Good Guys vs. Bot Guise: Mimicry Attacks Against Fast-Flux Detection Systems†

Matthew Knysz, Xin Hu, Kang G. Shin
The University of Michigan, Ann Arbor, MI 48109-2121, USA

***Abstract*—In this paper, we explore the escalating "arms race" between fast-flux (FF) botnet detectors and the botmasters' effort to subvert them, and investigate several novel mimicry-attack techniques that allow botmasters to avoid detection. We first analyze the state-of-art FF detectors and their effectiveness against the current botnet threat, demonstrating how botmasters can—with their current resources—thwart detection strategies. Based on the realistic assumptions inferred from empirically observed trends, we create formal models for bot decay, online availability, DNS-advertisement strategies and performance, allowing us to demonstrate the effectiveness of different mimicry attacks and evaluate their effects on the overall online availability and capacity of botnets.**

I. Introduction

A botnet is a vast collection of compromised computers under the control of a botmaster utilizing a Command-and-Control (C&C) infrastructure. Among the numerous criminal uses of botnets, one of the more advantageous is the botnet-based hosting service, which proxies or redirects unsuspecting users to illegal or nefarious content. This strategy grants criminals a high level of anonymity while enabling easy and centralized management of the malicious content. However, because botnets are composed of thousands of disparate compromised systems from around the globe, all with varying resources and network connectivity, it is not uncommon for them to unpredictably go offline, disrupting the availability of their nefarious service/content. As a result, botmasters adopt fast-flux (FF) DNS techniques, which frequently change the offline bots' domain-name mappings to different bots' IP addresses. This increases the probability that the malicious domain resolves to an online bot's IP and provides highly available and reliable content-hosting services despite frequent node failures/disconnectivity.

While their tremendous success as service networks has spurred the development of novel detection strategies, FF botnets remain a persistent threat. The advent of fast, reliable FF detection systems has not yet eradicated FF botnets; rather, it coaxed them to evolve, developing more robust, efficient, and stealthy mechanisms for subverting detection. This cycle continues, with defenders and botmasters caught in an ever-escalating "arms race". Unfortunately for the good guys, bots are free, easy to come by, capable of granting significant amounts of coordinated processing power, and incredibly effective sources of revenue.

† The work reported in this paper was supported in part by the Office of Naval Research under Grant N00014-09-11042

During our global monitoring of FF botnet domains, we have observed that despite the detection mechanism or mitigation strategy imposed, botnets constantly evolve methods for subverting them, growing into ever more formidable systems—in many ways resembling enterprise-level Content Delivery Networks (CDNs). While efficient when first introduced, many FF detection systems quickly become outdated; they are designed to detect the current advertising strategies of FF botnets, which are all too easily and quickly adapted to avoid detection. Thus, it is not sufficient to base FF detection on the current class of differentiating features. Instead, improvements could be made if the botnets' limitations were also taken into consideration. While previous research has focused on identifying behavioral features uniquely intrinsic to FF botnets for detection, we have decided to take an alternate approach; assuming the role of the enemy, we explored botnets' mimicry capabilities and limitations in evading detection. Analyzing the resources currently available to FF botnets, we developed models for their bot decay, online availability, DNS-management strategies, and performance. Using these models, we examined the potential success of novel mimicry attacks against state-of-art FF detection systems, demonstrating that such attacks are easily within the capacity of current botnets. By better understanding the current arsenal botnets have at their disposal and how it can be—and is being—applied to defeat current detection strategies, we hope to foster improvements to existing systems, as well as yield new insight into the mimicry limitations of FF botnets.

II. Background

In this section, we investigate the DNS IP-advertisement patterns of current FF botnet and benign domain types. First, we will describe how we set up a globally distributed DNS monitoring system. Then, we will discuss different types of domains and their unique features discovered through over 4 months of monitoring. Because most existing detection systems rely on DNS features to detect FF botnets, our unique, global perspective of IP advertising strategies helps us gain insight into the current state of FF domains and their ability to successfully evade detection via mimicry attacks.

A. Global DNS-Monitoring System

We created a distributed DNS-query engine called *DIGGER*, deployed on 312 geographically disparate nodes in the PlanetLab testbed [10]. The nodes were chosen based on the location of the DNS servers they queried, such that DIGGER would

Continent	N. America	Europe	Asia	S. America	Oceania	TOTAL
DIGGER Nodes	143	94	46	19	10	312
% of TOTAL	45.83%	30.13%	14.74%	6.09%	3.21%	

TABLE I: Global distribution of DIGGER nodes by continent

issue queries to DNS servers in different geographic locations around the world. Table I shows the continental distribution of DIGGER nodes, which is reflective of the overall distribution of available PlanetLab nodes.

DIGGER was deployed for over 4 months in early 2010 and gathered global DNS-query results for domains compiled from multiple sources, including online repositories of phishing and malware websites as well as the top 1000 most popular domains. On each node, for malicious and benign domains, DIGGER performed DNS queries on their A (address) records, NS (authoritative name server) records, NA records (A records on name servers) and the *reverse DNS* (rDNS) lookup (i.e., PTR records) for the A and NA record IPs. DIGGER continued to dig active domains periodically based on their observed TTL. Domains determined to be offline[1] were dug every 24 hours to discover if they came back online. For each domain, DIGGER also collected connectivity information (used to derive a bot online-decay model) on both A and NA record IPs by attempting to establish TCP connections on ports 80 and 53[2]. By applying simple heuristics on the aggregated data, we manually identified and verified 45 FF domains by looking for IP addresses with rDNS names indicative of compromised computers (e.g., dynamic, dialup).

B. Domain Types

1) Fast-Flux Domains: FF domains are malicious domains utilizing a FF DNS-advertisement strategy, typically built atop botnets and used for scams where the potential profits depend on the availability of the hosted services/content. To counter the unreliable connectivity of the bots hosting the malicious services/content, botmasters adopt FF DNS techniques and advertise numerous IPs in their DNS-query results with frequently-changing mappings between the domain name and different bots' IP addresses.

Figure 1-a illustrates the global IP usage—across all 312 DIGGER nodes—for an example FF domain. In the figure, the *Time* axis represents the time (in seconds) since DIGGER started monitoring the domain; *Node Index* represents the DIGGER node that the IP was observed on, with positive values indicating an A record IP and negative values an NA record IP; *IP Index* is a unique index incrementally assigned to each newly observed IP. From the figure, we can see that the FF domain slowly and nearly continuously advertises new, unique IPs over its entire online lifetime. Over the 4-month monitoring period, we have observed that a typical FF domain usually advertises thousands of unique IP addresses, with the most aggressive botnets advertising over 35,000. As we will demonstrate later, this huge IP pool affords the botmaster great flexibility and abundant resources to mimic a wide range of benign DNS behaviors for evading detection.

2) Benign Domains: CDN domains are benign domains that use a Content Delivery Network (CDN), such as Akamai, to improve the delivery of their content. CDNs—consisting of a system of computers networked together for the purpose of improving the performance and scalability of content distribution—produce DNS-query results resembling those of malicious FF domains: numerous, changing IPs per query with short TTL values. For instance, *nfl.com*, a CDN domain shown in Fig. 1-b, has a short TTL (20 seconds) and constantly changes its A record IPs, resulting in the accumulation of almost 1200 IP addresses over all DIGGER nodes during our monitoring period. This affinity between CDN and FF domains is a consequence of their similar goal to provide reliable content delivery despite node failures, as well as their shared assumption that any node can temporarily or permanently fail at any time. Consequently, it is possible for botmasters to cloak their malicious DNS-advertisement strategy as normal, benign CDN behavior. However, a CDN's DNS-advertising profile depends on whether a location-aware[3] or load-balancing[4] strategy dominates and, thus, can be highly dependent on the DNS server monitored. For example, most of the servers for *nfl.com* reside in N. America and Europe, with a small amount in Asia. As a result, in N. America and Europe, location-aware techniques dominate, with DNS queries consistently returning a small set of IPs belonging to the nearest servers. On the other hand, DNS queries in continents such as S. America, where *nfl.com* has no servers, are influenced far less by location, and load-balancing techniques dominate; servers from all over N. America, Europe, and Asia are advertised based on their current load, resulting in hundreds of IPs observed per S. American DIGGER node over the monitoring period.

Non-CDN domains are benign domains that do not use a CDN for delivery of their content. Typically, non-CDN domains use a few stable content servers and a modest number of name servers. Some popular non-CDN domains may advertise more than 18 IPs in a single DNS query, using the same set of IPs in each query and rotating the order across queries for load-balancing purposes. This type of DNS strategy is often referred to as *round robin DNS*.

III. Fast-Detection Systems (2 queries)

Given the serious threats posed by FF botnets, researchers have proposed various detection systems. In this section, we first analyze early fast detection systems ("good guys") and then propose mimicry strategies botmasters can use for evasion ("bad guise"). Advanced detectors and mimicry attacks against them will be described in Section IV.

A. Good Guys

The original FF detection system proposed by Holz *et al.* [2] (i.e., Holz detector) shown in Eq. (1) and RB-Seeker's first-

[1] a domain is offline if its DNS query returns no A record

[2] although DNS primarily uses UDP protocol to serve requests, DNS servers also accept TCP connections on port 53 in order to support response data exceeding 512 bytes or for tasks such as zone transfer [6]

[3] advertising IPs geographically near DNS servers to reduce transmission overhead due to distance

[4] advertising the IPs of servers with lower load to increase performance

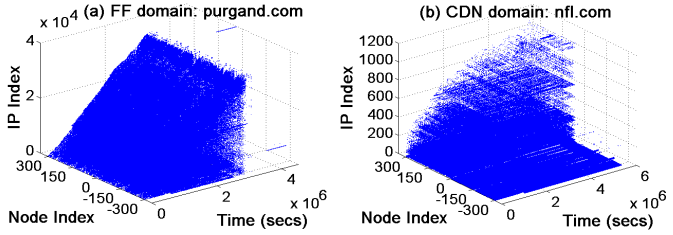


Fig. 1: Global DNS-query results for Fast-Flux and CDN domains

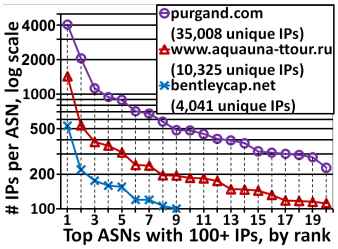


Fig. 2: IP distribution for top 20 ASNs

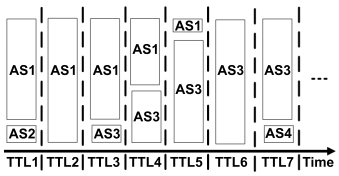


Fig. 3: ASN-mimicry strategy (2 DNS queries)

tier detector [3] shown in Eq. (2) are considered fast-detection systems, as they are capable of detecting FF domains with high accuracy from only 2 DNS queries. This is achieved through the use of a linear decision function containing weighted terms derived from the DNS queries and a bias term.

$$f(x) = 1.32 \cdot n_A + 18.54 \cdot n_{ASN} - 142.38 \quad (1)$$

$$f(x) = -1.257 \cdot N_{unique\_IPs} - 26.401 \cdot N_{ASN} - 13.024 \cdot N_{DNS\_bad\_words} + 162.851 \quad (2)$$

In Eq. (1), the number of unique A records and Autonomous System Numbers (ASNs) are represented by n_A and n_{ASN}, respectively. In Eq. (2), N_{unique_IPs} represents the number of unique IPs seen in the A records, N_{ASN} the unique ASNs, and $N_{DNS\_bad\_words}$ the number of rDNS lookups containing "bad words" indicative of compromised home computers, such as comcast, dynamic, dialup, etc. In both equations, the magnitude of $f(x)$ represents the degree of confidence when classifying domain x, with positive values indicating a FF domain for Eq. (1) and a benign domain for Eq. (2).

We implemented both detectors and applied them to our manually verified set of 45 FF domains to determine how they would fare against today's FF botnet threat. They identified 6 (Holz detector) and 12 (RB-Seeker) FF domains respectively, resulting in 86.7% and 73.3% false-negative rates and demonstrating the extent to which botnets have evolved. Both papers realize their weights and thresholds used for detection must be periodically re-trained to counter future mimicry attacks. However, doing so will increase the false-positive rate, causing benign domains—whose DNS activity is being mimicked by FF domains—to be misclassified as malicious.

B. Bot Guise

1) ASN-Mimicry Attack: From Eqs. (1) and (2), we can see that the dominant factor in identifying FF domains is the number of unique ASNs. Clearly, an effective mimicry attack against these fast-detection systems should reduce the number of ASNs to levels seen for benign domains. Since DNS queries on benign domains often contain A record IPs from 2 ASNs (e.g., *www.avast.com*), let us assume that a fast-detection system adopts the following overly strict policy: *over 2 DNS queries, any domain containing IPs from more than 2 ASNs will be flagged as malicious.* While this policy can result in false-positives for benign domains, such as some CDNs, if it can be effectively subverted, so can more lenient constraints.

To discover if this is feasible with current botnet resources, we aggregated the IPs for each FF domain globally monitored by DIGGER, determined their ASNs, and then analyzed their IP distribution across ASNs. We found that, despite the size of the botnet, the distribution was long-tailed, with at least one ASN containing a disproportionately large number of IPs. This is possibly due to certain ASNs (e.g., ISP networks) containing a large proportion of vulnerable computers, or from botmasters targeting certain institutions. Fig. 2 plots this trend for 3 representative FF domains of varying sizes; to keep the graph readable, we have only plotted the distribution for the top 20 ASNs from which the botnets have 100+ IPs.

Assuming botnets contain a suitable number of IPs from at least a single ASN (as our data indicates), there is a simple IP-advertisement strategy for mimicking the ASN behavior of benign domains. This mimicry strategy is demonstrated in Fig. 3, with each TTL (i.e., fresh DNS query) showing the distribution of IPs from various ASNs. For example, assume the botnet controls a large number of IPs from AS1 and a moderate to small number of IPs from AS2. At TTL1, the majority of advertised IPs are from AS1 with a smaller subset from AS2. While there exists a sufficiently large pool of online IPs from AS1, this is not the case with AS2, eventually requiring the introduction of IPs from a different ASN. However, because the detection window is 2 DNS queries, the botmaster must ensure that all the IPs seen over 2 consecutive queries belong to no more than 2 ASNs. Thus, before IPs from a new ASN can be introduced, she must first advertise only IPs belonging to one of the ASNs present in the previous TTL, as shown in TTL2. Then, at TTL3, she is free to utilize IPs from the new ASN, AS3. If she happens to control a large number of IPs in AS3, she can slowly replace AS1 as the dominant ASN, as shown in TTL3–TTL6. In this way, botmasters can successfully mask their use of numerous ASNs from fast-detection systems.

2) rDNS-Mimicry Attack: From Eq. (2), we see that the second most influential term when identifying FF domains is $N_{DNS\_bad\_words}$. However, RB-Seeker asserts that a rDNS lookup on an IP will not always return a result, although when it does, it can be useful. Despite its inconsistency, the term is still an order-of-magnitude more important than the number of unique IPs. Therefore, an effective mimicry attack should include a mechanism for subverting this detection metric.

Let us assume the following aggressive detection policy: *over 2 DNS queries, any domain with more than 2 "bad words" in its rDNS results will be flagged as malicious.* Certainly, this policy is overkill, as many legitimate domains (e.g., *www.comcast.com*) will have rDNS results that contain "bad words". However, if botnets can defeat this harsh

limitation, more realistic thresholds can also be subverted. If current FF botnets contain enough IPs without rDNS results (i.e., rDNS=NONE IPs), then a mimicry strategy similar to that proposed for ASNs in Section III-B1 could be applied. To determine the feasibility of this approach, we aggregated the IPs for each FF domain monitored globally by DIGGER and determined the percentage of rDNS=NONE IPs. We discovered that for each FF domain, at least 15% of its total IPs lacked a rDNS result. Furthermore, for ≈24% of the FF domains, over 50% of their IPs lacked a rDNS result. Considering the large proportion of rDNS=NONE IPs and the fact that rDNS results for bots that aren't compromised home computers will be free of "bad words," the mimicry strategy proposed earlier for ASNs can easily be applied: IPs without rDNS results (or without "bad words") can be used in conjunction with IPs containing "bad words", such that only 2 "bad words" are observed over 2 queries. Thus, we have confirmed the unreliable nature of rDNS, demonstrating that it can easily be subverted if used as a detection metric.

However, to be truly effective, we need to ensure that this strategy can be combined with the previous ASN-mimicry attack. Thus, for each FF domain, we analyzed the distribution of rDNS=NONE IPs across ASNs, once again observing the long-tailed distribution. This phenomenon is shown in Fig. 4 for 3 representative domains of varying sizes. The majority of botnets we observed still possessed enough IP-dense ASNs to sufficiently mount the dual mimicry attacks.

3) IP Mimicry: Having determined that current botnet resources are capable of instigating ASN- and rDNS-mimicry attacks, we turn our attention to the final attribute utilized by the fast-detection systems in Eqs. (1) and (2), the number unique IPs. It stands to reason that the more IPs a FF domain advertises per query, the more likely some of the bots will be online. Furthermore, because most DNS servers perform round-robin scheduling within a given TTL, advertising more IPs per query decreases the load imparted on each bot, thereby increasing the botnet's total service capacity and its revenue. Since benign non-CDN domains may also advertise a large number of stable IPs (e.g., *hostingprod.com* uses 18 IPs per DNS query), FF domains are afforded a fair amount of freedom in the number of bots they can advertise; this is supported by Eqs. (1) and (2), where the number of unique IPs is the least influential detection feature. However, non-CDN domains advertise the same set of IPs for every TTL, causing their total unique IPs to remain bounded and facilitating the use of a maximum IP threshold, N_{thresh}, for detection.

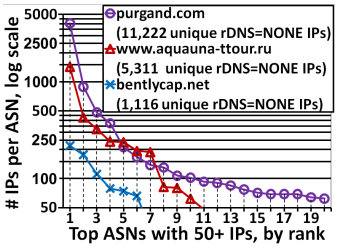


Fig. 4: rDNS=NONE IP distribution for top 20 ASNs

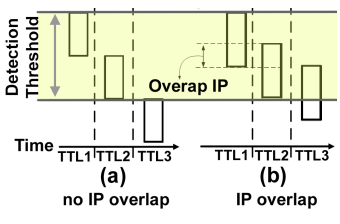


Fig. 5: DNS IP-advertising strategies

When performing an IP-mimicry attack, there are two basic strategies for keeping the total number of IPs over two TTLs below N_{thresh}. The first, shown in Fig. 5-a, has no IP overlap, with the botnet advertising a completely new set of IPs every TTL. The alternate strategy, shown in Fig. 5-b, has IP overlap, with some of the IPs being advertised for multiple TTLs. Each strategy has certain pros and cons. Having no IP overlap allows for the rapid replacement of offline IPs; however, as can be seen from Fig. 5, this reduces the number of IPs that can be used for any given TTL, which, in turn, decreases the botnet's service capacity. On the other hand, with an increase in IP overlap, more IPs can be advertised per TTL, decreasing the load per bot (from Fig. 5, we can see that the total number of IPs advertised per query is equal to the number of overlapped and new IPs, i.e., $N = N_{overlap} + N_{new}$); however, this reduces the rate at which offline IPs can be replaced, resulting in a greater proportion of dead bots and failed victim connections. Considering bots' unreliable connectivity, finding the optimal IP-advertisement strategy for FF domains requires a better understanding of the underlying bots' online availability (i.e., at any given time, what is the probability of bots being online).

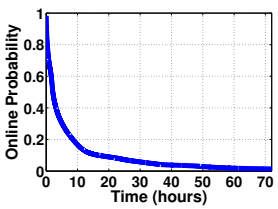


Fig. 6: Bot online-decay model (first 72 hours)

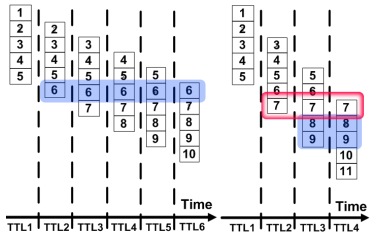


Fig. 7: Persistence of overlapped IPs

4) Bot Online-Decay Model: We developed a bot online-decay model, $P_{online}(t)$, to predict the probability a bot will be online after time t. In building the model, we aggregated all the bot IPs seen for FF domains globally monitored by DIGGER, recording the time they were observed and if they were online and reachable at that time (i.e., a connection could be established). Notice that while each *individual* DIGGER node monitors the bot IPs at the granularity of the domain's TTL, they are not synchronized and do so *independently* and at different times due to competing PlanetLab workloads. Furthermore, many botnets are used for multiple online scams; thus, many of the same bot IPs will be observed in queries on different FF domains.[5] Therefore, by combining all available data points for each bot IP—regardless of the DIGGER node's location or the FF domain it was observed for—we can build a fairly complete picture of the online times of bots currently used by FF domains. If an IP is not seen by any DIGGER node for over 12 hours, we assume that it has gone offline during that time. The resulting bot online-decay model has a long-tailed distribution. In Fig. 6, which plots the first 72 hours of this model, the y-axis represents the probability that a bot

[5]over 11% of all FF IPs were advertised by ≈22.2% or more of the monitored FF domains; over 51% of all FF IPs were advertised by ≈6.7% or more FF domains; of the less than 34% of FF IPs advertised by only a single FF domain, over 15.7% were either only seen once or dead (i.e., never reachable) over our entire monitoring period

is continuously online for more than some time t, represented by the x-axis. From the plot, it is clear that the probability of a bot being online decays exponentially with time, such that, after a day, there is less than a 10% chance it's still online. These findings reassert the notion that a bot's connectivity is highly unreliable, resulting from the varied usage patterns of the compromised computers' owners.

5) Performance Model: Using our online-decay model, we can determine the optimal IP-mimicry strategy in terms of performance, which we evaluate based on the number of incoming connections per unit time the botnet can handle. This metric is chosen because the revenue of FF botnets comes from victims visiting the hosted content, and therefore, botmasters want to maximize the number of connections the botnet can support. If the mimicry attack drastically reduces this amount, then the bots will become overwhelmed, resulting in dropped connections and decreased revenue. We assume both the inter-arrival time of victim connections and the bots' service time are Poisson processes with Markovian (i.e., exponential and memoryless) distributions with λ or μ representing the incoming connection rate and average service times, respectively. Within a given TTL, most DNS servers perform round-robin scheduling when responding to DNS-queries. As a result, incoming victim connections will be evenly dispersed among the *online* bots advertised for that TTL. Therefore at each TTL, FF botnets can be modeled as N_{online} parallel identical M/M/1/K queues [1], where K is the online bots' queue length, (i.e., the maximum connections each can queue before dropping additional connections). Applying queueing theory[1] to this model, we can calculate the *connection loss probability* (i.e., the probability that an online bot will drop connections due to a full queue) as:

$$P_{loss} = \begin{cases} \frac{\rho^K - \rho^{K+1}}{1-\rho^{K+1}} & : \quad \rho \neq 1 \\ \frac{1}{K+1} & : \quad \rho = 1 \end{cases} \quad \text{where } \rho = \frac{\lambda}{N_{online} \cdot \mu}$$

Because we assume that each online bot is identical, an individual bot's P_{loss} is equivalent to that of the entire botnet, allowing us to compare the various IP-mimicry attacks' performance; a higher probability of dropped connections results in fewer exploitable victims and decreased revenues.

6) DNS-Strategy Model: To successfully apply the performance model, we must first establish a formal relationship between an IP-mimicry attack's DNS-advertisement strategy and our online-decay model, $P_{online}(t)$, so that we can estimate the potential number of online IPs, $N_{online}(t)$, during a given TTL. This relationship is straightforward when there is no IP overlap, as in Fig. 5-a. Since each TTL contains a fresh set of IPs under this strategy (i.e., $N = N_{new}$), they can only decay for the time, t, that has elapsed in the current TTL; thus, for a max TTL of T_{ttl} seconds, $N_{online}(t) = N \cdot P_{online}(t)$, where $0 \leq t < T_{ttl}$.

Determining $N_{online}(t)$ becomes more complicated for a strategy utilizing IP overlap, as in Fig. 5-b. Because IPs are persistent for multiple TTLs, they suffer an increased probability of going offline. For modeling purposes, we must rely on the reasonable assumption that older IPs—being more likely to be offline—will always be replaced before newer IPs. Additionally, to best distribute load among their bots, we can assume that botmasters will choose to advertise as many IPs as possible without exceeding the detection threshold N_{thresh}. These two assumptions imply an optimal replacement strategy for botmasters, from which we can deduce the following intrinsic properties: in any given TTL, (1) there exist a total of N_{new} IPs also present in the previous $0, 1, 2, \ldots, \lfloor \frac{N}{N_{new}} \rfloor - 1$ TTLs, and (2) there exist a total of $(N \bmod N_{new})$ IPs also present in the previous $\lfloor \frac{N}{N_{new}} \rfloor$ TTLs. The effect of these properties can be seen in Fig. 7 for two examples. Thus, for any given DNS-query, we can formulate formulate $N_{online}(t)$ in terms of $P_{online}(t)$ as:

$$\begin{aligned} N_{online}(t) \quad = \quad & (N \bmod N_{new}) \cdot P_{online}(t + \lfloor \frac{N}{N_{new}} \rfloor \cdot T_{ttl}) \\ & + \sum_{n=0}^{\lfloor \frac{N}{N_{new}} \rfloor - 1} N_{new} \cdot P_{online}(t + n \cdot T_{ttl}) \end{aligned} \tag{3}$$

Having defined $N_{online}(t)$ in terms of the IP-advertisement strategy, we can use it in our definition of $P_{loss}(t)$:

$$P_{loss}(t) = \frac{\rho(t)^K - \rho(t)^{K+1}}{1-\rho(t)^{K+1}} \qquad \text{where } \rho(t) = \frac{\lambda}{N_{online}(t) \cdot \mu} \tag{4}$$

7) Evaluation of current FF botnet strategies: Before evaluating the performance of our mimicry strategies, we establish a basis for current FF botnet performance. We examine the 3 FF domains shown in Table II using our online-decay model and Eq. (4). We first compare these various strategies by applying $\overline{N}$ and $\overline{N}_{new}$ to Eq. (3), finding $\overline{N}_{online}$ when the system reaches a steady state, shown in Table II[6]. From the results, botmasters appear quite adept at configuring their DNS strategies to minimize the effect of bot decay. Through the skillful manipulation of their T_{ttl}, $\overline{N}$ and $\overline{N}_{overlap}$, these remarkably different strategies were all able to achieve greater than 90% online availability (i.e., $\frac{\overline{N}_{online}}{\overline{N}}$).

Fast-Flux Domain	T_{ttl} seconds	N (# IPs per query)			$N_{overlap}$ (# of IP overlap)			$\overline{N}_{online}$ (# online IPs)	$\overline{P}_{loss}$ (Connection Loss Prob)
		min	avg	max	min	avg	max		λ = 100 conn/s, μ = 10 conn/s, K = 10
old-and-girl.net	600	3	5	5	0	1	5	4.53	54.00%
bentleycap.net	300	10	10	10	0	3	10	9.41	12.10%
mountainready.com	120	4	19	20	0	14	20	17.8	0.14%

TABLE II: Current FF DNS strategies and performance

Next, we examine the influence each type of strategy has on the botnet's overall capacity (i.e. $\hat{P}_{loss}$), which translates to the amount of potential victims and revenue. We use the values $\lambda = 100$, $\mu = 10$ and $K = 10$ for their ease of computation since, for comparison purposes, the actual choice for these values is trivial, so long as we are consistent and use the same values when evaluating each strategy. Applying Eq. (4) to each of the 3 botnets' DNS strategies, we determine $\overline{P}_{loss}$ once the system achieves a steady state, shown in Table II. While the varying DNS strategies offered comparable performance in terms of $\frac{\overline{N}_{online}}{\overline{N}}$, they clearly differ in the total capacity each botnet can support. This is a direct consequence of the number of bots available during a given TTL, with *mountainready.com* having approximately twice as many as *bentlycap.net* and 4x

[6] $\overline{N}$, $\overline{N}_{new}$ and $\overline{N}_{online}$ are the average values for N, N_{new} and N_{online}

as many as *old-and-girl.net*. With both a large $\overline{N}$ and $\overline{N}_{overlap}$, it seems that *mountainready.com* is attempting to capitalize on the load-balancing benefits provided by a large number of advertised IPs, while using large IP overlap to keep the total number of unique IPs over 2 queries relatively low; additionally, its use of a fairly small T_{ttl} indicates a proactive approach to countering the bot-decay phenomena, which will be accentuated due to its large $\overline{N}_{overlap}$. Conversely, *old-and-girl.net* makes use of a far different strategy. With its $\overline{N}_{overlap}$ constituting only a small fraction of its $\overline{N}$, the effect of bot decay due to IP overlap is less severe, permitting less diligent IP replacement and allowing for a longer T_{ttl}. Interestingly, its decision to use a small $\overline{N}$ and $\overline{N}_{overlap}$ appears to be a double-edged sword; while keeping the total unique IPs over 2 queries low, it also results in fewer IPs per TTL for load-balancing purposes, reducing the botnet's overall capacity. Lastly, *bentlycap.net* seems to have found some middle ground between the other techniques, with a T_{ttl} and $\overline{N}$ almost exactly between the those of others. However, like *old-and-girl.net*, it has chosen a small ratio of $\frac{\overline{N}_{overlap}}{\overline{N}}$, reducing the amount of bot decay and the need for more rapid IP replacement.

8) IP-Mimicry Attack: Now we present how our proposed IP-mimicry attack influences the connectivity and capacity of the aforementioned FF domains. The key idea of the attack is to manipulate $\overline{N}$ and $\overline{N}_{overlap}$ such that the online time and capacity are maximized while keeping the number of IPs below the detection threshold. First, we retain the FF domains' T_{ttl} values, assuming that they were chosen by the botmasters in response to how diligently they were willing to monitor and replace IPs. Second, in order to reduce false-positives from benign, non-CDN domains advertising a large number of stable IPs, such as *hostingprod.com*, we assume — for the purposes of this mimicry attack—a detection threshold of $N_{thresh} = 20$ IPs, resulting in the following policy: *over 2 DNS-queries, any domain with more than 20 unique A record IPs will be flagged as malicious*.[7]

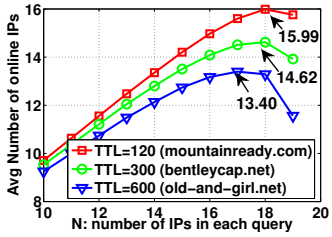


Fig. 8: $\overline{N}_{online}$ optimization

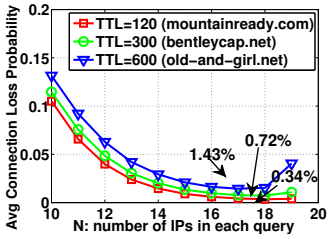


Fig. 9: $\overline{P}_{loss}$ optimization

It is clear from the results in Section III-B7, that the more online IPs available during a given TTL, the greater the botnet's overall capacity. Therefore, an optimal DNS strategy will necessarily advertise the maximum IPs allowed by the detector's threshold, N_{thresh}. This reduces the problem to an optimization problem, i.e., to determine an optimal $N_{overlap}$ that maximizes Eq. (5) or minimizes Eq. (6), subject to the constraints: $2 \cdot N - N_{overlap} = N_{thresh}$.

$$\overline{N}_{online} = \frac{\sum_{t=1}^{T_{ttl}} N_{online}(t)}{T_{ttl}} \quad (5) \qquad \overline{P}_{loss} = \frac{\sum_{t=1}^{T_{ttl}} P_{loss}(t)}{T_{ttl}} \quad (6)$$

For the FF domains in Table II, Figs. 8 and 9 show the results of Eqs. (5) and (6) across the search space $N \in [\lceil \frac{N_{thresh}}{2} \rceil, N_{thresh} - 1]$. From the figures, we can see that while *mountainready.com* and *bentlycap.net* achieve optimal performance with $N = 18$, for *old-and-girl.net*, $N = 17$. Apparently, its longer T_{ttl} of 600 seconds results in additional bot decay, causing $N = 17$—with its 2 fewer overlapped IPs—to provide better performance. We also find that for *bentlycap.net* and *old-and-girl.net*, their $\overline{N}_{online}$ has increased to 14.62 and 13.4, while their $\overline{P}_{loss}$ has decreased to 0.72% and 1.43%, respectively. While neither of these FF domains would have been detected by the imposed N_{thresh} under their original DNS strategies, utilizing the IP-mimicry attack has kept them from being detected while also greatly increasing their performance and capacity. On the other hand, the mimicry attack caused *mountainready.com* to suffer a reduction in $\overline{N}_{online}$, dropping from 17.8 to 15.99. The attack also caused its $\overline{P}_{loss}$ to more than double, increasing from 0.14% to 0.34%. However, *mountainready.com*'s original DNS strategy advertised 24 unique IPs over 2 queries, exceeding the detection threshold. Thus, the IP-mimicry attack has allowed it to successfully evade detection with only a minor decrease in performance—its average probability of a connection loss remains under 1% and its average online IPs has been reduced by less than 2.

IV. Extended-Window Detectors (more queries)

A. Good Guys

A logical extension to the fast-detection systems of the previous section is to increase the monitoring window to analyze more queries. Examining multiple TTLs when making a decision exploits a commonly known property of FF domains: they need to continuously advertise fresh IPs to account for their unstable constituent bots. While non-CDN domains may advertise a large number of IPs in their queries, they will be stable IPs and will not change over time. Thus, FF domains will quickly become exposed once additional queries are examined. Furthermore, while CDN domains can demonstrate the fluxy behavior characteristically attributed to FF botnets, for many CDNs, a longer detection window can allow their more stable nature to emerge from the chaos.

Current detectors, such as FluXOR [8] and RB-Seeker's second-tier detector, make use of longer detection windows (e.g., 1 week) to increase accuracy and support the detection of stealthy FF domains, which use slower DNS advertisement strategies to fool fast-detection systems. Like the Holz and RB-Seeker detectors, FluXOR examines the number of unique A records and ASNs. These are augmented with additional features aimed at capturing the quickly changing and dispersed nature of FF domains, such as TTL and the number of returned qualified domain names, or top-level domains (TLDs). Next, we examine how botmasters can mimic these features to evade detection.

[7] an attack defeating the ASN and DNS-"bad word" thresholds of Sections III-B1 and III-B2 can evade detection by the Holz and RB-Seeker detectors with as many as 79 and 66 unique IPs, respectively

B. Bot Guise

1) ASN-Mimicry Attack: Regrettably, extending the detection window in time does little to weaken the ASN mimicry attack described in Section III-B1. Because botnets seem to invariably control a sizable number of bots from within at least one ASN, the same essential attack can be performed by simply repeatedly using IPs from the same ASes to accommodate the larger detection window as shown in Fig. 10.

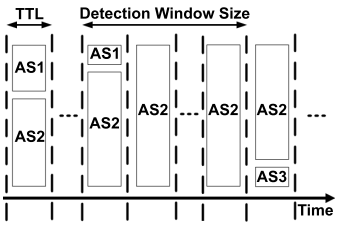


Fig. 10: ASN mimicry strategy (multiple queries)

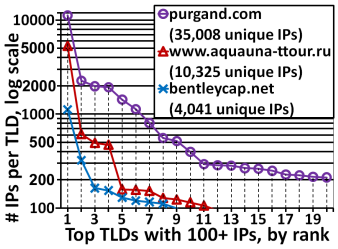


Fig. 11: IP distribution for top 20 TLDs.

2) rDNS-Mimicry Attack: While the specifics of FluXOR's "returned qualified domain" metric are not revealed in their paper, we can assume it operates as any TLD metric would. Essentially, for any rDNS results returned, the number of unique TLDs are calculated—the insight being that FF botnets, consisting of bots scattered across many networks, will return numerous TLDs. However, this feature also suffers from the inherent shortcoming of the rDNS lookup process, which doesn't always return a result. This allows for a sufficient quantity of rDNS=NONE IPs (adequately distributed across ASNs) and IPs from other TLDs to perform a similar dual-mimicry attack. Additionally, we analyzed the distribution of bot IPs across TLDs and found a similar distribution as across ASNs, in that there exist some TLDs from which a large number of bots belong. In Fig. 11, we have plotted this distribution for representative FF domains of varying sizes. Like the ASN distribution, it is long-tailed. While rDNS=NONE IPs dominate, there are clearly other TLDs with a sufficient number of IPs to similarly be used in the aforementioned mimicry attack, providing botmasters additional freedom in DNS advertisement strategies. Thus, we find that rDNS can be effectively mimicked based on FF botnets' current resources.

3) Improved DNS-Strategy Model: The DNS-strategy model developed in Section III-B6 can be extended to accommodate the larger detection window. First, let us assume the detector uses a detection window of D_{ttl} fresh DNS queries (where $TTL = Ttl$ seconds) and applies a threshold, N_{thresh}, on the number IPs seen during this detection window. The choice of N_{thresh} and D_{ttl} creates two scenarios botnets must overcome to avoid detection when replacing dead IPs. In the first case, when $D_{ttl} \leq N_{thresh}$, they can simply replace at least one new IP every T_{ttl}. However, if $D_{ttl} > N_{thresh}$, they can no longer introduce new IPs each T_{ttl} without exceeding N_{thresh}. To keep their total IPs below the threshold, botnets must repeat the same set of IPs over multiple T_{ttl} before introducing any new IPs. We term this the botnet's *repetition window* and define it as R_{ttl} DNS queries (also of length T_{ttl}) for which the botnet repeats the same set of IPs, effectively extending T_{ttl}. Thus, we can determine $N_{online}(t)$ by substituting $R_{ttl} \cdot T_{ttl}$ for T_{ttl} in Eq. (3). The choice of R_{ttl} determines A_{ttl}, i.e., *at most* the amount of IP changes any detection window D_{ttl} will observe. This relationship is shown in Eq. (7) and an example is given in Fig. 12, where we see that $A_{ttl} = 2$ when $D_{ttl} = 4$ and $R_{ttl} = 2$. Clearly botnets can add N_{new} IPs every R_{ttl} queries, so long as Eq. (8) is satisfied.

$$A_{ttl} = \lfloor \frac{D_{ttl} - 2}{R_{ttl}} \rfloor + 1 \quad (7) \qquad N + A_{ttl} \cdot N_{new} \leq N_{thresh} \quad (8)$$

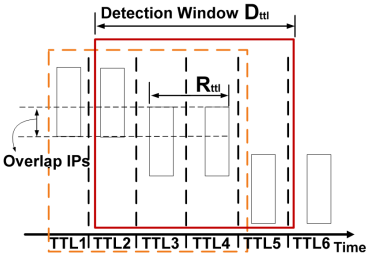


Fig. 12: Relationship between A_{ttl}, R_{ttl} and D_{ttl}

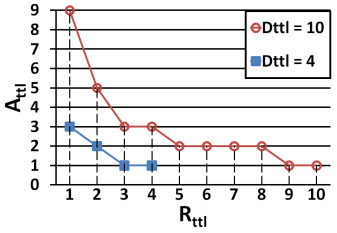


Fig. 13: A_{ttl} when $R_{ttl} \in [1, D_{ttl}]$ and $D_{ttl} = 4, 10$

4) IP-Mimicry Attack (TTL-based Detection Window): We apply the improved DNS strategy using R_{ttl} to our previous performance model to determine how the IP-mimicry attack fares against a larger detection window. For this purpose, we examine the same real-world FF domains as in Section III-B8, again, fixing their T_{ttl} to the values originally used by each domain. Modifying Eqs. (5) and (6) to incorporate the increased detection window produces:

$$\overline{N}_{online} = \frac{\sum_{t=1}^{R_{ttl} \cdot T_{ttl}} N_{online}(t)}{R_{ttl} \cdot T_{ttl}} \quad (9) \qquad \overline{P}_{loss} = \frac{\sum_{t=1}^{R_{ttl} \cdot T_{ttl}} P_{loss}(t)}{R_{ttl} \cdot T_{ttl}} \quad (10)$$

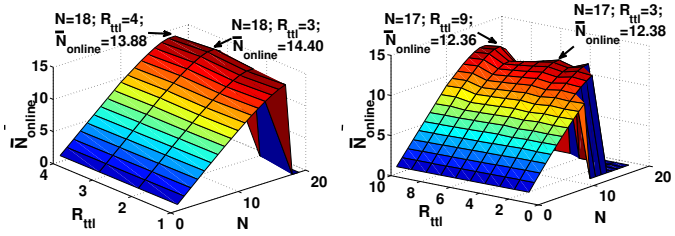
The goal for botmasters is to find the optimal values for R_{ttl} and N that maximize Eq. (9), or minimize Eq. (10), under the constraints that $R_{ttl} \in [1, D_{ttl}]$ and $N \in [\lceil \frac{N_{thresh}}{A_{ttl}+1} \rceil, N_{thresh} - A_{ttl}]$. The search space for N is found from Eq. (8) and the observation that $1 \leq N_{new} \leq N$. The optimization results for $N_{thresh} = 20$ and $D_{ttl} = 4, 10$ are shown in Table III. Notice that *all* the FF domains under their original DNS strategies (i.e., Table II) would have been caught by the extended detection window, with the exception of *old-and-girl.net* when $D_{ttl} = 4$. However, by using the proposed IP-mimicry strategy in Table III, they not only successfully circumvent detection but also achieve better capacity in most cases (i.e., *bentlycap.net* and *old-and-girl.net*). While *mountainready.com*'s performance goes down marginally, its $\overline{P}_{loss}$ is still less than 1% for $D_{ttl} = 4$ and less than 3% when $D_{ttl} = 10$. These slight decreases in performance are easily justified, considering that its original DNS strategy would have resulted in immediate detection, with 34 unique IPs seen for $D_{ttl} = 4$ and 64 for $D_{ttl} = 10$.

In Fig. 14, we show an example of the $\overline{N}_{online}$ optimization plots for *mountainready.com*'s $T_{ttl} = 120$, marking some local optimal points. To explain these plots, recall the relationship between D_{ttl}, R_{ttl} and A_{ttl} defined in Eq. (7) and shown for $D_{ttl} = 4, 10$ in Fig. 13. From the figures, we find that for values of R_{ttl} resulting in the same A_{ttl}, the lowest R_{ttl} is optimal.

Parameter Setup			Optimization Results				
D_{ttl} # TTLs in detection win	Fast-Flux Domain	T_{ttl} seconds	N (# IPs per query)	$N_{overlap}$ (# of IP overlap)	R_{ttl} # TTLs botnet repeats IPs	$\overline{N}_{online}$ (# online IPs)	$\overline{P}_{loss}$ λ = 100 conn/s μ = 10 conn/s K = 10 (Connection Loss Prob)
4	old-and-girl.net	600	16	15	1	11.20	4.92%
	bentlycap.net	300	17	16	1	12.71	3.11%
	mountainready.com	120	18	16	3	14.40	0.841%
10	old-and-girl.net	600	14	12	3	8.31	19.8%
	bentlycap.net	300	15	13	3	10.18	8.42%
	mountainready.com	120	17	16	3	12.38	2.58%

TABLE III: Optimization Results: IP-mimicry attack against D_{ttl}

This is best exemplified when $D_{ttl} = 10$ in Figs. 13 and 14-b, with local maxima at $R_{ttl} = 1, 2, 3, 5,$ and 9. To understand this behavior, recall that the amount of bot decay increases with R_{ttl} (due to repeating the same set of IPs over an extended duration). Since N and N_{new} are maximized with respect to A_{ttl} and N_{thresh} in Eq. (8), if A_{ttl} remains constant while R_{ttl} increases, performance will necessarily degrade, resulting in the observed trend.



(a) $D_{ttl} = 4$ (b) $D_{ttl} = 10$

Fig. 14: $\overline{N}_{online}$ optimization: $T_{ttl} = 120$ and $N_{thresh} = 20$

5) IP-Mimicry Attack (Time-based Detection Window): Thus far, we have defined the detection window in terms of the number of fresh DNS queries (D_{ttl}), showing that it can be subverted through the use of a repetition window, R_{ttl}. However, a detection window can also be defined in terms of absolute time (i.e., D_t seconds), requiring that the FF domain adhere to the IP threshold imposed over the duration D_t. Thus, the longer the duration, the more the FF domain's IPs are subjected to bot decay, decreasing performance. We model this detection technique and evaluate its susceptibility to IP-mimicry attacks under current botnet resources. For the purposes of this evaluation, we adopt a D_t equal to 1 week (as used in RB-Seeker). Certainly, requiring longer than a week to arrive at a detection decision grants botnets sufficient time to perpetrate their scams under a given domain. To find a suitable IP threshold, N_{week}, we analyzed the number of unique IPs accrued by benign CDN domains over 1 week. Not surprisingly, due to load-balancing techniques, CDN domains can advertise a large number of unique IPs depending on the DNS server monitored. For example, certain DIGGER nodes observed 171 IPs used by *nfl.com*. The amount was even greater for *www.myspace.com*, with many DIGGER nodes witnessing over 400 unique IPs, and in one case, over 700. We model the IP mimicry attacks against varying values of $N_{week} \in [100, 800]$, to see how increasing the threshold—to reduce false-positives—will affect a botnet's performance. To ensure that the mimicry attack would also continue to subvert fast-detection systems, we imposed the additional constraint of N_{thresh} IPs over *any* 2 DNS queries as before (i.e., $N + N_{new} \leq N_{thresh}$). Then, for each value of N_{week}, we calculate the maximum queries for which D_t can observe new IPs without violating N_{week} as $A'_{ttl} = \lfloor \frac{N_{week} - N}{N_{new}} \rfloor$. If we assume IPs are changed every TTL, then we can calculate the optimal T_{ttl} as $T_{opt} = \frac{T_{week}}{A'_{ttl}}$, where T_{week} is the number of seconds in a week. Under these constraints, the FF domain won't exceed the threshold of N_{week} unique IPs over the detection window $D_t = 1$ week. Furthermore, for any 2 queries, the number of unique IPs will satisfy the threshold N_{thresh}. Finally, notice that a repetition window, R_{ttl}, can be applied to T_{opt} to defeat a D_{ttl} detection window.

Table IV shows our optimized results for N, $N_{overlap}$, and T_{opt} with $N_{thresh} = 20$ under varying thresholds of N_{week}. For all values of N_{week}, the optimal values are $N = 19$, $N_{overlap} = 18$, and thus $N_{new} = 1$. This is because it is necessary to provide as many IPs per query as possible to counteract the enhanced botnet decay resulting from the longer T_{opt}. From the table, it is apparent that even for the strictest threshold $N_{week} = 100$ [8], the botnet will continue to have online IPs. Despite the high probability of lost connections, the botnet is still reachable and thus can continue to generate revenue. For the larger thresholds of $N_{week} = 200$ and above, the botnet capacity is greater than that of *old-and-girl.net* under its original configuration. These results affirm that, because benign CDN domains legitimately advertise large amounts of unique IPs over time, current botnet resources can sufficiently mount IP-mimicry attacks despite an increased detection window, D_t.

N_{week} max # unique IPs / week	T_{opt} (optimal TTL) seconds	$\overline{N}_{online}$ (# online IPs)	$\overline{P}_{loss}$ λ = 100 conn/s μ = 10 conn/s K = 10 (Connection Loss Prob)
100	7,466	2.89	71.10%
200	3,341	4.97	50.30%
250	2,618	5.79	42.20%
400	1,587	7.7	24.40%
700	888	9.95	9.37%
800	774	10.5	7.11%

TABLE IV: Optimization results: IP-mimicry attack against $D_t = 1$ week ($N_{thresh} = 20$: $N = 19$, $N_{overlap} = 18$)

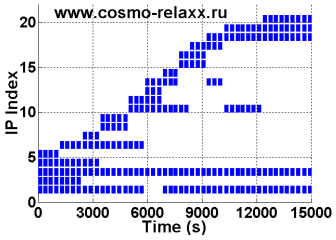


Fig. 15: Empirical observation of FF domain adopting certain evasion techniques ($T_{ttl} = 10$ seconds)

C. Empirical Observations

We discovered several FF domains in the wild adopting some of the mimicry attacks we have presented. While the strategies employed by FF domains in the wild aren't as meticulously regular as those in our models, they are close, only deviating from their average values rarely and in small amounts. Analysis of these domains show that many of them were able to defeat the Holz and RB-Seeker detectors in Section III-A. An example FF domain is shown in Fig. 15, with each box in the plot representing a unique IP seen in its DNS results. Observe that it adds 1 or 2 IPs every ≈1,000 seconds, replacing older IPs to keep the total number equal to 5 per query; thus, it uses an $N = 5$ and an $N_{overlap} = 3, 4$. Since it has a $T_{ttl} = 10$, it's essentially using a repetition window of $R_{ttl} = 100$. Under this DNS strategy, the FF domain

[8] a low threshold could result in many false positives since it's well below the number of IPs seen at individual DIGGER nodes for some CDN domains, such as *nfl.com* and *www.myspace.com*

can defeat a fast-detection system with an $N_{thresh} \geq 7$, as it occasionally introduces 2 new IPs per query. Furthermore, it will also defeat an extended detection window with $D_{ttl} \leq 101$ and $N_{thresh} \geq 7$. By using an average $N_{overlap} = 4$ in our model, the domain is estimated to achieve an $\overline{N}_{online} = 3.73$ (i.e., an average of 75% of its advertised IPs being online). This clearly shows that FF domains are beginning to incorporate advanced mimicry techniques to subvert detection systems, requiring novel detection methods exploiting the mimicry limitations of botnet resources. One such technique we are exploring is the use of multiple, cooperating, and geographically disparate detectors to take advantage of the spatial mimicry limitations of botnets.

V. Related Work

Recently, a number of techniques have been proposed to effectively detect FF domains [2], [8], [3], [9], [4]. They started with collecting DNS queries for a large number of suspicious domains through either active or passive monitoring, over time periods ranging from 1 or 2 TTLs to weeks. From these traces, they extracted similar sets of DNS features that can be used to characterize FF domains, for instance, the number of unique IPs, ASes, TLDs and spatial distribution [4] of IPs (similar to the number of ASes, the spatial distribution feature captures the dispersive nature of FF botnet IPs). Classification algorithms, such as support vector machines (SVM)[2], decision trees[8] and Bayesian network[4], were applied to the extracted features, determining if each domain is a FF domain. In this paper, we have demonstrated that, with the abundant resources currently available to botmasters, most of these features can be effectively subverted by the proposed mimicry attacks.

The concept of a mimicry attack was first proposed for host-based intrusion detection systems (IDSes), which typically monitor application behavior in terms of system-call sequences. Mimicry attackers attempt to slip under the radar by cloaking malicious system calls with innocuous-looking system-call sequences. Wagner and Soto [11] proposed a method that embeds nullified system-call sequences (i.e., "semantic no-ops") between malicious system calls. Kruegel *et al.* [5] devised techniques that allow an attacker to regain control after a system call by corrupting the memory and manipulating code pointers. This allows attackers to extend traditional mimicry attacks on more sophisticated IDSes. More recently, Parampalli *et al.* [7] proposed the persistent control-flow interposition techniques that make mimicry attacks simpler, more reliable and stealthy. Similar to these previous work, in this paper we design and evaluate several mimicry strategies that attackers may exploit to circumvent FF detection systems. The goal of our work is to anticipate attackers' next moves and better understand their capability in launching potential mimicry attacks. We hope this will foster improvements to existing defensive systems, allowing them to withstand future attacks and remain effective longer.

VI. Discussion and Conclusion

In this paper, we have examined the current state-of-art FF detectors, analyzing their effectiveness in detection. In doing so, we developed accurate models for bot decay, online availability, DNS advertisement, and performance, which we used to evaluate novel mimicry attacks against FF detection systems. Based on these models, empirical evidence, and logical assumptions, we have demonstrated that current botnet resources are sufficiently capable of subverting state-of-art FF detection mechanisms. We have discovered evidence of current FF domains adopting aspects of our proposed mimicry attacks, although they aren't managed as assiduously as our optimal models assume. Nevertheless, as detection systems improve and become more pervasive, we expect botmasters will increase their diligence in IP management to extract the most from their botnets' resources. We have shown that the incorporation of more advanced views—such as an extended-detection window—serves to encumber mimicry attacks by introducing additional, necessary parameters unknown a priori to botmasters. However, since security by obscurity is always a bad idea, ultimately, they are still susceptible to determined adversaries; for example, trial-and-error reconnaissance missions could derive the detection window size and type. Still, subverting the more advanced detectors requires significantly more effort from botmasters, making a strong argument for their adoption in augmenting the simpler, fast-detection systems. We hope showing the mimicry potential currently attainable by FF domains will foster improvements to existing detection systems, as well as provide new insight into the adaptive limitations of FF botnets. Our future work includes extending these models to handle a spatial dimension, allowing us to evaluate FF domains' current ability to mimic the location-aware advertisement strategies of CDNs. By incorporating multiple, cooperating, and geographically disparate detectors, we hope to impose additional constraints, straining botnet resources beyond their ability to perform mimicry attacks.

References

[1] Andreas Willig. A short introduction to queueing theory. http://www.tkn.tu-berlin.de/curricula/ws0203/ue-kn/qt.pdf, 1999.

[2] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling. Measuring and detectin fast-flux service networks. In *Proc. of network and Distributed System Security (NDSS) Symposium*, 2008.

[3] X. Hu, M. Knysz, and K. G. Shin. RB-Seeker: Auto-detection of Redirection Botnets. In *Proceedings of the NDSS'09*, 2009.

[4] S.-Y. Huang, C.-H. Mao, and H.-M. Lee. Fast-flux service network detection based on spatial snapshot mechanism for delay-free detection. In *Proceedings of the 5th ASIACCS*, 2010.

[5] C. Kruegel, E. Kirda, D. Mutz, W. Robertson, and G. Vigna. Automating mimicry attacks using static binary analysis. In *In USENIX Security Symposium*, 2005.

[6] P. Mockapetris. Domain names - implementation and specification. *RFC 1035*, 1987.

[7] C. Parampalli, R. Sekar, and R. Johnson. A practical mimicry attack against powerful system-call monitors. In *Proceedings of the 3rd ASIACCS*, 2008.

[8] E. Passerini, R. Paleari, L. Martignoni, and D. Bruschi. FluXOR: detecting and monitoring fast-flux service networks. In *Proceedings of DIMVA'08*, 2008.

[9] R. Perdisci, I. Corona, D. Dagon, and W. Lee. Detecting malicious flux service networks through passive analysis of recursive dns traces. In *Proceedings of ACSAC '09*, 2009.

[10] Planetlab. An open platform for developing, deploying and accessing planetary-scale services. http://www.planet-lab.org/.

[11] D. Wagner and P. Soto. Mimicry attacks on host-based intrusion detection systems. In *Proceedings of the 9th ACM CCS*, 2002.