Securing Clicks Against Clickjacking

Rahul Govindkumar
Computer Engineering
Pillai College of Engineering
Mumbai, India
rahulgovindkumar@gmail.com

Quasim Rizvi
Computer Engineering
Pillai College of Engineering
Mumbai, India
quasi.rizvi@gmail.com

Alankrita Singh
Computer Engineering
Pillai College of Engineering
Mumbai, India
alan.singh244@gmail.com

Megha Revoo
Computer Engineering
Pillai College of Engineering,
Mumbai, India
megha.revoo@gmail.com

Madhumita Chatterjee
Computer Engineering
Pillai College of Engineering,
Mumbai, India
mchatterjeee@mes.ac.in

***Abstract*— The Internet has played a crucial role in revolutionizing the entire world. Our thought process, the way we conduct business, communicate, entertain ourselves et cetera all have been modified since the birth of the World Wide Web. All this is accompanied by enormous security threats. Although many security mechanisms have been deployed in the past but they have turned obsolete with time. The need has always been to provide secure Internet access to users across the globe. However, Internet being an open medium this has so far been an unfulfilled dream. CSS, being vital for web pages, has turned out to be a source of a new type of attacks commonly known as Clickjacking. Clickjacking is the malicious practice of manipulating a website user's activity by concealing hyperlinks beneath legitimate clickable content, thereby leading the user to perform actions of which they are unaware. Clickjacking is peculiar in itself, as in this the user falls prey to a trap set based on human cognitive abilities, hence emancipating the possibility of such an attack is almost impossible. We aim to design a solution to prevent users falling prey to such attacks.**

Keywords-components: Clickjacking, Cyber Security, Extension, Opacity, Z-index, Cursor Spoofing.

I. INTRODUCTION

Day after day, web users are increasing at an enormous rate and human dependence on web services is growing exponentially. The concept of the global village is possible only because of the internet which facilitates information exchange anywhere and at any time. Unfortunately, all these facilities come with security risks. The various cyber threats include headless worms, machine-to-machine attacks, jailbreaking, ghostware and two-faced malware. Although this may sound intimidating, given everything that is at stake, we have cyber security (to fight against it) as the preventive measure.

CYBER SECURITY

Cyber security is the body of technologies, processes, and practices designed to protect networks, computers, programs and data from attack, damage or unauthorized access. It involves protecting information and systems from major cyber threats, such as cyber terrorism, cyber warfare, and cyber espionage. In their most disruptive form, cyber threats aim secret, political, military, or infrastructural assets of a nation or its citizens. Cyber security is, therefore, a critical part of any government's security strategy.

CLICKJACKING

Clickjacking attack was introduced by Robert Hansen and Jeremiah Grossman in 2008, to steal user-initiated mouse clicks to perform actions that the user is not interested in. The attacker achieves the goal by smartly setting a trap at a clickable region on a web page e.g. the region where the login button on the web page is located and the user is asked to enter his or her username and password. On clicking, malicious web page loads from the website inside an iframe, which makes use of Cascading Style Sheets (CSS) to make the targeted region transparent. In this region, different flavours of Clickjacking are used to trick the user like deploying fake cursor, transparent buttons, et cetera. The region might also be overlapped by another element on the website. Technically, both the JavaScript and CSS are used to place the iframe under the mouse cursor to make the user click in the targeted region resulting in a malicious action the attacker is intended to do.

The vulnerability can occur in all the browsers to embed the code or a script of Clickjacking, which executes without the user's knowledge. Clickjacking attack can cause several threats like stealing personal data such as bank account information, credit card information and social security numbers or installing software applications on a computer.

II. Literature Survey

In paper [1] they have devised new clickjacking attack variants, which bypass existing defences and cause more harm than previously known, such as compromising webcams, user data, and web surfing anonymity.

To defend against clickjacking in a fundamental way, they have proposed InContext, a web browser or OS mechanism to ensure that a user's action on a sensitive UI element is in context, having visual integrity and temporal integrity. The concept of context integrity is introduced and is used to define and characterize clickjacking attacks and their root causes. They have designed, implemented, and evaluated InContext, a set of techniques to maintain context integrity and defeat clickjacking.

The authors in [2] have proposed attacks based on Likejacking and Cursor spoofing. They mostly affect the users who are very sensitive about their personal information. The attacks may also be modified to steal the user credential in form of username, just two extensions for the prevention of Clickjacking attack named as:

A. *Zscaler Likejacking Prevention*

The Zscaler Likejacking Prevention detects hidden Facebook widgets and warns users about Likejacking.

B. *I'd like to confirm* the password, pictures, and any private information that has more value for the users.

The proposed attacks are launched into two different scenarios as Use of CAPTCHA and Use of Interest. The proposed attack is a type of human authentication scheme in which user is asked to follow a certain pattern to allow the user access to the actual website. This paper has proposed defence by creating Google Chrome extension to prevent user against Likejacking and Cursor Spoofing attacks. Google Chrome was selected because it has it has just two extensions for the prevention of Clickjacking attack adds a confirm dialog to every Facebook Like button in order to prevent Clickjacking. The proposed defence covers the functionality of both the existing extensions and also ensures pointer integrity. Hence the name given to it is Cursor Spoofing and Clickjacking Prevention (CSCP). CSCP has the functionality of detecting and preventing Clickjacking attacks on the Facebook. When the pointer clicks on like or follows button, a pop-up appears to the user that is clicked. When a cursor spoofing is detected on the websites, it displays both the fake and real cursors and warns the user that the website is compromised.

III. Browser Wise Solution for clickjacking

<u>Solution in Google Chrome:-</u>

There are two options to prevent clickjacking in Google Chrome.

1. Javascript framebuster:

A Framebuster script basically prevents frames from external websites from displaying the target website without permission, often as part of the clickjacking attack. JavaScript Framebuster script detects such clickjacking and breaks the frame when that external website is loaded and redirects the visitor to the target website.

2. X-frame options:

X-frame-option is the better solution as it solves the problem completely by blocking framing. X-frame-options are server-side set http headers. They are segment of the security protection against malicious attacks like Clickjacking. Chrome will not let you ignore or modify it.

<u>Solutions in Mozilla Firefox:</u>

Solution 1: Disable scripting and plugins

By selecting ***tools>add ons>plugin>disable***, we can keep the malicious attackers from creating transparent layers and luring the users into opening harmful sites. Once we disable the plugins we need to shut down the browser and delete adobe flash and remove it from the system.

Solution 2: By downloading the latest version of No Script Firefox plugin

Once we download the latest version of NoScript plugin, NoScript plugin pre-emptively blocks all the malign sites and let the user access java, javascript from only those sites which are trusted and authorized. Once we install NoScript plugin, we need to restart the system then select the NoScript icon located at the bottom of the right ***statbar>options>forbid[iframe]>ok.*** Thus, we can be assured of the fact that users will be able to gain access to only safe and trusted sites.

Solution 3: X-Frame options

One of the best solutions to clickjacking is X-Frame server response header. We need to set the X-Frame options header to the same origin or deny preventing the target web application from being loaded with an I-frame hence the clickjacking attacks will be averted successfully.

Solutions in Internet Explorer:

The solution that was devised to tackle clickjacking attack was by using X-frame options response header. It is used to check whether the page that will be displayed is safe to be displayed in a frame or no.
There are three values for this:
Deny: The page would not get displayed in the frame even if the site tries to do so.
Same origin: Page A will get opened in a frame if and only if the site that contains the frame is same as the site which will be serving that Page A in the frame.
Allow from URI: The page gets displayed in a frame on the origin that is specified.

IV. PROPOSED SYSTEM

There are various defences available to prevent Clickjacking but all the techniques have some limitation which gives the attacker a way to attack. Our proposed system is also an approach to prevent Clickjacking attack. In this system, we are making an extension named **CJ** which is a small piece of code and is browser independent. Our proposed system will detect Clickjacking attack on all websites. For detection, it will check for *cursor spoofing*, *Z-index*, and *opacity*. Fig 3.1 shows the architecture of the proposed system.

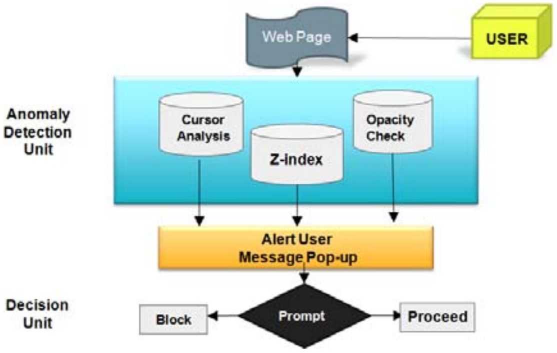


Figure 1: Architecture of Proposed System

IMPLEMENTATION METHODOLOGIES

Anomaly Detection Unit

The anomaly detection unit will start searching as soon as the web page loads. It will check whether Clickjacking is happening or not and if it is then a corresponding alert is generated to alert the user and he is prompted to either block it or proceed at his own risk.

A. *Opacity Check*

First one is for opacity check. In this, our extension will inspect the source code to check if there is any element which has both its background color and font color as transparent. If it is so then there is really something wrong and we will generate an alert for the user.

B. *Z-Index Check:*

Next is the Z-index check. Z-index basically defines which layer of the webpage is closer to the human eye. First, our extension would shortlist all elements which have position attribute not set as static as z-index is not defined for such elements. Then it will filter out elements closest to our eye, i.e. having max z-index. If these filtered elements are found to have transparent bg color and font color then an alert will be generated.

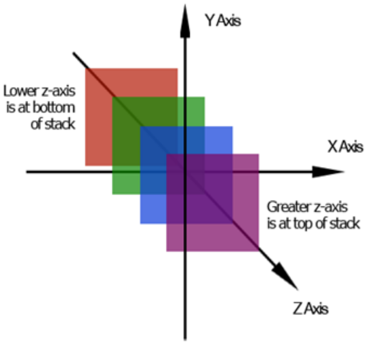

Y Axis
Lower z-axis is at bottom of stack
X Axis
Greater z-axis is at top of stack
Z Axis

Figure 2: Z-Index (http://vanseodesign.com/css/css-stack-z-index/)

C. *Cursor Analysis:*

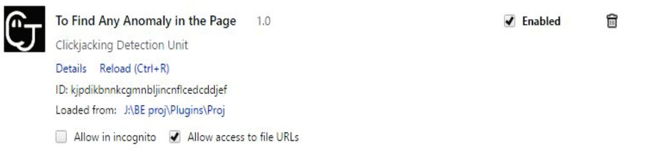
Last is the Cursor Analysis. In this, our extension will check if the cursor is set to none because in that case, it will be not visible. If found so then an alert will be generated.

All these algorithms will keep on executing after a random time interval to keep Clickjacking under check.

V. RESULTS ACHIEVED

For testing the CJ against various test cases one must install CJ Google Chrome Extension as mentioned in the illustration given below.

Enabling the extension from *Settings > Extension > Enable*

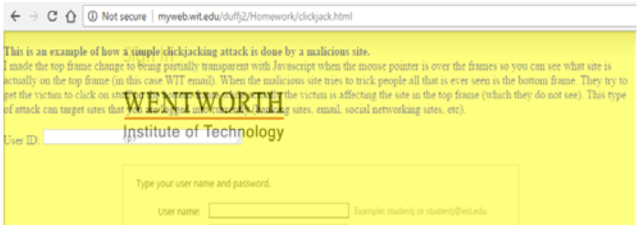


To Find Any Anomaly in the Page 1.0 ☑ Enabled
Clickjacking Detection Unit
Details Reload (Ctrl+R)
ID: kjpdikbnnkcgmnbljincnflcedcddjef
Loaded from: J:\BE proj\Plugins\Proj
□ Allow in incognito ☑ Allow access to file URLs

Figure 3 : Enabling the Extension

A. Opacity Anomaly detection

The figure below shows a site with opacity anomaly. As it loads completely CJ detects it and generates an alert as shown. CJ thoroughly checks for any transparent outlay, which has no purpose on the web page other than clickjacking.



Not secure | myweb.wit.edu/duffj2/Homework/clickjack.html

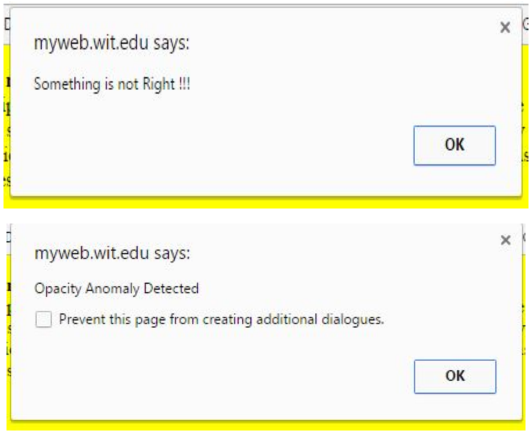
This is an example of how a simple clickjacking attack is done by a malicious site.
I made the top frame change to being partially transparent with Javascript when the mouse pointer is over the frames so you can see what site is actually on the top frame (in this case WIT email). When the malicious site tries to trick people all that is ever seen is the bottom frame. They try to get the victim to click on stuff on the bottom frame while actually the victim is affecting the site in the top frame (which they do not see). This type of attack can target sites that you are logged into currently (banking sites, email, social networking sites, etc).

User ID:

WENTWORTH
Institute of Technology

Type your user name and password.
User name: Example: studentj or studentj@wit.edu

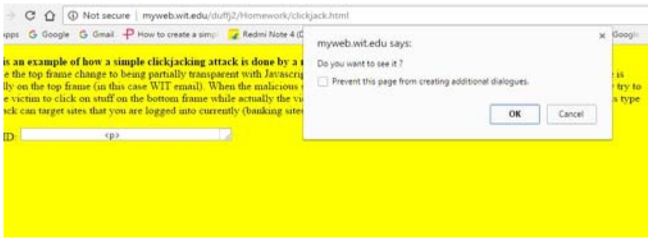
Figure 4: Web page [9] containing Clickjacking code



myweb.wit.edu says:
Something is not Right !!!
OK

myweb.wit.edu says:
Opacity Anomaly Detected
□ Prevent this page from creating additional dialogues.
OK

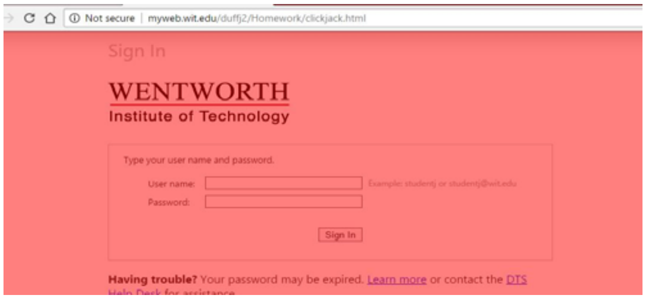
Figure 5: CJ Detects Opacity Anomaly



Not secure | myweb.wit.edu/duffj2/Homework/clickjack.html

myweb.wit.edu says:
Do you want to see it ?
□ Prevent this page from creating additional dialogues.
OK Cancel

Figure 6: Prompts User to Take Action



Not secure | myweb.wit.edu/duffj2/Homework/clickjack.html

Sign In

WENTWORTH
Institute of Technology

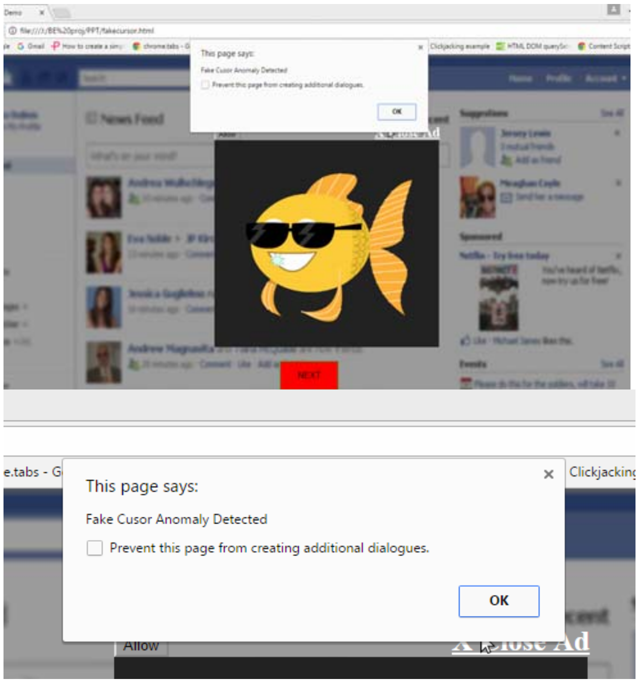
Type your user name and password.
User name: Example: studentj or studentj@wit.edu
Password:
Sign In

Having trouble? Your password may be expired. Learn more or contact the DTS Help Desk for assistance.

Figure 7: Exposes the hidden elements.

B. Fake cursor detection

The figure below shows a website having fake cursor. Without our extension, one can end up granting permission of some application. With our CJ such sites can be easily detected and hence fake cursor anomaly is prevented from happening.



file:///J:/BE%20proj/PPT/fakecursor.html

This page says:
Fake Cusor Anomaly Detected
□ Prevent this page from creating additional dialogues.
OK

This page says:
Fake Cusor Anomaly Detected
□ Prevent this page from creating additional dialogues.
OK

Figure 8: CJ Detects Fake Cursor

C. Fake Button Detection

The figure below shows a website having hidden malicious buttons. Without our extension, one can get amused by playing such games and thereby providing multiple responses which can be used by the hacker in his interests. With our CJ such scenarios can be easily detected and hence clickjacking can be prevented from happening.

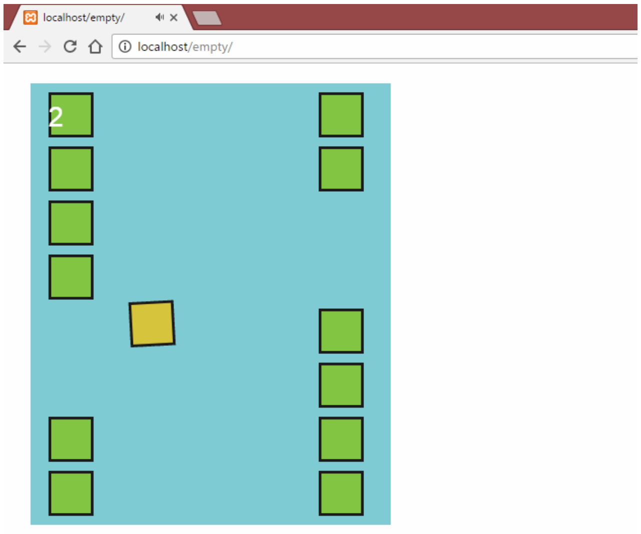
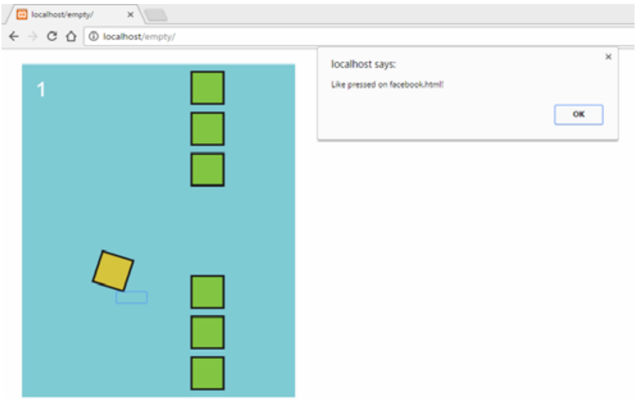


Figure 9 : Game with hidden malicious buttons

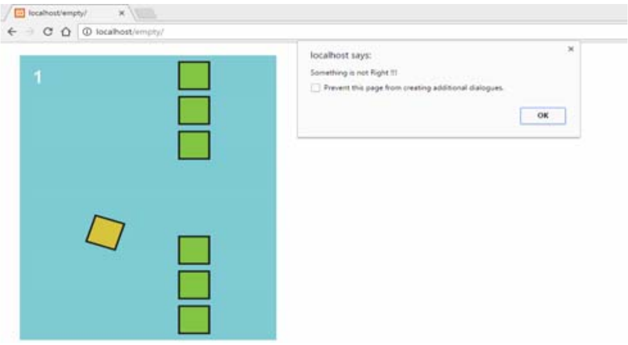


localhost says:

Like pressed on facebook.html!

OK

Figure 10: Falling into hackers trap



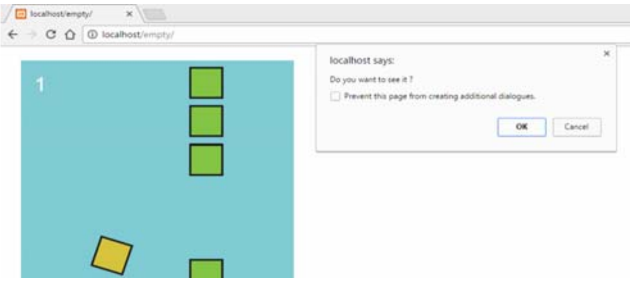
localhost says:

Something is not Right !!!

□ Prevent this page from creating additional dialogues.

OK

Figure 11: Detection By Our CJ



localhost says:

Do you want to see it ?

□ Prevent this page from creating additional dialogues.

OK Cancel

Figure 12: Prompts User to Take Action

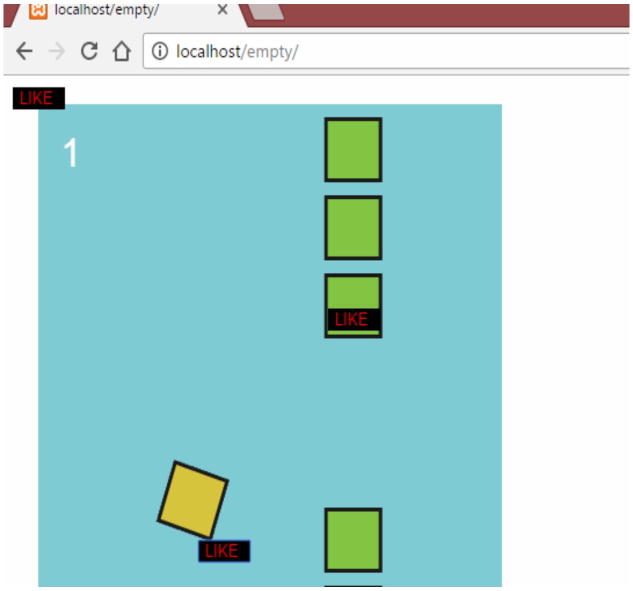


Figure 13: Revelation Done By CJ

The following Table 1 shows the pros and cons of the different recommendation techniques:

Browser	Server Side Security	Detects Manipulation	Notify User	Similar Extension Available
Default Chrome	Not very secure	X	X	X
Default Mozilla Firefox	Secure	X	X	✓
Default IE	Secure	X	X	X
Opera	Not available	X	X	X
Safari	Not Supported	X	X	X
Chrome Browser with our Extension	Not applicable	✓	✓	--

Table 1: Our Solution versus Existing Browser Solution

V. ACKNOWLEDGEMENT

We would like to take this opportunity to express our profound gratitude and deep regard to Prof. Dr. Madhumita Chatterjee for her guidance and constant encouragement throughout the course of this project. We are immensely obliged for her cordial support, supervision and providing necessary information.

We remain immensely obliged to Dr. Madhumita Chatterjee for introducing this topic, and for her invaluable support in garnering resources for us either by way of information or computers also her guidance and supervision which made this project happen. We are thankful to our college, Pillai College of Engineering for providing us healthy competitive environment and outstanding educational facilities that played an important role in keeping us highly motivated to achieve our goals.

VI. CONCLUSION

With an increase in the usage of the internet, protection against Clickjacking will become a necessity in coming days. Many solutions came and became obsolete with time. However while designing CJ extra care was taken to make it robust and a solution anyone could use with ease. This system will check for any anomalies pertaining to Clickjacking attacks present in web pages. Since it is an Extension:

- It is Operating System independent.
- It is very simple to install unlike intricate installations of OS based applications.
- It is user-friendly yet efficient enough to minimize cyber threats like XSS, Phishing and Social Engineering threats which can be carried out using Clickjacking,

Hence preventing Clickjacking would imply further fortification of cyber security by minimizing these attacks.

In future, this system can be redesigned to handle tapjacking by giving it a shape of a mobile application. With each passing day, handheld devices are gaining prominence over desktop computers due to many reasons, but the major attraction is in it being a versatile all in one package.

Every click is crucial and looking at security measures currently in use, **"We are just one click away from being clickjacked, but our CJ will secure this one click."**

References

[1]Alex Moshchuk, Collin Jackson, Helen J. Wang, Lin-Shung Huang, Stuart Schechter, "Clickjacking: Attacks and Defenses", Microsoft Research, Carnegie Mellon University, April 2011, https://www.linshunghuang.com/papers/clickjacking.pdf

[2]Nazar Abbas Saqib, Muhammad Kaleem, Ubaid Ur Rehman, Waqas Ahmad Khan, "On Detection and Prevention of Clickjacking Attack for OSNs", College of Electrical and Mechanical Engineering National University of Sciences and Technology Islamabad and School of Electrical Engineering and Computer Science National University of Sciences and Technology Islamabad, Pakistan, December 2013, https://www.researchgate.net/publication/261279635

[3]Collin Jackson, Dan Boneh, Elie Bursztein, Gustav Rydstedt, "Busting Frame Busting: a Study of Clickjacking Vulnerabilities on Popular Sites", Carnegie Mellon University, Stanford University, June 2010, https://crypto.stanford.edu/~dabo/pubs/papers/framebust.pdf,

[4]Agnes.A, Hajera.S.H, Jeena James, "Analysis Detection and Prevention of Users from Click Jacking Attack Using DDoS", Computer Science and Engineering, DMI College of Engineering, Chennai, November 2010, https://www.ijedr.org/papers/IJEDRCP1501002.pdf,

[5]Mrs. Dipti Y. Pawade, Ms. Abhilasha Lahigude, Ms. Divya Reja, "Review Report On Security Breaches Using Keylogger And Clickjacking", Dept. of IT, KJCOE, Mumbai, January 2015, http://www.ijafrc.org/Volume2/ncrtit2015/9.pdf,

[6]Borase Prashant, Jaware Mayuri, Mahajan Neha, Prof. V.M. Vasava, "Online Detection & Prevention of Clickjacking Attacks" Dept. Of Computer, Gangamai College of Engineering, Nagaon, Maharashtra, India, October 2015, http://www.ijarsmt.com/docs/issues/mahajan-nehajaware-mayuri-borase-prashantprof-vm-vasava--24.pdf, [Accessed: August14, 2016 & 10:20PM]

[7]Neha Gupta,Sayali Kandarkar, Shreya Bapat, Prof. Dipti Pawade, "A Survey on Clickjacking and Tapjacking Solutions Provided by Different Browser", Dept. of I.T., K. J. Somaiya College of Engineering, Mumbai, India, December 2015

http://www.ijircce.com/upload/2015/december/30_A_Survey.pdf,

[8]Christopher Kruegel, Davide Balzarotti, Engin Kirda,

Manuel Egele, Marco Balduzzi, “A Solution for the Automated Detection of Clickjacking Attacks”, Institute Eurecom Sophia-Antipolis, University of California, Technical University Vienna, https://seclab.cs.ucsb.edu/media/uploads/papers/asiaccs10_click.pdf, [Accessed: August10, 2016 & time:11:20AM]

[9]Clickjacking Test case. http://myweb.wit.edu/duffj2/homework/clickjack.html