

Data Exfiltration by Hotjar Revisited

Libor Polčák^a and Alexandra Slezáková

Brno University of Technology, Faculty of Information Technology, Božetěchova 2, 612 66 Brno, Czech Republic

Keywords: Web Privacy, Session Replay, Data Protection.

Abstract: Session replay scripts allow website owners to record the interaction of each web site visitor and aggregate the interaction to reveal the interests and problems of the visitors. However, previous research identified such techniques as privacy intrusive. This position paper updates the information on data collection by Hotjar. It revisits the previous findings to detect and describe the changes. The default policy to gather inputs changed; the recording script gathers only information from explicitly allowed input elements. Nevertheless, Hotjar does record content reflecting users' behaviour outside input HTML elements. Even though we propose changes that would prevent the leakage of the reflected content, we argue that such changes will most likely not appear in practice. The paper discusses improvements in handling TLS. Not only do web page operators interact with Hotjar through encrypted connections, but Hotjar scripts do not work on sites not protected by TLS. Hotjar respects the Do Not Track signal; however, users need to connect to Hotjar even in the presence of the Do Not Track setting. Worse, malicious web operators can trick Hotjar into recording sessions of users with the active Do Not Track setting. Finally, we propose and motivate the extension of GDPR Art. 25 obligations to processors.

1 INTRODUCTION

Website operators want to monitor the interaction of the visitors of the websites, for example, to detect problems. Session recording tools allow detailed monitoring of user behaviour (Filip and Čegan, 2019; Acar et al., 2020; Grodzinsky et al., 2022). Consequently, session recording scripts are widespread even though users are typically unaware of the detailed monitoring of their behaviour (Acar et al., 2020; Grodzinsky et al., 2022). Session recording scripts record users' mouse movements, clicks, and keystrokes. The recording script sends the collected data to servers of the recording party, which provide aggregated statistics in the form of heat maps but also entire recordings of unique users. Recordings often contain sensitive user data, such as medical conditions, credit card information, and other data that can subsequently be misused (Acar et al., 2020).

(Acar et al., 2020) analysed six session recording companies in 2018 and revealed password, credit card, and health data leaks. Similarly to (Grodzinsky et al., 2022) that revisited one of the companies, FullStory, this position paper revisits and updates the results of the study of Acar et al. This paper focuses


on Hotjar as it is one of the most popular replay service companies in the market. Acar et al. detected Hotjar on many sites. Even so, Acar et al. do not list any change in the data collection of Hotjar.

In 2018, Hotjar collected (1) texts typed into forms before the user submitted the form and (2) precise mouse movements. Both without any visual indication to the user (Acar et al., 2020, Section 6.1). At the time of the study, Hotjar collected the text inputs verbatim by default, except for passwords, credit card numbers, and partially addresses. Moreover, (Acar et al., 2020) detected session replay companies that transfer the content of HTTPS websites over HTTP, not protected by any encryption.

This position paper revisits Hotjar in 2023 and reveals that Hotjar's default behaviour changed. The content of all inputs is no longer collected by default. Hotjar changed its policy to collect only inputs marked with `data-hj-allow` attribute. (Acar et al., 2020) argued that the opt-out from data collection is not practical. Hotjar likely accepted the arguments.

(Acar et al., 2020) also warned that the displayed content on a page could reflect user-specific content¹.

¹Reflected information is any information initially gathered from a user that is subsequently propagated to the DOM or HTML at a different position.

^a  <https://orcid.org/0000-0001-9177-3073>

For example, online stores typically show the content of the user basket and the filled billing information as a part of the page just before checkout. Our results show that Hotjar's scripts gather the content of all elements except inputs by default, so the user-specific information leaks the reflected information.

The general shift to encrypted HTTP connections motivated Hotjar to operate through HTTPS. Moreover, Hotjar refuses to work on HTTP sites. Even more, we show that Hotjar respects the do not track settings signalled by the browser. However, we also tested that malicious sites can force Hotjar to record users who opt out of tracking.

(Acar et al., 2020) observed that by bringing attention to the flaws during their research, the companies operating data exfiltration services improved the data collection practices. As we highlight the unsolved problem of collecting personal data reflected outside input elements, we hope data-collecting companies like Hotjar will address the problem.

This paper is organised as follows. Section 2 provides the necessary theoretical background on session replays, heat maps, options to avoid being recorded, and other related work. Section 3 describes the methodology used to analyse Hotjar with achieved results in section 4. Section 5 discusses our findings and proposes to extend the obligations of data protection by design and default to processors. The paper is concluded in section 6.

2 BACKGROUND AND RELATED WORK

A *web session* is a series of requests and responses between a web server. Several techniques exist for maintaining a session state, but sessions are often bound to a cookie (Bortz et al., 2011). Cookies consist of data stored in the browser that is sent with every request to the server and can be modified with each response. Web pages track sessions by identifiers in cookies (Bortz et al., 2011).

Session recordings are usually used by web operators with the primary goal of a better understanding of user behaviour (Filip and Čegan, 2019). Consequently, web operators can improve the user experience. Web operators can replay the users' interaction with the website as a playback. Such playback can be analysed and annotated by web operators. For example, the operators can identify sessions during which users were confused or frustrated with a web page, analyse the behaviour, and improve the page.

2.1 Heat Maps

A heat map visualises users' interaction with the website. It uses colour intensity to demonstrate the clicks. The brightness of a heat map reflects how popular a particular website section is. Without displaying the data numerically, a heat map provides quick and simple-to-understand visuals (Kaur and Singh, 2015), facilitating the analysis and better understanding of how the user interacts with a website.

Click heat maps visualise areas where the user clicks the most, as shown in Figure 1. Click heat maps are derived from click coordinates, target element, device resolution, and web page content. This information reveals where users click and which element looks clickable (Filip and Čegan, 2019).

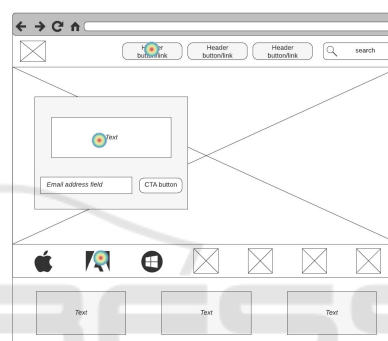


Figure 1: Click map.

Mouse-tracking heat maps, shown in Figure 2, are similar to click maps, but they visualise areas where users hover the most.



Figure 2: Mouse-tracking heat map.

Scroll heat maps visually represent the user's scrolling behaviour. They provide information on how far a user scrolls down on a website, see Figure 3.

2.2 Session Replay Scripts

A session replay script records the user's interaction with websites or applications and sends the recordings to the server, where it is processed and converted to

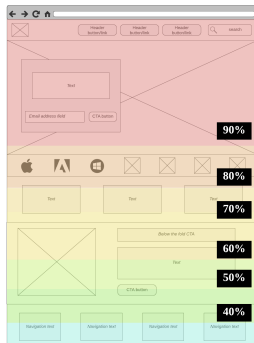


Figure 3: Scroll map.

a format that can be replayed. Additionally, the server aggregates the recordings and generates heat maps.

Session recording companies usually offer session replays and heat maps as a service deployed by many websites. Both session replays and heat maps depend on gathering data such as the content of the whole page, displayed text, mouse movements and clicks, and key presses. As the user’s device is identified by the generated tracking ID stored in a cookie, web operators can monitor user behaviour over a long period (Filip and Čegan, 2019).

Web operators can install session replay scripts easily. Session recording companies market themselves as being easy to deploy (Acar et al., 2020). For instance, Hotjar provides the instructions depicted in Figure 4. This code inserts another script into the website. The page environment is initialised to set up the data gathering.

However, users are not typically aware of the detailed monitoring of their activities (Acar et al., 2020; Grodzinsky et al., 2022). Such data collection practice has been particularly problematic in cases involving sensitive data unless the application developer has manually redacted the website or application to ensure the user’s privacy (Acar et al., 2020). Gathered data can be analysed to derive heat maps, recordings of mouse movements, or the success of filling forms.

2.3 Privacy Impact Analysis

Several studies have already analysed the impact of the data exfiltration from web pages with recording scripts. Passwords, credit card numbers, and health data do leak to third parties, often without any awareness by the website owner (Acar et al., 2020; Senol et al., 2022). Pages like contact forms and registration and authentication forms leak data that third parties can use to de-pseudonymise web site users (Starov et al., 2016; Chatzimpyrros et al., 2020; Dao and Fukuda, 2021). Companies perform such de-pseudonymisation, for example, to track users on dif-

ferent devices (Brookman et al., 2017) as such identifiers are stable and persistent (Senol et al., 2022; Dao and Fukuda, 2021).

Worse, session replay scripts and other trackers that leak private information are omnipresent (Acar et al., 2020). At least some of the session replay script vendors leave the responsibility to configure web sites to their customers (Grodzinsky et al., 2022). Nevertheless, previous research identified that web sites leak private information. For example, (Krishnamurthy and Wills, 2011) reported that 56 % of sites leak private information, and (Dao and Fukuda, 2021) discovered that 42.3 % of sites leak authentication data to third parties. Other research reports a lower share of leaking sites (Chatzimpyrros et al., 2020).

Previous research shows that trackers are less frequent in the EU compared to the US (Senol et al., 2022). Studies of privacy notices show that they are not clear (Dao and Fukuda, 2021; Brookman et al., 2017; Chatzimpyrros et al., 2020).

Although (Grodzinsky et al., 2022) considered malicious web operators from the ethical and legal point of view, to our best knowledge, we are the first to study technical ways that mislead the collectors to gather data that are not recorded by the unmodified session replay script.

2.4 Ways to Avoid Tracking

Tracker blockers employ lists of URLs or parts of URLs that are considered harmful to user privacy or security. The advantage for the user is that many tools focus on blocking (for example, uBlock Origin², EFF Privacy Badger³, Ghostery⁴) and block-lists are usually compatible with several blockers. Browsers like Firefox (Kontaxis and Chew, 2015) and Brave include tracking prevention by default. Previous research shows that by blocking trackers, users improve the performance of their browsers (Kontaxis and Chew, 2015). The downside of the list-based blockers is that blockers can evade detection by changing the script URL so that the rule in the block list no longer matches the URL of the tracker (Merzdovnik et al., 2017). Worse, block lists contain only previously detected trackers and, by definition, cannot contain tracking domains in advance. Consequently, researchers find tracking domains that are not a part of any popular block list (Senol et al., 2022).

Do Not Track (DNT) is a web browser setting that allows users to signal opt-out from tracking. DNT aims to enable users to communicate their tracking

²<https://github.com/gorhill/uBlock#ublock-origin>

³<https://privacybadger.org/>

⁴<https://www.ghostery.com/>

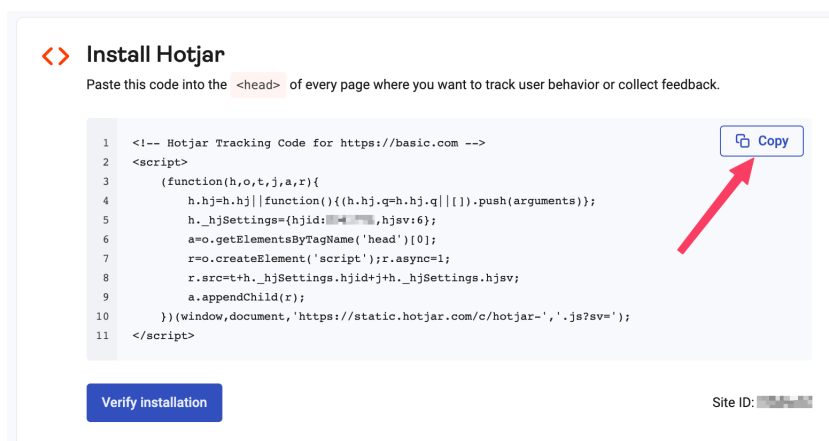


Figure 4: Hotjar installation script.

preferences to each server. A user's web browser sends an HTTP header called DNT. Once a server receives the DNT: 1 header, it should stop tracking the user. DNT: 0 means that the user prefers to allow tracking. DNT also allows access to the preference via JavaScript API `navigator.doNotTrack`. DNT was supposed to become a W3C standard (W3C, 2019). Nevertheless, websites do not generally respect the signal (Hils et al., 2021), and the standard is retired due to a lack of enforcement and minor effects. Even so, browsers still allow users to activate DNT.

DNT signal received a little success (Hils et al., 2021). (Acar et al., 2020) did not study if Hotjar and other session collecting companies respect DNT. Moreover, (Grodzinsky et al., 2022) claim that users typically do not have the option of opting out. Nevertheless, Hotjar mentions respect for DNT in its privacy statements. We validated the code and confirmed that Hotjar indeed respects DNT. Even so, we warn that malicious website operators or attackers can circumvent the DNT detection and force Hotjar to collect data from users signalling their no-tracking preferences.

3 METHODOLOGY

We reproduce the tests mentioned in the original study (Acar et al., 2020) and determine if something has changed in Hotjar to ensure users' privacy.

Firstly, we studied Hotjar policies and documentation and opened a testing account. We validated that Hotjar respects the Do Not Track signal (Section 4.1). To do so, we read the Hotjar script and located the code responsible for Do Not Track handling. Next, we checked if the interaction with the testing account was encrypted (Section 4.2).

Secondly, we analysed the nature of collected user data with a focus on dynamically generated data by both the user and the page as a result of the users' interaction with the page (Section 4.3).

In parallel, we investigated the possibility of the web server operators modifying the Hotjar script to increase the amount of collected data (Sections 4.1–4.3).

We created three web applications:

1. A log-in form with a password input and text input for email username. As many log-in forms allow a user to show the verbatim password, we included a button that changes the input type of the password input to text. The goal was to (1) compare data collected from password inputs to other input types and (2) validate that passwords are treated consistently if the user decides to peak into the field.
2. A shipping form where a user needs to fill in name, address, email, phone, credit card details, and other information that is typically needed for shipping. The goal was to study other input types. Hence, the page reflects the content of an input element in other elements.
3. A page that changes the content according to the user input. The page contains a text input. The page offers country names matching the input text as the user types text in.

We created the log-in and shipping form in three varieties:

1. The first scenario follows Hotjar's documentation. ID attributes of both input fields were set to the appropriate values: `pass` or `password` for the password input field and `email` for the email input field. According to documentation, Hotjar should not record such input fields. Instead, it should

replace their content with asterisks of arbitrary length. The shipping form also employs specific ID attributes to prevent the potential leakage of sensitive personal information.

2. Even though Hotjar does not record keystrokes by default, the web operator can allow collecting the content of any input element by adding attribute `data-hj-allow`. Consequently, Hotjar records the content that the user inserts into such fields verbatim and does not replace it with asterisks. The form did not use any well-known ID for the inputs. Instead, we chose IDs randomly.
3. The two scenarios above use the original Hotjar session recording script. The third scenario is the same as the first scenario, but we modified the Hotjar script. For example, we modified the function that masks the username and password. The original function replaces the real text with a random number of asterisks. The modified version does not transform the original string in any way.

Finally, we tested the default configuration of multiple browsers and validated if the built-in protections prevent pages from recording the sessions, see section 4.4.

4 RESULTS

This section provides the results of the experiments designed in Section 3.

4.1 Respecting DNT

Unlike websites that refuse to respect DNT signals (Hils et al., 2021), Hotjar script reads `navigator.doNotTrack` property, see Figure 5. When the property carries the value 1, the recording of a particular session is omitted. Nevertheless, browsers with DNT activated still need to download and execute the Hotjar script as the check is not part of the code inserted directly to the web page using Hotjar services (the code in Figure 4).

```
"1"!==navigator.doNotTrack&&
"1"!==window.doNotTrack&&
"1"!==navigator.msDoNotTrack
```

Figure 5: The script at <https://script.hotjar.com/modules.1e98293c16a88afdf1b7.js> gathers data only if it does not detect DNT.

The retired DNT standard allows specifying extensions to the `navigator.doNotTrack` property (W3C, 2019, Section 5.2.1). The clause in

Figure 5 would not detect such extensions, and the preference not to be tracked would not be honoured in the presence of extensions (such as `"navigator.doNotTrack == 1xyz"`). Nevertheless, the retired standard warns against using extensions, so hopefully, DNT implementers follow the suggestion not to set extensions.

A malicious website can change the value of the `doNotTrack` property to record all sessions, for example, by executing the code in Figure 6 before including the Hotjar script. Although browsers still send the `DNT: 1` HTTP header, Hotjar captures the recording. Thus, a malicious site can confuse Hotjar scripts to record sessions of users opting out of tracking.

```
Object.defineProperty(navigator,
    "doNotTrack", {
    get: function () { return "0"; },
    set: function (a) {},
    configurable: false
});
```

Figure 6: A malicious website can reconfigure browsers to allow tracking by Hotjar.

4.2 TLS Support and Dashboard Data Encryption

Since user data ends in the session recording, recording services must prioritise security. Otherwise, the protection of user data may fail. Hotjar used to deliver playbacks within an HTTP page⁵, even for recordings on HTTPS pages. This allowed a man-in-the-middle to insert a script into the page to extract all the data from the record. Today, Hotjar uses HTTPS.

Moreover, the Hotjar script refused to work when deployed on web site without TLS. Hence, a man-in-the-middle adversary cannot misuse Hotjar to record sessions by adding the Hotjar script to HTTP web pages.

4.3 Transferred Data

Hotjar records the position and timestamp of each click or mouse hover and entered data in input fields, including the time and the timestamp of the input, selector, and input field type. Hotjar also records location, operating system and resolution. Nevertheless, recording of the user's location can be disabled in Hotjar settings by the web operator.

⁵<https://freedom-to-tinker.com/2017/11/15/no-boundaries-exfiltration-of-personal-data-by-session-replay-scripts/>, section 4: "The publisher dashboards for Yandex, Hotjar, and Smartlook all deliver playbacks within an HTTP page, even for recordings which take place on HTTPS pages."

4.3.1 Usernames and Passwords

The original study (Acar et al., 2020) claims that passwords are excluded from the recordings to prevent password leaks. To validate this statement, we created a sign-in form with two input fields of type email and password and tested different scenarios mentioned in section 3. Here, we provide the test results for each scenario.

Hotjar collecting script masks both inputs (username and password) of the log-in form, following Hotjar guidelines. The Hotjar session playback displayed asterisks in both input fields. The number of asterisks is different from the actual password length. Hence, the replay does not leak the real length of the password.

The other log-in form contains the password input element. Both input elements for username and password are decorated with `data-hj-allow` attribute. Even so, Hotjar does not record the content of both input fields. However, once the user interacted with the *show password* feature, our test website changed the type of input fields, causing the user's password to become visible. In this case, Hotjar failed to mask the password field, resulting in capturing the password in the clear.

As expected, the modified script that did not replace the input text let the entered data leak to Hotjar. The usernames, as well as passwords after using the *show password* feature, were then available to the website operator through the Hotjar dashboard.

4.3.2 Shipping Details

Firstly, all entered data to the form created according to Hotjar guidelines to prevent leaks of personal information were replaced with asterisks. Credit card information was replaced with asterisks no matter the data type of the input. Moreover, the phone number was replaced with "1111".

Since most input fields in the shipping form are of the text type, web operators can allow data collection using `data-hj-allow` attribute. Such change allows the web operator to collect first and last names, phone numbers, company names, and credit card information.

(Acar et al., 2020) warned that the `autocomplete` attribute of HTML input elements could leak user passwords outside the log-in process without the awareness of the user. The Hotjar recording script does not mask the content of input elements with both `data-hj-allow` and `autocomplete` attributes.

The modified session replay script allows adversaries to record all entered data, including credit card information.

4.3.3 Rendered Website Content

Hotjar collects rendered page content. Unlike user input recording, the collecting script does not suppress the rendered content unless redacted manually by `data-hj-suppress` attribute. The default configuration leaks all displayed content in our tests. In the testing form with one input field, the script did not collect the content of the input field as it lacked the `data-hj-allow` attribute. However, the search results leaked into the recordings. This allows one to guess the content of the input search field.

4.4 Protections Offered by Browsers

Section 2.4 describes how to change the read-only `navigator.doNotTrack` property, which lets Hotjar inject the recording scripts. The session was recorded using web browsers such as Google Chrome, Microsoft Edge, Firefox or Opera. However, not every session was recorded. Firefox contains tracking protection that blocks content loaded from domains that track users. Opera also offers tracking protection called Tracker Blocker, which blocks the Hotjar session replay script.

5 DISCUSSION

Article 29 of the EU Directive 95/46/EC established The Article 29 Working Party (WP29). GDPR transformed WP29 into the European Data Protection Board (EDPB) with increased powers. Both WP29 and EDPB publish guidelines and opinions with the aim of consistent application of data protection law. WP29 published an opinion on the ePrivacy Directive (Directive 2009/136/EC) consent exception. By applying the opinion on session replay scripts, it is clear that users need to consent to session recording whenever the ePrivacy Directive applies. Even so, the code displayed in Figure 4 does not seek any consent. Consequently, the code may violate European data protection law.

The number of users that signal the DNT flag is not known. However, Mozilla reported that approximately 11% of global Firefox users and approximately 17% of US users signal DNT (Fowler, 2013). Today, the share of users is likely different. Hence, it is difficult to estimate the impact on the privacy of web users.

The shift to TLS is likely affected by the global adoption of TLS (Aas et al., 2019). Recordings no longer transit Hotjar servers in the clear. Moreover, Hotjar disabled its scripts at HTTP-only sites, further

stimulating global HTTPS adoption.

Distinguishing user-specific non-input elements on the page will likely prove difficult. As explained in Section 2.2 and illustrated in Figure 4, the ease of installation is one of the goals of Hotjar. Since non-input elements typically do not contain user-specific data, collecting all data by default makes sense. The controller can add `data-hj-suppress` attribute to disallow data collection from elements containing user-specific content.

However, as (Acar et al., 2020) highlight in the economic analysis of the failures, web operators lack the budget to hire experts to annotate elements that should not be collected. (Selzer et al., 2021) studied the costs of complying with GDPR, the height and the likelihood of fines: small and middle-sized companies should only pay a few cents or euros for GDPR legal compliance. Hence, web operators are not motivated to launch audits on their data collection.

Hotjar can treat all HTML elements to have `data-hj-suppress` by default, which would replace all generated and reflected content with asterisks at the cost of removing all static content. That would motivate web site operators to clearly mark texts that are safe to be recorded. We see two risks with this approach: (1) it makes the deployment of session replay scripts more time-consuming, and (2) the web operators might take the easy way and mark all elements with the `data-hj-allow` attribute. Even though such steps might be illegal, the risk of a fine is so small (Selzer et al., 2021) that such behaviour might appear in practice.

(Acar et al., 2020) considered potential violations of laws like GDPR in the EU and HIPAA (health-care) and FERPA (education) in the United States. To their best knowledge, courts have not decided claims concerning processorship and joint-controllerhip of session recording companies. In contrast, we believe that the rulings of the Court of Justice of the European Union (CJEU) on the concept of data controllers and processors (CJEU, 2018; CJEU, 2019a) are applicable to the case of session recording companies. EDPB published guidelines following the judgments (EDPB, 2021). As long as the session recording companies follow the instructions of the website operators, they can be assumed processors.

As it is the operator that needs to explicitly enable data collection from input elements with the `data-hj-allow` attribute, we believe it is the web operator who controls the data collection, and Hotjar is a processor.

Article 25 of GDPR mandates that personal data controllers apply data protection by design and default. However, it does not prescribe the same obliga-

tions to processors. As numerous web page operators (controllers) deploy session recording scripts, but the number of session recording vendors is small (Acar et al., 2020), legislators should consider extending Art. 25 obligations to processors. We believe that such requirements would remove the option of session replay vendors shifting the responsibility to web operators when it is not a sincere expectation that they would hold to their responsibilities (Grodzinsky et al., 2022; Selzer et al., 2021).

Section 4 explains several ways how malicious web site operators can modify the data-collecting script. We expect that the web site operators would be liable for such misconduct. However, that might prevent session replay vendors from applying technical measures to prevent unauthorised data collection.

(Grodzinsky et al., 2022) highlights that the need to deploy session replay scripts is decreased when following best practices to design UI, and their deployment should be as narrow as possible and with clear objectives. CJEU clarified the meaning of strict necessity (CJEU, 2019b, recital 46) that would likely need to be fulfilled to make data collection legal in the European Union without the consent of data subjects.

6 CONCLUSION

Session recordings allow web operators to detect problems and optimise websites with a limited budget. However, extensive data collection practices can violate laws like the ePrivacy Directive, HIPAA, and FERPA. This position paper focused on Hotjar, a session recording company originally investigated by (Acar et al., 2020) several years ago.

We show that the privacy of users on websites with Hotjar scripts improved. For example, the content of input elements is no longer collected by default as it typically contains user-specific data. Nevertheless, we point out several possibilities of adversaries trying to circumvent the protections of Hotjar. User-specific data stored outside input elements are still collected by Hotjar by default; even though we suggest the application of the `data-hj-suppress` attribute to all DOM elements by default, we argue that data collection of reflected content will likely continue. We also argue that European data protection law likely makes web site operators liable for the issues. However, the price of a data collection audit is several magnitudes higher than the expected costs of non-compliance for almost all companies.

Future research should (1) validate and compare other session recording companies, (2) study the pres-

ence of recording without consent, (3) study the liability of session recording companies for recordings without consent, and (4) propose incentives to audit data collection on websites.

ACKNOWLEDGEMENTS

This work was supported by the Brno University of Technology grant Smart information technology for a resilient society (FIT-S-23-8209).

REFERENCES

- Aas, J., Barnes, R., Case, B., Durumeric, Z., Eckersley, P., Flores-López, A., Halderman, J. A., Hoffman-Andrews, J., Kasten, J., Rescorla, E., Schoen, S., and Warren, B. (2019). Let's encrypt: An automated certificate authority to encrypt the entire web. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS '19*, page 2473–2487, New York, NY, USA. Association for Computing Machinery.
- Acar, G., Englehardt, S., and Narayanan, A. (2020). No boundaries: data exfiltration by third parties embedded on web pages. *Proceedings on Privacy Enhancing Technologies*, 2020:220–238.
- Bortz, A., Barth, A., and Czeskis, A. (2011). Origin cookies: Session integrity for web applications. In *Web 2.0 Security and Privacy (W2SP)*.
- Brookman, J., Rouge, P., Alva, A., and Yeung, C. (2017). Cross-device tracking: Measurement and disclosures. volume 2017, pages 133–148.
- Chatzimpyrros, M., Solomos, K., and Ioannidis, S. (2020). You shall not register! detecting privacy leaks across registration forms. In *Computer Security*, pages 91–104, Cham. Springer International Publishing.
- CJEU (2018). Case C-210/16: Unabhängiges Landeszentrum für Datenschutz Schleswig-Holstein v. Wirtschaftsakademie Schleswig-Holstein GmbH. ECLI:EU:C:2018:388.
- CJEU (2019a). Case C-40/17: Fashion ID GmbH & Co. KG v. Verbraucherzentrale NRW eV. ECLI:EU:C:2019:629.
- CJEU (2019b). Case C-708/18: TK v. Asociația de Proprietari bloc M5A-ScaraA. ECLI:EU:C:2019:1064.
- Dao, H. and Fukuda, K. (2021). Alternative to third-party cookies: Investigating persistent pii leakage-based web tracking. In *Proceedings of the 17th International Conference on Emerging Networking Experiments and Technologies, CoNEXT '21*, pages 223–229, New York, NY, USA. Association for Computing Machinery.
- EDPB (2021). Guidelines 07/2020 on the concepts of controller and processor in the GDPR. https://edpb.europa.eu/system/files/2021-07/eppb_guidelines_202007_controllerprocessor_final_en.pdf, Version 2.1.
- Filip, P. and Čegan, L. (2019). Comparing tools for web-session recording and replaying. In *2019 International Conference on Sustainable Information Engineering and Technology (SIET)*, pages 257–260.
- Fowler, A. (2013). Mozilla's new Do Not Track dashboard: Firefox users continue to seek out and enable DNT. Available online at <https://blog.mozilla.org/netpolicy/2013/05/03/mozillas-new-do-not-track-dashboard-firefox-users-continue-to-look-out-and-enable-dnt/>.
- Grodzinsky, F. S., Miller, K. W., and Wolf, M. J. (2022). Session replay scripts: A privacy analysis. *The Information Society*, 38(4):257–268.
- Hils, M., Woods, D. W., , and Böhme, R. (2021). Privacy preference signals: Past, present and future. *Proceedings on Privacy Enhancing Technologies*, 2021:249–269.
- Kaur, K. and Singh, H. (2015). Analysis of website using click analytics. *International Journal of Science, Engineering and Computer Technology*, 5(6):185.
- Kontaxis, G. and Chew, M. (2015). Tracking protection in firefox for privacy and performance. In *Web 2.0 Security & Privacy Workshop*.
- Krishnamurthy, B. and Wills, C. (2011). Privacy leakage vs. protection measures: the growing disconnect. In *Proceedings of the Web 2.0 Security and Privacy Workshop*.
- Merzdovnik, G., Huber, M., Buhov, D., Nikiforakis, N., Neuner, S., Schmiedecker, M., and Weippl, E. (2017). Block me if you can: A large-scale study of tracker-blocking tools. In *2017 IEEE European Symposium on Security and Privacy (EuroSP)*, pages 319–333.
- Selzer, A., Woods, D., and Böhme, R. (2021). Practitioners' corner: An economic analysis of appropriateness under Article 32 GDPR. *European Data Protection Law Review*, 7(3).
- Senol, A., Acar, G., Humbert, M., and Borgesius, F. Z. (2022). Leaky forms: A study of email and password exfiltration before form submission. In *31th USENIX Security Symposium (USENIX Security 22)*, pages 1813–1830. USENIX Association.
- Starov, O., Gill, P., and Nikiforakis, N. (2016). Are you sure you want to contact us? Quantifying the leakage of PII via website contact forms. volume 2016, pages 20–33.
- W3C (2019). Tracking preference expression (DNT). The World Wide Web Consortium (W3C), Tracking Protection Working Group, <https://www.w3.org/TR/tracking-dnt/>.