CLICKJACKING

BRIN PEREIRA

May 12, 2017

1. Abstract:

Clickjacking (User Interface redress attack) is a malicious technique of tricking a Web user into clicking on something different from what the user sees when they are clicking on. This can make user to perform an unintended action that is advantageous for the attacker and harmful to user confidential data. In this survey paper, we will discuss on new proposed variant attacks of clickjacking and also see proposed solution for same and lastly, we will compare the solutions.

2. Introduction:

Robert Hansen and Jeremiah Grossman introduce Clickjacking attack was in 2008. It is an act of stealing or hijacking user mouse clicks to perform actions which is beneficial to the attacker. The attacker achieves this goal by choosing a clickable region and tricks the user by routing him to another web page. The vulnerability can occur in all the browsers as it can insert the code or a script of Clickjacking, which executes without the user's knowledge. Let us consider example of target site (T) such as online bank portal which is important to attacker as it needs important information. Attacker has its own malicious site which is created in such a way that a transparent IFRAME having the content of T overlays the content of M. Since User is not aware of the invisible IFRAME, it correctly aligns T over M.

A real-world clickjacking attack was the example of Twitter where message was propagated among Twitter users. In this attack, the malicious page was embedded in Twitter.com on a transparent IFRAME. The malicious page has a button labeled "Don't Click." and was aligned with "Update" button of Twitter but hidden behind update button. Once user click on update, the link was posted on user profile.

3. Literature work:

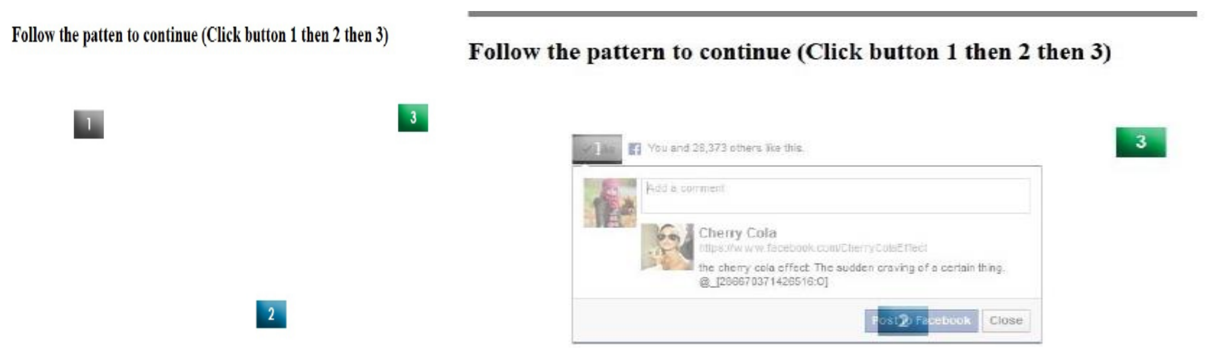
In the paper [1], "On detection and prevention of clickjacking attack for osns", authors have developed attack such as Likejacking and cursor spoofing attack and also proposed solution called Cursor Spoofing and Clickjacking Prevention (CSCP). In Paper [2], “ClickSafe: Providing security against clickjacking attacks.", authors have provided ClickSafe as defense.

4. Survey Details:

4.1. Classification:

In paper [1], author has proposed two attacks (Like Jacking and Cursor Spoofing).

Like Jacking attack: It is a Facebook attack where malicious links are hidden behind Facebook like button. When user clicks on like button, malicious link inside it gets posted on user timeline. This attack follows certain pattern of buttons is at specific location, which shows that behind each button, some function is performed. When user clicks on 1 i.e. click on Facebook like button, a malicious page is inserted and when user clicks on 2 the liked page is shared on the user wall without knowing to the user. When click on 3, proceeds to the actual-website.


Follow the patten to continue (Click button 1 then 2 then 3)
1
3
2
Follow the pattern to continue (Click button 1 then 2 then 3)
1
You and 28,373 others like this.
Add a comment
Cherry Cola
https://www.facebook.com/CherryColaEffect
3
Post to Facebook
2
Close

the cherry cola effect: The sudden craving of a certain thing. @_[288670371428516:0]

Figure 1: Likejacking attack Pattern

Cursor Spoofing / Cursor Jacking: It is user interface redressing technique that changes the position of the cursor and attacker replaces the real cursor by the fake cursor. The user reads quote on Daily Quote website and want to read next quote and clicks on next quote button. The cursor that clicks on the next quote button is a fake cursor and real cursor is spoofed and invisible and is actually present at follow button of Facebook. When user click on Next quote he actually clicks on the follow button of the Facebook and starts following person or page on the Facebook which he or she may not interest to follow.

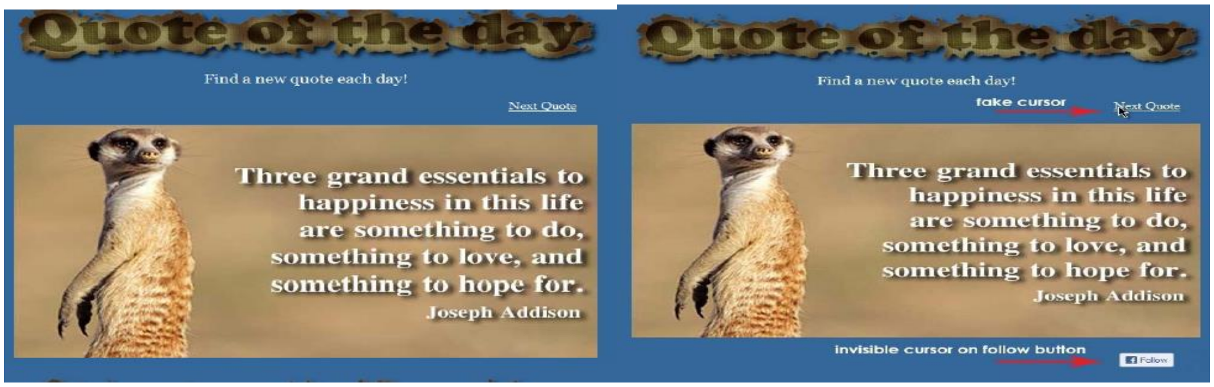

Quote of the day
Find a new quote each day!
Next Quote
Three grand essentials to happiness in this life are something to do, something to love, and something to hope for.
Joseph Addison
fake cursor
invisible cursor on follow button
Follow

Figure 2 : Cursor Spoofing attack for follow button

In paper [4] , A loud video ad plays automatically due to which user clicks on a “skip this ad” link. Cursor on “skip this ad” is a fake cursor and the real click actually lands on the Adobe Flash Player webcam settings dialog that grants the site permission to access the user’s webcam.

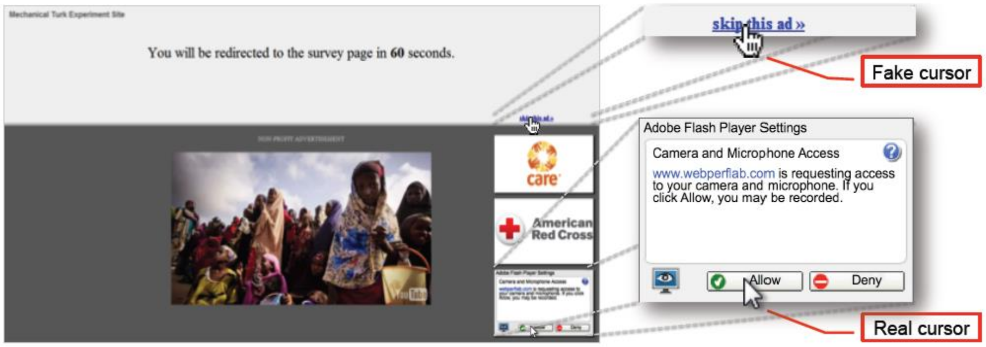

Mechanical Turk Experiment Site
You will be redirected to the survey page in 60 seconds.
skip this ad »
Fake cursor
Adobe Flash Player Settings
Camera and Microphone Access
www.webperflab.com is requesting access to your camera and microphone. If you click Allow, you may be recorded.
Allow
Deny
Real cursor
care
American Red Cross

Figure 3 : Cursor spoofing for changing webcam settings

Authors of paper [1] proposed solution called as *Cursor Spoofing and Clickjacking Prevention*. ***CSCP***: It is a browser-based solution called as Cursor Spoofing and Clickjacking Prevention (CSCP). It is a Google Chrome extension and reason behind selecting Google Chrome was because it has two extensions: *Zscaler* that detects hidden Facebook widgets and other is *I’d like to confirm* which adds a confirm dialog. CSCP has the functionality to detect and prevent Clickjacking and also ask for user confirmation. Also, when cursor spoofing is detected on Webpage, then CSCP will displays both the fake and real cursors and warns the user.

In paper [2], authors have proposed solution Known as *ClickSafe* and it has three major components: *Detection unit, Mitigation unit and feedback unit.* The ***detection unit*** detects malicious elements that redirect users to external webpage. This detection of the external link is done implicitly via JavaScript as well as explicitly via HTML anchor tags. Implicit redirection detection is done when web page loads. ClickSafe will pull all external and internal JavaScript from the page, analyze it using a parser, and then recursively check for redirection. JavaScript can cause a page to redirect in multiple ways, some of which are as follows:

```
window.location.href =””;
window.navigate();
 self.location=””;
```

For explicit redirection attempts, preventDefault() and stopPropagation() are used to prevent the anchor tag event handler from the executing the hyperlink. The hyperlink is executed later when the user confirms that she wants to be redirected a specific page. The ***mitigation unit*** provides warnings to users whenever a redirection is found. A popup displays the URL clicked and site’s rating is displayed to the user. The user then can decide to continue or cancel. ClickSafe has ***feedback unit*** that records the user’s actions and converts them into ratings. Every time when user responds to the popup by clicking a yes or a no, her actions are recorded and submitted to a server as feedback. To the question, “Do you want to continue?” preceded by information about the link, a yes will give the site a positive rating while a no will give the site a negative rating. This feedback can be changed later if the user has accidentally provided the wrong feedback.

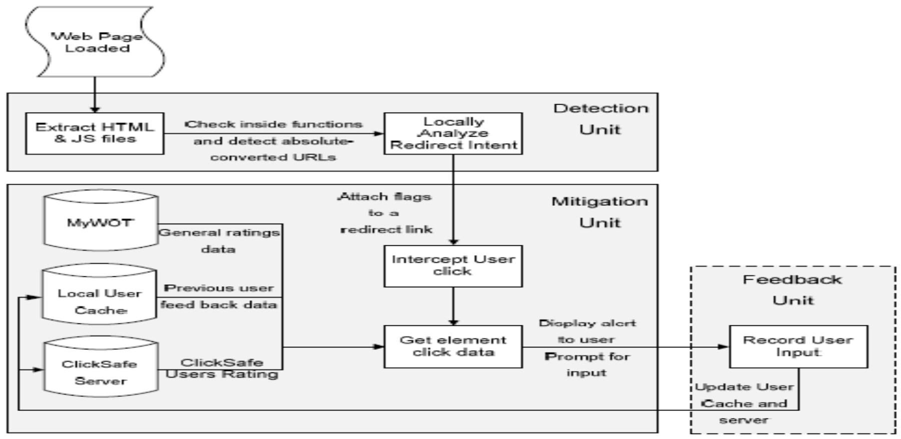
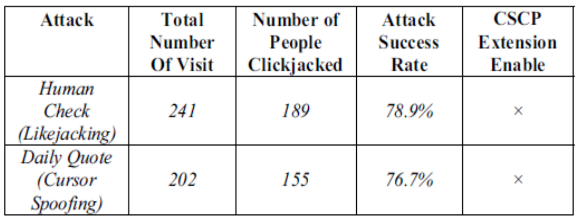


Figure 4: ClickSafe Architecture

4.2. Comparison:

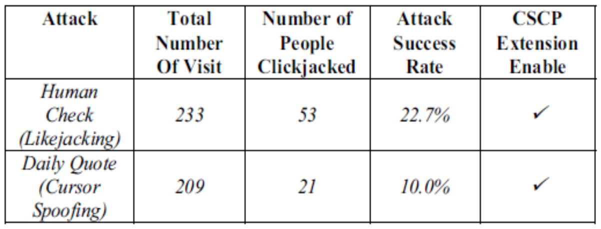
Authors of paper [1], have provided details of CSCP experiment. The log was maintained for website using File Transfer Protocol (FTP), which generate two files. One file has counter that increment when a user visits the web page and other file also has the counter that increments when a user clicks on the like or follow button. Below table shows the success rate of attack before CSCP and after CSCP.

TABLE 1: SUCCESS RATE OF ATTACK BEFORE CSCP



Attack	Total Number Of Visit	Number of People Clickjacked	Attack Success Rate	CSCP Extension Enable
Human Check (Likejacking)	*241*	*189*	*78.9%*	×
Daily Quote (Cursor Spoofing)	*202*	*155*	*76.7%*	×

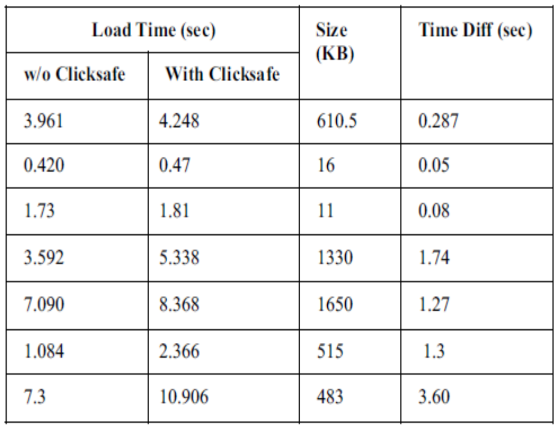
TABLE 2: SUCCESS RATE OF ATTACK AFTER CSCP



Attack	Total Number Of Visit	Number of People Clickjacked	Attack Success Rate	CSCP Extension Enable
Human Check (Likejacking)	*233*	*53*	*22.7%*	✓
Daily Quote (Cursor Spoofing)	*209*	*21*	*10.0%*	✓

Figure 5: comparison of success rate before CSCP and after CSCP

Authors of Paper [2], conducted the experiment and found out that the difference of webpage load time with ClickSafe and without ClickSafe.



Load Time (sec)		Size (KB)	Time Diff (sec)
w/o Clicksafe	With Clicksafe		
3.961	4.248	610.5	0.287
0.420	0.47	16	0.05
1.73	1.81	11	0.08
3.592	5.338	1330	1.74
7.090	8.368	1650	1.27
1.084	2.366	515	1.3
7.3	10.906	483	3.60

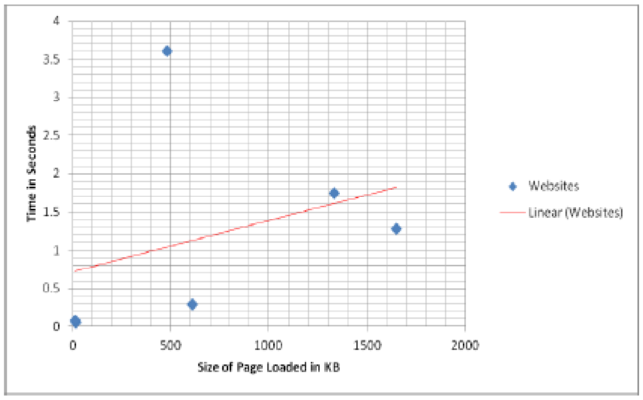


Fig. 2 - Page Load time as a function of page size

Figure 6: comparison of webpage load time with and without ClickSafe

4.3.Shortcomings and weakness:

Drawbacks for ClickSafe:

- User Feedback is unreliable. Attacker can provide false feedback.
- Bots can be used to damage ratings of websites.

- The dynamic nature of JavaScript can start as few lines of code initially and then multiply itself via Ajax calls and function rewriting to produce hundreds of lines of code. This disrupts static analysis.

4.4. Lessons Learned:

At present, it is unclear to what extent clickjacking can be used by attackers and how much damage it can cause on the security of Internet users. There are many websites which still have not implemented effective protection against clickjacking attacks. Clickjacking attack can cause several threats like stealing personal data such as bank account information, credit card information, user credentials and social security numbers or installing software applications on a computer or changing webcam settings or browser settings.

5. Conclusion and Future Work:

Future scope for ClickSafe can be extending the work by building dynamic analysis of JavaScript code obfuscation methods. Moreover, it will be better to have encryption/authentication system which does not allow the feedback system to be compromised.

6. Bibliography/Citation:

1. Rehman, Ubaid Ur, et al. "On detection and prevention of clickjacking attack for osns." Frontiers of Information Technology (FIT), 2013 11th International Conference on. IEEE, 2013.
2. Shamsi, Jawwad A., et al. "Clicksafe: Providing security against clickjacking attacks." High-Assurance Systems Engineering (HASE), 2014 IEEE 15th International Symposium on. IEEE, 2014.
3. Balduzzi, Marco, et al. "A solution for the automated detection of clickjacking attacks." Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. ACM, 2010.
4. Huang, Lin-Shung, et al. "Clickjacking: Attacks and Defenses." USENIX Security Symposium. 2012.