Performing clickjacking attacks in the wild: 99% are still vulnerable!

Daehyun Kim
Department of Computer Science and Engineering,
Sungkyunkwan University, Republic of Korea
Email: dan.0901@skku.edu

Hyoungshick Kim
Department of Computer Science and Engineering,
Sungkyunkwan University, Republic of Korea
Email: hyoung@skku.edu

***Abstract*—Clickjacking is an attack that tricks victims into clicking on invisible elements of a web page to perform unintended actions that might be advantageous for the attacker. To defend against clickjacking, many techniques have been proposed, but it is still questionable whether they are effectively deployed in practice. We investigated how vulnerable Korean websites are to clickjacking attacks by performing real attacks on the top 500 most popular Korean websites as well as all of the financial websites. Our results are quite significant: almost all Korean websites (99.6%) that we looked at were vulnerable to clickjacking attacks. Extending our observation to top 500 global websites, we found that 390 of them (78%) were also vulnerable to clickjacking attacks and identified which type of website is particularly insecure against clickjacking.**

Keywords*—*Clickjacking, frame busting, Korean websites.

I. Introduction

A web framing attack, called *clickjacking* [1], uses a transparent `iframe` (the HTML tag to specify an inline frame which is used to embed another document within the parent HTML document) to hijack the users' clicks. In a typical clickjacking attack scenario, a malicious web page is constructed by an attacker in order to trick the victim into clicking on elements of the web page within an invisible `iframe` to perform unintended actions. Recently, clickjacking attacks have been considerable interest and many prevention techniques have been proposed [2]. However, it is still unclear whether these defense mechanisms are effectively deployed in practice.

We present an empirical study in which we analyze the feasibility of clickjacking attacks by testing the 500 most popular and all 36 financial institution websites in Korea (a total of 526 unique Korean websites). Surprisingly, our experimental results show that 524 out of these 526 websites (about 99.6%) were still vulnerable to clickjacking attacks. Only two of the websites could not successfully be framed by clickjacking attacks. To compare this disastrous situation with the global standard, we also investigated the clickjacking possibility of the top 500 global websites and found that 390 of them (78%) were vulnerable to clickjacking attacks.

Unlike the previous study by Rydstedt et al. [3], we focused on exploring the regional websites in a country, as well as global websites. We were curious as to whether an attack such as clickjacking could still be effective in a particular country (i.e., Korea) even though the attack and defense techniques are well known in another country (i.e., United States). Our key observations can be summarized as follows:

- First, our results show that the most popular Korean websites (524 out of 526 websites – about 99.6%) were much more vulnerable to clickjacking attacks than the global websites (390 out of 500 websites – 78%). We also extended these results to a more generalized cross-country analysis; there were overall country differences in the feasibility of clickjacking attacks.
- Second, we categorized the tested websites by type and analyzed the risk of clickjacking attacks for them. We observed that those websites related to businesses such as banking (11 out of 33, about 33.3%) were more secure than the other types of websites, which might largely be due to differences in the level of investment in web security. As would be expected, those websites with a bug bounty program were also more secure. This suggests that running a bug bounty program can help companies enhance their security.

II. What is clickjacking?

Clickjacking is a web-based attack that was first reported by Jeremiah Grossman and Robert Hansen in 2008 [1]. It is a technique in which the user is induced to click on an element of a web page which is designed by the attacker [3]. Fig. 1 illustrates an example of clickjacking attacks. Fig. 2 also shows how to create such a transparent inline frame. We can see that an invisible inline frame (i.e., `iframe`) can be simply made by setting the Cascading Style Sheet (CSS) property to control the opacity of the HTML elements.

To generalize clickjacking attacks, we use N to represent a normal web page (e.g., web-portal, banking, social networking services) with an innocuous UI element UI_N. An attacker creates a malicious web page M with an element UI_M to perform a malicious action (e.g., downloading malware, sending an email message to the attacker, liking the attacker's post) by transparently overlaying UI_M on top of UI_N. Hence, when he or she tries to click on UI_N within N, the victim indeed clicks on UI_M within N which triggers an unintended action instead of UI_N. Clickjacking attacks can also be more sophisticated than simply hiding the target element as follows:

- Using the CSS cursor property, attackers can hide the default cursor and programmatically draw a fake

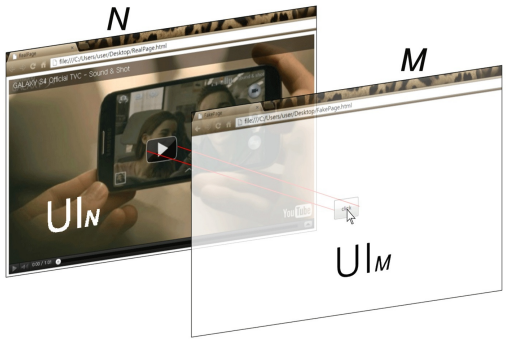


Fig. 1. An example of clickjacking attacks. An invisible web page (M) with a button (UI_M) to run a malicious action is put on top of what appears to be a normal web page (N) with a video clip (UI_N). The victim can see the play button on the video clip, but cannot see the button in the transparent web page. When he or she clicks on the play button, the victim actually clicks on the button in the transparent web page which triggers a malicious action.

```
<html>

<title>An example page</title>

<body>

<style>
iframe {
        filter:alpha(opacity=0);
        opacity:0;
}
</style>

<iframe src="./FakePage.html" border="0"
    scrolling="no">
</iframe>

</body>

</html>
```

Fig. 2. An HTML document with a transparent inline frame. An element in the HTML document can be hidden by setting the CSS property `opacity:0`, which makes the inline frame's CSS opacity value zero. In addition, an attack can specify the option for scrolling to remove the scroll bar from the inline frame.

cursor elsewhere [4], or alternatively set a custom mouse cursor icon to a deceptive image that has a cursor icon shifted several pixels from the original position [5].

- Attackers can use JavaScript – a single click can be changed into a double click which can click on UI_N as well as UI_M at the same time. Unless the intended action is performed by clicking on UI_N, however, careful users may suspect that the visiting web page might be strange or deceptive.
- Clickjacking can be used with other attacks such as Cross-Site Request Forgery (CSRF) which is a widely exploited website vulnerability whereby unauthorized commands are transmitted from the user that the website trusts [6].

III. Feasibility of Clickjacking Attacks

We evaluated the risk of clickjacking attacks for the 500 most popular websites and 36 financial websites in Korea, which might be obviously high-risk targets for clickjacking. We also compared these test results with the global standard by conducting the same experiments with the 500 most popular global websites.

A. Analysis for Korean websites

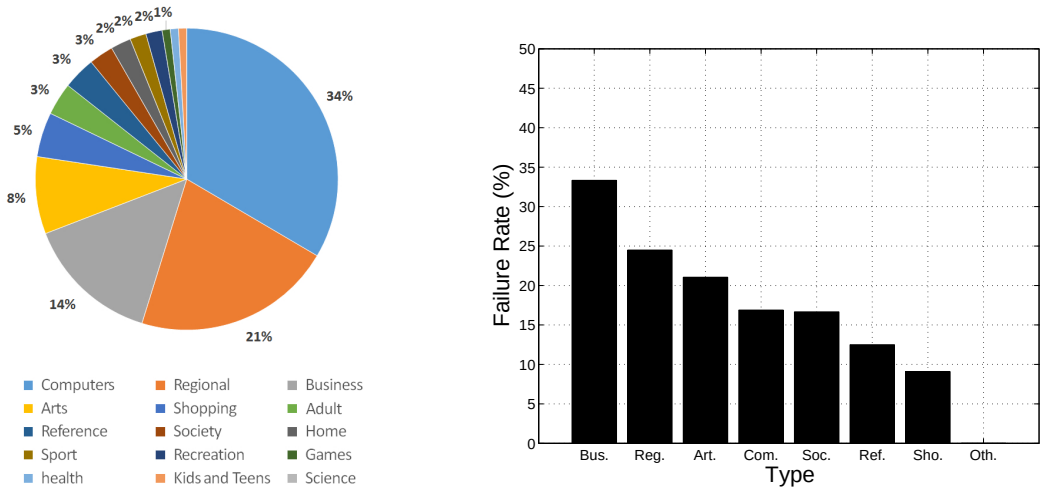
We used (1) the 500 most popular Korean websites and (2) all 36 financial institution websites to analyze the feasibility of clickjacking attacks for Korean websites. Consequently, we used a total of 526 unique Korean websites since they were not mutually exclusive. To test whether clickjacking attacks can be successfully achieved, we used the two most popularly used web browsers (Internet Explorer and Chrome) which run on a desktop PC. The reason why we chose only two web browsers is that we had already found that there was no difference between the five web browsers [7]; we also used onlt Internet Explorer 8 for the same reason. That is, for each combination of website S in the top 526 websites and web browser B in $\{$Internet Explorer 8, Chrome$\}$, we tested whether website S can be successfully framed by the test website representing the attacker's malicious website M and still properly work.

We found that all but eight (Google, Facebook, Twitter ,Yahoo, Pinterest, League of Legends, Citi Bank and Netsgo) of the Korean websites were vulnerable to clickjacking attacks. Interestingly, only two of these are Korean websites (Citibank and Netsgo). We surmise that this is because global websites such as Google and Facebook have already experienced such attacks and developed defense mechanisms to mitigate them. To analyze the difference between the Korean and global websites, we also evaluated the top 500 global websites taken from the Alexa Top-500 Global Sites (http://www.alexa.com/topsites) in the same manner (see Section III-B) and found that 390 of them could be effectively framed by our clickjacking attacks; this proportion (75%) also seems high but is significantly less than that of the tested Korean websites ($p < 0.0001$, Fisher's exact test). This shows that the Korean websites were significantly more vulnerable to clickjacking attacks than global websites.

As mentioned above, only two of the Korean websites (Citi Bank and Netsgo) were secure against clickjacking attacks. For this purpose, the Netsgo website uses `X-Frame-Options`, while the Citi Bank website uses JavaScript code.

The `X-Frame-Options` HTTP response header (used for Netsgo) can be applied to indicate whether a web page within an inline frame can be rendered by web browsers [8]. This approach can be effectively implemented if the web browser supports the related functionality. Both the Internet Explorer 8 and Chrome browsers used in our experiments already supported this feature.

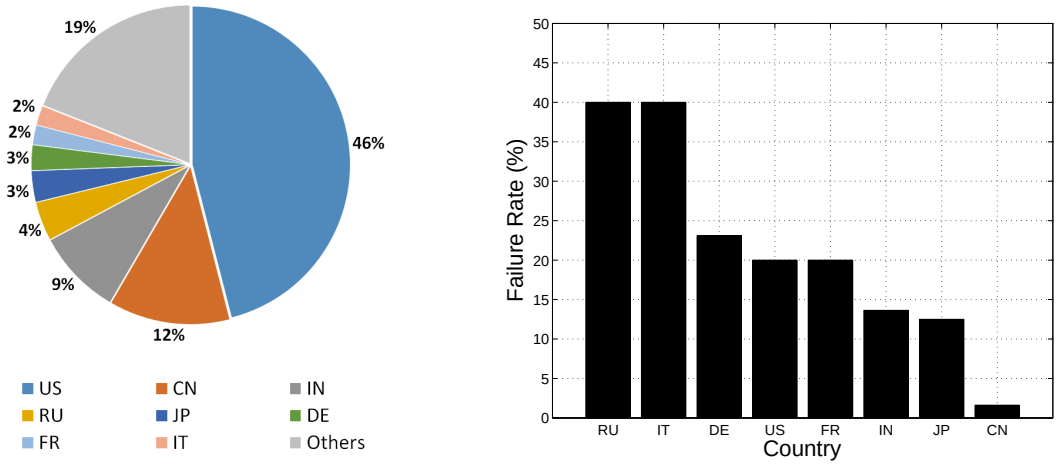
Using a frame busting code is another popular approach to prevent clickjacking. A common frame busting code is made up of a conditional statement (e.g.,



(a) Types of websites

(b) Attack failure rates by type

Fig. 3. Clickjacking attack failure rate by website type.



(a) Countries of websites

(b) Attack failure rates by country

Fig. 4. Clickjacking attack failure rate by website country.

`top.location != self.location`) and a counter action (e.g., `top.location = self.location`) to prevent the web pages from being loaded within an inline frame [3]. Twelve of the Korean websites not including the Citi Bank and SinHyup Bank websites, used such frame busting codes which rely purely on JavaScript to detect framing and prevent it, but can easily be bypassed by disabling the JavaScript HTML attributes; if JavaScript is disabled in the context of an `iframe`, the frame busting codes might not work properly against clickjacking attacks. We tried to restrict the use of JavaScript with the attribute `security="restricted"` for Internet Explorer 8 and `sandbox` for Chrome, Firefox, and Safari.

There was only one website (Citi Bank - http://www.citibank.co.kr/) which use a properly working defense code against clickjacking, which is similar to Rysdstedt's recommendation [3]: In this case, when a web page is first loaded, the style tag hides all of the contents on the web page (`html{display:none;}`). If JavaScript is disabled, the web page will remain blank. Similarly, if the web page is framed, it will either remain blank or it will attempt to frame bust. The script only reveals the document's contents if the web page is not running in a frame.

The SinHyup Bank website (http://www.cu.co.kr/) uses an interesting defense code [7] to block clickjacking attacks. This code uses the property of `document.domain` which returns the domain name of the server that loaded the current document. This defense code seems secure at first glance, but can still be framed by an attacker who controls a domain with the substring "openbank.cu.co.kr" (e.g., kopenbank.cu.co.kr).

B. *Analysis for Global websites*

We tested the feasibility of clickjacking attacks on global websites; we used the top 500 global websites from the list on Alexa which has frequently been used for research purposes.

Fig. 3 (a) shows the types of these websites (here we categorized only 230 out of 500 websites according to the category information in Alexa (http://www.alexa.com/topsites/category). The majority of the websites; 77 out of 230 (about 34%) were categorized as being of the computers type. To analyze the characteristics of the website types, we measured the attack failure rates for these websites by type. Fig. 3 (b) shows these results. In Fig. 3 (b), 'Bus.', 'Reg.', 'Art.', 'Com.', 'Soc.', 'Ref.', 'Sho.' and 'Oth.' represent 'Business', 'Regional', 'Arts', 'Computers', 'Society', 'Reference', 'Shopping' and the other remaining websites, respectively. We observed that the 'Business' websites (11 out of 33, about 33.3%) were particularly secure against clickjacking attacks. This is because bank websites were included in the 'Business' category. Interestingly, however, most of the shopping websites (1 out of 11, about 9.1%) were vulnerable to clickjacking. This indicates that the shopping websites tended to be more vulnerable than the bank websites even though both types might be typical targets of clickjacking attacks. Another observation is that the 'Regional' (12 out of 49, about 24.5%) websites were more secure than the websites in the other categories except for 'Business'. This is because several major Internet service providers such as Google, Facebook, and Yahoo were categorized as 'Regional' in the list of Alexa. Similarly, the websites in 'Arts' (4 out of 19, about 21.1%) were also relatively secure since social media websites such as Facebook, YouTube, and Twitter were categorized as 'Arts'.

We also categorized the websites by country to show the differences between countries,. Fig. 4 (a) provides a pie graph showing the percent of countries. The majority of the websites (230 out of 500 (46%)) are in the United State. To analyze the characteristics of the websites in terms of their country, we measured the attack failure rates for the tested websites by country. Fig. 4 (b) shows the results. In Fig. 4 (b), 'RU', 'IT', 'DE', 'US', 'FR', 'IN', 'JP' and 'CN' represent 'Russia', 'Italy', 'Germany', the 'United States', 'France', 'India', 'Japan' and 'China', respectively. We observed that the websites in 'Russia' (8 out of 20, 40%) and 'Italy' (4 out of 10, 40%) were relatively secure against clickjacking attacks compared with those in the other countries. Interestingly, the Asian websites were particularly weak compared with the Western websites; in addition to the Korean websites (524 out of 526 websites, about 99.6%), most of the websites in 'India' (38 out of 44, 86.4%), 'Japan' (14 out of 16, 87.5%), and 'China' (61 out of 62, about 98.4%) were also vulnerable to clickjacking attacks. This implies that clickjacking attacks might be more popular for Western websites compared with Asian ones. We compared the Korean websites and websites in other countries with at least 20 websites. From Table I, it can be seen that the Russian websites are significantly more secure that those in the other countries (all $p < 0.05$, Fisher's exact test). In fact, the test results show that the security levels of all these countries are significantly different except for the comparisons between 'US and IN' and between 'CN and KR'. That is, there are significant gaps between the countries in preventing clickjacking even though this attack is already well known and its defense techniques have been intensively studied [3], [2].

We also found that companies with higher rewards for finding security bugs are more secure against clickjacking. Among the 500 global websites that we tested, 26 websites (5.2%) are owned by companies (Google, Facebook, Twitter, etc.) which run a bug bounty program. Among these 26 websites, 17 websites (about 65.4%) cannot be framed by our attacks. This result shows that websites having a bug bounty program are more secure than typical global websites ($p < 0.0001$, Fisher's exact test). This might provide evidence that running a bug bounty program can help companies enhance their security. A recent study [9] also supported the conclusion that running a bug bounty program is effective to mitigate web vulnerability.

TABLE I. COMPARISON RESULTS FOR THE NUMBER OF SECURE WEBSITES BETWEEN COUNTRIES BY USING THE FISHER'S EXACT TEST.

Countries	US (20%)	CN(1.6%)	IN (13.6%)	RU (40%)
CN (1.6%)	$p < 0.0001$			
IN (13.6%)	$p = 0.4062$	$p < 0.02$		
RU (40%)	$p < 0.05$	$p < 0.0001$	$p < 0.03$	
KR (0.4%)	$p < 0.0001$	$p = 0.2846$	$p < 0.0001$	$p < 0.0001$

IV. DISCUSSION

Clickjacking attacks and defenses are well known and have also been intensively studied [3], [2] since they were first reported almost five years ago [1]. Interestingly, however, almost all of the Korean websites (about 99.6%) are still vulnerable to these attacks.

This is probably because clickjacking attacks have not been popular in Korea. We can see that only a small number (14 out of 526 – about 2.6%) of websites have tried to prevent clickjacking attacks. This is because, so far, the main targets of clickjacking attacks have been highly concentrated. For example, there have been two kinds of widespread clickjacking attacks in the wild: Likejaking [10] and Tweetbomb [11]. Fortunately, Korean websites don't seem to be attractive targets yet for clickjackers.

A few Korean websites have used defense codes to detect framing and prevent it, but they are naively implemented and thus impractical. Unlike global websites such as Google, Facebook and Twitter, the Korean websites don't use code obfuscation techniques for their defense codes to make the codes themselves difficult to analyze. To make matters worse, almost all of frame busting codes used in the Korean websites can be easily bypassed with a simple HTML tag. Perhaps this is another example of "security theatre", which tackles the feeling but not the reality – vendors (or engineers) may demonstrate this perfunctory function to convince their customers (or managers) that their websites are really secure against clickjacking attacks.

We have to understand which defenses are practically effective. At the technical level, we recommend that Korean websites implement a secure code such as the frame busting script used by Citi Bank [7] and also use the `X-FRAME-OPTIONS` HTTP header. According to our observation, however, only two web browsers currently support the `ALLOW-FROM` feature (see Table II).

TABLE II. THE X-FRAME-OPTIONS HEADERS BASED ON THE WEB BROWSER.

Feature	Basic options support	`ALLOW-FROM` support
Chrome	4.1.249.1042	Not Supported
Fireforx	3.6.9	18.0
IE	8.0	8.0
Opera	10.5	Not Supported
Safari	4.0	Not Supported

In practice, it is not easy to trace new vulnerabilities in dynamically updated websites over time. The existence of a bounty program for security bugs seems helpful to address the clickjacking problem. For example, when a clickjacking vulnerability was found in Google Docs (http://docs.google.com) [12], it was fixed within two weeks. Many researchers claim that companies with higher rewards for finding bugs will become more secure and sustainable. Surely, our results confirmed this: about 78% of the global websites which are secure against clickjacking are owned by companies (Google, Facebook, Twitter, etc.) which run a bug bounty program.

V. CONCLUSION

We analyzed the feasibility of clickjacking attacks by testing the 500 most popular and all 36 financial institution websites in Korea (a total of 526 unique websites). Our experiments showed that 524 of these 526 websites (about 99.6%) were vulnerable to clickjacking attacks. Compared with the global websites (390 out of 500 websites – 78%), the Korean websites were significantly more vulnerable. We found that even those Korean websites with clickjacking defenses could be easily defeated with a simple HTML tag to disable JavaScript. We believe that these inter-country comparisons are important to define the security requirements and determine whether in fact it is necessary to invest in the next generation of web security technology.

Since clickjacking defense techniques are clearly understood and fairly easy to implement, the reason why Korean websites are still vulnerable to clickjacking seems quite clear: because Korean websites have been unattractive targets for clickjackers until now; only a small number (14 out of 526 – about 2.6%) of websites have tried to prevent clickjacking attacks but 12 of them failed in practice. However, we need to prepare for forthcoming attacks. Therefore, we recommend a secure implementation of frame busting codes and client side plugin (e.g., NoScript). Also, the introduction of a bounty program would be helpful to identify and fix security bugs such as clickjacking quickly.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (No. 2014R1A1A1003707), the ITRC (IITP-2015-H8501-15-1008, IITP-2015-R0992-15-1006), and the IITP (2014-044-072-003).

REFERENCES

[1] R. Hansen, "Clickjacking." [Online]. Available: http://ha.ckers.org/blog/20080915/clickjacking/

[2] L.-S. Huang, A. Moshchuk, H. J. Wang, S. Schechter, and C. Jackson, "Clickjacking: attacks and defenses," in *Proceedings of the 21st USENIX conference on Security symposium*, 2012, pp. 22–22.

[3] G. Rydstedt, E. Bursztein, D. Boneh, and C. Jackson, "Busting frame busting: a study of clickjacking vulnerabilities at popular sites," in *in IEEE Oakland Web 2.0 Security and Privacy (W2SP 2010)*, 2010.

[4] K. Kotowicz, "Cursorjacking again." [Online]. Available: http://blog.kotowicz.net/2012/01/cursorjacking-again.html

[5] E. Bordi, "Proof of concept - cursorjacking (noscript)." [Online]. Available: http://static.vulnerability.fr/noscript-cursorjacking.html

[6] I. Ristic, *Apache Security*. O'Reilly Media, 2005.

[7] D. Kim and H. Kim, "We are still vulnerable to clickjacking attacks: about 99% of korean websites are dangerous," in *The 14th International Workshop on Information Security Applications*, 2013. [Online]. Available: http://seclab.skku.edu/wp-content/uploads/2013/08/KoreanClickjacking_CR.pdf

[8] E. Lawrence, "Ie8 security part vii: Clickjacking defenses," 2009. [Online]. Available: http://blogs.msdn.com/b/ie/archive/2009/01/27/ie8-security-part-vii-clickjacking-defenses.aspx

[9] M. Finifter, D. Akhawe, and D. Wagner, "An empirical study of vulnerability rewards programs," in *Proceedings of the 22st USENIX conference on Security symposium*, 2013. [Online]. Available: http://www.cs.berkeley.edu/~daw/papers/vrp-use13.pdf

[10] SophosLabs, "Facebook worm - "likejacking"," 2010. [Online]. Available: http://nakedsecurity.sophos.com/2010/05/31/facebook-likejacking-worm/

[11] M. Mahemoff, "Explaining the "don't click" clickjacking tweetbom," 2009. [Online]. Available: http://softwareas.com/explaining-the-dont-click-clickjacking-tweetbomb

[12] M. Kumar, "Hacking google users with google's goopass phishing attack," 2013. [Online]. Available: http://thehackernews.com/2013/03/hacking-google-users-with-googles.html