Privacy, Standards and Anti-Standards

Peter Snyder, Privacy Researcher, pes@brave.com

Overview

- Standards as a privacy focused implementor
- How the standards process makes privacy difficult (and how it can be fixed)
- Bonus concerns and conclusions

Overview

- **Standards as a privacy focused implementor**
- How the standards process makes privacy difficult (and how it can be fixed)
- Bonus concerns and conclusions

Privacy in Brave

- Tighter Default Storage Controls
- Tor Integration
- Resource Blocking
- Web API / DOM Modifications

Privacy in Brave

- Tighter Default Storage Controls
- Tor Integration
- Resource Blocking
- Web API / DOM Modifications

Web Standards / W3C / IETF

Browser Fingerprinting: A survey

PIERRE LAPERDRIX, CISPA Helmholtz Center for Information Security, Germany
NATALIIA BIELOVA, Inria Sophia Antipolis, France
BENOIT BAUDRY, KTH Royal Institute of Technology, Sweden
GILDAS AVOINE, Univ Rennes, INSA Rennes, CNRS, IRISA, France

With this paper, we survey the research performed in the domain of browser fingerprinting, while providing an accessible entry point to newcomers in the field. We explain how this technique works and where it stems from. We analyze the related work in detail to understand the composition of modern fingerprints and see how this technique is currently used online. We systematize existing defense solutions into different categories and detail the current challenges yet to overcome.

CCS Concepts: • **Security and privacy** → **Web application security**; **Browser security**; **Privacy protections**;

Additional Key Words and Phrases: Browser fingerprinting, user privacy, web tracking

1 INTRODUCTION

The web is a beautiful platform and browsers give us our entry point into it. With the introduction of HTML5 and CSS3, the web has become richer and more dynamic than ever and it has now the foundations to support an incredible ecosystem of diverse devices from laptops to smartphones and tablets. The diversity that is part of the modern web opened the door to device fingerprinting,

Browser Fingerpri

PIERRE LAPERDRIX, CISPA
NATALIIA BIELOVA, Inria S
BENOIT BAUDRY, KTH Roy
GILDAS AVOINE, Univ Renn

With this paper, we survey the re
an accessible entry point to newc
from. We analyze the related worl
this technique is currently used o
detail the current challenges yet

CCS Concepts: • **Security and pr**
tions;

Additional Key Words and Phras

1 INTRODUCTION

The web is a beautiful platforr
of HTML5 and CSS3, the web
foundations to support an in
and tablets. The diversity that

Table 4. Overview of four studies measuring adoption of browser fingerprinting on the web.

	Cookieless Monster [96] (2013)	FPDetective [69] (2013)	The Web Never Forgets [68] (2014)	1-million study with OpenWPM [78] (2016)
Fingerprinting techniques detected	Detection of 3 known fingerprinting libraries	JS-based and Flash-based font probing	Canvas fingerprinting	Canvas fingerprinting, canvas-based font probing, WebRTC and AudioContext
Sites crawled	10K sites (up to 20 pages per site)	1M sites (homepages) 100K sites (25 links per site) for JS 10K (homepages) for Flash	100K sites (homepages)	1M sites (homepages)
Prevalence	0.4%	0.04% (404 of 1M) for JS-based 1.45% (145 of 10K) for Flash-based	5.5%	1.4% for canvas fingerprinting 0.325% for canvas font probing 0.0715% for WebRTC 0.0067% for AudioContext
Detection method	Presence of JS libraries provided by BlueCava, Iovation and ThreatMetrix.	Logging calls of font probing methods. A script that loads more than 30 fonts or a Flash file that contains font enumeration calls is considered to perform fingerprinting.	Logging calls of canvas fingerprinting related methods. A script is considered to perform fingerprinting if it also checks other FP-related properties.	Logging calls of advanced FP-related JavaScript functions.

51v1 [cs.CR] 3 May 2019

Browser Fingerpri

PIERRE LAPERDRIX, CISPA
NATALIIA BIELOVA, Inria S
BENOIT BAUDRY, KTH Roy
GILDAS AVOINE, Univ Re

With this paper, we survey the re
an accessible entry point to new
from. We analyze the related wor
this technique is currently used o
detail the current challenges yet

CCS Concepts: • **Security and pr**
tions;

Additional Key Words and Phras

1 INTRODUCTION

The web is a beautiful platform
of HTML5 and CSS3, the web
foundations to support an inc
and tablets. The diversity that

Table 4. Overview of four studies measuring adoption of browser fingerprinting on the web.

	Cookieless Monster [96] (2013)	FPDetective [69] (2013)	The Web Never Forgets [68] (2014)	1-million study with OpenWPM [78] (2016)
Fingerprinting techniques detected	Detection of 3 known fingerprinting libraries	JS-based and Flash-based font probing	Canvas fingerprinting	Canvas fingerprinting, canvas-based font probing, WebRTC and AudioContext
Sites crawled	10K sites (up to 20 pages per site)	1M sites (homepages) 100K sites (25 links per site) for JS 10K (homepages) for Flash	100K sites (homepages)	1M sites (homepages)
Prevalence	0.4%	0.04% (404 of 1M) for JS-based 1.45% (145 of 10K) for Flash-based	5.5%	1.4% for canvas fingerprinting 0.325% for canvas font probing 0.0715% for WebRTC 0.0067% for AudioContext
Detection method	Presence of JS libraries provided by BlueCava, Iovation and ThreatMetrix.	Logging calls of font probing methods. A script that loads more than 30 fonts or a Flash file that contains font enumeration calls is considered to perform fingerprinting.	Logging calls of canvas fingerprinting related methods. A script is considered to perform fingerprinting if it also checks other FP-related properties.	Logging calls of advanced FP-related JavaScript functions.

Web API Modifications

Fingerprinting methods blocked in Fingerprinting Protection Mode

- Canvas fingerprinting: it should report a fixed value on tests like panopticlick
- WebGL fingerprinting: it should report as undefined on tests like panopticlick
- AudioContext fingerprinting
- WebRTC IP leakage
- SVG fingerprinting (specifically, the `SVGTextContentElement.prototype.getComputedTextLength` and `SVGPathElement.prototype.getTotalLength` methods)
- HSTS fingerprinting

Privacy protection enabled regardless of whether Fingerprinting Protection Mode is on

This list is not complete. See https://github.com/brave/brave-browser/wiki/Deviations-from-Chromium-(features-we-disable-or-remove) for other things which are disabled in Brave but not in Chrome.

Web API Modifications

Fingerprinting methods blocked in Fingerprinting Protection Mode

- Canvas fingerprinting: it should report a fixed value on tests like panopticlick
- WebGL fingerprinting: it should report as undefined on tests like panopticlick
- AudioContext fingerprinting
- WebRTC IP leakage
- SVG fingerprinting (specifically, the `SVGTextContentElement.prototype.getComputedTextLength` and `SVGPathElement.prototype.getTotalLength` methods)
- HSTS fingerprinting

Privacy protection enabled regardless of whether Fingerprinting Protection Mode is on

This list is not complete. See https://github.com/brave/brave-browser/wiki/Deviations-from-Chromium-(features-we-disable-or-remove) for other things which are disabled in Brave but not in Chrome.

Web Audio Fingerprinting

- Standard says websites can query hardware
- Hardware is pseudo-identifying
- Enough pseudo-identifiers yield a real identifier
- So Brave breaks the standard…

brave

Web Audio API

W3C Candidate Recommendation, 18 September 2018

This version:
https://www.w3.org/TR/2018/CR-webaudio-20180918/

Latest published version:
https://www.w3.org/TR/webaudio/

Editor's Draft:
https://webaudio.github.io/web-audio-api/

Previous Versions:
https://www.w3.org/TR/2018/WD-webaudio-20180619/
https://www.w3.org/TR/2015/WD-webaudio-20151208/
https://www.w3.org/TR/2013/WD-webaudio-20131010/
https://www.w3.org/TR/2012/WD-webaudio-20121213/
https://www.w3.org/TR/2012/WD-webaudio-20120802/
https://www.w3.org/TR/2012/WD-webaudio-20120315/
https://www.w3.org/TR/2011/WD-webaudio-20111215/

Feedback:
public-audio@w3.org with subject line "`[webaudio] … message topic …`" (archives)

Test Suite:
https://github.com/web-platform-tests/wpt/tree/master/webaudio

Issue Tracking:
GitHub

Editors:
Paul Adenot (Mozilla (https://www.mozilla.org/))
Raymond Toy (Google (https://www.google.com/))

Former Editors:
Chris Wilson (Until Jan 2016)
Chris Rogers (Until Aug 2013)

Bug Tracker:
https://github.com/WebAudio/web-audio-api/issues?state=open

Breaking Standards for Privacy

- Hardware Detection:
  - Web Audio
  - WebGL
  - WebUSB
  - Battery API
- Network Information
  - WebRTC
- Font Enumeration:
  - Canvas
  - SVG
- Display Information:
  - Client Hints
- Browsing History:
  - Referrer Policy

Overview

- Standards as a privacy focused implementor
- **How the standards process makes privacy difficult (and how it can be fixed)**
- Bonus concerns and conclusions

Three Standards
Privacy Anti-Patterns

F

s

1. Defined Functionality, Non-Normative Mitigations

Privacy Risk w/ Non-Normative Mitigations

- Privacy-harming / risky functionality
- “Privacy considerations" section, but non-standardized mitigation
- The Web assumes the dominant implementation, instead of the standard
- **Result:** Harm is “locked in” / out of control of the standards process

Referrer Policy

Editor's Draft, 20 April 2017

This version:

https://w3c.github.io/webappsec-referrer-policy/

Latest published version:

http://www.w3.org/TR/referrer-policy/

Version History:

https://github.com/w3c/webappsec-referrer-policy/commits/master/index.src.html

Feedback:

public-webappsec@w3.org with subject line "`[referrer-policy] … message topic …`" (archives)

Issue Tracking:

GitHub

Inline In Spec

Editors:

Jochen Eisinger (Google Inc.)

Emily Stark (Google Inc.)

Tests:

web-platform-tests referrer-policy/ (ongoing work)

Abstract

§ 1. Introduction

This section is not normative.

Requests made from a document, and for navigations away from that document are associated with a `Referer` header. While the header can be suppressed for links with the `noreferrer` link type, authors might wish to control the `Referer` header more directly for a number of reasons:

§ 1.1. Privacy

A social networking site has a profile page for each of its users, and users add hyperlinks from their profile page to their favorite bands. The social networking site might not wish to leak the user's profile URL to the band web sites when other users follow those hyperlinks (because the profile URLs might reveal the identity of the owner of the profile).

Some social networking sites, however, might wish to inform the band web sites that the links originated from the social networking site but not reveal which specific user's profile contained the links.

§ 1.2. Security

A web application uses HTTPS and a URL-based session identifier. The web application might wish to link to HTTPS resources on other web sites without leaking the user's session identifier in the URL.

Alternatively, a web application may use URLs which themselves grant some capability. Controlling the referrer can help prevent these capability URLs from leaking via referrer headers. [CAPABILITY-URLS]

Note that there are other ways for capability URLs to leak, and controlling the referrer is not enough to control all those potential leaks.

§ 1.3. Trackback

A blog hosted over HTTPS might wish to link to a blog hosted over HTTP and receive trackback links.

3. Set *url*'s username to the empty string.
4. Set *url*'s password to `null`.
5. Set *url*'s fragment to `null`.
6. If the `origin-only flag` is `true`, then:
    1. Set *url*'s path to `null`.
    2. Set *url*'s query to `null`.
7. Return *url*.

§ 9. Privacy Considerations

§ 9.1. User Controls

Nothing in this specification should be interpreted as preventing user agents from offering options to users which would change the information sent out via a `Referer` header. For instance, user agents MAY allow users to suppress the referrer header entirely, regardless of the active referrer policy on a page.

§ 10. Security Considerations

§ 10.1. Information Leakage

The referrer policies `"origin"`, `"origin-when-cross-origin"` and `"unsafe-url"` might leak the origin and the URL of a secure site respectively via insecure transport.

Those three policies are included in the spec nevertheless to lower the friction of sites adopting secure transport.

Authors wanting to ensure that they do not leak any more information than the default policy should instead use the policy states `"same-origin"`, `"strict-origin"`, `"strict-origin-when-cross-origin"` or `"no-referrer"`.

Result

- Well described functionality
- Vaguely / undefined / unclear mitigations
- Web assumes the defined functionality, privacy-harm gets locked in
- **Solution:** Make mitigations normative and standardized!

1. Defined Functionality, Non-Normative Mitigations

2. Uncommon Use Case, Common Availability

Uncommon Use Case, Common Availability

- Genuinely useful functionality, for niche scenarios
- Functionality is made widely available (first-party, third-party, frames, etc.)
- Co-opted by tracking, code-paths assume availability
- **Result:** can't be removed, even from irrelevant sites

HTML

Living Standard — Last Updated 10 May 2019

← 4.12 Scripting — Table of Contents — 4.13 Custom elements →

The **`toDataURL(type, quality)`** method, when invoked, must run these steps:

MDN ▸ HTMLCanvasElement/toDataURL

1. If this `canvas` element's bitmap's origin-clean flag is set to false, then throw a `"SecurityError" DOMException`.
2. If this `canvas` element's bitmap has no pixels (i.e. either its horizontal dimension or its vertical dimension is zero) then return the string "`data:,`". (This is the shortest `data:` URL; it represents the empty string in a `text/plain` resource.)
3. Let *file* be a serialization of this `canvas` element's bitmap as a file, passing *type* and *quality* if given.
4. If *file* is null then return "`data:,`".
5. Return a `data:` URL representing *file*. [RFC2397]

The **`toBlob(callback, type, quality)`** method, when invoked, must run these steps:

MDN ▸ HTMLCanvasElement/toBlob

1. If this `canvas` element's bitmap's origin-clean flag is set to false, then throw a `"SecurityError" DOMException`.
2. Let *result* be null.
3. If this `canvas` element's bitmap has pixels (i.e., neither its horizontal dimension nor its vertical dimension is zero), then set *result* to a copy of this `canvas` element's bitmap.
4. Run these steps in parallel:
    1. If *result* is non-null, then set *result* to a serialization of *result* as a file with *type* and *quality* if given.
    2. Queue a task to run these steps:
        1. If *result* is non-null, then set *result* to a new `Blob` object, created in the relevant Realm of this `canvas` element, representing *result*. [FILEAPI]
        2. Invoke *callback* with « *result* ».

    The task source for this task is the **canvas blob serialization task source**.

Search or jump to... / Pull requests Issues Marketplace Explore

Valve / fingerprintjs2

Watch 390 | Star 7,724 | Fork 1,094

Code | Issues 58 | Pull requests 1 | Projects 0 | Wiki | Insights

Modern & flexible browser fingerprinting library https://fingerprintjs.com

javascript detection identification fingerprint fraud-detection fraud audio-fingerprinting

428 commits | 5 branches | 55 releases | 55 contributors | View license

Branch: master | New pull request | Create new file | Upload files | Find File | Clone or download

Valve Update README.md — Latest commit 640928b 27 days ago

.github	Create pull_request.md	6 months ago
flash	Simplify and refactor font enumeration code:	4 years ago
tests	Add more specs	a month ago
.eslintrc	[headless-chrome] starting the migration	3 months ago
.gitignore	gitignore dist/	10 months ago
.travis.yml	[headless-chrome] finilize the migration to Chrome Headless testing	3 months ago
LICENSE	Update LICENSE	9 months ago
README.md	Update README.md	27 days ago
bower.json	[headless-chrome] starting the migration	3 months ago
fingerprint2.js	Add more specs	a month ago
gulpfile.js	Fix release	7 months ago
index.html	Remove Google Analytics script from index.html (#449)	a month ago

```
var getCanvasFp = function (options) {
  var result = []
  // Very simple now, need to make it more complex (geo shapes etc)
  var canvas = document.createElement('canvas')
  canvas.width = 2000
  canvas.height = 200
  canvas.style.display = 'inline'
  var ctx = canvas.getContext('2d')
  // detect browser support of canvas winding
  // http://blogs.adobe.com/webplatform/2013/01/30/winding-rules-in-canvas/
  // https://github.com/Modernizr/Modernizr/blob/master/feature-detects/canvas/winding
  ctx.rect(0, 0, 10, 10)
  ctx.rect(2, 2, 6, 6)
  result.push('canvas winding:' + ((ctx.isPointInPath(5, 5, 'evenodd') === false) ? 'y

  ctx.textBaseline = 'alphabetic'
  ctx.fillStyle = '#f60'
  ctx.fillRect(125, 1, 62, 20)
  ctx
  Cmd + click to follow link
  // https://github.com/Valve/fingerprintjs2/issues/66
  if (options.dontUseFakeFontInCanvas) {
    ctx.font = '11pt Arial'
  } else {
    ctx.font = '11pt no-real-font-123'
  }
  ctx.fillText('Cwm fjordbank glyphs vext quiz, \ud83d\ude03', 2, 15)
  ctx.fillStyle = 'rgba(102, 204, 0, 0.2)'
  ctx.font = '18pt Arial'
  ctx.fillText('Cwm fjordbank glyphs vext quiz, \ud83d\ude03', 4, 45)

  // canvas blending
  // http://blogs.adobe.com/webplatform/2013/01/28/blending-features-in-canvas/
  // http://jsfiddle.net/NDYV8/16/
  ctx.globalCompositeOperation = 'multiply'
  ctx.fillStyle = 'rgb(255,0,255)'
  ctx.beginPath()
  ctx.arc(50, 50, 50, 0, Math.PI * 2, true)
  ctx.closePath()
```

```
  } else {
    ctx.font = '11pt no-real-font-123'
  }
  ctx.fillText('Cwm fjordbank glyphs vext quiz, \ud83d\ude03', 2, 15)
  ctx.fillStyle = 'rgba(102, 204, 0, 0.2)'
  ctx.font = '18pt Arial'
  ctx.fillText('Cwm fjordbank glyphs vext quiz, \ud83d\ude03', 4, 45)

  // canvas blending
  // http://blogs.adobe.com/webplatform/2013/01/28/blending-features-in-canvas/
  // http://jsfiddle.net/NDYV8/16/
  ctx.globalCompositeOperation = 'multiply'
  ctx.fillStyle = 'rgb(255,0,255)'
  ctx.beginPath()
  ctx.arc(50, 50, 50, 0, Math.PI * 2, true)
  ctx.closePath()
  ctx.fill()
  ctx.fillStyle = 'rgb(0,255,255)'
  ctx.beginPath()
  ctx.arc(100, 50, 50, 0, Math.PI * 2, true)
  ctx.closePath()
  ctx.fill()
  ctx.fillStyle = 'rgb(255,255,0)'
  ctx.beginPath()
  ctx.arc(75, 100, 50, 0, Math.PI * 2, true)
  ctx.closePath()
  ctx.fill()
  ctx.fillStyle = 'rgb(255,0,255)'
  // canvas winding
  // http://blogs.adobe.com/webplatform/2013/01/30/winding-rules-in-canvas/
  // http://jsfiddle.net/NDYV8/19/
  ctx.arc(75, 75, 75, 0, Math.PI * 2, true)
  ctx.arc(75, 75, 25, 0, Math.PI * 2, true)
  ctx.fill('evenodd')

  if (canvas.toDataURL) { result.push('canvas fp:' + canvas.toDataURL()) }
  return result
}
```

Browser Characteristic	bits of identifying information	one in x browsers have this value	value
User Agent	13.54	11932.41	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.91 Safari/537.36
HTTP_ACCEPT Headers	3.15	8.87	text/html, */*; q=0.01 gzip, deflate, br en-US,en;q=0.9
Browser Plugin Details	0.91	1.88	undefined
Time Zone	4.22	18.66	420
Screen Size and Color Depth	5.49	44.81	1680x1050x24
System Fonts	3.9	14.89	Andale Mono, Arial, Arial Black, Arial Hebrew, Arial Narrow, Arial Rounded MT Bold, Arial Unicode MS, Comic Sans MS, Courier, Courier New, Geneva, Georgia, Helvetica, Helvetica Neue, Impact, LUCIDA GRANDE, Microsoft Sans Serif, Monaco, Palatino, Tahoma, Times, Times New Roman, Trebuchet MS, Verdana, Wingdings, Wingdings 2, Wingdings 3 (via javascript)
Are Cookies Enabled?	0.27	1.21	Yes
Limited supercookie test	0.4	1.32	DOM localStorage: Yes, DOM sessionStorage: Yes, IE userData: No
Hash of canvas fingerprint	5.68	51.1	cf04c1dcb26ef79705764e5c22d0e711
Hash of WebGL fingerprint	3.89	14.78	undetermined
DNT Header Enabled?	1.24	2.37	False
Language	1.0	1.99	en-US
Platform	3.26	9.59	MacIntel
Touch Support	0.76	1.7	Max touchpoints: 0; TouchEvent supported: false; onTouchStart supported: false

- Widely Available
- Sites / benign code expects
- Removing / blocking breaks benign sites

Lots of rare-use-case functionality

- Brightness sensors
- WebVR
- Machine Learning APIs
- High Resolution Timers
- Vibration
- WebGL operations
- Tracing APIs
- Many many many more…

Lesson Learned

- Assume people will find bad uses for your functionality
- General access -> difficult to remove / modify
- **Solution:** Restrict access to the use cases you care about
  - User gestures
  - Permission prompts
  - Not-in-frames

1. Defined Functionality, Non-Normative Mitigations

2. Uncommon Use Case, Common Availability

3. “No worse than the status quo”

“No worse than the status quo”

- Privacy-harming / risky functionality
- “Information is available elsewhere, so no additional harm”
- **Result:** Web compat difficulty expands…

HTTP Working Group
Internet-Draft
Intended status: Experimental
Expires: November 11, 2019

I. Grigorik
Google
May 10, 2019

HTTP Client Hints

draft-ietf-httpbis-client-hints-07

Abstract

HTTP defines proactive content negotiation to allow servers to select the appropriate response for a given request, based upon the user agent's characteristics, as expressed in request headers. In practice, clients are often unwilling to send those request headers, because it is not clear whether they will be used, and sending them impacts both performance and privacy.

This document defines two response headers, Accept-CH and Accept-CH-Lifetime, that servers can use to advertise their use of request headers for proactive content negotiation, along with a set of guidelines for the creation of such headers, colloquially known as "Client Hints."

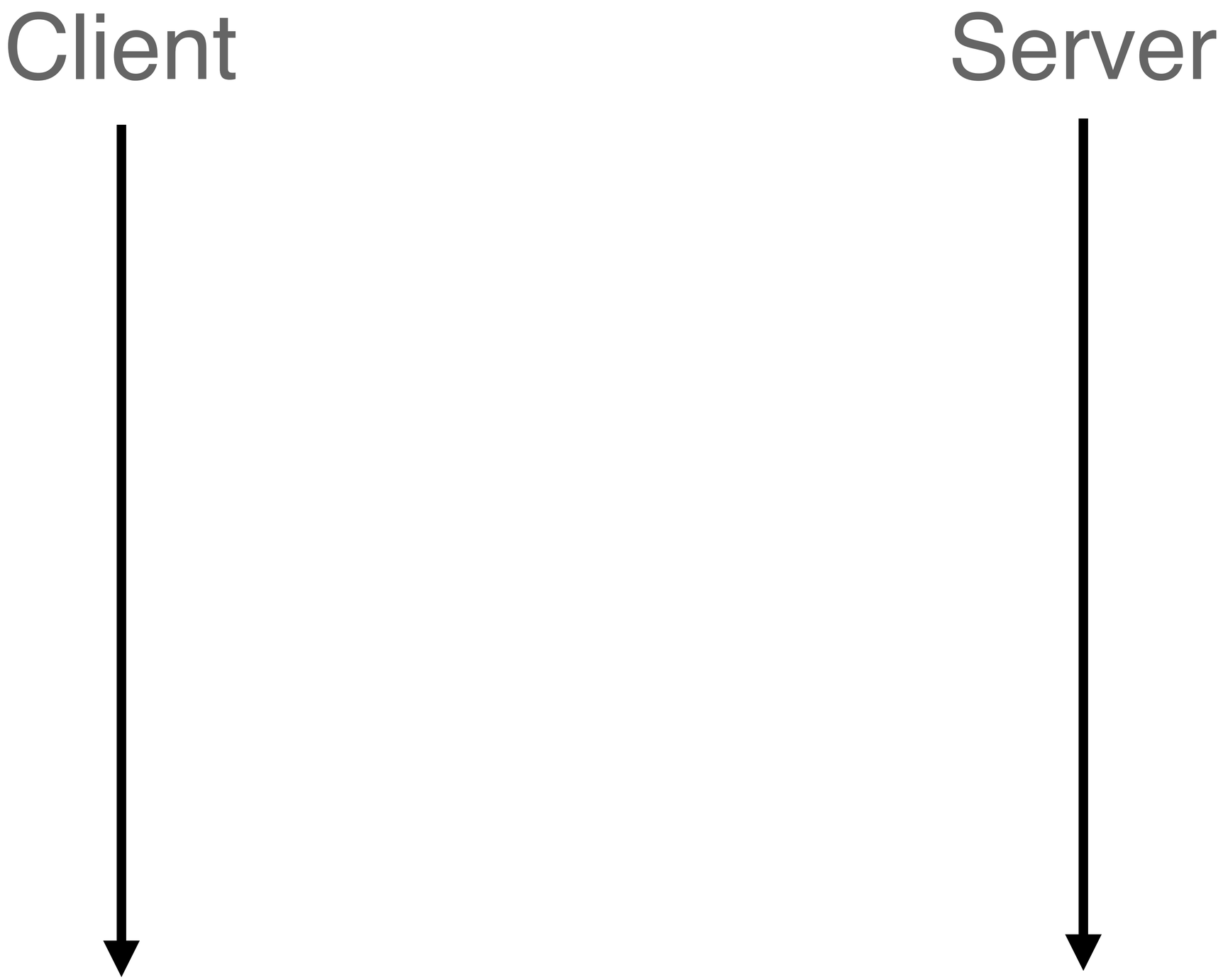
Note to Readers

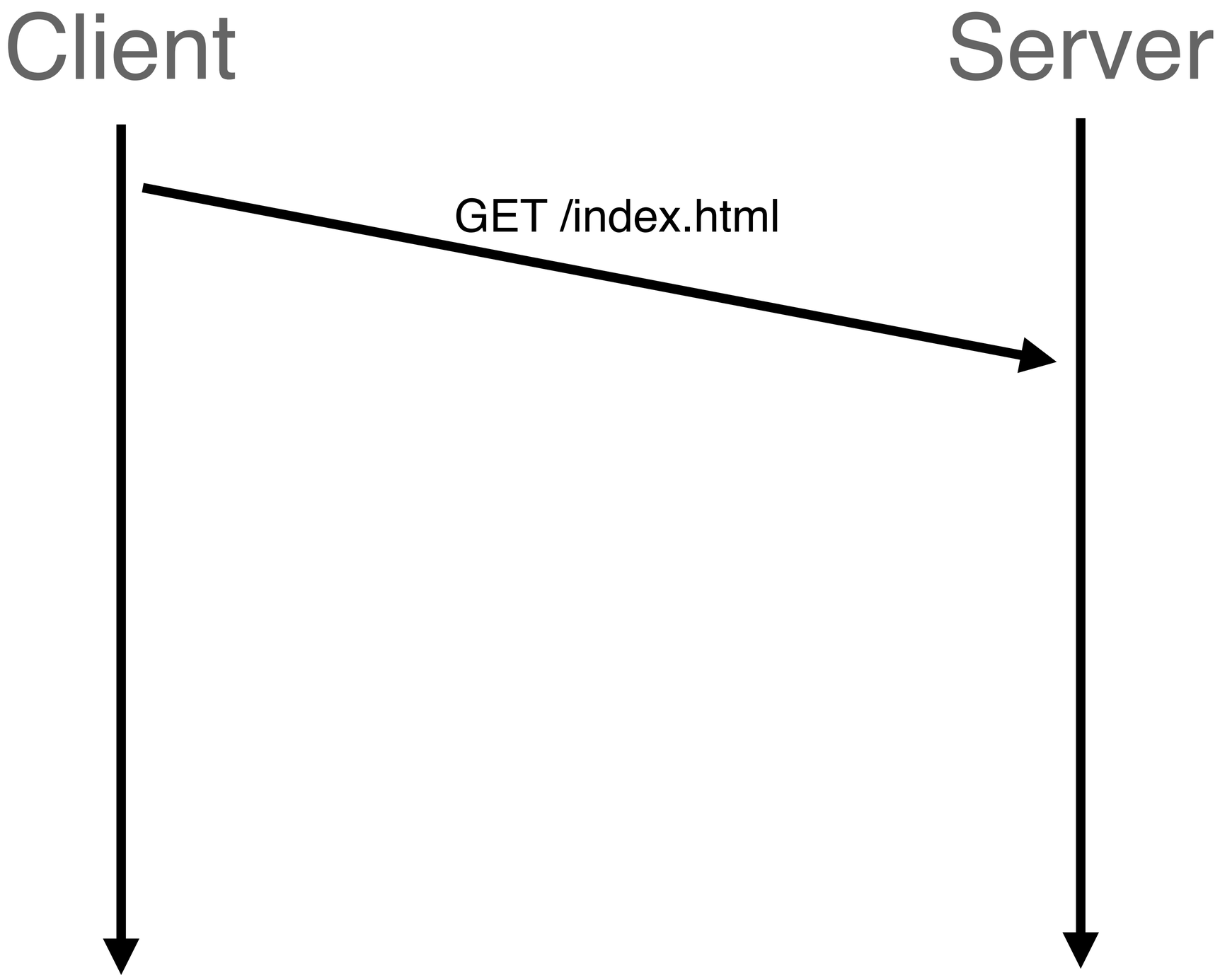
Discussion of this draft takes place on the HTTP working group mailing list (ietf-http-wg@w3.org), which is archived at https://lists.w3.org/Archives/Public/ietf-http-wg/.

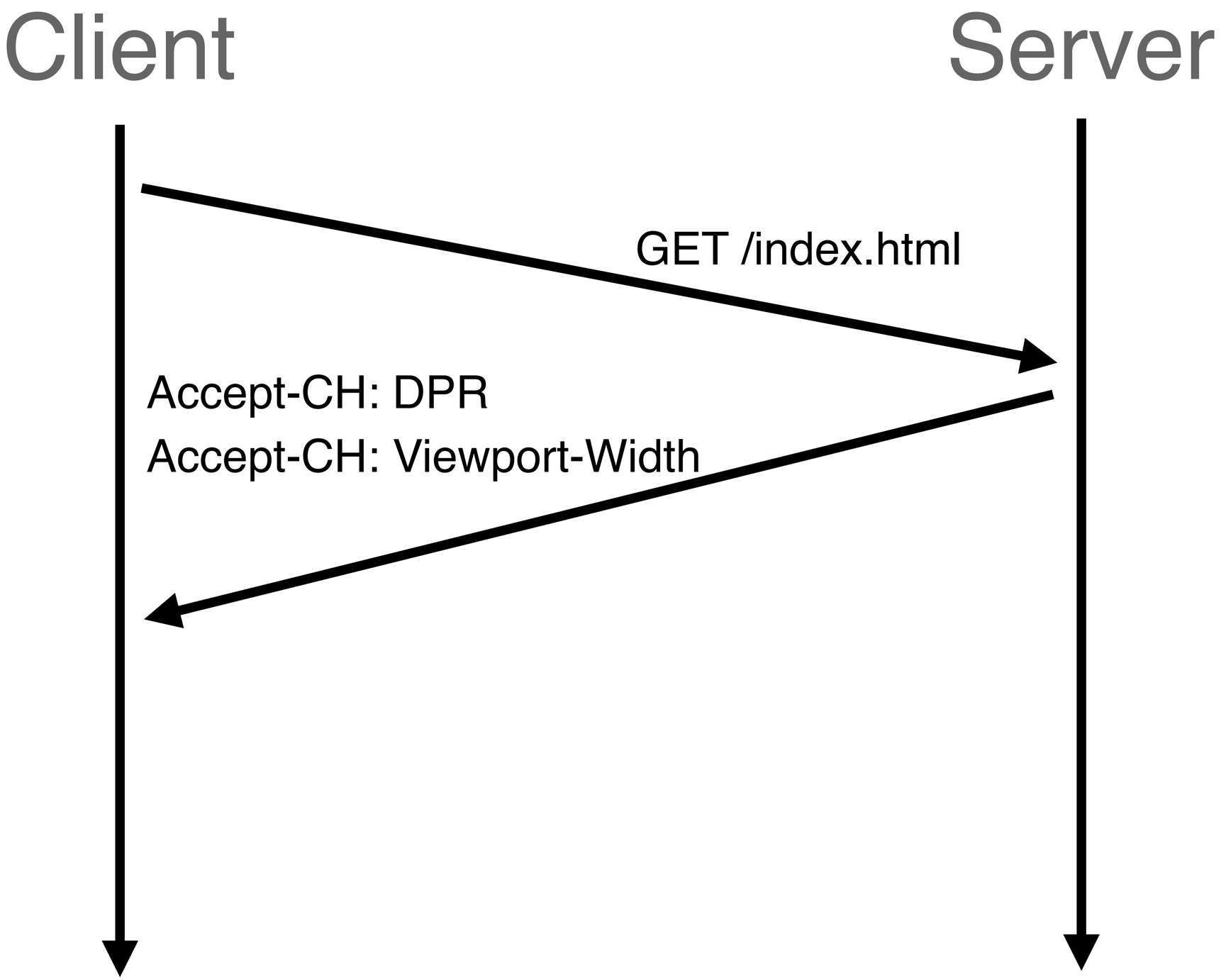
Working Group information can be found at http://httpwg.github.io/; source code and issues list for this draft can be found at https://github.com/httpwg/http-extensions/labels/client-hints.

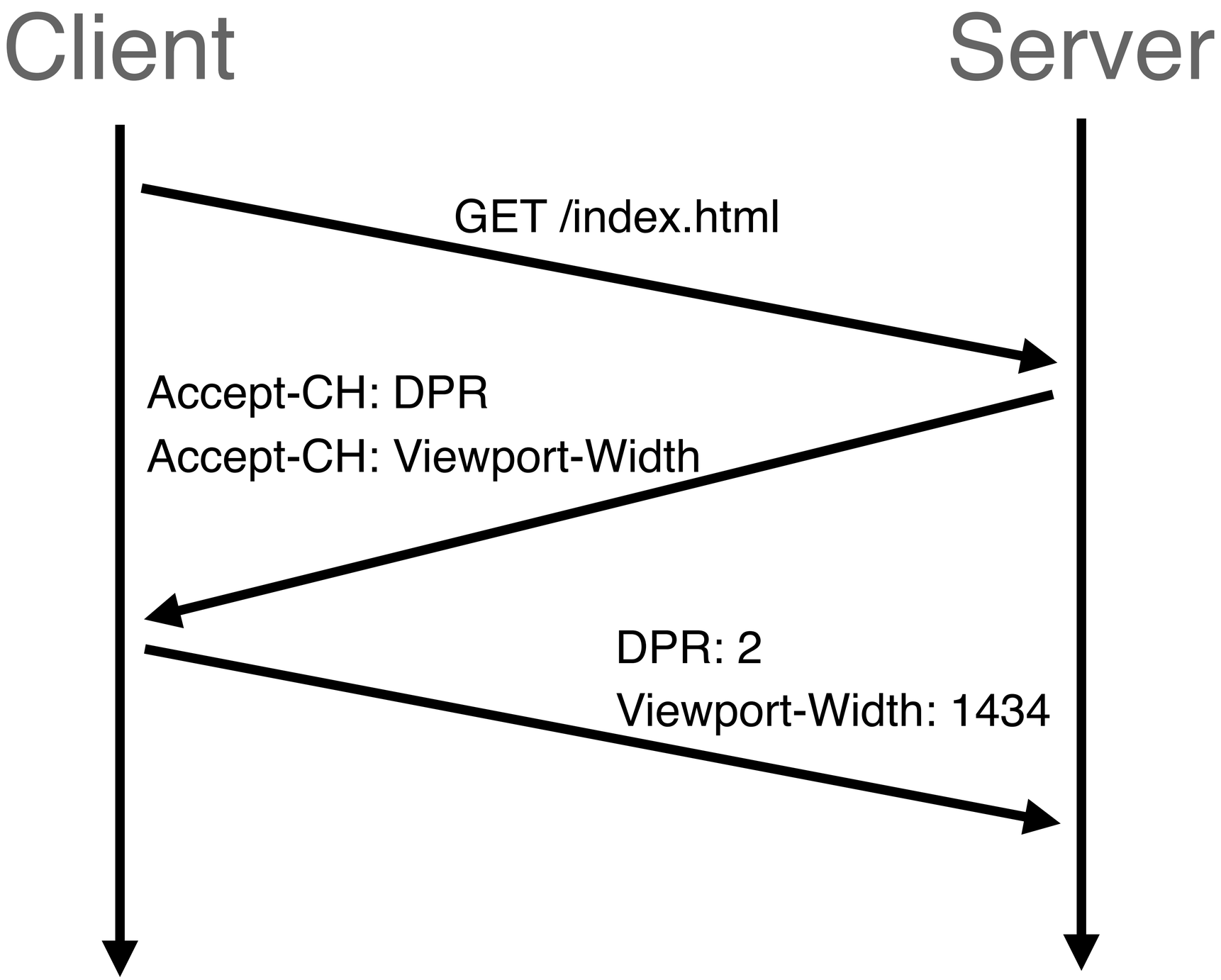
Status of this Memo

Table of Contents


Client
Server


Client
Server
GET /index.html


Client
Server
GET /index.html
Accept-CH: DPR
Accept-CH: Viewport-Width


Client
Server
GET /index.html
Accept-CH: DPR
Accept-CH: Viewport-Width
DPR: 2
Viewport-Width: 1434

Values in Client Hints are Identifying

- **Eckersley, Peter. "How unique is your web browser?." PETS 2010**
  Viewport height and width

- **Laperdrix et al. "Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints." *S&P* 2016.**
  Device color depth

- **Englehardt et al. "Online Tracking: A 1-million-site Measurement and Analysis." CCS 2016**
  The above are being used often!

Client Hints Authors' Current Position

- This information is already available

- No further exposure / no marginal harm

- **Brave's Concerns with the Client-Hints Proposal**
  https://brave.com/brave-and-client-hints/


WE'LL DIG OUR WAY OUT!

Lesson Learned

- "Horizontal" privacy risk is technological debt
- Same data in more places entrenches the risk
- **Solution:** Treat all additional privacy risk as equally problematic

Overview

- Standards as a privacy focused implementor
- How the standards process makes privacy difficult (and how it can be fixed)
- **Bonus concerns and conclusions**

Bonus anti-patterns

- “This privacy concern is addressed by an upcoming standard…”
- “This just formalizes existing bad practice…”
- "Site owners want it, users like sites, so by the transitive property…”

Bonus suggestions / concerns / worries / rants

- Pump the breaks on everything
- Complexity is a privacy risk
- Amount of “standards” work that is shipped-than-standardized

Overview

- Standards as a privacy focused implementor
- How the standards process makes privacy difficult (and how it can be fixed)
- **Bonus concerns and conclusions**

Conclusion

- Privacy preserving standards are important to improving the Web.
- Weak standards make it difficult for privacy-interested parties to improve things.
- A few small changes to privacy criteria in standards would make a huge difference.

- **Pete Snyder**
  Privacy Researcher
  pes@brave.com