Interactive Website Filter for Safe Web Browsing

INSOON JO[1], EUNJIN (EJ) JUNG[2] AND HEON Y. YEOM[1]
[1]*School of Computer Science and Engineering*
Seoul National University
Seoul, 151-742 Korea
[2]*Department of Computer Science*
University of San Francisco
San Francisco CA 94117, USA

Though popularly used for safe web browsing, blacklist-based filters have fundamental limitation in the "window of vulnerability", the time between malicious website launch and blacklist update. An effective way of seamless protection is to use an add-on filter based on heuristics, but most of prior heuristics have offered the limited scope of protection against new attacks. Moreover, they have either suffered from low detection accuracy or incurred unacceptable slowdown. This paper presents an interactive website filter based on heuristics for detecting malicious websites. As the key feature, our filter considers the disparity between a website's true identity (e.g., host domain) and its observed identity (e.g., frequent terms or source domains of iFrames). A website with significant disparity is considered as malicious. Users are warned against a website identified as malicious, and determine if it is safe to proceed. Incorporating user-interaction into discovering the true identity of the suspect websites lets our filter avoid false positives caused by automatic detection. Our main contribution is that we found a common and efficient characteristic to filter malicious websites. Not only is such disparity inherent in exploit mechanisms of malicious websites whether to aim for phishing or malware distribution, but its measuring by textual relevance incurs negligible overhead. Experimental results demonstrate that our filter is lightweight while delivering considerably high detection accuracy for both malicious websites.

Keywords: phishing, malware distribution, drive-by downloads, browser extension, usable security, machine learning, reasoning

1. INTRODUCTION

According to the Gartner survey, 75% of attacks on the Internet are now believed to come through the Web. Websense reported an alarming rise in the growth of the malicious websites in 2010 compared with the year before – almost 111.4% [1], where malicious websites include phishing [2] and malicious software (malware) distribution websites [3]. Considering these, it is important to filter these malicious websites before users visit them. As web browsers like FireFox and Internet Explorer do, popular way to filter them is using blacklist. However, blacklist-based systems are not effective in the "window of vulnerability", i.e. between the launch of malicious websites and their registration in the blacklist. An effective way of seamless protection is using heuristics as a browser extension, which can be used either alone or in conjunction with blacklists in order to deliver higher detection accuracy. There are three conditions heuristics must satisfy to be used as an add-on filter. First, their scope of detection should not be limited. Secondly,

* This research was supported by NSF grant 1063745 and the KCC (Korea Communications Commission), Korea, under the CPRC (Communications Policy Research Center) support program supervised by the KCA (Korea Communications Agency) (KCA-2011-1194100004-110010100).

they should deliver reasonable detection accuracy. Lastly, they should not incur such a significant overhead to cause noticeable delay in web browsing.

We argue that prior heuristics are not suitable to run with web browsers, because no heuristics satisfy these conditions. All heuristics except *Beyond Blacklists* [4] have the limited scope of detection. That is, they target either phishing or malware distribution websites, not both of them. Half protection would be no barrier to attack, because unprotected scope is fully open. On the other hand, using two filters for full protection will impact browsing performance. However, the scope of detection is not the only problem. Early heuristics [5, 6] focused on html anomalies. However, their main anomalies such as string encoding and small iFrame are not malicious themselves, and benign websites use them if necessary. Due to these confusing features, anomaly heuristics have suffered from low detection accuracy. To address this accuracy issue, later heuristics have either considered URL/html anomalies and a set of external references together [4, 7], or monitored malicious activities by dynamic emulation [8-12]. These heuristics may be quite precise, but their analysis is expensive. Therefore, they are appropriate for back-end filtering at a large scale, not real-time detection.

This paper presents an interactive website filter based on heuristics for detecting malicious websites. Our filter targets both phishing and malware distribution websites. As the key feature, it considers the disparity between a website's true and observed identities. Given a website, our filter derives clues about its true identity (e.g., host domain) and observed identity (e.g., frequent terms or source domains of iFrames) from its WHOIS record, URL, and content. Then, it measures the disparity between these identities by their textual relevance. Our main contribution is that we found a common and efficient characteristic to identity malicious websites: the disparity between two kinds of identities. The reason why almost all the prior heuristics have offered half protection is that they believed there would be no common characteristic between malicious websites using different exploit mechanisms. However, such disparity is inherent in exploit mechanisms of malicious websites whether to aim for phishing or malware distribution. Besides, it can be efficiently measured by textual relevance. A website with significant disparity is considered as malicious. Users are warned against a website identified as malicious, and decide whether to visit this suspicious website or not. This user interaction helps our filter discover more knowledge (true identity) on the website in question. The reason why our filter does not completely rely on automation is that no automatic heuristics can achieve perfect detection accuracy. Involving users in validation process helps our filter avoid false positives. Experimental results show that our mechanism delivers considerably high detection accuracies with negligible overhead.

Many attempts have been made to address this issue of malicious website detection. We show that these attempts are not as general, lightweight, and reasonably accurate as ours in Section 2. Section 3 covers how our filter works. In Section 4 we present experimental results. Section 5 summarizes the contributions of this work.

2. RELATED WORKS

There have been many heuristics to identify malicious websites. However, none of them is suitable for a lightweight add-on filter, which can detect both of phishing and

malware distribution websites with reasonable accuracy.

Although there have been many efforts in identifying malicious websites, detecting both has not been considered intensively. As far as we know, *Beyond Blacklists* [4] is the only approach to identify both websites. Given a website, it uses lexical properties of URL, WHOIS and DNS records, blacklists, and geographical information, but ignores its content. Querying multiple external servers incurs overhead concern, so this approach is appropriate for back-end filtering at a large scale. Moreover, disregarding content may lead to wrong decision. For example, this will consider malicious websites launched in compromised legitimate domain and not in blacklist as benign. According to Canali *et al.* [7], *Beyond Blacklists* showed low detection accuracy.

As to detecting phishing websites, the most accurate heuristics have referenced search engines [13, 14]. Given a webpage, they extract search terms from html using TF/IDF, and send them to one or more search engines such as Google and Yahoo!. If its domain name is in top N search results, then it is deemed safe. However, the roundtrip time to search engines must be so long that authors noted that their major limitation is performance problem involved in querying search engines. Besides, there is a severe privacy concern in sending keywords that identify every page a user visit. This makes it possible to easily track all of user's behavior on the web.

As to detecting malware distribution websites, early heuristics focused on html anomalies [5, 6]. However, their main anomalies such as presence of small size elements, long strings, and string encoding/decoding are very confusing. For example, passing an encoded string to unescape/eval functions is a very common obfuscation technique used in malicious websites against string lookup detection [15]. However, this technique is also used in benign scripts as shown in Fig. 1, a very common script for Google analytics. Html anomalies are not malicious, and benign websites can contain them if necessary. Anomaly heuristics are very likely to fail on correctly identifying those benign websites.

```
var gaJsHost = (("https:"== document.location.protocol) ? "https://ssl ." : "http://www.");
document.write(unescape("%3Cscript src='" + gaJsHost + "google-analytics.com/ga.js'
type = 'text/javascript'%3E%3C/script %3E"))
```

er
mu
[4.

lookup delay caused from multiple external queries cannot be short. Especially, DNS lookup is an issue. Over the past years, websites have increasingly used third party DNS resolvers, which provide advanced services such as phishing site blocking and suggestions for failed lookups. Contrary to local or ISP resolvers, third party resolvers can be located overseas, and their location has a noticeable user impact [17]. The latter have rendered a website either in a real browser or in an emulated browser, and monitored malicious activities [8-12]. Actually, the tools deploying dynamic heuristics run for a time ranging from tens of seconds to even tens of minutes for a single page. These resource-intensive techniques must incur infeasible delay in web browsers. Therefore, two later heuristics are not appropriate for real-time detection.

3. SYSTEM DESCRIPTION

Our heuristics assume that there is a high disparity between a website's true and observed identities if it is malicious. We insist that such disparity is inherent in the exploit mechanisms of malicious websites, and discuss its detail below.

3.1 Phishing Websites

Phishing is an attack where fraudulent websites impersonate legitimate counterparts to steal users' confidential information. To lure users into their websites, attackers clone the content of victim websites, but do not care about domain replication. Not only do most users fall for websites that have only content similarity in visual cues [18, 19], but creating an exact replica of domain name is a non-trivial attack by itself. Attackers should either spoof DNS records or hack DNS servers. Given how successful content replication is, there is little incentive on domain replication. In result, attackers often use one domain to host simultaneous attacks against multiple victim websites [20]. This trend has caused significant disparity between true identity and observed one in phishing websites.

Fig. 2 and Fig. 3 show a typical example of phishing website and its legitimate counterpart.

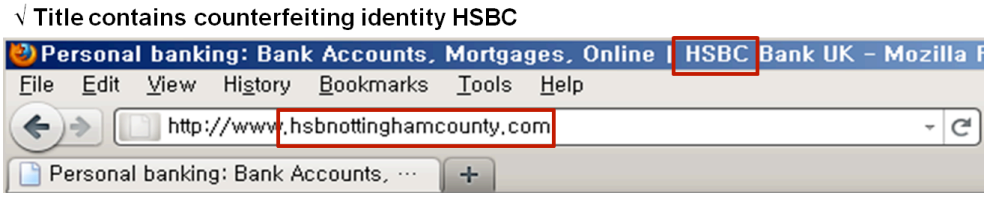


Fig. 2. A HSBC phishing website (http://www.hsbnottinghamcounty.com).

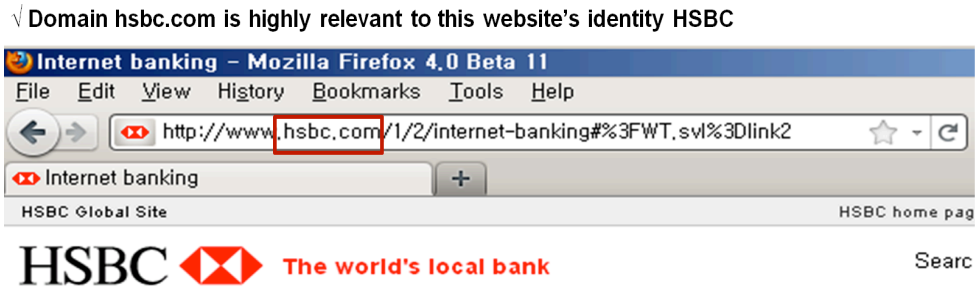


Fig. 3. The legitimate HSBC website (http://www.hsbc.com/1/2/internet-banking).

In this example, both websites claim to be HSBC, but the phishing website clearly shows less relevance (greater disparity) between its true identity and observed identity clues. In the phishing example in Fig. 2, we can obtain the observed identity clues such as "HSBC" from its title and image/anchor domain "hsbc.com" from the website content. In this case, the textual relevance between the observed identity clues (i.e., HSBC and

hsbc.com) is very high. Contrary to this, the textual relevance between the true identity and observed identity clues (i.e., hsbnottinghamcounty.com and HSBC, hsbnottingham-county.com and hsbc.com) is low. Consistency in observed identity clues but significant disparity between the true and observed identities suggests that this is a phishing website. The observed identity clues are what the malicious website wants users to believe as its identity, thus have higher relevance among them than to the true identity. On the other hand, in the legitimate website in Fig. 3, we can obtain the observed identity clue "HSBC" from its WHOIS record and copyright holder, but all domains in content belong to host. Because content contains no other domains than host domain and the textual relevance between the true and observed identities (i.e., hsbc.com and HSBC) is high, this website can be considered as benign.

3.2 Malware Distribution Websites

Malware distribution websites use two kinds of exploit mechanisms. First mechanism requires user's consent, and it uses exactly the same "look-and-feel" lure as phishing websites [21, 22]. By impersonating popular legitimate websites, these websites dupe users into infecting themselves by clicking on the links that make them to download malware. Because using the same exploit mechanism as phishing websites, they must have the strong disparity between their true and observed identities. The second and more popular mechanism is known as drive-by-download, which causes user's web browser to download malware without user's consent.

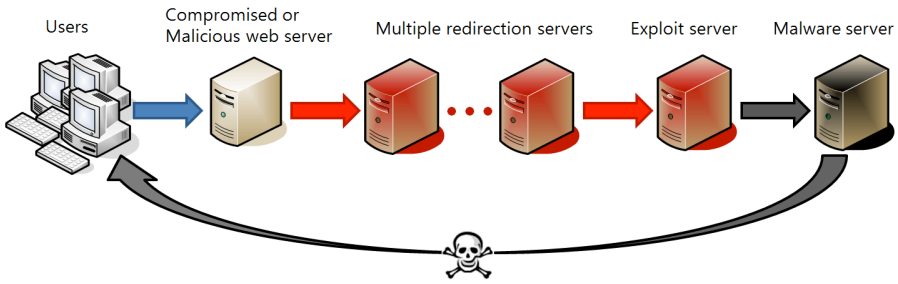


Fig. 4. Malware distribution networks.

Fig. 4 shows a typical scenario of drive-by-download attacks [3]. Users visit either compromised or malicious websites, which include iFrame/script elements that cause victim's web browser to request another page. Then, victim's browser secretly loads specified page, and the page redirects to another page at exploit server. If the exploit succeeds, malware is downloaded from malware server to victim's computer. Victims usually experience long redirection steps (more than 6 redirection steps in 50% cases [3]), because it is increasingly difficult to maintain trust along such long delivery chains.

During the delivery chain, our filter targets websites from exploit server, which have three characteristics. First, they are at the end of redirection steps. Secondly, they contain iFrame/script elements pointing to malware payload from malware server. Lastly, they ignore professional "look-and-feel" lure. Because user's browser is automatically redirected to their websites, they have no need to try to lure users into their websites. Con-

trary to phishing website, they display an empty or incoherent content. Therefore, it is not easy to obtain useful clues about the observed identity from their content. However, we can obtain the source domains of iFrame/script elements pointing to malware payload, which would have low textual relevance to host domain. Because the most malware serving networks are composed of tree-like structures with strong fan-in edges leading to the main malware distribution sites.

Fig.5 shows a website source from exploit server. This website satisfies three characteristics mentioned above. That is, it was at the end of multiple redirections, contained iFrame element pointing to malware payload, and displayed nothing. The only observed identity clue is "nuvolokijj.com", and this is textually irrelevant to the true identity "traffcin002.com". Therefore, our filter will consider this website as malicious.

```
<frameset rows="100%"><frame src="http://nuvolokijj.com/x/?src=dg&id=20689"></frameset>
```

Malicious websites tend to have poorly managed WHOIS records. For example, basic records such as registrant are missing or WHOIS lookup does not even succeed. Given a URL, our filter extracts its host domain, and sends WHOIS query if the lookup record of the corresponding domain is not cached. Then, it extracts information such as registrant, registration/update/expiration dates, and name server domains.

WHOIS lookup may raise concern about roundtrip delay and privacy. To address this issue, our filter performs lookup per domain rather than per URL, and caches lookup records locally. Therefore, there is no delay caused from WHOIS lookup when users visit websites visited before. For a website which is not visited before and lookup record of its host domain is not cached, WHOIS query is sent. Though users browse more pages in the website, there is no additional lookup. Not only does our filter use cached records, but it performs lookup per domain. Contrary to using search engine results, attackers can track neither all visited domains nor the users' behavior in a specific domain.

3.2.2 URL Features

Phishing websites tend to use URL obfuscation techniques whereas other websites hardly do. We categorized these obfuscation techniques into six different types. The first three (Type I, II, III) were also used in [23]. Drive-by-download websites commonly use insecure connection and multiple redirections. Type V and VI represent these characteristics. Table 1 provides details about these features.

Table 1. URL features and their examples.

Type I: presence of IP address or port number

e.g., http://85.30.110.170:57
Type II: domains embedded in URL except hostname e.g., http://arihantinfocom.com/www.uk.hsbc.co.uk/IBlogin.html
Type III: domains embedded in hostname e.g., http://myonlineaccounts2.abbeynational.co.uk.watersidehoa.net/
Type IV: presence of obfuscation with unusual characters e.g., http://www.kodakdalby.com/Services\%20We\%20Offer_files/images/
Type V: whether this website uses https connection
Type VI: whether this website is sent through any chain of redirections

3.2.3 Content Features

Content features contain ample clues about the observed identity. We provide details about a subset of these features and discussion below.

Type I: Entity Names. Websites for public users usually contain their entity names in head, meta, title, and copyright information. Because phishing and some malware distribution websites clone victim websites, their content contains victims' names as they are. Our filter collects entity names from elements such as title and copyright holder.

Type II: Frequent Terms. Websites for public users repeat their entity names throughout the websites. Cloned websites contain them as they are. Our filter collects the terms not HTML keywords and their frequencies, and chooses the most frequent terms.

Type III: Resource Domains. Websites usually use various resources such as images, links, and scripts. In the case of legitimate websites, most resources reside in host. Contrary to this, cloned websites still contain the references to the victims' domain. Drive-by-download websites refer to external malware via iFrame/script elements as well. The reference to external location can be a good clue to identity malicious websites. Our filter collects source domains from img, link, anchor, form, script and iFrame elements.

Type IV: Malicious Behavior. If a website forwards user's input to external location or most resources specify external locations, this website is highly suspicious. Our filter counts the number of form submissions to external locations and the number of resources with external source locations.

3.3 Disparity Measure

The disparity measure is the key of our heuristics for filtering malicious websites. We consider that the greater the disparity is, the more malicious a websites is. Disparity is measured by string similarity via q-gram distance metric [24]. Because the similarity ranges from zero to one, a website with zero-similarity has the greatest disparity, and considered as completely malicious. Based on collected features, our filter builds two candidates sets: Common Name and Domain Name candidates. The observed identity consists of these two. One way to distinguish them is that the Common Name candidates correspond to entity names, and Domain Name candidates correspond to domain names

except host domain. Table 2 represents example features contributing to Common Name or Domain Name candidates. Note that we exclude host domain from Domain Name candidates because it is not a candidate but rather a true/confirmed identity.

Table 2. Example of Common Name and Domain Name candidates.

	Common Name candidates	*Domain Name candidates*
WHOIS features	Registrant Entity name	Name server domains
URL features		Embedded domains in hostname Embedded domains in URL except hostname
Content features	Title Most frequent terms Copyright holder	Form submission domains Source domains of images Source domains of links/anchors Source domains of scripts/iFrames

Our filter first checks how consistent the observed identity clues are, and then how consistent the true identity and observed identity clues are. First, it measures the disparity between the observed identity clues. The observed identity consists of Common Name and Domain Name candidates, and we consider them as sets. Therefore, we can make subsets with two different elements. Our filter computes the string similarity of elements in each subset with the form {CN, CN}, {DN, DN}, and {CN, DN}, where CN is an element from Common Name candidates and DN is an element from Domain Name candidates. Then, it measures the disparity between the true identity and the observed identity clues. That is, it computes the string similarity of elements in each subset with the form {host domain, CN} and {host domain, DN}.

In the case of legitimate websites, not only their observed identity clues (e.g., HSBC from registrant, copyright holder, and frequent term) are textually relevant, but they have a high textual relevance to host domain (e.g., hsbc.com). On the other hand, if the textual relevance between the observed identity clues (e.g. HSBC and hsbc.com) are much higher than that between the true identity and the observed identity clues (e.g. hsbnottinghamcounty.com and HSBC), this website is highly suspicious. Usually, phishing websites focus on professional "look-and-feel" lure, and have the obvious identity claims. They provide many observed identity clues, and the disparity between the true and observed identities is significant. Contrary to this, drive-by-download websites ignore such visual lure. Because they may have poor WHOIS records, omit the title, or not even show anything to users, it is not easy to gather useful clues from them. Even though the textual relevance between the observed identity clues is low, our filter considers a website as malicious if it is redirected and the textual relevance between host domain (e.g., traffcin002.com) and Domain Name candidates (e.g., nuvolokijj.com) is low as well. As shown in our experiments, the disparity between host domain and Domain Name candidates works effectively against drive-by-download websites.

3.4 Composite Classification Scheme

Our filter adopted composite classification scheme [25], which combines several

classifiers in order to reduce misclassification while enhancing performance. In the training phase, each classifier is assigned weight of its decision. And the decision for test data is made with the weighed sum of each classifier decision.

Our filter combines two classifiers, and assigns the equal weight to them. The reason for using two classifiers together is that the effective features and their value characteristics are quite different according to exploit mechanism. Let us assume that tr_{hor} denotes the textual relevance between the observed identity clues, and tr_{ver} does the textual relevance between the true identity and observed identity clues. In the case of websites impersonating victim websites, their identity claim is very obvious, and they have many Common Name candidates. Their tr_{hor} is not only very high but significantly higher than tr_{ver}. On the other hand, the identity claim is vague in drive-by-download websites, which may have scanty Common Name candidates. They are redirected, and their tr_{hor} and tr_{ver} are very low.

To apply composite classification scheme, given a website, our filter builds two feature sets. The first set consists of features effective for identifying impersonating websites, and the second set consists of features effective for identifying drive-by-download websites. We call them "Strong-ID features" and "Weak-ID features", respectively. Our filter passes Strong-ID and Weak-ID features to their corresponding classifiers, and performs two classifications in parallel. To make a decision for the website's maliciousness, it considers both classifier decisions. The agreed decision is accepted as it is. For disagreed decisions, our filter compares the prediction powers (confidences) of decisions, and chooses the one with higher prediction power (higher confidence). In other words, two classifiers are given the equal weight.

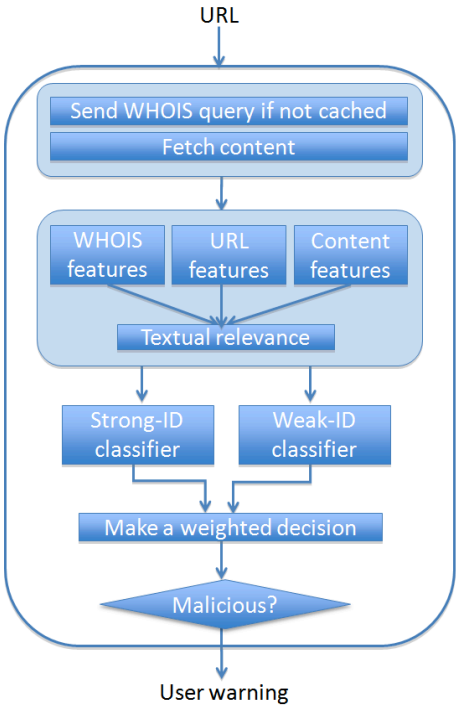


Fig. 6. The overall filtering process

Fig. 6 shows how our filter works. Given a URL, it fetches the website and sends WHOIS query if WHOIS record of its host domain is not cached. Then, it gathers features from WHOIS record, URL, and the fetched content. Based on these features, it builds Common Name and Domain Name candidates, and then computes the string similarity between these candidates and between host domain and these candidates. Using gathered features and string similarity values, it builds Strong-ID and Weak-ID feature sets. Each feature set is passed to its corresponding classifier. The filter's decision is made as described above. If the website is identified as malicious, our filter warns the user.

3.5 User-Interaction for a Final Decision

All the heuristics we mentioned in Section 2 have supported automatic detection, i.e., their whole detection process proceeds without users' intervention. Automatic detection tools are user-friendly and have achieved reasonable detection accuracy. However, they cannot entirely eliminate false positives and false negatives. Moreover, automation may prevent users from browsing benign websites misidentified as malicious and let attackers succeed in their attacks by devising ways to bypass it. To alleviate these problems, our filter does not completely rely on automation, but incorporates user-interaction into learning a website's maliciousness. If it identifies a website as malicious, it warns users that they are at risk of falling for an attack, with the likelihood that this website is benign, between zero and one. The less the probability is, the more likely this website is to be malicious according to our filter. Users will leverage our probability and their knowledge to correctly judge the website in question. Users may ignore our warning and proceed with their browsing if they decide that the website is misidentified as malicious. By interacting with users in the decision process, we have three advantages over automatic detection. First, users are likely to experience fewer false positives, i.e. not being able to browse benign websites. Second, the protection is more robust as it is more difficult for attackers to devise a way to bypass user interaction than automation. Third, as users interact with our filter, their awareness on malicious websites increases, which leads into user education.

4. EXPERIMENTAL RESULTS

To implement our filter, we had to choose classifiers for Strong-ID and Weak-ID features first. We used WEKA (version 3.6.3) [26], which provides implementations of various machine learning algorithms by GUI program and library. Using WEKA library, we chose the classifier combination with the best performance for our feature sets, and implemented our filter. User-interaction was excluded from the experimental results below. According to user skill, false positive rate will be able to reach even zero.

4.1 Data Collection

According to attack types, we collected two datasets: phishing and malware distri-

bution datasets. Table 3 shows the number of URLs in our datasets.

Table 3. Experimental datasets.

	Phishing dataset		Malware distribution dataset	
	benign	phishing	benign	malware distribution
Number of URLs	779	2,008	4,114	1,144

Phishing dataset consists of 2,008 phishing and 779 benign URLs. The source of phishing URLs was PhishTank [28], which is a well-known community where users post phishes and others verify whether they are phish or not. As the main source of benign URLs, we used URLs of legitimate websites impersonated by the collected phishing URLs. Some phishing websites shared the victim or their corresponding content could not be found from the impersonated websites. To make up for these cases, we added some URLs from Alexa top 500 list. Alexa [29] provides high-traffic URLs and has been used as a source of benign URLs in the literature [7, 13, 14].

Malware distribution dataset consists of 1,144 malware distribution and 4,114 benign URLs. Two sources of malicious URLs were MalwareDomainList [30] and MalwareURL [31], which provide up-to-date information about web-related threats such as URL, IP, and the description of exploit. The sources of benign URLs were the global top 5000 and the top 100 sites in each English-speaking country from Alexa.

Note that we did not share benign URLs across datasets. In the case of phishing websites, classifying phishing websites and their legitimate counterparts has been more challenging than classifying phishing and benign websites. Because phishing websites replicate victims' content, content-based heuristics could hardly differentiate between them. In order to illustrate our detection strength, we used the victims of the collected phishing websites as the main source of benign URLs in phishing dataset. On the contrary, drive-by-download websites do not impersonate other entities. So it is important to classify malware distribution websites and various kinds of benign websites. In the case of malware distribution dataset, we collected benign URLs from high-traffic websites.

The size of datasets in Table 3 may look small. Note that our datasets were refined. The URLs from websites reporting web-related threats should be filtered. First, the posted websites may be gone already, because malicious websites do not stay long. Secondly, a specific type of exploit is intensively posted for a short period. Phishing or malware exploit kits serve as the engine for internet-based exploits, i.e., identity thieves and other malware authors purchase exploit kits and deploy them on a number of malicious servers [32]. Therefore, a specific type of exploit tends to burst. From both datasets, we manually removed URLs which were duplicated, offline, or related to bursting exploit. The example of our refining steps is shown in Table 4.

Table 4. Refining process of malware distribution URLs.

	MalwareDomainList	MalwareURL	Total
Total URLs	6,316	127,040	133,356
Unique URLs	4,758	122,995	127,753
Live URLs	1,496	2,133	3,629
URLs with unique exploit	485	659	1,144

4.2 Derivation of Classification Model

Our datasets have 3,152 malicious URLs and 4,893 benign URLs in total. We split them into training and test datasets. Training dataset consists of 874 malicious and 1,104 benign URLs, which was used to choose a suitable classifier combination for Strong-ID and Weak-ID features. Using 57 classifiers from WEKA, we tried 57^2 combinations in total. We chose the classifier combination with the best performance, LADTree [27] as a Strong-ID classifier and ThresholdSelector as a Weak-ID classifier. We also split test dataset according to attack type. Phishing test dataset consists of 1,684 phishing and 400 benign URLs from phishing dataset, and malware distribution test dataset consists of 594 malware distribution and 3,389 benign URLs from malware distribution dataset.

4.3 Feature Comparison

To evaluate the worth of features, we used WEKA's gain ratio method [33], which evaluates the worth of each feature by measuring the gain ratio with respect to the class. The higher the gain ratio of a feature is, the more it can contribute to the correct decision. Table 5 and Table 6 show the top features in each dataset whose gain ratio is greater than or equal to 0.1. In other words, they are the key features highly contributing to the correct decision. In the case of phishing dataset, 7 out of 10 features are related to our disparity measure, except for *Https*, *# of Domain Name candidates in Hostname*, and *Numeric Hostname*. This illustrates that our disparity measure is very effective in identifying phishing websites.

Table 5. Key features of phishing dataset.

Feature	Gain Ratio	Description
Https	0.209	whether a website uses secure connection
Frequent Terms vs. Registrant	0.152	max. textual relevance between the most frequent terms in content and the registrant from WHOIS record
# of Domain Name candidates in Hostname and Content	0.146	the number of Domain Name candidates that appear both in hostname and content
# of Domain Name candidates in Hostname	0.146	the number of Domain Name candidates that appear in hostname
Numeric Hostname	0.146	whether hostname is an IP address
Title vs. Registrant	0.142	textual relevance between the title and the registrant from WHOIS record
Title vs. Domain Name candidates in Hostname	0.133	max. textual relevance between the title and Domain Name candidates in hostname
Frequent Terms vs. Host Domain	0.124	max. textual relevance between the frequent terms in content and host domain
Copyright vs. Registrant	0.113	textual relevance between copyright holder and registrant from WHOIS record
Domain Name candidates in	0.109	max. textual relevance between Domain Name

Anchors vs. Registrant		candidates from anchors and registrant from WHOIS record

Similarly, in the case of malware distribution dataset, 4 out of top 6 features are related to our disparity measure except for *Https* and *Redirection*. Moreover, all six consist of our detection scenario of malware distribution websites. If a website was redirected, uses insecure connection, and the identity disparity exists, it is highly suspicious.

Table 6. Key features of malware distribution dataset.

Feature	Gain Ratio	Description
Https	0.209	whether a website uses secure connection
Redirection	0.156	whether a website was reached through any chain of redirections
Frequent Terms vs. Host Domain	0.114	max. textual relevance between the most frequent terms in content and host domain
Domain Name candidates in Scripts vs. Host Domain	0.109	max. textual relevance between Domain Name candidates from scripts and host domain
% of Anchors with External Location	0.101	the ratio of anchors which specify external location
Title vs. Host Domain	0.095	textual relevance between the title and host domain

4.4 Performance Comparison

To illustrate strength of our filter, we compared it with seven other heuristics for detecting phishing or drive-by-download websites. Chosen heuristics are most recent and have shown outstanding detection accuracies in their fields. Table 7 shows the scope of detection of our filter and the others.

Table 7. The scope of detection.

	Ours	[13]	[14]	[34]	[5]	[6]	[7]	[4]
Phishing websites	O	O	O	O				O
Malware distribution websites	O				O	O	O	O

The first three [13, 14, 34] are phishing detection heuristics, and the three in the middle [5, 6, 7] are heuristics for detecting malware distribution websites. Only ours and *Beyond Blacklists* [4] protect users from both attacks while the others provide the limited scope of protection.

Table 8 shows the external references considered by ours and the others. It would be ideal if heuristics would not reference external information, because sending queries to external servers raises concern about roundtrip delay and privacy. However, referencing external servers has been inevitable for reasonable detection accuracy. There have been some heuristics solely considering content anomalies [5, 6], but they have suffered from low detection accuracy. To deliver higher accuracy, heuristics have increased the number of external references considered [4, 7, 14]. Except privacy concern, the lookup delay caused from multiple external queries would not be short. To accept long delay for increasing accuracy is not appropriate for real-time detection which an add-on filter requires. Among the external references, search engine and DNS lookup are most prob-

lematic as we mentioned in Section 2. Our filter considers only WHOIS records, and it minimizes the number of lookups by local caching.

Table 8. Used external references.

	Search Engines	WHOIS	DNS	Blacklists
Ours		O		
[13]	O	O		
[14]	O			
[34]				O
[5]				
[6]				
[7]		O	O	
[4]		O	O	O

Table 9 and Table 10 demonstrate the detection accuracies of ours and the others. We have two test datasets according to attack type. For each test dataset, TPR stands for true positive rate (hit ratio), the ratio of correctly identified malicious URLs to all malicious URLs in the test dataset. Similarly, FPR stands for false positive rate (false alarm rate), the ratio of wrongly identified benign URLs to all benign URLs. For comparison to other detection heuristics, we summarize the TPR and FPR results from the published articles.

The results in Table 9 show that our heuristics are very effective, especially in phishing websites. In the literature, the most accurate heuristics to identify phishing websites have introduced search engines [13, 14]. Compared to these approaches, ours performed well without severe overhead such as the long roundtrip time to search engines. This is because the disparity between the true and claimed identities is inherent in the exploit mechanism of phishing websites, and our heuristics understand and measure such disparity well.

Table 9. Detection accuracy of phishing dataset.

	Ours	[13]	[14]	[34]
TPR (%)	96.79	97	93.21	95
FPR (%)	1.5	6	2.26	3

Table 10. Detection accuracy of malware distribution dataset.

	Ours	[5]	[6]	[7]	[4]
TPR (%)	93.27	85.31	86.36	99.23	91.21
FPR (%)	9.56	13.70	N/A	9.88	14.83

The results in Table 10 show that heuristics except ours and *Prophiler* [7] suffered from low detection accuracy. Though *Prophiler* [7] performed best, it protects users only from drive-by-download websites and considers multiple external references. Because those references are main contributors to its high detection accuracy, it is not appropriate for real-time detection. While our FPR in Table 10 is obviously higher than ideal, we would like to point out that there are two ways to avoid false positives. First, our filter involves users in validation process. Therefore, FPR will be able to reach even zero according to

user expertise. Secondly, our filter is complementary and can be used in conjunction with other detection mechanisms, especially the ones that are resource intensive but show low false positive rates. For example, the tools deploying dynamic heuristics may run for a time ranging from tens of seconds to even tens of minutes for a single page. This delay is not feasible to use these tools in web browser while user is interacting with websites. Instead, a web browser can use ours as the first filter with no noticeable delay in user experience, and then use the resource-intensive one as necessary.

5. CONCLUSION

In this paper, we propose an interactive website filter for safe web browsing. Our main contribution is that we found a common and efficient characteristic to filter malicious websites: the disparity between a website's true and observed identities. Our filter considers such disparity as the key feature, which is measured by string similarity via q-gram distance metric. Compared to expensive heuristics such as multiple external references and dynamic emulation which prior works have introduced to improve detection accuracy, textual relevance must incur negligible overhead. Experimental results demonstrate that our filter detects malicious websites with considerably high accuracies, especially without noticeable delay in user experience in web browsing.

As future work, we could use our heuristics for website classification, an essential task for web directories and focused crawling [35]. Unlike document classification, website classification has been a difficult task due to the enormous size and unstructured nature of the web [36]. The top-level classification of websites must express their nature in the simplest words possible, i.e., their identities [37], whose candidates we can provide with high accuracy. Another possible future work is to use our heuristics to automatically create a logo database. Thanks to the advance of high-resolution digital cameras and broadband network, getting access to digital information and services via logo recognition is of high interest. The fundamental subsystem for logo recognition is a logo database whose images link the digital identity to the real services world. Websites are very good source of logo images and an image tag embedding a website's logo tends to include the website's entity name in its attribute values such as src, alt and title. Therefore, our identity candidates will be able to greatly help automatically building logo database from websites.

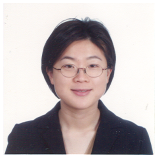
REFERENCES

1. Websense, "2010 Threat Report," *white paper*, 2010, http://www.websense.com/assets/reports/report-websense-2010-threat-report-en.pdf.
2. http://en.wikipedia.org/wiki/Phishing.
3. N. Provos, P. Mavrommatis, M. A. Rajab, and F. Monrose, "All Your iFRAMEs Point to Us," in *Proceedings of 17th USENIX Security Symposium*, 2008, pp. 1-15.
4. J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond Blacklists: Learning to Detect Malicious Web Sites from Suspicious URLs," in *Proceedings of 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 1245-1254.
5. C. Seifert, I. Welch, and P. Komisarczuk, "Identication of Malicious Web Pages

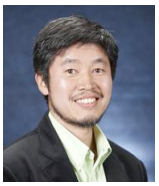
with Static Heuristics," in *Proceedings of Australasian Telecommunication Networks and Applications Conference*, 2008, pp. 91-96.
6. P. Likarish, E. Jung, and I. Jo, "Obfuscated Malicious Javascript Detection using Classification Techniques," in *Proceedings of 4th International Conference on Malicious and Unwanted Software*, 2009, pp. 47-54.
7. D. Canali, M. Cova, G. Vigna, and C. Kruegel, "Prophiler: A Fast Filter for the Large-Scale Detection of Malicious Web Pages," in *Proceedings of 20th International World Wide Web Conference*, 2011, pp. 197-206.
8. https://projects.honeynet.org/capture-hpc.
9. Y. Wang, D. Beck, X. Jiang, and R. Roussev, "Automated Web Patrol with Strider Honey-Monkeys: Finding Web Sites That Exploit Browser Vulnerabilities," in *Proceedings of 13th Annual Symposium on Network and Distributed System Security*, 2006.
10. http://wepawet.iseclab.org/index.php.
11. J. Nazario, "PhoneyC: A Virtual Client Honeypot," in *Proceedings of 2nd USENIX conference on Large-scale Exploits and Emergent Threats*, 2009.
12. M. Cova, C. Kruegel, and G. Vigna, "Detection and Analysis of Drive-by-Download Attacks and Malicious JavaScript Code," in *Proceedings of 19th International World Wide Web Conference*, 2010, pp. 281-290.
13. Y. Zhang, J. Hong, and L. Cranor, "CANTINA: A Content-Based Approach to Detecting Phishing Web Sites," in *Proceedings of 16th International World Wide Web Conference*, 2007, pp. 639-648.
14. G. Xiang and J. Hong, "A Hybrid Phish Detection Approach by Identity Discovery and Keywords Retrieval," in *Proceedings of 18th International World Wide Web Conference*, 2009, pp. 571-580.
15. http://www.ntobjectives.com/files/IsYourWebsiteAlreadyInfected.pdf.
16. http://www.google.com/intl/en/analytics/.
17. http://shaun.net/2011/02/how-third-party-dns-resolvers-can-impact-performance/.
18. R. Dhamija, J. D. Tygar, and M. Hearst, "Why phishing works," in *Proceedings of ACM CHI 2006 Conference on Human Factors in Computing Systems*, 2006, pp. 581-590.
19. R. Dhamija and J. D. Tygar, "The Battle Against Phishing: Dynamic Security Skins," in *Proceedings of 2005 Symposium on Usable Privacy and Security*, 2005, pp. 77-88.
20. http://www.antiphishing.org/reports/APWG_GlobalPhishingSurvey_1H2011.pdf.
21. http://www.spamfighter.com/News-10530-Impersonated-ImageShack-Site-Delivering-Trojan.htm.
22. http://threatpost.com/en_us/blogs/major-ad-networks-found-serving-malicious-ads-121210.
23. S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A Framework for Detection and Measurement of Phishing Attacks," in 2007 ACM workshop on Recurring malcode, 2007, pp. 1-8.
24. E. Ukkonen, "Approximate string-matching with q-grams and maximal matches," *Theoretical Computer Science 92*, Vol. 1, 1992, pp. 191-211.
25. R. Balasubramanian, S. Rajan, R. Doraiswami, and H. Stevenson, "A reliable composite classification strategy," in *Proceedings of IEEE Canadian Conference on*

Electrical and Computer Engineering, 1998, pp. 914-917.
26. http://www.cs.waikato.ac.nz/ml/weka/.
27. Y. Freund and L. Mason, "The alternating decision tree learning algorithm," in *6th International Conference on Machine Learning*, 1999, pp. 124-133.
28. http://www.phishtank.com.
29. http://www.alexa.com
30. http://www.malwaredomainlist.com.
31. http://www.malwareurl.com.
32. http://www.securelist.com/en/analysis/204792056/Drive_by_Downloads_The_Web_Under_Siege#5.
33. R. Wan, I. Takigawa, and H. Mamitsuka, "Applying Gaussian Distribution-Depend- ent Criteria to Decision Trees for High-Dimensional Microarray Data," in *Proceedings of 32nd International Conference on Very Large Data Bases*, 2006, pp. 40-49.
34. P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "PhishNet: Predictive Blacklisting to Detect Phishing Attacks," in *29th Conference on Computer Communications*, 2010, pp. 1-5.
35. J. Rennieyz, and A. K. McCallumz, "Using reinforcement learning to spider the web efficiently," in *16th International Conference on Machine Learning*, 1999, pp. 335-343.
36. J. M. Pierre, "Practical issues for automated categorization of web sites," in *ECDL 2000 Workshop on the Semantic Web*, 2000.
37. P. Luo, F. Lin, Y. Xiong, Y. Zhao, and Z. Shi, "Towards combining web classification and web information extraction: a case study," in *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 1235-1244.

Insoon Jo is a Ph.D. candidate of the School of Computer Science and Engineering of Seoul National University in Korea. She received B.S. and M.S. degrees in Computer Science from the same university. Her research interests are usable security and cloud computing.



Eunjin (EJ) Jung is an assistant professor at University of San Francisco. She received M.S. and Ph.D. in Computer Sciences from the University of Texas at Austin. Her research interests are database privacy, security policy management, and usable security.



Heon Y. Yeom is a professor with the School of Computer Science and Engineering of Seoul National University in Korea. He received M.S. and Ph.D. degrees in Computer Science from Texas A&M University. His research interests are cloud computing and pow-

power-saving systems.