

Battery Status Not Included: Assessing Privacy in Web Standards



Lukasz Olejnik
Independent Researcher
lukaszolejnik.com



Steven Englehardt
Princeton University
senglehardt.com



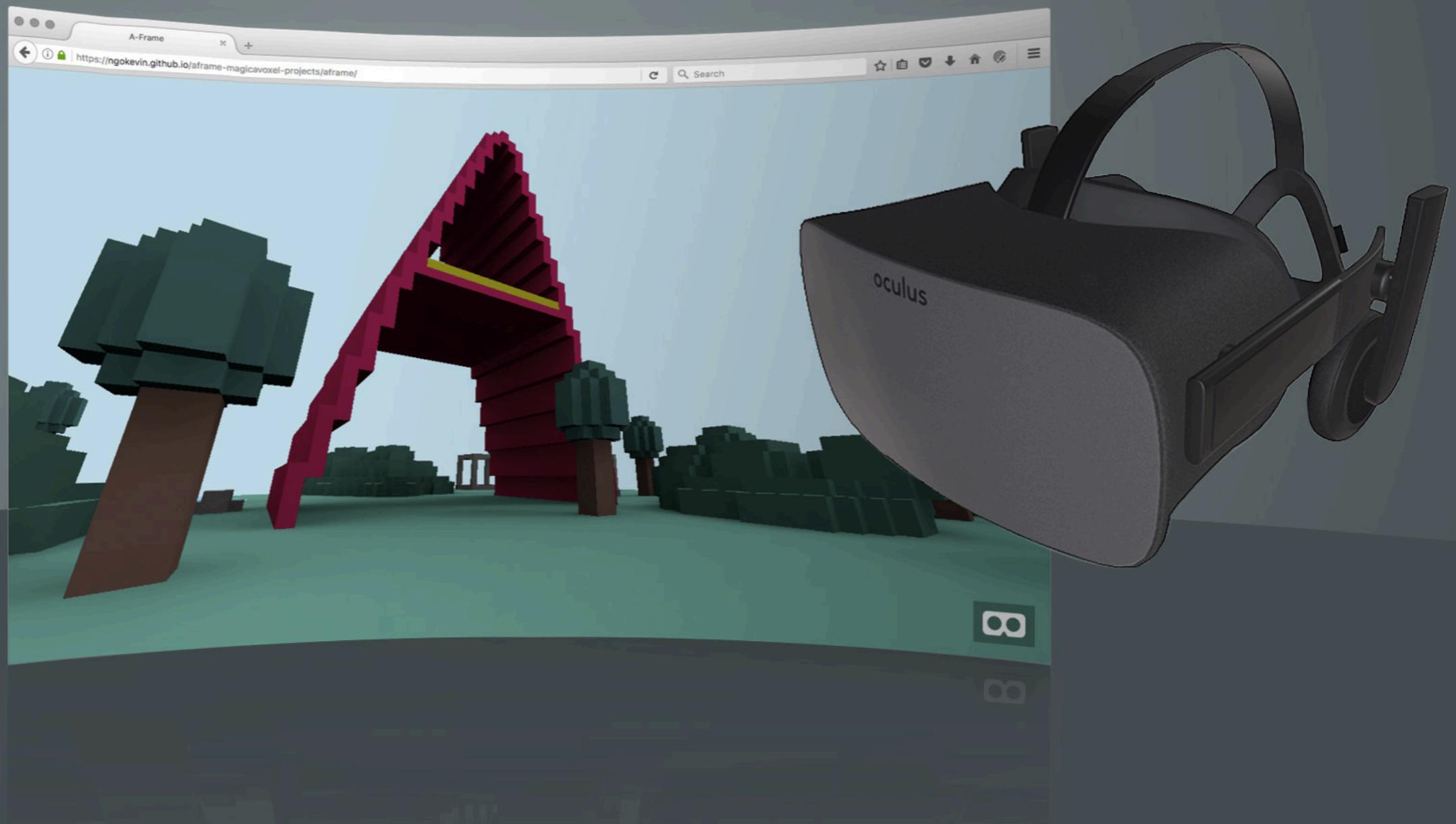
Arvind Narayanan
Princeton University
randomwalker.info

New web features lead to privacy concerns

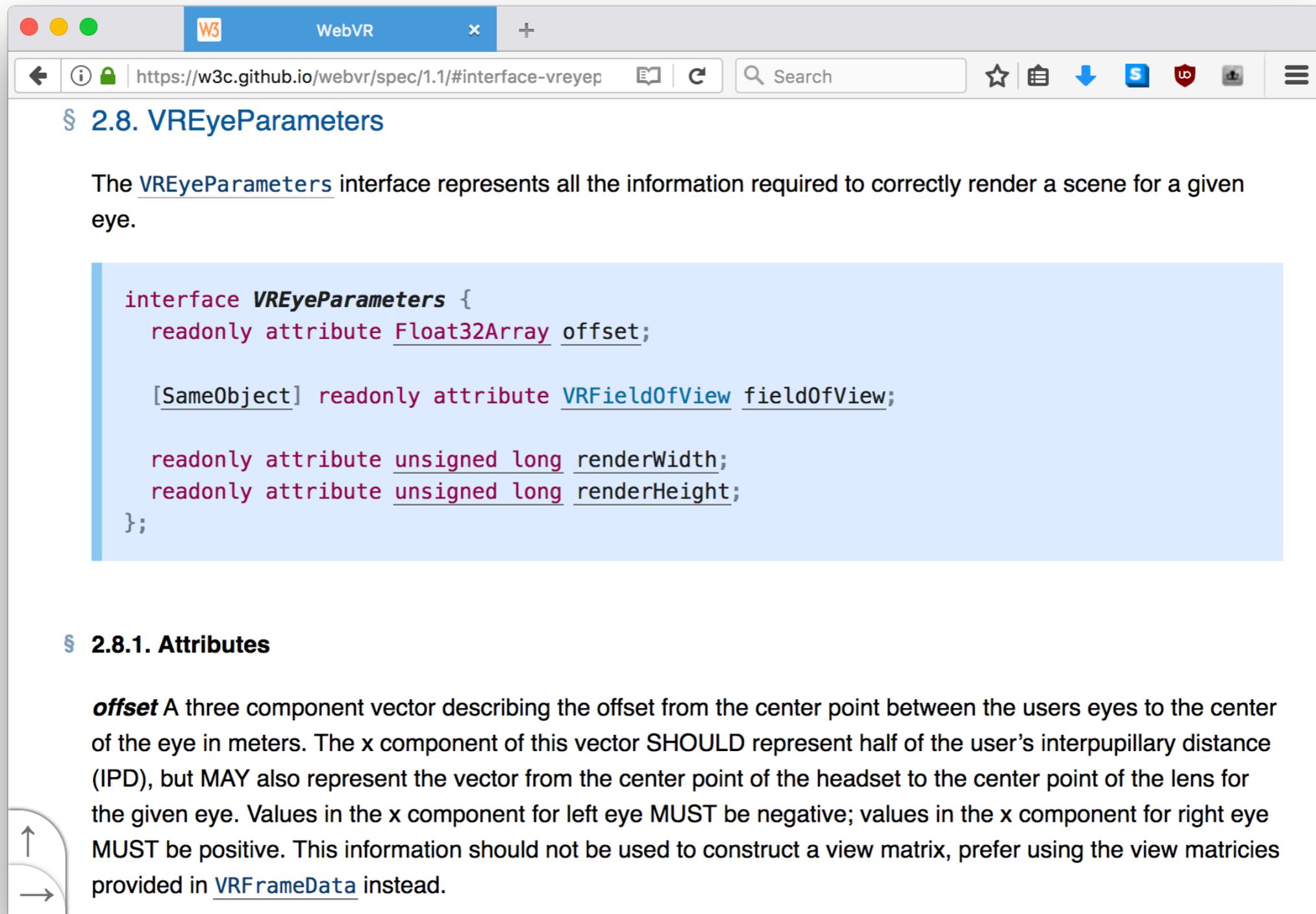
New web features lead to privacy concerns



New web features lead to privacy concerns



New web features lead to privacy concerns



The screenshot shows a web browser window with the title "WebVR" and the URL "https://w3c.github.io/webvr/spec/1.1/#interface-vreyep". The page content is from the W3C specification for VREyeParameters. It includes a section header "§ 2.8. VREyeParameters", a paragraph describing the interface, a code block for the interface definition, and a sub-section "§ 2.8.1. Attributes" with a detailed description of the "offset" attribute.

§ 2.8. VREyeParameters

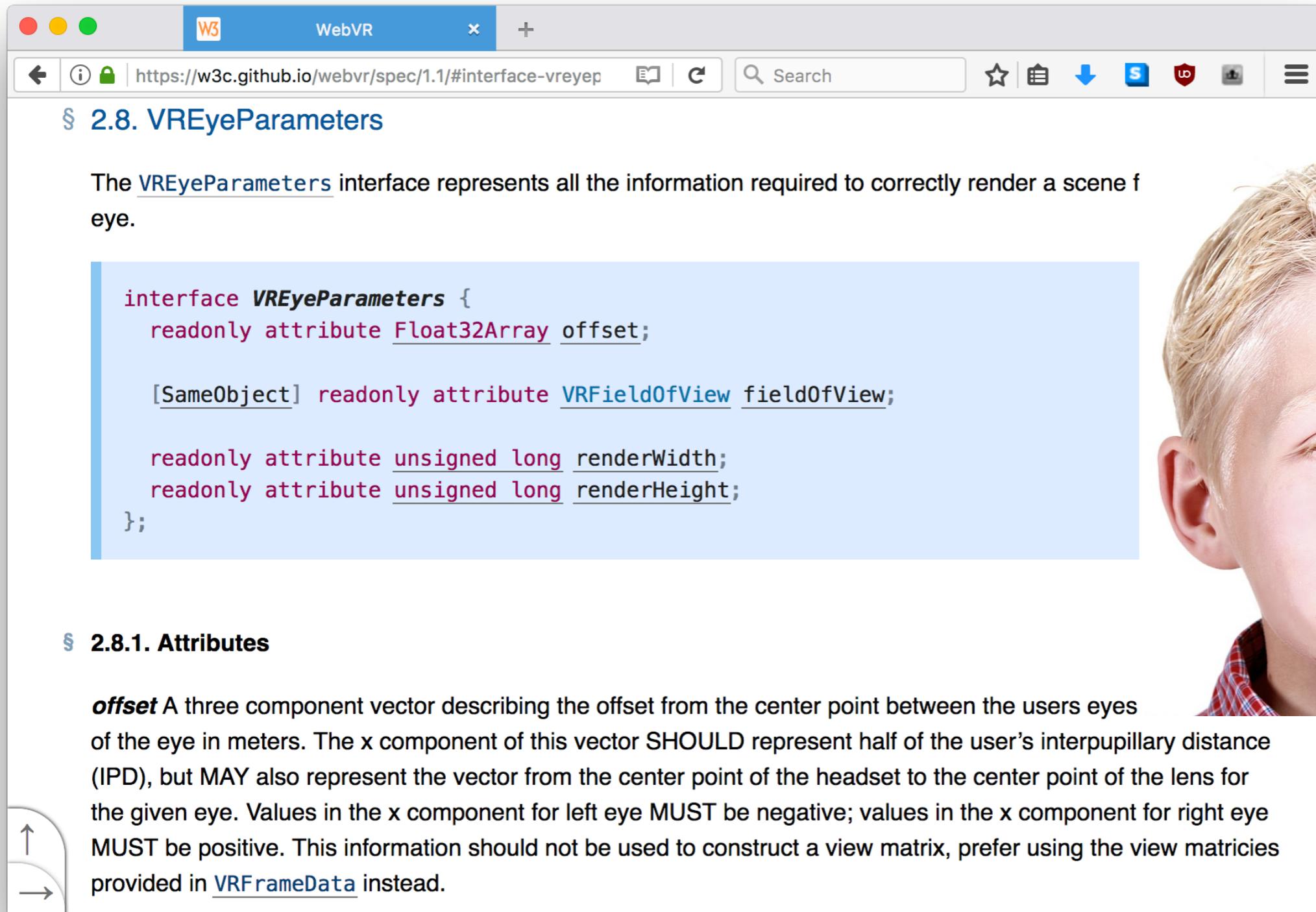
The [VREyeParameters](#) interface represents all the information required to correctly render a scene for a given eye.

```
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 to the center 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.

New web features lead to privacy concerns



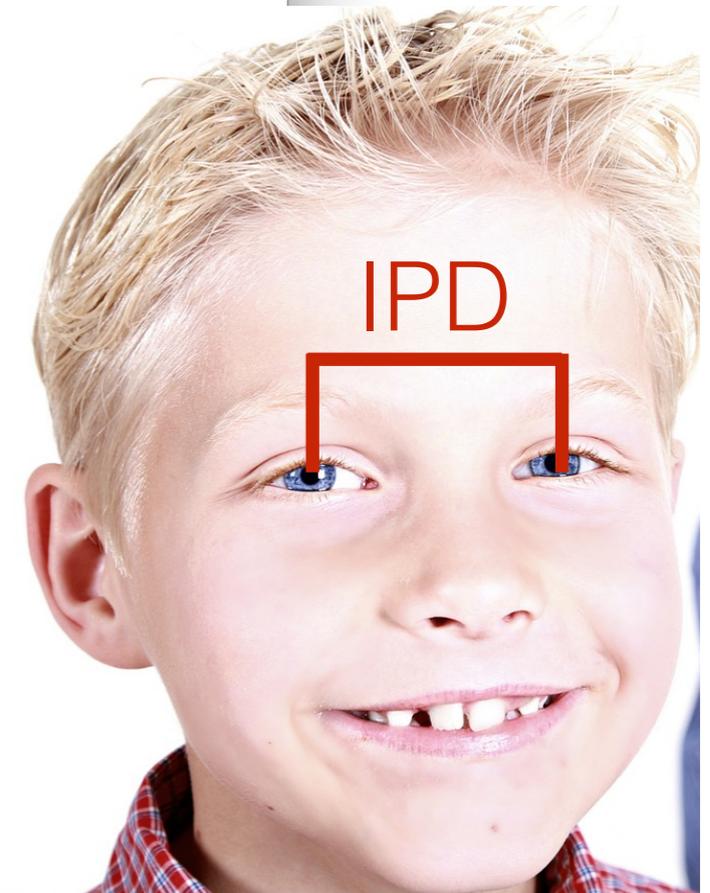
§ 2.8. VREyeParameters

The [VREyeParameters](#) interface represents all the information required to correctly render a scene for an eye.

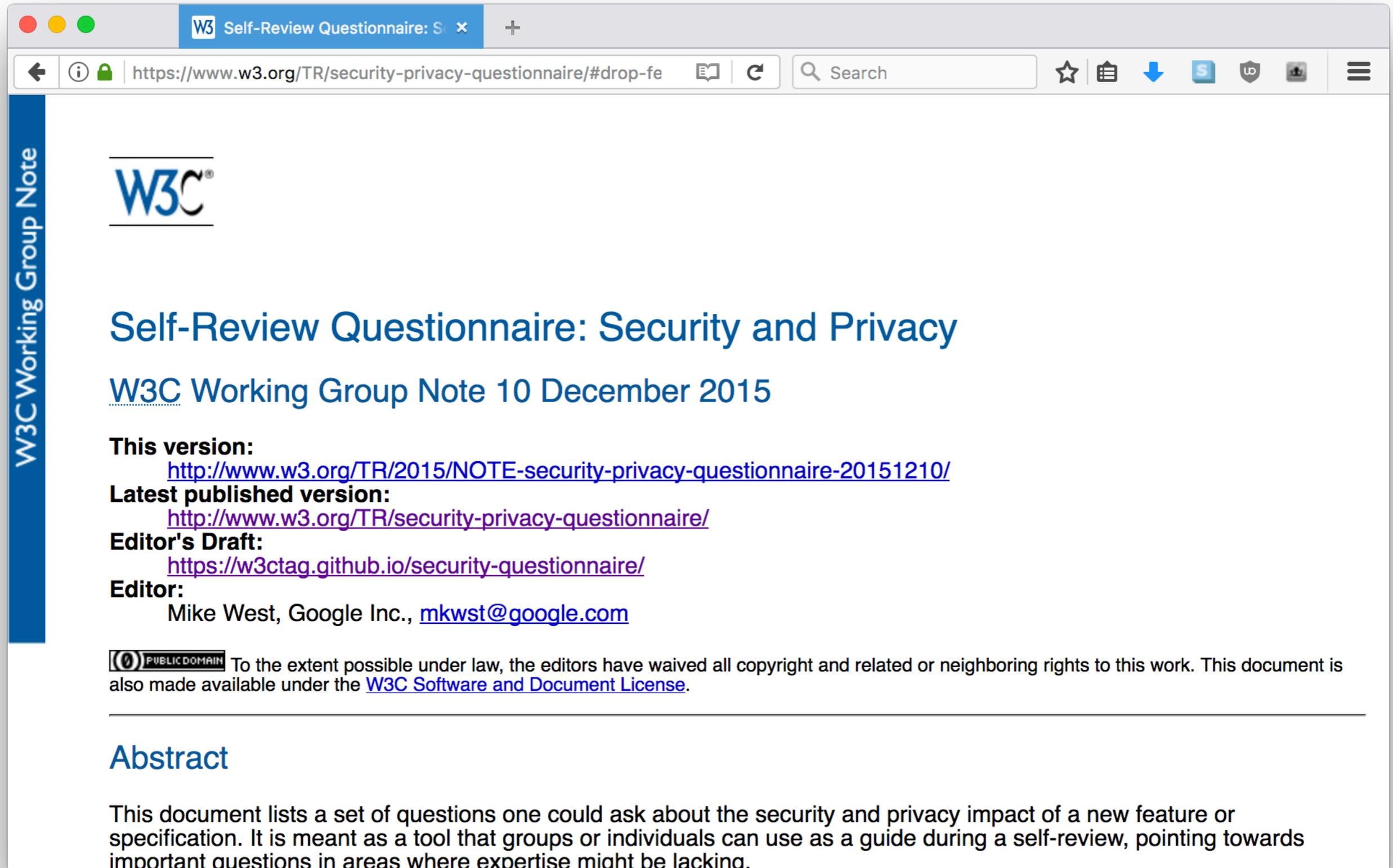
```
interface VREyeParameters {  
  readonly attribute Float32Array offset;  
  
  [SameObject] readonly attribute VRFieldOfView fieldOfView;  
  
  readonly attribute unsigned long renderWidth;  
  readonly attribute unsigned long renderHeight;  
};
```

§ 2.8.1. Attributes

offset A three component vector describing the offset from the center point between the user's 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



The screenshot shows a web browser window with the following elements:

- Browser Tab:** W3 Self-Review Questionnaire: S x
- Address Bar:** <https://www.w3.org/TR/security-privacy-questionnaire/#drop-fe>
- Page Header:** W3C Working Group Note (vertical text on the left)
- W3C Logo:** W3C[®]
- Section-Header:** Self-Review Questionnaire: Security and Privacy
- Text:** W3C Working Group Note 10 December 2015
- Links:**
 - 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:** 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](#).
- Section-Header:** Abstract
- Text:** 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

W3C Working Group Note

- 1 Introduction
- 2 Threat Models
 - 2.1 Passive Network Attackers
 - 2.2 Active Network Attackers
 - 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

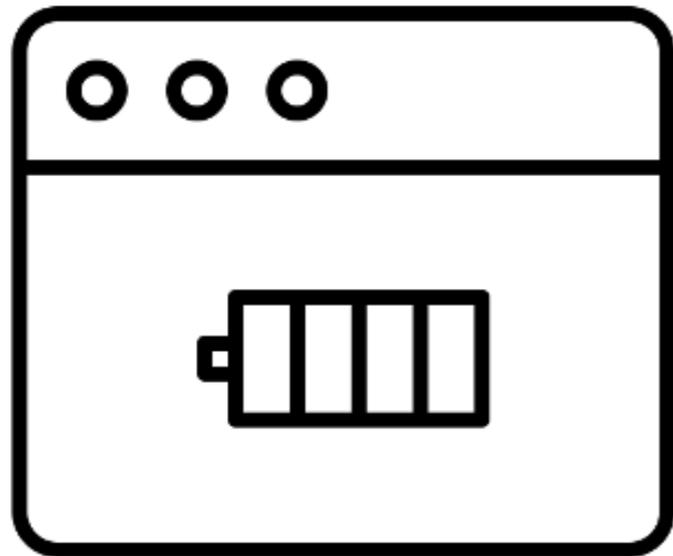
W3C Privacy Interest Group (PING) offers guidance and reviews

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.

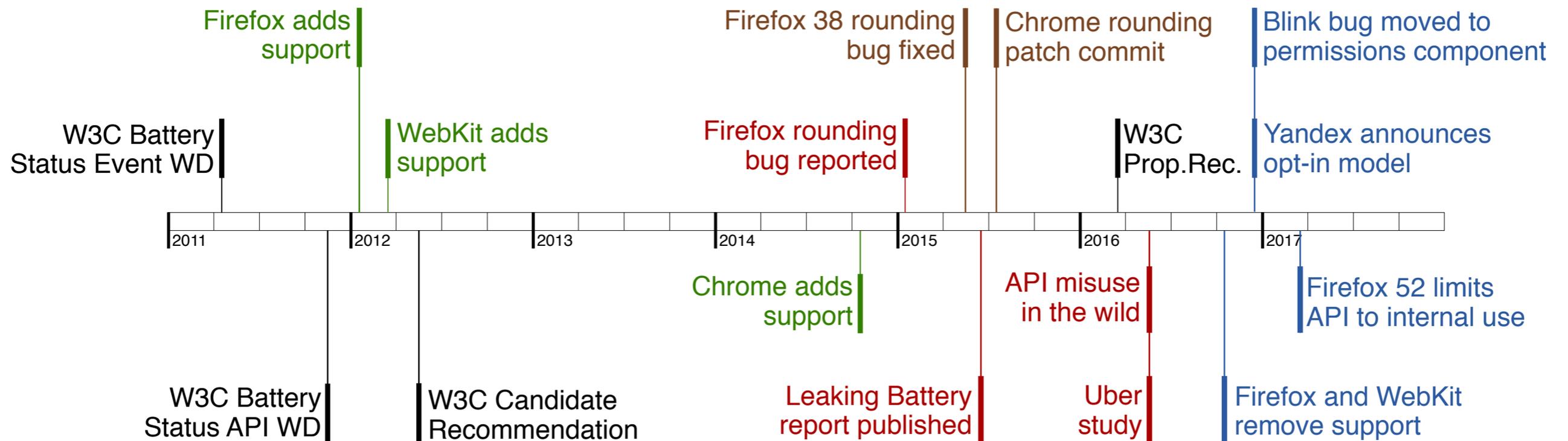
<https://www.w3.org/2011/07/privacy-ig-charter>

The Battery Status API

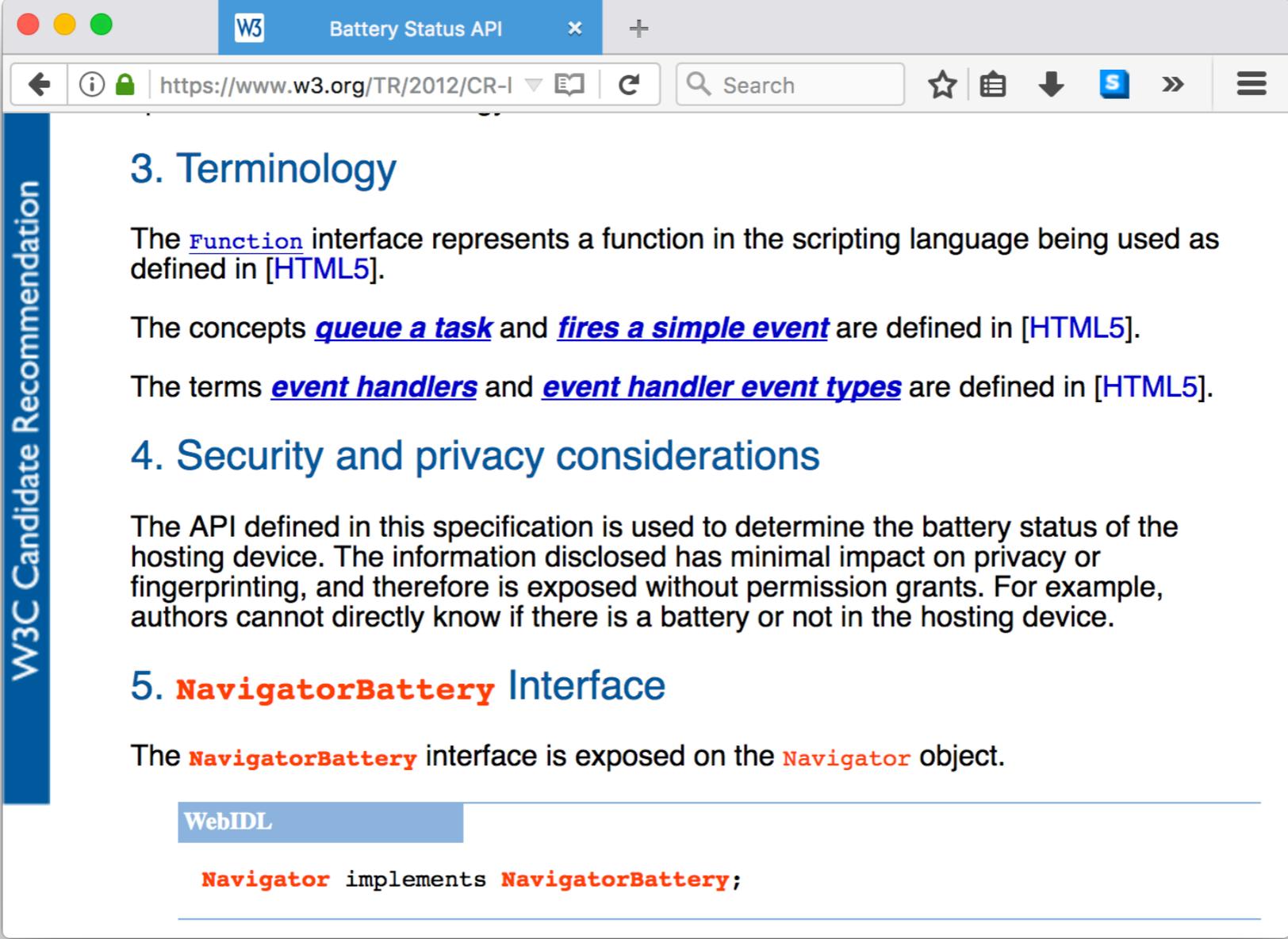


- charge **level**
 - value between 0 and 1
 - e.g 0.43 if the battery at 43%
- **charging** status
 - boolean indicator
- time to charge or discharge
 - **dischargingTime**
 - **chargingTime**
 - time in seconds

The development and adoption of the API



Mid 2012: Candidate Recommendation adds security and privacy considerations



W3C Candidate Recommendation

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

W3C Candidate Recommendation

3. Terminology

The `Function` interface represents a function in the scripting language 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

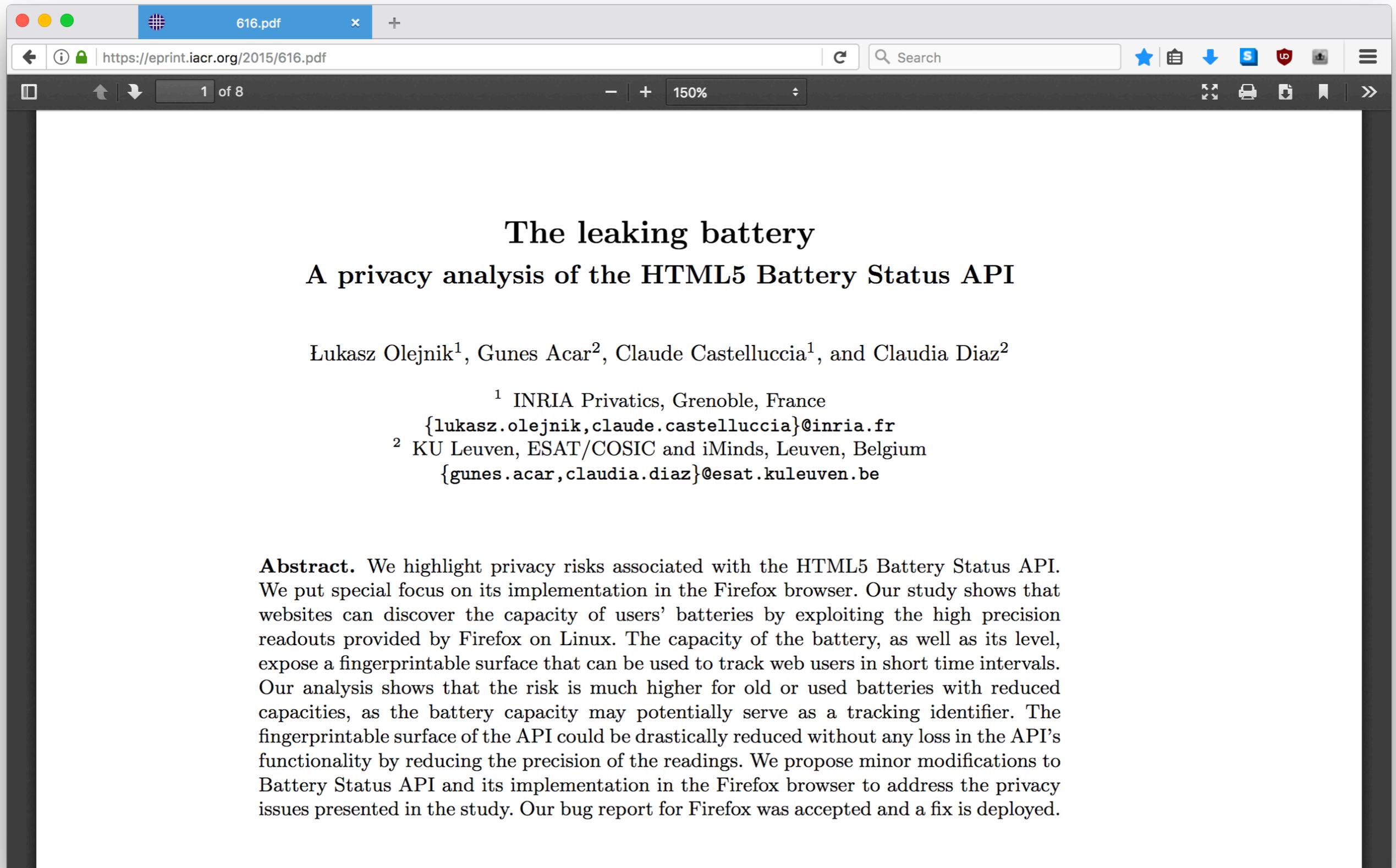
The `NavigatorBattery` interface is exposed on the `Navigator` object.

WebIDL

```
Navigator implements NavigatorBattery;
```

“the information disclosed has a **minimal impact** on privacy or fingerprinting”

New research exposes multiple privacy vulnerabilities



The screenshot shows a PDF viewer window with the following details:

- Browser tab: 616.pdf
- Address bar: <https://eprint.iacr.org/2015/616.pdf>
- Page indicator: 1 of 8
- Zoom level: 150%

The document content is as follows:

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,claudie.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

The screenshot shows a PDF viewer window with the following content:

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,claudie.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.

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

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

The image shows a web browser window displaying a PDF document. The browser's address bar shows the URL `https://senglehardt.com/papers/ccs16_online_tracking.pdf`. The document title is "Online Tracking: A 1-million-site Measurement and Analysis". The authors listed are Steven Englehardt (Princeton University, `ste@cs.princeton.edu`) and Arvind Narayanan (Princeton University, `arvindn@cs.princeton.edu`). A blue link states: "This is an extended version of our paper that appeared at ACM CCS 2016." The document content includes an abstract and the beginning of the introduction.

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

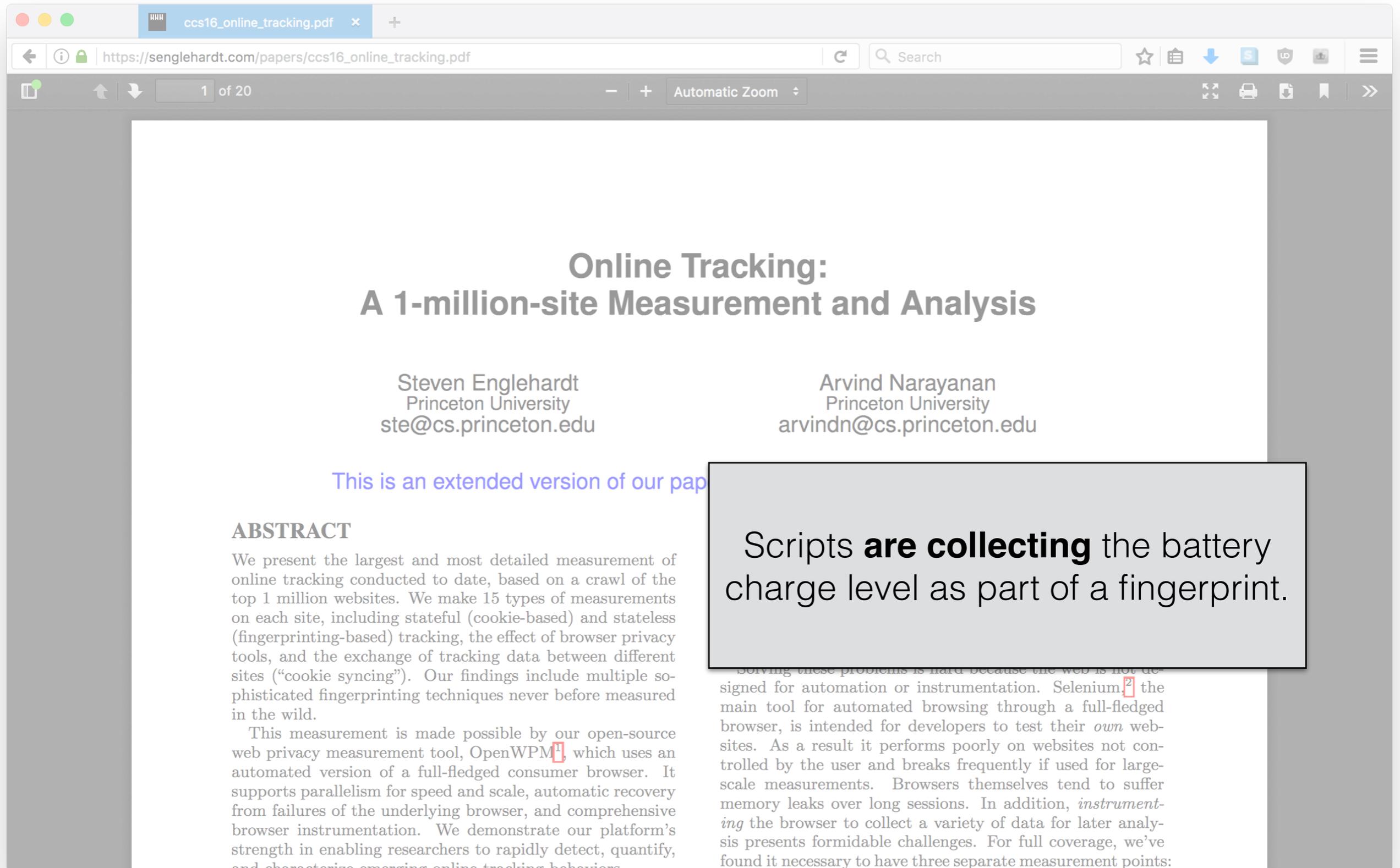
We present the largest and most detailed measurement of online tracking conducted to date, based on a crawl of the top 1 million websites. We make 15 types of measurements on each site, including stateful (cookie-based) and stateless (fingerprinting-based) tracking, the effect of browser privacy tools, 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, OpenWPM¹, 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

to resort to a stripped-down browser [31] (a limitation we explore in detail in Section 3.3). (2) We provide comprehensive instrumentation by expanding on the rich browser extension instrumentation of FourthParty [33], without requiring the researcher to write their own automation code. (3) We reduce duplication of work by providing a modular architecture to enable code re-use between studies.

Solving these problems is hard because the web is not designed for automation or instrumentation. Selenium², the main tool for automated browsing through a full-fledged browser, is intended for developers to test their *own* websites. As a result it performs poorly on websites not controlled by the user and breaks frequently if used for large-scale measurements. Browsers themselves tend to suffer memory leaks over long sessions. In addition, *instrumenting* the browser to collect a variety of data for later analysis presents formidable challenges. For full coverage, we’ve found it necessary to have three separate measurement points:

New research exposes multiple privacy vulnerabilities



The image shows a screenshot of a PDF viewer displaying the title page of a research paper. The browser's address bar shows the URL `https://sengehardt.com/papers/ccs16_online_tracking.pdf`. The document title is "Online Tracking: A 1-million-site Measurement and Analysis". The authors listed are Steven Englehardt and Arvind Narayanan, both from Princeton University. A blue link below the authors reads "This is an extended version of our paper". The abstract section begins with "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." A grey callout box with a black border is overlaid on the right side of the page, containing the text: "Scripts **are collecting** the battery charge level as part of a fingerprint." Below the callout, the text continues: "This measurement is made possible by our open-source web privacy measurement tool, OpenWPM¹, 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." At the bottom of the page, a paragraph starts with "Solving these problems is hard because the web is not designed for automation or instrumentation. Selenium², the main tool for automated browsing through a full-fledged browser, is intended for developers to test their own websites. As a result it performs poorly on websites not controlled by the user and breaks frequently if used for large-scale measurements. Browsers themselves tend to suffer memory leaks over long sessions. In addition, *instrumenting* the browser to collect a variety of data for later analysis presents formidable challenges. For full coverage, we've found it necessary to have three separate measurement points:"

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](#)

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, OpenWPM¹, 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.

Scripts **are collecting** the battery charge level as part of a fingerprint.

Solving these problems is hard because the web is not designed for automation or instrumentation. Selenium², the main tool for automated browsing through a full-fledged browser, is intended for developers to test their own websites. As a result it performs poorly on websites not controlled by the user and breaks frequently if used for large-scale measurements. Browsers themselves tend to suffer memory leaks over long sessions. In addition, *instrumenting* the browser to collect a variety of data for later analysis presents formidable challenges. For full coverage, we've found it necessary to have three separate measurement points:

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

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?”

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.”

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

Yandex.Browser

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?

The specification process should include
a privacy review of implementations

Specification requires two implementations to progress

The specification process should include
a privacy review of implementations

Specification requires two implementations to progress



Why not require a privacy review of these specifications?

The specification process should include
a privacy review of implementations

Specification requires two implementations to progress



Why not require a privacy review of these specifications?

—> Similar precision issues found during privacy review of Ambient Light Sensors API, which included implementation auditing.

API use in the wild should be audited after implementation

Trackers are the early adopters of any new API!

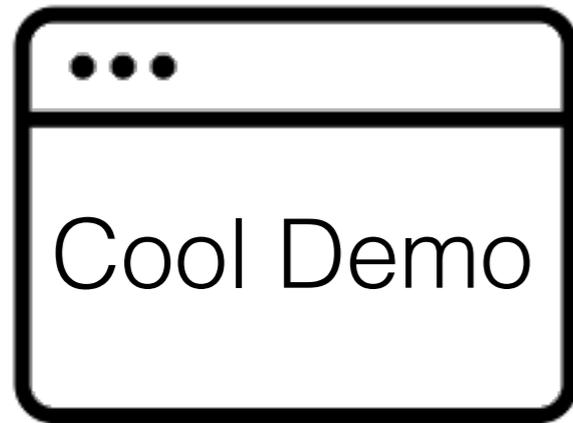
Trackers are the early adopters of any new API!

Intended Uses

Unintended Uses

Trackers are the early adopters of any new API!

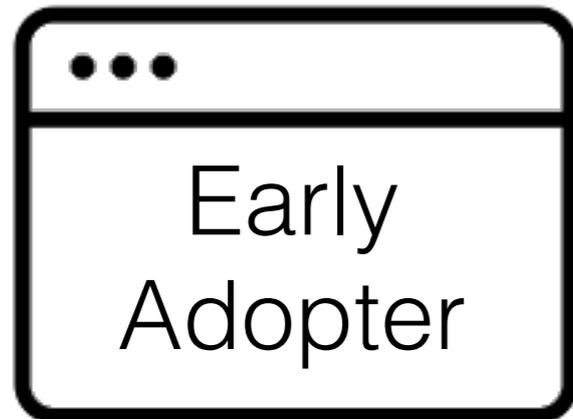
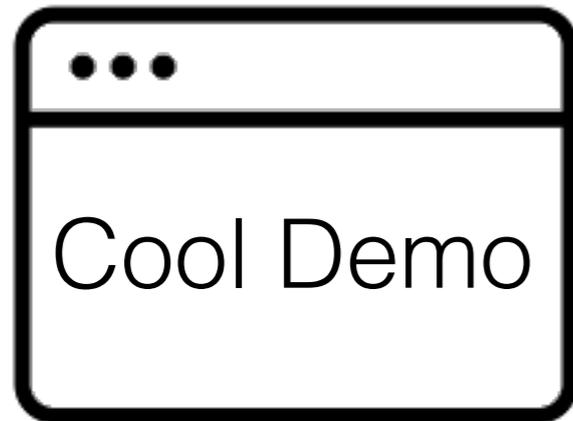
Intended Uses



Unintended Uses

Trackers are the early adopters of any new API!

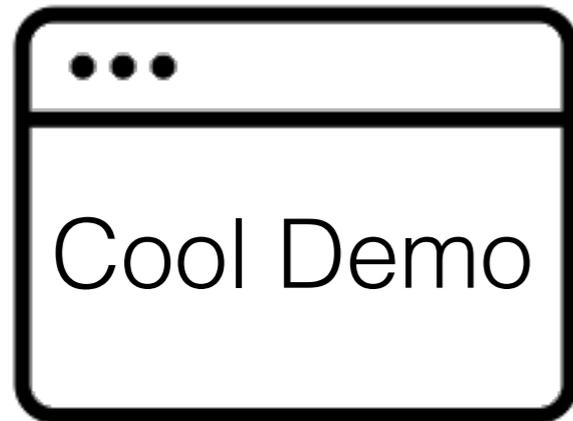
Intended Uses



Unintended Uses

Trackers are the early adopters of any new API!

Intended Uses

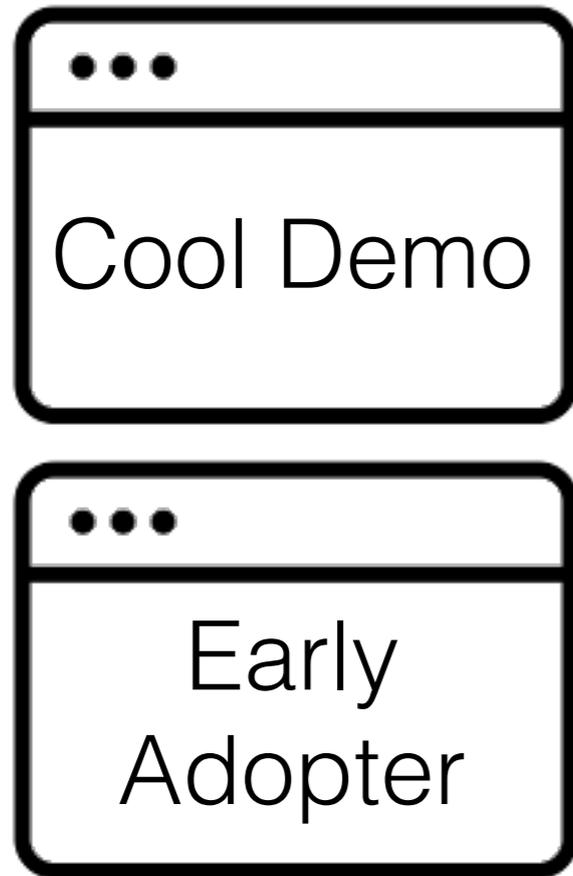


Unintended Uses

Implemented
by a tracker

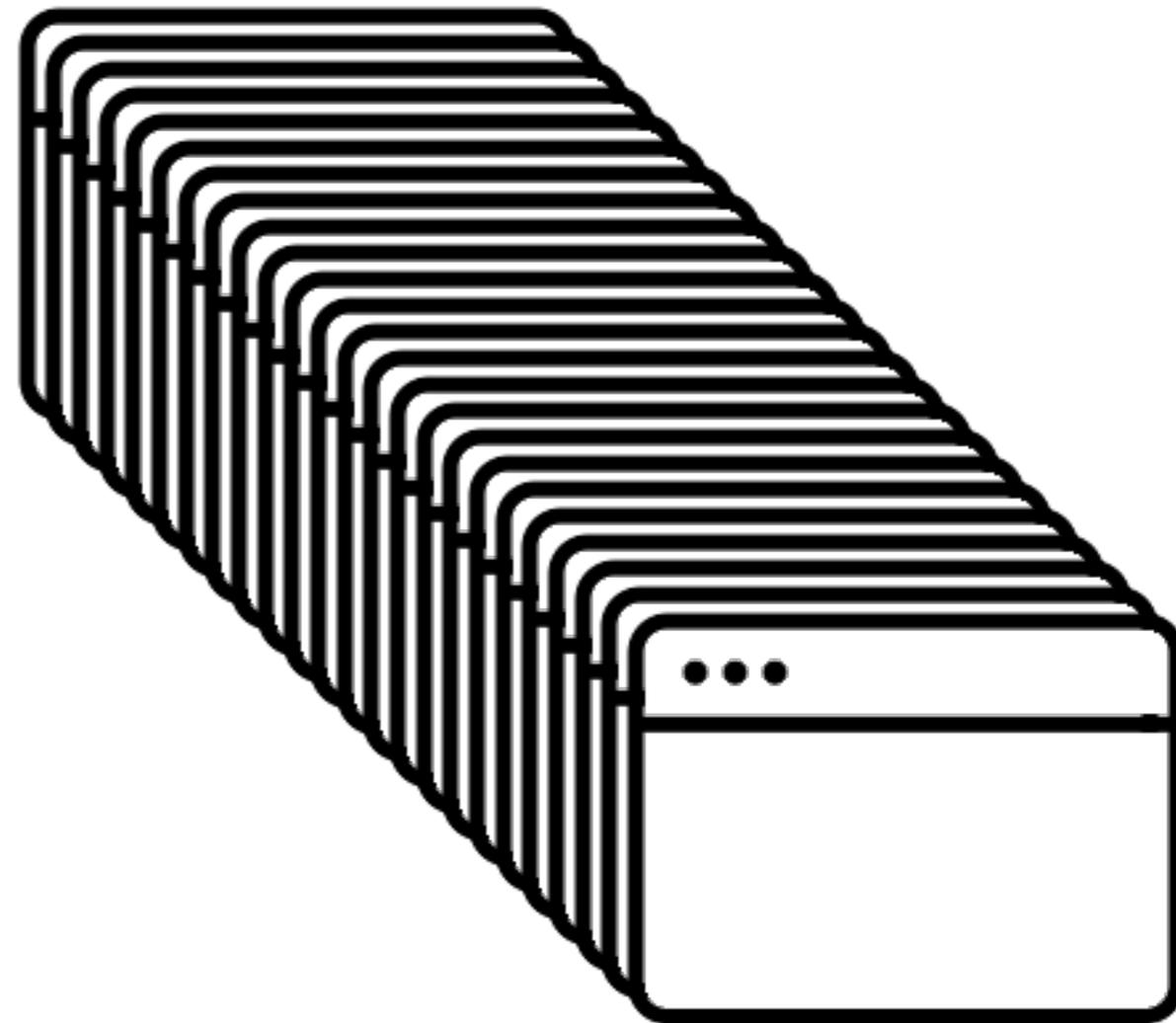
Trackers are the early adopters of any new API!

Intended Uses



Unintended Uses

Implemented
by a tracker



API use in the wild should be audited after implementation

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

API use in the wild should be audited after implementation

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

API use in the wild should be audited after implementation

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

Concerns:

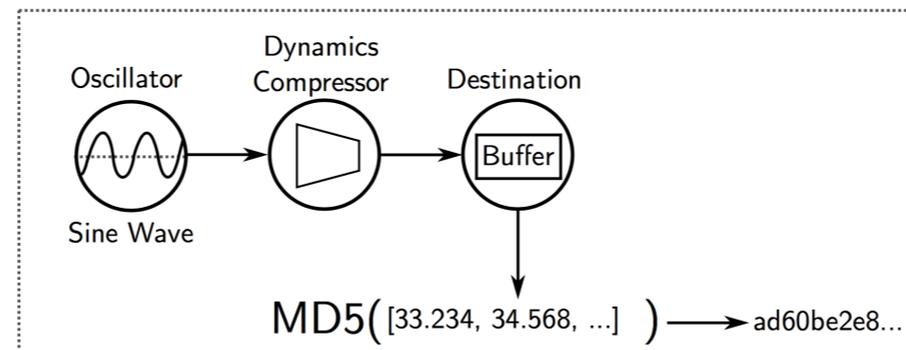
1. Lack of novelty in measurement techniques
2. Measurement of each new API is a small contribution
3. Specifications can't wait for the publication cycle

Suggestions:

1. Measurement through browser telemetry probes?
2. Regular measurement by browser vendors?
3. Public measurements by an NGO — something like archive.org

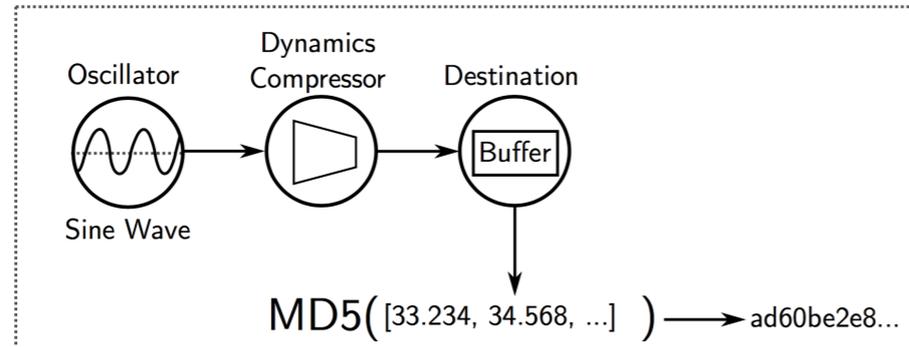
Specification authors should carry out privacy assessments with multiple threat models

An example: fingerprinting with the Audio API



Specification authors should carry out privacy assessments with multiple threat models

An example: fingerprinting with the Audio API

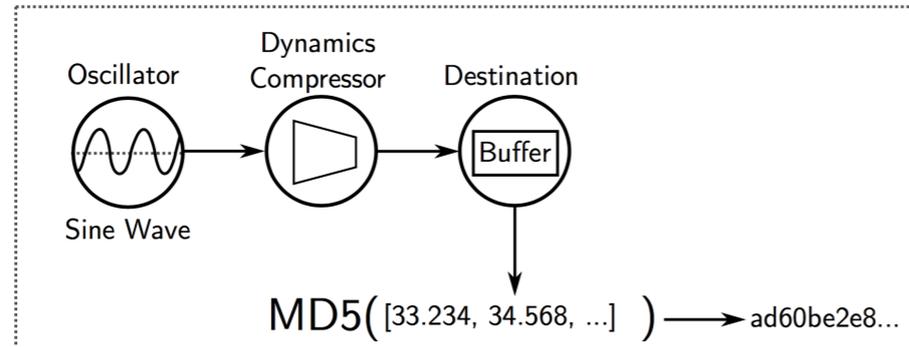


≈

User's OS and browser

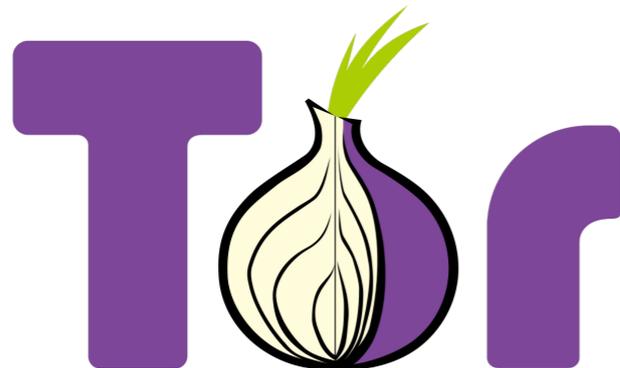
Specification authors should carry out privacy assessments with multiple threat models

An example: fingerprinting with the Audio API



≈

User's OS and browser



This is a concern for the Tor Browser!

Thank you!

In summary:

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

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