Fighting phishing the information-centric way

Nikos Fotiou, Giannis F. Marias and George C. Polyzos
Mobile Multimedia Laboratory
Athens University of Economics and Business
Athens, Greece
Email: {fotiou,marias,polyzos}@aueb.gr

***Abstract*—Recent research efforts have highlighted the role of information as key target future network architectures. These so-called information centric networks base their operation on the information itself, rather than on other aspects or entities of the architecture, such as location, host, or user identity. In this paper we apply this approach at the application layer of the current Internet in order to mitigate a significant threat: phishing attacks. We designed a mechanism capable of detecting phishing Web pages based on the information that the user perceives while browsing, i.e, the mechanism detects attacks based on the browser's graphical output, rather than on the code of the Web page. We implemented this mechanism in the form of a browser extension and we evaluated it using real data from a publicly available database of phishing URLs.**

Our extension, which is not affected by the location, popularity, or implementation of a Web page, detected 81% of the phishing attempts.

I. Introduction

Many of the security threats that users are currently facing can be rooted to the fact that (inter)networking is agnostic of the information transferred, it only recognizes end points' addresses and bytes traveling through the various mediums. The Internet's networking protocols make a best effort to transfer users' data, no matter if it is a bank transaction, a login form, a denial of service attack or a virus. Nevertheless this weakness of the current internetworking has been identified and various research efforts investigate the possibilities of an information-centric networking architecture; an architecture where information items hold the principal role and location and user identities are of insignificant importance. But can information-centrism be applied in the current Internet in order to solve some of its security problems? In this paper we examine this possibility by applying the principles of information-centric networking in the application layer in order to mitigate phishing attacks.

Examining phishing from an information-centric perspective and by considering a web page's URL as its identity, the problem can be defined as the inability of the architecture to detect that the identity and the content of an information item do not match, i.e., the architecture can not realize that a phishing page contains information regarding an item X but its name is Y. In order to mitigate this inability we developed a mechanism that creates signatures for web pages based on their graphical output–which is exactly what users understand as a page's content. The principal characteristic of these signatures–which we named 'information signatures'– is that two items containing the same–or similar–information have the same–or similar–signatures. Information signatures are lightweight and they reside on the user's side. We have used perceptual hashing to implement information signatures as a Chrome extension which detects phishing pages. Every time a user visits a web page that contains an input field its information signature is generated and stored locally. Phishing pages are then detected by applying a simple rule; a web page that has the same–or similar–information signature with another web page but their URLs are different, is probably a phishing page.

By abiding by the principles of information-centrism our mechanism has a significant advantage; it can not be easily circumvented. The developed mechanism will detect any phishing page that resembles to the original one, without being affected by the technology used to implement it, its popularity or its network location. We evaluated the effectiveness of our mechanism using real data from PhishTank [1] phishing URLs database and it detected 81% of the phishing pages, by having only 1% of false positive probability. What is important, is that all the detected pages were residing on a private web server, which means that its address could not have been blacklisted– therefore anti-phishing mechanisms relying on blacklisting would have failed.

The rest of this paper is organized as follows. In Section 2 we overview some anti-phising solutions and we discuss methods that can be used to circumvent them. Moreover we explain why these methods can not be used against our mechanism. In Section 3 we discuss the implementation details of our mechanism as well as its performance considerations. In Section 4 we evaluate the false positive ratio and the success ratio of our mechanism. Finally in Section 5 we present our conclusions and our future plans.

II. Related work

Over the last years phishing detection has been widely studied. Zhang et al. investigated the effectiveness of 10 anthiphishing tools [2] and found out that their performance is significantly affected by the the source and the freshness of the phishing page URL. Moreover they show that most anti-phishing tools can be bypassed using simple exploits. 9 out 10 of the examined tools used blacklists as their primary defense mechanism.

Blacklists are lists containing suspicious URLs, usually populated by users. Most modern browsers include anti-phishing mechanisms–such as Microsoft's SmartScreen Filter

[3] included in Internet Explorer 9 and Google's Safe Browsing [4] included in Chrome–that also based their operation primarily on blacklists. Blacklists are widely used however they do have drawbacks. There is a critical period of time between a phishing web page appearing in the wild and being reported–and validated–in a blacklist rarely contain phishing pages targeting a limited number of users, e.g., a phishing page targeting the webmail of a small company. Our mechanism is not affected by the popularity of a web page, it only requires that the user has visited at least once the legitimate web page.

In addition to blacklists many anti-phishing tools–such as Netcraft [5] and SpoofGuard [6]–are considering various host-based features, such as Domain name, IP address and WHOIS properties. All these features can be used as complementary mechanisms as they can be easily hidden–using for example CDNs, dynamic DNS or well known hosting services–and they also generate a large number of false positives. Our mechanism bases its operation on the actual content of the page, rather than on its host features.

Predictive algorithms based on URLs are also used as defense mechanisms against phishing attacks. Ma et al. [7] developed a phishing detection solution which mainly bases its operation on the features exposed by a page's URL. Their approach detects phishing pages by examining lexical features of the URLs including the length of the host name, the length of the entire URL, the number of dots and the number of tokens. In order for such a mechanism to be effective, it has to be trained using a set of URLs belonging to phishing pages. However attackers can easily manipulate a phishing page's URL using URL shortening services, internal frames, or the URL rewrite feature that most web servers provide. Our defense mechanism, i.e., information signatures, is not affected by any of the external features of the web page, it solely uses the content that users perceive. Although in our implementation URLs are used to identify web pages, modifying the URL of a phishing page has no impact on the performance of our mechanism, unless the phishing page and the original page have the same URL.

Content-based solutions have also been considered as phishing detection mechanisms. Cantina [8] is a content based solution for detecting phishing web pages using term frequency/inverse document frequency (TF/IDF) algorithm. Cantina's effectiveness depends on its ability to detect the terms with the highest TF/IDF value. During our evaluation process we came across phishing pages that were using a screenshot of the original page with a simple HTML form on top of it. In such cases it would demand complex solutions–such as OCR reading–in order to detect the web page's terms. Finally, various obfuscation techniques can be used to hide a web page's code making hard to detect its terms. Our mechanism is not based on the way a web page has been assembled; it rather uses the graphical output of the web page. Medvet et al. [9] use visual similarity to decrease the the false positives of an existing anti-phish toolbar. Our approach is a stand alone solution, although it can be used for similar purpose. Fu et al. [10] use Earth Mover's Distance to detect phishing web site extracted from emails that contain suspicious keywords. The solution we have developed detects phishing pages whiles user browses them.

III. Implementation

As a proof of concept we implemented an extension to the Chrome web browser that detects phishing web pages based on their information signature. As an information signature we used the perceptual hash of the web pages' screenshots.

A. *Perceptual Hash*

A perceptual hash is a hashing function that produces numerically close outputs for similar inputs, i.e., if $H(x) = y$ is the perceptual hash of x then $H(x') = y$ or close to y if x is similar to x'. In our implementation we used the pHash [11] open source perceptual hash library and we implemented information signatures by applying the Discrete Cosine Transform based hash (DCT) [12], the Marr-Hildreth Operator based hash(MH) [13] and the Radial Variance based hash(RAD) [14] over a web page's screenshot bytes.

As a metric of the similarity of two screenshots we used the normalized hamming distance of their hashes, with 0.0 being the distance of two completely different screenshots and 1.0 being the distance of two absolute the same.

To demonstrate the notion of similarity we use a screenshot of the start page of www.facebook.com. As it can be seen in Figure 1 we have defined 4 areas in this screenshot, the facebook logo, an image showing connected people, the footer of the page and a sign up form. Table 1 shows the values of the similarity metric when the screenshot of the original page is compared with a screenshot of the same page in Chinese and with a screenshot of the same page but with the various areas covered with their background color. Each column of the table corresponds to the algorithm used to generate the hash.

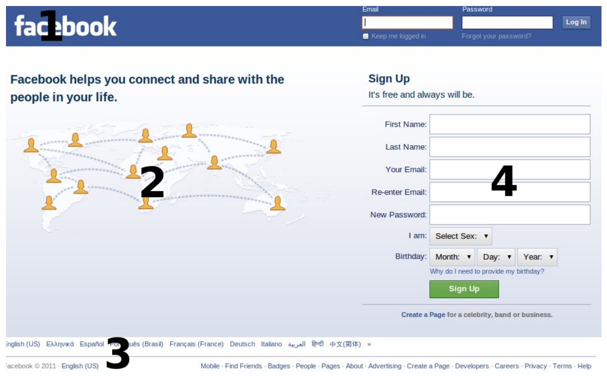
1
facebook
Email
Password
Log In
Keep me logged in
Forgot your password?
Facebook helps you connect and share with the people in your life.
2
Sign Up
It's free and always will be.
First Name:
Last Name:
Your Email:
Re-enter Email:
New Password:
4
I am:
Select Sex:
Birthday:
Month:
Day:
Year:
Why do I need to provide my birthday?
Sign Up
Create a Page for a celebrity, band or business.
3
Facebook © 2011 · English (US)
Mobile · Find Friends · Badges · People · Pages · About · Advertising · Create a Page · Developers · Careers · Privacy · Terms · Help

Fig. 1. Defined areas in facebook.com start page: 1. facebook logo, 2. image with connected people, 3. footer, 4. sign up form

It can be observed that the three algorithms do not have similar behaviour, therefore–as it is shown in evaluation section–a combination of all of them will yield the best result.

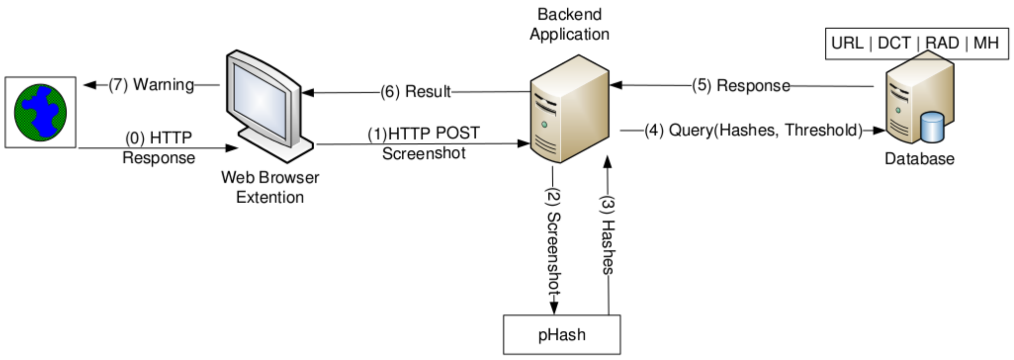

Fig. 2. Implementation overview

TABLE I
SIMILARITY METRIC VALUES WHEN THE ORIGINAL PAGE IS COMPARED WITH SOME VARIATIONS

Difference	DCT	MH	RAD
Original in Chinese	0.23	0.28	0.01
Original without area 1	0.13	0.03	0.01
Original without area 2	0.1	0.11	0.01
Original without area 3	0.1	0.07	0.01
Original without area 4	0.13	0.18	0.01
Original without areas 1 and 2	0.13	0.07	0.01
Original without areas 1 and 3	0.2	0.19	0.02
Original without areas 1 and 4	0.46	0.11	0.01

B. Implementation Overview

Our implementation consists of three parts; a web browser extension, a back-end application and a database. In the current version of the implementation all three components are located in the same machine, but destributed alternatives can also be considered.

The web browser extension is responsible for capturing screenshots of the web page that is currently loaded in the active tab. The back-end application calculates the information signature of the screenshot and the database is the storage point of all information signatures. The system has also one parameter, named the ‘similarity threshold’ that is configured separately for each information signature generation algorithm– DCT, MH and RAD. The similarity threshold denotes the maximum distance that the hashes of two “almost” the same screenshots can have. Two screenshots which hashes have numerical distance bigger than the similarity threshold are considered to belong to different pages.

Figure 2 gives an overview of the implementation architecture. During its lifetime the following steps are executed:

- Step 0: During this step a web page loads in the web browser. When loading finishes, and providing that the web page contains an input field, the browser extension captures a screenshot. The extension takes the screenshot be invoking the *chrome.tabs.captureVisibleTab* method of the chrome.tabs API [15].
- Step 1: In step 1 the extension makes an AJAX call to the back-end application POSTing this way the captured image.
- Steps 2-3: In these steps the back-end application uses the pHash library to calculate three signatures, one with each algorithm.
- Step 4: In step 4 the back-end application searches the database for signatures which distances is less than the specified similarity threshold and belong to web pages with different URL. If at least one record is found steps 5-6-7 are executed. If during this step no record is found then the three signatures and the page’s URL are stored in the database. In the current version of the implementation as key used in the database table that stores the signatures is the quadruplet [URL, DCT hash, MH hash, RAD hash], i.e., for each URL multiple hashes can be stored. The reasoning behind this design choice is that phishing pages may imitate an older version of the victim page.
- Steps 5-6-7: The result of the query is returned and a warning is displayed to the user informing him, that the web page he is currently browsing may be a phishing page.

C. Performance considerations

Information signatures introduce two performance considerations; how fast can they be calculated and what is their size.

Zauner [16] has measured that using a system with Intel Core 2 Duo 2.5 GHz and 4GB of RAM the pHash library calculates the DCT hash with rate 0.33 MB/sec, the MH based hash with rate 0.87 MB/sec and the RAD based hash with rate 2.54 MB/sec. The average size of the screenshot files in our evaluation was 215 KB, so it can be estimated that the signature calculation is achieved in less than 1.5 sec. The DCT based hash is 64 bits long, the MH based hash is 576 bits long and the RAD based hash is 320 bits long. So for each URL 120 bytes have to be stored.

IV. EVALUATION

In this section we evaluate the effectiveness of our implementation using two metrics; its false positive rate and its phishing detection success rate. Our goal is to achieve the maximum possible success rate while keeping the probability of a false positive very low.

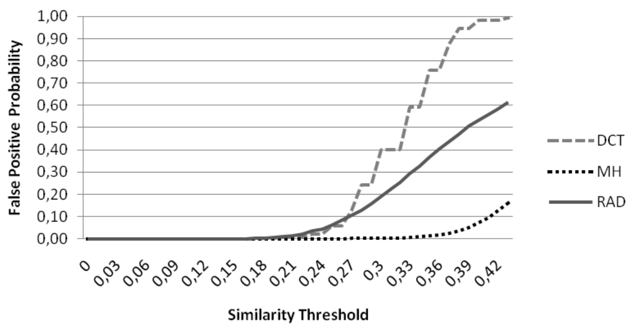

Fig. 3. Probability of false positive for each algorithm with various values of similarity threshold

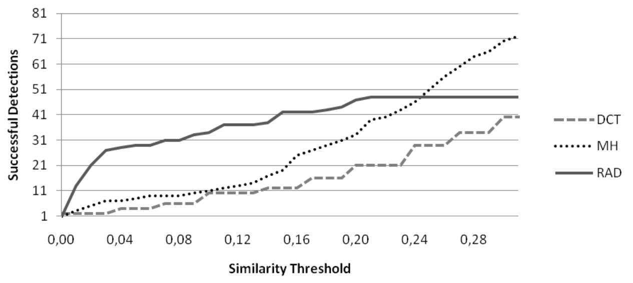

Fig. 4. Performance of each information signature algortihm with varying similarity threshold

A. False Positives

Our implementation may produce in some cases false positive warnings, i.e, it may warn a user that the web page she is currently browsing is a phishing page while it is not. This will happen in case a web page is very similar to another web page which the user has already browsed.

In order to calculate the possibility of a false positive warning we evaluated our implementation using the 100 sites included in Google's top 100 most visited sites in Unites States, belonging to the double click ad planner program [17]. The reasoning for choosing this particular list was that we wanted a sample that reflects the browsing habits of a group of people speaking the same language and having the same cultural background. Our goal was to find the probability for a user to receive a phishing warning while browsing all these 100 sites, using our mechanism with varying similarity thresholds. The result of this evaluation process are presented in Figure 3.

The probability of false positive was calculated by dividing the number of false positives when a particular similarity threshold was used by 4950, which is the total number of possible pairs. For example with similarity threshold equal to 0.23 the total number of false positives of the DCT algorithm was 100, which gives us 2.02% false positive probability. As it can be seen the MH algorithm is less prone to false positives whereas DCT and RAD have similar behavior when low or medium similarity threshold is used, but RAD outperforms DCT for higher similarity thresholds.

B. Detection

The principal functionality of our implementation is the detection of phishing pages.

In order to evaluate the detection capabilities of our mechanism we used real world phishing pages obtained from PhishTank [1] database of phishing URLs, which we manually stored in a separate web server. Our collection methodology was to save the complete phishing web page in a separate web server, providing that the exact same phishing web page was not already saved[1]. In some cases the saved web page was not the exact copy of the original phishing page, as the web browser was failing to save some images, which made harder for our implementation to detect the scam, i.e., in real-time the effectiveness of the extension will be even better. We stored in total 100 phishing pages, that were imitating 56 different web sites[2]

In the second step of this evaluation process we visited the corresponding legitimate pages with our extension enabled. After visiting all the legitimate pages and having recorded their information signatures, we visited the phishing pages. Figure 4 shows the performance of each information signature algorithm with varying similarity threshold. As it is observed the MH algortihm performs better as it was able to detect 71 phishing web pages using similarity threshold $= 0.29$.

However as it was observed in the previous sections the algorithms in use do not behave similarly; there were cases detected by one algorithm but they were not detected by the rest. In order to evaluate the cumulative performance of all three algorithms we configured our implementation with different similarity thresholds; one for each algorithm. The similarity threshold chosen for each algorithm was the threshold for which the algorithm does not surpass a fixed false positive probability. As an example in order to have maximum false positive probability equal to 1% we set the similarity threshold for DCT equal to 0.24, for MH equal to 0.34 and for RAD equal to 0.21. Figure 5 depicts the cumulative success ratio of our implementation. As it can be seen, by having set the similarity threshold for each algorithm so at to have 1% probability of false positive, our implementation detected 81 out of 100 phishing web pages.

It should be noted here that the Chrome web browser ver. 12, the Firefox web browser ver 4.6 and the Netcraft anti-phishing toolbar ver. 1.4.5 did not detect any of the phishing pages.

V. Conclusion and Future Work

In this paper we designed, implemented and evaluated an anti-phishing mechanism, inspired by the principles of information centric networking. Our mechanism bases its operation on the so-called information signatures; signatures that are

[1]Usually the same phishing web page was appearing almost simultaneously under many URLs; in these cases we stored only one of the multiple instances

[2]We considered that the phishing pages imitating the same page but in different language, or different pages of the same web site, were imitating the same site

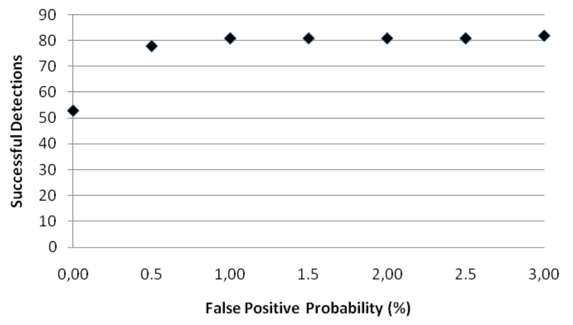

Fig. 5. Cumulative performance of the implementation with varying false positive probabilit

created using the information that end-users actually perceive. The advantage of such a mechanism is that it can not be easily circumvent; any phishing page that is similar enough to the original one, will be detected. Our mechanism works in real time with low false positive ratio. It is not affected by the location of the phishing page, its encoding or the popularity of the victim page. In our evaluation process–which was based on real word data–the implemented mechanism detected 81% of the phishing pages while maintaining the probability of false positive as low as 1%.

The proposed mechanism can be used in many other ways. During the evaluation process it was observed that multiple identical phishing web sites were appearing almost simultaneously under different URLs. So it can be investigated if the creation of a distributed and large scale database of blacklisted information signatures–instead of blacklisted URLs–can be effective. Similarly a whitelist based solution can also be considered, as an example legitimate web sites can include in their digital certificates their information signatures.

Moreover it was observed that the algorithms which were used to generate the perceptual hash of the web sites' screenshot, in some cases, did not have common behavior, i.e., there were cases in which one algorithm was detecting the phishing web page, where others were failing. Therefore it should be investigated what affects each algorithm's performance when it comes to phishing page detection.

The performance of our mechanism was dependent on the similarity threshold we set. This threshold was calculated–as already described in the evaluation section–based on the probability of having a false positive when comparing Google's top 100 most visited sites in Unites States, belonging to the double click ad planner program. Nevertheless user's daily web surfing habits may significantly vary, therefore there can be cases a were a higher threshold can be used–e.g., in a case of a user that visits very few web sites in daily basis, whereas there can be cases where a lower threshold is required. All in all, the performance of our mechanism where the similarity threshold is dynamically set by observation users' behavior is a future area of work.

VI. Acknowledgments

The work reported in this paper was supported by the FP7 ICT project PURSUIT, under contract ICT-2010-257217

References

[1] PhishTank, "PhishTank home page," http://www.phishtank.com, Visited 1 May 2011.

[2] Y. Zhang, S. Egelman, L. Cranor, and J. Hong, "Phinding phish: Evaluating anti-phishing tools," in *Proceedings of the 14th Annual Network and Distributed System Security Symposium (NDSS 2007)*, 2009.

[3] Microsoft, "SmartScreen Filter: frequently asked questions," http://windows.microsoft.com/en-US/windows7/SmartScreen-Filter-frequently-asked-questions-IE9, Visited 10 Jun 2011.

[4] Google, "Safe Browsing API," http://code.google.com/apis/safebrowsing/, Visited 10 Jun 2011.

[5] Netcraft, "Netcraft, netcraft anti-phishing toolbar," http://toolbar.netcraft.com/,Visited 10 Jun 2011.

[6] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. Mitchell, "Client-side defense against web-based identity theft," in *In Proceedings of The 11th Annual Network andDistributed System Security Symposium (NDSS '04*, 2004.

[7] 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 the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '09. New York, NY, USA: ACM, 2009, pp. 1245–1254. [Online]. Available: http://doi.acm.org/10.1145/1557019.1557153

[8] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: a content-based approach to detecting phishing web sites," in *Proceedings of the 16th international conference on World Wide Web*, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 639–648. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242659

[9] E. Medvet, E. Kirda, and C. Kruegel, "Visual-similarity-based phishing detection," in *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, ser. SecureComm '08. New York, NY, USA: ACM, 2008, pp. 22:1–22:6. [Online]. Available: http://doi.acm.org/10.1145/1460877.1460905

[10] A. Fu, L. Wenyin, and X. Deng, "Detecting phishing web pages with visual similarity assessment based on earth mover's distance (emd)," *Dependable and Secure Computing, IEEE Transactions on*, vol. 3, no. 4, pp. 301–311, 2006.

[11] C. Zauner, "pHash, open source perceptual hash library," http://www.phash.org, Visited 10 Jun 2011.

[12] C. Lin and S. Chang, "A robust image authentication method distinguishing JPEG compression from malicious manipulation," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 2, pp. 153–168, 2001.

[13] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 207, no. 1167, p. 187, 1980.

[14] M. Schneider and S. Chang, "A robust content based digital signature for image authentication," in *Image Processing, 1996. Proceedings., International Conference on*, vol. 3. IEEE, 1996, pp. 227–230.

[15] Google, "Tabs-google chrome extensions," http://code.google.com/chrome/extensions/ tabs.html, Visited 10 Jun 2011.

[16] C. Zauner, "Implementation and benchmarking of perceptual image hash functions," Master's thesis, Upper Austria University of Applied Sciences, Hagenberg Campus, 2010.

[17] Google, "The 100 most-visited sites: United States," http://www.google.com/adplanner/static/top100countries/us.html,Visited 1 Mar 2011.