A Taxonomy of Botnets

David Dagon[1] Guofei Gu[1] Cliff Zou[2] Julian Grizzard[1] Sanjeev Dwivedi[1]
Wenke Lee[1] Richard Lipton[1]
[1]Georgia Institute of Technology
{dagon@cc, guofei@cc, grizzard@ece, sanjeev@cc, wenke@cc, rjl@cc}.gatech.edu

[2]University of Central Florida
czou@cs.ucf.edu

Abstract

Attackers are increasingly using large networks of compromised machines to carry out further attacks (e.g., using botnets, or enormous groups of compromised hosts under the control of a single attacker). We consider the problem of responding to entire *networks* of attacking computers.

We identify key metrics for measuring the utility of a botnet, and describe various topological structures they may use to coordinate attacks. Using the performance metrics, we consider the ability of different response techniques to degrade or disrupt botnets.

Our models show that for scale free botnets, targeted responses are particularly effective. Further, botmasters' efforts to improve the robustness of scale free networks comes at a cost of diminished transitivity. Botmasters do not appear to have any structural solutions to this problem in scale free networks.

Our models also show that random graph botnets (e.g., those using structured P2P formations) are highly resistant to both random and targeted responses. This suggests the urgent need for further research into response strategies.

We validated our model on a particular class of botnets using star topologies. After tracking dozens of botnets over months, we located and performed a targeted response on a very large (100K member) botnet. This resulted in an over 90% reduction in the botnet population, and confirmed the utility of our taxonomy.

1 Introduction

Malware authors routinely harness the resources of their victims, creating networks of compromised machines called botnets. Worse still, these networks of victims machines are then used to attack other machines. The coordinated nature of these attacks requires the creation of new response strategies.

In order to respond to this threat, security researchers require an understanding of the structural and organizational potential of attacking networks. Similar to how previous work detailed key aspects of individual classes of worms [WPSC03], this paper provides a taxonomy of botnet organization.

Our taxonomy of organizational models demonstrate the utility of certain responses to botnets. To validate the models, we designed a response against a particularly common class of botnets. Our analysis also shows this response techniques will not work against all types of botnets. This experience may help guide research into this new and gathering threat.

Section 2 provides a background discussion of botnets (as opposed to bots). Readers familiar with botnets may wish to skip to Section 3, which details the structural organization of these networks. In Section 4, we analyze experimental response techniques designed to address a particular class of botnets. Since this area of research is new and rapidly changing, we conclude with suggestions for future work in Section 6.

2 Background

A botnet is best explained as a platform for distributed malicious computing [Sav05]. Thousands of victim computers make up the grid of this system. Spam, traditionally treated as a separate security problem, is just one application that "runs" on the botnet platform. Other applications running on the botnet platform include clickfraud, identity theft, denial of service, key cracking and copyright violations. For a detailed discussion of how botnets are being used, see [SS03, The05, CJ05].

Botnets are different from traditional discrete infections in that they act as a coordinated attacking group. Machines participating in a botnet frequently have numerous heterogenous infections: viruses, worms, and trojans. The cloud of victims can be used to create redundant, highly resilient networks for attacks. The infection that we must address is not merely the numerous binaries, but *the network of attackers itself.*

2.1 Botnet Propagation

It has been noted that "[i]n most cases, botnets are used to spread new bots" [The05]. But from where do botnets originally come? Many botnets use multiple penetration vectors ranging from one to a dozen or more methods [sym04, gos05]. These include the use of email [bab04, KE03], instant messaging [gos05, HC03, WPB04], remote software vulnerabilities [sym04, dip05, gos05, KE03, MPS$^+$03, SM04] (including vulnerabilities in other infections [dab04]), malicious web page content [aka04], unprotected network shares [sym04], and P2P files [sym04, gos05].

Significantly, botnets are now often created by other "seed" botnets [sym04]. Once an initial botnet is established, new botnets can grow, starting from an infected base of an existing botnet. As hosts are repaired or recruited, the botnet may shrink and grow.

2.2 Command and Control

Since victims are recruited through viruses, worms, and other random spreading processes, a common problem faced by malware authors is how to discover and organize their bots. The "command and control" (C&C) problem is frequently solved by having bots connect to named machines. Other types of solutions are possible.

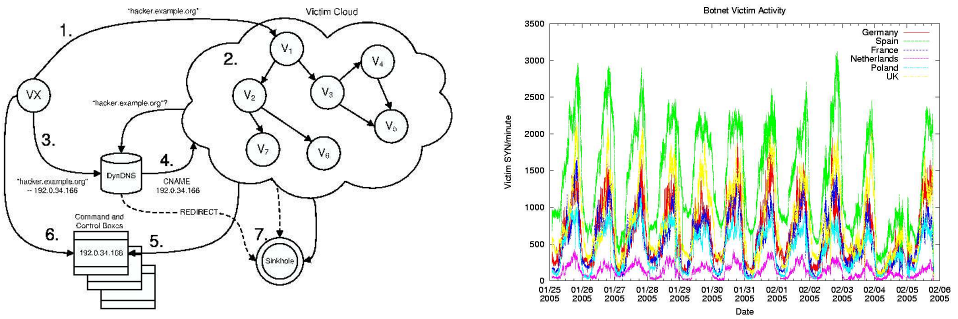
A simple example is instructive. A simple trojan will include the name of a machine hard coded as a string in the binary. (The string may be obfuscated or XOR'd to prevent trivial reverse engineering analysis.) Upon infection, the victim unpacks the string and contacts the C&C server or "rallying box". The malware author can then observe the connections, and communicate back to the victims. The problem with this naive approach is that the rallying box, used to gather all victims, is too easily cleaned. A single abuse report can cause the rallying box to be quarantined or the account suspended. Similarly, rival hackers can too easily DDoS a single C&C machine. Communication between the bots and the C&C machine is the weakest link in a botnet, without which the victim cloud does not behave as a coordinated network.

As a result, hackers have explored ways to create robust networks to rally their victims. A common technique is to use Internet Relay Chat (IRC) [Kal00], since these networks are very resilient, and designed to resist network faults. For a discussion of other rallying techniques, see [CJ05].

Server Migration. Cleaning up a single rallying box can potentially destroy the C&C for the botnet. Thus, malware authors try to keep their botnets mobile by using dynamic DNS (DDNS) [VTRB97], a resolution service that facilitates frequent updates and changes in machine locations. Each time the botnet C&C box is shut down by authorities, the botnet authors merely create a new command and control box, and update the appropriate dynamic DNS entry. The bots perform periodic DNS queries, and migrate to the new C&C location. This practice is known as "bot herding".

The general pattern of botnet creation is detailed in Fig. 1(a). To start, a malware author, VX in the diagram, will purchase one or several domain names (perhaps using stolen accounts).[1] The newly purchased

[1] Registrars usually charge under ten dollars (typically 10% - 20% markup over wholesale), and impose few or no Acceptable Use



(a) General Botnet Spread

(b) Diurnal Properties

Figure 1: (a) General spread of botnet (b) Diurnal properties in observed botnet.

domains are initially "parked" at 0.0.0.0, reserved for unknown addresses. The malware author then hardcodes the string names of their domains into a virus, and spreads the binary, as shown in steps 1 and 2 of Fig. 1(a).

While the virus spreads, the malware author also creates a C&C "rallying" box for the victims. This is typically one of two types: a high-bandwidth compromised machine, or (more frequently) a high-capacity co-located box (perhaps rented with stolen funds). The C&C box is set up to run an IRC service (often a modified version), to provide a medium for the bots to communicate.

Finally, the malware author will also arrange for DNS resolution of the domain names, and register with a DDNS service, as shown at step 3 in Fig. 1(a). The IP address they provide is for the C&C box. As DNS propagates, more victims join the network.

When a DDNS server suspends the account, the botmaster just moves on, and secures DNS from yet another company[2]. Similarly, if the co-location service revokes the C&C contract (or cleans the box, in the case where the malware author has used a compromised C&C box), the malware author just rents or steals another C&C box. Alternatively, the DNS provider can redirect traffic to a sinkhole, depicted in Step 7 of Fig. 1(a).

2.3 Recent Developments

The basic propagation scheme outlined above is the most common form of botnet organization. But many new botnets use different organizational techniques, including:

1. *Decentralized Naming Resolution*. Using techniques pioneered by spammers (e.g., Leo Kuvayev's "mushrooms software attack" [Cox05]), botnets now act as their own DNS cloud, and avoid centralized name resolution. Victims are instructed to use existing botnets for DNS resolution, avoiding the centralized use of naming services.

2. *Tor botnets*. IRC operators have reported botnets on the Undernet [Vun05] network connecting from TOR [MD05] exit nodes. Tor is a proxy network that uses a type of MIX-net routing to anonymize traffic. Botnets using Tor therefore pose difficult detection and response problems.

3. *Tunneling bots*. We note the increase in the number of bots that tunnel through existing protocols [bob04]. Bots potentially use net news, web blogs, and other resources. Fundamentally, these sorts of bots pose more of a problem for detection.

Policy (AUP) restrictions.

[2] There are over five dozen popular DDNS providers [DMO05] and hundreds of smaller providers.

3 Botnet Taxonomy

The evolving and evasive nature of botnets requires researchers to anticipate possible topologies. A significant early contribution in this area is [CJ05], which listed three topologies (centralized, peer-to-peer, and random) for botnets, and roughly evaluated performance metrics in terms of high, medium and low.

To more fully understand the threat, we expand on [CJ05] and propose a taxonomy of possible botnet topologies. This taxonomy is necessarily incomplete, since new tactics may arise. However, we believe it fully captures existing botnet structures, and describes all of the new, developing trends observed in the wild. Additionally, we define important metrics for botnet performance, to evaluate and compare response strategies.

3.1 Purpose and Goals

Taxonomies are most useful when they classify threats in dimensions that correspond to potential defenses [LJ97, LBMC94]. We agree with [KMT04] that "[a]n important and sensible goal for an attack taxonomy ... should be to help the defender."

Our botnet taxonomy will help researchers identify what types of responses are most effective against botnets. Our design goals are similar to [WPSC03]: (a) assist the defender in identifying possible types of botnets, (b) describe key properties of botnet classes, so researchers may focus their efforts on beneficial response technologies.

Our taxonomy is driven by possible responses, and not detection. Space does not allow us to adequately consider botnet detection techniques. There is some initial work in botnet detection [CJ05, Dag05, DTG+04, FHW05]. The considerable body of literature on worm detection has identified detection techniques that can be adapted to botnet detection [XKO+04, Par04, GSQ+04, JXWS05, ZGGT03, ZGT02, BGB03, ZTGC03].

3.2 Key Metrics for Botnet Structures

Naively, one could suppose that bots will organize according to various regular network topologies such as star, mesh, or bus networks. These topologies are useful for formal analysis of discrete network properties, but do not let us describe the robustness of large complex botnets.

Instead, we need to pay attention to key *discriminators* that let one compare important attributes of botnets. We identify three important measures of botnets: size, network diameter, and redundancy. We acknowledge there are other characteristics the botmaster may desire, but these are not easily designed into the victim network. For example, botmasters may desire victims with high-bandwidth; however, creating a botnet of only high-bandwidth victims is not straight forward. Table 1 lists a few botnet uses, and key relevant metrics. More than one metric can be relevant to a botnet use, and botnets certainly have multiple uses. However, the table lists key metrics critical to the botnet's specified function.

By *size* we do not mean the total population count usually used in worm epidemiology studies [MPS+03, MSVS03, SM04, Moo02]. Instead, we mean the "giant" component of the botnet, or largest connected (or online) portion of the graph [Bol85, NSW01]. Botnets are of course more powerful if they have large infected population, but the giant component lets us directly measure the damage potentially caused by certain botnet functions.

In the case of DDoS, the giant component, S, lets us measure the largest number of bots that can receive instructions and participate in an attack. This contrasts with the total population of all infected victims, which may not always be reachable by the botmaster. Figure 1(a) shows the pattern of SYN packets sent by botnet victims in Europe to their C&C machine, for a large (100K victim) botnet observed over months. (The data collection steps are detailed in Section 4.) Given the diurnal property of botnets, we selected giant as a metric instead of total infected population.

By *network diameter* we mean the average geodesic length of a network, l. This expresses the average length of the shortest edge connecting any two nodes in the network. If l is large, the dynamics of the network

Major Botnet Behavior	**Key Metrics**	**Variable**	**Comment**
Bandwidth-oriented (e.g., DDoS)	Giant portion	S	Large numbers of victim increases the likelihood of high-bandwidth bots. Diurnal behavior favors S over total population.
Proxied abuse (spam, clickfraud)	Diameter	l^{-1}	Bots coordinating spamming, distributed key logging and dispersed attacks require communications.
Storage and computing (warez, keycracking tables)	Local transitivity	γ	Bots used for storage or keycracking achieve reliability through redundancy. Triads coordinate the storage and processing of data.

Table 1: Botnet Uses and Relevant Metrics

(communications, information, epidemics) is slow. In Milgram's famous paper, social networks were shown to have short average geodesic lengths, approximately log N, or $l \approx 6$ ("six degrees of separation") for the earth's population [Mil67], while the web has a larger estimated length, $l \approx 17$ [AA99].

As in [HKYH02], we use the inverse geodesic length, l^{-1}, instead of l, defined as:

$$l^{-1} = \left\langle \frac{1}{d(v,w)} \right\rangle = \frac{1}{N(N-1)} \sum_{v \in \mathcal{V}} \sum_{w \neq v \in \mathcal{W}} \frac{1}{d(v,w)'} \tag{1}$$

This way, if nodes v and w are disconnected, the distance d is zero. Further, the inverse length is normalized, ranging from 0 (no edges) to 1 (fully connected). In the context of botnets, l^{-1} refers to the overlay network of bot-to-bot connections created by the malware, instead of the physical topology of the internet. Thus, bot victims on the same local network (one hop away) may be several edges apart or even unconnected in the overlay bot network created by the malware.

This metric is relevant because with each message passed through a botnet, there is a probability of detection and failure. Some researchers have already investigated zombie detection via stepping stone analysis, or the detection of messages being relayed through victim proxies [ZP00]. It is difficult to express this chance of detection precisely, since botnet identification is a new, developing field. But in a simple sense, we can express this as the basic chance of intercepting (i.e., detecting and corrupting or halting) a message between two bots in a network. Assume that bots u and v are connected through n possible paths, $P_1, \ldots P_n$, and that each node in the path can be recovered (cleaned) with probability α. If ϵ_i is the chance that path P_i is corrupted, quarantined or blocked, then all paths between u and v are blocked with probability:

$$\prod_{i=1}^{n} \epsilon_i \leq (1-\alpha)^n \tag{2}$$

While nodes u and v are connected through *some* path with probability $1-(1-\alpha)^n$, the chance of failure increases with α (i.e., as detection technologies improve). Section 4 characterizes the performance of l^{-1} under increasing link decay.

As noted in Table 1, we see DDoS as a distinct use compared to proxy-related activities such as spam and clickfraud. Even logarithmically connected networks would enjoy near fault-free messaging under Eqn.(2), albeit through potentially lengthy paths. This may be appropriate for DDoS where basic connectivity lets botmasters send the "attack" message to all the bots. But it is not optimal for spam and clickfraud, where bot-to-bot coordination is useful, and shorter paths are desirable.

The incentive of the botmaster is therefore to increase l^{-1}, at least for selected purposes noted in Table 1. Under an ideal $l^{-1} = 1$, every bot can talk directly to every other bot. Since a botnet with more interconnections has more short paths, it passes messages quickly, and provides fewer detection opportunities.

To some degree this metric correlates with an improved redundancy. We more precisely capture the robustness of networks using local transitivity to measure *redundancy*. Local transitivity measures the likelihood that nodes appear in "triad" groups. That is, given two node pairs, $\{u, v\}$ and $\{u, w\}$, that share a common node, u, local transitivity measures the chance that the other two, v and w, also share an edge. A clustering coefficient γ, measures the average degree of local transitivity [WS98], in a neighborhood of vertices around node v, Γ_v. If E_v represents the number of edges in Γ_v, then γ_v is the clustering coefficient of node v. Where k_v represents the number of vertices in Γ_v, then we have:

$$\gamma_v = \frac{E_v}{\binom{k_v}{2}}, \gamma = \langle \gamma \rangle = \frac{1}{N} \sum_{v \in \mathcal{V}} \gamma_v. \tag{3}$$

The average clustering coefficient $\langle \gamma \rangle$ measures the number of triads divided by the maximal number of possible triads. Just like l^{-1}, γ ranges from $[0, 1]$, with 1 representing a complete mesh. Local transitivity is an important measure for certain botnet uses. Warez (stolen programs) and key cracking require reliable, redundant storage, particularly since botnets exhibit strongly diurnal properties. To ensure uninterrupted key cracking, or that file resources are always available, botmasters routinely designate multiple victims to store identical files. (For examples, consult [Cal05].) Botmasters could use quorum systems in addition to simple backups. However, the transitivity measure γ index generally captures the redundancy of a botnet.

3.3 Types of Responses to Botnets

To measure the robustness of different botnet architectures, we must further specify the types of response actions available to network administrators. In a general sense, botnets can suffer random and targeted responses. Random failures correspond to patching by normal users, diurnal properties of computers being powered off at night, and other random failures in a network. Targeted responses are those that select "high value" machines to recover or patch. These response types all correspond to actions directed at botnet vertices. Edge-oriented responses (e.g., quarantine, null routing) have been considered elsewhere, e.g., [ZGT03].

3.4 Botnet Network Models

Expanding on the general categories of botnets noted in [CJ05], we consider different types of graphs studied in the extensive literature on complex networks. Our taxonomy uses the major models from that field. For a comprehensive overview of complex network mechanics, see [AB02].

3.4.1 Erdös-Rényi Random Graph Models

To avoid creating predictable flows, botnets can be structured as random graphs. In a random graph, each node is connected with equal probability to the other $N - 1$ nodes. Such networks have a logarithmically increasing l^{-1}. The chance a bot has a degree of k is the binomial distribution:

$$Pr(k) = \binom{N-1}{k} p^k (1-p)^{N-1-k} \tag{4}$$

Particularly for large networks like botnets, it makes sense to limit the degree k to a maximum number of edges, L. For our analysis below, we select an average $\langle k \rangle$ appropriate to botnets, instead of $\langle k \rangle \approx 2L/N$ used by others studying general network complexity problems [HKYH02]. Without such a limitation, a pure Erdös-Rényi random botnet would potentially create individual bots with hundreds of edges, even for small (5K victim) botnets. Large numbers of connections on a client host are highly unusual, even for P2P software [RFI02, LCC+02]. So, unless the victim is a rare high-capacity server, botmasters would keep $\langle k \rangle$ small, say $\langle k \rangle \approx 10$.

One difficulty in random graphs is easily overcome by certain types of botnets. Since each node has a probability $Pr(k)$ of being connected to each vertex, the creation of the graph requires some central collection (or record) of vertices. That is, each bot must either know or learn the address of all the other bots, in order to have a chance of sharing an edge. Botmasters clearly do not want to create such a central list, and some bots, e.g., those created by the zindos worm [LUR04], take explicit steps to limit the number of victim addresses stored in one place.

This creates a technical problem for botnets that propagate through traditional (e.g., scanning, mass-mailing) techniques. The first victims will not know the address of subsequent victims, and have a $Pr(k)$ biased towards zero. But as noted in Section 2, botnets are now frequently used to create new botnets. So, random graph overlays are easily created on top of existing centralized "star" botnets. Attackers merely have to keep track of victims joining their botnet, generate a desired topology overlay, and transmit the edge sets to each bot.

Bot masters can easily select a desired $\langle k \rangle$ to generate such a network. For example, they may select $\langle k \rangle \leq 10$, so that bots appear to have flow behavior similar to many peer-to-peer applications [RFI02, LCC$^+$02]. A botmaster could of course select a higher $\langle k \rangle$, even one close to N to create a mesh, but such structures quickly exhaust bot resources, and may be easily detected by administrators.

If existing botnets are not available to generate a random graph, one solution was proposed by [CJ05], where bots could randomly scan the Internet to find fellow bots. Although noisy, this approach provides a last-resort technique for botnet creation. Assuming random scanning up to L connections, the resulting botnet would have a poisson k distribution, and both the clustering and diameter properties of a random graph.

3.4.2 Watts-Strogatz Small World Models

In a Watts-Strogatz network, a regional network of local connections is created in a ring, within a range r. Each bot is further connected with probability P to nodes on the opposite side of the ring through a "shortcut". Typically, P is quite low, and the resulting network has a length $l \approx \log N$. See [AB02] for further discussion of small world networks.

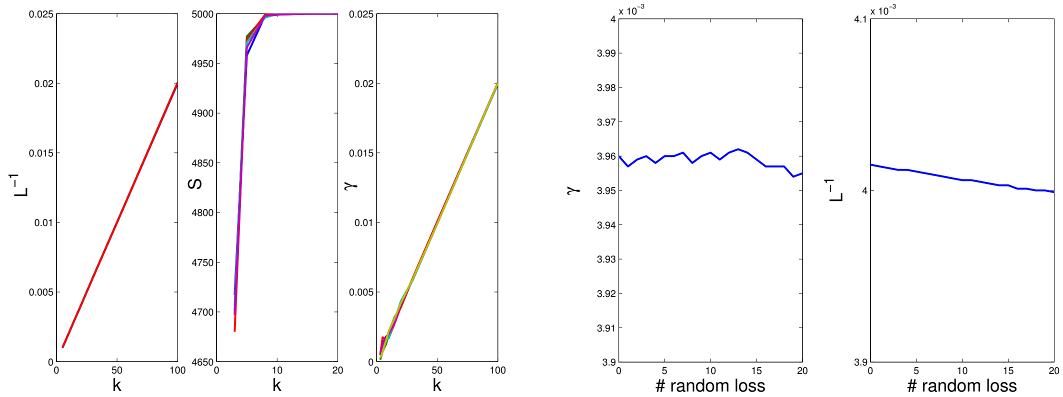
Intuitively, we can imagine a botnet that spreads by passing along a list of r prior victims, so that each new bot can connect to the previous r victims. To create shortcuts in the small world, bots could also append their address to a growing list of victims, and with probability P connect back to a prior bot. As noted in section 2, we have witnessed botnets that create prior victim lists [LUR04]. To frustrate remediation and recovery, the lists are typically small $r \approx 5$. In the case of propagation-created botnets, botmasters may prudently use $P = 0$, to avoiding transmitting a lengthy list of prior victims. Otherwise, a bot would have to append its address to a growing list of IPs forwarded to each new victim. As noted above, if a botmaster desired to have shortcuts in a small world botnet, they could instead just use an existing botnet.

3.4.3 Barabási-Albert Scale Free Models

The previous botnet structures are characterized by variations in clustering, and each node exhibits a similar degree, $k \approx \langle k \rangle$. In contrast, a Barabási-Albert network is distinguished by degree distribution, and the distribution of k decays as a power law. Many real-world networks have an observed power-law distribution of degrees, creating a so-called scale free structure.

Scale-free networks contain a small number of central, highly connected "hubs" nodes, and many leaf nodes with fewer connections. This has a significant impact on the operation of the network. As discussed in Section 4, random node failures tend to strike low-degree bots, making the network resistant to random patching and loss. Targeted responses, however, can select the high degree nodes, leading to dramatic decay in the operation of the network. This phenomenon is explored in many articles, e.g., [AJB00].

Researchers have noted that bots tend to organize in scale free structures, or even star topologies [Bru03, DTG$^+$04, FHW05]. For example, botnets might use IRCd [Kal00] for coordination, which explicitly uses a hub architecture. For now, but perhaps not for long, botnets predominantly rely on scale free structures [DTG$^+$04].



(a) Node Degree in Random Botnets

(b) Random Loss in Botnets

Figure 2: (a) Changes in length l^{-1}, giant (s), and local transitivity (γ) in response to changes in critical values of k, for 5K victim botnet. (b) Effect of loss on random networks.

3.4.4 P2P Models

In a P2P model, there are structured and unstructured topologies [QPC$^+$02, RFH$^+$01]. For example, a structured P2P network might use CHORD [SMK$^+$01], or CAN [RFH$^+$01], while an unstructured P2P might use the hub-and-spoke networks created under gnutella or kazaa [QPC$^+$02].

The unstructured P2P networks tend to have power-law link distributions [QPC$^+$02]. We therefore treat this type of P2P network as a Barabási-Albert (scale free) model in our analysis. Similarly, structured P2P networks are similar to random networks, in the sense that every node has almost the same degree.

In section 2, we noted the appearance of new P2P-based botnets. Since our selected metrics concern only basic botnet properties (length, giant, and local transitivity), we can treat these networks as random or scale free in our analysis. We encourage others to refine these models to identify distinct P2P botnet features that distinguish them from random and scale free networks. For our purposes, however, we will address P2P botnets as special cases of the previous categories.

4 Taxonomy-Driven Botnet Response Strategies

The previous discussion of botnet organization suggests the need for diverse response strategies. To guide research in this developing area, we model different responses to each botnet category. Our analysis confirms the prevailing wisdom [CJ05] that command-and-control is often the weak link of a botnet. We confirm our model with an empirical analysis of a real-world botnet response. Significantly, our analysis also shows that targeting the botnet C&C is not always an effective response. Some botnets will require new response strategies that research must provide.

4.1 Erdös-Rényi Models

For ranges appropriate to botnets, we evaluate the relationship between node degree, k, and the diameter of the botnet, expressed as l^{-1}. We assume that, to evade trivial detection, botnets will attempt to limit $\langle k \rangle$ to some value similar to P2P. Empirical studies of P2P systems reveal very low median link scores (e.g., $k \approx 5.5$) [RFI02, LCC$^+$02]. Figure 2(a) plots $\langle k \rangle$ against l^{-1} for realistic values, $k \leq 20$. Others have noted that for increasing average degrees, $\langle k \rangle$, random Erdos-Renyi models have logarithmically increasing diameters [HKYH02]. However, in figure 2, realistic values of k show a *linear* relationship to l^{-1}.

We also note that giant improves significantly with increases in k, enabling connections with most of the botnet when $k \approx 10$ for a 5K botnet. This agrees with the general principle noted in Eqn. (2), where logarithmically connected networks enjoy nearly universal broadcasting.

Local transitivity, γ, also increases logarithmically with k. But for a range of small values of k, typical of botnets, it shows a linear increase. This means that each additional value of k equally improves the general robustness of the botnet. We also note a slight flare at the base of the γ plot for Figure 2(a), for very low values of k. Intuitively, this means botnets with a very low average degree have difficulty forming triads, but this is quickly overcome as k increases. Botmasters therefore have incentives to increase k.

Random botnets are particularly resilient to response. Figure 2(b) shows that both random responses fail to significantly reduce either the diameter or transitivity values. We omit plotting changes to s under random loss, since the node loss is merely some small constant of the number of nodes removed.

If a botnet uses a random topology, then infrequent user patching fails to diminish the number of triads in the botnet. We also omit plotting the performance of random networks under targeted responses. Targeting nodes can at best remove a few nodes with k slightly higher than $\langle k \rangle$. The result is asymptotically the same as random loss.

In section 3 we noted that structured P2P networks are very similar to random networks, at least in terms of the metrics we care about: length, giant and transitivity. Structured P2P networks in fact have a constant k (often set equal to the log N size of the network), so they are slightly more stable than purely random networks. Thus, changes in γ and S, and l^{-1} are constant with the loss of random nodes.

Clearly botnets with random topologies (including structured P2P networks) are therefore extremely resilient, and deserve further study. We speculate that the most effective response strategies will include technologies to remove large numbers of nodes at once. Detecting and cleaning up large numbers of victims (perhaps at the host level) appears to be the most viable strategy. Some existing research (e.g., seurat [XKO$^+$04]) might be adapted to address this problem.

4.2 Watts-Strogatz Models

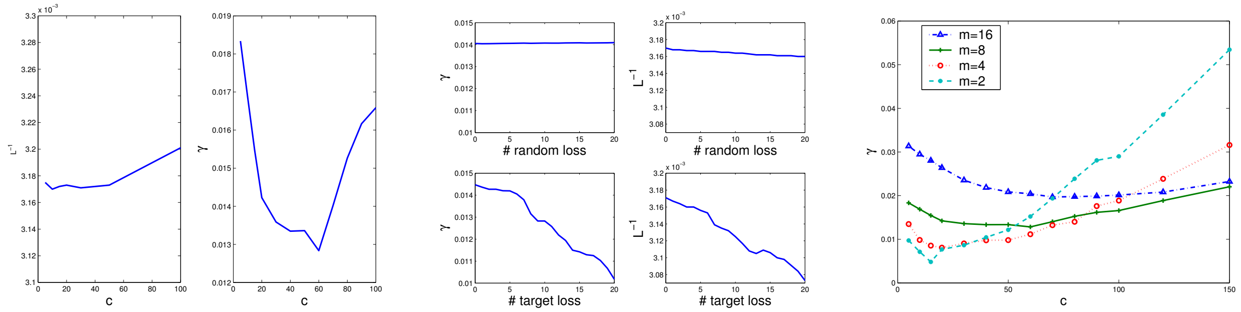
There are some experimental botnets [LUR04] that use small world structures, but overall they do not appear to yield significant benefits under the selected metrics. The average degree in a small world is $\langle k \rangle \approx r$, or the number of local links in a graph. Thus, random and targeted responses to a small world botnet produce the same result: the loss of r links with each removed node. Thus, the key metrics for botnets, s, γ, l^{-1} all decay at a constant rate in a small world.

We presumed that shortcut links in a small world botnet are not used ($P = 0$), but even if present, they would not affect γ with $r \geq 4$. That is, if the number of local links is large enough to form triads, the absence of shortcuts does not significantly increase the number of triads (which are already formed by r local neighbors).

There may be other benefits (e.g., propagation stealth), for which we have not devised a metric. But overall, small world botnets do not have benefits different from random networks. In other domains, researchers have noted that small world graphs are essentially random [HKYH02].

4.3 Barabási-Albert Models

While random networks present a challenge, at least scale free networks provide some good news for researchers. Figure 3(a) plots the change in diameter and transitivity against changes in the "core" size of the botnet, C. The "core" of a scale free botnet is the number of high-degree central nodes–the routers and hubs used to coordinate the soldier bots. As more core nodes are added, the diameter of the scale free botnet stays nearly constant for small regions of C. Intuitively, splitting a hub into smaller hubs does not significantly increase the length of the overall network.



(a) Scale Free Properties

(b) Loss in Scale Free Networks

(c) Scale Free Networks and Transitivity

Figure 3: (a) Changes in diameter and transitivity vs. core size, for a 5K scale free botnet. (b) Loss in scale free networks. (c) Changes in link count for leaves in a scale free network.

The local minima in Figure 3(a) has an intuitive explanation. If we have a single hub in a scale free network, $C = 1$, many of the added leaf nodes have a good chance of forming triads. The scale-free generation algorithm we chose prefers high degree nodes, and tends to form many triads when there are few hubs.

As we increase C, we create several high degree hubs that attract distinct groups of leaf nodes. This creates many "squares", where hubs are connected to each other, and leaves are connected to each other. But transitivity is only measured locally (in triads, and not other polygon paths). Thus, increasing C diminishes γ slightly. As we increase C more, we observe a tendency for the hubs themselves to form triads, so γ grows logarithmically.

Can botmasters avoid this drop in transitivity? In Figure 3(c), we compare changes in γ against core size using different link counts for leaf nodes. If nodes have more links, $m \approx 16$, the loss in γ shallows out. But increasing the link count of nodes can help detection. This reveals a curious mix of incentives. On the one hand botmasters would like to have $C >> 1$, since a single core node is too easily removed. But increasing C just a little drops local transitivity. To recover the loss in transitivity, botmasters would have to increase link counts to rates far in excess of average P2P degree counts.

Responses to scale free botnets are more effective. As expected, random losses in scale free botnets are easily absorbed. Figure 3(b) shows that random patching has almost no affect on a botnet diameter or the frequency of triad clusters. Intuitively, because of the power law distribution of node degrees, random losses tend to affect low-degree nodes (e.g., the leaves), and not important nodes (e.g., hubs).

Targeted responses, however, can select key nodes for response. This results in a dramatic increase in diameter, and loss of transitivity. This suggests that researchers should focus on technologies that allow targeted responses to high-degree nodes in botnets. Figure 3(b) validates the intuitive idea that by removing a botnet C&C, the network quickly disintegrates into a collection of discrete, uncoordinated infections.

4.4 Empirical Analysis of Responses to Scale-Free Botnets

Our analysis of the characteristics of botnets shows that targeted responses work best against scale-free networks. To confirm our previous analysis, we tested a targeted response against a botnet.

4.4.1 Targeted Response Using DDNS Sinkholing

We worked with a Dynamic DNS provider to identify botnets using DNS to rally victims, as discussed in section 2. Once a botnet was identified and confirmed, the provider entered a record update so the C&C box instead resolved to an experimental sinkhole. The sinkhole operated as a tarpit [Har02, Lis01], and passively recorded traffic. Over weeks, we could observed nearly every victim in the botnet. We controlled for DHCP churn by limiting our population samples to a 2-hour range.

4.4.2 Analysis

We wish to measure how many bots are removed from a network after a targeted response. Evaluating the success of our response requires us to know N, the total population of infected individuals. This is difficult to measure even in retrospect. We can however, use simple estimates from population biology to generally gauge this response.

Population estimates for closed systems (e.g., the Lincoln-Peterson index [Kre99]) require at least two independent samples, M, and C, for the mark and capture sets. The second sample is merely a random selection of the set $\binom{N}{C}$. We define M as the number of individuals marked by the first sample, C as the number of individuals observed in the second, and R as the number both samples have in common (i.e., the recaptured population).

With R conditioned on M and C, the distribution of R is hypergeometric:

$$f(R|M,C) = \frac{\binom{M}{R}\binom{N-M}{C-R}}{\binom{N}{C}} \tag{5}$$

If the mark and capture population samples are suitably large percentages of the total population, i.e., $M + C \geq N$, the estimate $\hat{N}$ is unbiased even for small sample sizes [Kre99]:

$$\hat{N} = \frac{(M+1)(C+1)}{R+1} + 1 \tag{6}$$

The sinkhole may not always yield sufficiently large mark and capture samples to estimate $\hat{N}$. In situations where the infectious binary (obtained in this case through honeypots or interactions with individual victims) uses only one or a limited number of DDNS providers, this may be a safe assumption.

With a normal distribution for $\hat{N}$, we can further calculate a 95% confidence interval for this population as $\hat{N} \pm 1.96\sqrt{v}$, where:

$$v = \frac{(M+1)(C+1)(M-R)(C-R)}{(R+1)^2(R+2)} \tag{7}$$

We identified a botnet in our data set that used two separate but similar DDNS names, identical binaries, and had similar DNS registration histories. We therefore deemed these to be two samples (mark and capture) of a single botnet. The botnets had 107,848 and 168,068 victims for M and C respectively, with 100,099 victims in common. Using Eqs.(6), (7), and a 95% confidence interval, we estimate an overall population, $\hat{N} = 181,078 \pm 506$. Thus, over 90% of the estimation bot population was captured by the DDNS redirection.

We acknowledge our simple sampling technique has shortcomings. For example, there were likely new bots joining the network between samplings, and some patching. However, the sampling time was just one day, so we consider the contribution of random loss was minimal. In general, the experiment shows that, for scale-free botnets, particularly those using star-topologies, targeted responses such as DNS manipulation can be effective in capturing most of an infectious network. This suggests that other targeted response techniques (e.g., recovering the C&C machine) may be similarly effective.

The length of the botnet was presumed to be a small constant, since the botnet used a star topology. Thus, we did not measure l^{-1}. Similarly, the transitivity, γ of the simple botnet structure was presumed to be a constant. We were unaware of any victim-to-victim connections in this botnet (and the binary sample did not produce any in a honeypot). Thus, we did not consider γ, and have not identified a way to measure it empirically in bots that do not join the sinkhole.

5 Related Work

Our work fits into the larger body of literature addressing the statistical mechanics of complex networks [AB02]. Others have studied the brittle nature of scale-free networks and resilience of random networks in other contexts [AJB00, HKYH02, NA05]. Our work adapts these findings to the particular domain of botnets.

Botnet research is still maturing. The work in [CJ05] anticipated many of the general categories of botnets analyzed in Section 3, including the difficulty in responding to different type of botnet taxonomies. The models and empirical data we presented flesh out the intuitive discussion in [CJ05].

The topology of networks under active decay was analyzed in [NA05]. Many of the results in [NA05] anticipate our own. The authors took a fascinating look at all domains of network structures (e.g., including terror cells, and global history), and not just computer networks. By restricting our analysis to botnets, we identified several unique and interesting phenomena not considered in [NA05]. For example, the authors in [NA05] suggest a strategy of splitting high-degree nodes to avoid targeted responses. This is analogous to increasing C in scale free networks, discussed in 4. Since we focused on the botnet domain, we were able to further observe that this results in a degraded transitivity.

Other researchers have indirectly addressed botnets by trying to mitigate against particular attacks. For example, in [KKJB05], the authors designed sets of Turing tests (puzzles) that users must solve to access overtaxed resources. This clever approach rations resources under a DoS attack. Similarly, there is an extensive body of research on stopping DDoS attacks (e.g., through traceback) after an attack occurs [MVS01].

Our taxonomy and discussion of general response options presumes a sensitive detection system. We have not considered detection of botnets, and urge further research. We note preliminary detection work in misuse systems [Han04], and IRC traces [Bru03]. Significantly, this early work focuses on tracking *individual* bots (e.g., to obtain a binary) and not the *network* cloud of coordinated attackers addressed in our study. The only other research focused directly on countering bot*nets* (as opposed to individual bots) is [FHW05,CJ05], which used honeypots and broad sensors to track and infiltrate botnets.

6 Conclusion

Botnets present significant new challenges for researchers. The fluid nature of this problem requires researchers anticipate future botnet strategies and design effective response techniques. To assist in this effort, we presented a taxonomy of botnets based on topological structure.

Our analysis shows that random network models (either direct Erdös-Rényi models or structured P2P systems) give botnets considerable resilience. Such formations resist both random and targeted responses. Our analysis also showed that targeted removals on scale free botnets offer the best response.

We have demonstrated the utility of this taxonomy by selecting a class of botnets to remediate. Our analysis suggested that by removing command and control nodes, targeted removal was an effective response to scale-free botnets. We measured the impact of such responses in simulations, and using a real botnet.

6.1 Future Work

Our response strategies considered only targeted and random responses to botnets. Byzantine failures in a botnet, where administrators infiltrate a network, e.g., [FHW05], may present a third response option. To some extent, [NA05] anticipates some issues in such failures. Future work should assess the impact of such failures on key metrics, and identify metrics for evasion and detection.

Because of the difficulty in measuring botnets, our empirical analysis necessarily considered only changes in giant, s, in scale free botnets. Future work will investigate the potential of honeypots to measure local transitivity in botnets, including P2P botnets. The botnets we captured were very large (100K+ members), and proved difficult to manage using current honeypot technologies.

References

[AA99] A.-L. Barabási and R. Albert. *Science*, 286(509), 1999.

[AB02] Réka Albert and Alert-László Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1), 2002.

[AJB00] Réka Albert, Hawoong Jeong, and Alert-Lászloó Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378=382, 2000.

[aka04] Akak trojan analysis. `http://www.lurhq.com/akak.html`, August 2004.

[bab04] I-worm baba analysis. `http://www.lurhq.com/baba.html`, October 2004.

[BGB03] V.H. Berk, R.S. Gray, and G. Bakos. Using sensor networks and data fusion for early detection of active worms. In *Proceedings of the SPIE AeroSense*, 2003.

[bob04] Bobax trojan analysis. `http://www.lurhq.com/bobax.html`, March 2004.

[Bol85] B. Bollobás. *Random Graphs*. Academic Press, 1985.

[Bru03] David Brumley. Tracking hackers on IRC. `http://www.doomdead.com/texts/ircmirc/TrackingHackersonIRC.htm`, 2003.

[Cal05] Edwin Calimbo. Packetnews: The ultimate irc search engine. `http://www.packetnews.com/`, 2005.

[CJ05] Evan Cooke and Farnam Jahanian. The zombie roundup: Understanding, detecting, and disrupting botnets. In *Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI '05)*, 2005.

[Cox05] Richard Cox. SBL advisory. `http://www.spamhaus.org/sbl/sbl.lasso?query=SBL30115`, 2005.

[dab04] Dabber worm analysis. `http://www.lurhq.com/dabber.html`, May 2004.

[Dag05] David Dagon. The network is the infection. `http://www.caida.org/projects/oarc/200507/slides/oarc0507-D\agon.pdf`, 2005.

[dip05] Dipnet/oddbob worm analysis. `http://www.lurhq.com/dipnet.html`, January 2005.

[DMO05] DMOZ Open Directory Project. Dynamic dns providers list. `http://dmoz.org/Computers/Software/Internet/Servers/Address_Management/Dynamic_DNS_Services/`, 2005.

[DTG$^+$04] David Dagon, Amar Takar, Guofei Gu, Xinzhou Qin, and Wenke Lee. Worm population control through periodic response. Technical report, Georgia Institute of Technology, June 2004.

[FHW05] Felix C. Freiling, Thorsten Holz, and Georg Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. Technical Report ISSN-0935-3232, RWTH Aachen, April 2005.

[gos05] Malware evolution: January - march 2005. `http://www.viruslist.com/en/analysis?pubid=162454316`, April 2005.

[GSQ$^+$04] Guofei Gu, Monirul Sharif, Xinzhou Qin, David Dagon, Wenke Lee, and George Riley. Worm detection, early warning and response based on local victim information. In *20th Annual Computer Security Applications Conference (ACSAC)*, 2004.

[Han04] Christopher Hanna. Using snort to detect rogue IRC bot programs. Technical report, October 2004.

[Har02] John D. Hardin. The scanner tarpit howto. `http://www.impsec.org/linux/security/scanner-tarpit.html`, 2002.

[HC03] Neal Hindocha and Eric Chien. Malicious threats and vulnerabilities in instant messaging. Technical report, Symantec, October 2003.

[HKYH02] Petter Holme, Beom Jun Kim, Chang No Yoon, and Seung Kee Han. Attack vulnerability of complex networks. *Phys. Rev.*, E65(056109), 2002.

[JXWS05] Xuxian Jiang, Dongyan Xu, Helen J. Wang, and Eugene H. Spafford. Virtual playgrounds for worm behavior investigation. Technical Report CERIAS Technical Report (2005-24), Purdue University, February 2005.

[Kal00] C. Kalt. Internet relay chat: Architecture. http://www.faqs.org/rfcs/rfc2810.html, 2000.

[KE03] Darrell M. Kienzle and Matthew C. Elder. Recent worms: A survey and trends. In *WORM'03: Proceedings of the 2003 ACM workshop on Rapid Malcode*, pages 1–10, New York, NY, USA, 2003. ACM Press.

[KKJB05] Srikanth Kandula, Dina Katabi, Matthias Jacob, and Arthur W. Berger. Botz-4-sale: Surviving organized ddos attacks that mimic flash crowds. In *2nd Symposium on Networked Systems Design and Implementation (NSDI)*, May 2005.

[KMT04] Kevin Killourhy, Roy Maxion, and Kymie Tan. A defense-centric taxonomy based on attack manifestations. In *International Conference on Dependable Systems and Networks (ICDS'04)*, 2004.

[Kre99] C.J. Krebs. *Ecological Methodology.* Benjamin/Cummings, 1999.

[LBMC94] Carl E. Landwehr, Alan R. Bull, John P. McDermott, and William S. Choi. A taxonomy of computer program security flaws, September 1994.

[LCC$^+$02] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker. Search and replication in unstructured peer-to-peer networks. In *ICS '02: Proceedings of the 16th international conference on Supercomputing*, pages 84–95, New York, NY, USA, 2002. ACM Press.

[Lis01] T. Liston. Welcome to my tarpit - the tactical and strategic use of labrea. `http://www.hackbusters.net/LaBrea/LaBrea.txt`, 2001.

[LJ97] Ulf Lindqvist and Erland Jonsson. How to systematically classify computer security intrusions. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, pages 154–163, 1997.

[LUR04] LURHQ. Zindos worm analysis. `http://www.lurhq.com/zindos.html`, 2004.

[MD05] Steven Murdoch and George Danezis. Low-cost traffic analysis of tor. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2005.

[Mil67] S. Milgram. The small world problem. *Psychology Today*, 2(60), 1967.

[Moo02] D. Moore. Code-red: A case study on the spread and victims of an internet worm. http://www.icir.org/vern/imw-2002/imw2002-papers/209.ps.gz, 2002.

[MPS$^+$03] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. Inside the slammer worm. *IEEE Magazine on Security and Privacy*, 1(4), July 2003.

[MSVS03] D. Moore, C. Shannon, G. M. Voelker, and S. Savage. Internet quarantine: Requirements for containing self-propagating code. In *Proceedings of the IEEE INFOCOM 2003*, March 2003.

[MVS01] David Moore, Geoffrey Voelker, and Stefan Savage. Inferring internet denial-of-service activity. In *Proceedings of the 2001 USENIX Security Symposium*, 2001.

[NA05] Shishir Nagarja and Ross Anderson. The topology of covert conflict. Technical Report UCAM-CL-TR-637, University of Cambridge, July 2005.

[NSW01] M.E.J. Newman, S.H. Strogatz, and D.J. Watts. Random graphs with arbitrary degree distributions and their applications. *Phys. Rev.*, E64(026118), 2001.

[Par04] Janak J Parekh. Columbia ids worminator project. http://worminator.cs.columbia.edu/, 2004.

[QPC$^+$02] L. Qin, C. Pei, E. Cohen, L. Kai, and S. Scott. Search and replication in unstructured peer-to-peer networks. In *16th ACM International Conference on Supercomputing*, 2002.

[RFH$^+$01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proceedings of the ACM Conference of the Special Interest Group on Data Communication (SIGCOMM)*, pages 161–172, August 2001.

[RFI02] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal*, 6(1), 2002.

[Sav05] Steven Savage. Large-scale worm detection. In *ARO-DHS Special Workshop on Malware Detection*, August 2005.

[SM04] Colleen Shannon and David Moore. The spread of the witty worm. *Security & Privacy Magazine*, 2(4):46–50, 2004.

[SMK$^+$01] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, San Diego, California, August 2001.

[SS03] S.E. Schechter and M.D. Smith. Access for sale. In *2003 ACM Workshop on Rapid Malcode (WORM'03)*. ACM SIGSAC, October 2003.

[sym04] Symantec internet security threat report. Technical report, Symantec, September 2004.

[The05] The Honeynet Project and Research Alliance. Know your enemy: Tracking botnets. `http://www.honeynet.org/papers/bots/`, 2005.

[VTRB97] Paul Vixie, S. Thomson, Y. Rekhter, and J. Bound. Dynamic updates in the domain name system (dns update). http://www.faqs.org/rfcs/rfc2136.html, 1997.

[Vun05] Joost Vunderink. Personal correspondence. June 2005.

[WPB04] Matthew M. Williamson, Alan Parry, and Andrew Byde. Virus throttling for instant messaging. Technical Report HPL-2004-81, HP Labs, April 2004.

[WPSC03] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham. A taxonomy of computer worms. In *2003 ACM Workshop on Rapid Malcode (WORM'03)*. ACM SIGSAC, October 2003.

[WS98] D.J. Watts and S.H. Strogatz. *Nature*, 393(440), 1998.

[XKO$^+$04] Yinglian Xie, Hyang-Ah Kim, David R. O'Hallaron, Michael K. Reiter, and Hui Zhang. Seurat: A pointillist approach to network security, 2004.

[ZGGT03] C. C. Zou, L. Gao, W. Gong, and D. Towsley. Monitoring and early warning for internet worms. In *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS'03)*, October 2003.

[ZGT02] C. C. Zou, W. Gong, and D. Towsley. Code red worm propagation modeling and analysis. In *Proceedings of 9th ACM Conference on Computer and Communications Security (CCS'02)*, October 2002.

[ZGT03] C. C. Zou, W. Gong, and D. Towsley. Worm propagation modeling and analysis under dynamic quarantine defense. In *Proceedings of ACM CCS Workshop on Rapid Malcode (WORM'03)*, October 2003.

[ZP00] Y. Zhang and V. Paxson. Detecting stepping stones. In *Proceedings of the 9th USENIX Security Symposium*, August 2000.

[ZTGC03] C.C. Zou, D. Towsley, W. Gong, and S. Cai. Routing worm: A fast, selective attack worm based on ip address information. Technical Report TR-03-CSE-06, Umass ECE Dept., November 2003.