Poster: CookieGuard: Isolating First Party Cookies using CookieGuard

1st Pouneh Nikkhah Bahrami
Department of Computer Science
University of California, Davis
Davis, USA
pnikkhah@ucdavis.edu

2nd Aurore Fass
CISPA Helmholtz
Center for Information Security
Sankt Ingbert, Germany
fass@cispa.de

3rd Zubair Shafiq
Department of Computer Science
University of California, Davis
Davis, USA
zshafiq@ucdavis.edu

***Abstract*—Web applications now heavily rely on third-party scripts to implement various features but this practice comes with serious risks, in particular if third-party scripts are included in the *main frame*. In this paper, we first investigate the indirect inclusion of additional third-party scripts which are engaged in cross-domain manipulation and exfiltration of cookies. To investigate this phenomenon, we designed and implemented a graph analysis tool based on PageGraph. We then use this tool to crawl the top 10,000 websites, to quantify the prevalence of direct and indirect third-party script inclusions, as well as cross-domain cookie manipulation and exfiltration by such scripts within the main frame. Our Results uncover significant cookie manipulation and exfiltration: cross-domain third-party scripts exfiltrate and manipulate cookies on almost half of the websites. Finally, we introduce a deployable intervention mechanism in the form of a browser extension. This extension can be effortlessly installed by users to protect cookie jar against unauthorized access.**

***Index Terms*—privacy, cookie, third-party, JavaScript**

I. Introduction

Modern websites heavily rely on third-party scripts to implement various features. However, third-party script inclusion introduces security and privacy risks [7]. To mitigate this risk, third-party scripts should be properly isolated, e.g., put in a separate iframe. When a third-party script is isolated in a separate iframe, the same-origin policy (SOP) would restrict the interactions between the script and resources in the main frame (or other frames) except via a narrow set of cross-origin communication mechanisms such as postMessage and cross-origin resource sharing (CORS).

However, developers often include different third-party scripts directly in the main frame for simplicity [6]. This means that the SOP and other origin- or frame-based security mechanisms do not apply. Thus, when a third-party script is embedded in the main frame (either directly by a developer or indirectly by other scripts), it has access to the shared resources of the main frame such as the cookie jar containing first-party cookies.

In this paper, we investigate privacy implications of third-party script inclusion in the main frame. Specifically, we focus on two types of interactions between third-party scripts in the main frame: (1) indirect injection of third-party scripts and (2) cross-domain storage manipulation and exfiltration of cookies in the cookie jar which is a shared resource between all entities in the main frame. We then propose an intervention mechanism to protect cookie jar from unauthorized scripts to prevent cross-domain storage manipulation and exfiltration of cookies.

While some work has studied the prevalence and impact of third-party script inclusion [5], [7], also showing that dynamically loaded third-party resources are disproportionately associated with advertising and tracking [3], they did not consider the privacy implications of cross-domain script interactions in the *main frame*. Another line of work uncovered JavaScript global identifier conflicts that arise when different scripts are included in the same frame [9], but they did not consider privacy implications of such conflicts.

While the exfiltration of cookies to other entities has been studied [2], [8], to the best of our knowledge, no existing interventions effectively prevent cookie exfiltration and manipulation from unauthorized access. Our proposed mechanism fills this gap by providing robust protection for cookies within the main frame's shared resources.

II. Measurement

We design and implement a graph based tool called PageGraph++ to detect and quantify the prevalence of cross-domain script manipulations and exfiltration of cookies in the main frame. We built this tool on top of PageGraph [1], an instrumented version of Chromium that can capture fine-grained page execution behavior. PageGraph++ includes a query tool that leverages the resulting graph structure to analyze web pages and capture dynamic script injection and shared resources (such as cookie) manipulation.

III. Measurement Study finding

PageGraph++'s deployment on top-10k websites [4] shows two key findings. First, we find that more than 90% of the websites injects at least one third-party script indirectly. Furthermore, 75% of these scripts are advertising or tracking scripts. Second, we find worrisome cross-domain storage manipulation by third-party scripts in the main frame. Cookies are overwritten by cross-domain scripts on nearly a third of the websites by 250 different third-party scripts. Cookies are exfiltrated by cross-domain scripts on almost half of the websites.

IV. Development of the Prevention Tool

To protect cookies against unauthorized access, we implement a browser extension called CookieGuard. CookieGuard's core principle is to ensure that scripts can only interact with cookies they have set themselves, thereby preventing unauthorized access and manipulation. This approach is predicated on intercepting and controlling both the setting and accessing of cookies by scripts running within web pages.

A. Features

CookiesGuard has three main features designed to enhance cookie jar protecting.

- Script Cookie Access Interception: This feature actively monitors access to *document.cookie*, capturing both "get" and "set" operations initiated by scripts on the web-page.
- Monitor set-cookie header in HTTP responses: This functionality investigates all HTTP responses that include a Set-Cookie header, ensuring comprehensive oversight of cookie setting via HTTP.
- Book keeping: Implements detailed logging for all first-party cookies, covering both first-party HTTP and JavaScript-generated cookies. Specifically, this includes maintaining a log of all first-party cookie names along with their corresponding setter domains.

B. components and Implementation

We implement CookieGuard as a browser extension. The extension has three main components.

- background.js: This component monitors HTTP responses for Set-Cookie headers and logs non-HTTPOnly, first-party cookies by name and setter domain in a dataset stored within the extension. It also manages communications with the content script, updating the dataset with cookies set via the *document.cookie* "set" function, and retrieving the dataset for the "get" function calls.
- contentScript.js: This component injects a script (cookieGuard.js) into the webpage. Forwards messages between the cookieGuard.js and the background.js.
- cookieGuard.js: Implements a wrapper around the *document.cookie* "get" and "set" functions to intercept cookie access. It ensures that when cookies are set, details are sent to background.js to update the dataset, and when cookies are fetched, only those set by the specific script are returned, preventing scripts from accessing cookies set by others.

V. Evaluation of CookieGuard

We do a manual analysis of CookieGuard to evaluate the web breakage while using this extension. We select 10 websites and evaluate them in two different configuration; with and without CookieGuard extension. We classify breakage into four categories: navigation (moving between pages), SSO (initiating and maintaining login state), appearance (visual consistency), and miscellaneous (such as chats, search, and shopping cart). Breakage is labeled as either major or minor for each category: Minor breakage occurs when it is difficult but not impossible to use the functionality. Major breakage occurs when it is impossible to use the functionality on a webpage. The result is shown in Table I. While there was no impact on Navigation and Appearance (0% breakage), we observed minor SSO disruptions on cnn.com and major functionality breakages on facebook.com due to our extension. These issues stem from the dependency on third-party cookies for session management and login processes. Specifically, Facebook's Messenger service was significantly affected because it relies on fbcdn.com within facebook.com. To mitigate such disruptions, implementing whitelists where all company-related domains are treated as first-party resolve the issue.

	Navigation	**SSO**	**Appearance**	**Miscellaneous**
Minor	0%	1%	0%	0%
Major	0%	0%	0%	1%

TABLE I
Breakage Summary

VI. Conclusion

Our results shed light into the privacy implications of including third-party scripts in the main frame. Web developers need to be more mindful of the indirectly included scripts that they implicitly trust. Our work also informs the deployment of targeted countermeasures by web developers, such as CSP, to limit the indirect inclusion of third-party scripts. Our extension enhances web privacy by implementing an access control system for the browser's cookie jar. It is easly deployable by all users.

References

[1] Brave software, pagegraph. https://github.com/brave/brave-browser/wiki/PageGraph, 2020.

[2] Chen, Q., Ilia, P., Polychronakis, M., and Kapravelos, A. Cookie swap party: Abusing first-party cookies for web tracking. In *Proceedings of the Web Conference 2021* (2021), pp. 2117–2129.

[3] Ikram, M., Masood, R., Tyson, G., Kaafar, M. A., Loizon, N., and Ensafi, R. Measuring and analysing the chain of implicit trust: A study of third-party resources loading. *ACM Transactions on Privacy and Security (TOPS) 23*, 2 (2020), 1–27.

[4] Le Pochat, V., Van Goethem, T., Tajalizadehkhoob, S., Korczyński, M., and Joosen, W. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Proceedings of the 26th Annual Network and Distributed System Security Symposium* (Feb. 2019), NDSS 2019.

[5] Lekies, S., Stock, B., Wentzel, M., and Johns, M. The unexpected dangers of dynamic {JavaScript}. In *24th USENIX Security Symposium (USENIX Security 15)* (2015), pp. 723–735.

[6] Luo, W., Ding, X., Wu, P., Zhang, X., Shen, Q., and Wu, Z. Scriptchecker: To tame third-party script execution with task capabilities.

[7] Nikiforakis, N., Invernizzi, L., Kapravelos, A., Van Acker, S., Joosen, W., Kruegel, C., Piessens, F., and Vigna, G. You are what you include: large-scale evaluation of remote javascript inclusions. In *Proceedings of the 2012 ACM conference on Computer and communications security* (2012), pp. 736–747.

[8] Sanchez-Rola, I., Dell'Amico, M., Balzarotti, D., Vervier, P.-A., and Bilge, L. Journey to the center of the cookie ecosystem: Unraveling actors' roles and relationships. In *2021 IEEE Symposium on Security and Privacy (SP)* (2021), IEEE, pp. 1990–2004.

[9] Zhang, M., and Meng, W. Detecting and understanding javascript global identifier conflicts on the web. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (2020), pp. 38–49.