2014 International Conference on Information Technology

TabSol: An efficient framework to defend Tabnabbing

Amandeep Singh
Department of Computer Science and Engineering
Indian Institute of Technology Patna
Patna, India 800013
Email: amandeep0619@gmail.com

Somanath Tripathy
Department of Computer Science and Engineering
Indian Institute of Technology Patna
Patna, India 800013
Email: som@iitp.ac.in

***Abstract*—Phishing attack is one of the most common cyber-attacks causing serious threat to global cyber world and economy. Tabnabbing is a variant of phishing attack evolved recently, in which a malicious page opened in the tab disguises itself to the login page of a popular one, like gmail or facebook, so as to defeat the traditional phish detection mechanisms. In this paper, we propose an efficient security framework called *TabSol* to defend against Tabnabbing attack while detecting the other variants of phishing attacks. We developed and tested *TabSol* in the Google chrome browser and found to be effective. The proposed framework is compared with the existing frameworks and fund to more effective. The most attractive features of *TabSol* includes its simple and fast implementation with less false positive.**

***Keywords*-Web Secuirty, Phishing attack, Tabnabbing attack.**

I. Introduction

Phishing attack is a type of identity theft, in which attackers attempt to acquire the victim's information like username, password, credit card no. etc.. In recent years, attackers could steal billions of dollars from internet users using this attacks. As reported by [1] in 2013, RSA identified on average 37,000 phishing attacks in each month. Also, it was estimated by them that the loss from Phishing is around $1.2 billion in 2012. The primary reason of success of these form of attacks is the incapability of human users in distinguishing the phishing sites from genuine ones.

A traditional Phish attacker crafts e-mails to provoke the users towards the malicious website which looks like the site of a financial institution or reputed company. It prompts the user to reveal its confidential information including credit card number, pasword etc. When the user provides this data to the fraudulent site, phishers exploit the victim's credential to get the financial profit. But, an intelligent user, could suspect and be cautious by looking at the page with login prompt and lose trust on that site.

This work focuses on a new form of Phishing attack called Tabnabbing attack [2]. In Tabnabbing attack the user is attracted to visit to malicious site but looks innocuous. Unlike the traditional Phishing attack, *Tabnabbing* does not prompt for confidential information at the beginning and thus become the user trusted. But, once the user switches to a new tab and opens some other websites, the attacker takes the advantage of the losing user's focus upon the (malicious) Tabnabbing site. During this period, the Tabnabbing page changes its appearances (page title, favicon and content) to look as the login screen of a popular site like gmail.com or facebook.com. When the user swtces back to the opened malicious tab, victim does not suspect the URL and enters her credentials through the login page. Thus the malicious site could obtain the login name and password of the victim. Further attacker can forward the same to the desired site, so vicitin could not understand the occurrence.

In this paper, we propose an efficient framework named *TabSol* to defend against Tabnabbing. This detection mechanism is based on the hash value comparison of the webpage at different instants. TabSol identifies the Tabnabbing phishing webpage by directly discovering the inconsistency of hash values between two states of that webpage. The two states of the webpage are: (i) when webpage is in focus, and (ii) when webpage regains its focus after the focus had been lost. *TabSol* is very simple to implement faster in operation, and on the top it has no false positives.

The rest of the paper is organized as follows. In Section II, we present some literature review regarding various type of phishing attacks. Adversarial assumption along with Tabnabbing attack and its operation are briefly discussed in Section III. The proposed framework *TabSol* is presented in Section IV. The penultimate section discusses the features of the proposed framework and compares with the important existing frameworks. Finally, we conclude the work in Section VI.

II. Related Work

Many frameworks to detect phishing attacks have been proposed are operating on different principles. CANTINA+[4] analyses the features of webpages using machine learning method to classify a webpage as phishing page or legitimate page. CANTINA+ generates less false positive and is able to detect the new attacks. Xaing [5] introduced two blacklist-enhanced content-based algorithms. One is hash based near duplicate page detection and other is Heuristic-constrained k-centroid clustering. Recently, Rafiqul Islam[10] presented a phishing detection mechanism using multi-tier classification model for phishing

978-1-4799-8084-0/14 $31.00 © 2014 IEEE
DOI 10.1109/ICIT.2014.56

IEEE computer society

email filtering based on feature extraction. Unfortunately none of these works has considered Tabnabbing attack.

Several techniques have been proposed that make use of features in HTML DOM to detect phish. In [9], the Iframes (HTML tags) are used by the web developers to embed another document within the current HTML document to detect Phish. However the attacker also could use Iframe tag to build malicious script evolving new types of attacks. Different variants of Tabnabbing can bypass these type of frameworks also.

NoTabNab[6] is an alternative technique to resist Tabnabbing attack. NoTabNab uses a number of open tabs and indicates whether one changes its layout, favicon and/or title to imitate a legitimate site. This mechanism can detect the layout changes and can be used for detecting Tabnabbing. But it would raise a huge false positive, as some times the user resizes its browser and some legitimate web pages are designed to re-layout themselves. Another area, explores the visual and image elements to protect users from phishing attacks. To exploit visual similarity between Web pages, Liu et al.[7] proposed a method using three similarity metrics as block level similarity, layout similarity, and overall style similarity, based upon Web page segmentation. A legitimate webpage owner can use this approach to search the Web for suspicious webpages which are visually similar to the true webpage. A webpage is reported as a phishing, if the visual similarity is higher than its corresponding preset threshold. Large no. of false positive, more running time and high network latency are the bottleneck for phishing detection using this approach. [8] uses techniques to compare the appearance of each tab and allowing the user to distinguish between legitimate changes and malicious masquerading.

III. Assumption and Tabnabbing Attack

A. Adversarial Assumption

It is assumed that the adversary is external. To lure a user, attacker can use some techniques by sending its (malicious) link through e-mail. For this let adversary knows the mail id of the target user. Note that the malicious link sent to attract a target looks genuine, like gaming link for playing, to avoid distrust upon the new link. Further, the web page embeds java script to detect, whether the page is not interacted or waits for specified time. Then the script replaces the title, favicon and imitate a popular web page like facebook.com showing the login prompt.

B. Tabnabbing attack:

Aza Raskin [2], the creative lead for Firefox described a new type of phishing attack and coined the name as Tabnabbing attack, in 2010. Traditionally, a Phish attacker attracts victims through an excited link containing at least one login form which waits for the victim to enter its credential. An intelligent user could be alerted by observing the login page. Unlike the common phishing attacks, Tabnabbing does not show any login form at the beginning to take advantage of user trust. The operational steps of Tabnabbing attack are as follows.

1) A user navigates to a normal looking malicious site like game as showed in Figure 1.
2) When the page has lost its focus and has not been interacted for a while, the *favicon* could be replaced with the Facebook like popular website *favicon* along with the specified similar title as showed in Figure 2. This can all be done by a simple JavaScript.
3) As user scans over many open tabs at a time, the *favicon* and title act as a strong visual. User memory is malleable and moldable, so most likely he thinks as, it was being logged out, and therefore provides its credentials through that fake login.
4) After the user enters the login credentials and sends back to the server, it is redirected to the Facebook. Because he was never logged out in the first place, it will appear as if the login is successful. Thus the attacker can be able to extract the credential of the victim without being identified.

Figure 1: Phishing page at the first open

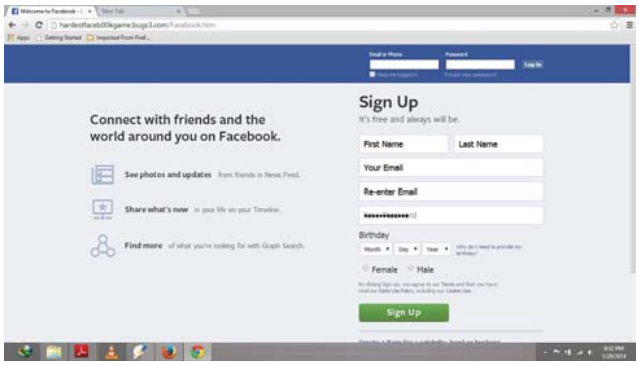

Figure 2: Phishing page after the tab regains focus.

C. Stronger Form of Phishing

The traditional phishing webpage has some login form asking the user credentials to enter into. So, most of

the phishing detection frameworks record the instance of webpage and extract the web page features by information extraction (IE) and information retrieval (IR) techniques and verify the presence of login form. However, such attacks can be strengthened to make the detection a difficult as follows.

- **Attack 1 (Image Based Login Form):** In this form, tabnabber hides the login keywords in images so that any simple login form detection approach cannot detect the login keywords. Login keywords are necessary to identify the presence of login form. Thus, attack can execute without being detected.
- **Attack2(Hidden Login Form):** Alternatively, attacker leaves the use of form tag to avoid the detection of login form.

Such forms could be detected using frameworks like CANTINA+. But, Tabnabbing can bypass these frameworks as it does not contain the login form and other features at the beginning, that are considered by CANTINA+. As an example, let an email contains a link to a malicious (Gamming) website, is opened (accidentally) by victim to play the Game as shown in Figure 1 and, he switches to some other tab after a while. When the user switches back to that phishing tab a login form is appeared asking the user to provide his credentials as shown in Figure 2, where the Gaming website link is replaced with the imitated Facebook login page. This attack uses the fragility of human mind and fake impression that browser tabs are immutable i.e. not susceptible to change.

IV. TabSol: Framework to detect Tabnab

In this section, we present *TabSol* to defend against Tabnabbing and other forms of the phishing attack including its strengthened variants discussed in previous section.

A. *Operation of* TabSol

The operation of TabSol is illustrated in Figure 3 and summarized as follows.

When a webpage is opened in a browser, *TabSol* checks whether the URL (address) is present in the domain white-list or not. A domain white-list is the set of legitimate pages which we know that they are secure and does not contain any phishing pages and therefore no need to execute this process further. Otherwise (if the URL is not found in domain white-list), the page is potential phish candidate so we save the state of the webpage by computing the hash digest of the webpage source. After the user switches to other tab, the phish tab replaces the webpage with some phishing webpage which contain the login form. Thus to detect this, when user switches back to the old tab, TabSol recomputes the hash digest of the page and compares with the stored digest. If they match, the webpage is assured to be legitimate, so the page is appended into the domain white-list. Otherwise (if both the hash values are not consistent) the page is treated as suspicious page. This is because, a little change in the layout of webpage leads to a greater change in the hash value. Finally to conform the phishing page, login form detection is used. Thus the complete framework of TabSol as showed in figure 3, comprises of 2 major phases.

- Phase 1 (*Inconsistent Identification*): Tabnab maintains the state information for each opened tab in the browser, when it is first opened. After sometime, the tab could lose its focus. Again, when the tab regains its focus new state of the tab is determined and compared with the previously stored state. If any inconsistencies arises between these two states, the site is susceptible to be Tabnabbing. The state information is computed as the hash digest of the source of that opened page. To find the hash value of the webpage any cryptographic hash function like SHA-1 algorithm [11] can be used.
- Phase 2 (*Login form Detection*): Presence of login form is a necessary condition for phishing webpage, so a method to detect the presence of login form is mandatory. But, login form detection process is too complex and consumes more time. So, this phase is executed only if inconsistency is being identified in Phase 1. In this phase the loginform detection method presented in CANTINA+ [4]can be used.

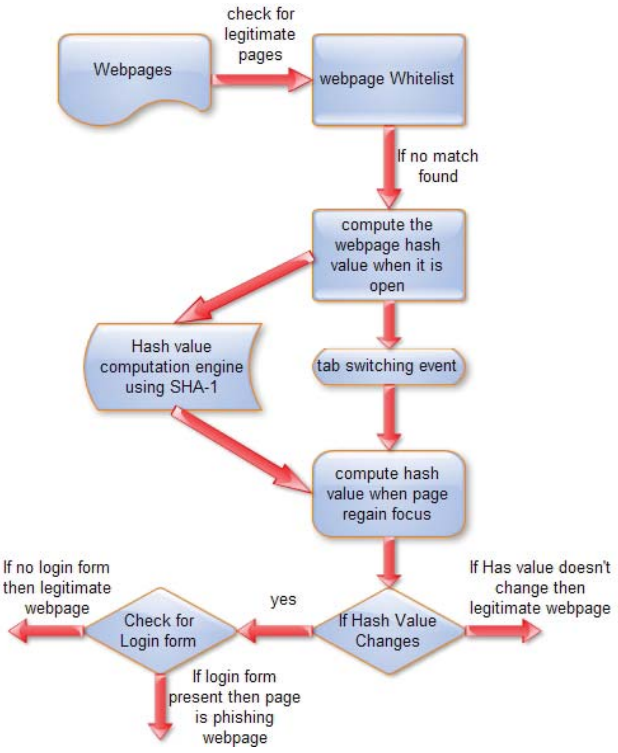

Figure 3: TabSol Framework.

B. *Efficiency of* TabSol

The proposed Phish detection framework comprised with two major phases to take care of the stronger form of

Phishing attack as well as defend against Tabnabbing attack. The *Inconsistency Identification Phase* can detect if the page changes after nabbing the tab. As it uses the standard cryptographic algorithm like SHA1 or SHA256, the probability of collision can be ignored, so limits the false positive rate. Note that the result may be a false negative which is reduced by the Login form Detection phase.

Second phase of *TabSol* is to detect if the Login form exists. As the genuine dynamic webpage does not contain login forms, the alarm would be cancelled.

For the genuine cases the first phase itself does not raise alarm so second phase is not invoked. Second phase is time consuming phase and is invoked only if the page seems to be suspicious in the first phase. Thus the performance is improved.

V. Results and Discussion

In Tabnabbing attack, user is redirected to some malicious webpage without his/her consent. To detect this attack we implement a browser extension for google chrome and evaluate its performance. We took top *Moz* 500 websites from [13] as a test set and evaluate the performance of our browser extension. The test set contains both static and dynamic websites. We evaluate the performance of our browser plugin considering three important parameters: Running time, memory usage and false positive as summarized in Table 1.

A. Running time

We tested the performance of our browser in a 2.53 GHz processor, 4 GB RAM, running on Windows 7 (64-bit) and observed that the *Tabsol* browser extension takes 157ms, to identify the correct website (detect Tabnabbing attack, if exists).

The Concept behind Tab shot is to find the discrepancy between the screenshots of first open webpage and that of the page revisits after a while. Thus, it needs to capture the screenshots and saves it in browser at the first visiting as well as once again the page regains the user focus. If inconsistency arises between the two images then TabShots generates a phishing alarm. The more time consuming of TabShot is attributed to capturing of screenshoots. It is observed that taking two snapshots in browser running on our machine takes around 269ms. Also the authors referred the complete Tabshot consumes 284ms. The increase in average running time of Tabshots is due to capturing the image and comparing. Similarly, NoTabNab[6] stores some features and compares if the changes in particular features occur after nabbing the tab. This comparison is obviously time consuming.

Phishzoo[14] is a framework which only works with webpage having SSL certificates and webpage should look like real sites or represent real sites but do not look like real sites. Phishzoo uses profiles of trusted websites appearances built with fuzzy hashing techniques to detect phishing. The average run time to compare the webpage with 10-60 profiles as referred by author is 4 seconds which is much higher than the running time of Tabsol and Tabshots. Another drawback of phishzoo is that only webpage with valid SSL certificate and valid profile is considered for test cases while, other type of webpages are considered as phishing webpage.

Thus TabSol requires less computational efforts and therefore consumes little time as it uses two hash digest computation for most of the cases, as compared with other frame works. Of course the time consuming part login form detection is used only for conforming if the attack exists to restrict false positives.

B. Memory usage

Memory used by browser extension has huge impact on browser performance. If the browser extension uses too much browser memory, it causes to hang the browser. So memory usage is an important performance parameter. Our framework only needs to save the hash value of webpage which is computed to be very few byes at the maximum, whereas Tabshots[8] store the screenshot of the same webpage twice. For webpage screenshot with resolution 1366x768, the average image size is about 500KB. So each webpage needs 1MB extra space whereas our new browser extension can hardly use few bytes (360/ 512 bytes) only. Phishzoo is another framework for detection of phishing attacks which uses database of webpage profiles, browser need to compare the currently open webpage with these profiles. These profiles are stored in database requires larger space too.

To observe the memory usage we opened different number of webpages through a browser and then compute the average memory usage due to TabSol and other existing security plugins. The memory usage in TabSol is found to be around 2.6MB, while that in NoTabNab is around 5.8MB. The memory usage for Tabshot is much more about 24.5MB.

C. False positives

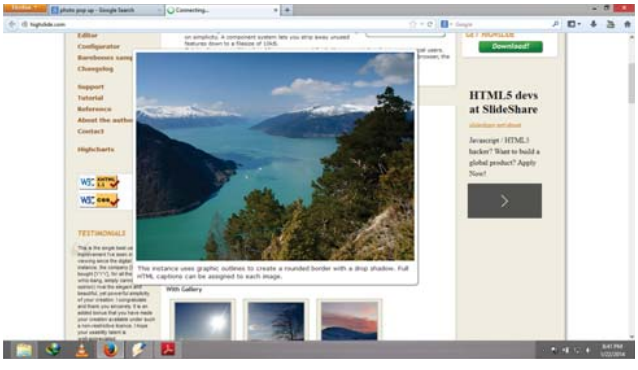

Figure 4: View of a webpage before tab switching event

Table I: Comparison with existing frameworks

	TabSol	**TabShots**	**NoTabNab**	**PhishZoo**
Average Running time	*Low*	*Moderate*	*High*	*MuchHigher*
No. of False positive	Low	High	High	High (webpages without ssl certificates) Low (webpages with ssl certificates)
Detection of Login form	Yes	No	No	No
Robust to new types of attacks	Yes	No	No	No
Browser performance	Less effect	Slow Down the browser	Slow down the browser	Slow down the browser

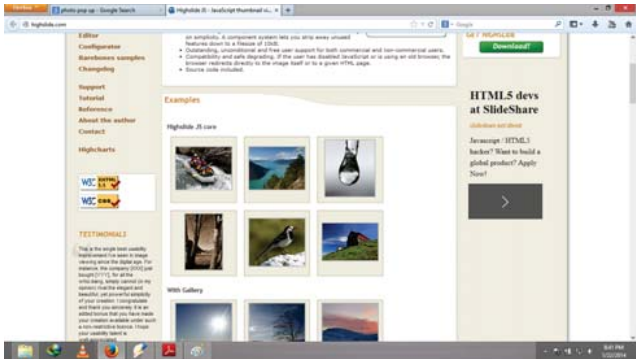

Figure 5: View of the webpage after tab switching

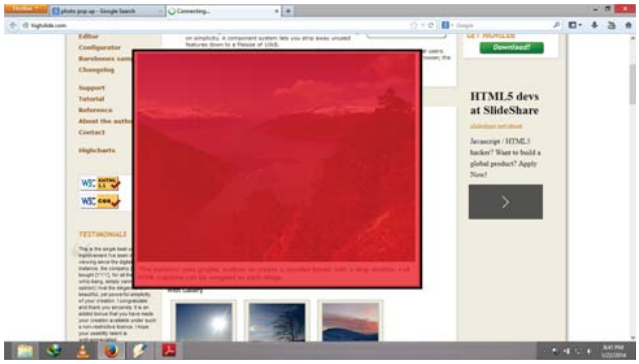

Figure 6: Red area show the difference between two screenshots of same website.

It is observed that the existing frameworks including TabShots and NoTabNab will raise false positive for dynamic sites. Tabshots take two screenshots of the webpage given in Figure 4 & Figure 5, and compute the difference between these two image given in Figure 6 and generate a false alarm. But *TabSol* computes hash value of the source of the page using SHA-1 algorithm. Later on when the user switches back to the said tab, TabSol recomputes the hash value of the page and compares with the stored value. Note that changing a small information lead to great change in hash value. Now, as the page is suspected (for a change in hash value), the login form detection process is invoked to check the presence of login form for Figure 5. In this case, there is no login form so the page is assured to be non malicious and does not raise any alarm. Similarly, the no. of false positive in NoTabNab is also high due to its inability to detect the login form. At the same time, Phishzoo only detects webpages with valid SSL certificates, for webpages containing no certificate could be alarmed.

VI. Conclusion

In this paper, we proposed a new phish detection framework named *TabSol* which defends Tabnabbing attack including other phish detection mechanism. *TabSol* uses collision resistant hash function and login form detection method to detect the Tabnabbing phishing attacks. The attractive features of the proposed framework *TabSol* include less computation, less storage and reduced time requirement to detect. On the top TabSol generates zero false positive. We have developed a browser plug-in to defend Tabnabbing attack, embedded into google chrome browser and tested to deter its efficiency.

References

[1] *http://www.emc.com/collateral/fraud-report/rsa-online-fraud-report-012013.pdf*

[2] Aza Raskin,Tabnabbing: A new type of phishing attack. *http://www.azarask.in/blog/post/a-new-type-of-phishing-attack/*

[3] *http://www.antiphishing.org/*

[4] Guang Xiang, Jason Hong, Carolyn P. Rose, and Lorrie Cranor, *CANTINA+: A Feature-Rich Machine Learning Framework for Detecting Phishing Web Sites*, ACM Transactions on Information and System Security (TISSEC),2011.

[5] Guang Xiang, Bryan A. Pendleton, and Jason Hong, *A hierarchical adaptive probabilistic approach for zero hour phish detection*, In Proceedings of the 15th European Symposium on Research in Computer Security (ESORICS'10), pages 268-285, 2010.

[6] Seckin Anil Unlu, Kemal Bicakci, *NoTabNab: Protection against the "Tabnabbing Attack"*, In eCrime Researchers Summit (eCrime), 2010, pages 1-5, Oct, 2010.

[7] Liu Wenyin, Guanglin Huang, Liu Xiao yue, Zhang Min, Xiao tie Deng, *Detection of Phishing Webpages based on Visual Similarity*, In Dependable and Secure Computing, IEEE Transactions, Volume 3, Issue 4, 2005.

[8] Philippe De Ryck, Nick Nikiforakis, Lieven Desmet, Wouter Joosen, *TabShots: Client-Side Detection of Tabnabbing Attacks*, In Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security, Pages 447-456, 2013.

[9] Rableen Kaur Suri, Deepak Singh Tomar, Divya Rishi Sahu, *An Approach to Perceive Tabnabbing Attack,* , In Internation Journal of Scientific & Technology Research, volume 1, 2012.

[10] Rafiqul Islam, Jemal Abawajy, *A multi-tier phishing detection and filtering approach*, Journal of Network and Computer Applications, Volume 36, 2013.

[11] Guido Bertoni, Joan Daemon, Michael Peeters, Gilles Van Assche, Ronny Van Keer, *http://keccak.noekeon.org/Keccak-implementation-3.2.pdf*, May 29,2012.

[12] Crypto++ 5.6.0 Benchmarks, http://www.cryptopp.com/benchmarks.html

[13] *http://moz.com/top500*

[14] Sadia Afroz and Rachel Greenstadt, *PhishZoo: An Automated Web Phishing Detection Approach Based on Profiling and Fuzzy Matching*, In Proceedings of the Semantic Computing (ICSC), 2011 Fifth IEEE International Conference.