An Agent Based Approach for Anti-Tabnabbing

Sarika S
Department of Computer Science
Bharathiar University, Coimbatore, Tamilnadu, India

Dr.Varghese Paul
Department of Information Technology
Cochin University of Science and Technology, Cochin, Kerala, India

Abstract- Internet security has received critical attention from both academia and industry as it is difficult to provide secure web services due to the emergence of new threats. Expensive security mechanisms can lead to reduced effectiveness and may consume excessive resources. Phishing is an art of luring people to submit their credentials to a counterfeited website masquerading as a benevolent one. Many research works has been conducted in this area, however the threat of phishing is not mitigated as modern types of attack keeps coming to the fore. Tabnabbing is a sophisticated phishing attack which tricks users to submit their personal information and credentials by leveraging the facilities tabs offer to web browsers. Existing anti-phishing techniques are still struggling for the detection and prevention of this dominant phishing attack as the attack takes action in inactive browser tabs. This paper explains the effectiveness of software agents in detecting Tabnabbing attack by making a comparison with an existing method.

Keywords – Distributed software agents, tabnabbing, multi agent system, antiphishing

I. Introduction

Phishing attacks are on the rise and phishers are devising new tricks everyday to confuse the victims. Through the past decades, the number of victims has increased exponentially as phishers change their tactics by exploiting loopholes in software. A number of strategic defense techniques have been introduced both from client side and server side. Server based solutions include anomaly detection[13],attack event monitoring which are implemented by service providers and client-based techniques like blacklist approach[10], whitelist based methods [11],webpage layout similarity [9],visual similarity of webpages [8] those are implemented by users through browser extensions or email client. But a fairly new phishing concept called tabnabbing is likely to deceive even the most security-conscious web surfers as it exploits user's trust and inattention in browser tabs.

The name 'tabnabbing attack'[4] was revealed in early 2010 by Aza Raskin, creative lead of Mozilla Firefox. The attack is simple to implement and silently tracks the victims. A user has a bunch of open tabs and as he navigates through them, phishers set up a counterfeited web site which looks exactly like the legitimate one and load the inactive tabs with the fake page. When the user switches back to the tab, it appears to be a site frequently used by the user and prompts the user to enter his credentials convincing the user that the site is authentic. As the user does not remember how each tab looked like before tab switch, he will give his credentials to the honest looking page and is trapped. There are different ways to perform a tabnabbing attack. The attack can change page layout, title and favicon of the page. In some cases, the fraudster may not change title and favicon and it acts as a strong visual cue to mimic the appearance of a legitimate page. It differs from traditional phishing attacks as the phisher may not even change URL displayed in the browser's navigation toolbar.This paper describes how software agents can be used to perceive phishing attacks like tabnabbing. The approach is a browser extension to google chrome browser that uses three levels of agents in a distributed platform.

The organization of this paper is as follows. Next section describes related work. Section III explains the core idea behind proposed system and section IV shows the system model. Section V describes the operation of proposed system. Implementation and experimental results of our system are presented in section VI. Section VII plots the comaparative analysis of our system with an existing method. Finally, in section VIII we conclude the paper.

II. Related Work

Mozilla has released a Firefox plugin called Account Manager for online identity management. Account Manager lets you store the logins which are already created, suggesting them whenever they can be used. It makes the logins more secure by generating random passwords too. NoScript is a Firefox add-on, for preventing websites from running JavaScript, Java, Flash or other plugins. It provides powerful protection against malicious scripts, XSS, CSR and clickjacking attacks. Another Firefox add-on proposed by Unlu and Bicakci is NoTabNab[1] which guards users from tabnabbing attack. This add-on lookouts open tabs and alert the user about changes in its layout, favicon or title to mimic another page. If an impersonation is happened, the address bar is highlightened in yellow or red according to the warning level. The problems that are indirectly connected to this technique are related to resizing the browser, as only some web pages are designed to re-layout themselves.

The method presented by Suriet al.[2] is also for detecting tabnabbing attack. They use a signature based detection mechanism to deal with tabnabbing attack. The detection mechanism defines a set of rules to scrutinize vulnerable JavaScript code. First the source file is converted into a text file and then into tokens. These values are given to the rule based system which is checked for vulnerabilities. The limitation of this method is that the presence of an iframe is not always necessary for a tabnabbing attack.

Tab-Shots[3] is a browser extension that remembers what each tab looked like, whenever a tab is changed. Tab-Shots record the favicon and screenshots of the presently focused tab at regular time periods. Then the screenshot is separated in to fixed-size tiles. Each tile of the present snapshot is compared to its counterpart in the stored data. If an exact match is not obtained, the non-matching area is marked by a coloured overlay. One probable shortcoming of this technique is the difficulty in detecting small changes in a page.

Current solutions will detect the layout change and warn the user only when the tab is on focus after being nabbed. The method continuously monitors the change in webpage layout in frequent intervals in all the tabs of a browser in parallel and warns the user about the attack at the same time itself. So, the user can be more conscious about the attack and act accordingly.

III. PROPOSED SYSTEM

The proposed method is an agent based approach to detect tabnabbing attack and is currently built as a browser extension to Google chrome browser which is vulnerable to most type of phishing attacks. Objective includes the behavior of Tabnabbing attack in various browsers and the effectiveness of this method in different browsers.

A. Tabnabbing Attack

Tabnabbing[4] is a new kind of phishing attack which asks users to submit their credentials to a malevolent site by masquerading as a genuine one. The Tabnabbing attack triggers when user is busy switching between different tabs of a browser and a certain tab is out of focus and is idle for some time. The different steps of a Tabnabbing attack are:

- The user is visiting a webpage which looks perfectly genuine. The user opens multiple webpages like news, mail account or a social networking site in other tabs of the browser.
- The user changes his tab to another or the user is forced to switch to another tab when the page takes time to load.
- When a tab is unattended for some time and is out of focus, the favicon, title, and layout of the page is replaced with some other site familiar to the user (a frequently used site by the user). In some cases, the title and favicon may not be replaced.
- As the user pays less attention to the URL in address bar, he will give his credentials to the honest looking site and is trapped.

B. Multiagent Based Computing

Phishing attack is a complex phenomenon. An intelligent agent[5] is a powerful tool for solving and handling such issues. Multiagent systems[7] divides the problem into modules which operate asynchronously. This simplification allows use of the best technique to solve problems. Interdependent problems are solved with co-ordinated effort from multiple agents. Every agent uses BDI (belief-desire-intention) architecture and has autonomy, heterogeneity, co-ordination with other agents and its own reasoning.

C. Phishing Detection using Multiagents

Agent based phishing detection is a scientific approach which needs modelling, designing and implementing multiagents in a platform to protect webpages from various attacks. The agents in this system are discrete and well defined within boundaries. Multiple agents deployed in the platform have dynamic strategic behaviour and are purely equipped with a strong prerequisite for using the platform.JADE software framework is used here for the development of agent applications in compliance with the FIPA specifications.

IV.SYSTEM MODEL

The proposed framework consists of four operational agents when a webpage is opened in a browser tab. The agents are hierarchically organized in three levels with the possibility to share and delegate activities and responsibilities. Figure 1 shows the architecture of the system. The agents in this system are:

- T-agent
- U-agent
- M-agent
- I-agent

U-agents and T-agents are level one agents managed by level two agent (M-agent). The interface between the user and the system is provided by level three agent(I-agent).

T-agent or Tabnab agent is a level 1 agent responsible for handling incoming requests from browsers for webpages. The T-agents in each tab performs its delegated task in two phases, Feature Extraction and Feature Comparison for detecting tabnabbing attack in a webpage. When a webpage is loaded, T-agent extracts its five tuple information(text, image, url ,title, favicon) and record for the next phase. The procedure is continued every 60seconds The values from subsequent feature extractions are matched with the recorded values to obtain a resemblance score for each pair of elements. If the resemblance score is higher than a threshold t, the currently visited web page is considered as similar to the recorded one.

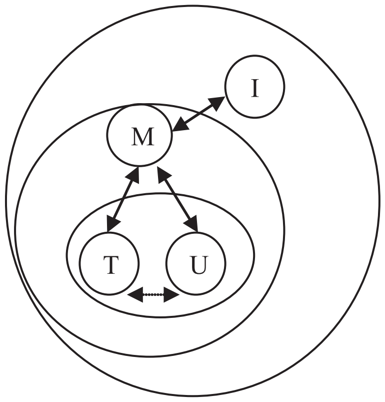

Figure. 1. Architecture of the system

U-agents perform its action when the URL of the webpage is changed after a tab switch event or inert tab. The technique uses a URL blacklist to find fraudulant URLs. Blacklisting works on the basis of a pre-compiled list of URLs which are found to be malicious at some point of time. The U-agent queries the URL blacklist to determine whether the currently visited URL is on this list. If the URL is included in the black list, the user is advised of the danger.

M-agent is purely a CBR agent act as a manager who is responsible for coordination, communication, decision-making and its evaluation. This agent evaluates the different decisions arrived at for the process operation and takes necessary actions immediately.

I-agent is an interface agent which deals with the interaction of the user with the system. The interface agent is having responsiveness, competence, and accessibility communicates to the user about the detection of attack and displays proper message. They are rule based applications and can act autonomously to perform operations without explicit directions from the user.

V. OPERATION OF PROPOSED SYSTEM

A. Feature Extraction

Feature extraction proceeds with extraction of text and images. Text extraction is done by using SAX parser[12]. SAX parser can be used as an effective mechanism to parse the webpages. SAX processing is based on an event-driven processing model. The data elements are deduced on a sequential basis and callbacks are implemented based on certain constructs. One of the biggest advantages of SAX is that it does not load any XML documents into memory. Therefore it is considered to be lightweight and fast. SAX processing model is very simple and it requires implementing a class that extends the DefaultHandler which contain the implementation code in callback methods.

A parsing handler class (extended from DefaultHandler) is required for reading and writing XML documents with SAX. This handler class provides logic coded in the callback methods defined. The parser then processes the input stream and invokes the handler's callback methods to perform the actual work. Figure 2 shows SAX processing model.

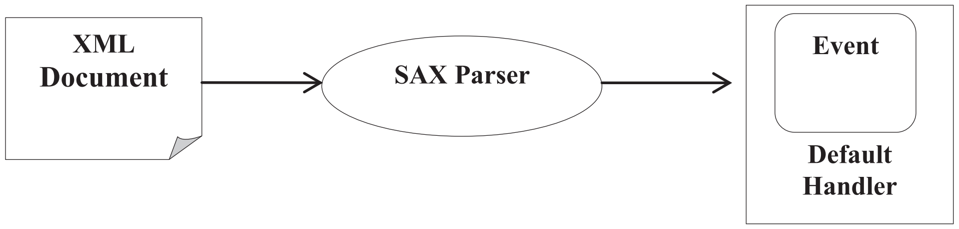

Figure. 2. SAX Processing Model

Image extraction is done by obtaining the source address of the image src attribute, the space occupied by the image in pixel and its position in webpage, and its RGB color histograms. The source address of the image can be obtained from the SAX parser output. The position of the image in webpage is obtained by finding the pixel positions.

B.Feature Comparison

Feature Comparison is accomplished by comparing matching elements separately. Textual contents are compared with the resultant file of SAX parser and find the resemblance score in a matrix as Rt. Then, compare all image pairs to obtain a resemblance score Ri.

For each image, comparison is performed as follows: a)comparison of source address of the image src attribute b) comparison of RGB color histogram c) comparison of pixel positions that image occupy .URL is matched with the stored URL value to obtain a resemblance score Ru. Favicons are compared by source to get a resemblance score Rf. Title of the webpage is matched with the stored value to obtain a resemblance score Rti. Finally, the overall appearances of the two pages are calculated using the above mentioned 5-tuple as $R=Rt+Ri+Ru+Rf+Rti$. If the resemblance score is greater than a threshold t, two pages are similar. Otherwise, pages are dissimilar.

VI. IMPLEMENTATION

The implementation of the method uses JADE[6] software framework to deploy agents in browser to detect URL based attack, hyperlink based attack and tabnabbing attack. The distributed multiagents communicate via FIPA ACL[14].This browser extension is installed in Google Chrome browser and initiated when browser starts functioning.

A.Dataset

The data set consists of a set of common webpages. For finding blacklisted URL's, a collection of real phishing sites from www.phishtank.com are taken. The experiment is conducted in selected 1000 webpages with login forms such as banking sites, web mail clients, credit cards, social networking sites etc. This is because the Tabnabbing attack targets webpages which can provide sensitive information of users. The sources of dataset are shown in Table 1.

Table -1 Sources of dataset

Sources	Percentage
Banks	**53%**
Alexa	**27%**
DMOZ	**20%**

B.Analysis and Results

The effectiveness of the proposed method is assessed using the following parameters.

- True positive (TP) – Legitimate websites detected as legitimate.
- True negative (TN) – Phishing websites detected as it is.
- False positive (FP) – Legitimate websites detected as phishing sites.
- False negative (FN) – Phishing websites detected as legitimate.

False detections were mainly from websites with more animated content. In all other cases, the method shows an impressive response time for accurate detection. Figure 3 shows our empirical result. The parameters taken for evaluation are false positives and false negatives among the number of tested websites.

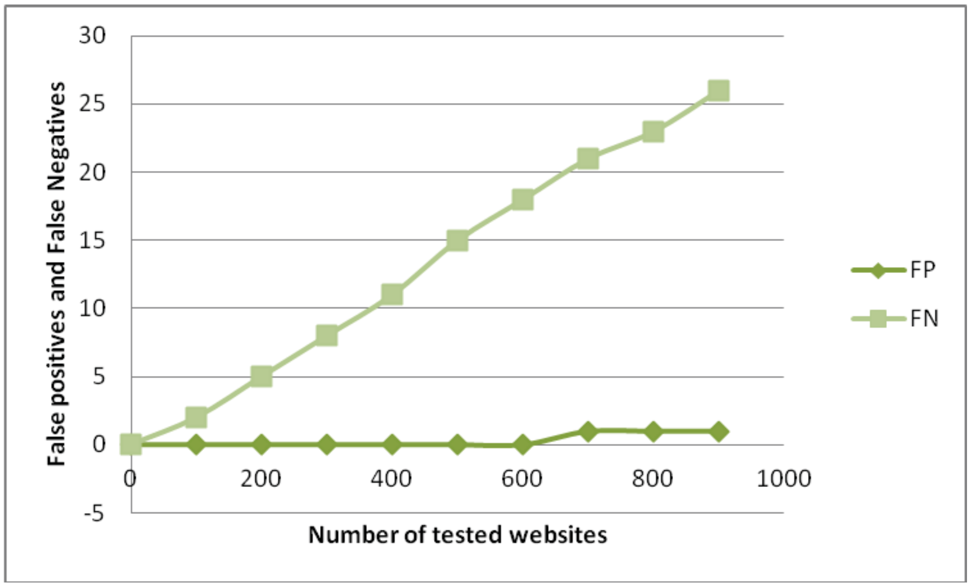

Figure. 3. No. of False Positives and False Negatives

VII. COMAPARATIVE ANALYSIS

In this section, a comparison study of the proposed approach against an existing method is plotted. The comparison is done based on accuracy of attack recognition. Percentage accuracy of proposed method can be calculated as:*(TP+TN)/Number of tested sites.* Figure 4 shows the comparison analysis of the proposed method with an existing anti tabnabbing method TabShots [3] which gives an accuracy of 78%. Our results give 91% accuracy to the proposed method. This implies that it can accurately detect about 91% of Tabnabbing attacks while misclassifying 9% of legitimate websites and thus outperforms the existing method.

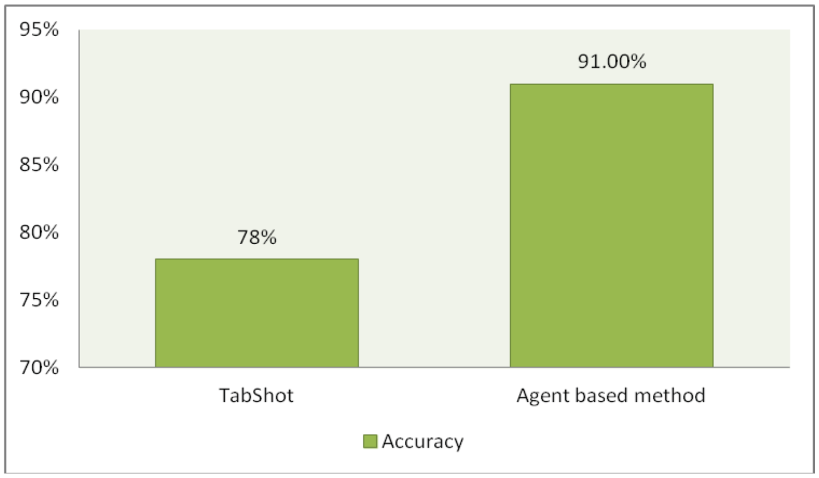

Figure 4.Comparative analysis

VIII.CONCLUSION

Tabnabbing is a Phishing Attack, which happens when a scammer tricks a user into giving away information about account details such as username and password by nabbing a tab. It is not unusual to have half a dozen or more tabs open at once and this wicked deception uses sites the users habitually visit. This paper, discusses about a distributed agent-based architecture that fights tabnabbing attacks. While preventing tabnabbing attack, it can also detect URL based attack and hyperlink based attack. Therefore, it can act as a three factor security framework. The autonomous agents in this system act automatically when an attack scenario happens and performs the action which is delegated to it. The cooperation between the agents helps in reaching a consensus about the attack. The experimental evaluation shows that this is a feasible approach to ensure online security and resistance from tabnabbing attack. The future work includes further development of this framework to battle common types of phishing attacks.

REFERENCES

[1] Seckin Anil Unlu, Kemal Bicakci." NoTabNab: Protection Against The Tabnabbing Attack", *IEEE*. 2010.

[2] R. K. Suri, D. S. Tomar, D. R. Sahu. "An Approach to Perceive Tabnabbing Attack". *International Journal of Scientific and Technology Research*.pp.90-94,2012

[3] De Ryck, Nick, L Desmet, Joosen. "TabShots: Client- Side Detection of Tabnabbing Attacks". Proceedings *of the 8th ACM SIGSAC Symposium on Information*. 2013.

[4] A. Raskin. Tabnabbing: A new type of phishing attack.2010, http://www.azarask.in/blog/post/a-new-type-of-phishing-attack/.

[5] Bradshaw. Software Agents. MIT Press. Cambridge: MA USA,2002.

[6] JADE (Java Agent DEvelopment Framework), http://jade.tilab.com/index.html

[7] Sycara. Multiagent Systems, American Association for Artificial Intelligence. 1998.

[8] Eric Medvet, Engin Kirda, Christopher Kruegel, "Visual similarity-based phishing detection", *in IEEE Symposium Computational intelligence in cyber security*,30-36,2009.

[9] A. Rosiello, E. Kirda, C. Kruegel, F. Ferrandi. "A Layout-Similarity-Based Approach for Detecting Phishing Pages", *IEEE International Conference on Security and Privacy in Communication Networks (SecureComm)*, 2007.

[10] Pawan Prakash, Manish Kumar, Ramana Rao Kompella, and Minaxi Gupta."PhishNet: Predictive Blacklisting to Detect Phishing Attacks". *In Proceedings of the 29th Conference on Information Communications, INFOCOM'10, IEEE Press, Piscataway, NJ, USA*,346-350,2010.

[11] A. Belabed, E. A□meur, and A. Chikh. "A Personalized Whitelist Approach for Phishing Webpage Detection". *In Proceedings of the 2012 Seventh International Conference on Availability, Reliability and Security, ARES '12, . IEEE Computer Society*, 249-254,2012.

[12] Jim Whitehead, SAX Parsing, Spring 2006.https://classes.soe.ucsc.edu/cmps183/ Spring06/lectures/sax-parsing.pdf

[13] Pan, Ying, and Xuhua Ding. "Anomaly based web phishing page detection."*Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual. IEEE,* 2006.

[14] Bellifemine, Rimassa, G Poggi. JADE - A FIPA-compliant Agent Framework. *Proceedings of the 4th International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agents.* London , 1999.