

# FIRE: FInding Rogue nEtworks

Brett Stone-Gross, Christopher Kruegel, Kevin Almeroth  
University of California, Santa Barbara  
{bstone, chris, almeroth}@cs.ucsb.edu

Andreas Moser  
Technical University Vienna  
andy@iseclab.org

Engin Kirda  
Institute Eurecom  
kirda@eurecom.fr

## Abstract

*For many years, online criminals have been able to conduct their illicit activities by masquerading behind disreputable Internet Service Providers (ISPs). For example, organizations such as the Russian Business Network (RBN), Atrivo (a.k.a., Intercage), McColo, and most recently, the Triple Fiber Network (3FN) operated with impunity, providing a safe haven for Internet criminals for their own financial gain. What primarily sets these ISPs apart from others is the significant longevity of the malicious activities on their networks and the apparent lack of action taken in response to abuse reports. Interestingly, even though the Internet provides a certain degree of anonymity, such ISPs fear public attention. Once exposed, rogue networks often cease their malicious activities quickly, or are de-peered (disconnected) by their upstream providers. As a result, the Internet criminals are forced to relocate their operations.*

*In this paper, we present FIRE, a novel system to identify and expose organizations and ISPs that demonstrate persistent, malicious behavior. The goal is to isolate the networks that are consistently implicated in malicious activity from those that are victims of compromise. To this end, FIRE actively monitors botnet communication channels, drive-by-download servers, and phishing web sites. This data is refined and correlated to quantify the degree of malicious activity for individual organizations. We present our results in real-time via the website [maliciousnetworks.org](http://maliciousnetworks.org). These results can be used to pinpoint and to track the activity of rogue organizations, preventing criminals from establishing strongholds on the Internet. Also, the information can be compiled into a null-routing blacklist to immediately halt traffic from malicious networks.*

## 1. Introduction

Anecdotal evidence indicates the existence of Internet companies and service providers that are under the influence of criminal organizations or knowingly tolerate their activities. Such companies typically control a number of networks with public IP addresses that are

abused for a wide range of malicious activities. One such activity is offering bullet-proof hosting, a service that guarantees the availability of hosted resources even when they are found to be malicious or illegal. These hosting services are often used for phishing purposes or for serving exploits and malware. Other malicious activities involve the sending of spam, hosting scam pages, or providing a repository for pirated software and child pornography.

An example of a rogue network that offered bullet-proof hosting was the Russian Business Network (RBN), who made headlines in late 2007 [5], [16]. Various sources alleged that the RBN hosted web sites, exploits, and malware that were responsible for a significant fraction of online scams and phishing. Once publicly exposed, the RBN ceased its operations in St. Petersburg, only to relocate and resume activities in different networks [10]. More recently, a report exposed Atrivo (Intercage), a US-based company that is frequently considered to provide hosting for malicious content [3], [17]. Often referred to as the RBN of the United States, this company is considered to be a “dedicated crime hosting firm whose customer base is composed almost, or perhaps entirely, of criminal gangs” [13]. Shortly after Atrivo made headlines, two more rogue networks, known as McColo and the Triple Fiber Network (3FN), were discovered to be major hosting providers for malicious content with ties to cybercrime [1], [2], [18]. Again, public outcry quickly lead reputable ISPs to sever their peering relationships with these organizations, cutting them off the Internet.

In this paper, we describe *FIRE*, a system that monitors the Internet for rogue networks. We believe that it is important to expose such networks, for a number of reasons. First, as the examples of the Russian Business Network, Atrivo, McColo, and 3FN demonstrate, criminals fear public attention. As a result of the increased media coverage, all four networks had to cease their immediate activity. In many cases, it is likely that their operations resumed elsewhere. However, it took some time before the miscreants could restructure their setup. Thus, by quickly bringing to light networks that act maliciously, it becomes more difficult for cyber-criminals to establish a home base.

The second advantage of identifying rogue networks is the possibility to generate blacklists that can block all traffic from a netblock, even when certain IPs within this netblock have not yet acted maliciously. This approach prevents criminals from cycling through the available IP space, quickly shifting to a new IP when a current host is blacklisted. Currently, there are manual efforts underway to establish blacklists based on the observation that certain networks are malicious. For example, Spamhaus maintains the *Don't Route Or Peer (DROP)* list, a collection of networks that they consider to be controlled entirely by professional spammers. Spamhaus suggests that traffic from these sources should simply be dropped, and recommends the use of this list by tier-1 ISPs and backbone networks. Another example is the list maintained by EmergingThreats, which identifies netblocks that are thought to belong to the Russian Business Network. While such efforts are beneficial, they are expensive and tedious to maintain. Moreover, these lists are often incomplete and limited in scope (for example, limited to spam operations or the RBN in particular). In contrast, *FIRE* operates in an automated fashion, and we aim to capture a broader range of malicious activity, independent of any *a priori* knowledge of criminal organizations.

To identify rogue networks, we rely on a number of data sources that report the malicious actions of individual hosts. Some of the data feeds are publicly available, such as lists of phishing web pages. The other data originates from our own analysis efforts, such as a list of hosts that provide botnet command and control servers and hosts that are found to exploit browser vulnerabilities. Of course, given the widespread use of botnets and the large number of exploited machines, the fact that a host performs malicious actions is no immediate indication that the corresponding ISP or netblock is malicious. Instead, when a host misbehaves, it is possible that attackers were able to compromise and abuse it for nefarious purposes. Thus, it is necessary to search the data for indicators that allow us to distinguish between hosts under the control of rogue (or grossly negligent) ISPs and infected machines of organizations that make a deliberate effort to keep their network clean.

Based on post-processed information obtained from different data sources, we compute a *malscore* (maliciousness score) for individual ASNs (Autonomous System Number). This score quantifies the amount of recent, malicious activity in a network and serves as an indicator for the likelihood that an ASN is linked to cyber-criminals, or at the least, being very negligent in removing malicious content. Using the malscores, it is easy to identify the worst offenders on the Internet and take appropriate actions (such as increasing the public

pressure, breaking peering relationships, or putting their IP address space on a blacklist). Moreover, we can track malicious activity over time.

The main contributions of this paper are as follows:

- We analyze a number of data sources to identify IP addresses of hosts that misbehave in different ways.
- We present techniques to filter these lists for hosts that likely belong to rogue ISPs. In particular, we combine the information from different data sources to compute a *malscore* that quantifies the malicious activities of an autonomous system.
- We show that our system is successful in identifying a number of rogue ISPs and can assist legitimate ISPs in cleaning their networks via our website [maliciousnetworks.org](http://maliciousnetworks.org).

## 2. System Overview

The goal of our system is to identify rogue networks. Thus, we first need to concretize what we consider to be a rogue network. Unfortunately, this question is not straightforward to answer. Some service providers are simply lax when it comes to the content that they offer, others are victims of remote exploits, and a few are well-known to blatantly host malicious content. Thus, the fact that a network is the source of unwanted activity does not necessarily qualify it immediately as being malicious.

We consider a rogue network to be a network that is under the control of cyber-criminals or that knowingly profits from cooperating with criminals. Of course, it is difficult to assert such criminal ties without thorough investigations by law enforcement agencies. Thus, we have to redefine our notion of rogue networks based on the activities that are typically associated with such networks. To this end, we consider a rogue network to be one in which significant malicious activity occurs. In addition, this activity lasts for an extended period of time, regardless of abuse complaints. Our logic behind this is that rogue networks provide hosting for malicious content that often remains up for many days (sometimes even months or years). In contrast, malicious activity in other networks tends to be more short-lived due to abuse reporting and honest attempts to undo the damage.

Given our notion of rogue networks, the basic idea to identify such networks is to check for the presence of a large number of long-lived, misbehaving hosts. To this end, we analyze a number of data sources for IP addresses that have exhibited malicious behavior for an extended period of time (the exact extent of this time span depends on the type of data source and is discussed later).

### 3. Data Collection

In this section, we discuss in more detail the three data sources that we use to identify hosts that likely belong to rogue networks. To this end, we first describe, for each data source, how we obtain the IP addresses of hosts that are *actively* engaged in malicious activity.

#### 3.1. Botnet Command and Control Providers

Despite the emergence of peer-to-peer-based bots, many botnets still rely on centralized command and control (C&C). For this C&C infrastructure, botmasters typically set up IRC servers that provide channels for bots to join, or web servers that can be periodically polled for new commands. The functioning of the complete botnet depends on the availability of these servers. Thus, a botmaster is interested in hosting his C&C infrastructure on a network where it is safe from takedown.

To identify and monitor the networks affiliated with botnet C&C servers, we utilize data collected from Anubis [4]. Anubis executes Windows-based malware binaries in a virtual environment and records file system and registry modifications, process information, and network communications. We are particularly interested in the network traffic (if any) generated by the malware.

**IRC-based botnets.** When Anubis monitors IRC traffic the corresponding nickname, server, and channel information is logged. To monitor whether IRC C&C channels are active, we use a custom IRC client that leverages the recorded credentials to connect to the IRC server and join the channel. Because we are primarily interested in the longevity of the C&C server, we resolve the C&C server's host name to one or more IP addresses, and then connect to each IP at regular intervals. When the C&C server is not identified by a DNS name but by an IP address, then this address is used directly. A host (an IP address) is considered to be active when our client can join the corresponding C&C channel. Sometimes, transient network problems prevent us from connecting to a host. In such cases, it would be undesirable and premature to declare a host as inactive. Thus, we require that an active C&C channel is unreachable for two days before declaring the corresponding IP address as inactive.

Interestingly, in a number of cases, we observed that a channel (and the corresponding server) was reachable, but no malicious activity was noticeable. This is frequently the case when a bot channel is created on a well-known IRC network (such as undernet or efnet). The reason is that the IRC administrators of these networks quickly ban the botmaster and remove

the channel. However, subsequent logins from bots or other users reopen the channel, thus making the channel available and leaving the impression that it is still active. To mitigate this problem, we modify our approach to determine whether a botnet C&C host is active. More precisely, in addition to the requirement that a server is reachable and the appropriate channel exists, we also require that the channel shows bot-related activity. To this end, we introduce heuristics that check the messages and channel topics for well-known IRC bot commands (such as *download*, *update*, *dos*) and signs of encoded or encrypted commands. A channel is considered up only when such indicators are present.

**HTTP-based botnets.** To identify and monitor web-based botnet C&C servers from samples collected by Anubis, we first require a mechanism to distinguish between legitimate HTTP traffic and traffic related to botnet commands. This is necessary because HTTP traffic sent by a malware sample does not immediately imply a connection to a C&C server (HTTP connections are often used to check for network connectivity or download updates). To identify HTTP C&C traffic, we manually define static, malicious characteristics (signatures) of requests used by well-known botnets. These characteristics include content from the HTTP request path and parameters, HTTP headers and POST data, and the HTTP response from the web server. Such static features are useful even for botnets that use encryption because they frequently send an encryption key, bot identifier, version number, and other parameters to the web server. Thus, the HTTP C&C server must know how to parse the request in a specific format.

As an example of a web-based botnet that we have been monitoring, consider Pushdo/Cutwail, which is believed to be one of the largest, active botnets used for spam. When a Cutwail bot connects to the C&C server, it will often request one or more executables. Although the botnet utilizes encryption, the request path for these binaries contains a predictable semi-static format, such as the prefix `/40E8`. The response from the web server contains one or more executables typically around 100KB. Currently, we are monitoring 24 different types of web-based botnets including Coreflood, Torpig, and Koobface.

#### 3.2. Drive-by-Download Hosting Providers

Our second data source is a list of servers that host malware executables distributed through drive-by-download exploits. Drive-by-downloads are a means of malware distribution where executables are automatically installed on victim machines without user interaction. Typically, the only requirement is for a

user to visit a web page that contains an exploit for her vulnerable browser. In some cases, the exploit and the malware executable is hosted on a compromised host, while in other cases, a compromised web page is only used to redirect the victim to a second machine that performs the exploit (often referred to as a mothership). These mothership servers are frequently located in rogue networks.

There are three data feeds that we use to identify drive-by-download servers. The first feed is through Wepawet [25], a system that checks user-submitted web pages (URLs) for malicious Javascript. In particular, we are interested in cases where malicious script contains shellcode that downloads and executes malware. When malware is discovered, Wepawet records the locations of these binaries and exports them to *FIRE*. The second data feed is through a daily compilation of URLs found in spam mails that are caught in the spam traps of a computer security company and an Internet Service Provider. The third feed is a daily-updated list of “spamvertised” URLs (advertised via spam) provided by Spamcop [23]. So far, after eliminating duplicates, we have recorded more than 1.2 million spamvertised links. Of course, not every URL in a spam email points to a site that launches a drive-by-exploit. Instead, these URLs frequently lead to shady businesses such as online pharmacies, casinos, or adult services. To identify those sites and pages that actively perform drive-by-exploits, we use the Capture Honey Pot Client (HPC) [21]. Capture is able to find web-based exploits by opening a potentially malicious web site in a browser on a virtual machine. After visiting a page, the state of the virtual machine is inspected and suspicious changes (i.e., the creation of new files or the spawning of new processes) are recorded, as they indicate that the guest system was compromised by a web-based exploit.

For our analysis, we use a total of eight virtual machines (VMs) dedicated to scanning web pages. All VM images are running Windows XP Professional (Service Pack 2), without any patches installed and automatic updates disabled. To catch recent exploits, we have installed the Flash and Quicktime plug-ins.

When the Capture honey client is compromised by visiting a certain URL, we inspect the network traces recorded from Capture HPC. We are not interested in the server that hosts the web site that contains an exploit. We have observed that those machines are often legitimate web servers that are victims of compromise and, therefore, do not yield much information about malicious networks. Thus, if the malicious binary that is part of an exploit is downloaded from the same server, we ignore that host for our analysis. In the more interesting case, an exploit has been injected into a web

page and the associated binary is hosted on a different machine (mothership server that usually serves binaries for many different exploits). Due to the importance of this mothership servers for the criminals behind the exploit, these machines are often located in malicious networks where the chance that it is being shut down is low. Thus, we only consider the IP addresses of those mothership servers for our analysis. Once we have discovered a download server, we revisit it once per day.

### 3.3. Phish Hosting Providers

The third data source to identify rogue networks is derived from information about servers that host phishing pages. Typically, phishing pages are set up to steal login credentials, credit card numbers, or other personal information. Often, these pages are hosted on compromised servers and are taken down quickly. To mitigate this problem, phishers often resort to hosting their phishing pages directly in networks where there is little or no control of the offered content.

To locate phishing pages, we leverage an XML feed provided by PhishTank [19]. Once a day, this feed provides our system with URLs of phishing pages that are verified by the PhishTank community. Interestingly, all URLs on the PhishTank list are considered to be online. However, our experiments have shown that phishing pages are often taken offline so quickly that the list is already outdated after one day.

To compute the status of phishing IPs, we attempt to download the web page located at a given phishing URL once per day. This is done until either the domain (of the URL) can no longer be resolved, or the site is offline for more than one week. A phishing site is considered offline by our system when the web server is not reachable anymore or when the phishing page has been replaced by another page that is not a phish (usually a HTTP 404 error page or a phishing warning page).

## 4. Data Analysis

In this section, we discuss our techniques to identify rogue networks and compute their malscores based on the analysis of the individual data sets that we collect.

### 4.1. Longevity of Malicious IP Addresses

The primary characteristic that distinguishes between rogue and legitimate networks is the longevity of the malicious services. Most legitimate networks are able to clean up illicit content within a matter of days. In contrast, we have observed malicious content that

has been online for the entire monitoring period of more than a year. Figure 1 shows the average uptime of malicious IPs per ASN. It can be seen that the vast majority of networks remove the offending content in less than ten days. However, there were 361 ASNs that had hosts with an average lifespan of more than ten days in our feeds. Also, we discovered that each type of malicious activity displays different behaviors and average uptime.

Since May 2008, we have observed botnet C&C servers on 1,269 IP addresses. Figure 2 displays the uptime of the botnet C&C servers from 0-60 days. Note that we observed C&C servers that were online for more than 60 days, but limited the x-range of the graph to illustrate the rapid decline in botnet C&C servers that are taken down after only a few days, mainly by reputable IRC and web hosting providers.

We have been monitoring 1,161 of drive-by-download servers since August 2008. These servers have a much higher average lifetime than the other sources depicted in Figure 3. In fact, the number of drive-by-download servers that have been online for more than 60 days is 92, or more than 15%. Also, there have been 17 (approximately 3% of all) drive-by-download servers that have been online since the start of our collection.

From July 2008, we recorded 12,149 IP addresses hosting phishing websites. Similar to botnet C&C servers, the majority of phishing websites were online for only a few days. However, we also observed a few phishing sites that were online for more than a year. Figure 4 shows the uptime for the first 60 days for phishing hosts.

As mentioned previously, we use the longevity of malicious services as a distinguishing feature of rogue networks. This insight is supported by the previously-shown data, which demonstrates that a small number of ASNs is responsible for most persistent, malicious activity. To discard IPs that have been active for a short time only, we introduce a threshold  $\delta$ . IP addresses that are active less than this threshold are not considered rogue and discarded from the subsequent malscore computation. This removes most of the phishing pages that are hosted on free web spaces or hacked machines, and legitimate IRC/web servers that are temporarily abused for botnet communications. As we will explain later in more detail (in Section 5.2), we do not use a threshold-based filter for drive-by-download servers. The reason is that such servers are difficult to set up, and thus, are typically a direct indication for rogue networks. This is also reflected in the uptime graph for drive-by download servers (Figure 3), which is different than the graphs for the other two data sources.

The output of the filtering step (which removes short-lived botnet C&C and phishing IPs) is a list of active, rogue IPs that constitute the input to the malicious score computation process, which is discussed in the next section. In Section 5.2, we will come back to the effects of selecting different values for the threshold  $\delta$  on malscores and ASN ranks.

## 4.2. Malscore Computation

Once per day, the data collection process produces three lists  $\mathcal{L}_i$  of active, rogue IPs (each derived from a different data source  $i$ ). In the next step, the goal is to combine this information to expose organizations that act maliciously. For this, we consider an organization to be equivalent with an autonomous system (AS). An autonomous system is a group of a single entity (RFC 1771). Thus, it is a natural choice to perform analysis at the AS-level.

To identify those autonomous systems that are most likely malicious, we first map all IP addresses on the three lists to their corresponding ASN. For this, we query the `whois` database, selecting the most specific entry for an IP address in case multiple autonomous systems announce a particular IP. We are aware that the `whois` data might not be completely accurate. However, even in case of small errors, the database is sufficiently complete and precise to recognize the worst offenders.

A straightforward approach to identify those autonomous systems that are most malicious is to compute, for each AS, the sum of the IPs on the three lists that belong to this AS. While simple, this technique is not desirable because it ignores the size of a network. Clearly, when an AS  $P$  controls many more live hosts than AS  $Q$ , we can expect that the absolute number of malicious hosts in  $P$  are higher than in  $Q$ , even though the relative numbers might show the opposite. To avoid this pitfall, we compute the maliciousness score (malscore)  $\mathcal{M}_A$  for an AS  $P$  as follows:

$$\mathcal{M}_P = \rho_P * \sum_{i=1}^3 n_i(P) \quad (1)$$

In Equation 1,  $n_i(P)$  is the number of IP addresses on list  $\mathcal{L}_i$  that belong to the autonomous system  $P$ . Moreover, the malscore for each AS is adjusted by a factor  $\rho$ , which is indirectly proportional to the number of hosts in a network. That is,  $\rho$  decreases for larger networks.

The purpose of  $\rho$  is to put into relation the number of incidents with the number of active hosts in an autonomous system. This requires, for each AS, the knowledge of the number of live (active) hosts that

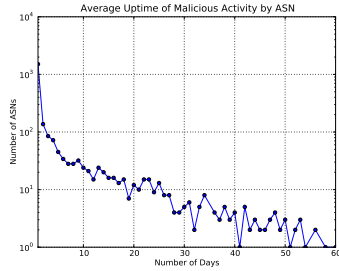


Figure 1: Average IP uptime by ASN.

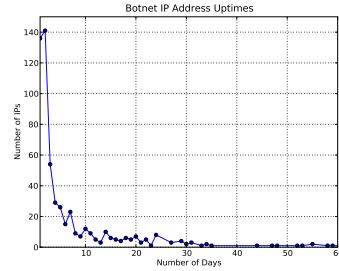


Figure 2: Botnet uptime between 0-60 days.

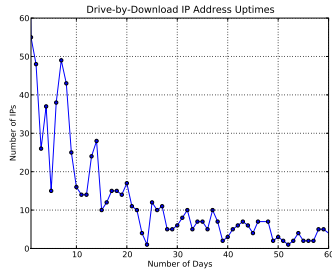


Figure 3: Drive-by uptime between 0-60 days.

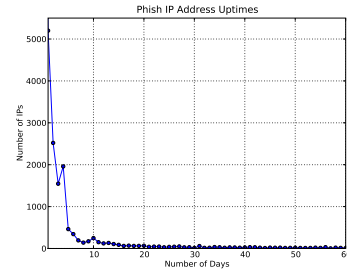


Figure 4: Phishing uptime between 0-60 days.

are operating in the networks of this AS. Clearly, this knowledge is difficult to obtain precisely, and it also can change over the course of several months. Previous work attempted to address this question [20], resorting to the idea of sending ping probes to a well-chosen subset of the IP addresses of a network. While these techniques can discriminate well between completely inactive (dark) regions and used networks, it is still quite difficult to determine the exact number of active hosts. Also, it is possible that networks are configured so that they do not respond to ping requests at all, thereby skewing the results. For these reasons, we decided to estimate the size of a network based on the size of the networks (i.e., the number of IP addresses) that an AS announces as routeable to the global Internet. To determine the size of the address space that an AS announces to the Internet, we leverage data provided by the Cooperative Association for Internet Data Analysis (CAIDA). CAIDA is a collaborative undertaking among organizations in the commercial, government, and research sectors that promotes cooperation in the engineering and maintenance of a robust, scalable, global Internet. In this role, CAIDA makes available a variety of data repositories that provide up-to-date measurements of the Internet infrastructure. One of these data repositories [14] shows a ranking of autonomous systems based on the size of their customer cones (address spaces). This information is compiled from RouteViews BGP tables.

We define  $size_p$  as the number of /20 prefixes that an AS  $P$  announces. With this, we define  $\rho$  as shown in Equation 2 below. As desired,  $\rho$  decreases when  $size_p$  increases.

$$\rho_p = 2^{-size_p/c}, \text{ where } c = 4 \quad (2)$$

Of course, we are aware of the fact that the announced address space is not a perfectly reliable indicator for the number of active hosts. For example, there are network telescopes or educational institutions such as MIT that announce huge address ranges while having few or no live hosts. However, such networks are infrequent and, given the shortage of available IPv4 address space, many networks densely populate their available space. On the other hand, masquerading (network address translation - NAT) might result in multiple hosts sharing a single IP address. Because of the imprecision that is inherent in estimating the number of active hosts, we limit the impact of  $size$  on  $\rho$  by a parameter  $c$ . Empirically, we found that a value of  $c = 4$  yields good results. In Section 5.2, we motivate this choice and discuss the influence of different values of  $c$  on our results.

## 5. Evaluation

In this section, we analyze the quality of our results. Moreover, we discuss in more detail the choice of im-

portant system parameters (such as the time threshold  $\delta$  and size parameter  $c$ ).

## 5.1. Analysis Results and Malicious Networks

Table 1 shows a snapshot of our system on June 1st, 2009, listing the ten entries with the largest malscores and the originating country (using the ip2location.com database). For this snapshot, we computed the maliciousness scores for all 417 autonomous systems that control at least one active, rogue IP.

Unfortunately, we do not have ground truth available that would allow us to evaluate the results of our system in a quantitative fashion. In fact, if such information would be available, then there would be no need for our system. Thus, we can only argue qualitatively that our system produces meaningful and interesting insights into the behavior of rogue networks.

**Correctness of results.** The top ten autonomous systems reported by *FIRE* on June 1st host a large number of persistent, malicious servers. In an attempt to confirm that our results are correct and meaningful, we leveraged a number of third party efforts that attempt to track down certain types of malicious activity on the Internet. More precisely, we first obtained a top-25 list, compiled by the ShadowServer Foundation [22], that shows the most malicious networks with regards to botnet activity. Then, we looked at Google’s Safe Browsing initiative [15] and extracted the top 150 ASNs, based on the absolute numbers of malicious drive-by servers that Google identified. In addition, we used the top-10 entries provided by ZeusTracker [26], a network that monitors and lists command and control servers for the Zeus botnet. Finally, we searched a number of blogs written by well-known security researchers for references to malicious and rogue ISPs and networks.

For each of our top ten entries, we then tried to find evidence in any of the third party lists that would confirm that a network is known to be rogue, or at least, strongly linked to certain malicious activities. Table 1 shows that we were successful for all ten entries.

In our list, IPNAP-ES (GigeNET) has consistently ranked among the top malicious networks, because it hosts the largest numbers of IRC botnet C&C servers. This is confirmed by the findings of ShadowServer. Some security forums have actually reported botnet activity from IPNAP as early as 2006. The Petersburg Internet Network (PIN), currently ranked second in Table 1, is known to be hosting the Zeus malware kit (also known as Zbot and WSNPoem).

It is also interesting to note that the “Novikov Aleksandr Leonidovich” AS has been linked to the recent Beladen drive-by-download exploit campaign [12],

which is believed to be run by the same criminals that operated the Russian Business Network.

**Completeness of results.** In addition to checking our own top entries and comparing them to information from third parties, we also decided to analyze the top entries that these third parties have listed. This might allow us to find malicious networks that our analysis missed. In many cases, we found that malicious networks in those lists were also identified and prominently listed by *FIRE* (although, of course, not always in the top ten). This is especially true for Google’s Safe Browsing list.

For the remaining entries that did not overlap with our results, we found that they mainly fit into two categories. In the first category, we find many large networks that were given an unfair bias in these lists due to the number of compromised hosts on their network. This includes large ISPs such as Cogent. We tagged these large networks with an  $X$  in each table to show that they are likely false positives. The second category consists of reputable networks that provide web and IRC hosting services (e.g., EUnet Finland hosts an IRC server for EFnet or FDCservers) with very short-lived malicious servers. That is, these networks just happen to be listed because they were under attack on a certain day, but they drop out quickly once the hosts or services are cleaned up. Thus, we believe that our results clearly show the importance of filtering ASNs by size and IP address longevity to accurately identify rogue networks while removing false positives.

## 5.2. Sensitivity of Important Parameters

**Longevity thresholds.** To distinguish between rogue and benign networks, *FIRE* uses thresholds  $\delta$  based on the longevity of a malicious server. If a malicious host is online/active longer than this threshold, the IP is considered malicious. If a host is taken offline before it reaches the threshold, *FIRE* discards the corresponding IP for the malscore computation. The choices of the thresholds is thus important for the correctness of the analysis. If a threshold is selected too low, many compromised (but benign) hosts would be considered part of malicious networks. If the threshold is chosen too high, true malicious servers will be missed.

To quantify the influence of different thresholds on the results produced by *FIRE*, we introduce a simple distance metric between two rankings (i.e., lists of malicious networks sorted by malscore). This metric works by computing the edit distance between the two rankings  $A$  and  $B$ ; that is, the distance between  $A$  and  $B$  is the number of insertions and deletions of ASNs that are needed to “convert” the ranking  $A$  into  $B$ .

Rank	ASN	Name	Country	Score	ShadowServer	Google	ZeusTracker	Blogs
1	AS23522	GigeNET	US	42.4	1	-	-	
2	AS44050	Petersburg Internet Network	UK	28.0	-	-	6	[9]
3	AS3595	Global Net Access	US	18.2	-	23	-	
4	AS41665	National Hosting Provider	ES	16.5	-	104	5	
5	AS8206	JUNIKNET	LV	14.1	-	30	-	
6	AS48031	Novikov Aleksandr Leonidovich	UA	14.0	-	-	-	[12]
7	AS16265	LEASEWEB	NL	13.0	24	14	-	
8	AS27715	LocaWeb Ltda	BR	11.6	-	130	-	
9	AS22576	Layered Technologies	US	11.5	-	64	-	[8]
10	AS16276	OVH OVH	FR	10.6	25	18	-	

Table 1: *FIRE* Top 10 for June 1st, 2009

ShadowServer Botnet C&Cs				Google Safe Browsing			
ASN	Name	FIRE Rank	Large Network	ASN	Name	FIRE Rank	Large Network
AS23522	GigeNET	1		AS4134	Chinanet Backbone No.31	17	X
AS3265	XS4ALL	118	X	AS21844	ThePlanet.com	13	
AS25761	Staminus Comm	-		AS4837	China169 Backbone	90	X
AS30058	FDCservers.net	-		AS36351	SoftLayer Technologies	30	
AS174	Cogent	148	X	AS26496	GoDaddy.com	15	X
AS2108	Croatian Research	-		AS41075	ATW Internet Kft.	23	
AS31800	DALnet	-	X	AS4812	Chinanet-SH-AP Telecom	89	X
AS13301	Unitedcolo.de	86		AS10929	Netelligent Hosting	12	
AS790	EUnet Finland	-		AS28753	Netdirect	11	
AS35908	SWIFT Ventures	68		AS8560	1&1 Internet AG	-	X

Table 2: ShadowServer Botnets / Google Safe Browsing Top 10 for June 1st, 2009

We then add to this value the number of those ASNs that appear in both rankings but that have a different number of rogue IPs.

We used our metric to understand the influence of different threshold values on the result. To this end, we first calculated a ranking for a small threshold value. Then, we iteratively increased the threshold by a small value, recalculating the rankings at each step. Finally, we compare the rankings between each pair of subsequent steps. The idea is to see whether rankings eventually “stabilize,” or whether they continuously fluctuate, depending on the specific values for  $\delta$ .

We applied our analysis to all three data sources, ranging the threshold  $\delta$  from 0 to 9. This was done for each day since January 1st, 2009, and the results were averaged. Figure 5 shows the results. Figures 5a and 5b indicate that for phishing servers and botnet control servers, there is significant fluctuation when threshold values are low. This is a direct result of the fact that these data sources contain many compromised servers that are taken offline after only one or two days by vigilant ISPs. Thus, we select the thresholds in a way that such compromised (but benign) servers are ignored. An ideal threshold value should be chosen high enough that the spikes at the beginning of both graphs are cut off, and the fluctuations around the threshold should be low. Thus, a threshold value that lies to the right of the initial peak in the curve is a

good choice. Consequently, *FIRE* uses thresholds of  $\delta_{phish} = 3$  and  $\delta_{bot} = 4$ .

For drive-by-download servers, we did not observe a stabilizing effect over time. On the contrary, Figure 5c shows a constant fluctuation. The reason is that most drive-by-download servers are not taken offline quickly. These servers are typically deployed by professional criminal organizations who do not want to risk that their exploits fail because the mothership server is taken offline. Thus, such servers are predominantly deployed in rogue networks. As a result, we do not take the uptime of drive-by-download servers into account when computing malscores.

**Size parameter.** As mentioned previously, *FIRE* decreases the malscores of large networks. This is to compensate for the fact that, due to their size, bigger networks are more likely to contain a significant number of rogue IPs. The extent to which the score of larger networks is decreased is influenced by the parameter  $c$ .

To show the effect of different choices for the parameter  $c$ , we calculated the rankings for varying values of this parameter. Again, we use the metric presented previously to quantify how changes of  $c$  influence the rankings. These result are shown in Figure 6. It can be seen that for  $c$  values (much) less than 1, the overall rank changes are small. This is due to the fact that, with small values for  $c$ , the resulting lists are dominated by ASN size, regardless

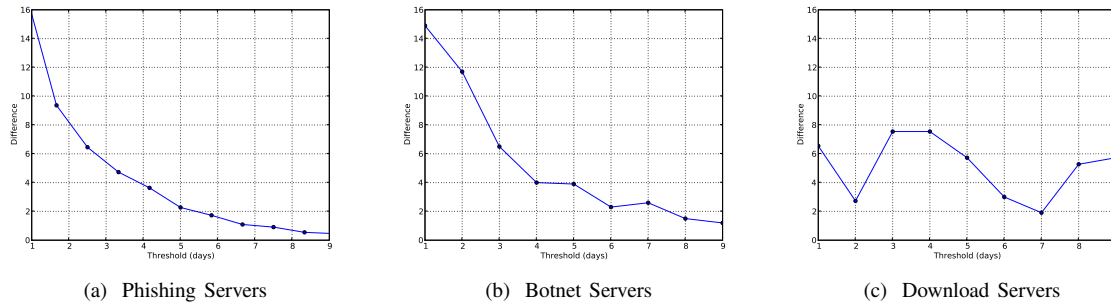


Figure 5: Ranking changes for varying thresholds.

of the number of incidents. Similarly, for  $c$  values much greater than 1, the rankings are dominated by incident count, regardless of the size of a network.

For our analysis, it is thus important to choose a value for  $c$  that is located on the right side of the peak shown in the graph, as we want to favor incident count over network size. However, we are interested in a value for  $c$  that has some effect and, in particular, reduces the rank of very large networks (such as tier-1 ISPs and backbone networks). This lead to the choice of the threshold  $c = 4$  for our malscore computation.

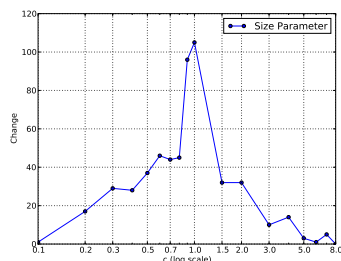


Figure 6: Sensitivity of parameter  $c$ .

## 6. Related Work

The work closest to ours are efforts that attempt to assign a reputation to networks or an individual IP address. In its simplest form, these efforts produce blacklists of IPs that have been observed to perform malicious actions. Most often, such blacklists are used to filter spam mails [23], [24], but there are also blacklists that warn users when they visit potentially harmful web pages [11], [19]. Many of the sites that offer blacklists also compile statistics of the worst offenders, typically by counting the number of incidents in a network. Unfortunately, this technique does not distinguish between compromised, bot-infected machines and hosts in networks that are deliberately malicious. As a result, the worst offenders are typically large

networks with many customers. The goal of our work, on the other hand, is to discard the large amounts of compromised machines and identify those (often smaller) networks likely controlled by determined adversaries.

We are aware of two recent papers [6], [7] that look at temporal and spatial properties of attack sources. In [6], the authors study the spatial-temporal characteristics of malicious sources on the Internet, using data from the DShield.org project. The conclusion is that 20% of all IPs are responsible for 80% of the observed attacks. In [7], the authors attempt to find IPs that are clustered (spatial uncleanliness) and persistent (temporal uncleanliness) in sending spam mails, launching network scans, and hosting phishing pages. This work is closest to ours in that the behavior of hosts is used to identify “unclean” (infected) netblocks. The difference to our approach is twofold: First, we attempt to identify networks that are operated by criminals, while their work was focusing on finding bot infections. As a result, the selection of the input data sets (we include drive-by download providers and botnet C&C servers, but do not consider scanning) and the filtering techniques are different. Moreover, we combine results from multiple feeds. Such correlation efforts were not part of the previous paper.

## 7. Conclusions

In this paper, we presented *FIRE*, a novel system to automatically identify and expose organizations and ISPs that demonstrate persistent, malicious behavior. *FIRE* can help isolate networks that tolerate and aid miscreants in conducting malicious activity on the Internet. It does this by actively monitoring different data sources such as botnet communication channels, drive-by-download servers, and feeds from phishing web sites. Because it is important to distinguish between networks that are knowingly malicious and networks that are victims of compromise, we refine the collected

data and correlate it to deduce the level of maliciousness for the identified networks. Our ultimate aim is to automatically generate results that can be used to pinpoint and track organizations that support Internet miscreants and to help report and prevent criminal activity. Furthermore, the networks we identify can also be used by ISPs as blacklists in order to simply block traffic that is originating from them. Hence, an ISP can enhance the security of its users by not allowing malicious traffic to reach them.

## Acknowledgments

The research was supported by the National Science Foundation under grant CNS-0831408.

## References

- [1] J. Armin, G. Bruen, G. Feezel, P. Ferguson, M. Jonkman, and J. McQuaid. McColo - Cyber Crime USA. <http://hostexploit.com/downloads/Hostexploit%20Cyber%20Crime%20USA%20v%20.0%201108.pdf>, 2008.
- [2] J. Armin, P. Ferguson, G. Bruen, G. Feezel, M. Jonkman, and J. McQuaid. McColo - Cyber Crime USA Supplement. [http://hostexploit.com/downloads/Hostexploit\\_McColo\\_supplement\\_111808.pdf](http://hostexploit.com/downloads/Hostexploit_McColo_supplement_111808.pdf), 2008.
- [3] J. Armin, J. McQuaid, and M. Jonkman. Atrivo - Cyber Crime USA. <http://hostexploit.com/downloads/Atrivowhitepaper082808ac.pdf>, 2008.
- [4] U. Bayer, C. Kruegel, and E. Kirda. TTAalyze: A Tool for Analyzing Malware. In *EICAR Conference*, 2006.
- [5] D. Bizeul. Russian Business Network Study. [http://www.bizeul.org/files/RBN\\_study.pdf](http://www.bizeul.org/files/RBN_study.pdf), 2007.
- [6] Z. Chen, C. Ji, and P. Barford. Spatial Temporal Characteristics of Internet Malicious Sources. In *Infocomm Mini-Conference*, 2008.
- [7] M. Collins, T. Shimeall, S. Faber, J. Janies, R. Weaver, and M. D. Shon. Using Uncleanliness to Predict Future Botnet Addresses. In *ACM Internet Measurement Conference (IMC)*, 2007.
- [8] D. Danchev. The Malicious ISPs You Rarely See in Any Report. <http://ddanchev.blogspot.com/2008/06/malicious-isps-you-rarely-see-in-any.html>, 2008.
- [9] D. Danchev. GazTransitStroy/GazTranzitStroy Rubbing Shoulders with Petersburg Internet Network LLC. <http://ddanchev.blogspot.com/2009/06/gaztransitstroygaztranzitstroy-rubbing.html>, 2009.
- [10] dn1nj4. The Shadowserver Foundation: RBN "Rizing". [http://www.shadowserver.org/wiki/uploads/Information/RBN\\_Rizing.pdf](http://www.shadowserver.org/wiki/uploads/Information/RBN_Rizing.pdf), 2008.
- [11] D. Glosser. DNS-BH - Malware Domain Blocklist. <http://malwaredomains.com/>, 2008.
- [12] D. Goodin. 40,000 sites hit by PC-pwning hack attack. [http://www.theregister.co.uk/2009/06/02/beladen\\_mass\\_website\\_infection/](http://www.theregister.co.uk/2009/06/02/beladen_mass_website_infection/), 2009.
- [13] V. Hanna. Spamhaus: Cybercrime's U.S. Hosts. <http://www.spamhaus.org/news.lasso?article=636>, 2008.
- [14] B. Huffaker. CAIDA: AS ranking. <http://as-rank.caida.org/>, 2008.
- [15] G. Inc. <http://google.com/safebrowsing/diagnostic?site=AS:27715>, 2009.
- [16] B. Krebs. Taking on the Russian Business Network. [http://voices.washingtonpost.com/securityfix/2007/10/taking\\_on\\_the\\_russian\\_business.html](http://voices.washingtonpost.com/securityfix/2007/10/taking_on_the_russian_business.html), 2007.
- [17] B. Krebs. Report Slams U.S. Host as Major Source of Badware. [http://voices.washingtonpost.com/securityfix/2008/08/report\\_slams\\_us\\_host\\_as\\_major.html](http://voices.washingtonpost.com/securityfix/2008/08/report_slams_us_host_as_major.html), 2008.
- [18] B. Krebs. FTC Sues, Shuts Down N. Calif. Web Hosting Firm. [http://voices.washingtonpost.com/securityfix/2009/06/ftc\\_sues\\_shuts\\_down\\_n\\_calif\\_we.html](http://voices.washingtonpost.com/securityfix/2009/06/ftc_sues_shuts_down_n_calif_we.html), 2009.
- [19] PhishTank. Clearinghouse for phishing data on the Internet. <http://www.phishtank.com>, 2008.
- [20] M. Rajab, F. Monroe, and A. Terzis. Fast and Evasive Attacks: Highlighting the Challenges Ahead. In *International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2006.
- [21] C. Seifert. Capture-HPC - Honeypot Client. <https://projects.honeynet.org/capture-hpc>, 2008.
- [22] Shadowserver. ASN Botnet Stats. <http://www.shadowserver.org/wiki/pmwiki.php/Stats/ASN>, 2009.
- [23] SpamCop. Blocking List. <http://www.spamcop.net/bl.shtml>, 2008.
- [24] Spamhaus. Zen: Comprehensive DNSBL. <http://www.spamhaus.org/zen/>, 2008.
- [25] Wepawet. <http://wepawet.iseclab.org/>, 2009.
- [26] ZeuSTracker. <https://zeustracker.abuse.ch/statistic.php>, 2009.