An Investigation of User Privacy and Data Protection on User-Side Storages

https://doi.org/10.3991/ijoe.v15i09.10669

Thamer Al-Rousan
Isra University, Amman, Jordan,
thamer.rousan@iu.edu.jo

Abstract—Along with the introduction of HTML5, a new user storage technologies; particularly, Web SQL Database, Web Storage, and Indexed Database API have emerged. The common goal of these storage technologies is to overcome the limitations of legacy of user-side storage mechanisms. All these technologies have many privacy and security concerns, and the main threat is user tracking. In this context, this study investigates the usage of these technologies and to find out which one of these technologies is primarily used by user trackers, and to calculate their frequency in context of 3rd-party tracking code. The result exposes that the adoption of Web Storage most commonly used amongst the three storage technologies. Motivated by the investigation results, this study examines the degree of protection which the popular web browsers supply to prevent privacy violations. The result reveals that the protection mechanisms that are provided by web browsers are almost the same, and in many occasions privacy violations do exist.

Keywords—User-side storages, User tracking, Cookies, Security and privacy

1 Introduction

Using the internet and the services that it provides has continuously increased as it has become the main source of information for thousands of millions of people, at work, at school, and at home. People spend over 60 hours a week browsing the internet, using personal computers or portable devices. However, during browsing the users may visit websites that host malicious software or phishing scams [1].

Using tracking mechanisms such as cookies, trackers are able to track purchases and gather personal data. In many cases it's hard to discover if trackers are gathering the users' data or the way they are using it. Data security and privacy had been the main concerns of internet users for many years [2]. Based on CIGI global survey [3], the majority of surveyed people said that "they are more concerned about their privacy now than the year before." Personal privacy is in great danger due to different threats from suspicious individuals, businesses and security agencies; by tracking the user's activity on the internet. In order to face these threats, Normal users rely on browsers to protect them from privacy violations.

HTTP cookie was the first and the most famous mechanism that allowed users to save data on local storages. However, cookies do not meet the special needs for recent web applications, where a part of the web application code is moved from the server-side to the user-side [4]. So, new mechanisms for user-side storages were needed. Recently W3C specified three new technologies for saving and using permanent data on the user-side: "Web SQL Database, Web Storage, and Indexed Database API (Indexed DB)." These user-side storage technologies bring many benefits, including: increasing storage capacity, faster access to the websites, standardizing accessing local storages and enhancing the user's experience. Beside these benefits, the user needs to be guaranteed regarding his data privacy and security [5].

In this context, this study concentrates on these three HTML5 user-side storage technologies and explores the usage of these technologies as a tracking vector to find out which one of them is primarily used by user trackers. As well as to inspect the degree of protection which is proposed by the most common web browsers on the desktop and examining the capabilities of these browsers in deleting the saved data.

The rest of the study is structured as follows: Section 2 briefly presents the needed background of user-side storages. Section 3 investigates the use of user-side storage and finds out which one of these technologies is primarily used by user trackers. Section 4 presents the level of protection of user-side storages. In Section 5 we present related work. The study concludes with feature work discussion in Section 6

2 Background

HTTP cookie was invented in 1994 by Netscape. It permits small amount of data (4KB) to be saved in user-side storage. The cookies' data can be either in the form of session cookies (for a single session), or persistent cookies (multiple sessions). Although cookies offer a simple mechanism for saving data on user-side database, they suffer from numerous drawbacks. One of these drawbacks is their small storage size, and the necessity to attach them to every HTTP request. Consequently, web traffic will be increased and user preferences will decrease [6].

Among the three technologies specified by W3C, Web Storage is the simplest and oldest mechanism for web applications to save data on user-side storages. Web Storage is a direct substitute of the cookies and provides more potential capabilities for saving data on user-side storage, via the use of JavaScript. Depending on the user Web browser, the capacity of web storage can range from 5MB to 25MB [7]. Web storage has two mechanisms for data saving, precisely: session storage (when the session is terminated the data disappears) and local storage (the data still available after the session termination), and both of them have the same API. Web Storage provides numerous advantages compared with HTTP cookies e.g., more storage capacity, no need to send data back with every HTTP request, better security and better performance [5].

A WebSQL database is another storage mechanism for saving and managing huge amounts of permanent relational data in the user's browser via the use of standard SQL syntax and JavaScript APIs. It resembles Google Gears, as both of them depend

on SQLite. Compared with Web storage, WebSQL database has numerous advantages, like supporting both asynchronous and synchronous database, and supporting more composite data types and more composite queries [8]. On the other hand, the main disadvantage of WebSQL database is that W3C no longer supports it. Consequently, this mechanism is no longer recommended and many browsers have stopped supporting it [5].

Indexed DB is the newest storage mechanism of the three user-side storages. It provides a strong mechanism for saving and managing permanent data via applying JavaScript APIs. Because of the features of IndexedDB, it can be counted as an upgraded and merged version of Web SQL Database and Web Storage [10]. Indexed database API is an object-oriented database. Different types of data can be saved in object storage, and it ranges from simple data types (e.g., date, string and array), to classified objects (e.g., JSON and JavaScript Object). It permits objects to be retrieved/ saved with keys [11]. Indexed DB has numerous advantages, like offering fast access to large volumes of structured data, a bigger storage capacity than what's available in other user-side storages, using indexes for effective searches, and the ability to duplicate the values of the keys. On the other hand, complexity is the main disadvantage of Indexed DB [10].

3 Investigating The Practical Use of Three User-Side Storages

This section explores the usage of three HTML5 storage technologies as a tracking factor. In order to do that, the study explores the use of these technologies on a large sample of the web.

3.1 Methodology

The scope of our analysis includes the most common user-side storages which are: Web SQL Database, Web Storage, and Indexed DB. The goal of the analysis is to explore the usage of these three technologies storages; to find out which one of these technologies is primarily used by user trackers. In this context, the static analysis was performed on wide-ranging sample of dataset in order to locate the constructs of JavaScript, which have been used by one of these three technologies to create, write and read data in the user-side. Then, we find the tracking domain that owns the located scripts. Figure 1 shows the Architecture of our investigation.

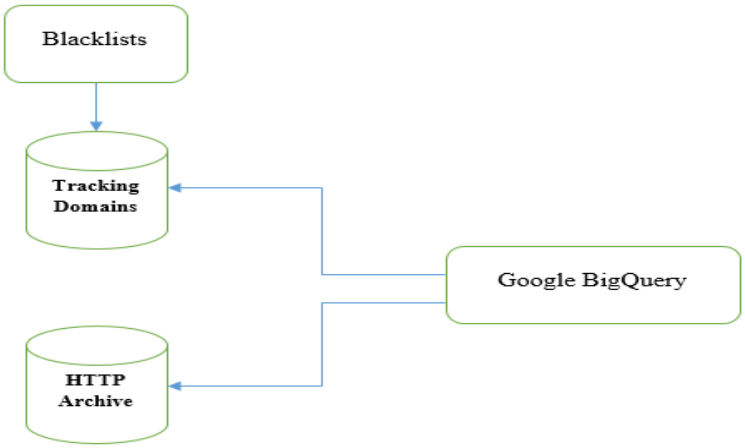

Fig. 1. The Architecture of the Investigation

So, matching rules have been identified to record the constructs needed to implement simple tasks such as creating, writing and reading data. For instance, in Indexed Database API., **transaction** construct is needed to get access to the object store, while, the **objectStore** construct is needed to retrieve an object store, and finally the **IndexedDB** construct is needed to access IndexedDB. The same processes have been followed for Local storage and Web SQL Database. The rules that are used for each rule used for each of the technologies store are presented in table 1.

Table 1. Matching Rues

User-Local Storage	Matching Rues
Web Storage	"locaStorage "AND "sitItem OR getItem"
Indexed Database API	"indexedDB" AND "transaction" AND "objectStrore"
Web SQL Database	"openDatabase" AND "transaction" AND "executeSql"

Web storage offers two different storages. It is important to mention that this study only considered the local storage mechanisms and neglected the session storage mechanisms; because the session storage is dismissed when a browsing session is terminated.

The dataset in this study was generated from **HTTP Archive [12]. HTTP Archive** is an open source project designed by Souders [13]. It periodically analyzes the top websites on the web which are gained from the "Alexa Top Sites", and lists detailed information about: the used web platform APIs, fetched resources and execution traces of each page. Since the size of the generated dataset is huge, the study applied "Google BigQuery" for its processing. Each one of the three technologies a set of SQL queries by using Google BigQuery, which is performed based on the matching rules [14]. Our presumption that if a webpage or any of its sub-resources includes any one of the identified constructs; it will likely use the storage technologies considered to save the data locally. For example, if the user-side script snippet includes the three

constructs: "transaction, indexedDB, objectStrore", then the Indexed DB API will be used to save the data. Similarly, the same rules mentioned in table 1 were used for Web SQL Database and Web Storage.

For the usability and reliability of the research measure, it is important to exclude the data that belongs to the trackers. Trackers collect data about the users' activities across applications or multiple websites that can't be used in its authentic format in this research. So, based on blacklists of trackers, the study creates database of tracking domains in order to identify the sub-resources that are involved with trackers. For this purpose, the study selected two common open source blacklists, i.e.: Easy List [15] and No Track [16] to create database of tracking domains. It is important to mention that the blacklists of tracking are counted as 3rd party components in the same regard as Ad-blocking extensions.

The study carried out the investigation on HTTP Archive dataset on 24th of October 2018, so the only content of data of blacklists that was considered, is the data that was relevant to the data that we preformed our investigation. Table 2 concludes the amount of websites and sub-resources in our investigation. The study also lists the margin of error which stands for the sub-resources that our matching rules are not indorsed.

Table 2. The Investigation Results

	Matching Data for All Dataset(Oct.2018)
Websites Number	481867
Sub-Resources Number	197283691
Sub-Resources not Counted	2.9

3.2 Discussion and Results

Table 3 summarizes the results of the executed queries on HTTP dataset. HTTP Archive includes detailed information about almost half a million of top sites, and it comprises of more than nineteen million sub-resources that include JavaScript files, HTML documents and style sheets. The analysis was organized in four sections. The first section shows the percentage of usage of the three technologies storage compared with the total set of data generated from HTTP Archive. The results show that the majority of the websites use Web storage (72%), the second highest ratio was Indexed Database API (8.3%), and only 1.3% used Web SQL Database.

In section two, the study analyzes the percentage of constructs on third party; this is because most websites contain a diversity of 3rd parties. Actually, in many situations, sub-resources that include the analyzed APIs are frequently recognized as trackers. The results disclose that the constructs are frequently found on 3rd party sub-resources.

In section three, the study analyzes the percentage of domain that contains no less than one of tracking sub-resources. The summary of the analysis shows that the majority of the websites containing a minimum of one user tracking where the Web Storage includes the highest percentage with (56.12%). Indexed Database API (2.54%), and only 0.8% found in Web SQL Database.

In section four, the study analyzes the tracking from a different perspective. For each one of the three technologies store, the study focuses on the percentage of sub-resources that track domain including comparing it with all the sub-resources that include constructs. Precisely, how often have these three techniques been used as "tracking trends?" One can observe that the user tracking trend is very high in all cases which means that the tracking scripts are the main challenge to these techniques even for Web SQL Database which is deprecated.

Table 3. Result for HTTP Dataset

User-Local Storage	Sub-Resources with Constructs (%)	Construct in 3rd party Sub-Resources (%)	Domains Number with Tracking Sub-Resources (%)	Percentage of Tracking Sub-Resource to Sub-Resources with Constructs (%)
Web Storage	72.1	67.8	56.12	78.6
Indexed Database API	8.3	6.2	2.45	38.5
Web SQL Database	1.3	1.15	0.8	48.67

3.3 Limitation

The usage of open source dataset is an objectively common method of making investigation research. Despite the growing popularity that HTTP Archive provides, it does not provide a complete picture of these investigations; as it only offers outlines of front pages of openly existing websites [11]. For example, HTTP Archive does not support some processes; for instance, logged in clients and menu tracking links. As the study concentrates on the global adoption of Web SQL Database, Web Storage, and Indexed DB, the real usage of these technologies storages is out of scope of our research. However, this limitation does not adversely affect the accuracy of the study results as these limitations are common in any static analysis method.

4 Level of Protection over User-Side Storages

The main concern of the user when browsing the web is the confidentiality of his private information. Browsers, as part of the process, are adopting different technics to secure the user's data. Most of users depend on default security modes that are provided by web browsers. The previous part exposes that, Web SQL Database, Web Storage, and Indexed DB are currently exploited by the trackers on regular basis. In this context, in this part, the study will check:

- If the popular desktop browsers support the three above-mentioned technology storages
- The efficiency of the deletion of the generated data saved by these technology storages after the browsing data is cleared
- The protection presented by private mode
- Whether the deleted information in local side storages can be recovered

4.1 Research methodology

In this part of the research work, an experiment was conducted to collect and analyze the remaining data that left on the user machine after the browsing session. Our experiments were conducted in November 2018 and concentrated on the most common desktop browsers. The main goal of this experiment was to expose and assess the security and privacy issues related to the use of one of these technology storages. For this propose, forensic methodology [17] was applied on computer lab at Isra University of Jordan. With nearly 60 computers, fresh Windows10 and Mac 10.12.5 with a single web browser were installed. The number of computers with Windows10 was 30 and the rest of the computers had Mac10.12.5. The most recent versions of the five most widespread desktop browsers were installed: Chrome 64, Firefox 59, Opera 49, Edge 40 and Safari 11.1.

To evaluate the browser's behavior in respect to session resumption, the study divided the computers in the lab into two parts. In the first part, all the above-mentioned browsers would be executed in private mode browsing (mixed Windows 10 and Mac10.12.5). In the second part, the above-mentioned browsers would be executed in default browsing mode. In this context, a set of third party tools such as Encase [18], SQLite manager [19] and WinHex [20] were selected. Additionally, a set of targeted websites such as Facebook, amazon and YouTube were selected to emulate a real user's actions in browsing session, and to evaluate a diversity of artefacts contained in web browsing. The following analysis included closing all of the browsing sessions and investigating if any artifacts remained after the browsing sessions. In most cases, this involved investigating particular keywords such as: bookmarks, browsing history, file downloaded, cache memory, cookies, or other artifacts of visited websites. Finally, another test was performed to look deeper at the security related matters with the use of the local side storages. The main goal of this test was to investigate whether the deleted information in local side storages could be recovered.

4.2 Experiment results and discussions

There are some particularities regarding how browsers support the local side storages that earn to be highlighted. For example, the Indexed DB was disabled in Edge and Firefox during private mode while the other local side storages stayed available in both cases. On the other hand, Safari seems to do quite the contrary; as it disabled both of "Web Storage and Web SQL Database," and supported Indexed Database. It is worth mentioning that disabling specific user-storage and enabling the use of another is not a good strategy to protect user privacy; because attackers can use multi-level methods based on a mixture of different storage paths to threat the user's protection.

The results also found that in some cases, browsers needed extra actions to delete saved data in local side storages; which threats violating the user's privacy. Some actions needed the users to add "offline website data" manually from the browser setting menu. In other cases, as the guest session in Chrome, the users had to completely quit the browser; in order to delete the saved data; which meant that just clos-

ing the browser without closing the application couldn't be enough to delete the data from local storages.

As these actions could be misleading for unexperienced users, the privacy and security cannot be ensured. It is worth emphasizing that most of unexperienced users use the default settings for saving their privacy. As result, the trackers can use these weaknesses as a backdoor to violate the users' anonymity. Table 1 summarized the results.

Table 4. Summary of the results

OS	Browser	Mode	Local Side Storages Supports			Effectiveness of removing data		
			Indexed Database	Local Storage	Web SQL	Indexed Database	Local Storage	Web SQL
Mac 10.12.5	Firefox 59	Default	Support	Support	Not Support	Data deleted "Only if selected by users"	Data Deleted	N/A
		Private	Disabled	Support	Not Support	N/A	Data Deleted	N/A
	Opera 49	Default	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
		Private	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
	Chrome 64	Default	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
		Incognito	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
		Guest	Support	Support	Support	Data deleted "Only if Closing Chrome"	Data deleted "Only if Closing Chrome"	Data deleted "Only if Closing Chrome"
	Safari 11.1	Default	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
		Private	Support	Disabled	Disabled	Data Deleted	N/A	N/A
Windows 10	Firefox 59	Default	Support	Support	Not Support	Data deleted "Only if selected by users"	Data Deleted	N/A
		Private	Disabled	Support	Not Support	N/A	Data Deleted	N/A
	Opera 49	Default	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
		Private	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
	Chrome 64	Default	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
		Incognito	Support	Support	Support	Data Deleted	Data Deleted	Data Deleted
		Guest	Support	Support	Support	Data deleted "Only if Closing Chrome"	Data deleted "Only if Closing Chrome"	Data deleted "Only if Closing Chrome"
	Edge 40	Default	Support	Support	Support	Data Deleted	Data Deleted	N/A
		Private	Disabled	Support	Not Support	N/A	Data Deleted	N/A

4.3 A security investigation test

The main goal of this test was to look deeper at the security issues associated with client-side storages. The steps of the test preformed were based on forensic methodology to investigate the ways that the security professionals can use to track the information [17]. The first step was to select several computers from a previous investigation. All the tested computers used Chrome browser as it supports the three types of local side storages. The second step was to delete information saved in client-side storages. The third step was to perform a series of tests to investigate the location of the deleted file in the hard drive, to find out if the file's name was changed after the files were deleted, and finally to examine whether the recovered files could be used in other web browser.

EnCase forensic tool was used to investigate the deleted information in our test. EnCase tool offers a specific functionality for examining and analyzing information from a computer and mobile devices. It provides access to deleted and protected data and it is used in the majority of criminal cases [22]. In addition, a write-blocker tool was used in all deleted cases to guarantee that writing information did not happen throughout the information recovering [23].

Results: Chrome browser saves browsing history in SQLite database which is found in "C:\Users –username-\AppData\ Local\Google\Chrome\UserData\Default\ <FileName>.sqlite." As Chrome browser saves all data in one place, it was presumed that a compression mechanism is used by Chrome for saving browsing data.

By using Copy/UnErase function, EnCase tool can recover all deleted files in a readable way. The recovered files were exported to a further hard drive. The result shows that the application could read all data and files which were generally available. Regarding the location of the deleted file, Figure 2 shows the location of the files after and before deletion. The deleted files are marked with a grey color.

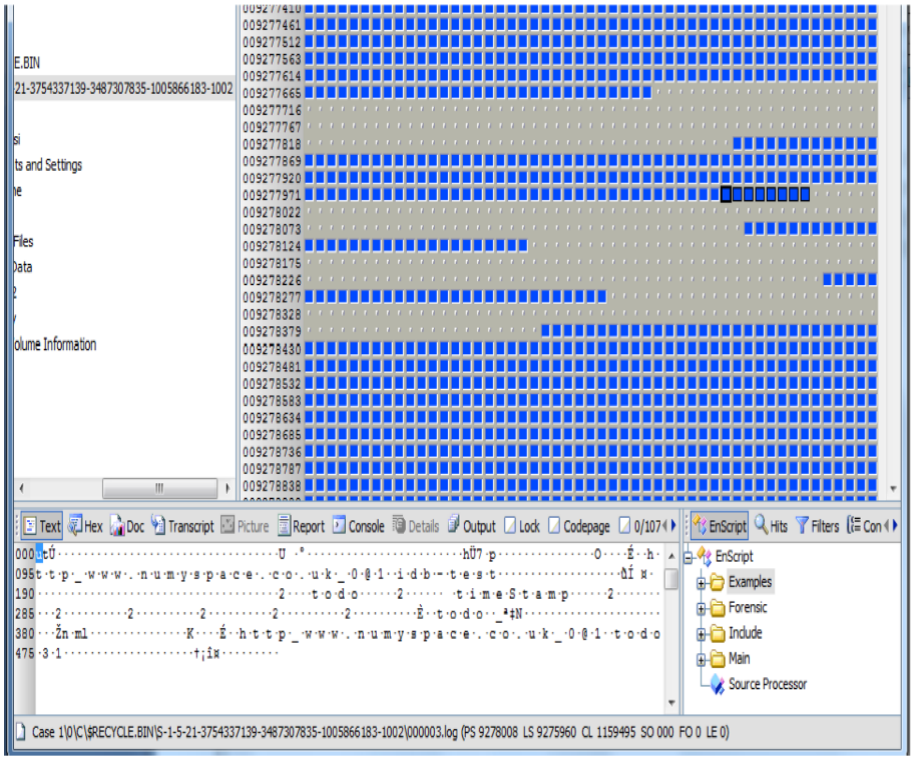

Fig. 2. The location in database file

The results of the test were predictable. Even though the data was marked as deleted, the locations of the deleted data still physically persisted on the hard drive; which means it was is not overwritten. Consequently, when the deleted data are exported into another database, the deleted data could be retrieved and re-used.

5 Related Work

One of the main functionalities included in HTML5 is the applicability to save data locally. The stunning growth of HTML5 usage in modern browsers is also raising concerns about security and privacy. The mechanisms of user-side storage have been investigated by **Hanna et al. (2010);** to identify the potential attacks that threaten user-side storages [24]. They found that the data saved in local storages are normally used by web application without appropriate security. The study listed many real life examples of possible attacks, and it also suggested many solution to protect saved data. One of these suggestion is that web browsers must "automatically remove any potentially executable script constructs inside database values before returning them."

Tump (2011) preformed similar investigation to find out the effect of XSS attack on user-side storages [25]. The study discussed other security and privacy issues that

might arise, such as the problems with same origin policy (SOP).The study revealed that sometimes, the SOP failed to enforce the attacks which renders the personal information saved in the user-side database in danger. **Lekies and Johns (2012)** performed an investigation on the use of Web Storage in real life by analyzing the main page of the top 500,000 websites [26]. They discovered that more than 5% of the studied websites used Web Storage in an unsecure manner. **Shah (2012)** listed the 10 security threats created by HTML5 [27]. The author suggested a mitigation procedure which depends on authenticating the integrity of the data saved in user-side storage.

Preuveneersa et al. (2013) examined the practicability of HTML5 in an e-health context [28]. Although there are possible advantages of saving data locally, the lack of security and privacy are main concerns. **Jemel and Serhrouchni (2014)** suggested an approach to secure the saved data [29]. Their approach is based on encrypting the saved data by using user credentials information. The third-party threats have been investigated by **Englehardt et al. (2015).** The study revealed that tracking cookies can be used as a tool by state-level intelligence agencies to mass surveillance [30]. Similar to other studies, **Kimak (2016)** claims that the current state of Indexed DB is inevitably insecure due the fact that saved data is not encrypted [31]. He suggested a model to encrypt saved data in the user-side storage with a key that is saved on the server side. The problem which arises with this model is that, the internet connection is needed to decrypt the saved data.

The protection of a private browsing mode was evaluated by **Tsalis et al.** (2017). The study found that the majority of browsers that have been analyzed keep traces of the user browsing in the user file system [32]. More recently, a study conducted by **Wu et al. (2018)** to assess the level of protection of a private browsing mode presented a similar result [33]. The study found that not all web browsers analyzed delete all created data on the user-side database after the in private mode is closed.

6 Conclusion

This study analyzed the three user-side storage technologies which are designed to overcome the weakness of the existing mechanisms for user-side storages, and enhancing the user experience by storing the data locally within the user's machine. These data certainly become targets for attackers which could lead to privacy violations. The three user-side storage technologies have not been studied widely as tracking vectors. In the first part of the study, we quantify the usage frequency of the three technology storages on a dataset obtained from HTTP Archive, which represented a sample of web, and check the extent to which these technologies utilized for tracking goals. The results show that the majority of the websites using Web storage (72%), Indexed DB (8.3%), and only 1.3% used Web SQL Database. The results also show that the majority of the websites including at least one of the user tracking, where the Web Storage has the highest percentage with (56.12%). Indexed Database API (2.54%), and only 0.8% founded in Web SQL Database. The results also show that the majority of the websites containing an at least one tracking element, where the

Web Storage has the highest percentage with (56.12%). Indexed Database API (2.54%), and only 0.8% founded in Web SQL Database.

The level of protection was evaluated in the second part of this study against privacy violations, which is supplied by the majority of web browsers. The result reveals that the protection level offered by the tested web browsers was similar, except Chrome guest mode. In many cases, the protection mechanisms that are provided by web browsers are incapable to delete the related data; they even use the private mode which threats the user's privacy violations.

Lastly, a test was conducted in order to check whether the deleted information in local side storages can be recovered. The results showed that when the data is deleted, it still physically exists even it is marked as deleted. Accordingly, when the deleted data are exported into another database, it can be retrieved and re-used.

We intend in future work to expand our dataset and develop a new methodology to validate the findings of this work. We also intend to target the mobile devices for further web browsers analysis, such as iOS and Android and compare the level of protection amongst the two samples.

7 Acknowledgements

I would like to thank the School of Information Technology, Isra University, Jordan, for providing a conducive environment during the course of my research.

8 References

[1] Bozic, J. & Wotawa, F. (2013). XSS pattern for attack modeling in testing. IEEE Security & Privacy, Vol 8, Issue 6, pp.24-31.

[2] Naseem, S.Z. Rousan, F. HTML5 local storage to restore more data. . 8th International Conference on Computer, 2017, pp.398-408.

[3] CGI-Iposs Global Survey on Internet Security and Trust. (2018). [Online]. Available: https://www.cigionline.org/internet-survey-2018.

[4] Anttonen, M. Salminen, A. Mikkonen, T. Taivalsaari, A (2011). Transforming the web into a real application platform. ACM Symposium on Applied Computing. New York, NY, USA. Pp. 800-807. https://doi.org/10.1145/1982185.1982357

[5] W3C Webstorage Recommendation. (2016). URL https://www.w3.org/TR/webstorage. Accessed March, 2018.

[6] Pandey, S. Chauhan, A. S. Secure Content Sniffing for Web Browsers. Advanced Computing, Networking and Informatics, 2014, pp. 325-332. https://doi.org/10.1007/978-3-319-07350-7_36

[7] Al-Rousan, T. Cloud Computing for Global Software Development: Opportunities and Challenges. In Transportation Systems and Engineering: Concepts, Methodologies, Tools, and Applications. IGI Global, 2015, pp. 897-908. https://doi.org/10.4018/978-1-4666-8473-7.ch045

[8] Olan, M. (2013). HTML5 jumpstart. Journal of Web Engineering, 28(3), 35-36 .

[9] Oosterhuis, H., Culpepper, J., Rijke, M. The Potential of Learned Index Structures for Index Compression. 23rd Australasian Document Computing Symposium. Dunedin, New Zealand, 2018,pp.213-218. https://doi.org/10.1145/3291992.3291993
[10] Robert , A., Ravenscroft, J. (2018). A web based tool for teaching linked lists and binary trees. Journal of ACM Transactions on Social Computing, 33(6), 97-106.
[11] Tahmasbi, N., & Rastegari, E. (2018). Public Cyberbullying on Twitter., Computing Sciences in Colleges, 1(4), 15-28. https://doi.org/10.1007/978-1-4302-6416-3_12
[12] Tsalis , N.,Virvilis, N., & Mylonas, A. Model for Content Aggregation in Browsers. In Security and Cryptography (Eds.). Lecture Notes (CCIS), Springer,2017.
[13] HTTP Archive.(2018). URL. https://legacy.httparchive.org/. Accessed April, 2018.
[14] Stevens, L. Owen, R. J. The Truth About HTML5 Web Apps, Mobile, and What Comes Next. The Truth About HTML5,2014 pp. 153-164.
[15] GoogleBigQuery.(2018). URL. https://cloud.google.com/bigquery/. Accessed March, 2018.
[16] EasyList and EasyList Variants.(2016). URL. https://easylist.to/index.html. Accessed March, 2018.
[17] Quidsup/notrack: No Track blocklists (2016). URL. https://github.com/quidsup/notrack. Accessed March, 2018.
[18] Montasari, R., & Peltola, P. Computer Forensic Analysis of Private Browsing Modes. 10th Conference on Global Security, Safety and Sustainability: Tomorrow's Challenges of Cyber Security. Springer International Publishing,2015. https://doi.org/10.1007/978-3-319-23276-8_9
[19] EnCase Forensic.(2017). URL. https://www.guidancesoftware.com/encase-forensic. Accessed April, 2018.
[20] SQLiteManager.(2018). URL. https://www.sqlabs.com/sqlitemanager.php. Accessed April, 2018.
[21] Hex Editor & Disk Editor. (2013). URL. https://www.x-ways.net/winhex/index-m.html.
[22] EnCase tool. (2015). https://www.guidancesoftware.com/encase-forensic. Accessed April, 2018.
[23] Write-blocker tool. (2018). [Online]. URL. https://forensicsoft.com/. Accessed April, 2018.
[24] Hanna, S., Shin,. and Song, D. The emperor's new APIs: On the (in) secure usage of new client-side primitives. 4th Web 2.0 Security and Privacy Workshop (W2SP), Oakland,2010, pp.48-67.
[25] Tump, E., Some Potential Issues with the Security of HTML5 IndexedDB. IET Systems Safety and Cybersecurity Conference, IET, 2011, pp.942-956. https://doi.org/10.1049/cp.2014.0971
[26] Lekies, S. and Johns, M., Lightweight integrity protection for web storage-driven content caching. 6th Workshop on Web, 2012.
[27] Shah, S., (2012), Top 10 Threats Stealth Attacks in HTML5, (2013), URL. https://www.thinkinfosecurity.com/infosec-notes/html5-top-10-security-threats-stealth-attacks-and-silent-exploits.
[28] .Preuveneers, D., Berbers, Y., Joosen, W., (2013). The future of mobile e-health application development: exploring HTML5 for context-aware diabetes monitoring. Procedia Computer Science, 21,.351-359. https://doi.org/10.1016/j.procs.2013.09.046
[29] Jemel, M. and Serhrouchni, A., Security enhancement of HTML5 local data storage. In Network of the Future (NOF). International Conference and Workshop on the, IEEE,2014, pp. 1-21. https://doi.org/10.1109/nof.2014.7119784

[30] Englehardt, S., Understanding the use of web. IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops),2015 pp. 500-505. https://doi.org/10.1109/percomw.2017.7917475

[31] Kimak, S., An investigation into possible attacks on HTML5 ,10th International Conference for Internet Technology and Secured Transactions (ICITST-2015), 2016,London, UK. https://doi.org/10.1109/icitst.2015.7412126

[32] Tsalis, N., Mylonas, A.,. and Katos, V., (2017). Research and design of E-commerce component. Computer Engineering and Design, 31(2), 374-377.

[33] Wu, D. Meng, Y. and Chen, H. Evaluating private modes in desktop and mobile browsers and their resistance to fingerprinting, in IEEE Conference on Communications and Network Security (CNS), Las Vegas, NV, USA, 2018, pp. 1-9. https://doi.org/10.1109/cns.2017.8228636

9 Author

Thamer Al-Rousan is an Associate Professor of Software Engineering at the Faculty of Information Technology at Isra University, Jordan. He holds a PhD in Software Engineering from University Sains Malaysia (USM). His research is centered in Web Engineering, Cloud Computing, Risk Management, Design and Architectures, Software Metrics and Quality Assurance. Thamer Al-Rousan has number of published papers in different Software Engineering topics. He is also a reviewer in different software engineering journals and conferences (thamer.rousan@iu.edu.jo)

Article submitted 2019-04-15. Resubmitted 2019-05-23. Final acceptance 2019-05-24. Final version published as submitted by the authors.