Connected Colors:
Unveiling the Structure of Criminal Networks

Yacin Nadji[1], Manos Antonakakis[2], Roberto Perdisci[3], and Wenke Lee[1]

[1] College of Computing, Georgia Institute of Technology
{yacin, wenke}@cc.gatech.edu
[2] Damballa, Inc.
manos@damballa.com
[3] Department of Computer Science, University of Georgia
perdisci@cs.uga.edu

Abstract. In this paper we study the structure of *criminal networks*, groups of related malicious infrastructures that work in concert to provide hosting for criminal activities. We develop a method to construct a graph of relationships between malicious hosts and identify the underlying criminal networks, using historic assignments in the DNS. We also develop methods to analyze these networks to identify general structural trends and devise strategies for effective remediation through takedowns. We then apply these graph construction and analysis algorithms to study the general threat landscape, as well as four cases of sophisticated criminal networks. Our results indicate that in many cases, criminal networks can be taken down by de-registering as few as five domain names, removing critical communication links. In cases of sophisticated criminal networks, we show that our analysis techniques can identify hosts that are critical to the network's functionality and estimate the impact of performing network takedowns in remediating the threats. In one case, disabling 20% of a criminal network's hosts would reduce the overall volume of successful DNS lookups to the criminal network by as much as 70%. This measure can be interpreted as an estimate of the decrease in the number of potential victims reaching the criminal network that would be caused by such a takedown strategy.

1 Introduction

Many of today's cyber-security threats make use of globally reachable network hosts that support cyber-criminal activities. For example, drive-by downloads need reliable hosting to infect the visitors of compromised sites. Pay-per install providers [6] need available hosting to distribute malicious binaries. Botmasters need a mechanism to command their bots, often relying on networks of command and control servers to provide redundancy for their critical communication channel to the compromised machines.

To avoid single points of failure, the miscreants make heavy use of DNS to provide *agility* to their network operations, thus preventing trivial blacklisting and comprehensive remediation efforts from easily disabling their malicious network resources. For example, to provide redundancy to their critical malicious infrastructure, attackers often use numerous domain names that map to multiple

hosts. As the network infrastructure relocates to survive blacklists and other remediation tools, old domains drift to new hosts and new domains are registered. This agility leaves a trail of breadcrumbs in historic DNS assignments, allowing us to build networks of related malicious hosting infrastructures and measure the threat landscape more holistically.

In this paper, we study *criminal networks*, their infrastructure, and their relationships that provide hosting for one or more types of threats. A criminal network infrastructure is often comprised of bulletproof hosting providers (or rogue networks [32]), auxiliary hosting providers and/or large swarms of compromised machines. In order to perform effective takedowns, we must understand how criminal networks are structured.

In this study we aim to (1) unveil the key components of criminal network infrastructures used to carry out a variety of malicious activities (hosting phishing sites, botnet command-and-control servers, sending spam emails, etc.), and (2) analyze the discovered malicious network infrastructures to better understand what actions could be taken to dismantle them completely or to inflict significant damage to the adversaries' criminal operations.

To this end, we adopt the following high-level process. First, we construct a graph of known malicious infrastructure and use passive DNS data to link related hosting providers. Then, we use community finding algorithms over this graph to identify different criminal networks likely operated by separate groups of adversaries. Finally, we study the characteristics of the criminal networks to identify techniques that may be employed to enact effective takedowns.

Our study is separated into two parts: the first part describes criminal network infrastructure at a high-level (Section 4), whereas the second part presents four case studies of interesting criminal networks (Section 5). We identify a class of criminal networks that, based on their graph structure, could be easily taken down in general. In addition, we analyze a number of large criminal networks that present interesting complex structures. In instances where comprehensive takedowns are difficult due to the complexity of the network, we pinpoint the critical infrastructure that should be the focal point of a takedown effort to maximize the damage done to the criminal network.

Our paper makes the following contributions:

Criminal network construction We provide a lightweight methodology to organize and find relationships between malicious infrastructure by leveraging historic information related to their use of DNS. Using community finding algorithms, we identify distinct criminal networks in the form of graphs in a scalable way.

Network structure analysis We analyze the structure of the criminal networks using two simple graph measures: the *graph density* and the *eigenvector centrality* of its vertices. The graph density characterizes graphs to identify common structures seen in real-world criminal infrastructure. The eigenvector centrality is used to identify the critical vertices in a criminal network. Both the graph density and eigenvector centrality assist us in making an informed decision on

the most effective takedown strategies that fit the properties and structure of each criminal network.

Takedown analysis We perform an in-depth analysis of four case studies using the graph measures to determine the effectiveness of different takedown strategies on sophisticated criminal networks. We quantify the amount of damage that would be caused by these takedowns by estimating the potential loss in victims. This loss is estimated by measuring the decrease in the volume of successful client lookups to domains related to the target criminal network caused by de-registering domain names or blocking IP addresses. This provides a quantitative basis to determine the most effective takedown strategy for a given criminal network.

2   Related Work

Prior work has focused on identifying autonomous systems (AS) known to host a disproportionate amount of malicious activity [32, 28, 33]. The idea of network cleanliness [9] has been explored as a potential indicator for future sources of maliciousness based on the assumption that malicious infrastructures tend to group together. We show that, in general, most criminal networks span across multiple autonomous systems, which makes knowing the worst ASs a moot point with respect to performing a comprehensive takedown. Disconnecting an AS from the Internet is not an easy task, and it often does not prevent malicious hosting in the long-term [24]. Focusing on high-level network structures, such as autonomous systems, does not provide sufficient knowledge to perform comprehensive takedowns. In contrast, we focus on identifying the web of smaller-sized networks that work together to provide reliable malicious hosting. Criminal networks that span multiple ASs can be disabled or heavily damaged since we identify not only the malicious networks, but their relationships with others.

On the other end of the spectrum, analysis can be done on individual domains and IP addresses. For example, prior work has studied the infrastructure used to support Rogue AV campaigns [11], fast-flux service networks [17], online scam infrastructure [18], command and control (C&C) networks [7], C&C migration [1], drop-zone infrastructure [15], and pay-per install infrastructure [6]. We consider a campaign to be a collection of domain names and IP addresses that serve a single malicious purpose and are associated with the same threat type, e.g., botnet C&C, drop-zones, etc. These studies provide invaluable insight into the low-level structure of campaigns, but this information also does not suggest how to perform takedowns effectively. The complex structure of criminal networks makes understanding the relationships of the hosting networks essential with regards to takedowns.

Graph-based infrastructure work either represents flows between networks or simply uses the graph abstraction as a way of linking related information. Nagaraja et. al. [25] used game theory and network analysis to suggest effective attacks and defenses against networks and network connectivity. BotGrep [26]

identifies botnet communities using random walks to detect dense community structures. Intuitively, peers in a botnet would communicate with patterns distinct from the less structured global Internet. Leontiadis et. al. [19] examined flows from redirections to study the infrastructure used to support illegitimate online prescription drug stores. These approaches all make a simplifying assumption, and treat network structure as simple messaging networks: i.e., two vertices communicating through a connected path in the graph. Christin et. al. [8] built a graph where vertices are domains, bank accounts, and phone numbers and edges are drawn when they appear together in a fraud campaign. This *link analysis* does not follow the typical communication network example, but still yields fruitful results by providing a concrete structure to group related data. Our graph building methodology follows the latter approach in spirit, but also makes use of community finding and network analysis to identify interesting features in the discovered criminal networks.

3 Goals and Methodology

Our main objective is to identify the components of network infrastructures used to carry out a variety of criminal activities – such as hosting spam- and phishing-related sites, deploying botnet command-and-control servers, sending spam emails, etc. – and to analyze these malicious network infrastructures to better understand how they are organized and what level of effort would be necessary to take them down. Towards this end, we perform these steps:

1. Enumerate hosts that participate in malicious activities, and find *network relationships* between them.
2. Analyze the structure of these network relationships to identify independent *communities of hosts* that constitute distinct *criminal networks* likely controlled by separate groups of adversaries.
3. Investigate the *criminal network landscape* to identify broad commonalities between classes of criminal networks with respect to remediation strategies.
4. Pinpoint the *critical infrastructure* within a given criminal network that should be targeted during coordinated takedown efforts to increase the likelihood of success, or to *maximize the damage to the adversary.*

To bootstrap the process of enumerating hosts involved in malicious activities and find their relationships, we leverage a large passive DNS database [35], which stores historic records of domain name to IP mappings as observed from live network traffic, and a variety of private and public sources of known malicious domains and IPs (Section 3.1). We build an undirected graph where vertices correspond to malicious infrastructure and edges denote a historic relationship between two vertices based on passive DNS evidence. Finally, we apply an analysis based on *community finding* algorithms to identify distinct criminal networks, and we compute the *eigenvector-centrality* of nodes within a criminal network to assess their importance and qualitatively estimate how much potential damage their takedown may cause to the entire criminal network (Section 3.3).

3.1 Data Sources

To enumerate hosts involved in malicious network activities, we leverage a variety of private and public feeds of domain names and IPs known to have been used for malicious purposes. Since we aim to provide a general picture of criminal networks that may involve different types of criminal activities, we use several sources of information, such as URLs embedded in spam emails, network traces from malware dynamic analysis, lists of known C&C servers, IP blacklists, etc. For example, given a spam URL, we extract the related domain name and use a large passive DNS database to enumerate the set of IP addresses that were recently resolved from this spam-related domain name. Our passive DNS database is constructed from 16 months worth of DNS resolutions collected at a major North American ISP spanning seven different geographical locations and serving several million users.

Our spam feed [16] includes URLs extracted from spam emails captured by a large spam trap. The malware-related data sources are from eleven public blacklists [10, 20, 13, 14, 21, 31, 22, 34, 30, 3, 29] and one commercial malware dynamic analysis feed. The source of information related to C&C servers is an internal company feed comprising domain names and IPs related to known C&C network infrastructures.

To find the network relationships between the enumerated hosts, we leverage two functions that can be defined over passive DNS data:

- *Related historic IPs* (RHIP): given a domain name or set of domain names d, RHIP(d) returns the set of routable IP addresses that d has resolved to at some point in the past.
- *Related historic domains* (RHDN): given an IP address or a set of IP addresses ip, RHDN(ip) returns the set of domain names that have resolved to ip at some point in their history.

Essentially, we consider two hosts to be related if they can be linked via the RHIP and RHDN functions.

After constructing the criminal network graphs, we leverage a commercial threat categorization and attribution process to identify specific criminal operators and malware families that are known to be affiliated with the identified malicious network infrastructures.

3.2 Constructing Criminal Network Graphs

In this section, we describe the procedure we use to build our criminal network graphs, which we represent using undirected weighted graphs.

An undirected graph G is defined by its sets of vertices V and edges E. Edges are bi-directional and are assigned a weight between $[0, 1]$ that expresses the "strength" of the relationship between its endpoints. A graph is *complete* if all pairs of vertices are adjacent, and is *connected* if for all pairs of vertices $v_i, v_j \in V$ there exists a sequence of adjacent vertices connecting v_i and v_j. A *disconnected* graph is made up of multiple *components*, or subgraphs of G. If a

component contains only one vertex, it is called an *isolated component* [36]. A vertex represents a collection of 256 IP-addresses (a Class C network or /24) and an edge connecting two vertices denotes a historic relationship, according to passive DNS data, between two IPs in the respective Class C networks.
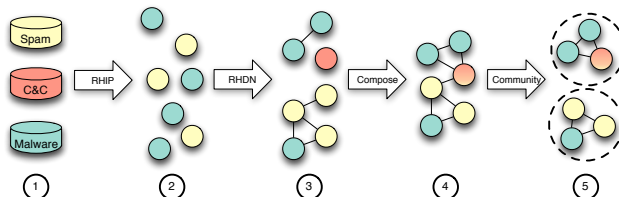

Fig. 1: Overview of process to generate criminal network graphs. Data sources are polled (1), domains are converted to IPs (2) and edges are drawn based on overlaps found in the passive DNS database (3). Different source type graphs are composed (4). Graphs are built and composed every day and community finding is performed to identify criminal networks (5).

A high level overview of the criminal network graph generation procedure is shown in Figure 1. Every day, the data sources are polled for new blacklisted network data (1). This network data comes in the form of known malicious IP addresses and domain names. Attackers are known to quickly migrate to new networks after takedowns [24], so in a deployed implementation we keep up with this drift by constantly adding newly discovered malicious network data. All malicious domain names are converted into IP addresses by looking up their related historic IP addresses (RHIP), and all of the IP addresses are binned into the Class C networks (2) that they belond to. Next, we look up each IP addresses' related historic domain names (RHDN) and edges are drawn between vertices when the intersection of their RHDN's is non-empty (3). If network hosts are found to be related to whitelisted domains, these IPs are removed to reduce the occurrences of non-malicious infrastructure in our graphs. Graphs from different sources are composed and edges are redrawn (4). Edges are weighted using the Jaccard index J, a ratio of the cardinalities of the intersection and union of two sets. Given two vertices v_i and v_j that are adjacent, their edge weight is defined by Equation 1,

$$J(v_i, v_j) = \frac{|D(v_i) \cap D(v_j)|}{|D(v_i) \cup D(v_j)|} \tag{1}$$

where $D(v)$ is the set of domains that historically point to IP addresses in vertex v. Graphs from multiple days are composed and community finding is used to identify criminal networks (5).

Whitelisting Our whitelist contains the top 10,000 Alexa domain names and domains of several popular content delivery and advertisement networks. The

whitelisting process works by examining the domain name sets generated by RHDN for every IP. Consider an IP ip, if its RHDN(ip) contains a domain that is whitelisted, or is a sub-domain of a whitelisted domain, we remove ip from our graph. For example, consider the domain name `doubleclick.net` which is used by Google's doubleclick advertising service. The top 10,000 Alexa *does not* contain doubleclick.net (only doubleclick.com), however, the IP that doubleclick.net resolves to, 216.73.93.8, has an RHDN set that contains doubleclick.com, which is whitelisted and the IP address 216.73.93.8 would be removed from our graph. If an attacker is aware of our whitelisting strategy there is little room for abuse. For an attacker to abuse our whitelisting strategy to evade our analysis, they would have to commandeer and point a whitelisted domain to their malicious infrastructure.

It is important to stress that we are seeking relationships between IPs as seen from the DNS, *not* from malware samples. For example, a given malware sample may intersperse its connection to its C&C server with spurious lookups to benign domains, these networks will not be connected unless there is an explicit relationship according to our passive DNS database.

Community Finding False positives can still be introduced, despite our whitelisting, which may cause edges to be drawn unnecessarily. For example, if a network host sinkholes multiple domains belonging to distinct criminal networks, our graph building process will erroneously show them as related. To address this problem in general, we leverage graph structure to identify the criminal networks using *community finding* algorithms.

The community finding process can automatically infer these scenarios based on the graph structure and correctly partition the underlying criminal networks. To perform community finding, we use the Louvain method [4], an algorithm known to scale well to graphs with hundreds of millions of vertices and billions of edges. We apply the community finding algorithm to each non-isolated component in our graph at step 5 of Figure 1.

3.3 Graph Analysis

Definitions: Understanding whether a graph is dense or sparse is a useful measure for summarizing graph structure. The *density* of a graph G, δ, is defined by $\delta = |E|/\binom{|V|}{2}$ and is the ratio of edges present in G to the number of possible edges in G. A graph with a density of 1 is complete and with a density of 0 has no edges. In our graphs, vertices are not of uniform importance, so quantifying the centrality of a vertex in a graph is a useful way of estimating the node's relative importance in the graph based on its structure. The *eigenvector centrality* (EC) is a measure of a vertex's centrality which often reflects its importance based on the graph's structure. Using EC, a vertex is considered important if it has many neighbors, a few important neighbors, or both. More formally, the eigenvector centrality x_i for a vertex i in a graph G is defined in Equation 2,

$$x_i = \kappa_1^{-1} \sum_j A_{ij} x_j \qquad (2)$$

where A is the adjacency matrix of G, κ_1 is its largest eigenvalue, $0 \leq x_i \leq 1$, and x_j are i's neighbors eigenvector centralities [27]. The EC is a useful metric for identifying "important" vertices in a graph independent of the underlying data being represented. We will use this to help determine a takedown strategy that attempts to maximize damage to a criminal network. Removing important vertices targets portions of the criminal network that are used both frequently and collectively to host the operations of multiple criminals.

Consider a social network, such as Facebook, where a vertex represents an individual and an edge drawn between two vertices represents a friendship. Vertices in this graph with high eigenvector centrality will be individuals with a large number of friends, a few friends that have many friends, or both. Similarly, high eigenvector centrality vertices in a criminal network graph are hosting providers that provide redundancy for many smaller hosting providers, a few larger hosting providers, or both. As an example, consider that a botnet operator could host her C&C server using a benign hosting provider, but when the C&C server is discovered, the diligent hosting provider will likely respond to abuse complaints and disable it. Thus, our operator uses a less scrupulous hosting provider to provide redundancy in the event of such a remediation attempt. One can imagine this behavior occurring in several criminals, and aggregated over time one would expect some kind of structure to emerge where the least scrupulous and most diligent hosting providers have the highest and lowest eigenvector centralities, respectively. This intuition suggests that targeting more structurally important vertices can help make takedown attempts more damaging to criminal networks.

There is an important caveat in the social network analogy that concerns connectivity. In a social network, removing social ties can sever friendships between individuals, but the same is not true in criminal networks. This is because nothing flows between connections in a criminal network in a literal sense, like friendship flows between mutual friendships. The assumption that does hold true is that someone with high social standing is likely to befriend additional high status individuals or several individuals en masse. Considering criminal networks, this means high eigenvector centrality networks are more likely to continue and expand their malicious activity into the future and therefore are where remediation efforts ought to be focused.

Simulating Takedowns Our ultimate goal is to determine how to perform effective and damaging takedowns of criminal networks. We first provide a bird's eye view of the criminal network landscape to search for recurring graph structures that are susceptible to takedowns. In other words, graph structures that lend themselves to comprehensive takedowns that require marginal effort. Next, we focus on specific cases of large criminal networks where we identify critical infrastructure to target during remediation to maximize the damage inflicted on a criminal network when a comprehensive takedown is prohibitively expensive.

Using the graph analysis measures we defined above, we identify potential weak points in a criminal network graph that may be susceptible to takedowns, and analyze how successful our takedowns would be by estimating the potential loss in future successful lookups. Not all criminal networks have the same structure, and some structures may be more or less amenable to different types of takedowns, such as taking down specific subnetworks or remediating groups of domain names affiliated with the network.

We consider the two main methods for takedown: *network-level takedown*, accomplished by raiding a hosting facility, or a *domain-level takedown*, accomplished by "revoking" domain names associated with the criminal network in cooperation with the domain names registrars. The goal of these takedown methods is to prevent potential victims from reaching key parts of the criminal network infrastructure.

To determine the order in which to take down infrastructure for a given criminal network G, we define the *criticality* of the vertices $v \in G$ by:

$$crit(v) = v_{ip} \times v_d \times v_{ec} \tag{3}$$

where v_{ip} is the number of malicious IPs within vertex v, v_d is the number of malicious domains that have pointed into v, and v_{ec} is the vertex's eigenvector centrality. The first two measures quantify the vertex's historic career of maliciousness and the eigenvector centrality quantifies the vertex's structural importance to the criminal network.

Input: M_D: a set of known malicious domains
Output: Returns, for each criminal network, the suggested order of networks to eliminate for performing a comprehensive takedown
$M_{\text{IP}} \leftarrow \text{RHIP}(M_D)$
$M_{\text{Net}} \leftarrow$ bin IPs in M_{IP} into Class C networks
$M_{\text{Net}} \leftarrow \forall_{v \in M_{\text{Net}}}$ remove v if $RHDN(M_{\text{Net}}) \cap$ whitelist $\neq \emptyset$
$E \leftarrow \{\}$
for $v_1, v_2 \in M_{\text{Net}}$ **do**
    if $RHDN(v_1) \cap RHDN(v_2) \neq \emptyset$ **then**
        $E \leftarrow E \cup (v_1, v_2)$
    end
end
$G \leftarrow (M_{Net}, E)$
$CriminalNetworks \leftarrow CommunityFinding(G)$
takedowns $\leftarrow \{\}$
for $subgraph \in CriminalNetworks$ **do**
    takedowns $\leftarrow$ takedowns $\cup$ sort descending by $\arg\max_{v \in subgraph} crit(v)$
end
return *takedowns*

Algorithm 1: High-level overview of how criminal networks are discovered and nodes are prioritized for takedown.

In an operational environment, takedowns would be performed based on the output of Algorithm 1. The system takes sets of known malicious domains and

outputs, for each identified criminal network, the nodes that should be targeted during a comprehensive takedown to maximize damage to the hosting infrastructure. The infrastructure used by the malicious domains are identified using the passive DNS database call to RHIP. These IPs are pruned using our whitelisting procedure and are grouped into their parent Class C (/24) networks. For each pair of networks, we identify domain name overlaps using the RHDN function. This identifies networks that share the burden of providing malicious infrastucture and if a takedown were desired, must be taken down *simultaneously* to perform a comprehensive takedown. The graph is partitioned using the described community finding algorithm to identify distinct criminal networks and by analyzing the graph structure we can determine which networks provide essential redundant hosting for criminal activity. Because malicious activity is so heavily distributed, targeting the worst individual hosting facility is insufficient. To perform comprehensive takedowns, one must consider the criminal network structure holistically, which motivates the use of the graph-based representation. It allows us to focus on the entire structure such that we can maximize the damage against the network.

For every criminal network in our case study, we order the vertices by their criticality using Equation 3 and estimate the benefit in taking down the criminal network using either network-level or domain-level takedowns. For each type of takedown, we present a cumulative distribution function (CDF) showing the proportion of domain names or networks removed from the criminal network against the total amount of potential victim lookups with respect to the entire criminal network. The intuition is that revoking domain names and blocking IP addresses that received a large volume of queries in the recent past has the potential of preventing a large fraction of the victim population from reaching the criminal network hosts in the future. If we successfully targeted critical infrastructure, the CDF will be superlinear denoting that eliminating key pieces of infrastructure severely impacts the lookups destined for the criminal network. If a strategy is unsuccessful, we should see linear/sublinear CDFs.

4   Threat Landscape

In this section, we present general observations about the graphs we built for our study. We discuss source type distributions and describe a case of a frequently occurring graph structure that could be easily taken down.

### 4.1   General Graph Statistics

Starting in May 2011, we began building graphs every day for a period of 8 months. Our final graph contains 64,030 vertices and 1,957,614 edges and represents 127,597 malicious IPs and 3,018,077 malicious domain names. The graph is disconnected, where 54% of the vertices are isolated components. These are threats that do not distribute their infrastructure using the DNS. As we mentioned earlier, many of these isolated components may also be due to false positives from non-distributed hosting not present in our whitelist. Figure 2a shows

a breakdown of threat types between isolated and non-isolated components. Most isolated vertices hosted spam sites or malware-related threats, and very few hosted any others. Our malware and spam sources are fundamentally noisy which, could explain the large difference between the isolated and non-isolated type distributions.

Since we are building our graphs with historical data, it is possible that originally bad IPs are remediated and used later on for legitimate purposes. If the new domains that resolve to the remediated IP space are whitelisted they will be removed from the graph, but if they are not they would still be flagged as malicious. To address this problem in future work, a shorter window of analysis can be used to reduce the likelihood of this behavior becoming commonplace.

4.2 Criminal Network Landscape

The remaining vertices form 4,504 distinct communities where each represents a criminal network. Of the 4,504 criminal networks identified, approximately 87% of them formed complete subgraphs. In addition to being complete, Figure 2b shows that most criminal networks contain few domains and second-level domains (2LD) and even fewer networks. In over half of the complete cases, a criminal network could be disabled by de-registering as few as five domain names or three 2LDs. This strongly suggests that a large number of small criminal networks can be easily remediated.
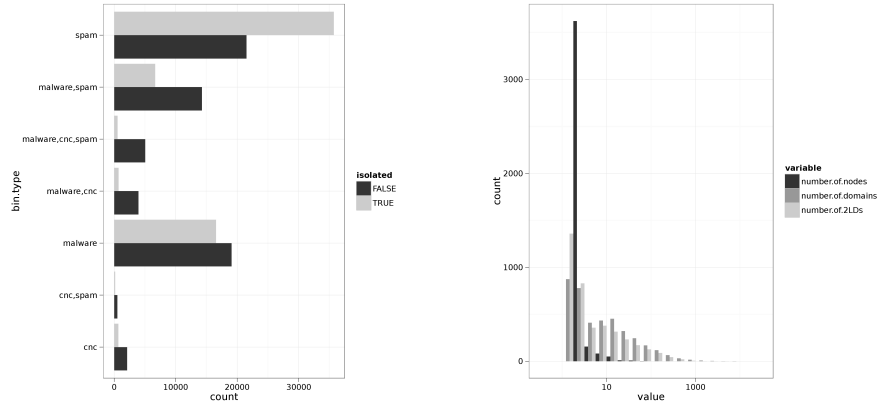
(a) Type breakdown-isolated vs. non-isolated. The y-axis represents the threat type seen in each vertex of our graph. Most host a single threat type (e.g., spam or malware), but many host multiple threat types, even reusing the same IP address (e.g., malware,spam, etc.).

(b) Log-scale distribution of the criminal network size, domains and 2LDs in complete criminal networks.

Fig. 2: Threat landscape breakdown

5 Case Studies

We describe four case studies of large and structurally interesting criminal networks that represent the different classes of infrastructure we saw in the wild. The case studies were not chosen automatically, but were chosen based on the visualizations of the output of our community finding algorithm described in Section 3.2. We used simple graph metrics to select the case student criminal networks by focusing on large graphs (e.g. many vertices) that had high and low graph densities. In all AS graph visualizations, vertex color encodes the autonomous system number while the vertex size encodes the number of known malicious domains that historically pointed into the network. Furthermore, the edges are drawn when one or more domains are shared between two vertices, unless otherwise specified. In all eigenvector centrality (EC) graph visualizations, vertex shade encodes the eigenvector centrality (darker is more important), and vertex size and edges are defined as they are for AS graphs, unless otherwise specified. The authors suggest that visualizations of the case studies be viewed in a PDF viewer if a high-resolution color printer is not available to get a clear view of the infrastructure.

For each criminal network presented, we provide a breakdown of the identified criminal operators using them as well as a breakdown of the sources polled to generate the vertices in the criminal network. Prior to investigating each case study, we were unaware of the underlying criminal affiliations. We will see that EC is a key factor we can use to dynamically obtain a metric for the critical vertices in the criminal network. As we noted in Section 3.3, EC is analogous to PageRank [5] for undirected graphs and provides a similar measure of the importance of a vertex in a graph.
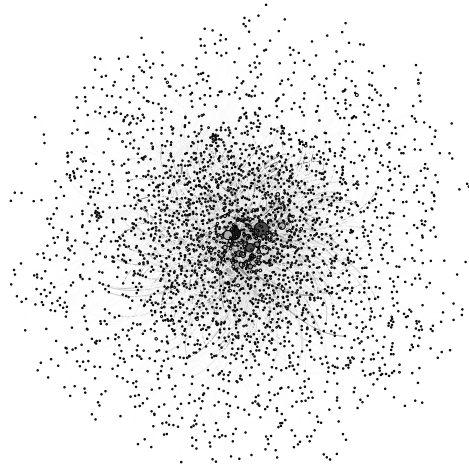
5.1 Rustock Criminal Network

Rustock criminal network was among the largest criminal networks we identified with 3,177 vertices and 7,128 edges. Rustock [23] was a large spam-oriented botnet generally used for fraudulent pharmaceutical sales. We describe the malicious hosting infrastructure used by Rustock and that was still in use during our study by other criminals.

Rustock criminal network's most distinguishing features can be seen in Figure 3a. It is sparse (graph density of 0.001) and the graph contains a dense core of networks that contain a large proportion of the domain names compared to the remaining vertices, shown by their larger size. In addition to the number of malicious domains they host, these vertices are also considered important based on their eigenvector centrality, shown in Figure 3b.

The top ASs by eigenvector centrality in the Rustock criminal network are shown in Table 1. This criminal network employs a mixture of bulletproof hosting, cloud-based hosting and compromised home user machines as part of its infrastructure. The inclusion of GoDaddy is due to parking sites the malicious domains pointed to before and/or after their malicious lifetime. CloudFlare is currently running sinkholes for Kelihos and most likely for other botnets as well,
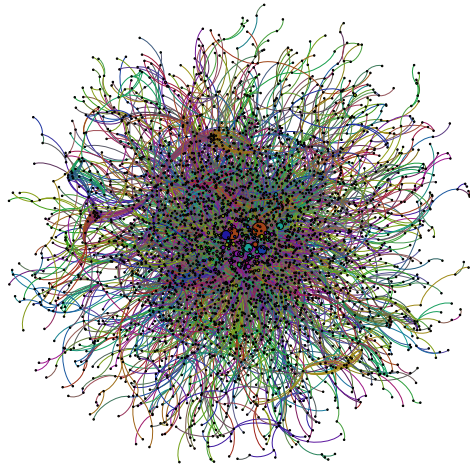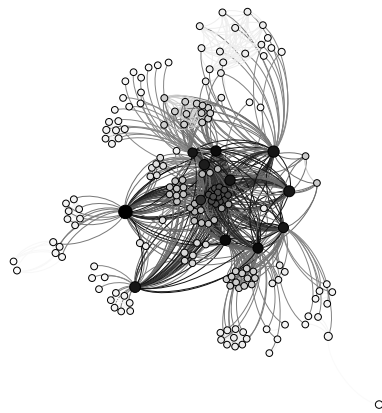
(a) Rustock criminal network AS graph
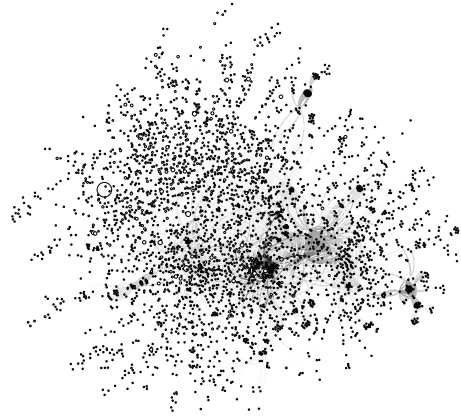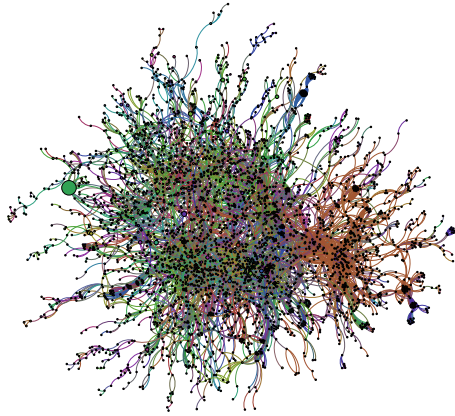
(b) Rustock criminal network EC graph
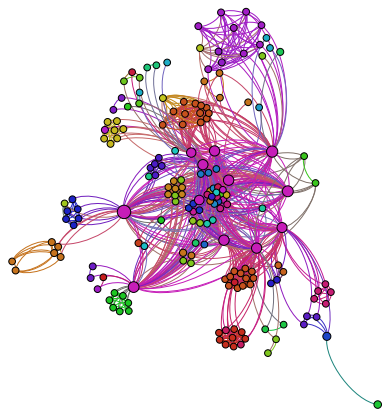
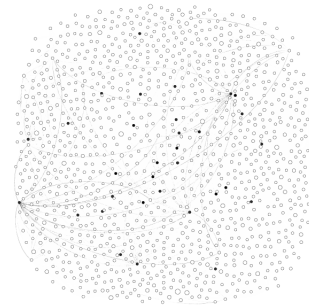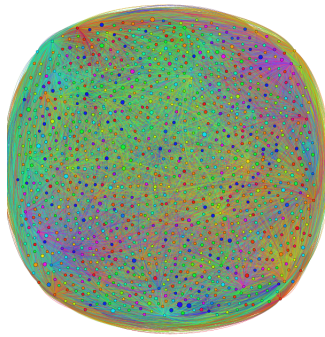(c) MojoHost benign hosting network AS graph

(d) MojoHost benign hosting network EC graph

Fig. 3: Case Study Visualizations [2]

(a) Masterhost criminal network AS graph (b) Masterhost criminal network EC graph

(c) Botnet criminal network
AS graph

(d) Botnet criminal network
Inverted EC graph

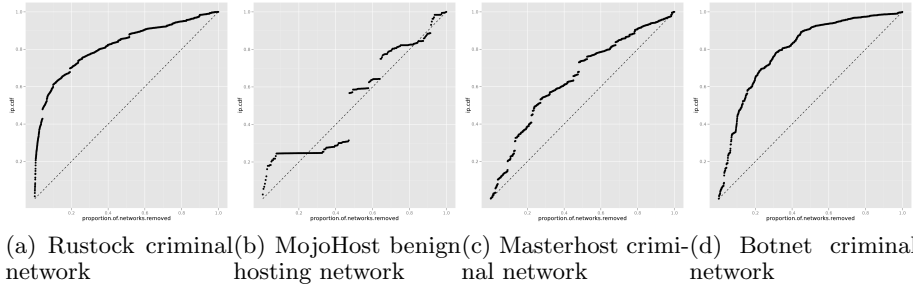Fig. 4: Case Study Visualizations cont.

(a) Rustock criminal network (b) MojoHost benign hosting network (c) Masterhost criminal network (d) Botnet criminal network

Fig. 5: Network-level takedown CDFs

AS#	AS Description	# of Domains
33626	Oversee	14,262
22489	Castle Access Inc.	124,321
15146	Cable Bahamas	55,465
13335	CloudFlare Inc.	21,770
16509	Amazon	6,772
32421	Black Lotus Communications	9,070
32592	Hunt Brothers	14,373
21844	The Planet	12,511
26496	GoDaddy	45,654
4635	Confluence Network Inc.	4,635

Table 1: Top 10 ASes in Rustock criminal network by eigenvector centrality

which would explain its high importance in this criminal network. Castle Access Inc. and Cable Bahamas are known to be used for domain parking monetization, which would explain their presence.

Rustock was taken down in March of 2011 (Operation b107), however the Rustock criminal network has facilitated other criminal operations until this day. This shows that single botnet takedown approaches can solve only the short term problem of a threat (i.e., spamming activity facilitated by Rustock botnet). In the case of Rustock criminal network, we saw that Internet abuse continued to use the same criminal infrastructure, as the Rustock botnet used to use, long after the botnet was taken off-line. During the 8 months of our experiment, we observed 4,381 new malicious domain names per day that began to use this criminal network.

5.2 MojoHost Benign Hosting Network

The MojoHost benign hosting network (Figure 3c) is an example of a benign hosting provider being abused by Internet miscreants for criminal infrastructure. We want to make the distinction clear that we are not saying MojoHost is complicit in criminal activity, but rather, malicious threats abuse MojoHost to build their criminal network. It is a smaller community of 255 vertices that has several distinct campaigns, the "orbiting" sub-communities, using it as infrastructure. The most structurally significant vertices are colored by their eigenvector centrality (Figure 3d). These 12 black vertices all belong to a single AS (AS27589) which provides redundancy for the malicious campaigns.

We identified seven distinct operators using the MojoHost benign hosting network for their malicious infrastructure, primarily to act as C&C servers. There were three distinct Zeus kit campaigns, two Blackhole exploit kit campaigns, and three unidentified malware family campaigns running C&C servers. In addition to C&C servers, the community was also home to three data exfiltration drop sites used by a mixture of Zeus instances. The Blackhole exploit kits facilitated drive-by downloads that infected victims with a Delf malware family instance, which is used to perform the second-stage of a two-stage binary drop. Most domains were registered through dynamic DNS providers which are commonly used in Blackhole exploit kit instances.

Despite the fact that the MojoHost community is benign, it presents an interesting hierarchical structure that would intuitively be fairly resistant against AS-level take downs. While the main support structure for the campaigns exists in a single AS, the orbiting communities are spread across 58 ASs in total. If a criminal network contained several layers in this hierarchical fashion, it would be difficult to cripple it quickly due to the redundancy. Maintaining this level of structure may prove to be difficult in scale, which may explain why criminal networks seen in practice are much less organized (Sections 5.1 and 5.4).

5.3 Botnet Criminal Network

This criminal network is a large botnet that provides fast flux services across 1,226 vertices, most of which belong to consumer dynamic IP address space. The graph is almost complete with a graph density of 0.956 (see Figure 4c). It is in the botnet operator's best interest to keep this structure as it maximizes the redundancy of the vertices using DNS agility. Since the graph is nearly complete, it is reasonable to assume that most of the vertices are of about equal importance. The eigenvector centrality, however, reveals interesting underlying structure by highlighting the vertices considered less important to the overall criminal network. In Figure 4d, we see the eigenvector centrality graph where the vertex shading is inverted (darker is **less** important in this case), which highlights 32 vertices within the botnet's sub-structure that are used for other purposes. Specifically, these vertices with lower than normal EC appears to be C&C servers and data exfiltration drop sites for Zeus v2 (a.k.a. Zeus Group B) and Blackhole kit generated malware for a single operator. In this case it is important to note that the only way to truly disable the network is to target the central nodes. Eliminating lower centrality nodes would quickly disable the smaller campaigns contained within, but would not cause damage to the larger criminal network, which is the focus of this paper. Furthermore, significant portion of the domain names in this botnet are related with FakeAV/RogeAV type of threats. One of the main differences of the FakeAV threats facilitated by this criminal network is that they are primarily delivered by search engine optimization poisoning techniques.

Botnet criminal networks are likely to present themselves as dense or complete graphs with a relatively uniform eigenvector centrality distribution due to the fundamental nature of how they are operated by criminals. Furthermore, by

looking for vertices that are considered less important by centrality measures, we may identify underlying substructures that differ in function.
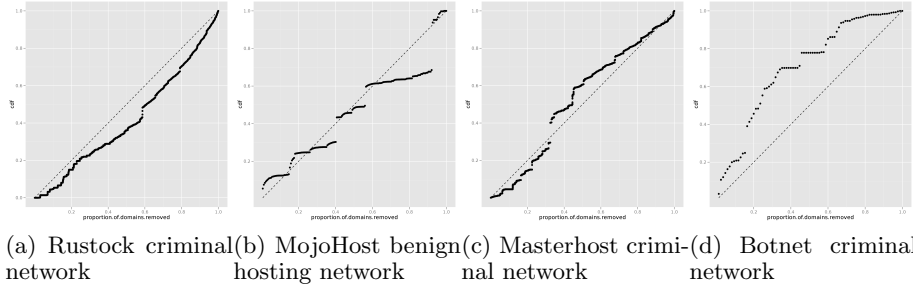

(a) Rustock criminal network (b) MojoHost benign hosting network (c) Masterhost criminal network (d) Botnet criminal network

Fig. 6: Domain-level takedown CDFs

5.4 Masterhost Criminal Network

At 3,725 vertices and 11,519 edges, the Masterhost criminal network is the largest criminal network we identified during our study (Figure 4a). Much like the Rustock criminal network, the Masterhost criminal network is very sparse (graph density of 0.002), but the densely malicious networks are missing from the center. In this criminal network, dense vertices are *not* considered structurally important as shown by Figure 4b. This means that the malicious domains contained within these dense structures are *not* heavily replicated throughout the criminal network, making these good candidates for AS-level takedowns.

AS#	AS Description	# of Domains
25532	Masterhost	12,281
21788	Network Operations Center Inc.	3,692
3561	Savvis	3,285
7303	Telecom Argentina	2,830
32613	iWeb Technologies	2,684
21740	eNom, Inc.	2,292
25847	ServInt	2,275
16509	Amazon Inc.	2,254
7788	Magma Communications Ltd.	2,225
6939	Hurricane Electric, Inc.	2,201

Table 2: Top 10 ASes in Masterhost criminal network by number of malicious domains

The top 10 ASes by number of hosted malicious domains in the Masterhost criminal network are shown in Table 2. Notice the number of domains per AS is substantially smaller than it was for the Rustock criminal network due to the lack of centralized malicious hosting. The biggest AS, with the respect of the domain names that facilitate resolutions for, is the "Masterhost". Masterhost is a very well known bulletproof network that has been identified by the security

community since 2007 and it is highly related with the Russian Business Network organization [12]. In the 8 months of our experiments, we observed a median of 1,065 new malicious domain names every day that began to use the Masterhost criminal network.

5.5 Simulating Takedowns

Using Equation 3, we identify critical vertices in the case study networks and simulate takedowns by producing the network-level and domain-level takedown CDFs in Figure 5 and Figure 6, respectively. These CDFs show the proportion of networks or domain names removed from the criminal network against the loss in the total amount of potential victim lookups that were made to the entire criminal network. Successful takedowns will manifest as superlinear CDFs, denoting that we can eliminate many potential victims by selectively removing few critical vertices in the criminal network. The aggregate DNS lookup volume to the malicious infrastructure proxies the potential loss in victim population; intuitively, infrastructure that is queried frequently is likely to cause the greatest problems to the attacker if it is taken down. In the two largest cases, the Rustock criminal network and Masterhost criminal network, we see the network-level takedowns are very effective (Figure 5a/5c). In the Rustock criminal network, removing only 20% of the criminal network infrastructure decreases to total number of lookups by 70%. In the Masterhost criminal network, we can decrease total lookups by 40% by focusing our takedown efforts on the worst 20% of the networks. Recall from Figures 3b and 4b that the Rustock criminal network had a dense core of dedicated malicious hosting, while the Masterhost criminal network did not. This would explain the difference in takedown performance between the two criminal networks. Figures 6a and 6c show that domain-level takedowns for these two criminal networks are ineffective, based on the sublinear and linear CDFs. Intuitively, this makes sense as the graphs are very sparse. A single domain name is unlikely to substantially damage the infrastructure because the domain names are less distributed.

Figures 5b and 6b illustrate the difficulty in taking down a well structured network seen in the MojoHost benign hosting network. Since the underlying network infrastructure is benign, the miscreants abusing MojoHost must take great care in distributing their malicious activities, which makes takedowns more difficult. This also suggests that creating hierarchical criminal networks resilient against takedowns is possible, but we did not find these structures in the wild.

For the Botnet criminal network, both network-level (Figure 5d) and domain-level (Figure 6d) takedowns were successful; eliminating 40% of the networks or domains associated with the botnet caused an 80% and 70% decrease in total lookups, respectively. Since Botnet criminal network has a much higher graph density than the other case studies, it makes sense that the domain-level takedown would be effective. However, understanding the success of the network-level takedown requires an understanding of the type of threats the network facilitates the hosting infrastructure for. Most of the malicious hosting that uses the Botnet criminal network are for C&C servers, which need to be highly available.

This availability requirement causes the dense structure, which lowers the discriminatory function of the EC metric as most nodes will be considered highly important. Our selection process compensates for this by targeting networks densely populated with malicious domain names and IPs.

6 Conclusion

In this paper, we proposed a graph-based method to representing criminal network infrastructures and unveiling their key components. Furthermore, we proposed an approach to analyze the graph properties of malicious network infrastructures and better understand what actions could be taken to dismantle them completely or to inflict significant damage to the adversaries' criminal operations. We showed that in many smaller criminal networks, their network graph structure and domain name distribution make complete takedowns possible, by revoking the domains associated with the criminal network with the help of the domain registrars. In more complex cases, we provided three key metrics that can identify critical components of a criminal network, and quantified the effectiveness of our suggested takedown measures.

References

1. M. Abu Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A multifaceted approach to understanding the botnet phenomenon. *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 41–52, 2006.
2. M. Bastian, S. Heymann, and M. Jacomy. Gephi: An Open Source Software for Exploring and Manipulating Networks. In *International AAAI Conference on Weblogs and Social Media*, 2009.
3. T. Bates, P. Smith, and G. Huston. CIDR report bogons.
4. V. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008.
5. S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.
6. J. Caballero, C. Grier, and C. Kreibich. Measuring Pay-per-Install: The Commoditization of Malware Distribution. In *Proceedings of the USENIX Security Symposium*, 2011.
7. C. Cho, J. Caballero, and C. Grier. Insights from the inside: A view of botnet management from infiltration. In *Proceedings of the USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2010.
8. N. Christin, S. S. Yanagihara, and K. Kamataki. Dissecting one click frauds. In *Proceedings of the 17th ACM Conference on Computer and Communiations Security (CCS)*, 2010.

9. M. Collins, T. Shimeall, S. Faber, J. Janies, R. Weaver, and M. D. Shon. Predicting future botnet addresses with uncleanliness. In *Proc. of IMC*. CERT Network Situational Awareness Group, 2007.

10. A. D. Correa. Malware patrol.

11. M. Cova, C. Leita, O. Thonnard, A. Keromytis, and M. Dacier. An analysis of rogue AV campaigns. In *Recent Advances in Intrusion Detection*. UCSB, Columbia, 2010.

12. dn1nj4. RBN "Rizing". Technical report, Shadowserver.org, 2008.

13. DNS-BH. Malware prevention through DNS redirection.

14. dnsbl.abuse.ch. dnsbl.abuse.ch.

15. T. Holz, M. Engelberth, and F. Freiling. Learning more about the underground economy: A case-study of keyloggers and dropzones. In *Computer Security– ESORICS 2009*, 2010.

16. Internet Systems Consortium. Security Information Exchange Portal.

17. M. Konte, N. Feamster, and J. Jung. Fast flux service networks: Dynamics and roles in hosting online scams. Technical report, 2008.

18. M. Konte, N. Feamster, and J. Jung. Dynamics of online scam hosting infrastructure. In *Passive and Active Network Measurement*. Georgia Tech and Intel Research, 2009.

19. N. Leontiadis, T. Moore, and N. Christin. Measuring and analyzing search-redirection attacks in the illicit online prescription drug trade. In *Proceedings of the USENIX Security Symposium*, August 2011.

20. L. Lu, V. Yegneswaran, P. Porras, and W. Lee. BLADE: an attack-agnostic approach for preventing drive-by malware infections. In *Proceedings of the 17th ACM Conference on Computer and Communiations Security (CCS 2010)*. Georgia Tech, SRI International, 2010.

21. Malc0de. Malc0de DNS blacklist.

22. Malware Domain List. Malware domain list.

23. D. McCoy, A. Pitsillidis, G. Jordan, N. Weaver, C. Kreibich, B. Krebs, G. M. Voelker, S. Savage, and K. Levchenko. Pharmaleaks: Understanding the business of online pharmaceutical affiliate programs. In *21st Usenix Security Symposium (USENIX 2012)*, 2012.

24. R. McMillan. After takedown, botnet-linked ISP Troyak resurfaces, 2010.

25. S. Nagaraja and R. Anderson. The topology of covert conflict. In *Workshop on the Economics of Information Security (WEIS)*, 2006.

26. S. Nagaraja, P. Mittal, C.-Y. Hong, M. Caesar, and N. Borisov. Botgrep: finding p2p bots with structured graph analysis. In *Proceedings of the 19th USENIX conference on Security*, USENIX Security'10, pages 7–7, Berkeley, CA, USA, 2010. USENIX Association.

27. M. Newman. *Networks: An Introduction*. Oxford University Press, USA, 1 edition, May 2010.

28. F. Roveta, L. D. Mario, F. Maggi, G. Caviglia, S. Zanero, and P. Ciuccarelli. BURN: Baring Unknown Rogue Networks. In *VizSec*. Politecnico di Milano, 2011.

29. Snort Labs. Snort DNS/IP/URL lists.

30. SpamHaus. drop.lasso.

31. SpyEye Tracker. SpyEye tracker.

32. B. Stone-Gross, C. Kruegel, K. Almeroth, A. Moser, and E. Kirda. Fire: Finding rogue networks. In *ACSAC*. UCSB, Technical University Vienna, Eurocom, 2009.

33. P. Stranger, J. McQuaid, S. Burn, D. Glosser, G. Freezel, B. Thompson, and W. Rogofsky. Top 50 Bad Hosts and Networks. *Tech Report*.

34. Team Cymru. Bogons.

35. F. Weimer. Passive DNS replication. In *17th Annual FIRST Conference on Computer Security Incidents*, 2005.

36. D. B. West. *Introduction to Graph Theory (2nd Edition)*. Prentice Hall, 2 edition, Sept. 2000.