxonomy provides a good overview of the existing detection methods but falls short in distinguishing between the different targets of botnet detection methods. Researchers [130] established a broad framework for evaluating the evadability of automated bot/botnet detection mechanisms. This bears slight similarity with our overview of the effect of different botnet features on botnet detection in subsection 3.4.2. However, their work focuses on evasion mechanisms specific to six research papers related to automated botnet detection. Moreover, they do not discuss weaknesses in botnets that defenders can use to improve their detection approaches. Garcia et al. [131] laid out a topological map of network-based botnet detection methods.

A section of research concerns itself with the area of botnet defense methods. MAAWG (Messaging Anti-Abuse Working Group) [117] published a comprehensive report about how large scale bot infections can be mitigated in residential networks. A number of reports provide a detailed discussion of policies [114], detection, measurement, disinfection and defense [6] against botnets. It discusses defense mechanisms against botnets in great detail, however there is no elaborate structure in the way this information is presented. Resultantly, there is a chance that the reader will be overwhelmed and confused as the discussion becomes more involved. A number of papers [132], [133] give useful insights into botnet defense but do not span the entire gamut of possible botnet countermeasures.

A common thread to all the efforts described above is that they only partially illuminate the entire botnet phenomenon. A number of works [134], [135], [136], [137] perform a wider study of botnets, describing their behavior, detection, defense and future challenges. Our work distinguishes itself from these efforts in that we structure existing botnet literature into a generic taxonomy steering away from case-specific details. This makes our classification system relevant for most, if not all, existing and future botnets. Furthermore, the organization of our work aids understanding by making it possible to visualize complex botnet-related concepts.

6 Future Trends

In this section, we spell out some botnet trends that we expect to continue into future.

Botnets are turning to cloud computing to expand their potentials. Botnets can utilize cloud platforms in two ways; (i) host the C&C server(s) on the cloud [138], [139] or (ii) create bots on the cloud (botcloud) instead of infecting user machines [140], [141]. Hosting C&C servers on the cloud is an attractive option because of its on demand pay-as-you-go nature. Moreover, cloud-based hosting is quite inexpensive. Another trend, though still mostly research-based, is to create bots on the cloud (botclouds). This has a number of benefits. A botnet can be created almost instantly in contrast to traditional botnets where a substantial time is wasted in recruiting bots. A botcloud uses resources much more efficiently. The bot population is always online unlike traditional botnets where bots disappear because of users shutting down their computers. Moreover, full processor and bandwidth utilization can be achieved. This is not the case in traditional botnets as botnet activity has to be scaled down enough not to be detected by security software. Finally, cloud security is still in a transient stage [142] and most existing detection approaches do not scale to clouds. Thus clouds provide a nice cover to botnets for carrying out their malicious activities.

We expect to see a surge in botnets based on smartphones. While researchers [143] have already indicated the possibility of mobile phones to be used by botnets, it is in recent times that we have seen an increase in the number of such botnets [144], [145], [146]. The ubiquitous use of mobile devices make them an important frontier for botnets. Moreover, mobile phones can utilize a number of communication technologies (3G, 4G, WiFi, SMS), which multiplies the possibilities for C&C and malware propagation.

The issue of Advanced Persistent Threats (APT, section 2.4.1) is expected to exacerbate. 2012 witnessed the emergence of a multitude of sophisticated botnets geared to APT, such as Flame [147], Duqu [148], Gauss [149], Shamoon [150] and Mahdi [151]. Flame even incorporated an MD5 collision attack to spoof Microsoft's digital certificates.

The popularity and viral nature of social networks will continue to attract botnets to use them as spam targets (victim) and for malware propagation (facilitator). However, the wide adoption of social networks for C&C communication does not seem likely (section 2.3.2).

7 Conclusion

The number of users connected to the Internet almost doubled in the last five years [152]. This rate is expected to increase in view of technological advancements in the area of wireless communication. However, the remarkable growth in Internet usage is disproportionate to security knowledge of common users [153]. Botnets take cyber attacks to the next level by abusing the aforementioned discrepancy. Botnets employ a variety of mechanisms to compromise users' machines. A botnet distinguishes itself from other malware in the ability of its compromised machines to establish command and control with remote server(s) controlled by human misfeasor(s).

In this paper, we have presented three comprehensive taxonomies related to botnet behavioral features, detection and defense. Our first taxonomy aims to demystify the adversary by exploring the botnet phenomenon from different angles, such as propagation, rallying, C&C and purpose. Moreover, we have also enumerated evasion mechanisms employed by a botnet for obscuring its different parts. The second taxonomy classifies botnet detection approaches. We have introduced the notion of 'dimension' in structuring mechanisms relevant to botnet detection. This empowers a defender to evaluate botnet detection approaches by different yardsticks. Furthermore, we have shown how different botnet features listed in our first taxonomy affect the accuracy of botnet detection approaches from our second taxonomy. Our third taxonomy provides a systematic analysis of botnet defense mechanisms. Together, these three taxonomies provide a comprehensive framework that could be utilized to understand the botnet problem and its solution space. The insights gained from this characterization can be used by defenders to identify shortcomings in existing approaches for botnet detection and defense and devise improved strategies.

We conclude by identifying current trends in botnet detection and defense, followed by future work. A quarter of the world's computers were estimated to form part of a botnet half a decade ago [154]. The situation is no different today as botnets are still a primary threat to security of the Internet [155]. The area of botnet detection and defense has evolved over the last few years. In particular, legal countermeasures have succeeded in disbanding a number of botnets [112], [12], [156], [157]. However, the long-term efficacy of these methods is debatable [158]. Reactive defense approaches like removal of C&C servers yields promising results in the short-term. Botnets find a way to resurrect and resume their malicious activities. There is a pressing need to address the botnet problem closer to the source. ISPs can play a significant role in this context. This notion is reinforced by China's success in handling its spam problem by extending liability to Chinese ISPs [159]. Moreover, preventive detection and defense approaches at home computers and routers would go a long way in curtailing the botnet phenomenon. Currently, there are very few bot detection tools [160], [161]. Majority of the existing detection approaches do not incorporate defense. It is not practical to delegate defense to home users as majority of them are not technologically competent. These insights motivated us to develop a framework for botnet detection and defense [162]. An integrated strategy where high-level dismantlement of botnet C&C hubs is complemented by detection and defense on home and network computers is expected to yield promising results in tackling the botnet problem.

Acknowledgments

This work is supported by Pakistan National ICT R&D Fund.

References

[1] D. Dagon, "Botnet Detection and Response-The network is the infection," *Cooperative Association for Internet Data Analysis DNS-OARC Workshop, July*, vol. 25, 2005.

[2] T. Micro, "Taxonomy of botnet threats," *Micro*, pp. 1–15, November 2006. [Online]. Available: http://tinyurl.com/c7mlsjo

[3] D. Dagon, G. Gu, C. P. Lee, and W. Lee, "A taxonomy of botnet structures," *Twenty-Third Annual Computer Security Applications Conference ACSAC 2007*, vol. 36, pp. 325–339, 2007. [Online]. Available: http://tinyurl.com/8kxuknw

[4] J. Nazario, "Bot and botnet taxonomy," http://tinyurl.com/6bcthj, 2008, [Online; accessed 15-December-2011].

[5] G. Ollmann, "Botnet Communication Topologies," 2009. [Online]. Available: http://tinyurl.com/8kosjv3

[6] D. Plohmann, E. Gerhards-Padilla, and F. Leder, "Botnets: Detection, measurement, disinfection & defence," European Network and Information Security Agency, Tech. Rep., 2011.

[7] P. Porras, H. Sadi, V. Yegneswaran, P. Porras, H. Sadi, and V. Yegneswaran, "A multi-perspective analysis of the storm (peacomm) worm. available at: http://www.cyber-ta.org/pubs/stormworm/report," 2007.

[8] P. Wang, S. Sparks, and C. C. Zou, "An advanced hybrid peer-to-peer botnet," in *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*. Berkeley, CA, USA: USENIX Association, 2007, pp. 2–2. [Online]. Available: http://dl.acm.org/citation.cfm?id=1323128.1323130

[9] A. Dainotti, A. King, k. Claffy, F. Papale, and A. Pescapè, "Analysis of a "/0" stealth scan from a botnet," in *Proceedings of the 2012 ACM conference on Internet measurement conference*, ser. IMC '12. New York, NY, USA: ACM, 2012, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/2398776.2398778

[10] T. Chen, "Stuxnet, the real start of cyber warfare?" *IEEE Network*, vol. 24, no. 6, 2010.

[11] J. Baltazar, J. Costoya, and R. Flores, "The real face of koobface : The largest web 2 . 0 botnet explained," *Trend Micro Threat Research*, 2009. [Online]. Available: http://tinyurl.com/c95m86f

[12] T. Werner, "The inside story of the kelihos botnet takedown," http://tinyurl.com/3gzmtzd, threatpost:The Kaspersky Lab Security News Service, 2011, [Online; accessed 2-December-2011].

[13] S. Stover, D. Dittrich, J. Hernandez, and S. Dietrich, "Analysis of the storm and nugache trojans: P2P is here," in *USENIX ;login*, vol. 32, no. 6, 2007.

[14] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freilling, "Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm," in *Proceedings of the 1st Usenix Workshop on Large-Scale Exploitsand Emergent Threats (LEET*, Berkeley, CA, USA, 2008.

[15] P. Barford and V. Yegneswaran, "An Inside Look at Botnets," in *Malware Detection*, ser. Advances in Information Security, M. Christodorescu, S. Jha, D. Maughan, D. Song, and C. Wang, Eds. Boston, MA: Springer US, 2007, vol. 27, ch. 8, pp. 171–191. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-44599-1_8

[16] I. Arce, E. Levy, and E. Levy, "An analysis of the slapper worm," *IEEE Security & Privacy*, vol. 1, pp. 82–87, 2003.

[17] J. Stewart, "Phatbot trojan analysis," http://tinyurl.com/9srw4gh, 2004, [Online; accessed 15-December-2011].

[18] A. Nappa, A. Fattori, M. Balduzzi, M. Dell'Amico, and L. Cavallaro, *Detection of Intrusions and Malware, and Vulnerability Assessment*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2010, ch. Take a Deep Breath: A Stealthy, Resilient and Cost-Effective Botnet Using Skype.

[19] A. Berger and M. Hefeeda, "Exploiting sip for botnet communication," *2009 5th IEEE Workshop on Secure Network Protocols*, pp. 31–36, 2009. [Online]. Available: http://tinyurl.com/8n9rkex

[20] J. Caballero, P. Poosankam, C. Kreibich, and D. Song, "Dispatcher: Enabling active botnet infiltration using

automatic protocol reverse-engineering," in *ACM Conference on Computer and Communications Security*, Nov 2009.

[21] C. Y. Cho, D. Babi ć, E. C. R. Shin, and D. Song, "Inference and analysis of formal models of botnet command and control protocols," in *Proceedings of the 17th ACM conference on Computer and communications security*, ser. CCS '10. New York, NY, USA: ACM, 2010, pp. 426–439. [Online]. Available: http://doi.acm.org/10.1145/1866307.1866355

[22] J. Nazario, "Twitter based botnet command and control," http://tinyurl.com/8ojo5wl, 2009, [Online; accessed 15-December-2011].

[23] E. J. Kartaltepe, J. A. Morales, S. Xu, and R. Sandhu, "Social network-based botnet command-and-control: emerging threats and countermeasures," in *Proceedings of the 8th international conference on Applied cryptography and network security*, ser. ACNS'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 511–528. [Online]. Available: http://dl.acm.org/citation.cfm?id=1894302.1894342

[24] A. Lelli, "Trojan.whitewell: Whats your (bot) facebook status today?" http://tinyurl.com/yeb5rvb, 2009, [Online; accessed 15-December-2011].

[25] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szyd-lowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: Analysis of a botnet takeover," in *Proceedings of the 16th ACM conference on Computer and Communications Security (CCS)*, Nov 2009.

[26] R. Westervelt, "Botnet masters turn to google, social networks to avoid detection," http://tinyurl.com/8ta9nly, 2009, [Online; accessed 15-December-2011].

[27] Damballa, "The command structure of the operation aurora botnet: History, patterns, and findings," Tech. Rep., 2010.

[28] R. Deibert, A. Manchanda, R. Rohozinski, N. Villeneuve, and G. Walton, "Tracking ghostnet: Investigating a cyber espionage network," *Network*, vol. JR02-2009, no. JR02-2009, p. 53, 2009. [Online]. Available: http://tinyurl.com/d5q3cj

[29] A. Cole, M. Mellor, and D. Noyes, "Botnets: The rise of the machines," in *Proceedings on the 6th Annual Security Conference*, 2007.

[30] D. Wang, S. Savage, and G. M. Voelker, "Juice: A longitudinal study of an seo campaign," in *NDSS*, San Diego, CA, USA, 2013.

[31] N. Lewis, "Zeus botnet analysis: Past, present and future threats," http://tinyurl.com/d3vwcll, 2010, [Online; accessed 15-December-2011].

[32] A. Decker, D. Sancho, L. Kharouni, M. Goncharov, and R. McArdle, "Pushdo/cutwail botnet: A study of the pushdo/ cutwail botnet," TrendMicro Labs, Tech. Rep., 2009.

[33] I. Traynor, "Russia accused of unleashing cyberwar to disable estonia," http://tinyurl.com/5pmk5g, May 2007, [Online; accessed 12-December-2011].

[34] B. Stone-Gross, G. S. T. Holz, and G. Vigna., "The underground economy of spam: A botmasters perspective of coordinating large-scale spam campaigns," in *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2011.

[35] E. Athanasopoulos, A. Makridakis, S. Antonatos, D. Antoniades, S. Ioannidis, K. Anagnostakis, and E. Markatos, "Antisocial networks: turning a social network into a botnet," in *Proceedings of the 11th Information Security Conference*, Taipei, Taiwan, 2008.

[36] CWSandBox, "Cwsandbox," http://mwanalysis.org/, [Online; accessed 15-December-2011].

[37] A. Mushtaq, "The dead giveaways of vm-aware malware," http://tinyurl.com/4ejusr2, Jan 2011, [Online; accessed 15-December-2011].

[38] S. Shin and G. Gu, "Conficker and beyond: A large-scale empirical study," in *Proceedings of Annual Computer Security Applications Conference (ACSAC)*, 2010.

[39] L. T. Borup, "Peer-to-peer botnets: A case study on waledac," Master's thesis, Technical University of Denmark, Kongens Lyngby, Denmark, 2009.

[40] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. Dagon, "From throw-away traffic to bots: detecting the rise of dga-based malware," in *Proceedings of the 21st USENIX conference on Security symposium*, ser. Security'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 24–24. [Online]. Available: http://dl.acm.org/citation.cfm?id=2362793.2362817

[41] V. Tiu, "Information about worm:win32/conficker.d," http://tinyurl.com/8c84vhl, March 2009, [Online; accessed 15-December-2011].

[42] Tor, "Tor: Anonymity online," https://www.torproject.org/, [Online; accessed 20-December-2011].

[43] D. Brown, "Resilient botnet command and control with tor." Presented at DEF CON 18, Las Vegas, Nevada, USA, 2010.

[44] Tor, "Tor hidden services," https://www.torproject.org/docs/hidden-services.html.en, [Online; accessed 12-November-2012].

[45] D. Fisher, "Storm, nugache lead dangerous new botnet barrage," http://tinyurl.com/8folbmw, 2007, [Online; accessed 15-December-2011].

[46] US-Cert, "Malware tunneling in ipv6." http://tinyurl.com/6zvv568, 2005, [Online; accessed 15-December-2011].

[47] E. Cooke, F. Jahanian, and D. McPherson, "The zombie roundup: Understanding, detecting, and disrupting botnets," *ACM USENIX Workshop on Steps to Reducing Unwanted Traffic on the Internet SRUTI*, vol. 7, pp. 39–44, 2005. [Online]. Available: http://tinyurl.com/8h94o9u

[48] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "Botminer: Clustering analysis of network trafficfor protocol- and structure-independent botnet detection," in *Usenix Security Symposium*, 2008.

[49] P. Wurzinger, L. Bilge, T. Holz, J. Gobel, C. Kruegel, and E. Kirda, "Automatically generating models for botnet detection," in *European Symposium on Research in Computer Security (ESORICS)*, 2009.

[50] J. Goebel and T. Holz, "Rishi: Identify bot contaminated hostsby irc nickname evaluation," in *USENIX Workshop on HotTopics in Understanding Botnets (HotBots'07)*, 2007.

[51] M. Roesch, "Snort - lightweight intrusion detection fornetworks," in *Proceedings of USENIX LISA'99)*, 1999.

[52] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "Bothunter: Detecting malware infection through ids-driven dialog correlation," in *Usenix Security Symposium*, 2007.

[53] L. Zhuang, J. Dunagan, D. Simon, H. Wang, I. Osipkov, G. Hulten, and J. Tygar, "Characterizing botnets from email spam records," in *USENIX Workshop on Large-Scale Exploits and Emergent Threats*, 2008.

[54] A. Ramachandran, N. Feamster, and D. Dagon, "Revealing botnet membership using dnsbl counter-

intelligence," in *Conference on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, 2006.

[55] R.Villamarin-Salomon and J. Brustoloni, "Identifying botnets using anomaly detection techniques applied to dns traffic," in *Proc. 5th IEEE Consumer Communications and Networking Conference (CCNC 2008)*, 2008.

[56] H. Choi, H. Lee, H. Lee, and H. Kim, "Botnet detection by monitoring group activities in dns traffic," in *Proc. 7th IEEE International Conference on Computer and Information Technology (CIT 2007)*, 2007.

[57] J. Zhang, Y. Xie, F. Yu, D. Soukal, and W. Lee, "Intention and origination: An inside look at large-scale bot queries," in *to appear in NDSS*, 2013.

[58] L. Aniello, G. Lodi, and R. Baldoni, "Inter-domain stealthy port scan detection through complex event processing," in *Proceedings of the 13th European Workshop on Dependable Computing*, ser. EWDC '11. New York, NY, USA: ACM, 2011, pp. 67–72. [Online]. Available: http://doi.acm.org/10.1145/1978582.1978597

[59] L. Aniello, G. A. Di Luna, G. Lodi, and R. Baldoni, "A collaborative event processing system for protection of critical infrastructures from cyber attacks," in *Proceedings of the 30th international conference on Computer safety, reliability, and security*, ser. SAFECOMP'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 310–323. [Online]. Available: http://dl.acm.org/citation.cfm?id=2041619.2041651

[60] F. Doelitzscher, C. Reich, M. Knahl, and N. Clarke, "Incident detection for cloud environments," in *Proceedings of the Third International Conference on Emerging Network Intelligence (EMERGING 2011)*, Nov 2011.

[61] G. Gu, J. Zhang, and W. Lee, "Botsniffer: Detecting botnet command and control channels in network traffic," in *Network and Distributed System Security Symposium (NDSS)*, 2008.

[62] T. Wang and S.-Z. Yu, "Centralized botnet detection by traffic aggregation," in *IEEE International Symposium on Parallel and Distributed Processing with Applications*, 2009.

[63] T. Strayer, R. Walsh, C. Livadas, and D. Lapsley, "Detecting botnets with tight command and control," in *Proceedings 2006 31st IEEE Conference on Local Computer Network*, 2006.

[64] T. Strayer, D. Lapsley, R. Walsh, and C. Livadas, *Botnet detection based on network behavior*, ser. Advances in Information Security. Springer, 2008, vol. 36, pp. 1–24.

[65] T.-F. Yen and M. K. Reiter, "Traffic aggregation for malware detection," in *Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, 2008.

[66] L. Liu, S. Chen, G. Yan, and Z. Zhang, "Bottracer:execution-based bot-like malware detection," in *11th Information Security Conference*, 2008.

[67] E. Stinson and J. C. Mitchell, "Characterizing bots' remote control behavior," in *International Conference on Detectionof Intrusions & Malware, and Vulnerability Assessment (DIMVA)*, 2007.

[68] W. Lu, M. Tavallaee, and A. A. Ghorbani, "Automatic discovery of botnet communities on large-scale communication networks," in *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, ser. ASIACCS '09. New York, NY, USA: ACM, 2009, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/1533057.1533062

[69] J. Binkley and S. Singh, "An algorithm for anomaly-based botnet detection," in *Usenix Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI)*, 2006.

[70] A. Karasaridis, B. Rexroad, and D. Hoeflin, "Wide-scale botnet detection and characterization," in *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*, 2007.

[71] T.-F. Yen and M. K. Reiter, "Are your hosts trading or plotting? telling p2p file-sharing and bots apart," in *International Conference on Distributed Computing Systems*, 2010.

[72] E. Passerini, R. Paleari, L. Martignoni, and D. Bruschi, "Fluxor : Detecting and monitoring fast-flux service networks," *Detection of Intrusions and Malware and Vulnerability Assessment*, pp. 186–206, 2008. [Online]. Available: http://tinyurl.com/c735zn9

[73] T. Holz, C. Gorecki, K. Rieck, and F. Freiling, "Detection and Mitigation of Fast-Flux Service Networks," in *Proceedings of NDSS 2008*, San Diego, CA, USA, Feb. 2008. [Online]. Available: http://tinyurl.com/9q64vk5

[74] G. Gu, V. Yegneswaran, P. Porras, J. Stoll, and W. Lee, "Active botnet probing to identify obscure command and control channels," in *Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC)*, 2010.

[75] M. Neugschwandtner, P. M. Comparetti, and C. Platzer, "Detecting malware's failover c&c strategies with squeeze." in *ACSAC*, R. H. Zakon, J. P. McDermott, and M. E. Locasto, Eds. ACM, 2011, pp. 21–30. [Online]. Available: http://tinyurl.com/9wz2uvj

[76] R. Perdisci, W. Lee, and N. Feamster, "Behavioral clusteringof http-based malware and signature generation using malicious network traces," in *USENIX Symposium on Networked Systems Design & Implementation (NSDI)*, 2010.

[77] C. Rossow and C. J. Dietrich, "Provex: Detecting botnets with encrypted command and control channels," in *DIMVA*, 2013, pp. 21–40.

[78] T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, 2008. [Online]. Available: http://tinyurl.com/9hpoa3d

[79] C. Livadas, R. Walsh, D. Lapsley, and W. Strayer, "Using machine learning technliques to identifybotnet traffic," in *Proceedings of the 2nd IEEELCN Workshop on Network Security*, Nov 2006.

[80] G. Jacob, R. Hund, T. Holz, and C. Kruegel, "Jackstraws: Picking command and control connections from bot traffic," in *USENIX Security Symposium*, 2011.

[81] S. Savage, D. Wetherall, A. R. Karlin, and T. E. Anderson, "Practical network support for ip traceback," *Computer Communication Review*, vol. 30, pp. 295–306, 2000.

[82] D. X. Song and A. Perrig, "Advanced and authenticated marking schemes for ip traceback," in *IEEE INFOCOM*, 2001, pp. 878–886.

[83] S. M. Bellovin, M. Leech, and T. Taylor, "ICMP traceback messages," Obsolete Internet draft, February 2003. [Online]. Available: http://tinyurl.com/be2sa93

[84] A. Mankin, D. Massey, C. long Wu, S. F. Wu, and L. Zhang, "On design and evaluation of "intention-driven" icmp traceback," in *International Conference on Computer Communications and Networks*, 2001.

[85] A. Belenky and N. Ansari, "Ip traceback with deterministic packet marking," 2003.

[86] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against ddos attacks," in *IEEE Symposium on Security and Privacy*, 2003, pp. 93–107.

[87] D. Ramsbrock, X. Wang, and X. Jiang, "A first step

towards live botmaster traceback," in *Proceedings of the 11th international symposium on Recent Advances in Intrusion Detection*, ser. RAID '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 59–77. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-87403-4_4

[88] A. C. Snoeren, "Hash-based ip traceback," *Computer Communication Review*, vol. 31, pp. 3–14, 2001.

[89] S. Staniford-Chen and L. T. Heberlein, "Holding intruders accountable on the internet," in *Proceedings of the 1995 IEEE Symposium on Security and Privacy*, ser. SP '95. Washington, DC, USA: IEEE Computer Society, 1995, pp. 39–. [Online]. Available: http://dl.acm.org/citation.cfm?id=882491.884246

[90] Y. Zhang and V. Paxson, "Detecting stepping stones," in *USENIX Security Symposium*, 2000.

[91] X. Wang, D. S. Reeves, and S. F. Wu, "Inter-packet delay based correlation for tracing encrypted connections through stepping stones," in *European Symposium on Research in Computer Security (ESORICS)*, 2002, pp. 244–263.

[92] D. L. Donoho, A. G. Flesia, U. Shankar, V. P. J. Coit, , S. Staniford, J. Coit, and S. Staniford, "Multiscale stepping-stone detection: Detecting pairs of jittered interactive streams by exploiting maximum tolerable delay," in *Proc. of The 5th International Symposium on Recent Advances in Intrusion Detection (RAID)*. Springer, 2002, pp. 17–35.

[93] A. Blum, D. Song, and S. Venkataraman, "Detection of interactive stepping stones: Algorithms and confidence bounds," in *Conference of Recent Advance in Intrusion Detection (RAID), (Sophia Antipolis, French Riviera*. Springer, 2004, pp. 258–277.

[94] L. Zhang, A. G. Persaud, A. Johnson, and Y. Guan, "Detection of stepping stone attack under delay and chaff perturbations," in *International Performance, Computing, and Communications Conference*, 2006.

[95] X. Wang and D. S. Reeves, "Robust correlation of encrypted attack traffic through stepping stones by manipulation of interpacket delays," in *Proceedings of the 10th ACM conference on Computer and communications security*, ser. CCS '03. New York, NY, USA: ACM, 2003, pp. 20–29. [Online]. Available: http://doi.acm.org/10.1145/948109.948115

[96] W. T. Strayer, C. E. Jones, I. Castineyra, J. B. Levin, and R. R. Hain, "An integrated architecture for attack attribution," *Tech. Rep. BBN REPORT-8384*, Dec 2003.

[97] K. Yoda and H. Etoh, "Finding a connection chain for tracing intruders," in *Proceedings of the 6th European Symposium on Research in Computer Security (ESORICS)*, 2000, pp. 191–205.

[98] X. Wang, X. Wang, D. S. Reeves, D. S. Reeves, S. F. Wu, S. F. Wu, J. Yuill, and J. Yuill, "Sleepy watermark tracing: An active network-based intrusion response framework," in *Proc. of the 16th International Information Security Conference*, 2001, pp. 369–384.

[99] J. B. Grizzard and T. Johns, "Peer-to-peer botnets: Overview and case study," in *USENIX Workshop on Hot Topics in Understanding Botnets (HotBots07)*, 2007.

[100] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "A multifaceted approach to understanding the botnet phenomenon," in *ACM Internet Measurement Conference(IMC)*, 2006.

[101] M. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "My botnetis bigger than yours (maybe, better than yours) : Why size estimates remain challenging," in *USENIX Workshop on Hot Topics in Understanding Botnet*, 2007.

[102] J. Zhang, X. Luo, R. Perdisci, G. Gu, W. Lee, and N. Feamster, "Boosting the scalability of botnet detection using adaptive traffic sampling," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '11. New York, NY, USA: ACM, 2011, pp. 124–134. [Online]. Available: http://doi.acm.org/10.1145/1966913.1966930

[103] FireEye, "Next generation threat protection- fireeye inc." http://www.fireeye.com/, [Online; accessed 12-December-2011].

[104] Damballa, "Damballa::homepage," http://www.damballa.com/, [Online; accessed 12-December-2011].

[105] T. Holz, M. Engelberth, and F. Freiling, "Learning more about the underground economy: A case-study of keylog-gers and dropzones," in *European Symposium on Research in Computer Security (ESORICS)*, 2009.

[106] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage, "Spamalytics: An empirical analysis of spam marketing conversion," in *Proceedings of the 15th ACM Conference on Computer and Communications Security*, Alexandria, Virginia, USA, Oct 2008, pp. 3–14.

[107] R. Ford and S. Gordon, "Cent, five cent, ten cent, dollar: hitting botnets where it really hurts," in *Proceedings of the 2006 Workshop on New Security Paradigms (NSPW'06)*. New York, NY, USA: ACM, 2007, p. 310.

[108] URIBL, "uribl-website," http://www.uribl.com/, [Online; accessed 12-December-2011].

[109] T. A. Meyer and B. Whateley, "SpamBayes: Effective open-source, Bayesian based, email classification system," in *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, 2004.

[110] A. Pitsillidis, K. Levchenko, C. Kreibich, C. Kanich, G. Voelker, V. Paxson, N. Weaver, and S. Savage, " Botnet Judo: Fighting Spam with Itself ," in *Proceedings of the 17th Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, March 2010.

[111] T. Peng, C. Leckie, and K. Ramamohanarao, "Survey of network-based defense mechanisms countering the dos and ddos problems," *ACM Comput. Surv.*, vol. 39, April 2007.

[112] P. Bright, "How Operation b107 decapitated the Rustock botnet," http://tinyurl.com/4bajdjx, 2011, [Online; accessed 10-December-2011].

[113] B. Furfie, "Laws must change to combat botnets Kaspersky," http://tinyurl.com/9tpdyj5, Feb 2011, [Online; accessed 10-December-2011].

[114] A.-P. E. C. AEC, "Guide on Policy and Technical Approaches against Botnet," http://tinyurl.com/9b68qmj, Dec 2008, [Online; accessed 10-December-2011].

[115] J. Leyden, "Botnet-harbouring survey fails to accounts for sinkholes," http://tinyurl.com/cgohqqg, Oct 2010, [Online; accessed 10-December-2011].

[116] G. L. Orgill, G. W. Romney, M. G. Bailey, and P. M. Orgill, "The urgency for effective user privacy-education to counter social engineering attacks on secure computer systems," in *Proceedings of the 5th conference on Information technology education*, ser. CITC5 '04. New York, NY, USA: ACM, 2004, pp. 177–181.

[117] N. Mody, M. O'Reirdan, S. Masiello, and J. Zebek, "Common best practices for mitigating large scale bot infections in residential networks," *MAAWG*, july 2009.

[118] T. Ormerod, L. Wang, M. Debbabi, A. Youssef, H. Binsalleeh, A. Boukhtouta, and P. Sinha, "Defaming botnet toolkits: A bottom-up approach to mitigating the threat," in *Proceedings of the 4th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE)*, 2010.

[119] SecuriTeam, "Sasser worm remote ftpd buffer overflow exploit code (port 5554)," http://tinyurl.com/cdeyuy6, 2004, [Online; accessed 15-December-2011].

[120] C. Y. Cho and J. Caballero, "Botnet infiltration: Finding bugs in botnet command and control," http://tinyurl.com/cz4j2z4, [Online; accessed 15-December-2011].

[121] J. R. Douceur, "The sybil attack," in *Proceedings of the International workshop on Peer-To-Peer Systems (IPTPS)*, March 2002.

[122] A. Singh, T.-W. J. Ngan, P. Druschel, and D. S. Wallach, "Eclipse attacks on overlay networks: Threats and defenses," in *IEEE International Conference on Computer Communications (Infocom)*, 2006.

[123] E. Karamatli, "Modern botnets: A survey and future directions," http://tinyurl.com/cesnb7e, Bogazici University,Turkey, 2011.

[124] G. Ollmann, "Serial variant evasion tactics," 2009. [Online]. Available: http://tinyurl.com/93jnurm

[125] K. Bong and J. Brozyck, "Managing large botnets," http://tinyurl.com/blcuxbo, 2007, [Online; accessed 15-December-2011].

[126] M. Bailey, E. Cooke, F. Jahanian, Y. Xu, and M. Karir, "A survey of botnet technology and defenses," in *Proceedings of the 2009 Cybersecurity Applications & Technology Conference for Homeland Security*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 299–304. [Online]. Available: http://dl.acm.org/citation.cfm?id=1524292.1524347

[127] M. Feily, A. Shahrestani, and S. Ramadass, "A survey of botnet and botnet detection," *2009 Third International Conference on Emerging Security Information Systems and Technologies*, pp. 268–273, 2009. [Online]. Available: http://tinyurl.com/9njpehq

[128] H. R. Zeidanloo, M. J. Z. shooshtari, M. . Safari, P. V. Amoli, and M. Zamani, "A taxonomy of botnet detection techniques," *3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, pp. 158–162, 2010. [Online]. Available: http://tinyurl.com/9ttpjwm

[129] S. Axelsson, "Intrusion Detection Systems: A Survey and Taxonomy," Chalmers Univ., Tech. Rep. 99-15, Mar. 2000. [Online]. Available: http://tinyurl.com/coej8xx

[130] E. Stinson and J. C. Mitchell, "Towards systematic evaluation of the evadability of bot/botnet detection methods," in *Proceedings of the 2nd conference on USENIX Workshop on offensive technologies*. Berkeley, CA, USA: USENIX Association, 2008, pp. 5:1–5:9. [Online]. Available: http://dl.acm.org/citation.cfm?id=1496702.1496707

[131] S. García, A. Zunino, and M. Campo, "Survey on network-based botnet detection methods," *Security Comm. Networks*, p. n/a, Jun. 2013. [Online]. Available: http://dx.doi.org/10.1002/sec.800

[132] F. Leder, T. Werner, and P. Martini, "Proactive botnet countermeasures an offensive approach," in *Cooperative Cyber Defence Centre of Excellence*, Tallinn, Estonia, March 2009.

[133] S. Stankovic and D. Simic, "Defense strategies against modern botnets," *International Journal of Computer Science and Information Security*, vol. 2, no. 1, 2009.

[134] I. Ullah, N. Khan, and H. A. Aboalsamh, "Survey on botnet: Its architecture, detection, prevention and mitigation," in *ICNSC*, 2013, pp. 660–665.

[135] M. Eslahi, R. Salleh, and N. B. Anuar, "Bots and botnets: An overview of characteristics, detection and challenges," in *proceedings of 2012 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, 2012, pp. 349–354.

[136] Z. Zhu, G. Lu, Y. Chen, Z. J. Fu, P. Roberts, and K. Han, "Botnet research survey," in *Proceedings of the 2008 32nd Annual IEEE International Computer Software and Applications Conference*, ser. COMPSAC '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 967–972. [Online]. Available: http://dx.doi.org/10.1109/COMPSAC.2008.205

[137] S. S. C. Silva, R. M. P. Silva, R. C. G. Pinto, and R. M. Salles, "Botnets: A survey," *Computer Networks*, Oct. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2012.07.021

[138] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel, "Disclosure: Detecting botnet command and control servers through large-scale netflow analysis," in *ACSAC*, 2012. [Online]. Available: http://www.iseclab.org/papers/disclosure.pdf

[139] SecurityFocus, "Zeus botnet finds hold in amazon cloud," http://www.securityfocus.com/brief/1046, 2009, [Online; accessed 12-November-2012].

[140] I. Burke, "Who needs botnets if you have google?" Presented at ZaCon2, Johannesburg, South Africa, 2010.

[141] K. P. Clark, M. Warnier, and F. M. T. Brazier, "Botclouds - the future of cloud-based botnets," in *CLOSER*, F. Leymann, I. Ivanov, M. van Sinderen, and B. Shishkov, Eds. SciTePress, 2011, pp. 597–603. [Online]. Available: http://tinyurl.com/c6cqkkd

[142] Stratsec, "botcloud ? an emerging platform for cyberattacks," http://tinyurl.com/cubnghx, 2012, [Online; accessed 12-November-2012].

[143] C. Xiang, F. Binxing, Y. Lihua, L. Xiaoyi, and Z. Tianning, "Andbot: towards advanced mobile botnets," in *Proceedings of the 4th USENIX conference on Large-scale exploits and emergent threats*, ser. LEET'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 11–11. [Online]. Available: http://dl.acm.org/citation.cfm?id=1972441.1972456

[144] Symantec, "Android.bmaster: A million-dollar mobile botnet," http://tinyurl.com/a4bdljv, 2012, [Online; accessed 12-November-2012].

[145] K. Inc., "Irc bot for android." http://tinyurl.com/7xrmlcb, 2012, [Online; accessed 12-November-2012].

[146] X. Jiang, "Security alert: Anserverbot, new sophisticated android bot found in alternative android markets." http://www.csc.ncsu.edu/faculty/jiang/AnserverBot/, 2011, [Online; accessed 12-November-2012].

[147] Micorsoft, "Flame malware collision attack explained," http://tinyurl.com/dxxlb5j, June 2012, [Online; accessed 1-Aug-2012].

[148] L. of Cryptography of Systems Security (CrySyS), "Duqu: A stuxnet-like malware found in the wild, technical report," http://tinyurl.com/dxxlb5j, October 2011, [Online; accessed 1-Aug-2012].

[149] K. Lab, "Gauss:abnormal distribution,"

http://tinyurl.com/8p34yp7, September 2012, [Online; accessed 12-Dec-2012].

[150] ——, “Shamoon the wiper - copycats at work,” http://tinyurl.com/a9axwgx, August 2012, [Online; accessed 12-Dec-2012].

[151] Seculert, “Mahdi - the cyberwar savior?” http://tinyurl.com/brp64k4, July 2012, [Online; accessed 12-Dec-2012].

[152] InternetWorldStats, “Internet growth statistics,” http://www.internetworldstats.com/emarketing.htm, [Online; accessed 12-December-2011].

[153] J. Stanton, K. Stam, P. Mastrangelo, and J. Jolton, “Analysis of end user security behaviors,” *Computers Security*, vol. 24, no. 2, pp. 124–133, 2005. [Online]. Available: http://tinyurl.com/b3k6fg6

[154] J. Fielding, “25% of all computers on botnets,” http://tinyurl.com/9e7bdkr, January 2007, [Online; accessed 12-December-2011].

[155] CIOinsight, “Botnets still a major threat, researchers say at rsa,” http://tinyurl.com/cw5bypo, February 2011, [Online; accessed 12-December-2011].

[156] M. S. Mimoso, “Fbi takes down dns changer botnet; aided $14 million click fraud scheme,” http://tinyurl.com/8nltjzf, Nov 2011, [Online; accessed 12-December-2011].

[157] D. Goodin, “Waledac botnet ‘decimated’ by ms takedown,” http://tinyurl.com/7apnn9b, March 2010, [Online; accessed 12-December-2011].

[158] ITN-News, “‘slain’ kelihos botnet still spams from beyond the grave,” http://tinyurl.com/9esmb3e, Feb 2012, [Online; accessed 1-February-2012].

[159] R. McMillan, “China cleans up spam problem,” http://tinyurl.com/clnedlt, February 2011, [Online; accessed 12-December-2011].

[160] BotHunter, “Bothunter: A network-based botnet diagnostic system,” http://www.bothunter.net/, [Online; accessed 12-December-2011].

[161] TrendMicro, “Rubotted,” http://tinyurl.com/yd62cb8, [Online; accessed 12-December-2011].

[162] N. R. Ramay, S. Khattak, A. A. Syed, and S. A. Khayam, “Bottleneck: A generalized, flexible and extensible framework for botnet defense,” in *IEEE Symposium on Security and Privacy, 2012*, May 2012, poster Paper.

Sheharbano Khattak Sheharbano Khattak will start her PhD in Computer Science at University of Cambridge in October 2013. She did MS in Computer and Communication Security with distinction from NUST, Pakistan in 2013 and BS in Computer Science from IIUI, Pakistan in 2010. Her current areas of interest include Internet censorship, privacy and anonymity. She has worked on Network Intrusion Detection Systems, botnet detection tools and measurement of network data.

Naurin Rasheed Ramay Naurin Rasheed Ramay received her M.Sc in Security and Privacy as a Fulbright scholar from Stevens Institute of Technology. She completed her BE in Information and Communication Systems from NUST School of Electrical Engineering and Computer Science. She worked as a research assistant in the area of Network Security where she gained knowledge of botnets, intrusion and anomaly detection methodologies. Her research interests are network security, cryptography and privacy.

Kamran Riaz Khan Kamran Riaz Khan received his BS degree in Telecommunication Engineering in 2012 from National University of Computer and Emerging Sciences, Pakistan. He is working with SysNet labs at NUCES Islamabad as a Research Assistant in the area of network security. He has participated twice in Google Summer of Code as a student, with Ubuntu and Tor in 2010 and 2011 respectively. His research interests include privacy protecting technologies and network security.

Affan A. Syed Affan A. Syed is an Assistant Professor in EE department and leads the SysNet lab at National University of Computer and Emerging Sciences (NUCES), Pakistan. His research interest spans a wide spectrum; from exploring the deep linkages between sensing and energy in cyber-physical systems to distributed systems, embedded systems, and computer security. He has done research in underwater sensor networks, network time-synchronization, high-latency medium access control design, and industrial wireless-mesh networking. He received his B.S. and M.S. in Electrical Engineering from National University of Science and Technology (NUST), Pakistan and University of Southern California (USC) in 2001, and 2004 respectively. He completed his Ph.D. in Computer Science from USC in 2009 and then worked as a Post-Doctoral Research Associate at the Information Sciences Institute (ISI) in Marina del Rey, CA. He is a member of ACM and IEEE, and part of the Phi Kappa Phi Honor society.

Syed Ali Khayam Syed Ali Khayam received his MS and PhD degree in Electrical Engineering from Michigan State University in May 2003 and December 2006, respectively. From February 2007 to March 2012, he served as an assistant professor at the School of Electrical Engineering & Computer Science (SEECS), National University of Science and Technology (NUST), Pakistan. In June 2011, he co-founded a startup company, xFlow Research which provided consultancy services to several networking and semiconductor vendors, including Marvell Technologies, Broadcom, Dell, Netgear, Cavium Networks, Tellabs and Calient Technologies. Currently he is with PLUMgrid Inc., a network infrastructure software vendor. Dr. Khayam has over 70 publications in some of the most prestigious conferences and journals, including IEEE Transactions, ACM and Elsevier journals, ACM CCS, IEEE Infocom, RAID and ACSAC. He has received research awards from Nokia Research, Korean Research Foundation, Pakistan National ICT R&D Fund and Marvell Technologies. He also has 7 patents pending at USPTO, some of which were indigenously drafted and filed by him.