Report on Sensors APIs: privacy and transparency perspective

Lukasz Olejnik

W3C Invited Expert
lukasz.w3c@gmail.com
`http://lukaszolejnik.com`
Twitter: @lukOlejnik
April 11, 2016

Abstract. In this report, a number of proposed browser APIs is analyzed, in an attempt to discuss the possible privacy issues and to distill clear recommendations. In the end discussion it is concluded and suggested that it might be benefitial to revisit the concept of browser permissions. Conslusions regarding the overall privacy awareness (transparency) user interface are also presented.

1 Introduction

Web browsers are increasingly used for a variety of, perhaps, previously unanticipated activities. There is a permanent and significant incentive to broaden the functionalities of web applications, to make them more accessible and useful. To achieve this, web browsers will be providing access to a number of new data sources, in particular, sensors built into mobile devices.

In case providing access to additional data sources could lead to perhaps unobvious, unexpected and unwanted consequences that could result in the expansion of privacy risks surface - additional scrutiny is benefitial. Among the potential new risk vectors might be an introduction of new vectors for identifier creation, i.e. for fingerprinting [23,8] and tracking[17], including cross-device [3]), and possibly also information leaks [10,11]. In this report, the focus is placed on these and related issues.

It is now a growing understanding that web standards need to consider the possible implications for security and privacy [21]. This document is intended to analyze the possible issues in relation to the existing and proposed device APIs.

It is also recognized that solving or avoiding certain risks, such as fingerprinting, is a challenge [18,8]. Therefore, this note suggests incentives to research, design and architect new directions to tackle the problems, namely, by establishing more user-control, to enhance the transparency aspects.

2 An Introductory Privacy Analysis of Sensors APIs

In this section a selection of existing or proposed browser sensor/device APIs is analyzed. The possible use cases are described, then followed with an attempt to identify possible risks. The list of APIs listed on the site of Device APIs Working Group is used [6][1].

Each of the cases include an attempt to identify possible risks and issues - even possibly non-obvious ones, sometimes without any supporting research references. The aim of this analysis is thinking outside-the-box.

2.1 Battery Status API

Battery Status API allows every web site to access the status of battery on a user's device. [13] Among the queriable information is the battery level, time to full charge or discharge.

The standard [13] has recently received significant scrutiny and in the current editor's draft the privacy considerations are thoroughly laid out. The recommendations are as follows.

- The API should not expose sensor readouts with high precision (limit identifier creation surface).
- Allow the User-Agent (i.e. a web browser) to make the API subject to permissions (provide control to the user).
- The browsers should implement transparency mechanisms so that the users could verify the use patterns of the API (ensure users' awareness, also known as transparency).
- Possibly allow the user to obfuscate the output of the battery readout.

Due to the fact that just a readout of tuple $(batterylevel, charge/dischargetime)$ may potentially form a short-term identifier, it may be worth considering if exposing the full battery level is necessary.

For example, the API definition could consider allowing the readout of less informative data (minimization): instead of the full 100 levels - offer a number of more discrete levels. The names (labels) could then correspond to fixed levels of battery state, possibly associated with canonical names. Among the potential states could be the numericals (canonicals): 1 (full), 0.75 (good), 0.5 (half), 0.25 (low) 0.1 (critical), 0.05 (empty), 0.

It is worth noting that the recently updated text of the Battery API standard could be applied as a blueprint for any future security and privacy considerations section of other standards, possibly after adapting the details (and if relevant) to other device/sensor API. The recommendations are in line with the Device API Privacy Requirements [1], and in general encompass user notice, consent, minimization, control.

2.2 Proximity Events

Proximity Events enables web sites to discover information about the proximity between the user's device and some nearby object [14].

[1] Soon to be renamed to Device and Sensors Working Group

The draft [14] contains a seemingly thorough security and privacy considerations section, where various potential scenarios are considered.

However, the considerations remain on a rather general level and do not attempt to generalize or stipulate requirements and/or suggestions. Standard RFC 119 terminology [2] is also not used.

Furthermore, the considerations seem to delegate some of the responsibility to the web application developers, making them responsible for the use (and abuse) of this API. In such a case, possible implications due to the use of third-party content should be assessed. Especially if the injected third-party content would have impact on the top-level browsing context.

It is understood that the current proposal for level readouts is now for the range between 0 (low) and 152.4 *cm* (high) [3] Such granularity could introduce possible unobvious information leaks usable in profiling. Potential possibilities are listed.

- *Information leaks.* In case the average distance from the used device to the user's face is differing among the individuals, the API could provide a possible differentiator factor and a privacy risk vector. The future potential possibilities of behavioral analysis based on the proximity sensor data should be included in the considerations, with the aim of making the text future-proof.
- *Behavioral analysis.* the patterns of proximity readouts might be individually-attributable. In this case, this would offer a possibility to enhance user profiling based on the analysis of device use. Users could be classified as "heavy users" or "occassional users".

Recommendations follow.

Recommendations

- SHOULD expose adequate proximity levels. Browsers may discretize them accordingly, perhaps by defining specific fixed number of states such as "far", "near". Depending on the use. cases and requirements.
- SHOULD be subject to permissions.
- SHOULD be accounted by the browser privacy UI.
- The user SHOULD be able to review the past/current use patterns.

2.3 Ambient Light Sensor

The Ambient Light Sensor API utilize device sensors to detect the lighting conditions (i.e. light level) near the device [16].

The privacy and security considerations in Ambient Light Sensor [16] are similar to the Proximity sensor API. For this reason, the considerations discussed in the previous section are followed.

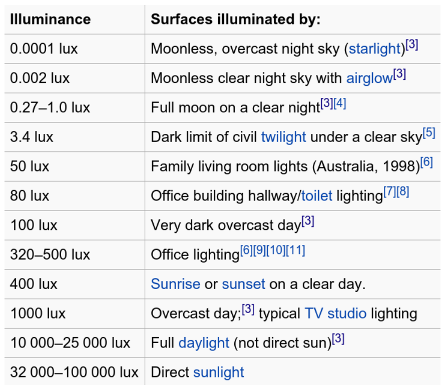
[2] The "MAYs" and "SHOULDs")

[3] `https://github.com/w3c/sensors/blob/3d1a845d59200e4df1e93647b38a8609f3d9ef10/api-sketch.md`

Furthermore it is understood that the current proposal for level readouts is now for the range between 0 and $10000 lux$[4]

Additionally, examples of the following issues ("case studies") may be relevant to consider:

- *Information leaks.* Without access to any other data sources, ambient light sensor might potentially introduce information leaks. Possible example follows.
  Rooms in appartments may be lit differently and this the possibility of revealing e.g. the information relating to the users' appartments/housing situation should not be excluded. While the user was physically moving the device to e.g. different rooms in a house/appartment, the readout will change in this case. Figure 1 shows how different environments correspond to different readouts measured in lux.



Illuminance	Surfaces illuminated by:
0.0001 lux	Moonless, overcast night sky (starlight)[3]
0.002 lux	Moonless clear night sky with airglow[3]
0.27–1.0 lux	Full moon on a clear night[3][4]
3.4 lux	Dark limit of civil twilight under a clear sky[5]
50 lux	Family living room lights (Australia, 1998)[6]
80 lux	Office building hallway/toilet lighting[7][8]
100 lux	Very dark overcast day[3]
320–500 lux	Office lighting[6][9][10][11]
400 lux	Sunrise or sunset on a clear day.
1000 lux	Overcast day;[3] typical TV studio lighting
10 000–25 000 lux	Full daylight (not direct sun)[3]
32 000–100 000 lux	Direct sunlight

Fig. 1: Different lighting conditions are contextual. Source: [22]

  In a possibly far-fetched scenario, precise readouts of light sensor could even contribute to a possible analysis of the users' financial situation. This could be enabled by discovering spaces with different lighting conditions, i.e. relating the number of rooms with price of houses/appartments. For this reason, minimizing the granularity or even introducing more discrete level readouts might be considered. Additionally, there are indications suggesting that verbose readout of lighting conditions may even lead to the recognition of of the surrounding environment of adevice, in some cases this could even led to the recognition of a currently watched online movie [19].
- *Behavioral analysis.* The patterns of the physical use of the device, as well as changing the lighting condition in the users' appartments may expose a channel of behavioral analysis of the user. Example: time of day analysis

[4] `https://github.com/w3c/sensors/blob/3d1a845d59200e4df1e93647b38a8609f3d9ef10/api-sketch.md`

when the device provides data relating to specific lighting conditions could reveal some user-related use patterns.

- *Cross-device linking/tracking.* If a web site script may access lighting events with sufficient accuracy, and cross-check it with a live readout of similar sensor data from the user's other device (other computer or device), it is potentially possible to reason that those devices reside in a same place, and consequently pair them together. Ultimately this could lead to an unobvious, unexpected vector of cross-device linking.
- *Cross-device communication.* It is concenivable that a verbose readout of Ambient Light Sensor data can be used to receive messages emitted by other devices in nearby location. An example messaging possibility could arise if a device would flash with a screen in a previously-established manner. This comment especially applies to the Internet and Web of Things paradigm.

Recommendations follow.

Recommendations

- SHOULD expose adequate, minimized ambient light levels. Browsers may discretize them accordingly, after considering if the exposure of raw data readouts (e.g. in lux) is needed. Possible discretized readout levels could reference specific lighting conditions, i.e. a sufficient number of states between "dark", "well lit", etc. - so that the API would still be useful. A possible naming scheme ("dim", "normal", "washed") could be adopted from the functionally equivalent Media Queries [9].
- SHOULD be subject to permissions.
- SHOULD be accounted by the browser privacy UI.
- The user SHOULD be able to review the past/current use patterns.

2.4 Vibration API

Vibration API enables web sites to induce vibrations on a device [12].

Vibration API [12] is not a source of data on its own. As such it is not producing any data possible to consume by the website. However, it is known that causing a device to vibrate can serve as a source of events for other APIs. In particular, it is known that certain sensors such as accelerators or gyroscopes [2,7] may be subject to tiny imperfections during their manufacturing. Vibration API can help discovering those imperfections by introducing vibration patterns, that are subsequently detected by different APIs, for example to extract a device fingerprint.

In this sense, Vibration API provides an indirect privacy risk, in conjunction with other mechanisms.

Recommendations follow.

Recommendations

– MUST offer an option of limiting vibration patterns.
– SHOULD be accounted by the browser privacy UI.
– The user SHOULD be able to review the past/current use patterns.

2.5 Unofficial drafts

In addition to the previously addressed draft standars, unofficial drafts, in particular Atmospheric Pressure Events [20], Ambient Temperature Events [5], Ambient Humidity Events [4] appear to be in early stages and do not currently contain security/privacy considerations sections. However, the recommendation might remain similar to the ones outlined previously in this document. There may also be similarities in terms of possible privacy risks. For example, information leaks and pattern of use analyses:

– Temperature/Humidity readout: when the user enters/leaves an appartment.
– Life activity patterns of use, based on the combined temperature/humidity/pressure change monitoring.

In addition, it is understood that the current proposal for level readouts in the Temperature API is now between 0 and 150 C [5]. This information could open similar behavioral analyses as in the case of Proximity and Ambient Light sensors.

Sensors API It is acknowledged that most of the previously mentioned APIs are to be merged in a single Sensors API. In this case all the comments are maintained, although the report is prepared for the current API drafts. It is very likely that each particular sensor offered by the Sensors API will need to be assessed independently.

Analysis of the implications due to readout of all the sensors combined, follows. In case it would be possible to obtain a readout of all the sensors, this versatile data could allow a potentially short-lived, but precise identifier (due to the precise readout of the sensors, and the natural differences between environments). This would be especially noteworthy from the *cross-device linking/tracking* perspective. In case several of the users' computers or devices were equipped with sensors with similar accuracy, combining the readout from all the sensors, on all the devices, could offer an approximate information that all the devices reside close to a particular user. Example follows. If a user maintains devices $d_1, d_2, d_3, ..., d_k$ – all equipped with sensors X, Y, Z (e.g. it could be a proximity, temperature, light events sensor, etc.), then if it's possible to interrogate (i.e. using web site scripts or readout from mobile applications) the devices and obtain a readout $X_i, Y_i, Z_i, ...$ (where i is an index of devices, in this example $i = \{1, 2, 3, ..., k\}$, and X_i belongs to a device d_i), for a subset of device

[5] `https://github.com/w3c/sensors/blob/3d1a845d59200e4df1e93647b38a8609f3d9ef10/api-sketch.md`

indices $\{1, ..., k\}$, it would be potentially possible to link those devices, should the sensor readout be similar. Exact readout overlap might not necessarily be required. This is a primary reason why a strong minimization agenda in case of Sensors API is worth following.

3 Discussion

Previous sections discussed APIs, possible privacy issues (i.e. information leaks, behavioral profiling, cross-device linking) and possible concrete recommendations. In this sections the discussion shifts to the recommendations and potential direct and indirect implications for other APIs and in the end – browser user interface (UX).

The inflation of permissions requests is often raised as an unwanted issue. This is because it distracts the user from his/her browsing and requires him/her to read through a number of prompts (prompt-fatigue); it is also generally decreasing the UX of Web browsing. Similar comments apply to the recommendations aiming at informing the user about the use of the API[6].

However, an important question is whether W3C standards should provide guidelines and rather leave the functional aspects to browser vendors. Adequate standards terminology allows browser vendors to flexibly choose how they intend to proceed with implementations – still maintaining a cross-browser API functionality. In fact, such works could lead to clear directions and work towards privacy awareness (transparency) user interfaces, and in general to the improvement of user awareness.

Furthermore, assuming all the possible (in clear or unclear manner) data sources are to be encompassed by permissions system, it might be worthwhile to revisit the actual Permissions API [15] altogether[7]. Perhaps it would be benefitial to consider introducing *sensitivity levels* of a given API. For example, "geolocation" could be marked as a "sensitive" data, and "proximity sensor" with a different *permission level*, i.e., "medium" or "low" or other significance. In this way browser privacy UI could introduce sane defaults offering guidance about which prompts to show, about which API usage to clearly inform, etc.

In addition to that, browsers could introduce *default sensitivity settings* allowing to use the APIs with specific *permission levels*. For example, the ones with "low" level could be allowed to be used without any prompt at all – by default. This way it would additionally be straight-forward to change the level, if additional information about the privacy sensitivity relevance should become available, or should the user be merely interested in being informed to a greater extent. At the moment, such a functionality is not offered to the users.

[6] Example with geolocation: it is marked with a "compass" icon in Chrome
[7] It is understood that a re-work of Permissions API is under considerations

4 Conclusion

This report analyzes a number of browser APIs, currently under development. Among the considered points are the possible privacy risk cases, specifically: information leaks, behavioral profiling, cross-device linking/tracking and attempted to distill clear recommendations, in line with the principles of minimization, user awareness and consent.

It was highlighted how the implementation of additional recommendations do not need to bring unwanted consequences resulting in the inflation of the number of permissions requests, which are generally not wanted from the usability point of view. A number of ideas possibly suggests the broadening of the Permissions API concept.

Lastly, the issue of privacy awareness (transparency), and the possible works needed towards similar directions were mentioned. It is the authors personal opinion that transparency guidelines should be clearly defined by W3C. It is advocated to understand the needs, research and desig new approaches, including the browser privacy user interface layers.

References

1. J. M. Alissa Cooper, Frederick Hirsch. Device API Privacy Requirements, 2010.
2. H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh. Mobile device identification via sensor fingerprinting. arXiv preprint arXiv:1408.1416, 2014.
3. G. H. G. N. Chris Calabrese, Katherine L. McInnis. Re: Comments for November 2015 Workshop on Cross-Device Tracking, 2015.
4. M. Cáceres and B. N.V. Ambient Humidity Events, 2015.
5. M. Cáceres and B. N.V. Ambient Temperature Events, 2015.
6. DAP. Device APIs Working Group, 2009.
7. S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi. Accelprint: Imperfections of accelerometers make smartphones trackable. In NDSS. Citeseer, 2014.
8. P. Eckersley. How unique is your web browser? In Privacy Enhancing Technologies, pages 1–18. Springer, 2010.
9. D. G. A. v. K. Håkon Wium Lie, Tantek Çelik. Media Queries Level 4, 2016.
10. C. Jackson, A. Bortz, D. Boneh, and J. C. Mitchell. Protecting browser state from web privacy attacks. In Proceedings of the 15th international conference on World Wide Web, pages 737–744. ACM, 2006.
11. D. Jang, R. Jhala, S. Lerner, and H. Shacham. An empirical study of privacy-violating information flows in javascript web applications. In Proceedings of the 17th ACM conference on Computer and communications security, pages 270–283. ACM, 2010.
12. A. Kostiainen. Vibration API, 2015.
13. A. Kostiainen and M. Lamouri. Battery Status API, 2012.
14. A. Kostiainen and D. D. Tran. Proximity Events, 2013.
15. M. Lamouri and M. Cáceres. Permissions API, 2015.
16. T. Langel and A. Kostiainen. Ambient Light Sensor, 2016.
17. J. R. Mayer and J. C. Mitchell. Third-party web tracking: Policy and technology. In Security and Privacy (SP), 2012 IEEE Symposium on, pages 413–427. IEEE, 2012.

18. M. Nottingham. Unsanctioned Web Tracking, 2015.
19. L. Schwittmann, V. Matkovic, M. Wander, and T. Weis. Video recognition using ambient light sensors. In 2016 IEEE International Conference on Pervasive Computing and Communications, PerCom 2016, Sydney, Australia, 14-18 March, 2016, pages 92–100, 2016.
20. D. D. Tran. Atmospheric Pressure Events, 2015.
21. M. West. Self-Review Questionnaire: Security and Privacy, 2015.
22. Wikipedia. Lux, 2015.
23. T.-F. Yen, X. Huang, F. Monrose, and M. K. Reiter. Browser fingerprinting from coarse traffic summaries: Techniques and implications. In Detection of Intrusions and Malware, and Vulnerability Assessment, pages 157–175. Springer, 2009.