An Approach to Perceive Tabnabbing Attack

Rableen Kaur Suri, Deepak Singh Tomar, Divya Rishi Sahu

Abstract The growth of Internet has many pros and cons to mankind, which is easily visible in day to day activities. The growth of Internet has also manifested into the other domain of cyber crimes. Phishing, web defacement, money laundering, tax evasion, etc. are some of the examples of cyber crimes that have been reported in literature. It has become vital to make technology reliable, so that it can record and intelligently apprise the user of illegal activity. The objective of this work is twofold. Firstly, in this paper tabnabbing which is a type of phishing attack is explored by developing an attack scenario. Secondly, the signature based detection mechanism is proposed to handle tabnabbing attack.

Keywords Iframe, Phishing, Tabnabbing, Web security

1 Introduction

In recent years, incidents of email fraud have created consequential problems for consumers and businesses.There are a lot of emails designed now a days to provoke the consumer to a web site, impersonated to look like the site of the financial institution or other company. It prompts the consumer to reveal sensitive information, such as account numbers and passwords, on the assumption that this information is required to deal with a supposedly urgent account problem. When a deceived consumer provides this data to the fraudulent website, phishers exploit the consumer's account; use to make a financial profit. This is termed as phishing. These attacks use integration of social engineering and deceiving techniques to allure users into giving away confidential information. Phishing attacks are evolving swiftly day by day. The Anti Phishing Work Group detected a total of 48,410 unique phishing sites in December, 2011[2]. A great many of users use cell phones to inquire their bank account balances. Since Smartphone and tablets have taken over corporate use now a days , this could be an increasingly provocative attack vector in coming years. In this paper a new type of phishing attack, "tabnabbing attack" is explored. The attack is distinctive from primitive phishing attacks, while they deceit users with a similar URL or circumvent them by imitating content of original site, this attack uses fragility of human mind and fake impression that browser tabs are immutable i.e., not susceptible to change. The paper is structured as follows: The next section describes the challenges. Section 3 explores Tabnabbing attack with attack scenario and sequence diagram. Section 4 discusses detection technique. Section 5 presents future work and concludes the paper.

- Rableen Kaur Suri, Department of CSE, MANIT, Bhopal, rableenkaur11@gmail.com
- Deepak Singh Tomar, Department of CSE, MANIT, Bhopal, deepaktomar@manit.ac.in
- Divya Rishi Sahu, Department of CSE, MANIT, Bhopal, divyarishi.sahu@manit.ac.in

2 Challenges

There has always been a hassle between web security and evolving tactics. Today's generation hackers seem to be boosting the risk to even higher levels. The witty hackers are using inter – blended, multi honed attacks. The attacks are more focused and use more refined social engineering techniques to trick even skillful and talented users into making a mistake, for example Tabnabbing attack which uses human factor to deceive. These attacks use JavaScript code which is embedded in HTML code. JavaScript is one of the most simple, versatile and effective languages used to extend functionality in websites, however it also possesses some negative effects that reflect on implementation JavaScript. It can also be used to exploit the user's system. Now a days , sophisticated attacks use personal information about the target to alter the phishing email message, making it harder to ignore, moreover making it appear legitimate. For example, knowing the name of the bank a person uses , a hacker can send an e-mail claiming, "Your account number is required for bank database updates ", the victim cannot escape opening it and becomes a prey.

3. Tabnabbing Attack at Client Side

Its a new method of attack, that can be used for phishing, revealed in early 2010 by Aza Raskin, Creative Lead of Firefox, exploiting the weakest element in the Humans. It is also called as tabjacking or tabnapping which means **Tab** + Kid**napping** It exploits users to submit their credentials and passwords to popular websites by forging those sites and convincing the user that the site is authentic. An inactive browser tab is replaced with a rogue page designed particularly to capture user's personal data without user even realizing it has happened It targets internet users who open multiple tabs on their browser simultaneously. This attack uses fragility of human mind and fake impression that browser tabs are immutable i.e., not susceptible to change. The attack works by using a client side script to detect when the user is not viewing the page i.e. it continuously checks user's focus by using mouse and keyboard events, then changes the page content to a phishing page. It changes -the favicon , title and content of the page. Figure 1 shows how a hacker exploits Tabnabbing attack. This attack has been performed by using iframes Iframes[5] allows one to load separate html files into an existing document. Iframes can be placed anywhere in the document flow .In this attack iframes are generated dynamically.

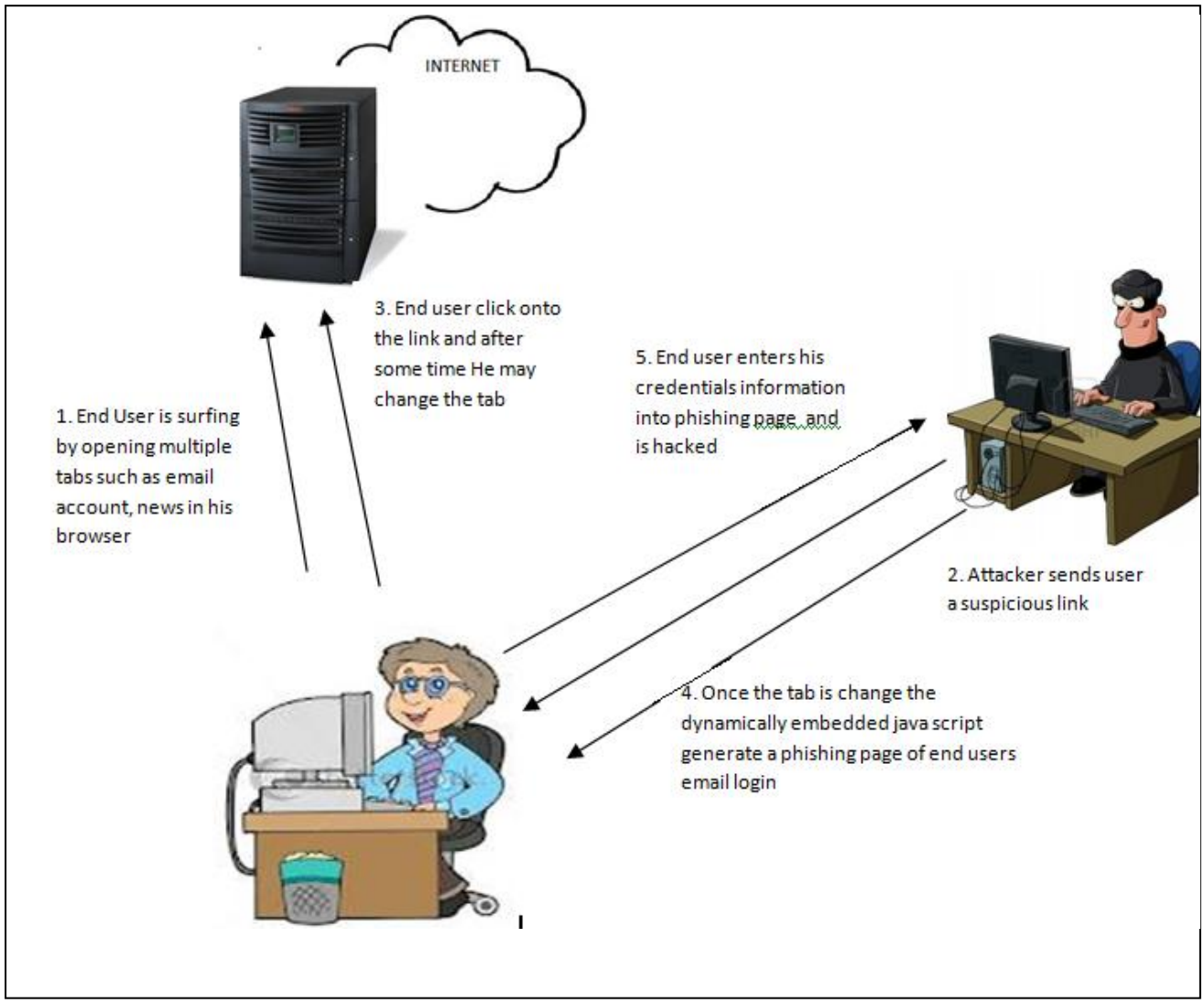

Figure 1 Steps for exploiting Tabnabbing Attack

The following code dynamically generates an iframe element, adding it to the end of the document:

```
Var el = document.createElement ("iframe");

el.setAttribute ('id', 'ifrm');

document.body.appendChild (el);

el.setAttribute ('src', 'http: //www.demo.com');
```

Following are the elements and events used in here with their description and syntax [7] Onfocus - JavaScript focus event occurs when an object or element gets focus. Its syntax is <Object>.onfocus="Execute JavaScriptCode". Onblur - JavaScript blur event occurs when an object or element loses focus. In other words, whenever a person first clicks an element, and then clicks anywhere outside of it. Its syntax is <Object>.onblur="Execute JavaScriptCode" Settimeout - Allows code to be executed or triggered after a particular time. It takes two parameters – the function or code to call and the number of milliseconds to wait before executing it. Its syntax is Setimeout ("function to be called", "time after which it is called"). Onmouseover - This event occurs when a user moves the mouse pointer over an element. Its syntax is <Object>.onmouseover="Execute JavaScriptCode" Onmouseout - This event occurs when a user moves the mouse out of an element. Its syntax is <Object>.onmouseout="Execute JavaScriptCode".

3.1 Attack Scenario

An attack scenario enlists and depicts the ways an attacker might make use of vulnerability. It describes a possible attacker, vulnerabilities, possible attack, the resources affected and the related events. Figure 2 shows the sequence diagram of attack. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

Possible Attacker- An attacker who misleads a user to click on a link, via having them view an email, or having them view another site under attacker control.

Vulnerability- Human memory weakness and false perception that browser tabs cannot change.

Related events - User login event and surfing event

Resource Effected - Loss of victim's personal or private information, loss of money, gaining access to sensitive data.

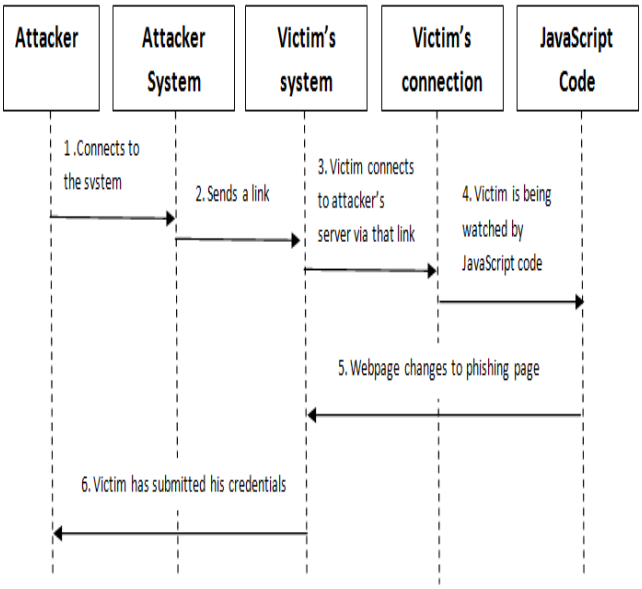

Figure 2 Showing the sequence diagram of the attack

Attack Scenario

1. End User has a bunch of open tabs in his web browser, an e-mail page, news webpage, bank account, social community and maybe more.
2. User opens his mails and finds a link. He browses the link. The opened website seems like original website and can easily navigate further.
3. While user is busy reading on some other tab, the attack is able to hone in on tabs that aren't in focus and replace the title, favicon and the title of the tab
4. When user click on that tab, a rogue page is loaded in its place, to look like a original login page

```
iframeid.onblur = function ()
{
TIMER        =        setTimeout
(loadboguspage, 3000);
}

iframeid.onfocus= function ()
{
If (TIMER) clearTimeout (TIMER);
        iframeid.focus;}
```

5 .Because user has already opened it before legitimately, it don't bother paying any attention to the URL in the address bar and enter his login personal information
6 .User has just sent his information to a vicious third party.

The code given checks for the inactivity of a tab for a predetermined time . If the user returns their focus to the tab containing the malicious script, the timer resets. It changes a legitimate site behind user's back leaving him unaware. It replaces an inactive browser tab with a fake page set up specifically to obtain user's personal data – without user even realizing it has happened. For the purpose of saving the space screenshots have not been included.

3.2 Variants of the Attack

Originally, tabnabbing requires scripting support which is already enabled by default on every modern browser. JavaScript can be used to observe mouse and keyboard events to detect user activity on a page and to change the favicon [no], title and content of the page. Alternatively, researcher Aviv Raff made a proof-of-concept demo of this attack which does not require scripting support and uses HTML refresh Meta tag [9]. So the legitimate page refreshes itself in the background and then it changes into a phishing page. This attack can be improved and can be made more devastating by using several other existing techniques known.

4 Detection Technique

4.1 Current Solutions

A Firefox add-on called No Script obstructs HTML objects, scripts and similar functionality on a domain basis. This ensures protection of users from malicious scripts, XSS, CSRF (cross-site request forgery) and clickjacking attacks. There is an enhanced version for HTML- tabnabbing attack particularly which prevent tabs from refreshing themselves in the background [1]. There is a solution given by Aza Raskin which is the planned Firefox account manager. With the help of this manager, users' account information can be managed by browser. The firefox manager will provide login details after once recorded and when users login information doesn't appear, they may notice it as a rogue page. It can be combined with the password manager .Theoretically, browser's password manager helps user to protect against this attack. User will notice a fraud page if a login page is not filled automatically and willingly look at the URL There is one more Firefox add-on [1] which protects users against this attack. This add-on keenly watches the open tabs to indicate whether a tab changes its layout, favicon and/or title to become like another site. It tracks web pages to calculate level of change in layout and warns user that their tab has been nabbed. Problems that are not related directly to this technique must be handled correctly, for example if the user resizes her browser, some web pages are designed to re-layout themselves and some are not.

4.2 Proposed Work

Iframes are used by the web developers to embed another document within the current HTML document.However the attacker also used Iframe tag to built malicious script. In the developed tabnabbing attack the dynamically iframe tag is combined with onfocus and onblur event of JavaScript. During the research it is found that the dynamically iframe never be used with onfocus and onblur event of JavaScript. Based on this concept signature based system is generated. In the detection technique shown in figure 4, initially the source file is converted into text file. Thereafter it is converted into tokens since for this technique data between script tags is required. Then these values are given to the rule based system wherein it checks for the vulnerabilities. State diagram of attack is shown in Figure 3. It is used to describe the behaviour of the attack. It require finite number of states**,** here there are 3 states -initial in which user open the link and the other two safe and unsafe state

States

q_0 - Open links
F - Safe state
F - Unsafe state (opens phishing page)
Q - {q_0 , safe, unsafe}

Events

E1 Surfs the page and closes the tab
E2 Surfs the page and changes the tab
E3 Returns back to the page within the time Specified
E4 Returns back to the page after the time Specified

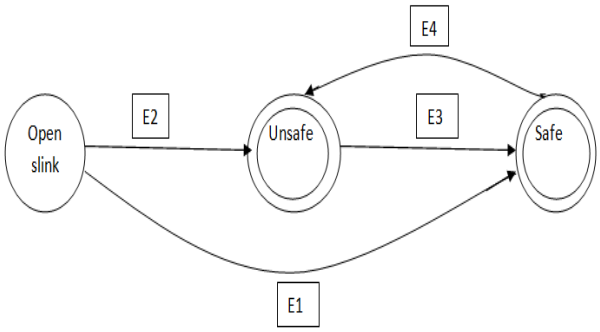

Figure 3 State Diagram

The attack can be detected if a proper rule based system is made which detect the vulnerable code. Here are the rules which can used to detect the attack

Signature Based Detection

Rule 1. If (Onblur and Onfocus events are used with dynamically generated Iframes) along with settimeout
Then (the code is said to be vulnerable).

Rule 2 If (mouse click events are used to detect if an iframe is in focus or not that means onmouseover and onmouseout) along with settimeout
Then (the code is said to be vulnerable).

These rules are applied in the algorithm proposed shown in Figure 5

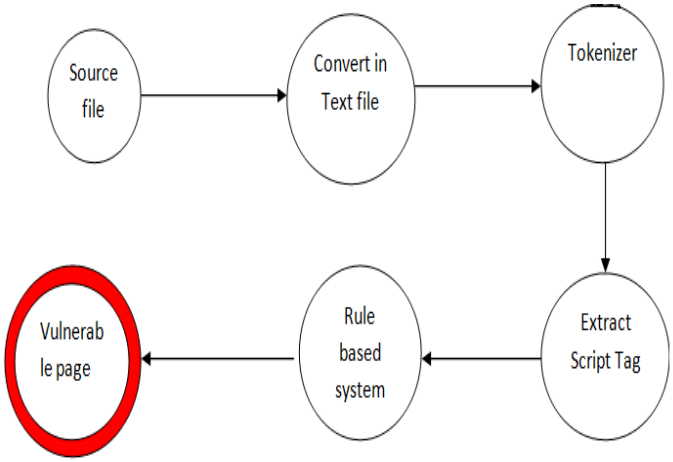

Figure 4 Steps for Detection

```
Define a set of rules to store in database
For Each vulnerable or potentially vulnerable page
        Extract the source code
        Extract the script tag and store it in array
      For Each word w in array W
      Check it against the rules
      If (w is a dynamically generated iframes)
        {
          For Each word from next index
               If (w is onblur)
                {
                  For Each word from next index
                    If (w is settimeout)
                      {
                  For Each word from next index
                              If (w is onfocus)
                              Then
                              Alert is generated
                            } }
                  Else
                  {
                  If (w is onmouseout)
                   {
                  For Each word from next index
                      If (w is settimeout)
                        {
                  For Each word from next index
                            If (w is onmouseover)
                              Then
                              Alert is generated
                                } }}

      Else The Page is Safe
        }
```

Figure 5 Proposed Algorithm

5. Conclusion

In this work contemporary tabnabbing attack has been explored by developing an attack scenario. To understand the behaviour of tabnabbing attack state and sequence diagram are design. During the research work it is found that mostly the tabnabbing attack are enforced through javascript code. Hence a novel method has been developed and proposed to scrutinize vulnerable javascript code in context of tabnabbing attack. The future work will be focused on exploring wider experiments involving scrutinizing vulnerable javascript code through taint analysis, parsing , AST analysis and data flow analysis.

References

[1] Unlu, S.A.; Bicakci, K NoTabNab: Protection against the tabnabbing attack eCrime Researchers Summit (eCrime), 2010.

[2] Anti Phishing Group [ONLINE] Available : http://www.antiphishing.org/reports/apwg_trends_report_h2_2011.pdf

[3] Aza Raskin Tabnabbing: A New Type of Phishing Attack [ONLINE] Available: http://www.azarask.in/blog/post/a-new-type-of-phishing-attack/

[4]Salvatore Guarnieri and Benjamin Livshits GATEKEEPER: Mostly Static Enforcement of Security and Reliability Policies for JavaScript Code

[5] Trenton: Advanced Tabnabbing [ONLINE] Available:http://www.hackyeah.com/2010/06/advanced-tabnabbing-with-iframes/

[6] Jeremiah grossman : CSS History Attacks http://jeremiahgrossman.blogspot.in/2006/08/i-know-where-youve-been.html/

[7] Html tutorial [ONLINE] Available: http://www.w3schools.com/default.asp

[8] Zhi Jian Zhu, Mohammad Zulkernine A model-based aspect-oriented framework for building intrusion-aware software systems Elsevier Information and Software Technology 51 2009 Pages 865–875

[9] Aviv Raff [ONLINE] Available : http://krebsonsecurity.com/2010/05/devious-new-phishing-tactic-targets-tabs/