A case study of a privacy-invading browser extension

Robin Carlsson, Sampsa Rauti, and Timi Heino

University of Turku, Turku, Finland
{crcarl,sjprau,tdhein}@utu.fi

Abstract. Today, many everyday tasks are carried out online, and traditional desktop applications are also being replaced by web applications. Consequently, the web browser has become an important execution environment. The security of the browser environment, however, has not fully kept up with this recent development. One notable threat to online privacy is found inside the user's own browser in the form of a malicious browser extension. To demonstrate how easy it is to develop such a privacy-invading extension, we build a platform-independent proof-of-concept implementation that records user's actions and data inside the web browser. We conclude that despite the preventive measures taken by browser vendors, malicious extensions are still a significant threat to users' privacy.

Keywords: Online privacy · Web browser security · Malicious browser extensions

1 Introduction

Nowadays, many everyday tasks and services are increasingly moving online. The COVID-19 pandemic has further expedited this development in recent years, moving both companies and consumers toward online channels [2]. At the same time, traditional desktop applications are also being replaced by web applications due to low costs, easy installation and maintenance, good uptime and high scalability. While the browser as an execution environment offers many benefits, it can be argued that the security of this environment has not fully kept up with this recent development. One significant threat to online security and privacy resides inside web browsers in the form of malicious browser extensions [12, 15].

Browser extensions are pieces of software that allow the customization of a web browser. These extensions offer a variety of different functionalities such as custom scripting and style modifications of web pages, altering user interfaces, and blocking advertisements. While tailoring browser functionalities according to the user's needs may be extremely helpful, users rarely fully realize how powerful browser extensions are and what kind of security and privacy risks may be involved when installing them [3, 19, 13].

Malicious browser extensions have been used for instance to modify and steal users' data, insert additional advertisements to web pages, redirecting users to

malicious websites, and earning money by generating fake clicks for advertisement campaigns. Browser vendors have taken a number of preventive measures to protect users from malicious extensions. For example, Google Chrome does not allow users to install extensions from web pages other than the Chrome Web Store, and extensions submitted to the store are vetted in automatic and manual analysis. Still, attackers regularly manage to slip their malicious extensions into the store or use different methods to circumvent the normal installation process [7]. The fact that the current browsers do not sufficiently monitor and restrict extensions' functionality does not help.

In this paper, we focus on online privacy and data-stealing extensions. More specifically, we demonstrate how easy it is to develop a privacy-invading extension for modern-day browsers. Our proof-of-concept implementation records the user's actions and steals any data they input inside the web browser. Compared to many data-stealing extensions found in the wild, our implementation shows how the users data and actions can be more comprehensively recorded only with minimal permissions, and using only browser-independent functionality such as basic JavaScript and DOM API. This study also provides an update on malicious browser extensions in 2022, highlighting that it is still necessary to further improve the existing countermeasures against malicious extensions.

The rest of the paper is organized as follows. Section 2 describes the attack scenario this study focuses on – a browser extension collecting the user's private information when they browse the internet. In Section 3, we demonstrate this scenario by building a proof-of-concept implementation of a privacy-invading browser extension. Section 4 discusses the key findings and practical implications of the study. Finally, Section 5 concludes the paper.

2 The Attack Scenario

Because encrypted HTTPS connections are used in almost all web services, attackers have in many cases moved their man-in-the-middle attacks, in which the users' data is being spied on or modified, to the connection endpoints. In other words, man-in-the-middle attacks have often transformed into man-at-the-end attacks [1], or more specifically, man-in-the-browser attacks taking place inside the web browser [8, 4, 16]. This way, the adversary does not have to find a way to access the data when it is in transit. Instead, they can alter or spy on it while it is displayed to the user or processed by the user inside the browser, in a web application's user interface. Browser extensions offer an easy way to implement this type of attack.

When employing a browser extension, a malicious adversary can collect sensitive data with almost no effort at all, compared to many other attack methods. It is no longer necessary to break encryption, intercept the web connection on the network level, or lure the victim to a scam website. The attacker is simply observing as the user interacts with the real web service and they do not need to break, intercept or fake anything [8]. The attack is also invisible to the user and the web service being used – as long as they do not notice the extra code

the malicious extension injects into the webpage, they cannot detect anything suspicious. Several traditional layers of protection usually used by web services, such as TLS encryption and authentication to web services, are completely circumvented as our malicious extension operates on the UI layer.

Our goal in this study is to implement a browser extension that records all user interaction with web pages. This includes all visited webpages, user inputs and interactions with pages. More specifically, our proof-of-concept extension includes the following features:

- *Log keystrokes.* The extension acts as a keylogger, recording all the keystrokes on the websites in a log, which enables the attacker to see the data the user has input on any given page. This, for instance, includes any user names and passwords the user types in text fields.
- *Record mouse movements and elements the user interacts with.* The information about the user's mouse movements and HTML elements the user interacts with complement the data collected keystroke data. For example, it is much easier to see which keystrokes are part of a password when there are log entries of text fields the user has interacted with, along with timestamps. It also gives the attacker a good sense of user actions that do not involve typing, such as hitting a like button.
- *Record visited websites.* The extension keeps track of the current URL. This information is collected so that the keystrokes and interactions with HTML elements can be connected to a specific website. Moreover, URL addresses alone can reveal private user data especially if the address includes parameters [6].

A browser extension gets access to user data by injecting a content script into a web page. Running in the context of web pages and using the Document Object Model (DOM), content scripts can read the data on the web pages accessed by the user and deliver it to their parent extension. In what follows, we will take a more detailed look at the implementation.

3 Implementation

Our proof-of-concept implementation demonstrates how privacy-invading malware can be implemented as an extension for the Google Chrome web browser. The malicious extension is implemented by attaching event listeners to window and document objects with DOM. The event listeners are used to capture keystrokes, record mouse movements and focused elements, and log webpages associated with the detected events.

While we will not share the potentially harmful content script, it consists of very basic JavaScript and DOM functions, only requiring a couple of lines of code for each malicious functionality. It is worth noting, however, that anyone with a basic knowledge of JavaScript and DOM could develop such extension in just a few hours. The source code for the extension is available upon request for research purposes.

The browser extensions are normally only acquired from the Chrome Web Store. However, the developer can also test extensions by enabling Chrome's developer mode. Such unpacked browser extension simply consists of two separate files: a manifest (manifest.json) and a content script (content.js). The latter contains the harmful functionality described in the previous section. As can be seen by looking at the following manifest file, our proof-of-concept extension does not need to define any explicit permissions in order to function:

```
{

  "manifest_version": 3,
  "name": "Data Spy POC",
  "version": "1.0",
  "description": "Tracks pages visited,
                  keystrokes, mouse
                  movements and focused
                  elements",

  "content_scripts": [
    {
      "matches": ["<all_urls>"],
      "js": ["content.js"]
    }
  ]

}
```

It does, however, specify the content script which runs on all visited websites, as indicated by the `<all_urls>` pattern in the manifest. On installation, the user is prompted to allow the extension to "read and change all your data on the websites you visit". This, of course, should be alarming to the user, but many other extensions also require this functionality, and the user may just routinely comply without fully appreciating the meaning and implications of this permission. Furthermore, if the extension is installed using Chrome's developer mode, the prompt to accept reading and changing data is not shown at all. It is also worth noting that the data collected by the malicious extension can be sent to the attacker without any additional permissions.

In the manifest file, the name and description of the extension, which are shown to the user on the web browser's extension page would naturally be changed in a real-world malware. Malicious browser extensions can often be classified as Trojans [8, 17]: they deceive the user by containing useful functionality, while also stealthily performing malicious operations in the background.

Figure 1 shows an excerpt of a log collected by the malicious extension. The excerpt includes mouse coordinates (the circles with numbers indicate that the

Fig. 1. A sample log collected by the proof-of-concept extension.

same coordinates have been recorded several times consecutively), any elements the user has interacted with and the keys the user has pressed. The attacker can then easily search this log for any inputs the user has given on different websites make observations about the user's interaction with various HTML elements.

While our extension was implemented for and tested on Google Chrome, it does not use any browser specific interfaces and could be used on Edge and Opera with the same malicious code. Firefox does not support extensions with manifest version 3 required by our extension yet, but is expected to do so by the end of 2022.

4 Discussion

Our proof-of-concept extension demonstrates that malicious extensions are still a significant threat for users' online privacy, even after browser extensions have been around for over 20 years and despite many preventive measures taken by browser vendors. The implemented extension can be used to stealthily collect the user's data inside the browser [5, 18], with can cause significant harm especially when it comes to the most critical web services such as online banking systems and patient data repositories.

As we have seen, a malicious extension is very easy to write. Only some basic JavaScript and rudimentary functions of the DOM interface are needed. Of course, an attacker does not have to write a data-stealing extension, as it would be easy to buy it as a ready-to-use solution. The attacker can easily search the collected data for a recording of user activities on a specific webpage, or simply find a password for a given web service. Some sensitive data fitting a specific

format, such as social security numbers, can also be found using regular expressions. The same data-stealing extension can be used for many different purposes and provides possibilities for a diverse group of privacy attacks. Depending on the attacker's motives, the collected data could be used to break into user's online accounts to steal the user's identity or to blackmail them, or simply be sold for a good price.

Google has been working to improve the process of reviewing Chrome extensions, making it more accurate and rigorous [14]. Moreover, users can no longer install extensions from websites other than Chrome Web Store. These countermeasures, however, have not completely prevented malware developers from foisting their harmful extensions into Chrome Web Store [9, 10]. With almost 70% market share, Google Chrome is currently an attractive platform for malware authors. Many adversaries have also used the tactic of first releasing a benign extension and then later adding malicious functionality. The attackers are also known to have circumvented Chrome's restrictions by installing their extension programmatically [11] with the help of another piece of malware. Last but not least, the adversary can use social engineering to coax the user into installing the malicious extension in Chrome's developer mode.

While Google has made installing malicious extension harder in Chrome, these efforts have not been enough to stop malicious browser extensions. Many other browsers have even more lax extension policies than Google Chrome. The JavaScript code of our proof-of-concept extension does not use any browser specific features. Therefore, it is browser-independent and could be effortlessly reused for malicious Edge and Opera extensions, for instance. The easiness of implementation and support across different browsers and platforms make browser extensions a very easy and tempting method to implement privacy-invading malware.

5 Conclusion

We have presented a case study on a privacy-invading browser extension. The implemented extension highlights how easily the malicious code can be written and how effortlessly users' privacy can be compromised despite preventive measures taken by browser vendors. Other than completely disallowing the use of browser extensions, there is no single silver bullet that would completely thwart the attacks employing privacy-invading extensions. Combining several security measures, enforcing strict security policies in modern web browsers, and educating users can considerably mitigate the problem in the future. However, achieving this objective is likely to require common efforts of browser manufacturers, users, employers and antivirus program vendors.

Acknowledgements

This research has been funded by Academy of Finland project 327397, IDA – Intimacy in Data-Driven Culture.

References

1. Akhunzada, A., Sookhak, M., Anuar, N.B., Gani, A., Ahmed, E., Shiraz, M., Furnell, S., Hayat, A., Khan, M.K.: Man-At-The-End attacks: Analysis, taxonomy, human aspects, motivation and future directions. Journal of Network and Computer Applications **48**, 44–57 (2015)
2. Amankwah-Amoah, J., Khan, Z., Wood, G., Knight, G.: Covid-19 and digitalization: The great acceleration. Journal of Business Research **136**, 602–611 (2021)
3. DeKoven, L.F., Savage, S., Voelker, G.M., Leontiadis, N.: Malicious browser extensions at scale: Bridging the observability gap between web site and browser. In: 10th USENIX Workshop on Cyber Security Experimentation and Test (CSET 17) (2017)
4. Dougan, T., Curran, K.: Man in the browser attacks. International Journal of Ambient Computing and Intelligence (IJACI) **4**(1), 29–39 (2012)
5. Eisen, O.: Catching the fraudulent man-in-the-middle and man-in-the-browser. Network Security **2010**(4), 11–12 (2010)
6. Goodin, D.: My browser, the spy: How extensions slurped up browsing histories from 4m users (2019), https://arstechnica.com/information-technology/2019/07/dataspii-inside-the-debacle-that-dished-private-data-from-apple-tesla-blue-origin-and-4m-people/
7. Goodin, D.: 500 chrome extensions secretly uploaded private data from millions of users (2020), https://arstechnica.com/information-technology/2020/02/500-chrome-extensions-secretly-uploaded-private-data-from-millions-of-users/
8. Gühring, P.: Concepts against man-in-the-browser attacks. Technical Report. (2006)
9. Jagpal, N., Dingle, E., Gravel, J.P., Mavrommatis, P., Provos, N., Rajab, M.A., Thomas, K.: Trends and lessons from three years fighting malicious extensions. In: 24th USENIX Security Symposium (USENIX Security 15). pp. 579–593 (2015)
10. Kapravelos, A., Grier, C., Chachra, N., Kruegel, C., Vigna, G., Paxson, V.: Hulk: Eliciting malicious behavior in browser extensions. In: 23rd USENIX Security Symposium (USENIX Security 14). pp. 641–654 (2014)
11. Marinho, R.: "Catch-All" Google Chrome Malicious Extension Steals All Posted Data. https://morphuslabs.com/catch-all-google-chrome-malicious-extension-steals-all-posted-data-f2472e272101 (2017)
12. Pantelaios, N., Nikiforakis, N., Kapravelos, A.: You've changed: Detecting malicious browser extensions through their update deltas. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. pp. 477–491 (2020)
13. Picazo-Sanchez, P., Tapiador, J., Schneider, G.: After you, please: browser extensions order attacks and countermeasures. International Journal of Information Security **19**(6), 623–638 (2020)
14. Protalinski, E.: Google updates Chrome Web Store review process and sets new extension code requirements. https://venturebeat.com/2018/06/12/google-disables-inline-installation-for-chrome-extensions/ (2018)
15. Rauti, S.: Man-in-the-browser attack: a case study on malicious browser extensions. In: International Symposium on Security in Computing and Communication. pp. 60–71. Springer (2019)
16. Rauti, S., Leppänen, V.: Browser extension-based man-in-the-browser attacks against ajax applications with countermeasures. In: proceedings of the 13th international conference on computer systems and technologies. pp. 251–258 (2012)

17. Toreini, E., Shahandashti, S.F., Mehrnezhad, M., Hao, F.: Domtegrity: ensuring web page integrity against malicious browser extensions. International Journal of Information Security **18**(6), 801–814 (2019)
18. Utakrit, N.: Review of browser extensions, a man-in-the-browser phishing techniques targeting bank customers (2009)
19. Varshney, G., Bagade, S., Sinha, S.: Malicious browser extensions: A growing threat: A case study on google chrome: Ongoing work in progress. In: 2018 International Conference on Information Networking (ICOIN). pp. 188–193. IEEE (2018)