An Information-Sharing Based Anti-Phishing System

Yueqing Cheng, Zhen Yuan, Lei Ma
Zhejiang University
{strongerwill, zju_yz, mlei_zju}@hotmail.com

Robert H. Deng
Singapore Management University
robertdeng@smu.edu.sg

Abstract

This paper presents the design of an information-sharing based or server-assisted anti-phishing system. The system follows a client-server architecture and makes decision based on not only client side heuristics but also collective information from multiple clients. When visiting a web site, a client side proxy, installed as a plug-in to a browser, decides on the legitimacy of the web site based on a combination of white list, black list and heuristics. In case the client side proxy does not have sufficient information to make a clear judgment, it reports the suspicious site to a central server which has access to more complete and up to date information and is in a much better position than individual clients to make informed decisions. Our system is designed to counter against deceptive phishing as well as DNS-hijack attack.

1. Introduction

Phishing attacks have been on the rise during the past several years and are probably the most widespread Internet frauds today [1]. Phishing is the use of deceptive methods, e.g., using emails to lure recipients to a spoof web site, to fraudulently obtain confidential personal information from the web site visitors.

The phishing problem differs from many other security problems in that we wish to protect users from themselves [2]. Phishing attacks not only exploit software vulnerabilities but also human vulnerabilities since the average skilled Internet users often do not understand security indicators and cannot distinguish between legitimate and faked web sites [3].

Many anti-phishing techniques have been proposed in recent years. Most of them use white list (list of known safe sites) and black lists (list of known fraudulent sites) in detecting phishing sites and depend solely on information from trusted third parties to update the lists. In this paper we present the design of a new information-sharing based or server-assisted anti-phishing system. The system employs a typical client-server architecture. When visiting a web site, a client side proxy, installed as a plug-in to a browser, decides on the legitimacy of the web site based on a combination of white list, black list and heuristics. In case the client side proxy does not have sufficient information to make a clear judgment, it reports the suspicious site to a central server which has access to more complete and up to date information and therefore is in a much better position than individual clients to make informed decisions..

Majority of existing anti-phishing systems warn users for potential dangerous operations. However, their high false positives and/or false negatives erode users' trust on the systems; as a result, users normally ignore the warnings and continue their operations. On the other hand, just blocking a suspicious web site is usually unacceptable unless it is absolutely certain that the site is a phishing site. Our system incorporates the design principle in [4] by placing user warnings into the workflow, so that the user has to react to the warning in order to continue his/her working process.

The rest of the paper is structured as follows. We introduce related work in Section 2 and then we present our anti-phishing system in Section 3. We compare the proposed system with the existing schemes in Section 4 and conclude the paper in Section 5.

2. Related work

Many countermeasures against phishing have been developed during the past several years. Here we provide a quick review of some the typical approaches.

Email Filter. Email filtering is the removal of unwanted emails, whether it is an outbreak of a new virus, spam or a phishing attempt on the mail servers before it reaches users' mailboxes. This approach is to stop phishing at the email level (e.g., [10]), since most current phishing attacks use broadcast email (spam) to lure victims to a phishing website. Email filter can prevent a lot of phishing emails. However, phishers are getting smarter. For example, they first get users

0-7695-3016-8/07 $25.00 © 2007 IEEE
DOI 10.1109/ISDPE.2007.65

personal information from public places, like company or university web sites, and then pretend to be the users' friends and send emails to them. Such emails look like from familiar email addresses and will not be filtered out.

Security Toolbars. There are currently a dozen or so toolbars, such as the ones from EarthLink, eBay, TrustWatch, Google and IE 7 [5], which are designed to detect phishing attacks. Most of them are mainly based on white list and black lists [5]. Some also employ heuristics to see if a URL is similar to a well-known URL based on information about the domain name or IP address block where the site is hosted [4, 5]. White and black lists depend on timely reports of phishing sites. As long as a phishing site has not been reported, phishers may steal personal data from visitors to the site. Heuristics such as domain name comparison, URL check, web content analysis, and image check are also adopted to detect fraudulent web sites [4] but heuristics have high false positives and normally need to work with white and black lists.

Web Wallet. The web wallet in [7] distinguishes between input of sensitive data and service usage data by strictly deactivating login forms in the browser. A user has to press a special security key whenever he wants to enter sensitive data. Then he types his data or retrieves his stored data; but before the web form is filled with sensitive user data, the web wallet checks if the current site is good enough to receive the sensitive data. If the current site is suspicious, the web wallet requires the user to explicitly indicate where he wants the data to go. If the user's intended site is not the current site, the web wallet will show a warning to the user, and give him a safe path to his intended site.

A related work is given in [10] which discusses a single-click approach of storing passwords in a wallet that may be cryptographically protected by keys saved on hardware tokens. To defend against malicious content, a browser sandbox model is used where unapproved web objects (e.g., unsigned content) are strictly blocked.

The advantage of web wallet is that it integrates security questions into the user's workflow so that its protection can not be ignored by the user and it reduces the risk of classical phishing attacks. However, the main problem of the web wallet is that its trusted analysis depends on the third parties like TRUSTe, Alexa, CNET. Besides, the web wallet asks users to press the security key first when they submit the sensitive information, which changes user's work flow of navigation. In addition, it does not prevent attacks that fake the user interface of the wallet. Web wallet also does not provide any means to prevent DNS-hijack attacks.

Dynamic Security Skins. This approach is to visually differentiate phishing sites from the spoofed legitimate sites. Dynamic Security Skins [8] proposes to use a randomly generated visual hash to customize the browser window or web form elements to indicate the successfully authenticated sites. It presents two interaction techniques to prevent spoofing. First, its browser extension provides a trusted window in the browser dedicated to username and password entry. It uses a photographic image to create a trusted path between the user and its window to prevent spoofing of the window and of the text entry fields. Second, it allows a remote server to generate a unique abstract image for each user and each transaction. This image creates a "skin" that automatically customizes the browser window or the user interface elements in the content of a remote web page [8]. Dynamic Security Skin requires users to recognize one image and low entropy password, and also have the ability of comparing two images visually. Although it makes phishing attackers harder to spoof customized security indicator, it also gives users a tired visual task. Moreover, it does not solve the attacks of spoofing the trusted window and the visual hashes [8].

Operating System Approaches. The Tahoma browser operating system for web applications is proposed in [12], where web applications run on a secure kernel. Each web site runs in an instance of the web browser which is isolated and restricted to communicate with each other by the security kernel. A virtual machine to counter malware phishing is proposed in [13] where each instance of the application runs in an independent virtual space.

3. The proposed anti-phishing system

In this section we introduce our anti-phishing system and describe both its static structure and dynamic operation. The evaluation of the system will be presented in the next section.

3.1. Architecture

The system follows a client-server architecture. The client side software, called web proxy, is the first line of defense and the origin of information. The web proxy is implemented as a plug-in to web browsers. When a user tries to navigate to a web site, the web proxy makes the first decision whether the site is a spoof site by checking the black and white lists in a database of the client. If the web site is not in the lists, the web proxy performs heuristic analysis on the content of the page and calculates a spoof value. The proxy reports the suspicious site to a central server.

Upon receiving reports from clients the server puts the reported web site in a waiting list. It then dynamically computes an aggregate spoof value of the site based on the report frequency and the client side spoof value. If the aggregate spoof value crosses a threshold, the sever moves the web site into a dangerous list and informs web-proxies to block it. The central server maintains and updates white and black lists based on clients' reports and other sources of information.

In the following, we describe the technical details for both the web proxy and the central server.

3.2. Web proxy

The basic operations of the web proxy are depicted in the flow chart below. There are various methods that attackers can use to produce misleading URLs. For example, a '@' in a URL causes the string to the left to be disregarded, with the string on the right treated as the actual URL for retrieving a web page. Combined with the limited size of the browser address bar, it is easy to write URLs that appear legitimate within the address bar, but actually cause the browser to retrieve a page from an arbitrary site.

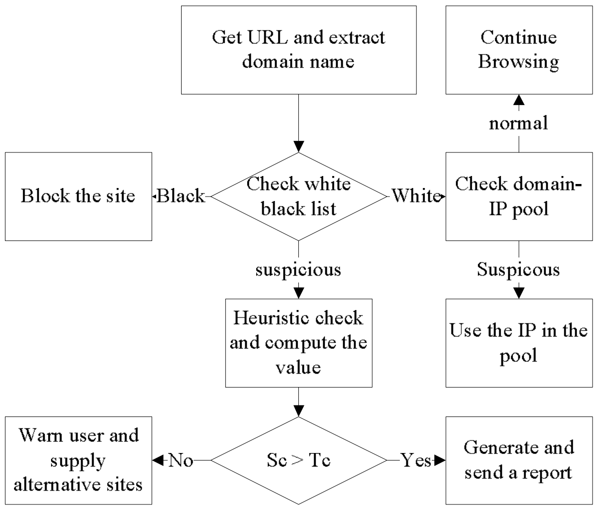

Figure 1. Operational flow chart of the web proxy.

Upon extracting a domain name from the URL, the web proxy checks it against a local black list and a white list. Customized versions of the black and white lists are stored on the client side while the full lists are kept by the central server. The web proxy blocks the web site if the domain name is in the black list. On the other hand, if the domain name is in the white list, the proxy proceeds to check the validity of the corresponding IP address.

The client as well as the central server maintains a database of <domain name, IP addresses> for all domain names in the white list. Every time a user wants to navigate to a web site the proxy checks the IP address of the domain name against the database. If the IP address is not in the database, then there is a possibility of DNS-hijack attack. We force the browser to use the most-recently-used IP address in the database instead of the suspicious IP address.

In the event that the domain name is not in both black and white lists, the web proxy scores the web site using a combination of the URL read, domain similarity check, certificate check, link check, image check and post data check. The scoring results are reflected in spoof value S_C which is calculated using the a standard aggregation function [4].

Domain similarity check finds out how closely the domain of a page resembles a standard or previously visited domain. As in [4], we compare domains by Hamming (edit) Distance using the *Levenshtein algorithm* (http://www.merriampark.com/ld.htm). For example ebay.com will raise the domain check if ebey.com is in the file of commonly spoofed sites or in the user history. During the link check, links contained in a page are examined. The link check fails for a page if at least one-fourth of the links fail the URL check described above.

Spoof sites usually contain images taken from the honest site. For example, the eBay logo may appear on spoofed eBay pages to give the user the impression that they are communicating with eBay. If the eBay logo appears on a login page unrelated to eBay, that page is suspicious. The same applies to other identifiably eBay-specific images such as banners and buttons. We note that corporate logos often legitimately appear on many e-commerce sites (e.g., the Amazon logo appears on sites that sell products through Amazon) and therefore we only count this test for pages that ask for private user input. What if the spoof page contains a slight modification of the real image? The image comparison test might fail to detect the spoof. Fortunately, as noted earlier, attackers often directly copy or link to images on the honest site. Nevertheless, Small image modification can be defended by storing an image hash rather than the actual image [4]. Image hashing refers to a hashing algorithm that produces the same hash for similar images.

The spoof value S_C obtained above is compared with a pre-specified threshold value T_C. If the spoof value exceeds the threshold, the web proxy generates a report including the suspicious URL (and domain name), the spoof value S_C and possibly some other information to the central sever. If the proxy gets the server's response in real time, the response is returned to the user. Otherwise, the proxy generates a warning signal to the user.

Experience shows that most users just ignore warning provided by anti-phishing systems and continue to surf the net. In our design, the system warns its user when it is not able to decide if a suspicious site is a spoof or not. Instead of just giving a simple warning, our system provides alternative web sites and mixes them with the original URL the user tries to navigate. The combined list of web sites is then presented to the user for him to pick his choice. We believe this will be more effective to prevent spoof since the user is forced to identify the web site he wants to visit. When the system is sure that a site is indeed a spoof , it forcefully stops the navigation to the web site, blocks the spoof page, and gives a detailed report about this web site to explain why it is spoof.

3.3. Central server

The main functionality of the central server is to maintain a database of black and white lists of web sites. Phishing sites do not last for a long time, normally from a few hours to a few days; therefore, the database, especially the black list, needs to be updated regularly. The server performs this update of the database through the following means.

First, the central server updates its database of black and white lists based on data from trusted third parties, such as Digital PhishNet and Anti-Phishing Working Group.

Second, legitimate web sites are provided with a trusted channel to report their existences to the central server. The central server then verifies the legitimacy of the reports and updates its database of white list and black list accordingly.

Third, the central server updates its database of black and white lists based on client reports. Upon receiving the first client report on a suspicious web site, the central server puts the reported web site in a waiting list, continues to monitor client reports on the web site and computes an aggregated spoof value of the suspicious web site. The aggregated spoof value, denoted $S_S(t)$, is the moving average of all spoof values reported by clients over an interval $[t - T, t]$, where T is the size of the sliding time window and t is the current time. The aggregated spoof value $S_S(t)$ is compared with a pre-defined threshold T_S. If the value is less than the threshold, the server will continue monitor client reports; once the aggregated spoof value exceeds the threshold, the suspicious web site is moved from the waiting list into a dangerous list. Web sites in the dangerous list will be further checked through other means, such as by checking with other trusted third parties. Until the status of a suspicious web site in the dangerous list is confirmed, the web site is treated as a spoof site and clients are informed to block the web site. The above process is shown in Figure 2. To prevent denial of service attack and spoofing attack, the central server digitally signs its reply to clients. Upon receiving a reply from the server, a client verifies the server's signature using an embedded server public key.

The central server is also designed to counter DNS-hijack attacks. First, the central server finds and verifies IP addresses of a web sites via trust third party services such as trusted domain name servers. Second, the central server's database records not only domain names of legitimate web sites, but also their corresponding IP addresses based on the fact that most of the sites with high reputation use a fixed set of IP addresses.

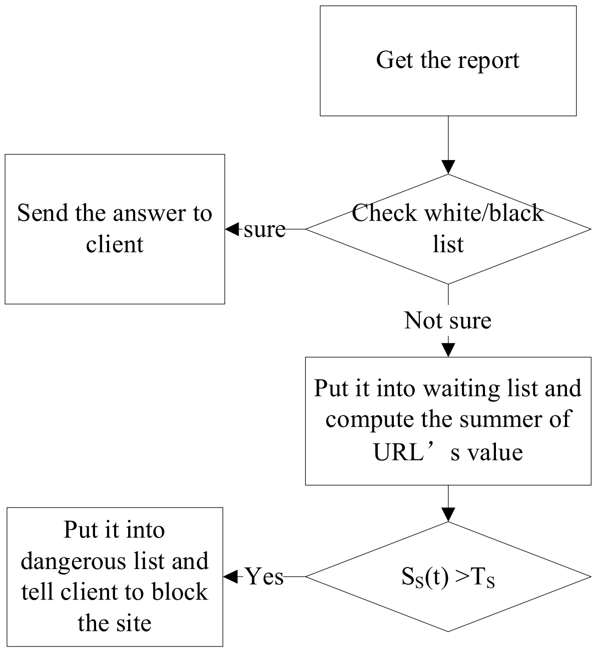
Get the report
Send the answer to client
sure
Check white/black list
Not sure
Put it into waiting list and compute the summer of URL's value
Put it into dangerous list and tell client to block the site
Yes
$S_S(t) > T_S$

Figure 2. Operational flow chart of the central server.

4. Discussion and comparison

There are many anti-phishing tools available to detect phishing attacks. Some are built into browsers, and some are built as add-ons in the form of toolbars. In the first large-scale, comprehensive study comparing leading anti-phishing technologies, 3Sharp LLC tested eight browser-based products to evaluate their overall accuracy in catching 100 live confirmed phishing websites over a six week period (May – July 2006) [14]. The toolbar and browser solutions tested including EarthLink, eBay, GeoTrust, Google Safe Browsing using Firefox, McAfee SiteAdvisor, Microsoft Internet Explorer 7, Netcraft, and Netscape. Results from the study place Microsoft Internet Explorer 7's anti-phishing technology at the top of the list as the most accurate, which is based on its ability to warn users about actual phishing sites while minimizing warning or block errors on legitimate website. Hence, we compare our system with Microsoft Phishing Filter in IE 7 in this section.

The Microsoft Phishing Filter uses a combination of dynamic reputation services from the industry and machine-learning heuristics to help deliver a robust anti-phishing solution for the browser. Microsoft has the advantage of having a very large data set of resources to draw from. On the technology side, they use white list and black list and heuristics on both the client and the server.

The Microsoft Phishing Filter is a two-stage warning system. The first level of warning (yellow) signals to users when the Phishing Filter detects a web site which contains characteristics similar to a phishing site. It recommends the user not to enter his or her personal information on the site. The second level of warning (red) automatically blocks users from a web site if it has been confirmed as a reported phishing site. Microsoft only tells users that a site is suspicious on the first level, but does not provide possible sites which users should go to. In our system, when a user meets a suspicious site, the system provides users a set of legitimate sites which are the most close to the suspicious site. Although Microsoft's Phishing Filter detects a phishing site and blocks it in the second level, it only tells users that it is a "phishing" site. Some users may not know the meaning of "phishing" and therefore simply ignore the warning. In our system, we incorporate warnings in the working flow so that they are not bypassed by users.

Microsoft's IE 7 employs heuristics on the server side and uses human operators to check the site. Microsoft updates its white and black lists mainly based on information from its industry and government collaborators, such as Digital PhishNet; it does not make use of information from its users. In our system, when the client meets a suspicious site, it uses heuristics to compute a spoof value for the site. It then sends the spoof value and the URL to the central server. The central server computes an aggregated spoof value based on reported spoof values by clients and as well as the number of reports over a sliding time window. The suspicious site is blocked if the aggregated spoof value is above a threshold value. We remark that the aggregated value is proportional to the client spoof values and to the number of reports in a given time interval. This approach makes sense since most phishing web sites are short lived, target at reputable web sites and the number of visits to phishing sites are highly bursty in nature.

5. Conclusion

We introduced an information-sharing based anti-phishing system. The system has a client side proxy, installed as a plug-in to browsers, decides on the legitimacy of the web site based on a combination of white list, black list and heuristics. In case the client side proxy does not have sufficient information to make a clear judgment, it reports the suspicious site to a central server which then computes an aggregated spoof value of the suspicious web site and takes appropriate actions accordingly. The algorithms for computing spoof values and setting threshold values for both client proxy and the central server are critical in affecting the efficiency and effectiveness of the system. We are currently implementing the proposed system into a prototype. Our next step is to conduct user experiments to further refine our design.

6. References

[1] A. Emigh, "Online Identity Theft: Phishing Technology, Chokepoints and Countermeasures", *Radix Labs*, October 3, 2005.

[2] D. Florencio and C. Herley, "Analysis and Improvement of Anti-Phishing Schemes", *MSR Tech Report*, Redmond, WA.

[3] J. D. Tygar, R. Dhamija and M. Hearst, "Why Phishing Works", *Proceedings of the Conference on Human Factors in Computing Systems (CHI2006)*, 2006.

[4] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh and J. C. Mitchell, "Client-side defense against web-based identity theft", *Proceedings of 11th Annual Network and Distributed System Security Symposium* (NDSS'04), San Diego, February 2004.

[5] L. Cranor, S. Egelman, J. Hong and Y. Zhang, "Phinding Phish: An Evaluation of Anti-Phishing Toolbars", *CMU-CyLab*, November 2006.

[6] D.A. Norman, "Design rules based on analyses of human error", *CACM*, Vol. 26 No. 4, pp. 254-258, 1983.

[7] M. Wu, R. C. Miller and G. Little, "Web Wallet: Preventing Phishing Attacks by Revealing User Intentions", *ACM SOUPS 2006*, pp. 102–113.

[8] R. Dhamija and J.D. Tygar, "The Battle Against Phishing: Dynamic Security Skins", *ACM SOUPS* 2005.

[9] FDIC, "Putting an End to Account-Hijacking Identity Theft" http://www.fdic.gov/consumers/consumer/idtheft study/identity_theft.pdf, 2004.

[10] A. Herzberg, "Protecting web users from phishing, spoofing and malware", *Cryptology ePrint Archive*, Report 2006/083, 2006, http://eprint.iacr.org/.

[11] D. Florencio and C. Herley, "Stopping Phishing Attacks Even When the Victims Ignore Warnings", *MSR Tech Report*, Redmond, WA.

[12] R. S. Cox, S. D. Gribble, H. M. Levy, and J. G. Hansen, "A Safety-Oriented Platform for Web Applications", *Proceedings of 2006 S&P*, pp. 350–364.

[13] S. T. King, Y. M. Wang, C. Verbowski, H. J. Wang, J. R. Lorch, C. SubVirt, "Implementing malware with virtual machines", University of Michigan Microsoft Research.

[14] P. Robichaux, D. L. Ganger, "Gone Phishing: Evaluating Anti-Phishing Tools for Windows", http://www.3sharp.com/ projects/antiphishing/gone-phishing. pdf, September 2006.