

Why Them?

Extracting Intelligence about Target Selection from Zeus Financial Malware

Samaneh Tajalizadehkhoob, Hadi Asghari, Carlos Gañán and Michel van Eeten¹

Delft University of Technology
Faculty of Technology, Policy and Management
{S.T.Tajalizadehkhoob, H.Asghari, C.H.G.HernandezGanan, M.J.G.vanEeten}@tudelft.nl

Abstract. Malware is used for online payment fraud that causes millions of Euros in damages each year. Not every payment service provider is equally popular among cybercriminals. This paper explores the incentives and strategies of attackers by analyzing the instructions sent to machines infected with Zeus malware between 2009—2013Q1. We investigated around 11,000 configuration files targeting 1.2 million URLs on 2,412 unique domains. We also developed metrics to rank the relative attractiveness of domains as a target. We found that attacks are concentrated: around 15% of the domains attract 90% of the attacks. Concentration is not driven just by target size. Approximating the size of the payment service provider via financial data from the FDIC and traffic rankings from Alexa, we observe that size is a threshold for getting attacked, but does not predict the intensity of attack. Attack persistence also varies widely. Half of the domains are targeted briefly (four weeks or less), which we believe is part of a process of trial-and-error to seek new targets. Surprisingly enough, even though new domains are tried continuously, a ceiling exists in the overall number of domains simultaneously attacked. This suggests bottlenecks elsewhere in the criminal value chain, e.g., in the recruitment of money mules. The ceiling remained in place even after the Zeus source code was leaked, lowering the entry barriers for new attackers. Next, we examined how inject code has evolved over time. Using a cosine similarity metric, we compared the 1.2 million inject codes, and observe that the vast majority of the inject code is repeated many times, with just 1% being never repeated. On average, across all Zeus botnets and attackers, code similarity is well over 90% from one attack to the next. This suggests code sharing, selling, or stealing among attackers. This, again, suggest low entry barriers as well as low development costs. Interestingly enough, these do not translate into growing attack levels. A more general implication of these findings could be that the underground market for malware-as-a-service, often portrayed as making attacks cheaper to execute, is not driving the attack volume or the selection of targets. Future work is needed to flesh out more precisely these mechanisms.

Keywords. *Botnets, cybercrime, cybercriminals financial malware, metrics, online banking fraud, quantitative analysis, target selection, target attractiveness, Zeus*

1 Introduction

Online banking fraud has increased in past decades as web-based online banking platforms have become popular among consumers and businesses. A variety of controls and countermeasures have been put in

¹ The authors thank Michael Sandee and Maurits Lucas from Fox-IT for the technical assistance and providing the Zeus configuration files. For sharing processed data on U.S banks, the authors would like to thank Tyler Moore and Lorrie Faith Cranor.

place by the banking sector and security firms, from better authentication of users to real-time supervision of transactions. Yet, online banking fraud remains a serious problem (Premchaiswadi et al., 2009). According to Anderson et al. (2013), the annual global losses caused by financial fraud are in the magnitudes of billions of euros. The European Central Bank (2014) recently published fraud statistics for the Single European Payment Area, reporting “card-not-present” (CNP) fraud at around €800 million (approximately \$1.1 billion).

Notwithstanding the fact that impacts are substantial across industrialized countries, we also see remarkable differences in fraud levels. For 2012, UK published a total loss of around €299 million for CNP-fraud (FFA UK, 2013). Over the same year, France reported €160 million and the Netherlands reported €35 million of online payment fraud (NVB, 2013; OSCP, 2013). Relative to the number of inhabitants, France and the Netherlands suffer roughly half the level of online fraud of the UK. Sullivan (2010) has estimated relative fraud levels in different countries in 2006. He found that Spain and Australia experienced the lowest rates of fraud, around \$.022 and \$.024 per \$100 of transactions respectively, whereas the UK and US suffered worse fraud levels, losing \$.086 and \$.092 respectively.

One of the foundations for designing effective mitigation strategies would be to better understand what factors drive the differences in fraud levels. An obvious driver is the extent to which payment services are targeted by attackers. Very little empirical work has been done into the underlying reasons of why certain targets are selected more often than others. Financial service providers differ in many respects; e.g., total revenue, market share, number of users, authentication mechanisms, money transferring policies, regulatory framework, and the properties of their home markets. Yet we do not know how these differences affect provider’s relative attractiveness as a target.

An important hurdle for such work is the fact that not much data is available from which target selection patterns could be extracted. In this paper, we present a hitherto untapped source of data on target selection by cybercriminals: the instructions sent to the machines infected with financial malware (“banking Trojans”). Given that many attacks are based on financial malware, these instructions provide an insight into the population of targets that the attackers have selected. We have studied a dataset of instructions, so-called configuration files, which have been distributed within the ecosystem of Zeus botnets.

Analyzing a set of 11,000 configuration files, intercepted over 4 years and containing 1.2 million targeted URLs, specifically we set out to answer the following questions:

- What services have been targeted via Zeus malware (section 4)?
- What metrics could be developed to rank the relative attractiveness of targets (section 5)?
- What factors could explain target selection (section 6)?
- How are new targets identified (section 7)?
- What is the effect of the Zeus source code leakage on target selection (section 7)?
- How does the inject code develop over time (section 8)?

The main contribution of this paper is to increase our understanding of the underground economy around malware-based financial fraud. It also helps lay the foundation for future research into the interactions between the security tradeoffs of financial service providers and those of the attackers.

2 Background

In the course of the 1990s, banks started offering access to the bank’s computer systems via the Internet using a browser or specific application (Claessens et al., 2002). The online channel reduced the need for costly retail branches and paper transactions. This was not only a way to offer new services, but also a strategy that created cost-savings for financial institutions (Claessens et al., 2002; Jaleshgari, 1999). More recently, mobile devices have become another channel for electronic banking activities. Predictably, these innovations also meant that financial services became the target of a variety of online attacks.

2.1 Online banking fraud

Online banking fraud is typically account takeover: removing money from someone else's bank account. It can take place via different attack vectors. Two of the most prominent types of attacks are credential stealing and content manipulation, which can be used separately or in combination. Credential stealing attacks attempt to access users' credentials via phishing or through the use of financial malware (Adham et al., 2013). Content manipulation, also called man-in-the-browser (MitB) attacks, installs malware to manipulate at the system level the ingoing and outgoing communication between the unaware user and the bank (Utakrit, 2009). This type of attack allows attacker to be selective in choosing the target domains and the type of data he intends to steal or manipulate (Adham et al., 2013).

2.2 Zeus malware

Zeus, also known as Zbot, is a readily available malware kit that contains the tools required to build and control a botnet. The kit is very simple to use since it does not require in-depth technical knowledge (Wyke, 2011). Zeus was first exposed on July 2007 and worked on computers using Microsoft Windows operating systems. Since 2012, there are also Zeus variants for Blackberry and Android phones. Zeus malware has been primarily known for its use in financial fraud, but its features also allow other types of data theft (e.g., password sniffing). The Zeus source code was leaked in 2011 and since then it has been sold and traded widely in underground forums. Numerous variants of the original Zeus malware have appeared (Kruse, 2011).

Zeus malware operates based on instructions that are specified in a so-called "configuration file". The configuration file has two parts: the static and the dynamic section. Information located in the static section will be hardcoded into the bot executable and contains the information that the bot will need when it is first executed, such as the URL where to get the dynamic section (Falliere & Chien, 2009; Macdonald, 2011).

The dynamic configuration file (config.bin) is downloaded by the bot immediately after it is installed on a victim's computer. The static configuration contains an RC4 key, which is used to encrypt the communication in the botnet, including the dynamic configuration file. In this implementation, a key stream is generated from the botnet password, and is XORed with the data (Macdonald, 2011). Note that the keys effectively segment the total population of Zeus clients into different botnets by connecting each one to a specific command-and-control (C&C) channel. It can therefore serve as a proxy for a distinct attack campaign and, thereby, of an attacker – though the latter is a lot less reliable, as an attacker can behind multiple campaigns simultaneously as well as over time.

The file is updated by the C&C server and is downloaded by the bot at certain time intervals, providing it with new instructions. Most of the entries in the file control how and what information is collected from the infected computer and where to inject html codes in the webpages visited by the infected machine (Falliere & Chien, 2009). Note that for the sake of simplicity, these dynamic configuration files are called configuration files in the remainder of this paper.

As soon as the victim's computer gets infected, the Zeus malware attaches itself to the user's web browser. This enables it to monitor everything the victim does on the web, including his/her online banking and credit card transactions (Macdonald, 2011). Zeus records everything the victim types in the browser, including usernames, passwords and banking credentials, and sends them back to the command and control server where information is stored in a dropzone. The criminal can then use this information directly to steal money from the victim's accounts or he can sell the information to other criminal organizations that have the infrastructure for large-scale online banking and credit-card fraud operations.

Zeus acts as a Man-in-the-Browser (MitB) and modifies what victim sees on her bank's web page. Fraudulent transactions are executed by modifying certain web pages and injecting data into certain fields, invisible to the user. These so-called 'inject codes' are located in a section called 'Web Injects' within the configuration file. To illustrate: sometimes criminals inject extra fields to the bank's login webpage that asks for additional login information, such as credit card details or PIN numbers, which are normally not required for the login process. In other types of attacks, the webpage is modified to show a

fake account balance to the user, thereby hiding a fraudulent transaction that has been executed in the background.

Zeus is the most widespread banking Trojan. This also implies that a better understanding of its operations can yield useful insights into target selection and other criminal behavior around online financial services. To this end, this work is dedicated to exploring Zeus configuration files.

2.3 State-of-the-art: Target Selection

Earlier work has often focused on the technical vulnerabilities of banking services and on developing more secure online banking technologies (Li et al., 2012). However, several authors have pointed out how the incentives of financial service providers shape the security decisions associated with these vulnerabilities (Cranor et al., 2013; Murdoch & Anderson, 2010; Murdoch et al., 2010).

Some earlier work has found that attackers tend to favor certain financial services over others. Moore and Clayton (2007) studied a sample of phishing sites and found that some banks are targeted much more frequently than others. PayPal was impersonated by 399 of the 1695 sites, while 52 banks were only spoofed once. They do not explain this discrepancy, except indirectly: banks can influence how long phishing sites stay online. Perhaps that serves as a deterrent. It is unlikely, however, that PayPal is much less vigilant than 52 rarely attacked banks.

A study on click fraud by Christin et al. (2010) concluded that online-only banks were targeted more than banks with physical branches. Levchenko et al. (2011) in their analysis of on spam value chain found that, there is a significant concentration on certain merchant banks that assist sellers of online pharmaceuticals.

As far as we know, there hasn't been an in-depth empirically investigation into the extent to which some banks are targeted more often than others. While there are many factors at play, here, it seems clear that the criminal's decisions play a major role. Recent security reports claimed that online banking attacks were getting more target-specific (Sherstobitoff, 2013; TrendMicro, 2012). This suggests a conscious selection process on the side of the criminals, but we have not yet uncovered the factors that drive this process. Is the selection process for online banking fraud based on specific characteristics of the bank, its policies, its location, or another set of considerations altogether? Perhaps the decisions of attackers are less guided by informed cost-benefit tradeoffs and more by herding behavior: in the absence of good information about the likelihood and magnitude of success, they mimic whatever other attackers are doing (Bikhchandani et al., 1992), driven by underground forums or chatrooms where experiences are exchanged. Yet another strategy might be to do the opposite: select targets that nobody else is attacking?

From the study of the economics of crime, we can draw upon Routine Activity Theory (RAT) to describe why criminals go after a certain target. RAT argues that for a crime to be committed, three ingredients are needed: a motivated offender, a suitable target, and the absence of a capable guardian at a specific time and place. RAT has been developed in the context of conventional "offline" crime. However, Yar (2005) has argued that the differences between the virtual and non-virtual worlds made the applicability of RAT to cyber-crime limited. These differences include cyberspace's different socio-interactional characteristics such as the collapse of spatial-temporal barriers, many-to-many connectivity, and the anonymity and plasticity of online identity.

Yar (2005) has adapted RAT to cybercrime. He identifies the four key properties that derive the so-called "suitable target" to be selected as: *value*, *portability*, *visibility*, and *accessibility*. Value of target in online banking fraud can be defined as the value that can be gained by the offender if the attack is successful. This might mean that banks in richer countries or with higher account balances would be selected more often, all other things being equal. Portability is about the ease with which the criminal gains can be moved, such as money being transferred in near real time via irreversible transactions. Visibility is about how visible the target is to the cybercriminals. Finally, accessibility is about how easy the target can be reached.

In sum, we are not aware of any prior empirical work addressing the research question of what factors drive criminals to target certain online financial service providers more often than others in online banking fraud.

3 Method

3.1 The dataset

In this paper, we analyzed a collection of Zeus financial malware data files provided to us by Fox-IT, a leading Dutch security firm with many clients in the financial sector. The dataset consists of around hundred and fifty thousand (144,625) captured files that were suspected to be configuration files. Of these, we investigated 10,673 configuration files that revealed targeted domains. The rest of the captured configuration files could not be used because of one of the following reasons: (1) the file could not be decrypted with one of the keys extracted from the executables or because it was in an unknown format; (2) the file was decrypted but not actually a configuration file; (3) the file was corrupted or incomplete (some files were only captured in full after multiple attempts); (4) the file was decrypted but did not contain a web inject section and therefore no targeted domains.²

The configuration files were collected over a period of just over four years (2009-2013Q1). They were captured using honeypots located all over the world (though more concentrated in Western countries and less in Asia). The configuration files were collected using two different methods: they are gathered by running live Zeus samples or by emulating the malware to download configuration files.

The configuration files are encrypted plain text files. Each captured file, along with the time stamp of when it was captured and with the key with which it was decrypted, if applicable, was stored in a MySQL database. As already explained in section 2.2, each Zeus configuration file contains a 'web inject' section which includes targeted URLs, attack instructions, HTML scripts to be injected into the pages served from the attacked URL, and mechanisms for bypassing the authentication procedures of the institution. Below is an example of the web inject section of Zeus configuration files. These configuration files typically contain multiple injects (113 of them, on average), each of which varies from just a few lines of code, as in the example below, to over two thousand lines:

```
set_url      https://REMOVED.com/OLB/secure/AccountList.aspx <FLAG_GET><FLAG_LOG>
data_before
id="dgDepositAcctsheader0"*>
data_after
</table>
data_inject
datas_end
```

3.2 Extracting targeted domains and botnet keys

To answer our research questions, we need to parse the configuration files. First, we extract targeted URLs from the configuration files and associated it with the time stamp and the RC4 key belonging to that file.

The configuration files revealed that a total of 14,870 unique URLs were targeted. Many of them contain different paths of the same domain. All in all, we identified 2,412 unique domain names. Extracting the

² Sometimes the malware only monitors the machines http and https post requests and gathers system information without modifying anything. That is why some of the configuration files do not contain the web inject section and could not reveal targeted URLs.

domains from targeted URLs was not a straightforward process, as some configuration files contain URLs with wild cards. Table 1 displays some examples of such URLs.

Table 1-Example of target URLs in Zeus configuration files

Targeted URL	Extracted Domain
/bancopostaonline.poste.it//formslogin.aspx	poste.it
*banking.*sparkasse*.de/cgi/anfang.cgi*	sparkasse.de
*mpresas*gruposantander.es*opaccesoempresasabe*	gruposantander.es
http://www.google.*&q=*	google.?

The algorithm with which we extracted the targeted domains from the URLs is displayed in Figure 1. In the cases of URLs without wildcards, we extracted targeted domains by trimming the path. In the cases of URLs with wild cards, we reconstruct the targeted domain using regular expressions for comparing the last part of the URL (path, query or fragment) against the set of targeted URLs without wildcards. If a URL matches the same URL without wildcards in more than 90% of the cases, we assume that the targeted domain is the same as the domain of the URL without a wildcard. For around 6% of all URLs we could not reliably determine the targeted domain. Either the URL did not match any of the URLs in our set or it matched with multiple URLs and none of them reached the threshold of 90%.

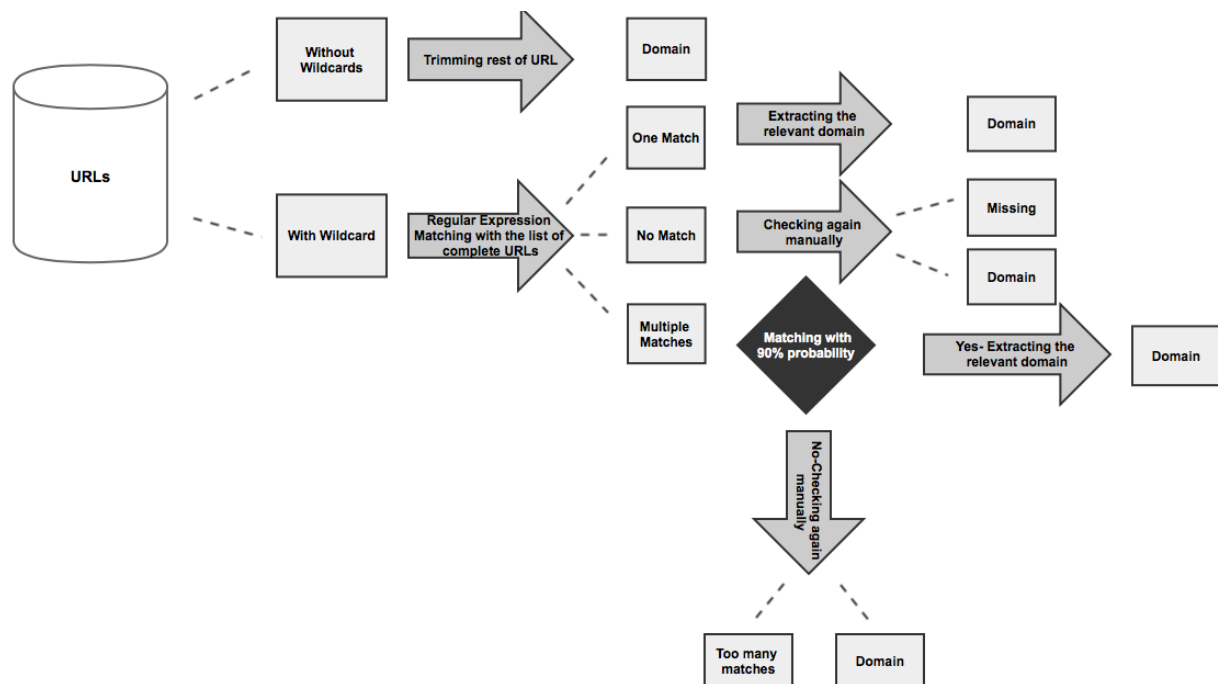


Figure 1- The algorithm used for extracting targeted domains from targeted URLs

Ideally, we would like to count the number of attackers going after each domain. There currently is no reliable way to measure this. We use the RC4 keys as proxies for the number of botnets in use, i.e., the number of campaigns that have been undertaken – in effect treating the keys as unique identifiers for different Zeus botnets. These keys can only be changed by updating the malware with a new binary and migrating the command-and-control server to a new URL. Typically, this happens when a botnet is taken down. The operator then continues with a new bots and a new location for downloading the configuration, which technically makes it a new botnet, as the old bots are no longer able to connect.

3.3 Country of targeted domains

We have enriched the dataset by adding some information about the targeted domains from other sources. The geographical location of each attacked domain was determined manually. For this purpose, we have used four sources of data:

- i. Where the server/infrastructure is located (MaxMind, 2014);
- ii. Where the traffic of the domain comes from, according to Alexa (Alexa Internet, 2014);
- iii. Where the site owner's headquarter is located, according to the domain's homepage;
- iv. The top-level suffix of the domain (TLD).

Most of the time, these sources consistently pointed to the same county. We manually checked cases where they didn't match.³

3.4 Size of targeted domains

We estimated the size of a target domain in two ways: by traffic volume (Alexa ranking) and, for U.S. financial institutions, also by the total of deposits held by the institution, as reported by the U.S Federal Deposit Insurance Corporation (FDIC). The FDIC is a government corporation operating as an independent agency that provides deposit insurance guaranteeing the safety of a depositor's accounts in member banks. As of February 2014, they insure 6,790 institutions (FDIC, 2014). They provide certain statistics for these institutions, such as its total assets, deposits, the locations of its headquarters, its web address, etc. Through the web address field, we can connect 170 of these institutions to the data in our Zeus dataset.

Data on the volume of web traffic of targeted domains is gathered from Alexa Internet, a subsidiary of Amazon.com that provides commercial web traffic data (Alexa Internet, 2014). Using data gathered via the Alexa toolbar and that provided by sites owners, Alexa ranks sites based on their traffic data. They also use data from the DMOZ open directory project to categorize sites ("Odp- Open Directory Project," 2014). We have pulled in this data from the Alexa website for the majority of the domains in our dataset. In later sections of the paper, we use the FDIC deposit and Alexa ranks as proxies for domain size.

3.5 Limitations

We already discussed the limitation of using RC4 keys as a proxy for botnets and attackers. Here, we focus on the possibility of biases in the collection method. Perhaps the honeypot network is less likely to capture attacks against certain domains or countries, or the captured and decrypted files are biased towards the less competent attackers.

To check how representative our sample of configuration files is, we crosschecked our data with Zeus Tracker data from Abuse.ch, a well-known anti-Zeus initiative. We compared the C&C URLs from which the configuration files were between the two datasets for the specific period that Zeus Tracker published this data (Sept 2010–March 2013). First of all, there is a large difference in volume. Where Zeus Tracker published 740 configuration URLs, our dataset identified around 7,500 URLs for the 11,000 files (and more than 23,000 URLs for the overall set of around 144,000 captured files). This large difference might partially be attributable to the fact that Zeus Tracker does not seem to log all the different URL paths for the same domains. That being said, it does seem that the Fox-IT collection method is more comprehensive. Of the 740 URLs listed by Zeus Tracker, 450 are also in our dataset with the exact same path. Looking at the domains only, the overlap is even larger. So our dataset seems to cover the bulk of the Zeus Tracker data, plus a lot more.

³ One interesting pattern that emerged and could be explored in future work is that quite a number of the banks were located in microstates and known tax havens. These 'offshore' banks are most probably used by a small fraction of the population, and it is interesting to see that Zeus has been used to target this small group.

As mentioned in section 3.1, the data was gathered via honeypots, predominantly located in European countries. This might have introduced a geographical bias. Indeed, in the next section we see that the European countries are more often targeted than those in the U.S. or Asian targets. This does not per se imply, however, that there is a bias in the data. Our industry partner argues that the attackers did not differentiate among the infected clients based on their geography directly. The dominance of Europe targets may reflect the fact that online banking services have been offered more widely by European providers and have been adopted more comprehensively by consumers. Also, many European banking systems have near real-time transaction processing, which makes it easier for criminals to move money out of the account before anti-fraud operations can stop it.

The situation in the U.S. is rather different. The market is less consolidated, with thousands of smaller banks, not all of them offering online banking facilities. Furthermore, of the banks that offer online services, not all of them provide cross-bank transferring capabilities, making it harder for attackers to cash out funds. Sometimes there is a need for in-person validation of the receiving account beforehand.

We did notice that Asian countries seemed underrepresented. That might indicate a sampling bias or it might reflect a different attack history. For example, the Zeus-variant Citadel has been reported to have started targeting Japanese banks only late in 2013, which is outside our observation period (Baylor, 2014). Another explanation is that Asian attacks were predominantly executed via other malware families, such as KRBanker. We should also mention that only a small percentage of configuration files in our data belong to the newer Zeus variants, which is understandable since many of the new variants have become active in the course of 2012.

4 Targeted domains in the dataset

As described in section 3.2, from a total of 14,870 unique targeted URLs, we have identified 2,412 unique domains. Not all of these are financial service providers. Among the targeted domains we could find anti-virus companies, news sites, webmail providers, and social networks, as well as domains we could not categorize, because the sites were no longer live. We do not know the exact proportions of financial service providers versus others. Using Alexa, we were able to map 43% of all domains to a specific sector. Of these mapped domains, 760 (74%) were financial service providers and 272 (26%) domains belonged to other sectors. The remaining 1,380 domains are uncategorized (see Figure 2). Among uncategorized domains, we selected a random sample of 100 and manually checked their associated sector(s). Of these domains, we were able to map 72% of them. Of these mapped domains, 53 (73%) were financial service providers and 19 (26%) were from other categories such as antivirus companies, security service providers, and online consultancy firms.

Over the whole period, on average 600 unique domains were attacked each month across all observed botnets (Figure 3). In section 6, we will explore the time trends and their relation with the number of attacked domains in more detail. In terms of geographical coverage of the data, the targeted domains cover 92 different countries all over the world (

Table 2). Unsurprisingly, some countries suffer substantially more attacks than others. In section 7 we will discuss these geographical distributions in more detail.

Categories of domains based on Alexa

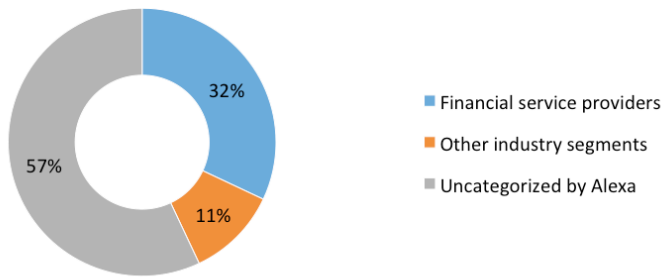


Figure 2- Categories of domains based on Alexa (Alexa Internet 2014)

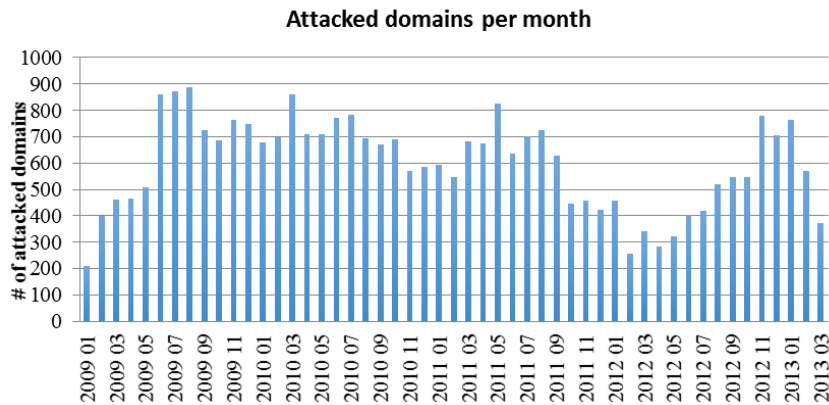


Figure 3- Number of attacked domains each month

Table 2- Geographical distribution of the attacked domains (number of domains per country in the dataset)

#	Country	#	Country	#	Country	#	Country
818	United States	14	Chile	5	Luxembourg	2	Costa Rica
187	Germany	14	Netherlands	5	New Zealand	2	Isle of Man
166	United Kingdom	12	Cyprus	5	Ukraine	2	Lebanon
140	Spain	12	Finland	4	Belarus	2	Malaysia
119	Australia	10	Romania	4	Bulgaria	1	Andorra
117	Italy	9	Czech Republic	4	Denmark	1	Antigua and Barbuda
74	Russia	9	Ireland	4	Estonia	1	Barbados
61	Canada	9	Venezuela	4	Gibraltar	1	Cocos (Keeling) Islands
42	France	8	Brazil	4	Nigeria	1	Egypt
36	Poland	8	Panama	4	Pakistan	1	Iceland
34	Switzerland	8	Singapore	4	Peru	1	Indonesia
29	India	7	China	4	Virgin Islands	1	Jamaica
24	Portugal	7	Hong Kong	3	Curaçao	1	Jordan
23	Latvia	7	Japan	3	Kuwait	1	Kazakhstan
22	Austria	7	Lithuania	3	Malta	1	Kenya
18	Colombia	7	Mexico	3	Puerto Rico	1	Liechtenstein
18	Norway	6	Dominican Republic	3	South Korea	1	Niger
18	Thailand	6	Greece	3	Tanzania	1	Philippines
17	Belgium	6	Hungary	2	Bahrain	1	Saint Vincent and the Grenadines
17	United Arab Emirates	6	Iran	2	Belize	1	San Marino
16	Saudi Arabia	6	Israel	2	Bermuda	1	Slovakia
16	Turkey	6	South Africa	2	Bolivia	1	Taiwan
14	Argentina	6	Sweden	2	Cayman Islands	1	Uruguay

There are 2,131 unique RC4 keys in our dataset, which we interpret as an approximation of the number of botnets in use. Figure 4 displays the trend of botnet activity from January 2009 up to March 2013. The blue (upper) line displays the number of configuration files sent each week by all botnets together. The red (lower) line indicates the number of botnets that were active in that week, as counted by the number of RC4 keys in use. The number of active botnets decreases over time. The same happened with the number of configuration files that were distributed. This might be attributed to the Zeus takedown efforts that were coordinated by Microsoft with different governments and security firms around the world (Domingues Boscovich, 2012), although the downward trend had started well before those efforts.

Comparing number of configuration files against number of active botnets per month (the two lines in Figure 4), we see that they roughly follow the same trend. This is to be expected, as the number of active botnets is determined by whether or not they have distributed a configuration file that week. However, it also can be seen that, on average, botnets sent out multiple files. This is unevenly distributed. Some are much active than others. This discrepancy highlights that raw counts of the number of times a domain shows up in configuration files is not really a good metric for the relative degree in which a domain is targeted. We explore more informative metrics in the next section.

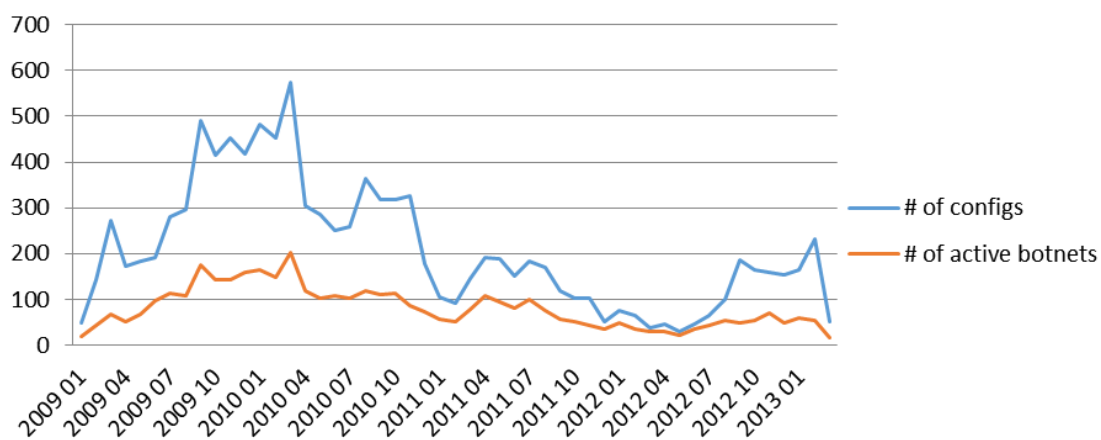


Figure 4- Number of configuration files and number of active botnets

5 Developing Metrics

In order to be able to understand how popular different domains are as targets of online banking fraud, we need to rank them against a meaningful metric. Until now, there is little empirical work on the popularity of targeted domains. Even where it exists, the ranking is based on poorly conceived metrics. To illustrate: a security paper published by F-Secure (F-Secure, 2012) reported a list of top 20 most attacked domains by the SpyEye malware in 2012. The rankings were calculated by simply counting the number of times a domain appeared in the malware configuration files. We believe that such raw counts are not reliable, mainly because the number of times a configuration file is sent is not necessary equal to the number of attacks. Taking the example of SpyEye malware, the configuration file is built into the binary, so attack instructions are released as often as the binary is changed. These changes are likely to be driven by signature updates in the antivirus software that SpyEye tries to evade, rather than by the target selection process of cybercriminals.

There are numerous ways in which bot herders may choose to update configuration files; one may update a configuration file once per days, while another one might adopt a lower update frequency, perhaps because she herds multiple botnets or because the botnet has more stable attack code. Therefore, the number of configuration files per day sent by a botnet may have little correlation with the actual attacks and, thus, with target popularity. To illustrate, in

Figure 5 we display the configuration files of three different Zeus botnets in the same week. Using raw counts we would say Botnet 1 attacked *ebay.com* three times this week and Botnet 3 two times, so in total

ebay.com is attacked five times in this week. However, in practice, all of the configuration files sent by botnets in one week are only the updated versions of the initial ones. It makes no sense to count them as separate attacks.

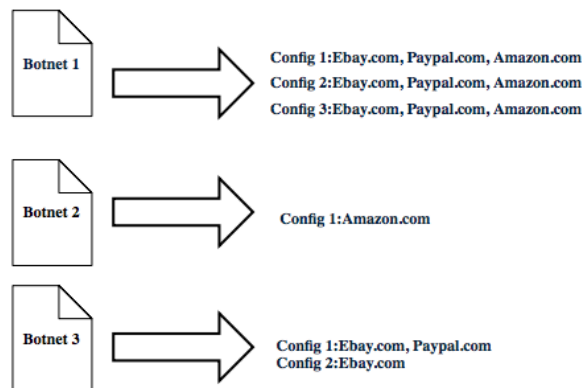


Figure 5- Example for explaining why 'raw counts' of attacked domains are not reliable as metric

In short, in order to be able to rank popularity of targets and to deal better with these differences, we need to develop more reliable metrics. Three alternatives could be:

- **Number of botnets attacking a domain.** Using this metric, a domain's attractiveness is defined by the number of Zeus botnets, counted by different RC4 keys that simultaneously targeted that domain.
- **Number of weeks a domain was under attack.** The Zeus data can also provide information about the persistency of an attack over time.
- **Average number of botnets attacking a domain per week.** This metric is basically a combination of the previous two. The metric eliminates some of the limitations of raw counts by normalizing the data: it merges all configurations for a single botnet sent during a week and then counts the number of botnets attacking that domain in that week. To compare over longer periods, one could add up the count for each week ('botnet weeks') or average them. The formula below displays how the metric is calculated when being averaged across n weeks.

$$\text{Average number of botnets attacking a domain per week} = \left(\sum_{k=1}^n \text{botnets including domain in their configuration file in week } (k) \right) / n$$

6 Relative Attractiveness of Targets

Now that we have defined more reliable metrics, we can inquire into the relative popularity of different targets and the attackers' incentives behind this pattern. We do this by discussing three questions: (1) how are the attacks distributed across targets? (2) what is the relation between target size and popularity? Or to put it differently: do bigger targets attract more attacks? And (3) what is the pattern of persistence of attacks against different targets?

6.1 Distribution of attacks

In Figure 6 we rank the popularity of domains as attack targets using the metric *number of botnets attacking a domain per week*. The rank shows a highly concentrated pattern. The pattern of attacks is in a power law distribution, where 15% of the domains account for 90% of the attacks.

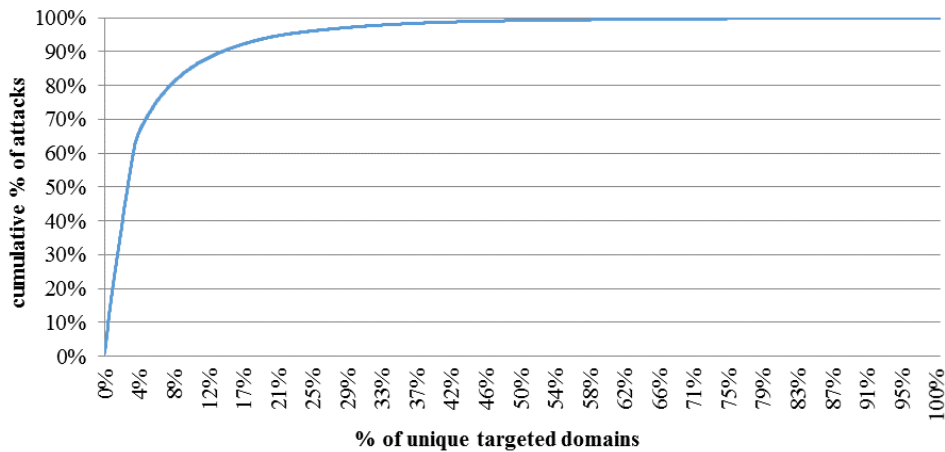


Figure 6 - Cumulative percentage of domains against attacks (in botnet-weeks)(2009-2013Q1)

Target popularity can also be expressed by metric *the number of weeks a domain is under attack*. Here, too, we see a highly skewed distribution: some domains are attacked only briefly, while others remain under attack for the whole period of our dataset. Figure 7 shows the domains distributed over the number of weeks they were under attack between January 2009 and March 2013. A small number of domains (88) were always under attack for the whole 216 weeks. A much larger group of domains (1,108) came under attack for four weeks or less. Finally, 1,216 domains fall between the two extremes: occasionally and often attacked domains.

One interpretation of Figure 7 could be that the range of potential attack targets for criminals is wide: the fact that some of the attacks are short-lived might indicate trial-and-error on the part of attackers, i.e., the attacks aren't successful or don't prove attractive. We will revisit this idea in section 7. An alternative explanation might be that some of these domains are attacked for a specific purpose, i.e., as a part of targeted attacks.

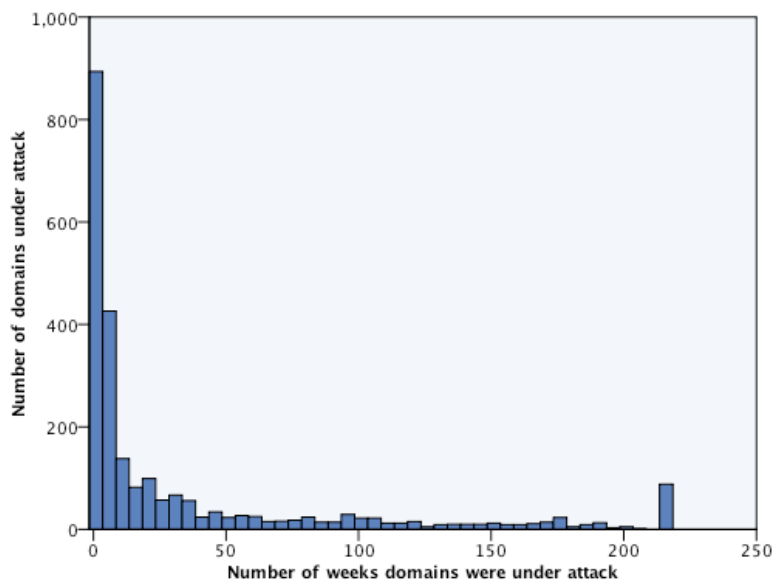


Figure 7 - Attack persistency on domains attacked by Zeus malware (2009-2013Q1)

We categorized domains in Figure 7 into the different groups (see Table 3). What stands out is the fact that the portion of targets that are financial service providers increases with attack persistency, while the number of countries decreases. Here, the attackers reveal their core business: domains is always attacked

category are located in a compact set of countries, most notably Spain, U.S., U.K., Italy, Russia, and Germany.

Table 3 – Domains grouped by attack persistence

Group	Definition	Domains	% Financial Institutions ⁴	Countries
Briefly attacked	Attacked for 4 weeks or less	1,108	54%	81
Occasionally attacked	Attacked >4 weeks and <6 months	571	80%	58
Often attacked	Attacked >6 months and <4 years	645	94%	46
Always attacked	Attacked all four years	88	91%	13

6.2 Size and attractiveness of U.S. banks

The concentrated patterns in the attacks raise the obvious question of incentives: why do so many attackers go after the same cluster of targets? Routine Activity Theory (see section 2.3) identifies four factors that drive target selection: *value*, *portability*, *visibility*, and *accessibility*. At this moment we cannot systematically assess portability or accessibility, though the configuration files do contain inject code aimed to bypass two-factor authentication mechanism of domains, which might tell us more in the near future. Value and visibility can also correlate with the size of the attacked financial service providers, as measured in its customer base and the wealth of those customers. In other words, are the largest providers in the richest markets attacked more? The logistics of malware-based attacks seems to favor banks with a large customer base, as it increases both the odds of finding infected customers as well as spreading out the costs of developing the inject code over more attacks – similarly to why most malware writers target Windows-based machines.

We have crudely estimated the size of a target via its web traffic volume (Alexa ranking) and, for the U.S. financial institutions, via FDIC data on the volume of deposits (see section 3.4 for more details). The FDIC lists U.S. banks and financial institutions with some of their financial and administrative properties, including assets, deposits, and net-income. In Figure 8 we have plotted the top fifty U.S. banks – in terms of deposits – against the average number of botnets attacking them per week over the whole period (2009-2013Q1). The red (top) line in the graph describes the amount of deposits⁵ of each of the institutions on a log-scale; the blue (bottom) line indicates the average number of botnets attacking all domains⁶ of that institution per week. Clearly, the two variables do not maintain a strong relationship.

Mapping the attacked domains to the FDIC list yields to two interesting points. The first relates to the fact that out of around 6,500 active financial institutions, only 175 have been targets of Zeus attacks in our data. Assuming that our sample is representative (see section 3.5), this is a surprising low number. Almost all of the largest banks (48 of the top 50) are attacked and present in the dataset.⁷ The situation for the smaller banks is completely different. This might be cause by the fact that many smaller banks in the U.S. either did not provide online banking services or have outsourced these to a smaller number of third parties who may or may not be among the attacked domains. Another explanation is that they

⁴ This percentage is calculated using the Alexa categories for the sites that are categorized by Alexa - the numbers do not include the uncategorized domains in each group.

⁵ We prefer deposits over assets as a measure of size, to distinguish between financial institutions that might provide mainly mortgages and have little banking and saving services; these will have high assets but low deposits, and so far have been of less interest to botnet-based forms of attacks.

⁶ Some of the institutions have multiple web addresses and domains. We aggregate the attacks on all of these related domains in this figure.

⁷ The two missing banks offer online banking via a third party. These sites handle online banking services for multiple banks. They are both present in our dataset, but attacks on them cannot be attributed to the individual banks.

simply are not attractive targets for Zeus-based attacks, given their small customer size or limitations on how and where funds can be transferred.

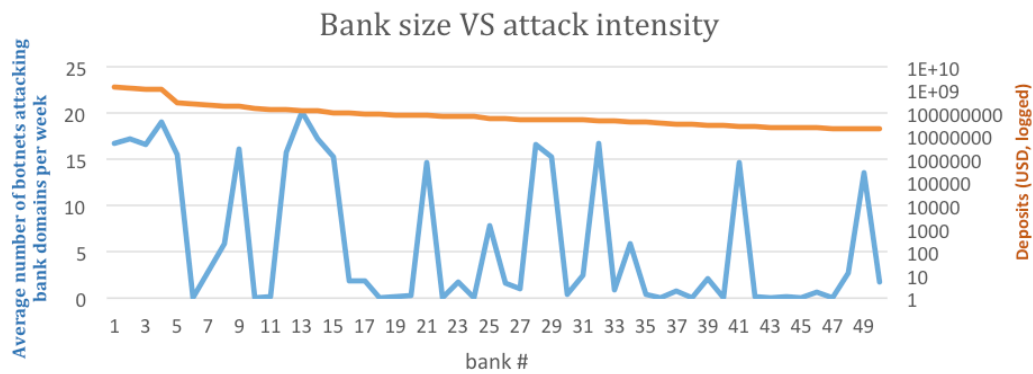


Figure 8 – Deposits (top) and Zeus attack intensity (bottom) for the top fifty U.S. financial institutions

So whether a bank gets attacked is related to its size; above a certain threshold, a bank becomes a target. Beyond the threshold, however, size no longer seems to be a factor. The intensity of attacks is hardly related to size: the average number of botnets attacking each week fluctuates from less than 1 to 20. The result of the regression analysis is an adjusted R^2 of 0.25 ($N=50, F=0.00$) – a weak correlation. This clearly suggests that there are other factors driving target attractiveness than merely the size of bank or its customer base.

6.3 Size and attractiveness worldwide

Does the pattern we observed for the U.S. banks hold across the whole population of targets? Figure 9 shows the relation between Alexa rank of a domain and the persistency of the attacks. On the right, we have included box plots of the Alexa ranks for the different persistence groups discussed in section 6.1. The Alexa rank is a proxy for the size of the domains, with lower rank numbers indicating more incoming traffic, i.e., more users.

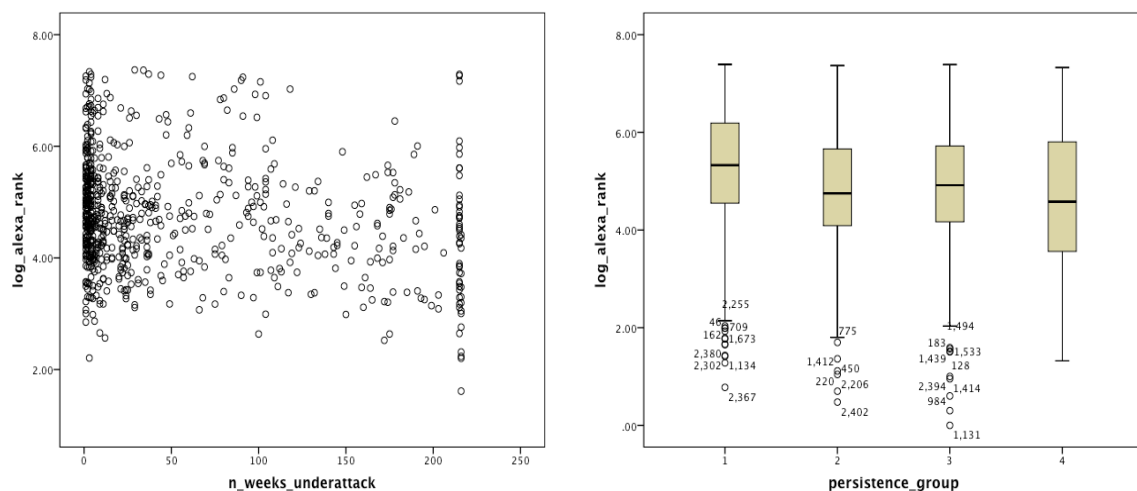


Figure 9 – (Left) Scatter plot of domain rankings vs. number of weeks they are targeted; (Right) box-plot of domain ranks of the different groups. (Only financial institutions, log-transformed. Lower rank number indicates a site has more visitors.)

Running Spearman’s rank correlation test, we see a weak but significant negative correlation between lower ranks (domains with more visitors) and number of weeks they are under attack ($\rho=-0.13, \text{sig}=0.00, N=1,995$). The results are similar if we rerun the correlation only for financial institutions ($\rho=-0.19, \text{sig}=0.00, N=731$). The correlation is driven mainly by the difference between the ranks of the

briefly attacked domains and the always attacked domains. The result of Kruskal-Wallis test for comparing sample means is presented in Table 4 supports this. The same test does not find a significant difference in the traffic ranks of domains that are briefly attacked and occasionally attacked, nor between those that are occasionally attacked and often attacked.

Table 4 - Kruskal-Wallis test results comparing means of Alexa ranks among different target groups

Ranks				Test Statistics ^{a,b}	
	persistence_group	N	Mean Rank		alex_rank
alex_rank	1	866	1112.76	Chi-Square	63.626
	2	479	889.73	df	3
	3	571	936.38	Asymp. Sig.	.000
	4	79	841.88	a. Kruskal Wallis Test	
	Total	1995		b. Grouping Var: persistence_group	

In short: the size of a financial service provider seems to influence target selection mostly in terms of a threshold: providers above a certain size are much more likely to be targeted. Beyond that threshold, however, size does not really seem to impact attack intensity. Within the top 50 of U.S. financial institutions, we see large differences in attack volume. The same holds for the wider group of larger targets. So far, it is unclear what other factors are at work here.

7 Seeking New Targets

A sizable number of domains (1,108, about half of the total) were targeted for four weeks or less in the four-year period (See Table 3). Taking a deeper look into the domains in that group, we realize that this group stands out from the others by its diversity: from the largest to the smallest domains, spread out over 81 countries and multiple industries, only half of which are financial services. This diversity makes sense if we interpret it as the result of a process of trial-and-error by the attackers. A new target is chosen for attack. The attacker identifies the relevant URLs, develops the inject code, and pushes the new configuration to bots under his command, and waits for victims. If the attack is successful, the attacker will persist. If, however, within a few weeks and a handful of attempts, the attack is not successful, the attacker has to decide how long to keep incurring costs before moving on to a different target. The lack of success might be caused by effective defense measures by the targeted institution or its users. It might also be the case that the attack was technically successful, but the value of the loot – e.g., the price that the underground market was willing to pay for the captured data – didn't merit further attacks.

The rate of trial-and-error is reflected in the number of new domains that show up over time. New domains are being tried all the time however, with peaks now and then: on average 119 domains are new each month – either never attacked before, or briefly in the months before the last. The overall number of domains getting attacked per month seems to be remarkably constant with a clear ceiling. An average of 601 domains each month become targets of Zeus attacks ($\sigma=172$, $CV= 0.29$). This is across all botnets in the dataset.

This stable ceiling on the number of targets pursued simultaneously is surprising, given the ongoing development of malware-as-a-service, which supposedly reduces entry barriers and would attract new attackers. It suggests there are bottlenecks elsewhere in the criminal value chain. With money mules, for example, there appear to be no economies of scale (Krebs, 2012). In fact, the security architect of a multinational bank recently told us that they work on the assumption that recruiting mules is the most expensive and vulnerable resource for the attackers. The bank changed its defense strategy based on this insight: when a fraudulent transaction was detected, it was not blocked, but simply flagged and tracked until it was completed and the mule had been revealed. Only then was the attack terminated. This meant that the attacker had to burn through his scare resource – mules – without knowing the odds of success.

The ceiling remained in place even after the Zeus source code was leaked and became widely available around May 2011. Several security firms predicted that this would increase the volume of attacks, as the leak would depress prices of Zeus-related services in the underground economy and further reduce the entry barriers for new attackers (Rashid, 2011; RSA FraudAction Research Labs, 2011).

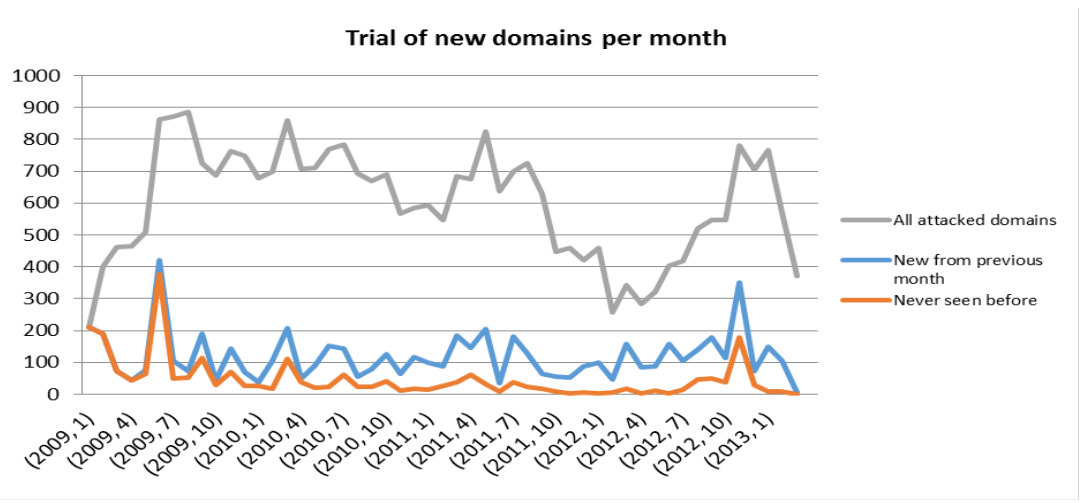


Figure 10- Trial of new Zeus targeted domains per month (2009-2013Q1)

Our results however do not support this prediction, even though there is hardly a shortage of potentially profitable targets.⁸ Table 5 summarizes the point: the last row in the table indicates the number of new domains that were targeted only after the code leak, and the number of new botnet-keys that were activated. The numbers are lower than those for the period prior to the leakage. If we normalize these counts per month, compensating for the fact that the earlier period lasted a bit longer (28 vs. 22 months), the rate before the code leak was 48 botnets per month vs. 33 afterwards (if we leave out the botnets active in both periods).

Table 5 - Number of domains attacked and botnets active before and after Zeus code leakage

	Domains attacked	Active botnets
Only before May 2011	786	1,334
Both before & after May 2011	949	87
Only after May 2011	519	712

The lack of growth in the population of Zeus targets resembles the phenomenon discussed by Florêncio and Herley (2013): the majority of users go unharmed each year, despite the claims of security experts that many attacks are getting cheaper and easier. One of their explanations is victim diversity: if the fraction of all users who succumb to a certain attack is too small then the entire attack is rendered unprofitable. This is especially true when the gains per victim are unclear.⁹

⁸ Others also reported the absence of a rise in attacks (Krebs, 2011).

⁹ An exception to this rule is when the attacker has information at hand showing that the victim is actually valuable. This might explain the existence of the banks from the microstates in the datasets, given the guess is that these are used for instance tax evasion or similar purposes, and belong to wealthy people.

The relatively stable pattern also belies another claim: that the Microsoft-coordinated takedown effort of Zeus command-and-control infrastructure – codenamed B71 – had a noticeable impact on the operations of the attackers. Microsoft never claimed that have fully disrupted Zeus, but rather a “strategic disruption of operations to mitigate the threat” (Domingues Boscovich 2012). Within the population of botnets that we have tracked over the four years, no such disruption is visible. Although there was a temporary dip in activity around the time of the takedown, March 2012, the decline towards that low has started well before operation B71. In fact, briefly after the operation, botnet activity started to rise again to previous levels.

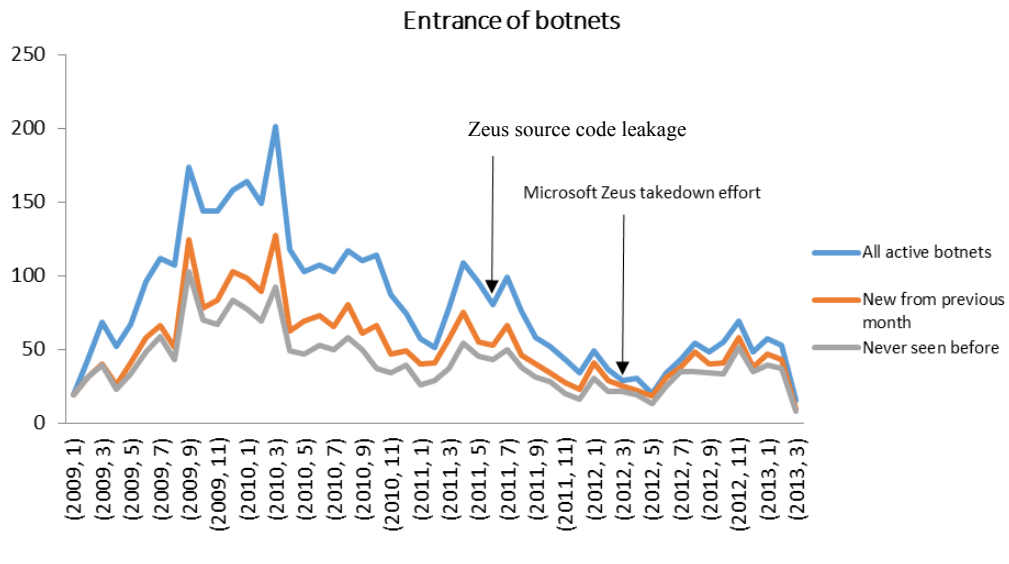


Figure 11 - Entrance of new Zeus attackers (botnet keys) per month

We also took a look into the relationship between botnet lifetime and the different groups of targets in terms of persistency of being attacked. To do that, first we categorized botnets in terms of their lifetime into four different groups (see Table 6). The first group contains botnets that were only active for one day and we treated them as a separate group. For the rest, we divided the total number of botnets or RC4 keys into groups of almost equal size.

Table 6 - Botnets grouped by lifetime

Group	Definition	# Botnets (RC4 keys)
Botnet lifetime 1	Active 1 day	1,315
Botnet lifetime 2	Active >1 and < 30 days	272
Botnet lifetime 3	Active equal or >30 and < 105 days	272
Botnet lifetime 4	Active equal or > 105 days	274

Next, we looked at the relationship between botnet lifetime¹⁰ and the attack persistency category that they attacked. Table 7 shows a cross table of variables ‘attack persistency’ and ‘botnet lifetime’ of number of botnets. As it can be seen from Table 7, more botnets attacked domains that are located in the ‘always attacked’ category rather than the domains that are located in the ‘briefly attacked’ category. Moreover,

¹⁰ Lifetime of each botnet or RC4 key is calculated by subtracting the first and last time that the key is seen. This has strong (rho = 0.97) significant correlation with the number of weeks that a botnet is active.

most of the attacks on domains that are briefly attacked are performed by botnets with the longest lifespan. We hypothesize that those might belong to the most professional attackers, who are able to keep the botnets up and running the longest. The attackers in this category also do the most trial and error, which fits with the hypothesis that they are also the most capable. The number of botnets in this cell is higher than the expected value (observed count: 70, expected count: 28.94)¹¹.

Table 7 – Cross table of number of botnets with different lifetime in different attack persistency categories

		Attack Persistency				
		Briefly attacked	Occasionally attacked	Often attacked	Always attacked	Total
Botnet lifetime 1	Count	62	209	657	1235	1315
	Expected Count	101.121	31.342	736.510	107.025	
Botnet lifetime 2	Count	23	89	217	241	272
	Expect Count	27.537	86.146	200.566	292.750	
Botnet lifetime 3	Count	32	121	236	250	272
	Expect Count	29.397	91.965	214.113	312.524	
Botnet lifetime 4	Count	70	166	252	262	274
	Expect Count	28.943	90.545	210.809	307.701	

Looking at the country of the attacked domains, the pattern seen in other graphs is confirmed. Figure 12 shows the overlap between the targeted countries across four years (2009-2012). Out of the total 92 attacked countries, seven were only attacked in 2009, and seventeen only in 2012. This shift in the variety of the attacked countries, despite the overall stability in the size of the attacks, points to a trial-and-error process with finite resources and players; i.e., the attacks are not spreading like mushrooms.

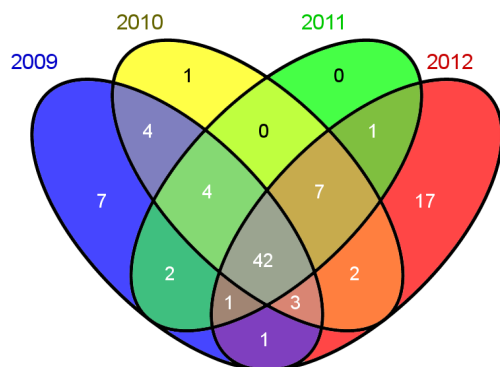


Figure 12 - Venn diagram of location of attacked domains across the years

8 Attack code development

8.1 Descriptive analysis

Our dataset contains 1,146,860 target URLs with associated inject codes. These inject codes are by no means unique. In fact, on average each inject code is repeated 27 times. Figure 13 shows the number of times a specific piece of code is used in different configuration files. Note that virtually all inject codes are reused two or more times among the different configuration files.

¹¹ It should be mentioned that cells in this table are not independent and therefore the applicability of chi-square expected value is limited.

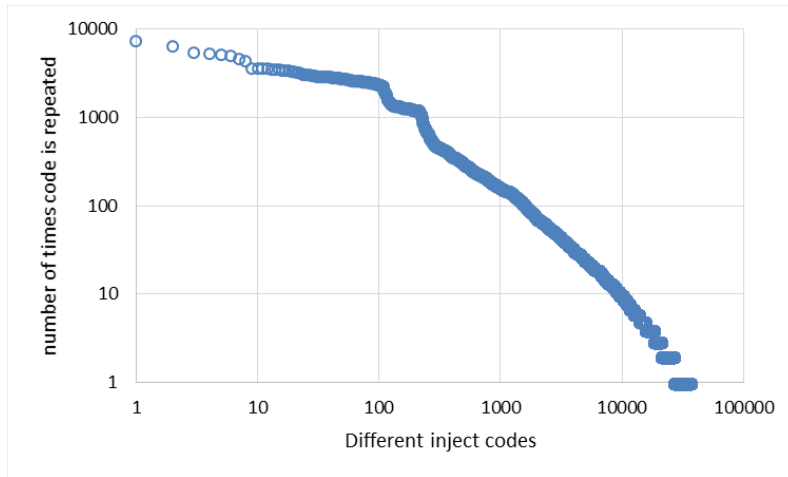


Figure 13 - Number of times an inject code is used among different configuration files

Just 5% of the inject codes was re-used less than 10 times. More striking is the fact that 43% of all inject codes was repeated over 1,000 times. It is a safe bet that any new configuration file found in the wild will contain some inject codes that are identical or trivially different of a previous file. Just 9,679 inject codes (1.19% of the total amount) that were not repeated in any other configuration file. Even in this group, the bulk consisted of slight revisions of previous code aimed at the same URL.

This high amount of code repetition is intriguing. Attackers have clear incentives to reuse old code: it is more cost effective to make incremental modifications and reuse the same proven configuration files than to develop new ones from scratch. The modifications are necessary to evade new security measures. What is puzzling is that the attackers can get away with this little effort. Despite many countermeasures that have been proposed (Mitropoulos et al., 2011), malicious users continue to use the same inject code. This pattern holds both before and after the code leakage.

To acquire a better understanding of these repetitions we analyze the amount of code lines per attack. Results show that the average length of an inject code is around 36 lines (with a standard deviation of 76.8 and coefficient of variation of 2.1). While there are complex attacks with more than 1,000 lines, they represent only 0.05% of the total number of attacks. The majority of the inject codes (56%) range between 10 and 100 lines. These are illustrated in

Figure 14. The high deviation in the number of lines gives an idea of the complexity and diversity of the attacks. Looking at the attack codes, we see a wide range of ideas, from a simple rendering of a page element suggestion to the user to install an older (i.e., more vulnerable) web browser, to larger inject codes containing actual scripts to grab personal information.

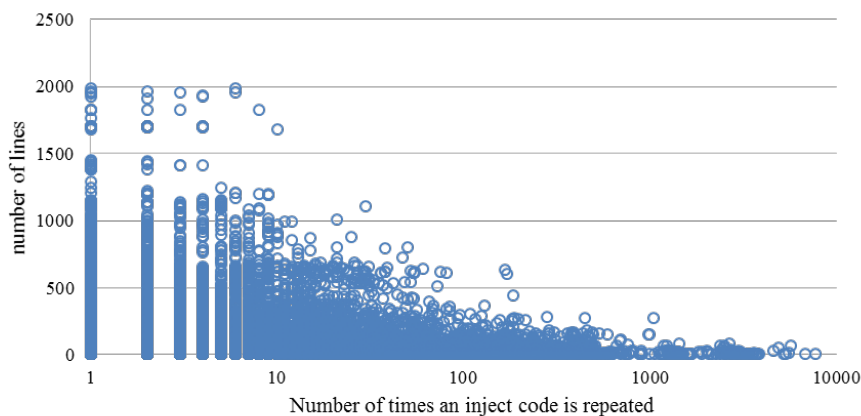


Figure 14 - Number of lines per attack code

8.2 Similarity analysis

To more systematically analyze the level at which attackers re-use existing code, we applied a comparison technique to the whole dataset. Different text comparison techniques can be applied to find discriminating features of different codes. Our approach is an adaptation of text retrieval matching using the so-called Term Frequency-Inverse Document Frequency (*tf-idf*) methods (Jones 1972). These techniques have been widely used for comparing different malware (See e.g. Karnik et al., 2007; Shahzad & Lavesson, 2012; Suebsing & Hiransakolwong, 2009), and for detecting plagiarized documents (Hoad & Zobel, 2003). We also report the differences between two inject codes, expressed as a minimal list of line changes to bring either file into agreement with the other in relation to the total number of code lines (Hunt & McIlroy, 1976).

First, we processed the configuration files to extract the inject code and conform the strings. Then, we tokenized the symbols found using the classic separators (e.g., dot, comma, colon, semi-colon, blank space, tab, etc.). In order to represent a string collection, a common approach in text comparison is to use a Vector Space Model, which represents documents algebraically, as vectors in a multidimensional space. This space consists only of positive axis intercepts. After that, we constructed a text representation of an inject code \vec{C} , which is formed by words s_i , such that $\vec{C} = (s_1, s_2, \dots, s_n)$, n being the number of words within the code. We defined the weight $w_{i,j}$ as the number of times the word s_i appears in the inject code \vec{C}_j ; if s_i is not present in \vec{C}_j , $w_{i,j} = 0$. Therefore, any attack code \vec{C}_j can be represented as the vector of weights $\vec{C}_j = (w_{1,j}s_1, w_{2,j}s_2, \dots, w_{n,j}s_n)$. Finally, we used *tf-idf* weighting schema, where the weight of the i th word in the j th injection code, denoted by $w_{i,j}$, is defined by:

$$w_{i,j} = tf_{i,j} \cdot idf_i = \frac{n_{i,j}}{\sum_k n_{k,j}} \cdot \log\left(\frac{\beta}{\gamma}\right),$$

where $n_{i,j}$ is the number of times the word s_i is not present in \vec{C}_j , $\sum_k n_{k,j}$ is the total number of words in \vec{C}_j , β is the number of codes being compared and γ is the number of codes under comparison that contain the word s_i .

Tf-idf method is based on vector similarity over dampened and discriminatively weighted term frequencies. In our case, we chose the cosine similarity that has proven to be a robust metric for scoring the similarity between two strings (Singhal, 2001). The basic idea behind cosine similarity is to transform each string into a vector in some high dimensional space such that similar strings are close to each other. The cosine of the angle between two vectors is a measure of how “similar” they are, which in turn, is a measure of the similarity of these strings. If the vectors are of unit length, the cosine of the angle between them is simply the dot product of the vectors. Thus, two attacks codes are more similar if they contain many of the same terms with the same relative number of occurrences of each.

Having defined the similarity metric, we used it to compare consecutive attack codes per URL. Figure 15 shows the average similarity per URL. More than 83% of the inject codes targeting a particular URL are more than 90% similar, and only 1.71% of the inject codes are very different (less than 50% similar). On average, across all Zeus botnets and attackers, code similarity is over 90% from one attack to the next. This suggests some mechanism of code sharing or stealing among the attackers.

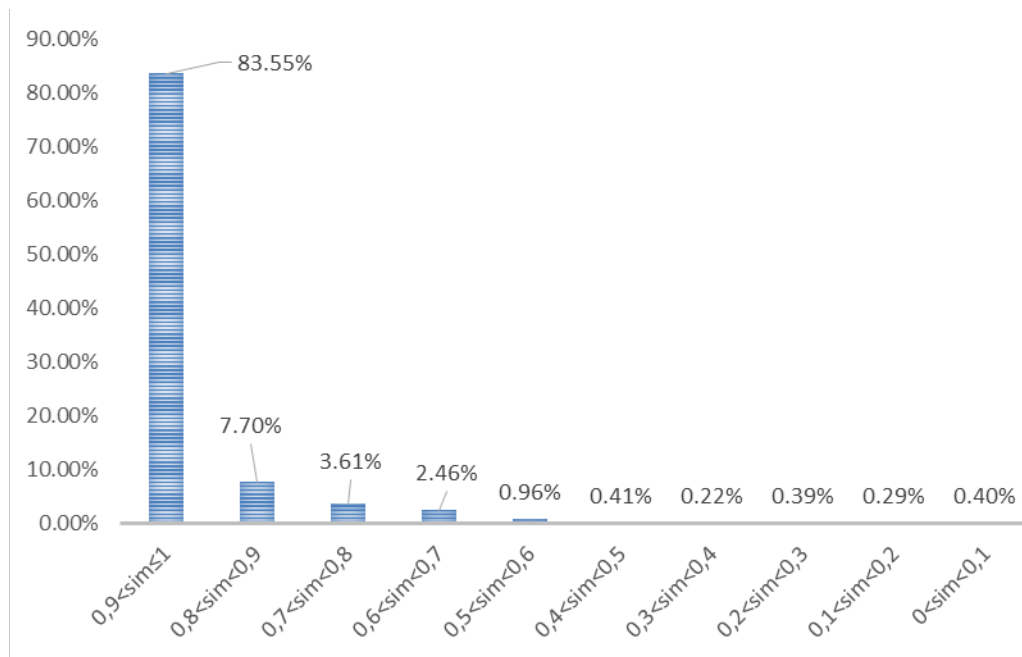


Figure 15 - Average code similarity per URL

If we also take a look a particular URL per a particular botnet, we see that the similarity between consecutive codes increases even more, reaching 97% in average (see Figure 16). A botnet attacking a particular URL rarely changes the inject code between consecutive attacks.

The high similarity between consecutive attacks could be due to (i) incentives of attackers to not change the code substantially if unnecessary, and (ii) operations related to essential characteristics of the targeted URL.

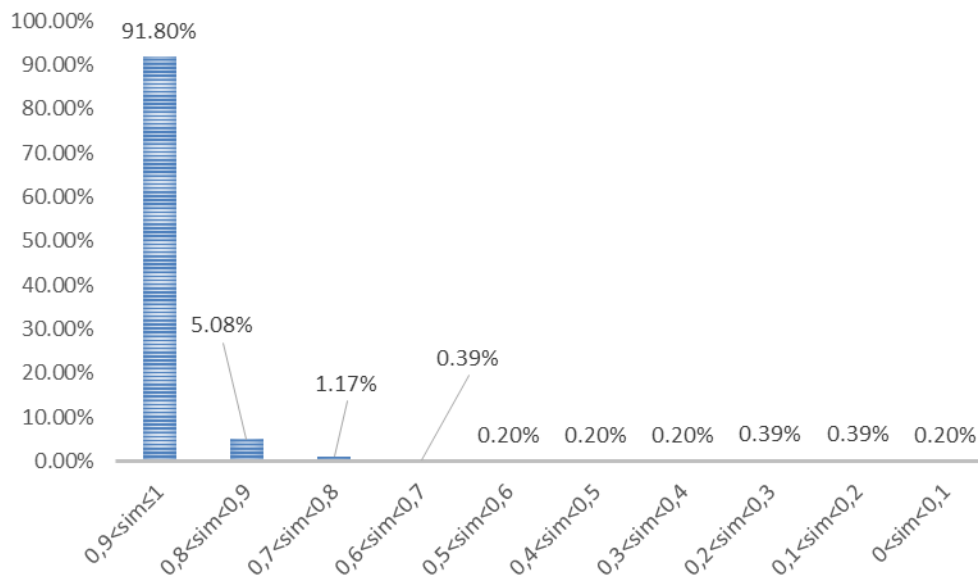


Figure 16 - Average code similarity per botnet and URL

Next, we analyze the impact of code length on similarity. In general, a clear trend does not appear between the two. As Figure 17 shows, large codes are repeated less than small codes, but when repeated, the inject code is more than 97% similar – though this partially reflects the size of the inject, of course. In

smaller injects, changing a few lines will drive down the similarity. In any case, consecutive code attacks to the same URL are more than 90% similar in average no matter their length.

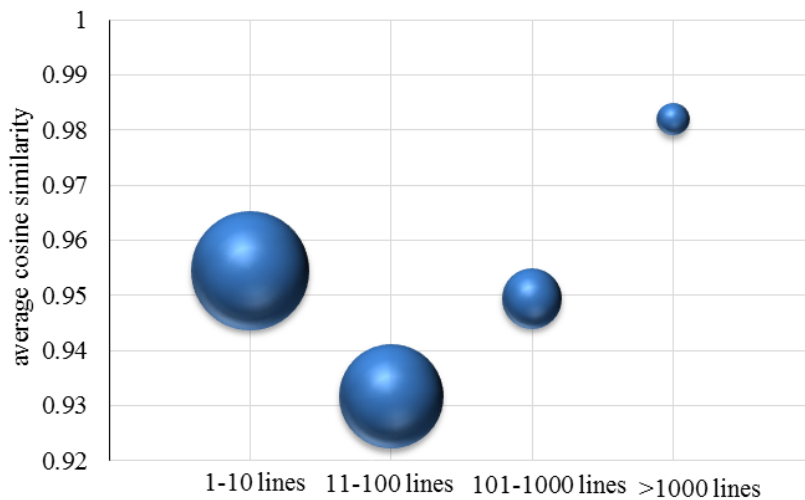
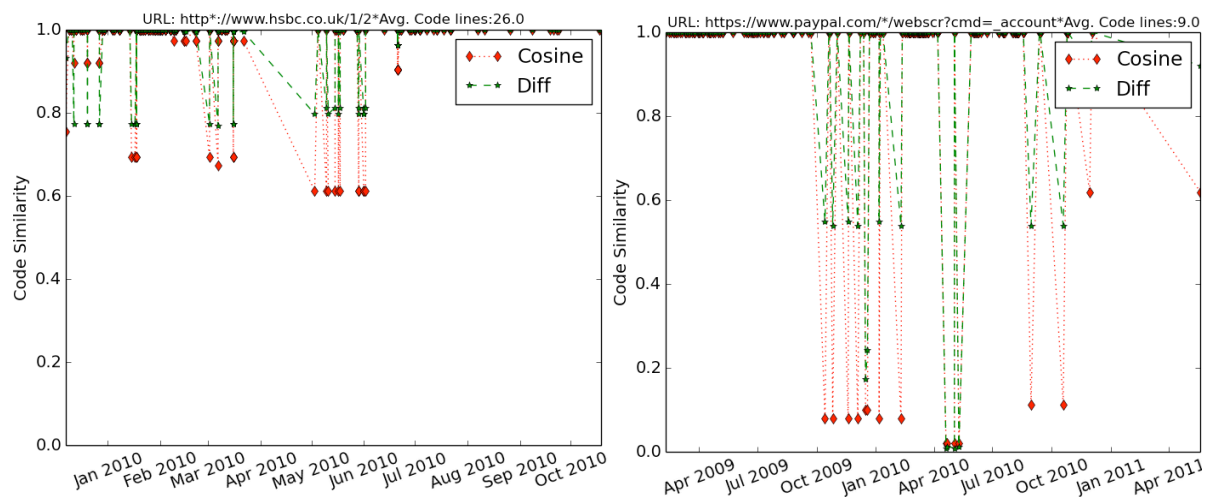


Figure 17 - Average code similarity versus code length and repetition (size of the bubbles)

We took a closer look at the evolution of attack code for four URLs among the top attacked domains (Figure 18). These URLs are from *PayPal*, *HSBC*, *Bank of America*, and *Alliance Leicester*. Besides the cosine similarity, we also compare the number of different code lines between consecutive inject codes. Both metrics follow the same pattern for all the URLs. As one would expect from the analysis so far, the inject codes in most of the instances is the exact copy of their predecessors. However, we can observe that the similarity metric drops at certain points in case of some URLs. These drops most probably reflect changes in the domain's webpage and defense measures by domain owners (financial service providers and other industries) that forced the attackers to adapt their code. In either case, it can be seen that after each drop in similarity, the next codes again become similar. Similarity drops vary in different levels for each of the URLs, reflecting the amount of change. Among these examples, a particular *PayPal* URL suffers from the most abrupt changes in the similarity metric, while the *HSBC* inject code's similarity only drops below 65% one time, with most of the consecutive attacks remaining identical.



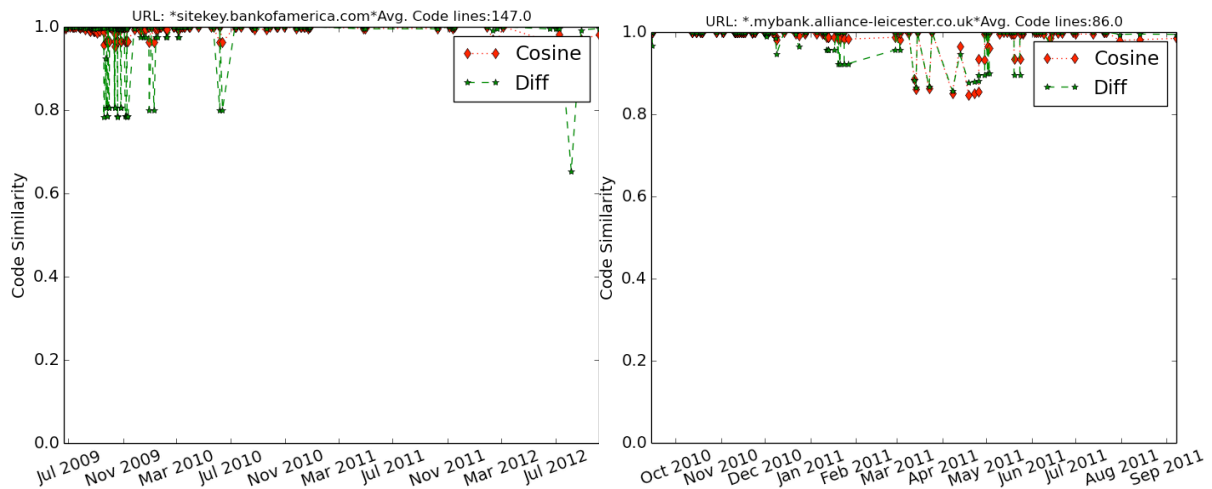


Figure 18 - Code similarity evolution for 4 URLs corresponding to the top attacked domains.

In sum, inject code is not updated with high frequency. This somewhat contradicts the anecdotal descriptions from the industry of highly dynamic cat-and-mouse games with constant adaptations between attackers and defenders. The attack activity observed in this period clearly shows that copies of a previous used inject code and also inject code that only marginally differs from previous ones are present in most configuration files.

In addition, if we take a closer look to the inject code similarity between botnets with different lifetime against targets in each persistence group (Table 8), we realize that in most of the cases code similarity increases in line with botnet lifetime, i.e., botnets that are active longer tend to have smaller code changes. It is unclear what this means exactly. If we assume that more rapid code evolution is a sign of attacker competencies, then this finding provides evidence against our earlier hypothesis that the longer-lived botnets are run by the more competent criminals. An alternative interpretation, however, is that these attackers write more robust code that requires fewer changes to stay functional.

Table 8 - Cross table of similarity of inject code of botnets with different lifetime in different attack persistency categories ($0 < \text{code similarity} < 1$)

		Attack Persistency			
		Briefly attacked	Occasionally attacked	Often attacked	Always attacked
Botnet lifetime 1	Code Similarity	0.893	0.897	0.951	0.926
Botnet lifetime 2	Code Similarity	0.959	0.968	0.970	0.968
Botnet lifetime 3	Code Similarity	0.965	0.970	0.968	0.969
Botnet lifetime 4	Code Similarity	0.966	0.973	0.971	0.972

In general, the overall pattern of high code reuse indicates that financial providers are not implementing changes that require rapid adaptation on the side of the attackers. The lack of code development also suggests that the cost for continuing these attacks is limited. Less skilled attackers could enter the market and survive on minor modifications of existing inject code. That we haven't seen an increase in the volume of attacks is further evidence for the earlier finding that low entry barriers to the criminal market do not translate into attack volume.

9 Conclusion

Financial malware on home computers and mobile devices causes millions of Euro in damages each year. Not every financial service provider is equally popular among cybercriminals. Why are some financial service providers targeted more often than others? There is very little comparative empirical research across providers and countries identifying the factors that contribute to the selection of financial service providers as targets.

This paper set out to explore the incentives and strategies of attackers from the instructions – contained in configuration files – sent to the machines that were infected with Zeus malware from 2009-2013Q1, during which period it was one of the dominant financial malware families. We investigated around 11,000 configuration files targeting 1.2 million URLs, which consisted of 14,870 unique URLs on 2,412 unique domains. We identified the attacked domains, which include financial services as well as other targets, and developed metrics to rank their relative attractiveness as a target, taking into account how criminals update the instructions for the bots under their control.

The attacks were concentrated: around 15% of the domains attracted 90% of the attacks. The concentration is not driven just by target size. Using financial data from FDIC and traffic rankings from Alexa as proxies for the size of the payment service provider, we observe that size is a threshold for getting attacked, but does not predict the intensity of attack.

Attack persistency varies widely, with around half of the domains targeted briefly (4 weeks or less), and 88 domains targeted during the whole period (216 weeks). We believe the brief attacks are part of a process of trial-and-error of attackers seeking new targets. Looking into it from the perspective of botnets, we realized that long-lived botnets are more probable to attack domains in this category comparing to the short-lived ones, which again supports the idea of trial-and-error.

Surprisingly enough, even though new domains are being tried over the whole period, there seems to be a ceiling in the overall number of domains being attacked simultaneously. This suggests bottlenecks elsewhere in the criminal value chain, for example, in the recruitment of money mules or in the involvement of the attackers in other stages of the attack (e.g., the need to take over banking sessions in real time). Despite what is expected, the ceiling remained in place both in terms of number of domains that were attacked and in terms of number of new botnets that entered the market even after when Zeus source code was leaked and became widely available. This suggests that in this market, low entry barriers do not translate into more crime.

We also studied the evolution of inject code over time. Using a cosine similarity metric, we compared the 1.2 million inject codes in the dataset. In short, the vast majority of the inject codes were merely modifications of previous codes. In fact, only 3,664 attacks were not exact copies of a previously seen code. In any case, consecutive code attacks to a same URL are more than 90% similar regardless of the length of the attack code. This suggests that attacks are much less dynamic than often presumed.

Code gets re-used to a remarkable degree: just 1% of the inject code is never repeated, and 226 different inject codes are repeated over one thousand times without any modifications. On average, across all Zeus botnets, code similarity is well over 90% from one attack to the next. This suggests some mechanism of code sharing or stealing. Within a same botnet, similarity goes up to 97%. Overall, it seems that cost of code development for attackers is limited. This could lower entry barriers and increase attacks, but as we found earlier, this does not occur. Entry barriers are not the factor that is keeping attack levels in check.

At a more general level, the implication of these findings might be that the underground markets for infected machines, malware-as-a-service, which have been portrayed as making attacks cheaper to execute and even as opening up cybercrime to the unschooled masses, are not main force in driving the attack volume, nor the selection of targets. This suggests that there is a need for more investigation on other parts of online banking fraud value chain such as money mules, or banks' money transferring policies. If the bottlenecks are not in the malware ecosystem, then investing in disrupting the ecosystem by defenders and law enforcement may not actually be the best allocation of scarce resources.

References

- [1]. Adham, Manal, Azodi, Amir, Desmedt, Yvo, & Karaolis, Ioannis. (2013). How to Attack Two-Factor Authentication Internet Banking. In A.-R. Sadeghi (Ed.), *Financial Cryptography and Data Security* (pp. 322-328): Springer Berlin Heidelberg.
- [2]. Alexa Internet. (2014). Alexa - the Web Information Company. from <http://www.alexa.com>
- [3]. Anderson, Ross, Barton, Chris, Böhme, Rainer, et al. (2013). Measuring the Cost of Cybercrime. *The Economics of Information Security and Privacy*, 265-300.
- [4]. Baylor, Ken. (2014). The Cutting Edge Is Honed: NSSLABs.
- [5]. Bikhchandani, Sushil, Hirshleifer, David, & Welch, Ivo. (1992). A Theory of Fads, Fashion, Custom, and Cultural Change as Informational Cascades. *Journal of Political Economy*, 100, 992-1026.
- [6]. Christin, Nicolas, Yanagihara, Sally S, & Kamataki, Keisuke. (2010). *Dissecting One Click Frauds*. Paper presented at the Proceedings of the 17th ACM conference on Computer and communications security.
- [7]. Claessens, Joris, Dem, Valentin, De Cock, Danny, Preneel, Bart, & Vandewalle, Joos. (2002). On the Security of Today's Online Electronic Banking Systems. *Computers & Security*, 21, 253-265.
- [8]. Cranor, Lorrie Faith, Idouchi, Kelly, Leon, Pedro Giovanni, Sleeper, Many, & Blase, Ur. (2013, June 2013). *Are They Actually Any Different? Comparing Thousands of Financial Institutions' Privacy Practices*.
- [9]. Domingues Boscovich, Richard. (2012). Microsoft and Financial Services Industry Leaders Target Cybercriminal Operations from Zeus Botnets. *The Official Microsoft Blog - TechNet Blogs*. from http://blogs.technet.com/b/microsoft_blog/archive/2012/03/25/microsoft-and-financial-services-industry-leaders-target-cybercriminal-operations-from-zeus-botnets.aspx
- [10]. European Central Bank. (2014). "Third Report on Card Fraud". Available from European Central Bank. Frankfurt. <http://www.ecb.europa.eu/press/pr/date/2014/html/pr140225.en.html>
- [11]. F-Secure. (2012). Threat Report H1 2012., from https://www.f-secure.com/static/doc/labs_global/Research/Threat_Report_H2_2012.pdf.
- [12]. Falliere, Nicolas, & Chien, Erin. (2009). Zeus: King of the Bots | Intellectual Takeout (Ito). from <http://www.intellectuالتakeout.org/library/research-analysis-reports/zeus-king-bots>
- [13]. FDIC. (2014). Federal Deposit Insurance Corporation: Institution Directory. from <http://www2.fdic.gov/idasp/main.asp>
- [14]. FFA UK. (2013). "Fraud the Facts 2013". Available from Financial Fraud Action UK. London. <http://www.financialfraudaction.org.uk/>.
- [15]. Florêncio, Dinei, & Herley, Cormac. (2013). Where Do All the Attacks Go? In B. Schneier (Ed.), *Economics of Information Security and Privacy Iii* (pp. 13-33): Springer New York.
- [16]. Garg, Vaibhav, & Camp, L Jean. (2013). *Macroeconomic Analysis of Malware*. Paper presented at the 20th Annual Network and Distributed System Security Symposium, San Diego, California, USA.
- [17]. Hoad, Timothy C., & Zobel, Justin. (2003). Methods for Identifying Versioned and Plagiarized Documents. *J. Am. Soc. Inf. Sci. Technol.*, 54, 203-215.
- [18]. Hunt, James Wayne, & McIlroy, M Douglas. (1976). *An Algorithm for Differential File Comparison*: Bell Laboratories.
- [19]. Jaleshgari, Ramin P. (1999). Document Trading Online. *InformationWeek*, 755, 136.
- [20]. Karnik, Abhishek, Goswami, Suchandra, & Guha, Ratan. (2007). *Detecting Obfuscated Viruses Using Cosine Similarity Analysis*. Paper presented at the Modelling & Simulation, 2007. AMS'07. First Asia International Conference on.
- [21]. Krebs, Brian. (2011). Something Old Is New Again: Mac Rats, Crimepacks, Sunspots & Zeus Leaks. from <http://krebsonsecurity.com/2011/05/something-old-is-new-again-mac-rats-crimepacks-sunspots-zeus-leaks/>
- [22]. Krebs, Brian. (2012). Thieves Replacing Money Mules with Prepaid Cards? — Krebs on Security. Retrieved from <http://krebsonsecurity.com/2012/04/thieves-replacing-money-mules-with-prepaid-cards/>
- [23]. Kruse, Peter. (2011). Csis: Complete Zeus Sourcecode Has Been Leaked to the Masses. from <https://http://www.csis.dk/en/ctsis/blog/3229/>
- [24]. Levchenko, Kirill, Pitsillidis, Andreas, Chachra, Neha, et al. (2011). *Click Trajectories: End-to-End Analysis of the Spam Value Chain*. Paper presented at the Security and Privacy (SP), 2011 IEEE Symposium on.
- [25]. Li, Shujun, Sadeghi, Ahmad-Reza, Heisrath, Sören, Schmitz, Roland, & Ahmad, Junaid Jameel. (2012, 2012). *Hpin/Htan: A Lightweight and Low-Cost E-Banking Solution against Untrusted Computers*

- [26]. Macdonald, Doug. (2011). Zeus, God of Diy Botnets. *FortiGuard Center*. from <http://www.fortiguard.com/legacy/analysis/zeusanalysis.html>
- [27]. MaxMind. (2014). Geopip | Ip Address Location Database. from http://www.maxmind.com/en/geolocation_landing
- [28]. Mitropoulos, Dimitris, Karakoidas, Vassilios, Louridas, Panagiotis, & Spinellis, Diomidis. (2011). Countering Code Injection Attacks: A Unified Approach. *Information Management and Computer Security*, 19, 177–194.
- [29]. Moore, Tyler, & Clayton, Richard. (2007, 2007). *Examining the Impact of Website Take-Down on Phishing*.
- [30]. Murdoch, Steven J., & Anderson, Ross. (2010). Verified by Visa and Mastercard Securecode: Or, How Not to Design Authentication. *Financial Cryptography and Data Security*, 336-342.
- [31]. Murdoch, Steven J., Drimer, S., Anderson, R., & Bond, M. (2010). *Chip and Pin Is Broken*. Paper presented at the 2010 IEEE Symposium on Security and Privacy (SP).
- [32]. NVB. (2013). "Scherpe Daling Fraude Internetbankieren". Available from Nederlandse Vereniging van Banken. from <http://www.nvb.nl/nieuws/2013/1812/scherpe-daling-fraude-internetbankieren.html?cookie=set>
- [33]. ODP- Open Directory Project. (2014). from <http://www.dmoz.org>
- [34]. OSCP. (2013). "Rapport Annuel 2012". Available from Observatoire de la sécurité des cartes de paiement.Paris. <http://www.banque-france.fr/observatoire/telechar/2013/Rapport-annuel-2012.pdf>.
- [35]. Premchaiswadi, Nucharee, Williams, James G., & Premchaiswadi, Wichian. (2009, October 2009). *A Study of an on-Line Credit Card Payment Processing and Fraud Prevention for E-Business*.
- [36]. Rashid, Fahmida Y. (2011). Zeus Source Code Leak Means Even More Banking Malware to Hit the Web. from <http://www.eweek.com/c/a/Security/Zeus-Source-Code-Leak-Means-Even-More-Banking-Malware-to-Hit-the-Web-253343/>
- [37]. RSA FraudAction Research Labs. (2011). Fraud News Flash – the Downfall of the Mighty – Zeus Trojan’s Source Code Leaked and Now Available Everywhere. *The RSA Blog and Podcast*. from <https://blogs.rsa.com/fraud-news-flash---the-downfall-of-the-mighty---zeus-trojan's-source-code-leaked-and-now-available-everywhere/>
- [38]. Shahzad, R.K., & Lavesson, N. (2012, August 2012). *Veto-Based Malware Detection*. Paper presented at the 2012 Seventh International Conference on Availability, Reliability and Security (ARES).
- [39]. Sherstobitoff, Ryan. (2013). Inside the World of the Citadel Trojan. from <http://www.mcafee.com/uk/resources/white-papers/wp-citadel-trojan.pdf>
- [40]. Singhal, Amit. (2001). Modern Information Retrieval: A Brief Overview. *Bulletin of the IEEE Computer Society technical committee on data engineering*, 24, 2001.
- [41]. Suebsing, Anirut, & Hiransakolwong, Nualsawat. (2009). *Feature Selection Using Euclidean Distance and Cosine Similarity for Intrusion Detection Model*. Paper presented at the Intelligent Information and Database Systems, 2009. ACIIDS 2009. First Asian Conference on.
- [42]. Sullivan, Richard J. (2010). The Changing Nature of U.S. Card Payment Fraud: Industry and Public Policy Options. *Economic Review*, 101-133.
- [43]. TrendMicro. (2012). Security Threats to Business, the Digital Lifestyle, and the Cloud. from <http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/spotlight-articles/sp-trend-micro-predictions-for-2013-and-beyond.pdf>
- [44]. Utakrit, Nattakant. (2009, 2009). *Review of Browser Extensions, a Man-in-the-Browser Phishing Techniques Targeting Bank Customers*.
- [45]. van Eeten, Michel, Bauer, Johannes M, Asghari, Hadi, Tabatabaie, Shirin, & Rand, Dave. (2010). The Role of Internet Service Providers in Botnet Mitigation: An Empirical Analysis Based on Spam Data: OECD Publishing.
- [46]. Wyke, James. (2011). What Is Zeus? from <http://www.sophos.com/en-us/why-sophos/our-people/technical-papers/what-is-zeus.aspx>
- [47]. Yar, Majid. (2005). The Novelty of ‘Cybercrime’ an Assessment in Light of Routine Activity Theory. *European Journal of Criminology*, 2, 407-427.