2011 International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11

A Survey on Latest Botnet Attack and Defense

Lei Zhang, Shui Yu, Di Wu
School of Information Technology
Deakin University
Burwood, VIC 3125, Australia
Email: {leiz, syu, wudi}@deakin.edu.au

Paul Watters
Internet Commerce Security Laboratory
University of Ballarat
Ballarat, VIC 3364, Australia
Email: p.watters@ballarat.edu.au

***Abstract*—A botnet is a group of compromised computers, which are remotely controlled by hackers to launch various network attacks, such as DDoS attack and information phishing. Botnet has become a popular and productive tool behind many cyber attacks. Recently, the owners of some botnets, such as storm worm, torpig and conflicker, are employing fluxing techniques to evade detection. Therefore, the understanding of their fluxing tricks is critical to the success of defending from botnet attacks. Motivated by this, we survey the latest botnet attacks and defenses in this paper. We begin with introducing the principles of fast fluxing (FF) and domain fluxing (DF), and explain how these techniques were employed by botnet owners to fly under the radar. Furthermore, we investigate the state-of-art research on fluxing detection. We also compare and evaluate those fluxing detection methods by multiple criteria. Finally, we discuss future directions on fighting against botnet based attacks.**

***Index Terms*—Survey; Botnet; Fast Fluxing; Domain Fluxing**

I. Introduction

Botnets have become one of the most malicious threats over the Internet. A botnet is a group of compromised computers also called bots or zombies which are controlled by the botmaster's malware code. Any unprotected computers in any locations in the world may become bots as long as they are connecting with Internet and have been infected by botnet malware code. The master computer communicates with its bots by a command and control (C&C) channel, which passes commands from the botmaster to bots, and transmits stolen information from bots to their master. The attacked bots can also infect other computers enabling them to be botnet members [1]. In recent years, fluxing techniques have been applied into many botnets to evade detection. This brings additional challenge to botnet researchers.

Botnets have been used by cyber-criminals to conduct many malicious activities, such as sending spam emails [2], launching DOS attacks [3], and stealing private data [4] [5]. A report from Symantec's MessageLabs shows 90.4% of total emails were spam in June 2009. Among all spam, 83.2% was sent through botnets. In addition, many spam emails included viruses, phishing attacks, and web-based malware. Therefore, sending spam through botnets can help to conduct further network attacks [6].

Researchers have applied signature-based methods to detect botnets for long time. These signature-based techniques have been widely employed by some Honeynet projects, which has been discussed in [7] [8]. However, these methods cannot detect newly developed botnets as the signatures of new botnets are unknown or some botnets are polymorphic [9]. Some IRC-based approaches were developed to overcome this problem. For example, Binkley et al. developed an anomaly based system combining IRC statistics and TCP work load [10], and Karasaridis et al. applied a passive anomaly-based characterization methodology based on botnets behavior characteristics [11]. However, these methods have high false alarm rates [9].

Many researchers focused on how to detect botnets or trace the botnet master. Meanwhile, many surveys reflected what had been done and summarized what future work should be. Feily et al. surveyed botnet mechanisms and botnet detection techniques based on different classes they identified: signature-based, anomaly-based, DNS-based, and mining-based. They also compared and evaluated the advantages and disadvantages of some typical researches from each category [12]. But what was missing in previous surveys is that no one has investigated why recent botnets are so difficult to be detected and how recent methods detect new fluxing features. In our survey, we focus on two advanced botnet mechanisms:

- Fast flux (FF): A mechanism that a set of IP addresses change frequently corresponding to a unique domain name
- Domain flux (DF): A mechanism that a set of domain names are generated automatically and periodically corresponding to a unique IP address

As far as we know, no surveys have focused on botnet fluxing detections up to now. Furthermore, no work has been done to evaluate the quality of a fluxing detection model. Our main contributions of this survey are as bellow.

- We summarized and classified the latest botnet fluxing features and detection techniques for the research community.
- We also explained how attackers could apply these techniques to improve their survivability against detection. Corresponding to this botnet "evolution", we investigate some most recent researches which can detect or mitigate fluxing botnets to some extent.
- We compared and evaluated the surveyed works against multiple criteria.

The rest of this paper is structured as follows. In Section II, we explain how fast flux (FF) and domain flux (DF) works

978-0-7695-4600-1/11 $26.00 © 2011 IEEE
DOI 10.1109/TrustCom.2011.11

in some advanced botnets. Following that, we introduce how recent approaches were developed to detect botnets based on fast fluxing and domain fluxing techniques in Section III. In Section IV, we compare these introduced detection methods by analyzing effectiveness through different evaluation criteria. In Section V, we briefly introduce the stage of mitigating fluxing botnets. At last, in Section VI, we provide future directions and highlight the main points in this survey in Section VII.

II. New Techniques Applied in Botnets

In this section, we investigate some advanced techniques of botnets in recent years. These techniques make the botnet more sophisticated to be detected. Some botnets might use fluxing methods to evade detections by hiding the domain-IP mappings. To achieve this, they might use fast flux (FF) or domain flux (DF) to improve the survivability. These techniques have been widely applied to some botnets for launching phishing, pharming, and malware distribution [13].

A. Fast Flux

Fast flux (FF) is an earlier way to evade botnet detections. By IP fast fluxing, the mapping between multiple IP addresses and one single domain name is rapidly changing [14]. This technique makes it more sophisticated to block or take down the C&C Server.

Networks which apply fast fluxing techniques are called fast fluxing (FF) Network. No matter legal or suspicious FF networks, they show almost the same characteristics, such as short TTLs and large IP pools [13]. Furthermore, fast-flux can be classified into two categories - single flux and double flux.

By single flux, a domain may be resolved to different IPs in different time ranges. For example, a user accesses the same domain twice in a short while. For the first time, the user sends a DNS query to the DNS server, which resolves that the corresponding IP address is IP1 "11.11.11.11". With IP1, the user accesses the fluxing agent FA1, which redirects the request to the real server "mothership". This "mothership" then processes the request and responds to FA1. Finally, FA1 relays the response back to the user. After a while, the user accesses the same domain again. However, due to a short TTL of IP addresses, IP1 has expired; therefore, the user needs to query the updated IP address again of that domain. Now, the DNS server responses to the user with a different IP address - IP2 "22.22.22.22". Then the user uses this new address IP2 to connect to another flux agent FA2 which redirects the user to the "mothership" [13].

Double-flux is a more sophisticated way of counter-detection. It involves the repeated changing of both the flux agents and the registration in DNS servers. That is to say, in addition to fluxing their own agents, the authoritative domain name server is a part of fluxing as well. This provides an additional layer of redundancy within malware networks. The fluxing nodes register and de-register in the Domain Name Server repeatedly [14].

Fast-fluxing network techniques have been abused by attackers to maintain their botnets. This is known as fast fluxing network attack (FFNA). In this case, almost all compromised computers become fluxing agents. Agents can be added or removed from the agent pool dynamically; thus, any mechanism which tends to block agents cannot take down the whole botnet [15].

B. Domain Flux

However, due to a single domain name, fast flux has a single point of failure once fluxing is identified. A more survivable mechanism is domain flux (DF). Stone-Gross et al. pointed out that some recent botnet programs such as Torpig resolved this problem by using DF. Inspired by [16] [17] [18], which discussed domain-generation techniques, in 2009, they provided a research report on how they took over an advanced DF based botnet - Torpig [19].

Stone-gross et al. explained the process of DF in their research of taking over Torpig. DF is based on the idea of generating domains through a domain generation algorithm (DGA). Both C&C server and its bots follow the same algorithm seeded by the same value to obtain consistent dynamic domain names. Bots try to contact the C&C server and other servers controlled by the botnet master according to a domain list until one DNS query succeeds. If the current domain has been blocked or suspended by authorities, bots will try to calculate other domains by the DGA [19].

The key idea is that the algorithm must make sure that all bots can generate domains by the same seed. Stone-gross et al. revealed that Torpig calculates sub-domains by using the current week and year first but independent of the current day, and then appends the Top Level Domain (TLD). The domains generated might be "weekyear.com" or "weekyear.biz", etc. Next, the bots will use these auto-generated domain names to contact the C&C server. If failed, bots will use the day information to calculate the "daily domain", such as "day.com" or "day.net", etc. If all these domains cannot be resolved, bots will try to use the hard-coded domain names in a configuration file at the end [19].

III. Botnet Detection Techniques

A. Botnet Fast Flux Detection Techniques

Holz et al. claimed that they were the first to develop a metric to detect fast-flux service network (FFSN) empirically. They identified three possible parameters which can be used to distinguish normal network behaviors and FFSNs - the number of IP-domain mappings in all DNS lookups, the number of nameserver records in one single domain lookup, and the number of autonomous system in all IP-domain pairs. Based on these three parameters, they defined a metric - flux-score, which was a result of a linear decision function to predict the FFSN. A higher score indicates a higher fluxing degree, and vice versa. They evaluated their metric by a 10-fold validation using a two-month observation data. Results show that their method can distinguish normal network behavior from FFSN with a very low false positive rate [20].

There are some limitations in detection methods which focus on detecting domains that are related to IP addresses with

short TTL in DNS query results in [20] [21]. In 2009, Zhou et al. overcame these limitations by applying a behavior-analysis model [15]. To achieve this, they began with characterizing the behaviors of FF domains at some points around these FF domains. Based on the analysis of those behaviors, they presented an analytical model which showed the number of DNS queries required to confirm an FF domain. In addition, they speeded up the detection by two schemes. One scheme is to associate IP addresses with the queries' results from multiple DNS servers; the other one is to correlate queries' results with multiple possible FF domains. They also proved that the detection speed had been speeded up because of those correlation schemes. To avoid single point of failure and improve the scalability, they developed a collaborative intrusion detection architecture LarSID to support the distributed correlation using a peer-to-peer publish-subscribe mechanism for sharing evidence. Their results show that their decentralized model was 16 to 10,000 times faster than previous centralized model [21] with the same correlation schemes [15].

Caglayan et al. developed a real-time detection model of fast flux service networks (FFSN) [22]. Their model can monitor the DNS of a website by minutes using both active and passive methods in a distributed architecture. This model includes three key components - sensors, FF monitor database, fast flux monitor (FFM). In the first key component, there are three kinds of sensors - active sensors, passive sensors, and analytic sensors. Active sensors were designed to monitor several indicating parameters including TTL, FF activity index, and footprint index. Passive sensors, however, are just functional replication of active FFM sensors by leveraging DNS replication services. Analytic sensors are mainly responsible for checking whether the IP addresses used by a certain website existing in a blacklist.

In this model, another key component is the FF monitor database, which records data such as known FFSNs, zombie IPs used by domain names, etc., collected from above sensors. By analyzing data in the FFM database, some analytical knowledge can be harvested such as FFSN size, growth rate estimates, social network of a FFSN where IP addresses are shared by diverse FFSNs, footprints of a FFSN for a given ISP in a given country, etc. Finally, they developed a FFM classifier, which applied a Bayesian belief network to integrate multiple active and passive sensors. This classifier was then trained by the parameter values monitored by the active sensors to calculate a prediction confidence. They demonstrated empirically how their model can generate report to assist security analysts to evaluate the security of a website with acceptable accuracies. To improve the detection accuracy, the model calculated with a decision every 10 minutes. The more 10-minute sampling performed, the more accuracies it will reach [22].

Perdisci et al. in 2009 developed a recursive DNS (RDNS) tracing methodology to detect malicious flux service networks in-the-wild. In their model, a sensor was deployed in front of the RDNS server passively monitoring DNS traffic and stored information from a FF domains into a centralized data collector. Furthermore, to target to botnets, they developed some pre-filtering rules that can identify the malicious FF network. They considered a network as malicious FF network by four characteristics (short TTL, the change rate of the set of resolved IPs returned from each query, a large number of resolved IPs, and resolved IPs scattered across many different networks). Based on these rules, they developed filters to gather the traffic they required. Besides, the filters in the sensors can store lists containing historic information [23].

Other than most previous works, they conducted a fine-grained analysis on collected data. Firstly, they clustered domains with high relations based on their common features. Next, the clusters of the domains were then classified according to the resolved IP addresses. Finally, they applied a statistically supervised learning to build a network classifier to distinguish between malicious flux services and legitimate ones. Results show that their model can distinguish and classify malicious and benign FF network clearly. Another advantage is that the model can be used to improve the accuracy of spam filtering applications [23].

Yu et al. mentioned that one critical step to detect botnet fast flux was to distinguish the fast fluxing attack network (FFAN) and benign fast fluxing service network (FFSN) [13]. Their idea was an improvement of the method discussed in [20], which cannot distinguish benign ones from malicious ones. They identified the FFAN by observing the agent lifespan. They pointed out that all agents in FFSN should keep alive almost 24/7; by contrast, the alive time of FFAN bots is unpredictable to some extent, because the compromised computers cannot be controlled by FFAN master physically. Based on this lifespan difference between FFSN and FFAN, Yu et al. proposed two metrics and developed a monitoring system.

The first metric they defined is average online rate (AOR), which was measured once per hour within 24 hours. The AOR for FFSN should be close to 100%, because in most cases, FFSN agents are available 24/7. However, FFAN fluxing agents (bots) were out of attackers' control, thus the AOR is normally far bellow the AOR of FFSN. The other metric is the minimum available rate (MAR), which is the result of the number of times available out of the total measured times. For the same reason, the MAR of FFAN was far lower than that of FFSN [13].

Based on above two metrics, they developed a flux agents monitoring system consisting mainly of four components. The dig tool was developed to gather information and add new IPs into IP record database. The second part is the agents monitor which sends HTTP request to the IPs in the IP records database and records the responses. The third one is the IP lifespan records database storing the service status - 1 means service is available, 0 means unavailable. The last key component is a detector which judges between FFAN and FFSN by IP lifepan records and the two metrics. Results show that their system can distinguish FFAN and FFSN clearly because all benign FFSN have high and stable AOR and MAR, but the values of the two metrics in FFAN are much lower and less stable than those of FFSN [13].

B. Botnet Domain Flux Detection Techniques

There are many pioneers who have researched into how to detect fluxing domains in recent few years.

Stone-Gross et al. conducted an in-depth research on taking over a real world botnet - Torpig in 2009 [19]. In their report, by reverse-engineering the domain generation algorithm (DGA), they revealed that Torpig masters will not pre-register all possible domains in advance. Therefore, they registered the .com and .net domains which would be used by the botnet for three consecutive weeks. To achieve this, they purchased services from two different hosting providers who were notable for being unresponsive to abuse complaints. Besides, they also registered their .net and .com domains in two different registrars.

In the next step, they set up their Apache web server to log bots requests and record all network traffic. By this means, Torpig was sinkholed by another server. Furthermore, they determined the size of that Torpig by counting node identifiers N_{id} which are unique in Torpig. They also analyzed the advantages of this method by comparing to IP count which could be misled by DHCP. Their method was quite different from previous works in [24] [25]. Rajab et al. focused on detecting the size of IRC-based botnets by querying DNS server caches to estimate the number of bots who had connected the C&C server [24]. Kanich et al. worked on detecting P2P storm network size by active probing and crawling the over-net distributed hash table (DHT) [25]. Stone-Gross et al. pointed out that neither way was ideal to determine the size of Torpig [19].

Ma et al. applied a supervised machine learning to detect and prevent users from visiting malicious websites based on automated URL classification statistically [26]. It is a lightweight model which investigated the lexical features and host-based properties of malicious domain URLs. The lexical features which they selected are the length of entire URL, and the number of dots in the URL with of a bag-of-words representation. For host-based features, they extracted IP address properties, WHOIS (registrars) properties, domain name properties (TTL, etc.), and geographic properties (physical location, link speed, etc.). To train and evaluate their features, they applied three classification models - Naive Bayes, Support Vector Machine (SVM), and Logistic Regression on data sets from four different sources (two malicious, two benign ones). They found that WHOIS and lexical features can provide rich information and the combination of all features to form a full feature set can reach the highest accuracy. To compare full feature performance with traditional blacklist method, Ma et al. used a ROC graph to explain how full feature made difference. At last, they show how their classifiers selected automatically from large amount of features and determined the most predictive ones and achieved a modest false positive rate - 95% to 99% [26].

Later in 2009, Ma et al. developed an online learning approach based on the same two groups of features from their previous work [26]. Based on previous study, this new model can identify suspicious URLs over time by a live feed of labeled URLs from a large web mail provider. Live feed made this model more appropriate for online learning and processing large number of URLs much more efficiently than their previous batch method. They also pointed out and demonstrated that continuous feed of new features is the key to detect new malicious URLs. For their feature selection, lexical and host-based types account for 62% and 38% respectively. They achieved the online learning by two main steps. Firstly, they designed a sequence of feature-vector label pairs. The algorithm makes a label prediction by a linear classifier. After obtaining actual labels from prediction, the algorithm checks it with the labels in the feature-vector label pairs. If they do not match, the algorithm will record it as an error [27].

Ma et al.'s online method is a combination of classical and recent algorithms including four sub-algorithms. The first sub-algorithm is perception by which linear classifier makes update to a weight vector when there is any mistake. The second one is a method of applying stochastic gradient descent to logistic regression. This provides an online means to optimize an objective function and approximate the gradient of the original objective. The objective function is defined as a sum of the examples' individual objective functions and the model parameters are updated incrementally by the gradients of individual objectives. The third sub-algorithm is the passive-aggressive algorithm which is used to make a minimum change to correct any mistakes and low-confidence predictions. The last sub-algorithm is confidence-weighted (CW) algorithm by which less-confident weights are updated more aggressively than more confident ones. It also models uncertainties in weights by a Gaussian distribution to describe the per-feature confidence. CW can perform a fine-grained distinction among all features' weights confidence; therefore it is extremely appropriate for detecting malicious URLs as long as dynamic features can be introduced by the continuous data feed. Finally, they reached a high detection accuracy up to 99% over a balanced dataset [27].

In 2010, Jiang et al. proposed a light-weight anomaly detection approach by DNS failure graphs based on the failed DNS queries (unproductive DNS queries) [28]. In this graph, they captured all interactions between hosts and unresolvable domain names which could be considered as auto-generated domains used by a botnet. Edges are represented as associations between hosts and domains. They learnt from previous studies that failed DNS queries come from a small portion of human errors and mis-configurations [29], and a large part of other malicious activities [30] [31]. In a botnet, as all bots use the same algorithm to generate sub-domains, the queries for these domains will attribute to correlated failures, which construct dense subgraphs in a DNS failure graph. They believed that such subgraphs would show some interaction patterns.

To confirm their assumption, they gathered DNS query data from several major DNS servers in a large campus network for three months. After that, they recursively de-composed the DNS failure graph, and extracted dense subgraphs by

applying a statistical graph decomposition technique, which is an extension of the tri-nonnegative matrix factorization (TNMF) algorithm [32]. By analyzing the structure properties, they classified the subgraphs into three categories: host-star, DNS-star, and bi-mesh. By referring to some external data source such as domain name blacklists, they found that bi-mesh structures where a group of hosts are strongly associated with a group of domains reflect botnet activities [28].

Later, Prakash et al. developed a Phishnet to protect system from phishing attacks [33]. As shown in the Symantec's MessageLabs report [6], 83.2% spam was sent through botnets in June 2009. Thus, Phishnet should be able to detect auto-generated domain names of those spam sent through botnets. This is a blacklisting-based method which was improved to overcome some limitations of traditional blacklisting methods. They pointed out that it was easy to evade URL blacklisting because blacklisting methods needed to exactly match target URLs with entries in the list.

Two key components in their model can help to overcome this limitation. In the first component, there are five heuristics to enumerate simple combinations of known phishing URLs to discover new phishing sites. It works in an off-line fashion, examines current blacklists, and generates new URLs based on these heuristics systematically. It is also responsible for confirming whether the new generated URLs are indeed malicious by DNS queries and content matching in an automated fashion. The five heuristics came from URL lexical similarities and their own observations in the PhishTank database. They studied these five heuristics to generate new URLs from existing Phishing URLs in a current blacklist. The first heuristic is to replace the top-level domains (TLD). The second one is the IP address equivalence, which means that they clustered phishing URLs by close IP addresses. Then, they created new URLs by combining different hostnames and pathnames in the same clusters. The third heuristic is the directory structure similarity. They considered that similar pathes might attribute to similar sets of file names. Therefore, they grouped together directories with similar structures, and then exchanged the filenames among URLs within the same group. The fourth one is query string substitution. For similar URLs, they exchanged the queried content string which is after the question mark in an URL. The last heuristic is the brand name equivalence. Some URLs use the same URL structure but different brand names. To confirm their heuristics, the existence of those generated child URLs were tested by a verification process. By DNS lookups, the URLs which cannot be resolved will be filtered out. For resolved URLs, the model will establish connections to the corresponding servers first and then uses HTTP GET to retrieve content from it. If the status code is 200/202 (successful), a content similarity will be computed between the parent and child URLs [33].

The second component is an approximate matching algorithm, which performs an approximate match of a new URL against an existing blacklist. The algorithm performs the match by measuring the syntactic and semantic variations. To achieve this, the algorithm breaks the input URLs into four different entities - IP address, hostname, directory structure and brand name. Each individual entity will be scored and then a final score will be computed by combining all individual scores. If the final score exceeds a certain threshold, the URL will be marked as suspicious. This model also supports the live feed of data set. They showed that the system can keep the low false negative rate under 3%, and false positive rate under 5%. To evaluate their model, they concluded that this model is much faster than Google Safe Browsing even though it employs approximate matching [33].

Yadav et al. researched how to investigate alphanumeric unigrams and bigrams (two consecutive characters) to identify domains generated algorithmically [34]. They were motivated by the observation that auto-generated domains are quite different from legitimate ones in terms of the spelling or pronounceable features. Hence, they developed their detection metrics based on signal detection theory and statistical learning technique. Their method focused on detecting domains generated from pseudo-random string generation algorithms and dictionary-base generators which produce pronounceable words but not in the English dictionary. Following that, they implemented their model by two main parts. Firstly, they grouped DNS queries by Top Level Domain (TLD), IP-addresses they were mapped to, or the connected component they belonged to. Next, they used their metrics to characterize the distribution of the alphanumeric characters or bigrams. They also experimented with three metrics (KL-distance, Jaccard Index, and Edit-Distance) to distinguish a set of legitimate domain names from malicious ones. They also performed in-depth per-domain analysis, per-ip analysis, and component analysis to evaluate the performance. By comparison, they concluded that Jaccard Index metric performed the best, and then the Edit Distance measure, and finally the KL divergence. Lastly, they highlighted that their methodology can be used to detect unknown and unclassified botnets; therefore, it can be used as the first alert for Tier-1 ISP [34].

IV. Comparison of Different Techniques

In this section, we compare the fast-flux (FF) and domain-flux (DF) detection techniques separately against multiple criteria. As far as we know, DF is a more recent and advanced technique compared to FF. A large amount of works has been done to detect FF botnets; however, DF researches received researchers' attention until around 2008.

A. Fast Flux Research Evaluation

Here, we use the following 5 criteria to analyze the researches of detecting fast flux botnets.

- Realtime Detection
- Accuracy
- Distinguish FFSN VS. FFAN
- Speed
- Mining Based

1) Realtime Detection: Realtime detection is an important feature that a model can monitor a domain dynamically. Among [20] [15] [22] [23] [13], three works support realtime detection. The model developed by Caglayan et al. supports realtime detection within minutes. Their model detects and classifies FF Service Networks by both active and passive DNS monitoring [22]. Meanwhile, the methodology developed by Perdisci et al. can detect malicious FF service network in-the-wild [23]. In the model of Yu et al., specified domains are periodically queried by a detector and the results from two metrics (AOR and MAR) are calculated over time [13].

2) Accuracy: Accuracy is an extremely important criteria to evaluate a detection model. Any high false positive or negative rate will definitely limit the practical usage of a model. The method developed by Holz et al. reached a very high accuracy up to 99.98% with a standard deviation of 0.05% [20]. Meanwhile, the method developed by Perdisci et al. received a high accuracy as well. They abstracted 12 features indicating whether the target network was a benign FF Service network or FF attack network. If they employ all features, their model will get 0.3% false positive rate. If using passive features only, their model will have a 0.6% false positives. They also experimented that the system got 0.7% false positives even only when three features (TTL per domain, number of domains per network, IP growth ratio) were selected [23].

In contrary, the model developed in [22] only has a acceptable accuracy. Their classifier can predict a FF domain with a 97% confidence. However, the confidence can be slightly improved by more rounds of 10-minute sampling. Yu et al.'s model can detect and distinguish FF service network and FF attack network easily. But there are strong limitations to ensure the system accuracy. If only a small number of agents involved, the accuracy of the two metrics could be significantly reduced. They found that some FF Attack Networks were dormant. If the FFAN returns a few agents or only one agent has a long TTL, this will make the metrics invalid. Besides, DNS poisoning can mislead the metrics because the domains are resolved to wrong IP addresses by this attack [13].

3) Distinguish FFSN VS. FFAN: Fast fluxing has both legitimate uses, such as load balancing, and malicious uses, such as evading botnet detections. Both usages may show similar features; therefore, to detect botnet successfully, it is critical to distinguish these two kinds of networks. Otherwise, the false negative rates of botnet detections can be very high due to the assumption that all fast flux networks are malicious.

Among the five works, three methods can achieve this. Caglayan et al.'s model can distinguish from basic fast flux hosting, name server fluxing, and double fluxing by measuring by fluxing index. Their experiments show that hp.com and video.google.com (FF for load balancing) were not classified as fast flux attacks [22]. Perdisci et al. applied a statistically supervised learning to build a classifier to predict whether the target network was malicious or benign fast flux [23]. Yu et al. achieved this by observing the agent lifespan as botnet master cannot maintain a complete physical control to the bots [13].

TABLE I
COMPARISONS OF FF BOTNET RESEARCHES

Research Works	Realtime	Accuracy	FFSN VS. FFAN	Speed	Mining Based
Flux score [20]	NO	HIGH	NO	-	NO
Collaborative detection [15]	NO	-	NO	HIGH	NO
FF monitor [22]	YES	ACCEPTABLE	YES	-	YES
Recursive DNS [23]	YES	HIGH	YES	-	YES
Agent lifespan [13]	YES	-	YES	HIGH	NO

4) Speed: The operation speed in Zhou et al.'s model is relatively high due to their correlation scheme. Evidence from multiple domain name servers can improve the detection speed to a large extent [15]. In the mean time, Yu et al.'s model could be fast because it just monitors domains by simply querying those domains. This model is relatively easy to implement and deploy [13]. On the other hand, the method developed by Perdisci et al. was supervised learning based. To predict the FF network as benign or malicious, the model needs to extract relevant features. Therefore, the speed of their model depends on how many features are selected [23].

5) Mining Based: According to our survey, many works have employed data mining to make a decision on whether a fast flux (FF) network was benign or malicious. Two works we investigated relied on data mining to detect or distinguish fast flux domains. Perdisci et al. applied a unsupervised learning to cluster domains by a single-linkage hierarchical clustering algorithm, and then distinguished fast flux networks as benign or malicious by applying a C4.5 classifier [23]. Similarly, in Caglayan et al.'s model, a classifier was trained to predict the FF domain with a confidence [22].

6) Wrap Up: Table I shows the features we compared for fast flux detection researches. If any feature was not discussed or unclear in a paper, we mark a dash sign in that field. According to our investigation, supporting realtime detection, high accuracy, distinguishing between FFSN and FFAN, and high speed are some of the most important features. As shown, many botnet fast flux researches have applied data mining into their model. Feature selection is one critical part to cluster or predict the FF network. Furthermore, how many and what features they select will have a significant influence on the accuracy and speed of detection.

B. Compare Domain Flux Techniques

In this part, the following 4 criteria has been used to evaluate those domain flux (DF) researches.

- Accuracy
- Speed
- Passive or Active Detection
- Mining Based

1) Accuracy: The importance of detection accuracy has been explained in Section IV A2. Among 6 researches we investigated, [27] [28] achieved highest accuracies. The second work [27] of Ma et al. extended the idea from their previous one [26] and developed an online learning model to predict the

TABLE II
COMPARISONS OF DOMAIN FLUX BOTNET RESEARCHES

Research Works	Accuracy	Speed	Passive or Active	Mining Based
Take-over [19]	HIGH	-	ACTIVE	NO
Batch method [26]	MODEST	DEPENDS	PASSIVE	YES
Online method [27]	HIGH	DEPENDS	PASSIVE	YES
DNS failure graph [28]	HIGH	-	PASSIVE	NO
Phishnet [33]	DEPENDS	HIGH	PASSIVE	YES
Unigram & bigram [34]	DEPENDS	DEPENDS	PASSIVE	YES

suspicious URLs. Their online model can obtain data feed over time to make their feature selections up to date. Therefore, the accuracy reached 99% compared to the accuracy 95% to 99% of previous one. Jiang et al. captured the unproductive DNS queries to generate the failed DNS graph. Their model can identify trojan (backdoor), spamming, and domain flux botnets with 100% accuracy [28]. Stone-Gross et al. contributed to improving the accuracy of botnet size detection by counting the node identifier N_{id} [19]. The model developed by Prakash et al. had a false negative rate under 3% and false positive under 5%. The actual accuracy rate depends on the final score threshold selection. Higher threshold will lead to increased false negatives, and lower threshold will increase the false positives [33]. Similarly, the accuracy of Yadav et al.'s model depends on which similarity measure will be selected and how many words will be in the test dataset [34].

2) Speed: Ma et al.'s two works need to extract large number of features. To reach a high accuracy, speed could be slow due to the processing of those features [26] [27]. PhishNet developed in [33] has a fast speed which is around 80 times than Google Safe Browsing. In [34], Jaccard Index measure can get the highest accuracy to detect large amount of domain names, but the speed is quite slow. The model faces a dilemma of judging between accuracy and speed.

3) Passive or Active Detection: The method of Stone-Gross et al. is an active approach to seize control of a DF botnet - Torpig. They pre-registered the domains that bots would contact with and then set up an Apache server to receive and log bots requests, and record all botnet traffic [19]. On the other hand, all other works [26] [27] [28] [33] [34] applied passive approaches to detect botnets in a lightweight fashion.

4) Mining Based: The two approaches of Ma et al. are similar by extracting lexical and host-based features from URLs for predicting. They applied supervised machine learning to detect and prevent users from visiting malicious websites based on automated URL classification [26] [27]. Meanwhile, Prakash et al. developed 5 heuristics, which can be used to train the model for predicting new malicious URLs [33]. The model in [34] can extract and learn features of unigram and bigrams so that it can predict whether a URL is algorithmically generated or not.

5) Wrap Up: Table II shows some features we compared in the papers we investigated. The detection accuracies of auto-generated domains are very high although some of them are conditional. Only in [33], the detection speed was very high. For almost all mining-based approaches, the speed was conditional depending on the features selected, or measures they applied. Moreover, researchers tend to apply data mining to detect domain names generated by automatic algorithms and most of researches used passive approaches to analyze by a lightweight means.

V. FLUXING MITIGATION

In this section, we surveyed what strategies can be used to mitigate fluxing botnets.

Fast flux (FF) attacks can be mitigated by collaborating with several parties, according to [20]. Their metric can apply to automatically identify the FF domains, which have been recorded in a domain blacklist. Registrars can be notified by the blacklist to shutdown the FF domains, and ISPs can use the blacklist to filter DNS queries for those FF domains. ISP can do something further by monitoring both incoming and outgoing flows. Thus, flux-agent can be located. Then, ISP can inject requests from that agent to the control node (mothership). By capturing the corresponding response, the location of the "mothership" can be identified [35]. Stone-Gross et al. registered domains which could be used by the Torpig and set up an Apache server to seize control [19]. This can be regarded as a way to mitigate Fluxing techniques as well.

Unfortunately, few works have been done to research into fluxing mitigation. As far as we know, blacklisting-related method is almost the only way to mitigate botnets with fluxing features. The deployment of fluxing botnet mitigation requires the collaboration of both registrars and ISPs.

VI. FUTURE DIRECTIONS

First of all, data mining [36] can be used widely to extract features of individual botnets. As we know, it is possible that one compromised computer may host a number of different kinds of bots. As a result, the observed DNS failures are the mixing of all the bots. Therefore, data ming technology can be used to differentiate them by the classification techniques.

Secondly, hidden Markov model [37] [38] can be hired to infer the internal state transferring, and further identify the botnet writers' URL generation algorithms. With an accurate algorithm in hands, we will be in a better position to beat botnet owners. The traditional method is re-engineering based on catched bots, but this is passive and time consuming.

Thirdly, new techniques can be applied to help us to understand botnet structure. We can see a botnet as a special complex network, it is unknown to defenders. New theories or skills, e.g. graph spectra [39], can be employed to study botnets.

Fouthly, it is critical to clear remote computers even we know they are compromised. This is a tough challenge, because the owners of the compromised computers can not trust the message or software tools that offered by a third party.

Finally, botnet writers may have developed new strategies against our detection methods. It is useful that we predict their possible strategies.

VII. Summary

In this survey, readers can gain a deep understanding of the necessity for investigating fluxing features of botnet. Our work can help researchers to understand how botnet writers can employ FF and DF features to evade detection. Research works we investigated have both advantages and disadvantages, thus our multiple evaluation criteria provides a clear perspective of their features. Our main aim is also to raise more awareness from botnet researchers to develop more efficient models. Finally, our future work provides a brief view that what can be done in future to fight against botnets.

References

[1] H. Zeidanloo, M. Shooshtari, P. Amoli, M. Safari, and M. Zamani, "A taxonomy of botnet detection techniques," in *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, vol. 2, july 2010, pp. 158 –162.

[2] A. Ramachandran, N. Feamster, and G. Tech, "Understanding the network-level behavior of spammers," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications (2006)*, 2006, pp. 291–302.

[3] D. Moore, G. Voelker, and S. Savage, "Inferring internet denial-of-service activity," in *In Proceedings of the 10th Usenix Security Symposium*, 2001, pp. 9–22.

[4] S. Saroiu, S. D. Gribble, and H. M. Levy, "Measurement and analysis of spyware in a university environment." in *NSDI*. USENIX, 2004, pp. 141–153.

[5] T. Holz, M. Engelberth, and F. C. Freiling, "Learning more about the underground economy: A case-study of keyloggers and dropzones." in *ESORICS*, M. Backes and P. Ning, Eds., vol. 5789. Springer, 2009, pp. 1–18.

[6] Symantec, "Cutwails bounce-back; instant messages can lead to instant malware," http://www.messagelabs.com/mlireport, 2009.

[7] P. Bacher, T. Holz, M. Kotter, and G. Wicherski, "Know your enemy: Tracking botnets," http://www.honeynet.org/papers/bots, 2008.

[8] Y. Tang and S. Chen, "Defending against internet worms: a signature-based approach," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 2, march 2005, pp. 1384 – 1394 vol. 2.

[9] C. Li, W. Jiang, and X. Zou, "Botnet: Survey and case study," in *Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on*, dec. 2009, pp. 1184 –1187.

[10] J. R. Binkley and S. Singh, "An algorithm for anomaly-based botnet detection," in *Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet - Volume 2*. USENIX Association, 2006, pp. 7–7.

[11] 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*. USENIX Association, 2007, pp. 7–7.

[12] M. Feily, A. Shahrestani, and S. Ramadass, "A survey of botnet and botnet detection," in *Emerging Security Information, Systems and Technologies, 2009. SECURWARE '09. Third International Conference on*, june 2009, pp. 268 –273.

[13] S. Yu, S. Zhou, and S. Wang, "Fast-flux attack network identification based on agent lifespan," in *Wireless Communications, Networking and Information Security (WCNIS), 2010 IEEE International Conference on*, june 2010, pp. 658 –662.

[14] Honeynet, "How fast-flux service network work," http://www.honeynet.org/node/132, 2008.

[15] C. V. Zhou, C. Leckie, and S. Karunasekera, "Collaborative detection of fast flux phishing domains." *JNW*, vol. 4, no. 1, pp. 75–84, 2009.

[16] P. Amini, "Kraken botnet infiltration," http://dvlabs.tippingpoint.com/blog/2008/04/28/kraken-botnet-infiltration, 2008.

[17] P. Porras, H. Saidi, and V. Yegneswaran, "A foray into conficker's logic and rendezvous points," in *In LEET'09: Proceedings of the 2nd USENIX conference on Large-scale exploits and emergent threats (2009)*, 2009.

[18] J. Wolf, "Technical details of srizbi's domain generation algorithm," http://blog.fireeye.com/research/2008/11/technical-details-of-srizbis-domain-generation-algorithm.html, 2008.

[19] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. A. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: analysis of a botnet takeover." in *ACM Conference on Computer and Communications Security*. ACM, 2009, pp. 635–647.

[20] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, "Measuring and detecting fast-flux service networks," in *NDSS*, 2008.

[21] C. V. Zhou, C. Leckie, S. Karunasekera, and T. Peng, "A Self-healing, Self-protecting, Collaborative Intrusion Detection Architecture to Trace-back Fast-flux Phishing Domains," in *Proceedings of the 2nd IEEE Workshop on Autonomic Communication and Network Management*, Apr. 2008.

[22] A. Caglayan, M. Toothaker, D. Drapeau, D. Burke, and G. Eaton, "Real-time detection of fast flux service networks," in *Conference For Homeland Security, 2009. CATCH '09.*, 2009, pp. 285 –292.

[23] R. Perdisci, I. Corona, D. Dagon, and W. Lee, "Detecting malicious flux service networks through passive analysis of recursive dns traces," in *Computer Security Applications Conference, 2009. ACSAC '09. Annual*, 2009, pp. 311 –320.

[24] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "My botnet is bigger than yours (maybe, better than yours): why size estimates remain challenging," in *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*. USENIX Association, 2007.

[25] C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, and S. Savage, "The heisenbot uncertainty problem: Challenges in separating bots from chaff." in *LEET*. USENIX Association, 2008.

[26] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: learning to detect malicious web sites from suspicious urls," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 1245–1254.

[27] S. S. Justin Ma, Lawrence Saul and G. Voelker, "Identifying suspicious urls: An application of large-scale online learning," in *In Proc. of the International Conference on Machine Learning (ICML)*, 2009.

[28] N. Jiang, J. Cao, Y. Jin, L. Li, and Z.-L. Zhang, "Identifying suspicious activities through dns failure graph analysis," in *Network Protocols (ICNP), 2010 18th IEEE International Conference on*, oct. 2010, pp. 144 –153.

[29] V. Pappas, D. Wessels, D. Massey, S. Lu, A. Terzis, and L. Zhang, "Impact of configuration errors on dns robustness," *Selected Areas in Communications, IEEE Journal*, vol. 27, pp. 275 –290, 2009.

[30] Z. Zhu, V. Yegneswaran, and Y. Chen, "Using failure information analysis to detect enterprise zombies." in *SecureComm*, vol. 19. Springer, 2009, pp. 185–206.

[31] D. Plonka and P. Barford, "Context-aware clustering of dns query traffic," in *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*. ACM, 2008, pp. 217–230.

[32] Y. Jin, E. Sharafuddin, and Z. L. Zhang, "Unveiling core network-wide communication patterns through application traffic activity graph decomposition," in *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems*. ACM, 2009, pp. 49–60.

[33] P. Prakash, M. Kumar, R. Kompella, and M. Gupta, "Phishnet: Predictive blacklisting to detect phishing attacks," in *INFOCOM, 2010 Proceedings IEEE*, 2010, pp. 1 –5.

[34] S. Yadav, A. K. K. Reddy, A. N. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in *Proceedings of the 10th annual conference on Internet measurement*. ACM, 2010, pp. 48–61.

[35] T. H. Project, "Know your enemy: Fast-flux service networks," http://www.honeynet.org/papers/ff/, 2007.

[36] J. Han, *Data Mining: Concepts and Techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005.

[37] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ASSP Magazine, IEEE*, pp. 4 – 16, 1986.

[38] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, pp. 257–286, 1989.

[39] P. Van Mieghem, *Graph Spectra for Complex Networks*. Cambridge press, 2011.