Is preventing browser fingerprinting a lost cause?

Obligations of W3C specification authors and reviewers to preserve passive privacy properties of the user agent.

What are we talking about?

- You call it “fingerprinting”
- I call it “feature detection”

- “Passive privacy”

- Different from active/stateful tracking
  - Cookies, HTML5 storage, FSOs
  - Even “HSTS supercookies”

Panopticlick

- http://panopticlick.eff.org/
- User Agent, HTTP_ACCEPT headers, plugin details, time zone, screen size, color depth, system fonts, cookies / local storage
- ~60 bits for my Win 7 + IE 9 test
- Unique in 2.5 million

BrowserSpy.dk

Accepted Filetypes
ActiveX
Adobe Reader
Ajax Support
Bandwidth
Browser
Capabilities
Colors
Components
Connections
Cookies
CPU
CSS
CSS Exploit
Cursors
Date and Time
DirectX
Document
Do No Track
.NET Framework
Email Verification
Flash
Fonts via Flash
Fonts via Java
Gears

Gecko
Geolocation
Google Chrome
Google Apps
GZip Support
HTTP Headers
HTTP
Images
IP Address
Java
JavaScript
Languages
Mathematical
MathML Support
MIME Types
Mobile
Network
Objects
Object Browser
Online / Offline
OpenDNS
OpenOffice.org
Opera Browser
Operating System
Google PageRank

HTTP Password
Ping
Plugins
Plugs
Prefetch
Proxy
Personal Security Manager
QuickTime Player
RealPlayer
Resolution
Screen
Security
Shockwave
Silverlight
Sound Card
SVG
Text Formatting
File Upload
UserAgent
VBScript
WAP Device
WebKit
Web Server
Window
Windows Media Player

Even deeper…

- http://cseweb.ucsd.edu/~hovav/papers/mbys11.html

- Fingerprinting Information in JavaScript Implementations
  - Micro-performance benchmarks

DIFFICULT ASYMMETRIES

User Constituency vs. Spec Author Constituency

Lots of them, few of us, we should take on necessary extra work in their interest.

Ease of Circumvention vs. Ease of Securing

Very hard to prevent fingerprinting, easy to find holes and widely disseminate knowledge or even productize such evasions, undoing the work of conscientious authors.

Ian Goldberg’s “Privacy Ratchet”

Easy to take away privacy, hard to add it back in.

Paradox of User-Controlled Privacy Technology

Fine-grained settings or incomplete tools used by a limited population can make users of these settings and tools easier to track.

Fingerprint bandwidth / Anonymity set size

- OS
  - version
  - sub-version, patch level
  - personalization (e.g. fonts)
- Browser
  - version
  - personalization
  - installed plugins
  - installed plugin versions
  - plugin personalization (e.g. NoScript whitelist)

Where to draw the line?

- There may be things we cannot prevent fingerprinting:
  - browser make and version, OS make and version
- We advertise them deliberately!
  - In the UA string, in DOM APIs
- Millions of lines of code and thousands of API points
  - *And you get to run arbitrary code inside them and communicate to the world*
- There will **ALWAYS** be detectable differences
  - Even if users change their UA string

But even these “public” characteristics can be differentiators that lead to unique identification when combined with other information like IP address

Where is the line today?

- Technology that has a privacy impact due to state maintenance typically addresses this in the spec, and UAs typically address this in implementations
  - Though often not the case for private plugins
- That features can be used as part of fingerprinting a browser version is not typically addressed as a privacy issue

Threat Models

- Who are we trying to defend against?
  - "Casual" commercial parties
    - Individual sites writing their own code
    - Limited incentive to invest in unique research
    - Satisfied with an opt-out regime that lets them track many/most users
  - Dedicated commercial parties
    - Selling a tracking system
    - Commercial motivation to invest in arms-race
  - State-level actors
    - May make much larger research investments than would be commercially justified, keep such research private
- What are we willing to promise?
  - Consequences of failure are VERY different in these scenarios

Realistic Threat Profiles Needed For Browser as a Whole

- We saw this in privacy objections to CSP
  - A bad faith actor could abuse CSP's reporting to "phone home" – opt-in requested
  - But, if acting in bad faith, could do the same with <img>, <form>, <href>, script, etc…
- No feeling in WG that closing one of a hundred holes would actually move the privacy needle
  - "Bad guy" who will only abuse one obscure feature and ignore dozens of equivalents isn't a credible threat

Norms matter

- If we expect people to do this work, need to convince them that everyone else is doing it too
- And need to take on the “back catalog” to make it all meaningful

- Can we do this? Is it a productive way to spend our efforts? To what degree? (back to Lampson…)

Related work in IETF

- http://wiki.tools.ietf.org/html/draft-iab-privacy-considerations-04
- Good layout of model, terms, threats, etc.

“Fingerprinting. In many cases the specific ordering and/or occurrences of information elements in a protocol allow users, devices, or software using the protocol to be fingerprinted. Is this protocol vulnerable to fingerprinting? If so, how? Can be designed to reduce or eliminate the vulnerability? If not, why not?

Context Context Context

- Eliminating fingerprinting and other privacy leaks may be an achievable goal when designing a new protocol with confined boundaries.

- Is anything we do at W3C ever really new?
  - We are almost always operating in the context of the browser : a high-bandwidth, high-functionality application execution environment

Browser Fingerprinting ~= Covert Channel

- We have known since Lampson's *"A Note on the Confinement Problem"* (1973) that such channels cannot be eliminated.
- Best we can do is limit their bandwidth.
- We know it is very difficult and expensive to do even this even when it is part of the initial design criteria.

We are talking about retroactively removing covert channels from a 20+ year old system with >10e9 nodes and LoC

Private User Agent community group

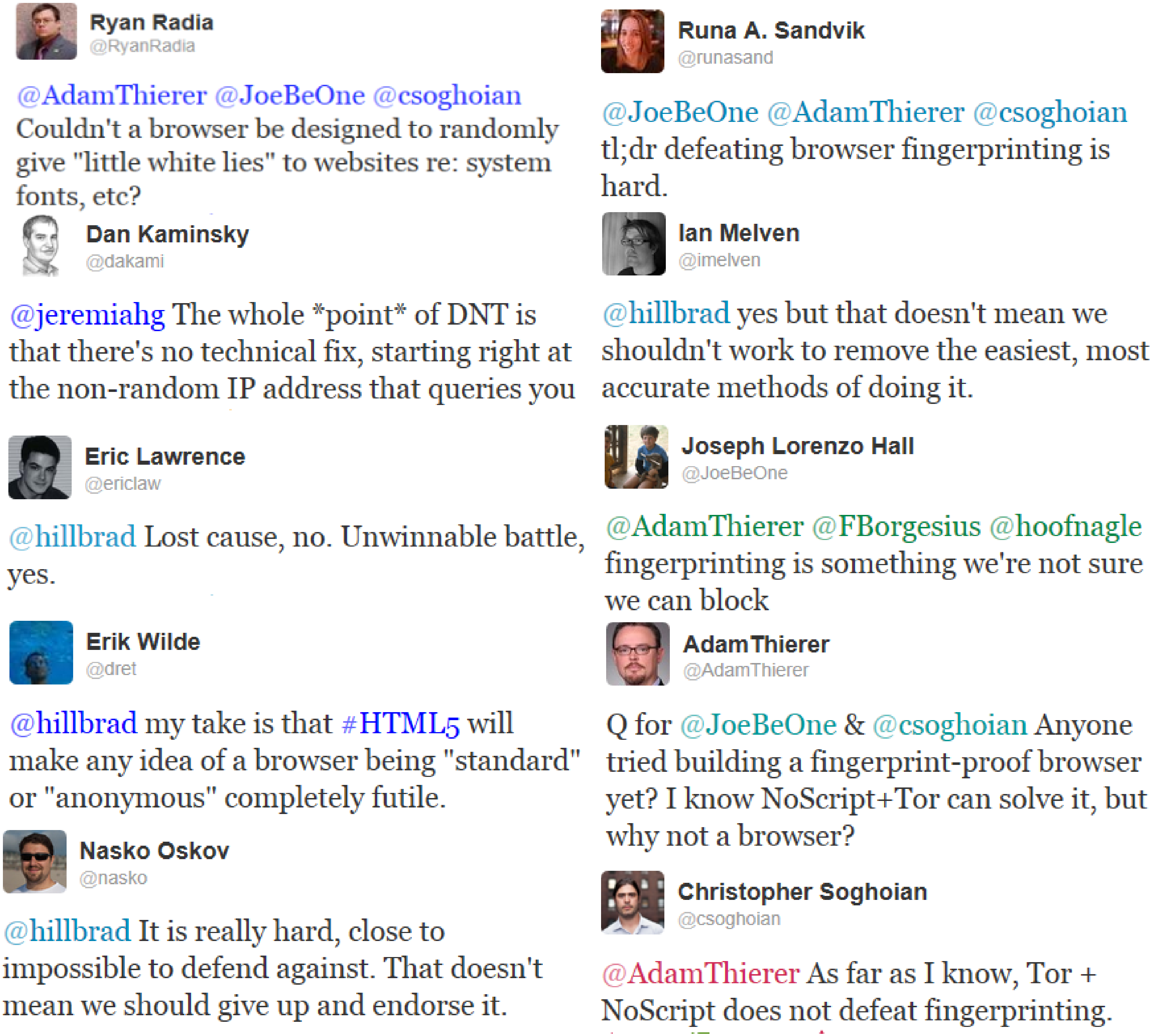
http://www.w3.org/community/pua/wiki/Draft

Javascript has access to a wide range of information about the UA and has access to communication channels to leak this information. Limiting access to back channels and limiting access to the UA state helps limit the covert sharing of the UA state.

One option is to continue to support JS but to add support for extra restricted JS contexts and two restricted contexts are considered below: 1. Shared Contexts that restrict access to defined private UA state but allow access to back channels; 2. Private Contexts that restrict access to back channels but allow access to the defined private UA state.

Alternate Approaches

- DNT: do not attempt to make tracking impossible, instead convey a user preference and manage/enforce at “layer 9”
- Incognito / In Private browsing modes
  - Adversary is other users of browser?
- Anonymity as a specific use case for specially-designed user agents: Tor Button
- Create a “standard” fingerprint.
  - Comment on Twitter: “saw a visitor from NSA, using hilariously out of date Firefox and XP”
    - Seems unlikely that’s true



Ryan Radia
@RyanRadia

@AdamThierer @JoeBeOne @csoghoian Couldn't a browser be designed to randomly give "little white lies" to websites re: system fonts, etc?

Dan Kaminsky
@dakami

@jeremiahg The whole *point* of DNT is that there's no technical fix, starting right at the non-random IP address that queries you

Eric Lawrence
@ericlaw

@hillbrad Lost cause, no. Unwinnable battle, yes.

Erik Wilde
@dret

@hillbrad my take is that #HTML5 will make any idea of a browser being "standard" or "anonymous" completely futile.

Nasko Oskov
@nasko

@hillbrad It is really hard, close to impossible to defend against. That doesn't mean we should give up and endorse it.

Runa A. Sandvik
@runasand

@JoeBeOne @AdamThierer @csoghoian tl;dr defeating browser fingerprinting is hard.

Ian Melven
@imelven

@hillbrad yes but that doesn't mean we shouldn't work to remove the easiest, most accurate methods of doing it.

Joseph Lorenzo Hall
@JoeBeOne

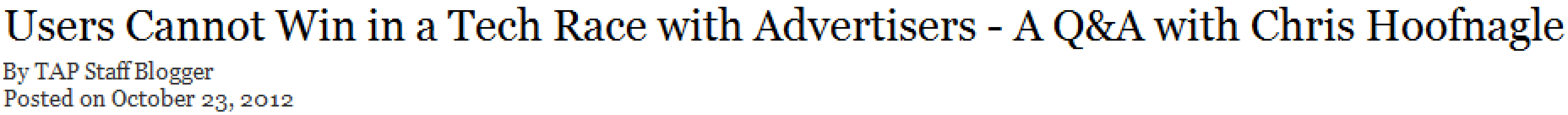
@AdamThierer @FBorgesius @hoofnagle fingerprinting is something we're not sure we can block

Adam Thierer
@AdamThierer

Q for @JoeBeOne & @csoghoian Anyone tried building a fingerprint-proof browser yet? I know NoScript+Tor can solve it, but why not a browser?

Christopher Soghoian
@csoghoian

@AdamThierer As far as I know, Tor + NoScript does not defeat fingerprinting.



Users Cannot Win in a Tech Race with Advertisers - A Q&A with Chris Hoofnagle

By TAP Staff Blogger
Posted on October 23, 2012

http://www.techpolicy.com/Blog/Featured-Blog-Post/Users-Cannot-Win-in-a-Tech-Race-with-Advertisers-A.aspx

TAP: You mentioned that “some companies have adjusted their tracking mechanisms to make it more difficult for users to avoid tracking.” Is there a way for consumers to block these additional mechanisms themselves?

Chris Hoofnagle: Users cannot win in a technology race with advertisers. The incentives are simply too powerful for companies to track users. The market for privacy-preserving tools is still in its infancy. Still, there are things consumers can do; the problem is that every privacy intervention can interfere with how websites and services function. On the most basic level, one can block third party cookies. This typically does not affect web browsing adversely, however, some authentication mechanisms fail when third party cookies are blocked. There are a number of privacy-preserving plugins that can help. Some of the best are Abine's DoNotTrackPlus, Ghostery and NoScript.

Solicited Email Correspondence

“I think that, all other things being equal, having W3C WGs consider the fingerprinting impacts of their work is worthwhile. Having W3C declare it irrelevant may impose a substantial, and hard to reverse, cost to privacy.”

Adam Shostack

“I think the objective of resistance to passive fingerprinting is both important and at potentially achievable.

Of the passively-measurable variables we know about, quite a few (excessive User Agent entropy, Accept headers, header ordering, header variation by request type) should be addressable through standards work.

Passively measurable variables that are harder include clock skew, clock error, and TCP stack variation. But my guess is that those are probably 5-10 bits of identity, and they could be reduced with some effort.

On the other hand active fingerprinting resistance (fonts, plugins, JS quirks,
etc) seems much harder, but less important, because people can crawl for and audit it.”
-Peter Eckersley

???