Battery Status Not Included: Assessing Privacy in Web Standards

Lukasz Olejnik
Independent Researcher
lukaszolejnik.com

Steven Englehardt
Princeton University
singlehardt.com

Arvind Narayanan
Princeton University
randomwalker.info

Presented by
Gunes Acar
Postdoctoral Research Associate
Center for Information Technology Policy
Princeton University
New web features lead to privacy concerns
New web features lead to privacy concerns
New web features lead to privacy concerns

§ 2.8. VREyeParameters

The VREyeParameters interface represents all the information required to correctly render a scene from an eye.

```javascript
interface VREyeParameters {
    readonly attribute Float32Array offset;
    [SameObject] readonly attribute VRFieldOfView fieldOfView;
    readonly attribute unsigned long renderWidth;
    readonly attribute unsigned long renderHeight;
}
```

§ 2.8.1. Attributes

**offset** A three component vector describing the offset from the center point between the users eyes of the eye in meters. The x component of this vector SHOULD represent half of the user’s interpupillary distance (IPD), but MAY also represent the vector from the center point of the headset to the center point of the lens for the given eye. Values in the x component for left eye MUST be negative; values in the x component for right eye MUST be positive. This information should not be used to construct a view matrix, prefer using the view matrices provided in VRFrameData instead.
The W3C has a self-review questionnaire

Self-Review Questionnaire: Security and Privacy

W3C Working Group Note 10 December 2015

This version:
http://www.w3.org/TR/2015/NOTE-security-privacy-questionnaire-20151210/

Latest published version:
http://www.w3.org/TR/security-privacy-questionnaire/

Editor's Draft:
https://w3ctag.github.io/security-questionnaire/

Editor:
Mike West, Google Inc., mkwst@google.com

[PUBLIC DOMAIN] To the extent possible under law, the editors have waived all copyright and related or neighboring rights to this work. This document is also made available under the W3C Software and Document License.

Abstract

This document lists a set of questions one could ask about the security and privacy impact of a new feature or specification. It is meant as a tool that groups or individuals can use as a guide during a self-review, pointing towards important questions in areas where expertise might be lacking.
The W3C has a self-review questionnaire

1 Introduction
2 Threat Models
   2.1 Passive Network Attackers
   2.2 Active Network Attacker
   2.3 Same-Origin Policy Violations
   2.4 Third-Party Tracking
3 Questions to Consider
   3.1 Does this specification deal with personally-identifiable information?
   3.2 Does this specification deal with high-value data?
   3.3 Does this specification introduce new state for an origin that persists across browsing sessions?
   3.4 Does this specification expose persistent, cross-origin state to the web?
   3.5 Does this specification expose any other data to an origin that it doesn’t currently have access to?
   3.6 Does this specification enable new script execution/loading mechanisms?
   3.7 Does this specification allow an origin access to a user’s location?
   3.8 Does this specification allow an origin access to sensors on a user’s device?
   3.9 Does this specification allow an origin access to aspects of a user’s local computing environment?
   3.10 Does this specification allow an origin access to other devices?
   3.11 Does this specification allow an origin some measure of control over a user agent’s native UI?
   3.12 Does this specification expose temporary identifiers to the web?
   3.13 Does this specification distinguish between behavior in first-party and third-party contexts?
   3.14 How should this specification work in the context of a user agent’s "incognito" mode?
   3.15 Does this specification persist data to a user’s local device?
   3.16 Does this specification have a "Security Considerations" and "Privacy Considerations" section?
   3.17 Does this specification allow downgrading default security characteristics?
4 Mitigation Strategies
   4.1 Secure Contexts
   4.2 Explicit user mediation
   4.3 Drop the feature
Conformance
Index
   Terms defined by this specification
   Terms defined by reference
Ref
W3C Privacy Interest Group (PING) offers guidance and reviews

The mission…is to improve the support of privacy in Web standards by:

1. Monitoring ongoing privacy issues that affect the Web
2. Investigating potential areas for new privacy work
3. Providing guidelines and advice for addressing privacy in standards development.

https://www.w3.org/2011/07/privacy-ig-charter
The Battery Status API

- charge **level**
  - value between 0 and 1
  - e.g. 0.43 if the battery at 43%
- **charging** status
  - boolean indicator
- time to charge or discharge
  - **dischargingTime**
  - **chargingTime**
  - time in seconds
The development and adoption of the API

- **2011**: W3C Battery Status Event WD
- **2012**: Firefox adds support
- **2013**: WebKit adds support
- **2014**: Firefox 38 rounding bug reported
- **2015**: Chrome rounding patch commit
- **2016**: W3C Prop.Rec.
- **2017**: Blink bug moved to permissions component
  - Yandex announces opt-in model
  - Firefox 52 limits API to internal use
  - Firefox and WebKit remove support
  - Leaking Battery report published
  - Uber study
  - API misuse in the wild
  - Chrome adds support
  - W3C Candidate Recommendation
Mid 2012: Candidate Recommendation adds security and privacy considerations

3. Terminology

The `Function` interface represents a function in the scripting language being used as defined in [HTML5].

The concepts `queue a task` and `fires a simple event` are defined in [HTML5].

The terms `event handlers` and `event handler event types` are defined in [HTML5].

4. Security and privacy considerations

The API defined in this specification is used to determine the battery status of the hosting device. The information disclosed has minimal impact on privacy or fingerprinting, and therefore is exposed without permission grants. For example, authors cannot directly know if there is a battery or not in the hosting device.

5. `NavigatorBattery` Interface

The `NavigatorBattery` interface is exposed on the `Navigator` object.

```webidl
Navigator implements NavigatorBattery;
```
Mid 2012: Candidate Recommendation adds security and privacy considerations

“the information disclosed has a minimal impact on privacy or fingerprinting”
The leaking battery
A privacy analysis of the HTML5 Battery Status API

Lukasz Olejnik¹, Gunes Acar², Claude Castelluccia¹, and Claudia Diaz²

¹ INRIA Privatics, Grenoble, France
{lukasz.olejnik,claude.castelluccia}@inria.fr
² KU Leuven, ESAT/COSIC and iMinds, Leuven, Belgium
{gunes.acar,claudia.diaz}@esat.kuleuven.be

Abstract. We highlight privacy risks associated with the HTML5 Battery Status API. We put special focus on its implementation in the Firefox browser. Our study shows that websites can discover the capacity of users’ batteries by exploiting the high precision readouts provided by Firefox on Linux. The capacity of the battery, as well as its level, expose a fingerprintable surface that can be used to track web users in short time intervals. Our analysis shows that the risk is much higher for old or used batteries with reduced capacities, as the battery capacity may potentially serve as a tracking identifier. The fingerprintable surface of the API could be drastically reduced without any loss in the API’s functionality by reducing the precision of the readings. We propose minor modifications to Battery Status API and its implementation in the Firefox browser to address the privacy issues presented in the study. Our bug report for Firefox was accepted and a fix is deployed.
New research exposes multiple privacy vulnerabilities

1. The Battery Status API can be used as a short-term identifier
2. High precision charge level values in Firefox allows the recovery of battery capacity as a long-term identifier
New research exposes multiple privacy vulnerabilities

Online Tracking: A 1-million-site Measurement and Analysis

Steven Englehardt
Princeton University
ste@cs.princeton.edu

Arvind Narayanan
Princeton University
arvindn@cs.princeton.edu

This is an extended version of our paper that appeared at ACM CCS 2016.

ABSTRACT
We present the largest and most detailed measurement of online tracking conducted to date, based on a crawl of the top 1 million websites. We make 15 types of measurements on each site, including stateful (cookie-based) and stateless (fingerprinting-based) tracking, the effect of browser privacy tools, and the exchange of tracking data between different sites (“cookie syncing”). Our findings include multiple sophisticated fingerprinting techniques never before measured in the wild.

This measurement is made possible by our open-source web privacy measurement tool, OpenWPM which uses an automated version of a full-fledged consumer browser. It supports parallelism for speed and scale, automatic recovery from failures of the underlying browser, and comprehensive browser instrumentation. We demonstrate our platform’s strength in enabling researchers to rapidly detect, quantify, and characterize emerging online tracking behaviors.

It is common for a single measurement point to be used for tracking analysis. However, this makes it difficult to distinguish between tracking that occurs only on that site and tracking that occurs on other sites. We discuss the implications of this and how it affects the results of previous studies.

To address this issue, we have developed a technique called “tracking fingerprinting,” which involves measuring the behavior of a user’s browser as they navigate the web. This allows us to identify sites that are actively tracking users, even if they are not the website being visited.

We have also developed a new method for estimating the number of unique users who are tracked by websites. This method takes into account the fact that users often share cookies and other tracking data between different websites.

In addition, we have developed a new method for estimating the number of unique cookies that are used for tracking. This method is based on the observation that cookies are often used in multiple tracking systems and that these systems tend to use cookies in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique browser fingerprints that are used for tracking. This method is based on the observation that browser fingerprints are often used in multiple tracking systems and that these systems tend to use fingerprints in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique device identifiers that are used for tracking. This method is based on the observation that device identifiers are often used in multiple tracking systems and that these systems tend to use device identifiers in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique IP addresses that are used for tracking. This method is based on the observation that IP addresses are often used in multiple tracking systems and that these systems tend to use IP addresses in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique domain names that are used for tracking. This method is based on the observation that domain names are often used in multiple tracking systems and that these systems tend to use domain names in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking domains that are used for tracking. This method is based on the observation that tracking domains are often used in multiple tracking systems and that these systems tend to use tracking domains in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking paths that are used for tracking. This method is based on the observation that tracking paths are often used in multiple tracking systems and that these systems tend to use tracking paths in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking actions that are used for tracking. This method is based on the observation that tracking actions are often used in multiple tracking systems and that these systems tend to use tracking actions in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking events that are used for tracking. This method is based on the observation that tracking events are often used in multiple tracking systems and that these systems tend to use tracking events in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking messages that are used for tracking. This method is based on the observation that tracking messages are often used in multiple tracking systems and that these systems tend to use tracking messages in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking requests that are used for tracking. This method is based on the observation that tracking requests are often used in multiple tracking systems and that these systems tend to use tracking requests in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking responses that are used for tracking. This method is based on the observation that tracking responses are often used in multiple tracking systems and that these systems tend to use tracking responses in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking connections that are used for tracking. This method is based on the observation that tracking connections are often used in multiple tracking systems and that these systems tend to use tracking connections in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking sessions that are used for tracking. This method is based on the observation that tracking sessions are often used in multiple tracking systems and that these systems tend to use tracking sessions in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking values that are used for tracking. This method is based on the observation that tracking values are often used in multiple tracking systems and that these systems tend to use tracking values in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking parameters that are used for tracking. This method is based on the observation that tracking parameters are often used in multiple tracking systems and that these systems tend to use tracking parameters in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking flags that are used for tracking. This method is based on the observation that tracking flags are often used in multiple tracking systems and that these systems tend to use tracking flags in a way that makes them difficult to distinguish from each other.

We have also developed a new method for estimating the number of unique tracking queries that are used for tracking. This method is based on the observation that tracking queries are often used in multiple tracking systems and that these systems tend to use tracking queries in a way that makes them dif
New research exposes multiple privacy vulnerabilities.

Scripts are collecting the battery charge level as part of a fingerprint.
The specification was updated to address privacy vulnerabilities

1. Should avoid high precision readouts
2. Should inform the user when and who is using the API
3. May ask the user for permission
4. May obfuscate or expose fake values
Late 2016: Mozilla proposes removing the API, citing privacy concerns and lack of use

https://groups.google.com/forum/#!msg/mozilla.dev.platform/5U8NHoUY-1k/9ybyzQlYCAAJ
Late 2016: Mozilla proposes removing the API, citing privacy concerns and lack of use.

**Question:**

“Can anyone point to a real website using the Battery API for a legitimate purpose?”

**Conclusion:**

“Everyone agrees there are theoretical good things that can be done with the battery API; we just don't observe them being done.”

https://groups.google.com/forum/#!msg/mozilla.dev.platform/5U8NHoUY-1k/9ybyzQlYCAAJ
Early 2017: Several vendors remove or restrict support, citing privacy and lack of use

- Restricted to non-web content
- Removed from source code
- Open bug (unknown?)
- Opt-in, otherwise dummy values
Our data supports Mozilla’s decision

We measured usage on the top 50,000 sites

33 third-parties on 815 sites use the API

• 16 used it for tracking
  • Mostly fingerprinting
• 8 used it for benign purposes
  • Mostly performance measurement
• 9 unclassified
How can we improve the privacy review process?
1. The specification process should include a privacy review of implementations.

Specification requires two implementations to progress.

Why not require a privacy review of these implementations?

—> Similar precision issues found during privacy review of Ambient Light Sensors API, which included implementation auditing.
2. API use in the wild should be audited after implementation

Trackers are the early adopters of new API!

It’s not clear that the measurement community will continue to support fingerprinting measurement

**Concerns:**

1. Lack of novelty in measurement techniques
2. Measurement of each new API is a small contribution
3. Specifications can’t wait for the publication cycle

**Suggestions:**

1. Measurement through browser telemetry probes?
2. Regular measurement by browser vendors?
3. Public measurements by an NGO — something like archive.org
3. Specification authors should carry out privacy assessments with multiple threat models

An example: fingerprinting with the Audio API

This is a concern for the Tor Browser!
4. Improve incentives for academics to contribute research

The specification process can benefit from a deeper connection to research.

- Incentivizing attempts to break the privacy assumptions of specifications
- Organizing a forum for academics and researchers to publish their privacy reviews
- Funding attack research
Thank you!

In summary:
1. Include audits of implementations in reviews
2. Audit API use after deployment
3. Carry out analysis in multiple threat models
4. Improve incentives for academics to contribute research
5. Avoiding over-specification supports innovative solutions
6. Provide guidance for web developers in addition to vendors

Slides credit: Steve Englehardt
Full paper: https://senglehardt.com/papers/iwpe17_battery_status_case_study.pdf

Image assets from the Noun Project:
Browser Battery by Aybige, Browser Window by amy morgan