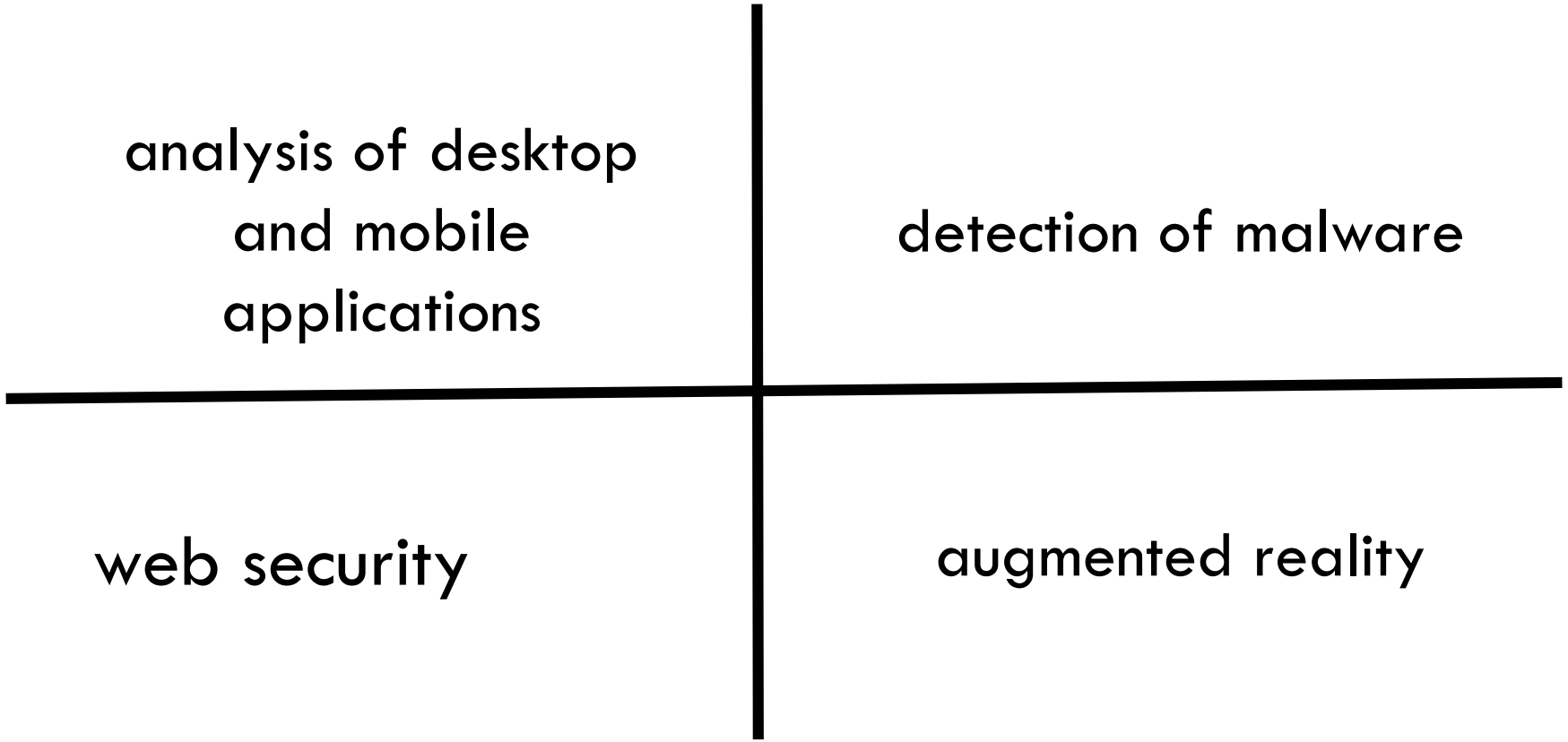
PriVaricator

PREVENTING STATELESS TRACKING ON THE WEB

Ben Livshits (MSR)
Nick Nikiforakis (SUNY Stony Brook)

INVOLVED IN A NUMBER OF TOPICS



PRIVARICATOR

PriVaricator: Deceiving Fingerprinters with Little White Lies

Nick Nikiforakis, Wouter Joosen
KU Leuven

Benjamin Livshits
Microsoft Research

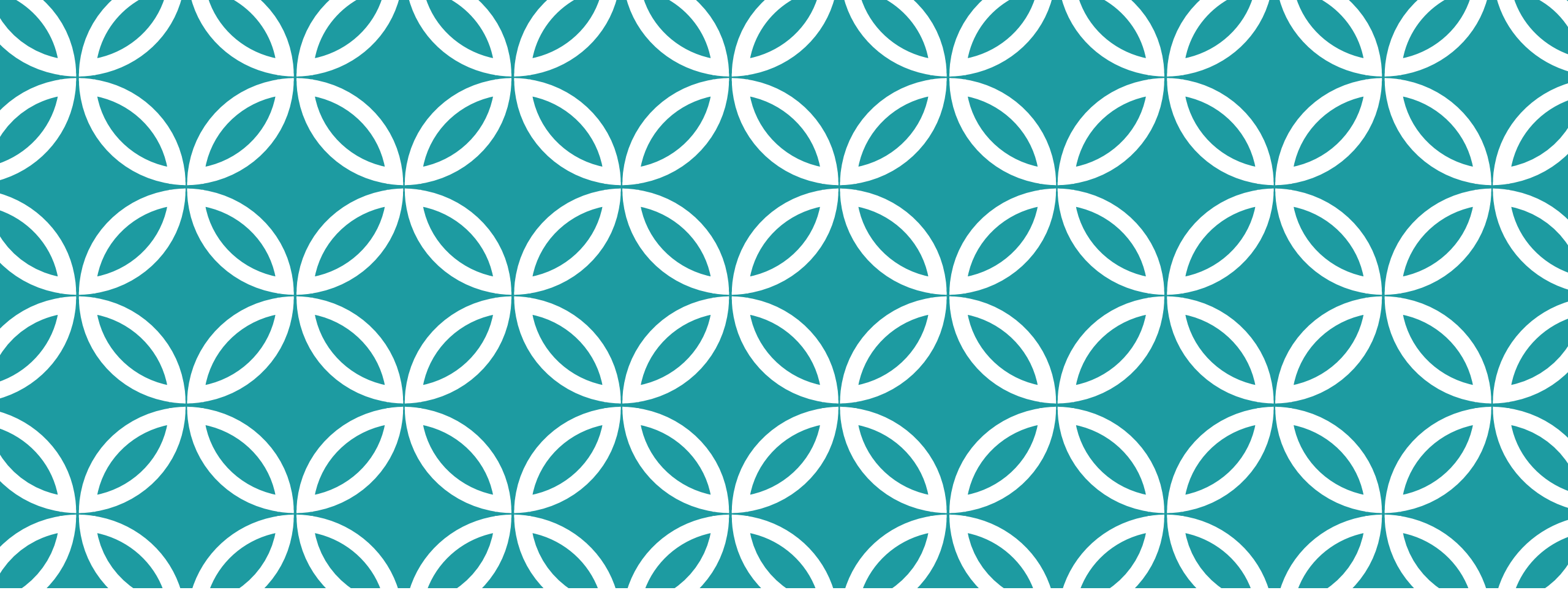
Abstract

This paper proposes a solution to the problem of browser-based fingerprinting. An important observation is that making fingerprints non-deterministic also makes them hard to link across subsequent web site visits. Our key insight is that when it comes to web tracking, the real problem with fingerprinting is not *uniqueness* of a fingerprint, it is *linkability*, i.e. the ability to connect the same fingerprint across multiple visits. In PriVaricator we use the power of randomization to "break" linkability by exploring a space of parameterized randomization policies. We evaluate our techniques in terms of being able to prevent fingerprinting and also in terms of not breaking existing (benign) sites. The best of our randomization policies renders *all* the fingerprinters we tested

Key insight: Much has been made of the fact that it is possible to derive a unique fingerprint of a user, primarily via JavaScript as shown by the Panopticlick project [8]. However, the insight behind our techniques is the realization that the culprit behind fingerprinting is not the fact that a user's fingerprint is unique, but that it is *linkable*, *i.e.* it can be reliably associated with the same user over multiple visits. While popular prevention techniques have attempted to make the fingerprints of large groups of users look *the same* [20], the key insight our paper explores involves doing the *opposite*. PriVaricator modifies the browser to make every visit appear different to a fingerprinting site, resulting in a *different* fingerprint that cannot be easily linked to a fingerprint from another visit, thus frustrating tracking attempts.

Upcoming paper in WWW'15

Read it for more details

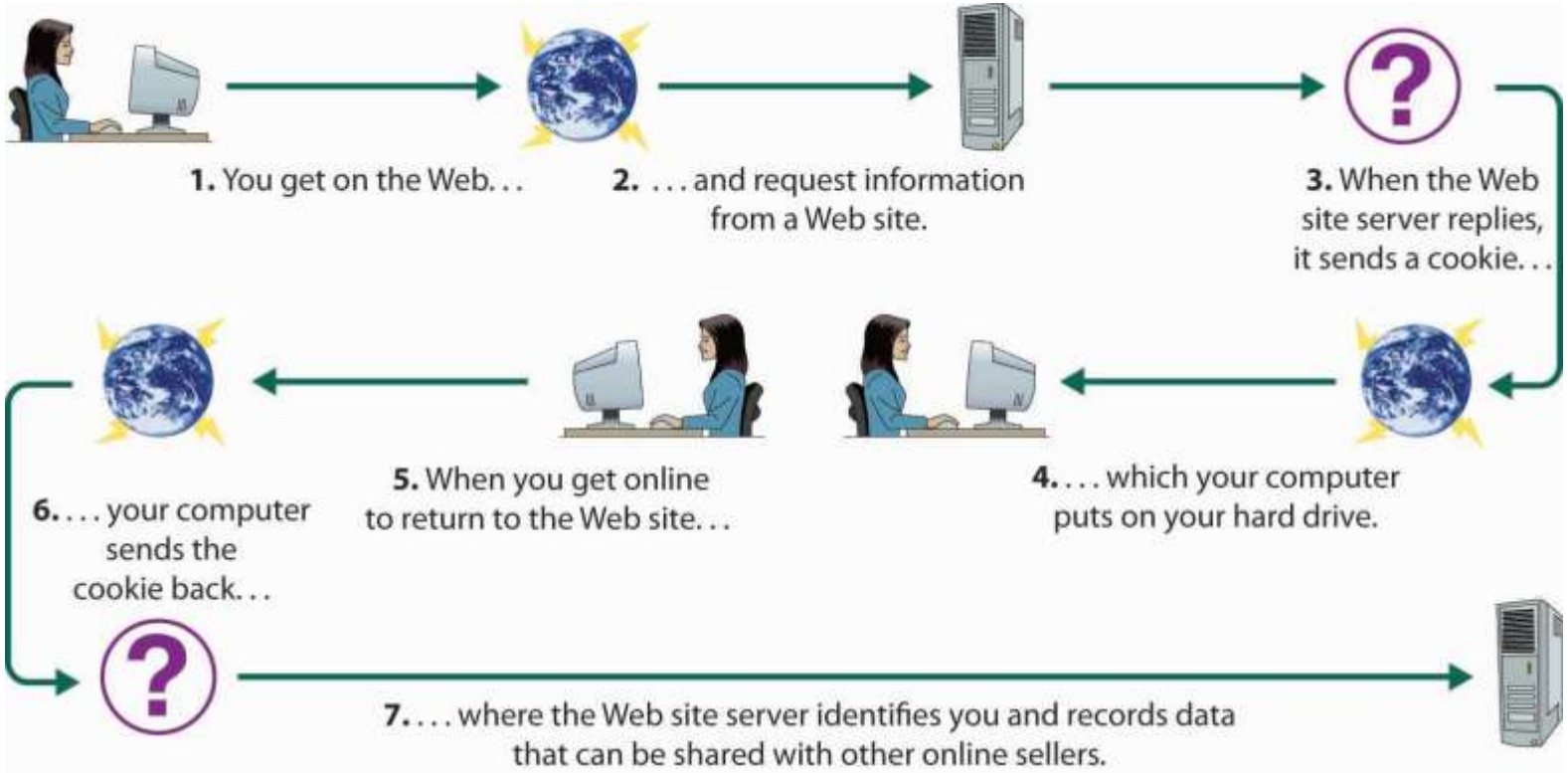


COOKIE-BASED TRACKING

COOKIES AND PRIVACY

A key topic in Web application privacy in the last several years

The majority of focus is on **cookie-based tracking**



LOTS AND LOTS OF ADVERTISING COMPANIES

a.com

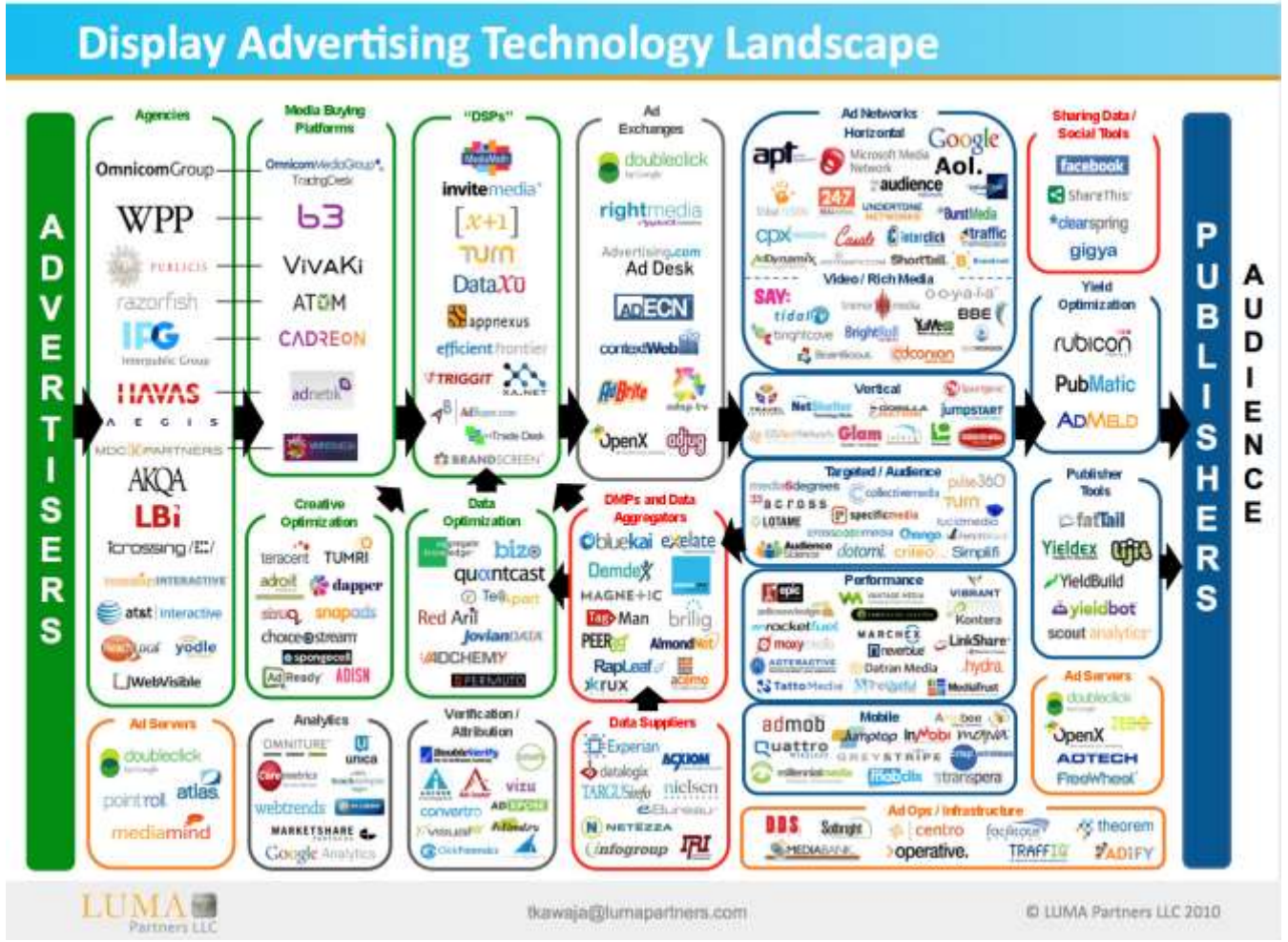
b.com

c.com

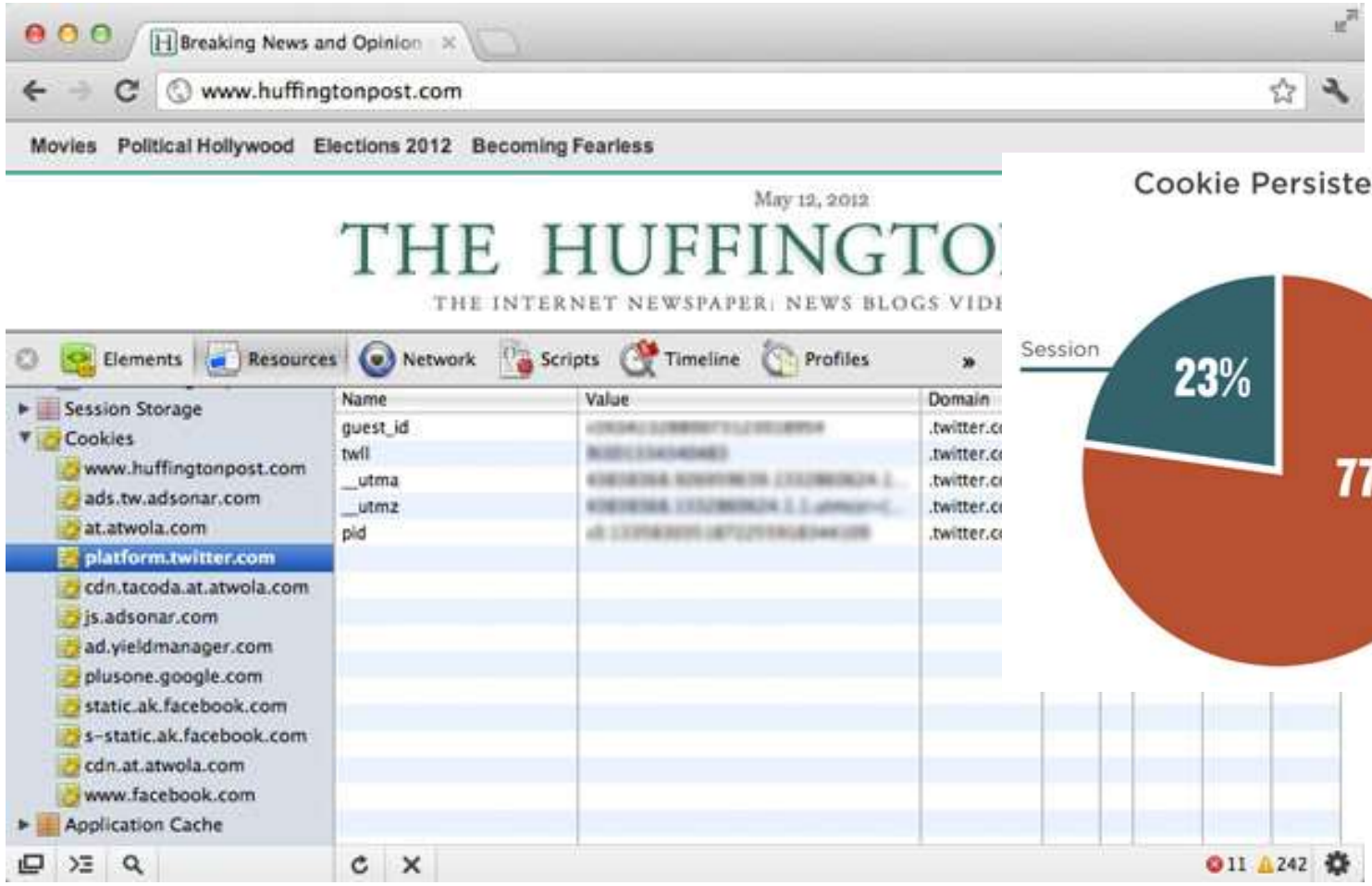
…

z.com

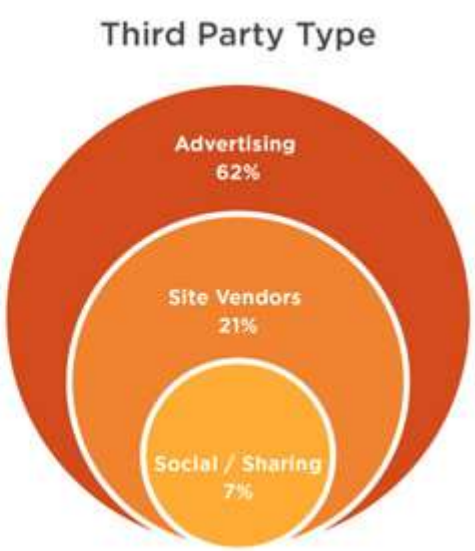
doubleclick by Google



COOKIES ON A POPULAR NEWS SITE



truste.com



TODAY, A VISIT TO HUFFINGTONPOST.COM RESULTS IN…

DATA GATHERED SINCE	YOU HAVE VISITED	YOU HAVE CONNECTED W
MAR 2, 2015	1 SITE	79 THIRD PART

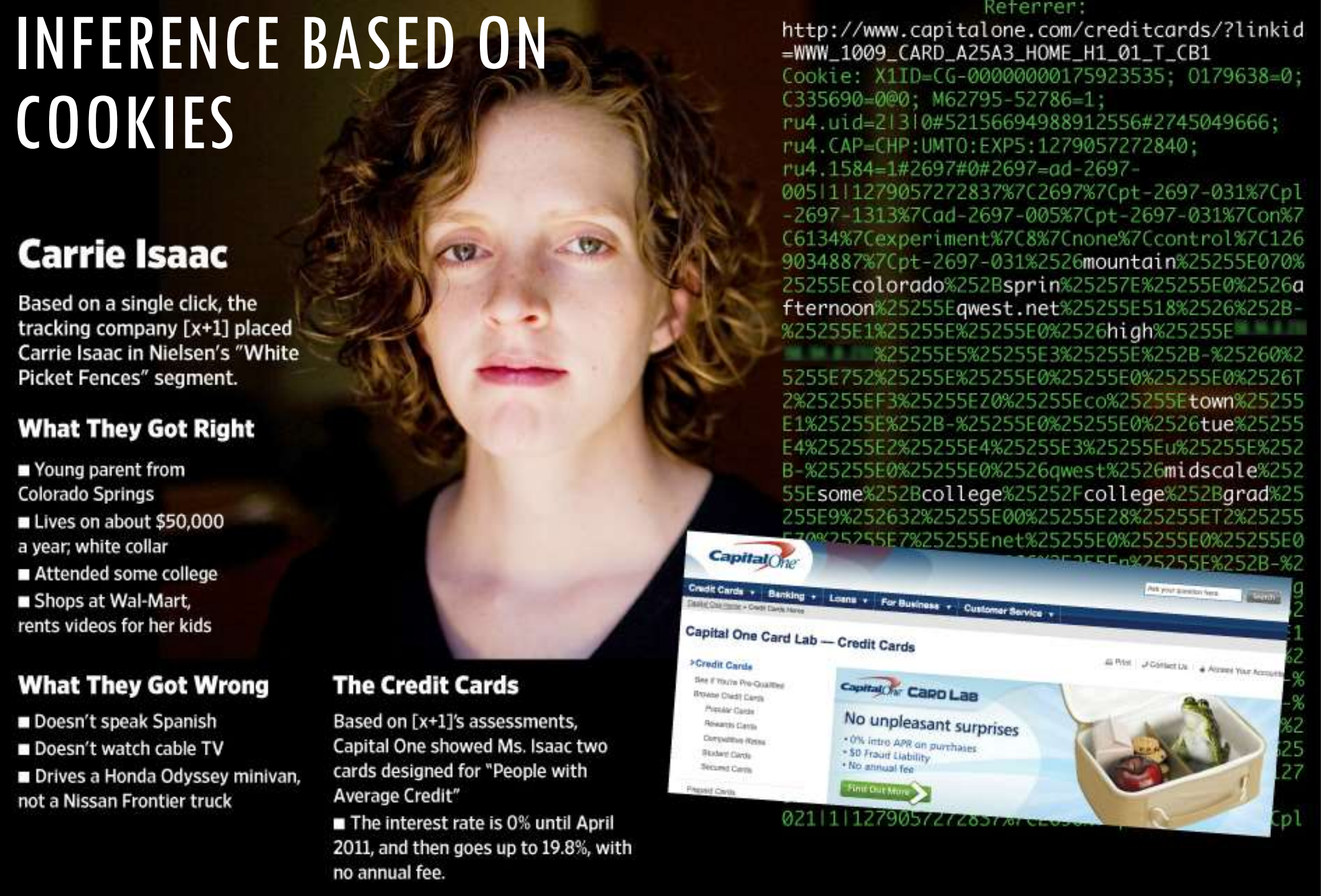
All Sites

Type	Prefs	Website
Third Party		turn.com
Visited		huffingtonpost.es
Third Party		tidaltv.com
Third Party		atwola.com
Third Party		amgdgt.com
Third Party		doubleclick.net
Third Party		gravity.com
Third Party		aolcdn.com
Third Party		google.com
Third Party		scorecardresearch.com
Third Party		facebook.com
Third Party		huffpost.com
Third Party		googlesyndication.com
Third Party		twitter.com
Third Party		gstatic.com
Third Party		weborama.fr
Third Party		grvcdn.com
Third Party		2mdn.net
Third Party		stickyadstv.com
Third Party		bluekai.com
Third Party		360yield.com
Third Party		mathtag.com
Third Party		audienceiq.com
Third Party		adnxs.com
Third Party		pubmatic.com
Third Party		contextweb.com
Third Party		smartclip.net
Third Party		liverail.com
Third Party		demdex.net
Third Party		huffingtonpost.com
Third Party		epimg.net
Third Party		quantserve.com
Third Party		googletagservices.com

A FUNDAMENTAL UNDERLYING QUESTION

Why profile the user?

INFERENCE BASED ON COOKIES



Carrie Isaac

Based on a single click, the tracking company [x+1] placed Carrie Isaac in Nielsen's "White Picket Fences" segment.

What They Got Right

- Young parent from Colorado Springs
- Lives on about $50,000 a year; white collar
- Attended some college
- Shops at Wal-Mart, rents videos for her kids

What They Got Wrong

- Doesn't speak Spanish
- Doesn't watch cable TV
- Drives a Honda Odyssey minivan, not a Nissan Frontier truck

The Credit Cards

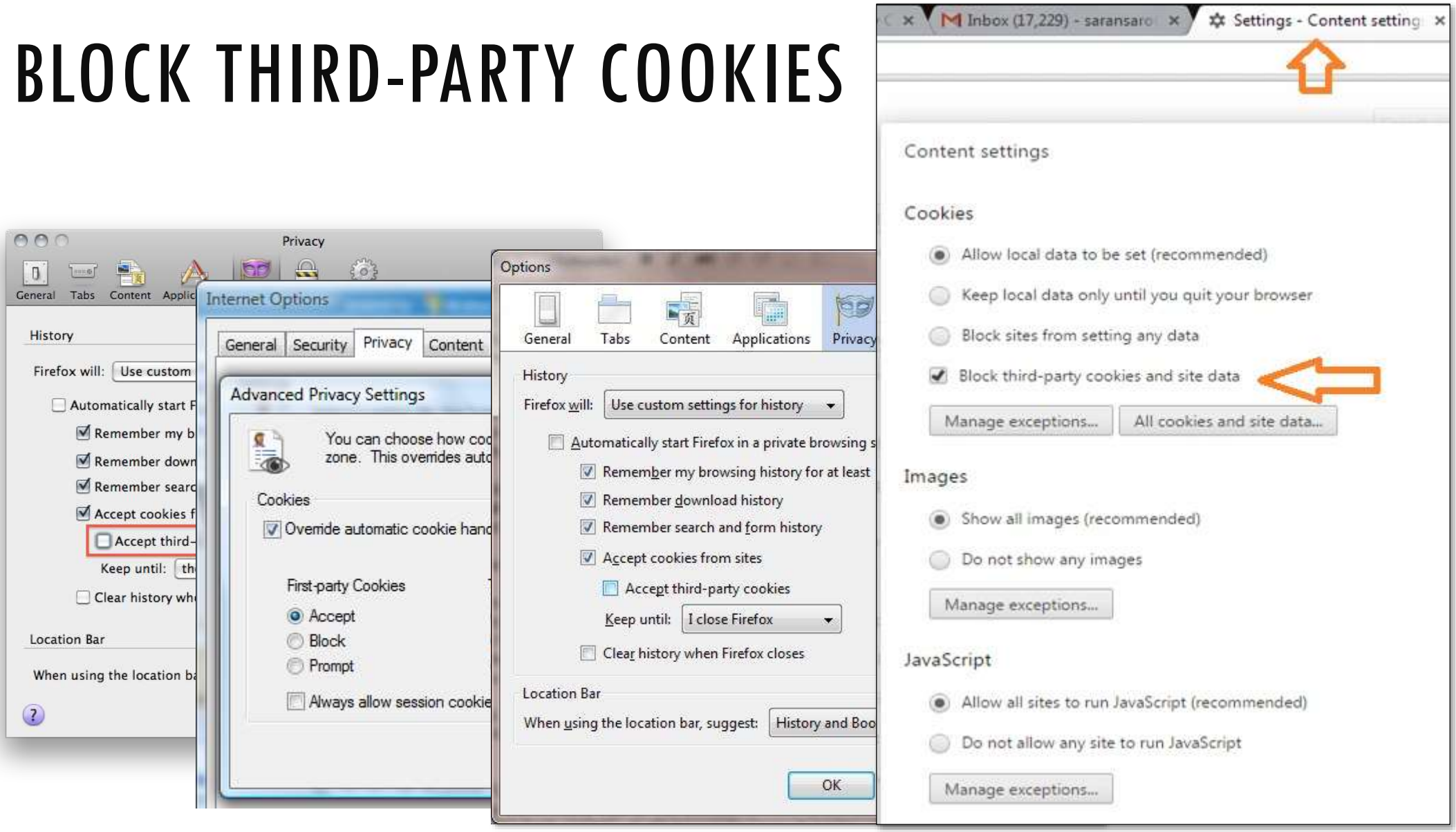
Based on [x+1]'s assessments, Capital One showed Ms. Isaac two cards designed for "People with Average Credit"

- The interest rate is 0% until April 2011, and then goes up to 19.8%, with no annual fee.

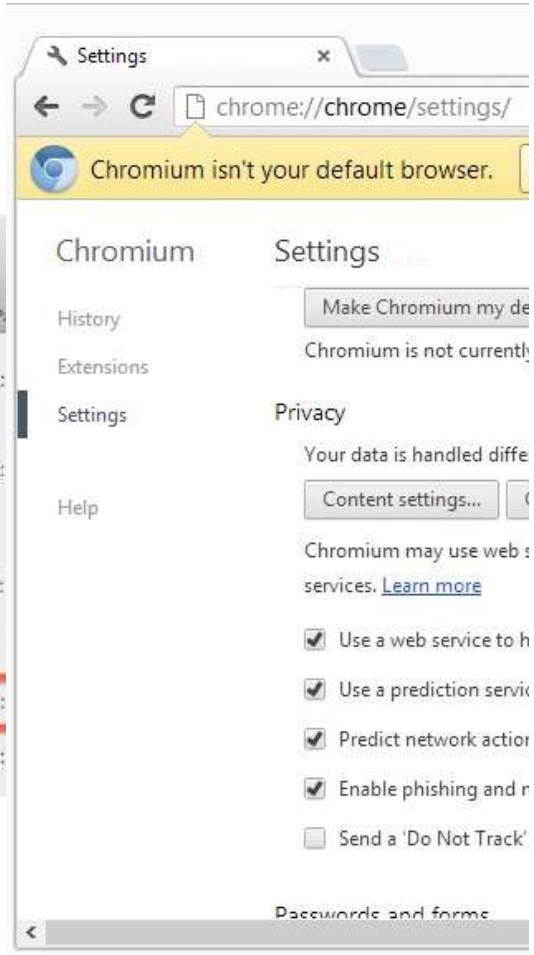
Ad tracking company [x+1] made predictions about users based on just one website click

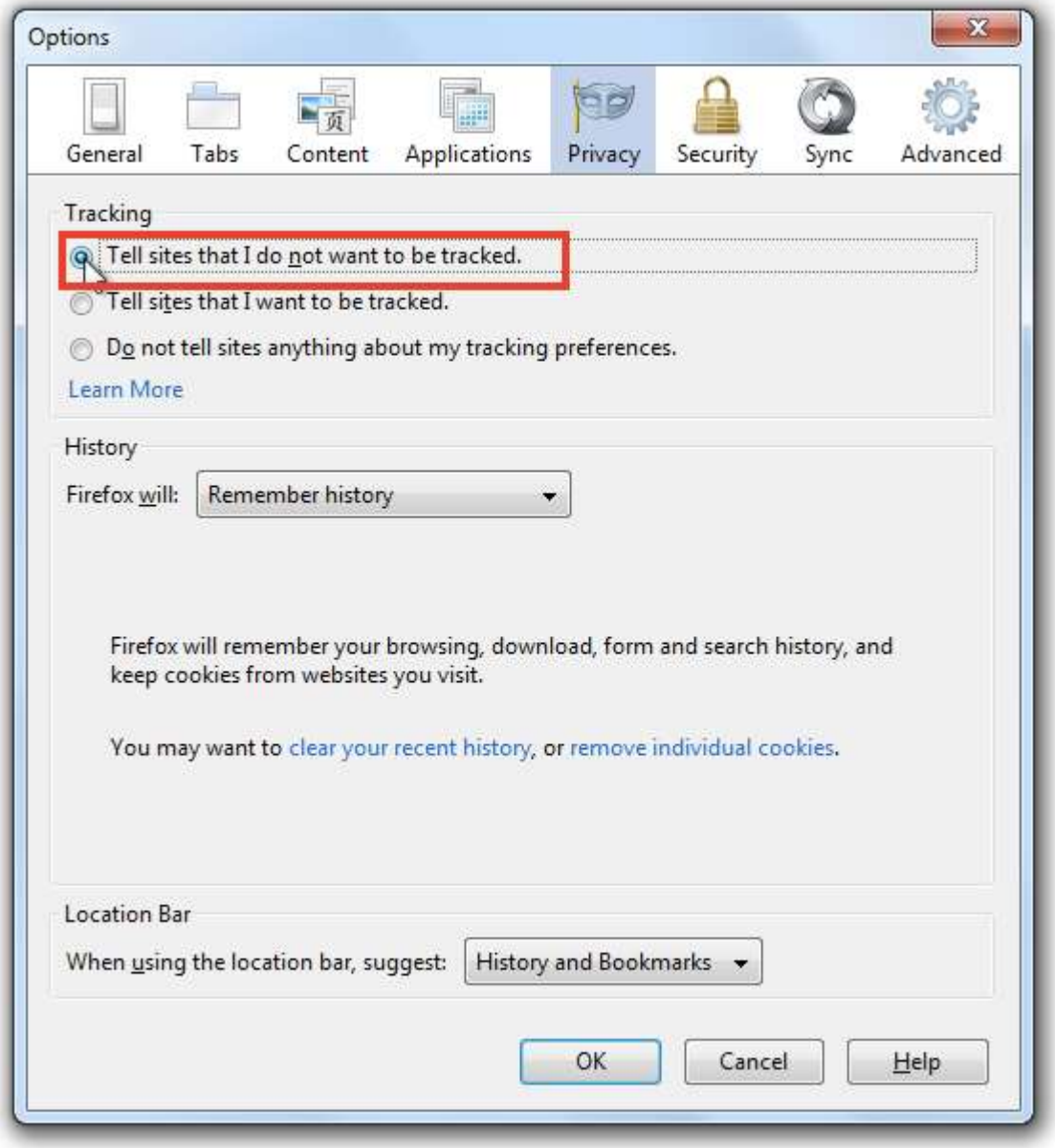
(from WSJ)

BLOCK THIRD-PARTY COOKIES

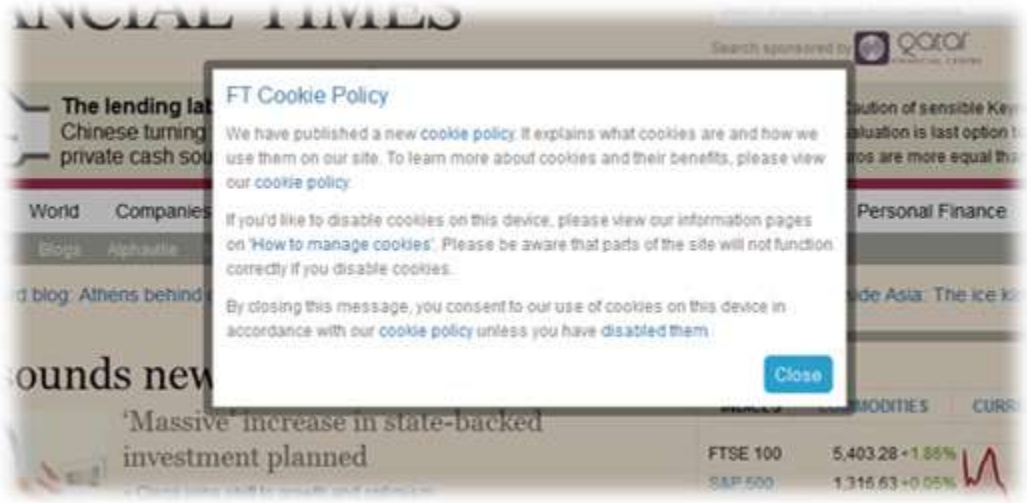


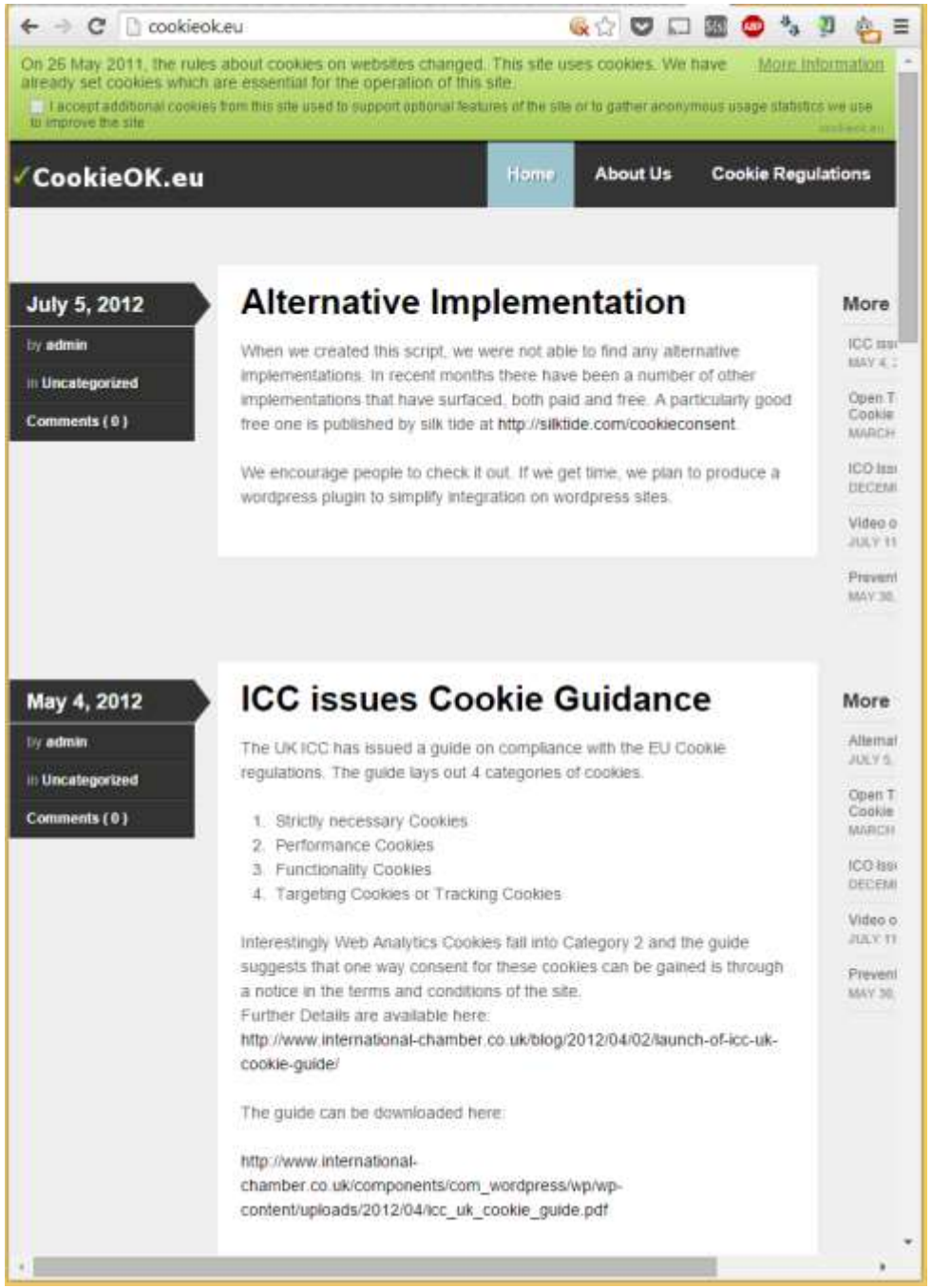
DO-NOT-TRACK INITIAT





EU COOKIE REGULATIONS





NOT EVERYBODY IS FOND OF THE COOKIE LAW

The Register®
Biting the hand that feeds IT

DATA CENTRE SOFTWARE NETWORKS SECURITY BUSINESS HARDWARE SCIENCE BOOTNOTES

Want to avoid another cookie law mess? Talk to EU bods next time

'Dear ICO, sue us ... We're sick of you and this ridiculous cookie law'

11 Sep 2012 at 08:19, OUT-LAW.COM 49 7 63

UK businesses should actively involve themselves in the debate over changes to EU law if they want to avoid problems stemming from the way those laws are drafted, an expert has advised.

Europe's Web-Cookie Warnings Are a Waste, Report Says

ARTICLE COMMENTS (2)

COOKIES EUROPEAN UNION REGULATION WEB BROWSERS

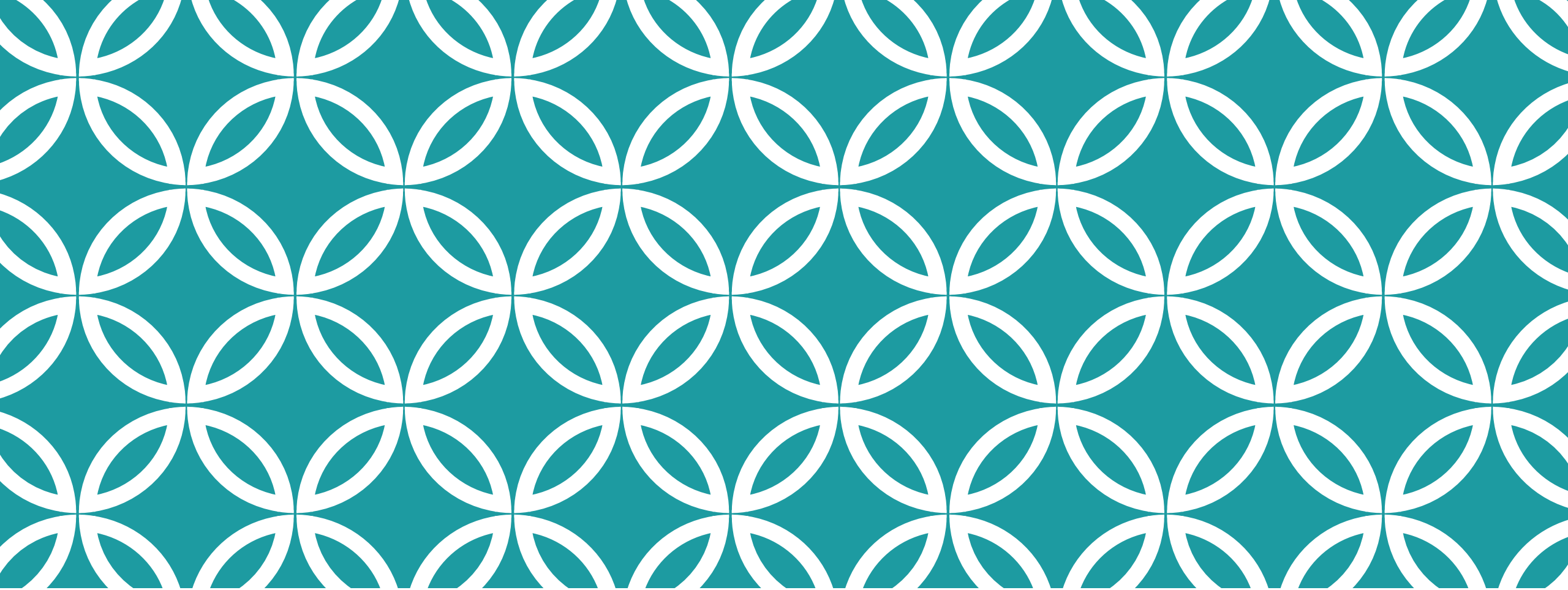
Email Print

By CHASE GUMMER

Jean-Claude Juncker, new president of the European Commission, has written to a colleague to urge a review of the cookie policy. — Agence France-Presse/Getty Images

Internet cookie notifications are costing European taxpayers a mint while offering netizens no real benefit, a new report by a Washington-based think tank says.

Web surfers in many European countries are greeted with banners and pop-up



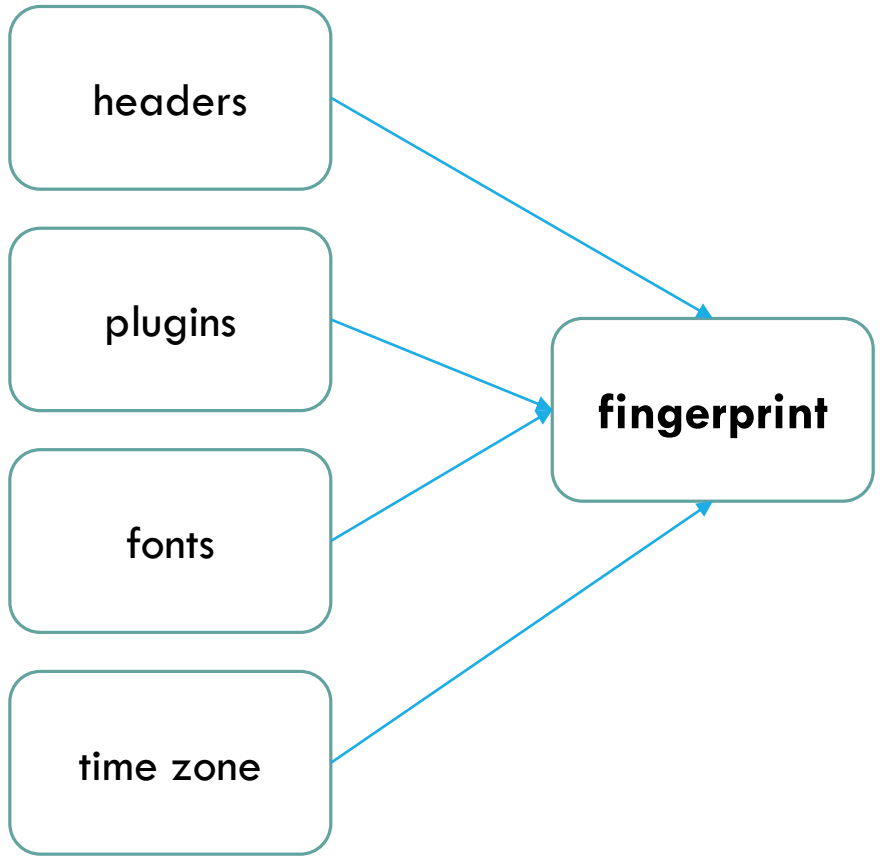
STATELESS TRACKING

STATELESS FINGERPRINTING

Emerges around 2010 as a project from the EFF

Since then, has been replicated in various settings, including by academic researchers

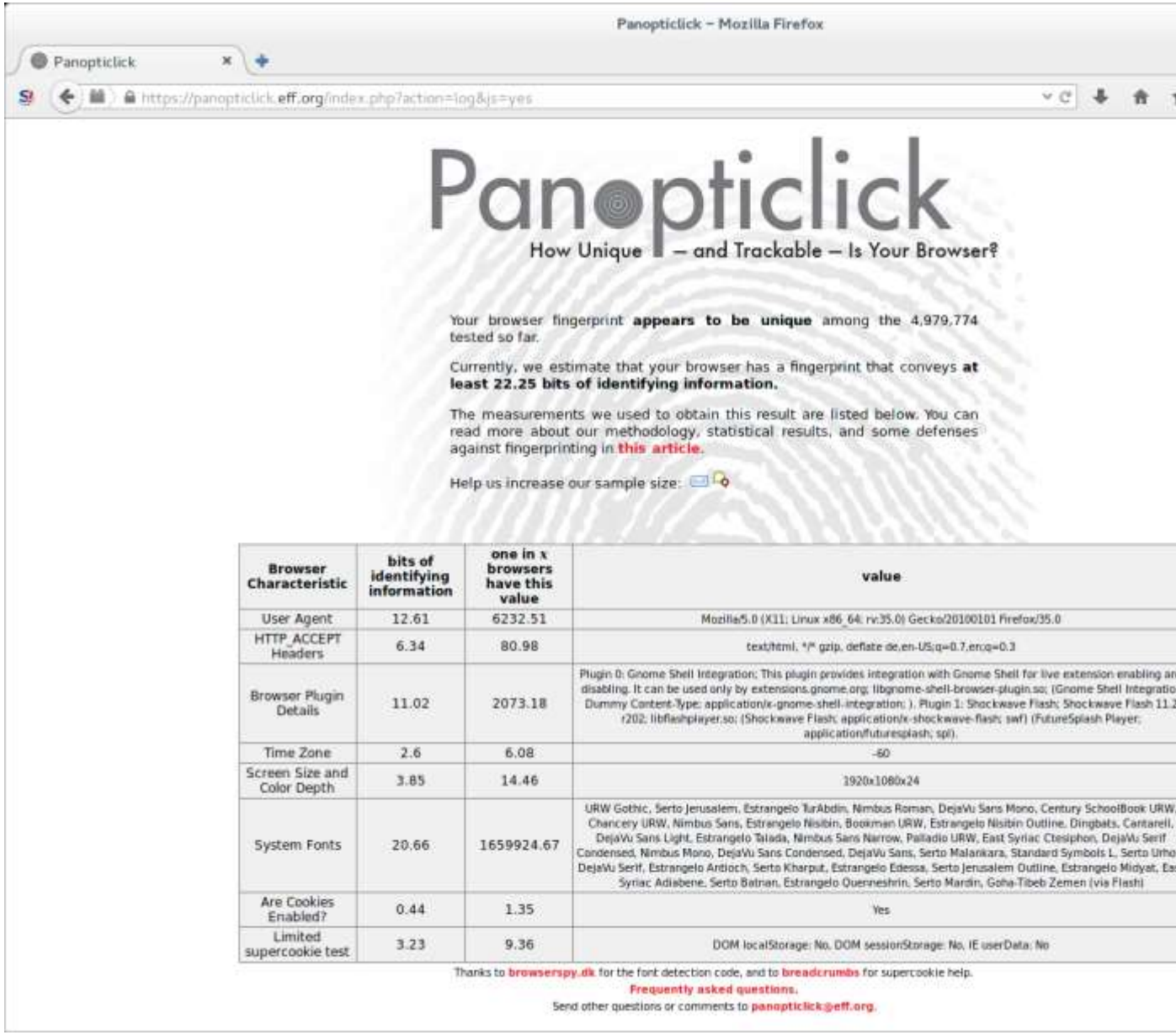
In the last two years we have seen active fingerprinting from several large advertising targeting companies: BlueCava, Iovation, and ThreatMetrix



PANOPTICLICK

Of the 470,000-plus users who had participated at that point in his public Panopticlick project, **84 percent** of their browsers produced unique fingerprints

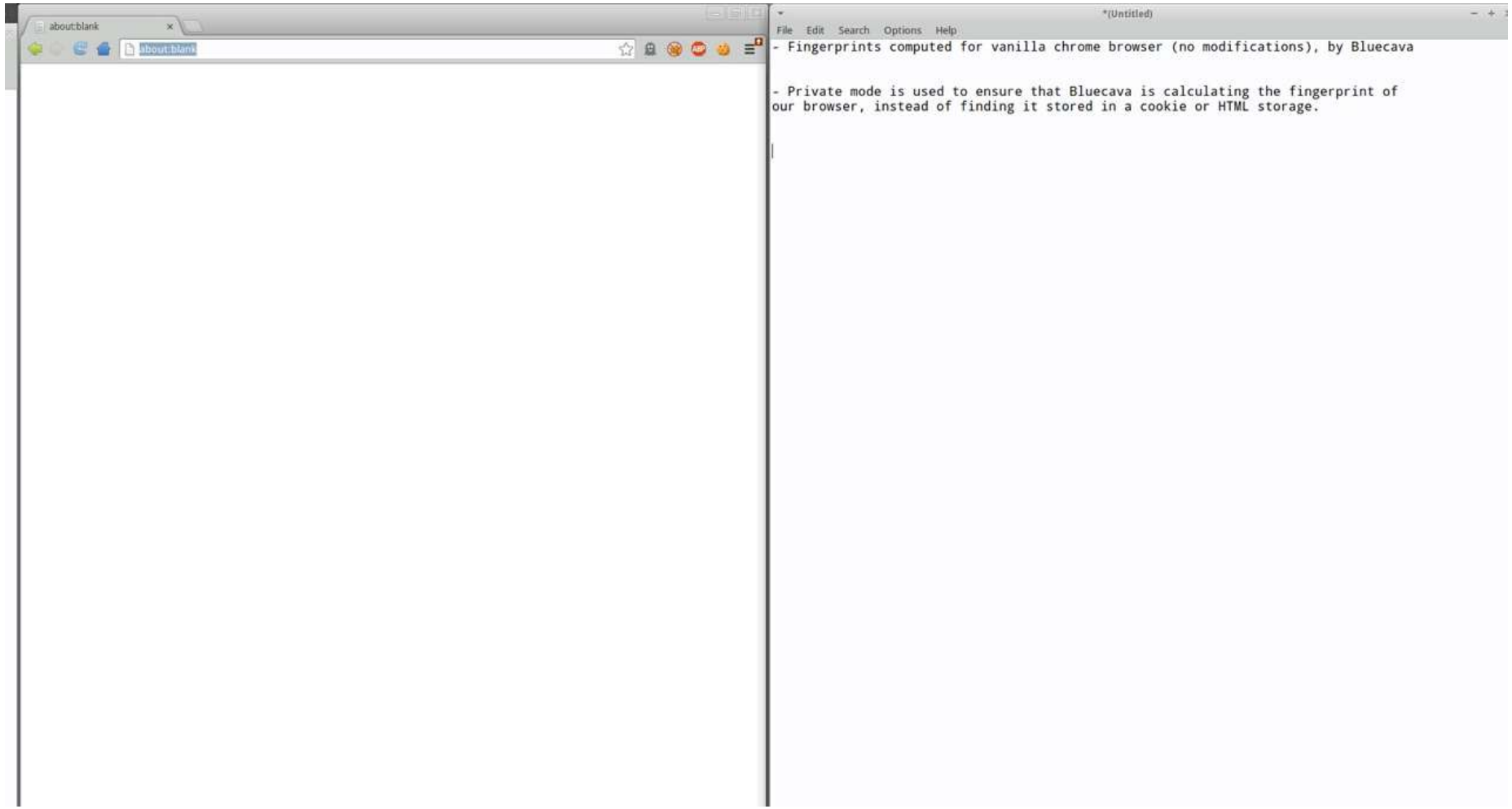
94 percent if you count those that supported Flash or Java)



FINGERPRINT.JS: FINGERPRINTING LIB ON GITHUB

```
Fingerprint.prototype = {
  get: function(){
    var keys = [];
    keys.push(navigator.userAgent);
    keys.push(navigator.language);
    keys.push(screen.colorDepth);
    if (this.screen_resolution) {
      var resolution = this.getScreenResolution();
      if (typeof resolution !== 'undefined'){ // headless brows
        keys.push(this.getScreenResolution().join('x'));
      }
    }
    keys.push(new Date().getTimezoneOffset());
    keys.push(this.hasSessionStorage());
    keys.push(this.hasLocalStorage());
    keys.push(!!window.indexedDB);
    //body might not be defined at this point or removed progra
    if(document.body){
      keys.push(typeof(document.body.addBehavior));
    } else {
      keys.push(typeof undefined);
    }
    keys.push(typeof(window.openDatabase));
    keys.push(navigator.cpuClass);
    keys.push(navigator.platform);
    keys.push(navigator.doNotTrack);
    keys.push(this.getPluginsString());
    if(this.canvas && this.isCanvasSupported()){
      keys.push(this.getCanvasFingerprint());
    }
    if(this.hasher){
      return this.hasher(keys.join('###'), 31);
    } else {
      return this.murmurhash3_32_gc(keys.join('###'), 31);
    }
  },
```

BLUE CAVA FINGERPRINTING IN ACTION



CURRENT STATE OF FINGERPRINTING?

Results in *Cookieless monster* showed that 159 of Alexa's 10,000 most-visited websites track their users with such fingerprinting software.

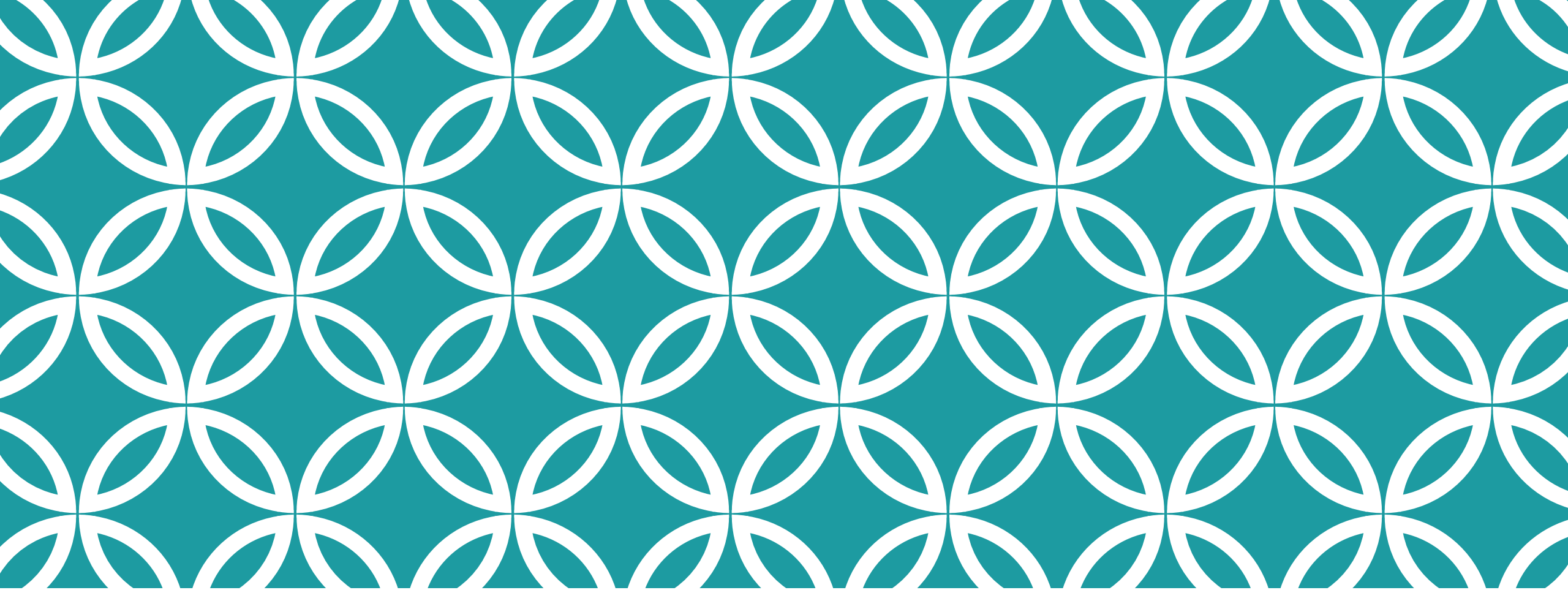
Also found that more than 400 of the million most popular websites on the Internet have been using JavaScript-only fingerprinting, which works on Flash-less devices such as the iPhone or iPad.

Users continue to be fingerprinted even if they have checked "Do Not Track" in their browser's preferences

But it's a little hard to say how much is really going on in practice

Fingerprinting is designed to remain pretty invisible

At the same time, we should expect more in this space because of cookie-based tracking becoming problematic



PRIVARICATOR

INSIGHT OF PRIVARICATOR

Most prior research focuses on making fingerprints not unique

For example, they make navigator.userAgent to always be Firefox

They strip revealing headers, etc.

Typically this is done via browser extenions

What is the effect of that?

The focus on user **uniqueness** is misguided

What matters is fingerprint **linkability**

Making fingerprints non-deterministic also makes them hard to link across browsing sessions

It's often easier to **randomize** the fingerprint than to keep in the same

USE "PLUGGABLE" RANDOMIZATION POLICIES

We explore a space of **randomization policies** designed to produce unique fingerprints

Change the way the browser represents certain important properties (**offsetHeight** used to measure the presence of fonts) and **plugins**, to the JavaScript environment

Creatively misrepresenting — or lying — about these values introduces an element of non-determinism, which generally makes fingerprints **unlinkable** over visits

Producing practically **impossible** combinations of, say, browser headers and the navigator object, can actually **reduce** user privacy

Blatant lying is *not* such a good idea

Can significantly degrade user experience by, for instance, by presenting Firefox-optimized sites to users of IE, leading to visual discrepancies or calls into missing APIs

A GOOD RANDOMIZATION POLICY SHOULD…

1) produce unlinkable fingerprints; and
2) not break existing sites

EXTENSION TO THE PRIVACY MODE

Browsers today already come with a private mode

Designed to combat stateful (cookie-based) fingerprinting

PriVaricator adds protection against stateless fingerprinting

Built on top of Chromium and can be integrated directly into the browser

Deploying it as an extension is not a such a good idea because it may make the user **more** identifiable, not less

WHAT TO MISREPRESENT?

Need to balance fingerprinting prevention with breaking existing sites

For example, navigator.userAgent is a *bad* thing to misrepresent

Likely to lead to a lot of site breakage

plugins

fonts

Fingerprinting provider	Script name	Plug n enumer tion	Screen properties	Uses canvas	Acces to offsetWidth	offsetHeight
BlueCava	BCAC5.js	✓	✓	✗	63	63
Perferencement	tagv22.pkmin.js	✓	✓	✗	155	155
CoinBase	application-9a3a[...].js	✓	✓	✗	592	197
MaxMind	device.js	✓	✓	✗	261	27
Inside graphs	ig.js	✓	✓	✗	1,050	48

SPACE OF RANDOMIZATION POLICIES

Policies for offset measurements

For the values of **offsetHeight, offsetWidth,** and **getBoundingClientRect** in PriVaricator, we propose the following numeric randomization policies

a) Zero

b) Random(1..100)

+/- 5% noise

The policies are parametrized by the *lying threshold* (denoted as θ) and a *lying probability* (denoted as *P*(*lie*)).

θ controls how fast PriVaricator starts lying, *i.e.*, after how many accesses to **offsetWidth** or **offsetHeight** values, will the policy kick in

Policies for plugins

P(plug_hide) the probability of hiding each individual plugin in navigator.plugins

SAMPLE RANDOMIZATION POLICY

$$\begin{aligned} \text{Rand\_Policy} &= \text{Zero}, \\ \theta &= 50, \\ P(lie) &= 20\%, \\ P(plug\_hide) &= 30\% \end{aligned}$$

respond with the value 0 when lying

start lying after 50 offset accesses

only lie in 20% of the cases

hide 30% of the browser's plugins

BREAKAGE CONCERNS

Out of Alexa 10,000
1.87% of scripts have 50+
accesses when visited

but don't want to break
spiegel.de

Alexa Rank	Domain	*Screen* colorDepth	*Screen* height	*Screen* pixelDepth	*Screen* width	*Navigator* mimeType	*Navigator* platform	*Navigator* language	*Navigator* userAgent	*Navigator* appName	*Navigator* vendor	*Navigator* appCodeName	*Navigator* appVersion	*Navigator* plugins	*HTML element* getBounding-ClientRect	*HTML element* offsetWidth	*HTML element* offsetHeight
6,444	bunte.de	0	1	0	1	0	0	2	8	0	0	0	0	1	2	205,115	202,909
8,039	nzz.ch	3	34	1	34	4	4	4	176	5	0	0	5	4	48	187,881	187,349
191	spiegel.de	2	4	0	4	0	0	0	15	3	0	0	1	4	7	154,265	149,293
4,037	wistia.com	1	2	0	2	0	0	3	81	0	0	0	0	0	0	109,347	109,299
1,369	zeit.de	0	4	1	4	0	0	2	8	3	2	0	0	5	1,318	70,025	72,268
8,894	menards.com	0	0	0	0	0	0	0	3,783	0	0	0	0	0	37	43,8[illegible]7	38,715
4,754	groupon.fr	0	0	0	0	0	0	0	1	0	0	0	0	0	15	150,[illegible]7	36,627
7,488	xinmin.cn	0	0	0	1	0	0	0	70,380	0	0	0	3	0	4,426	34[illegible]	31,996
2,320	celebuzz.com	2	55	0	55	4	2	0	23	0	0	0	0	4	326	27[illegible]	27,779
1,370	wetter.com	4	30	0	30	6	1	8	18	5	0	0	1	7	212	2[illegible]	21,764

most are ranked pretty low

82.3% of scripts have 0
accesses to **offsetHeight**

POLICY IMPLEMENTATION IN THE BROWSER

Strawman approach

Instrumented access to navigator.plugins at the source level

Try to intercept calls to **offsetWidth** and **offsetHeight** using DOM getters

However, it's difficult to know which element will be measured

offsetWidth and **offsetHeight** properties are not part of the HTMLElement prototype

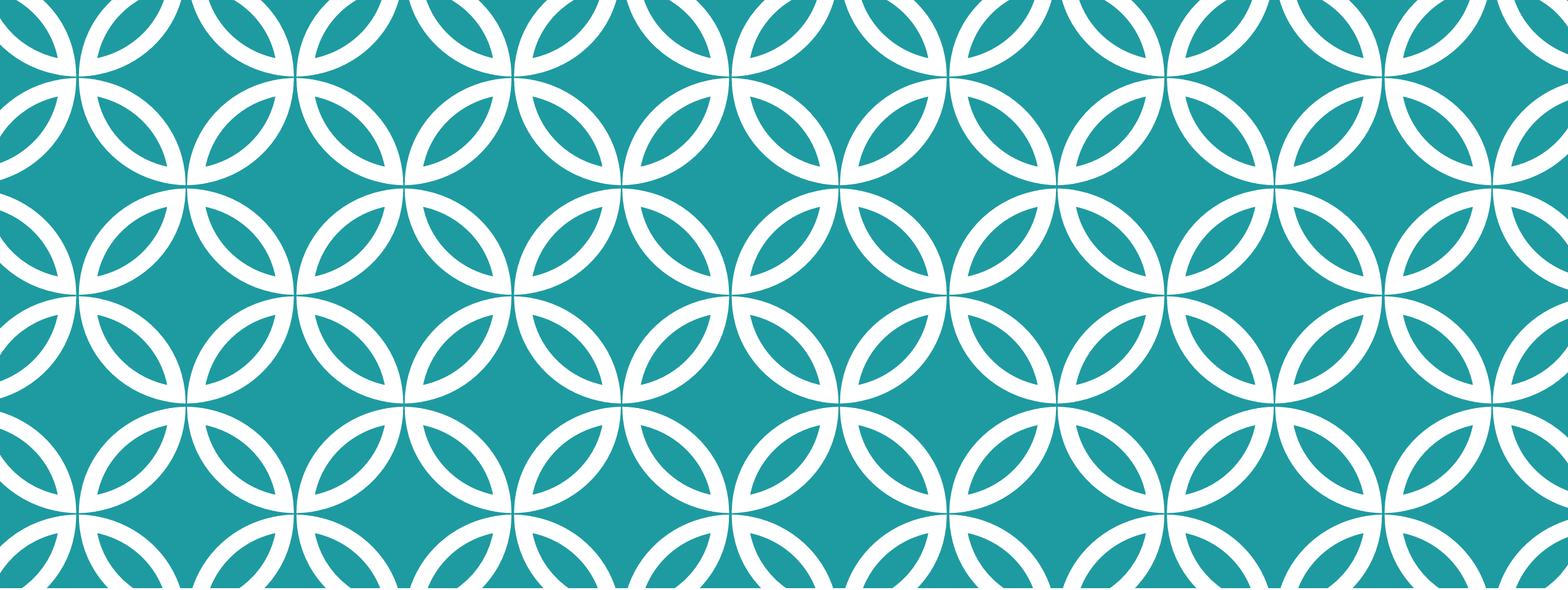
Real implementation

Instrument access to the properties of interest

Browser changes are, by nature, very local

Our full prototype involves modifications to a total of seven files in the WebKit implementation of the Chromium browser, version 34.0.1768.0 (242762)

947 lines of code added/changed



EVALUATION

EVALUATION: DIMENSIONS

Performance impact

Effectiveness in breaking existing fingerprinters

Minimizing breakage

SLOWDOWN? IN THE NOISE

Browser	JSBench		SunSpider		Kraken	
Chromium	72.31	±0.40	139.20	±1.00	1,146	±20.48
PriVaricator	72.10	±0.31	138.70	±0.49	1,142	±20.09

Executed each suite five times, clearing the browser's cache in between runs

The experiments were run on a desktop machine, running a recent Ubuntu Linux distribution, with an Intel Core i5-3570 CPU @ 3.40 GHz processor, and 8 GB of RAM

To calculate the upper bound of PriVaricator's overhead, we used the lying policy with the most computations (± 5% Noise) configured with the worst (from a performance point of view) parameter settings, i.e., , θ**=0** and *P*(*lie*)=100%

IS IT EFFECTIVE?

1) BlueCava
- http://bluecava.com/opt-out
- Shows fingerprints such as 18B1-EBFC-A3F0-6D81-6DE8-D8DA-CA56-A22B

2) PetPortal
- http://fingerprint.pet-portal.eu
- Similarly, get a fingerprint

3) Coinbase
- Obtained entirely client-side
- Can be captured
- MD5 applied to it and it's submitted via a cookie

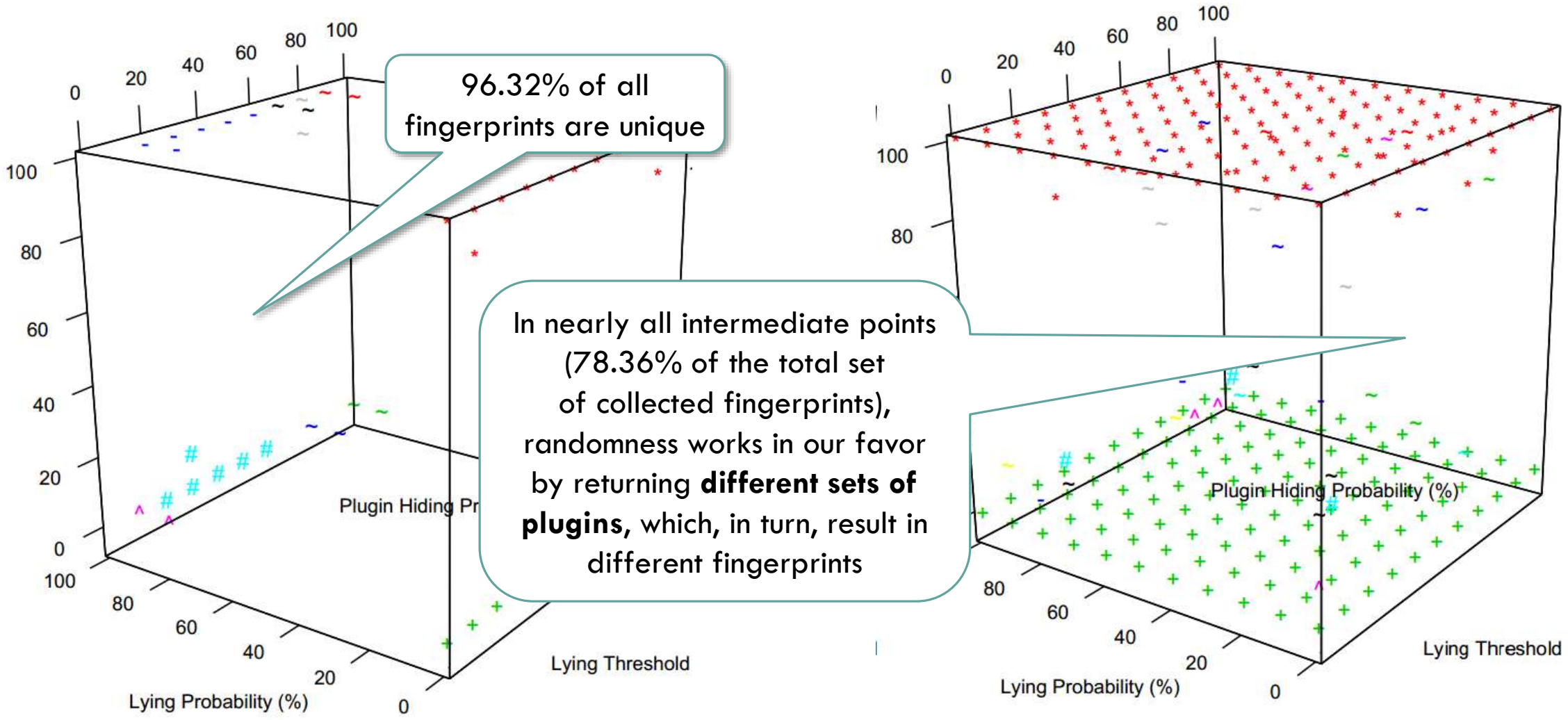
4) fingerprintjs
- That's the code we saw earlier

To explore the space of possible policies in detail, we performed an automated experiment where we visited each fingerprinting provider 1,331 times, to account for 11^3 parameter combinations, where each parameter of our randomized policy
- lying threshold
- lying probability, and
- plugin-hiding probability

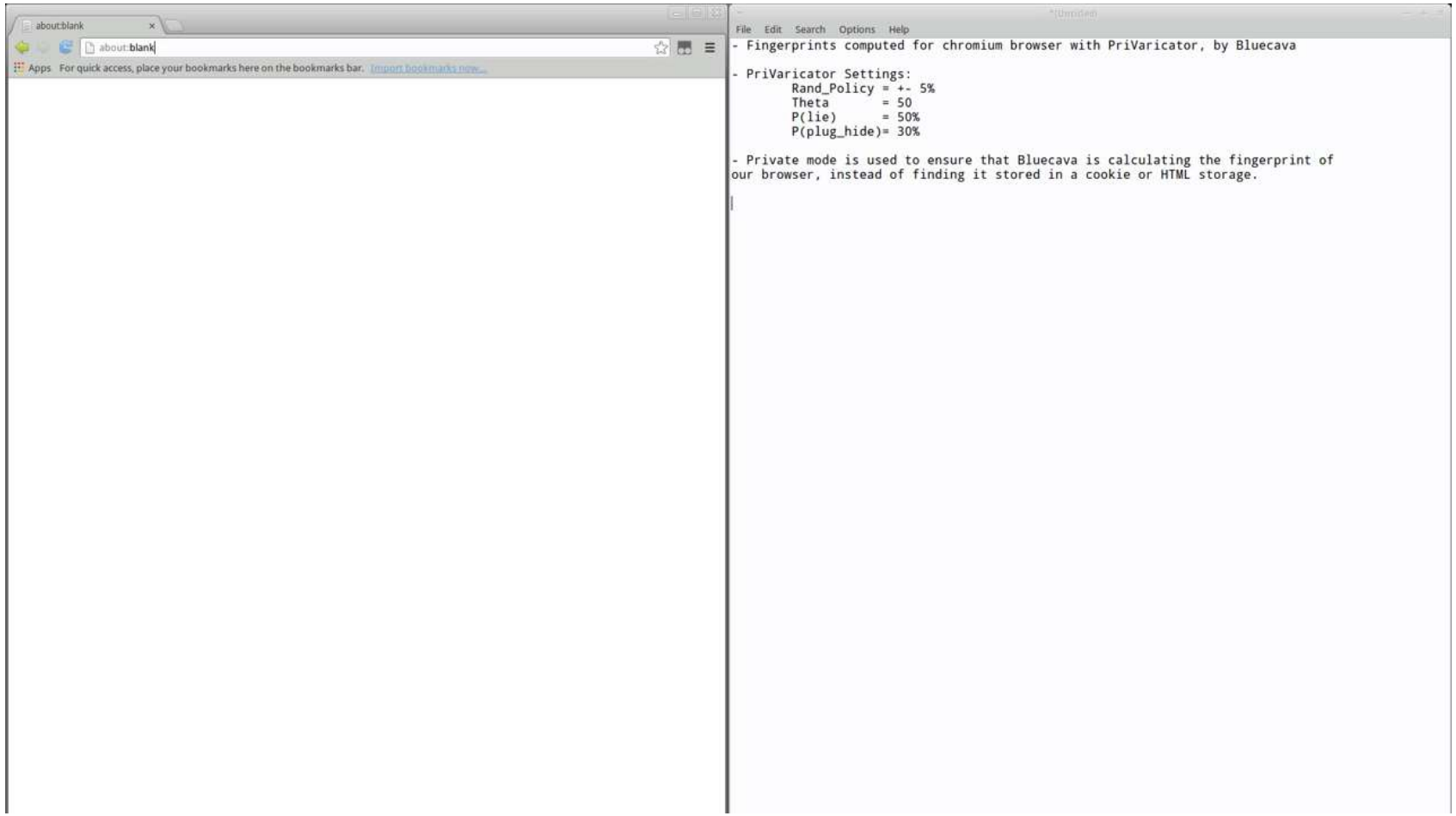
ranged from 0 to 100 in increments of 10

SUCCESS OF PRIVARICATOR

fingerprint.js



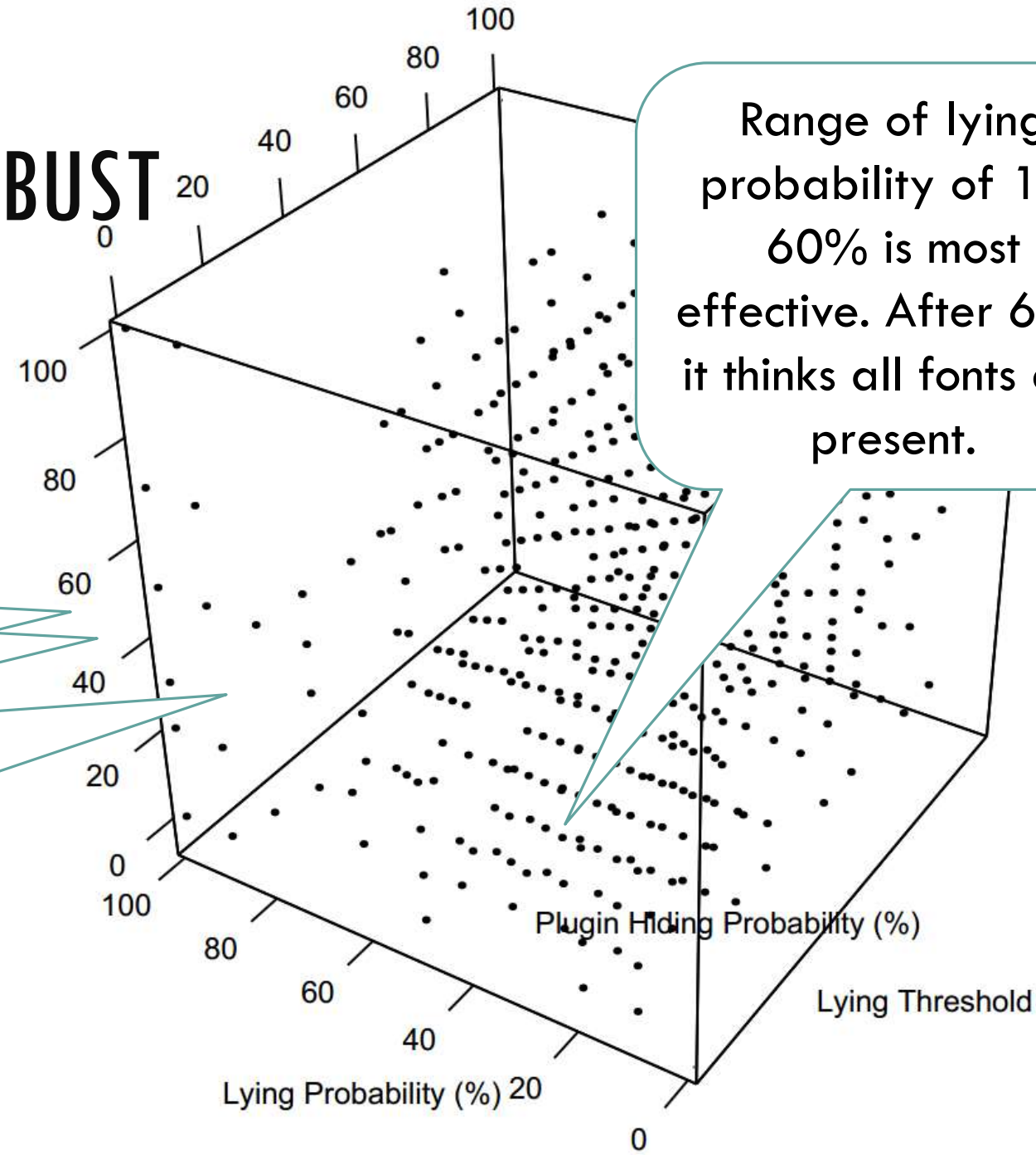
PRIVARICATOR STOPPING BLUE CAVA FINGERPRINTING



PETPORTAL: MOST ROBUST

Get unique fingerprints for "only" 37.83% of the 1,331 parameter combinations

eds more in tracking us than BlueCava, Coinbase, and fingerprintjs



Range of lying probability of 10-60% is most effective. After 60% it thinks all fonts are present.

MEASURING BREAKAGE

When PriVaricator lies about these values like **offsetWidth** and **offsetHeight**, it creates a potential for **visual breakage**

For example, by reporting that an element is smaller than it actually is, PriVaricator could cause the page to place it in a smaller container, hiding part of its content from the user.

Numerically, we define *breakage* as the fraction of pixels that are different when a site is loaded with a vanilla browser (PriVaricator turned off) and with PriVaricator

We instrumented Chromium to visit the main pages of the top 1,000 Alexa sites, for 48 different combinations of lying probability and lying threshold; these were the parameter combinations that resulted in unique fingerprints for PetPortal

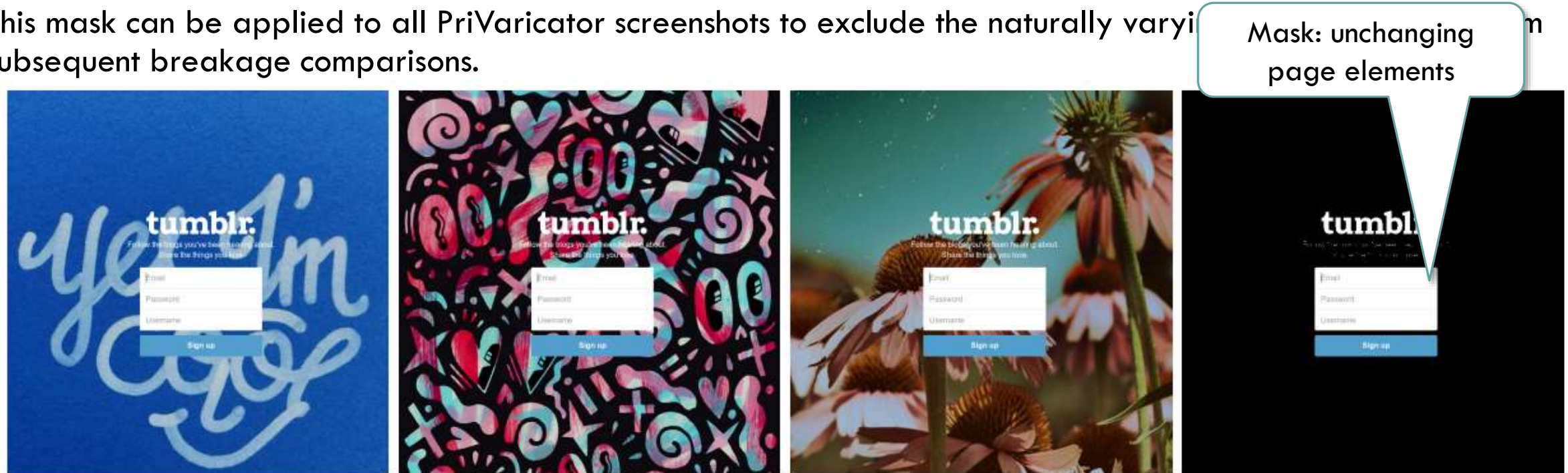
MEASURING BREAKAGE BY COUNTING PIXELS

Need to separate breakage caused by PriVaricator from naturally dynamic web pages

Collected a new vanilla-browser screenshot every ten visits of a page, resulting in a total of five extra screenshots

We computed a visual mask of differences appearing on them, and used it when comparing a screenshot captured using a specific policy parameter combination, to the vanilla one

This mask can be applied to all PriVaricator screenshots to exclude the naturally varyi m subsequent breakage comparisons.



EXAMINING BREAKAGE RESULTS

Policy	Min	Mean	Max %
Random(0..100)	0.8%	1.5%	2.3%
Zero	0.4%	0.9%	1.4%
± 5% Noise	**0.4%**	0.7%	1.0%

Overall, the results of our breakage experiments show that the negative effect that PriVaricator has on a user's browsing experience is negligible.

Manually reviewed the 100 screenshots with the largest breakage. In only 8 cases, the differences could be attributed to PriVaricator.

In many cases, the sites would show an "in-page" pop-up asking the user to participate in a survey

Next to surveys, the reported breakage was due to missing or not-fully loaded ads, error-pages and image carousels

In one case, PriVaricator had caused a slight stretch of a site's background image. While this led to a large computed breakage, users would not notice the change if they could not compare the page with the original non-stretched version.

We likely overestimated the breakage since most of the pages with the highest reported breakage turned out to be false positives.

CHALLENGES

Transparency

We do not claim to preserve transparency in PriVaricator; indeed, this is a tough property to maintain for just about any runtime protection mechanism

A motivated fingerprinter could test for the presence of unexpected randomness, e.g., by inquiring about the dimensions of an element 100 times

A statistical attack may collect multiple readings and average them over a large number of samples, in an effort to approximate the real measurement

Lie cache

Setting up a “lie cache”, a mechanism where the browser would report the *same* false value for multiple inquires about the same, unmodified element

To break linkability, the lie cache should be reset at the beginning of every new private mode session, *i*.e., when a user is opening a private mode tab or window of her browser.

This would enhance the transparency at the cost of linkability within the same private mode session.

CHALLENGES

Future fingerprinting vectors

Just like with most defense mechanisms, more sophisticated attacks often are developed in response to them.

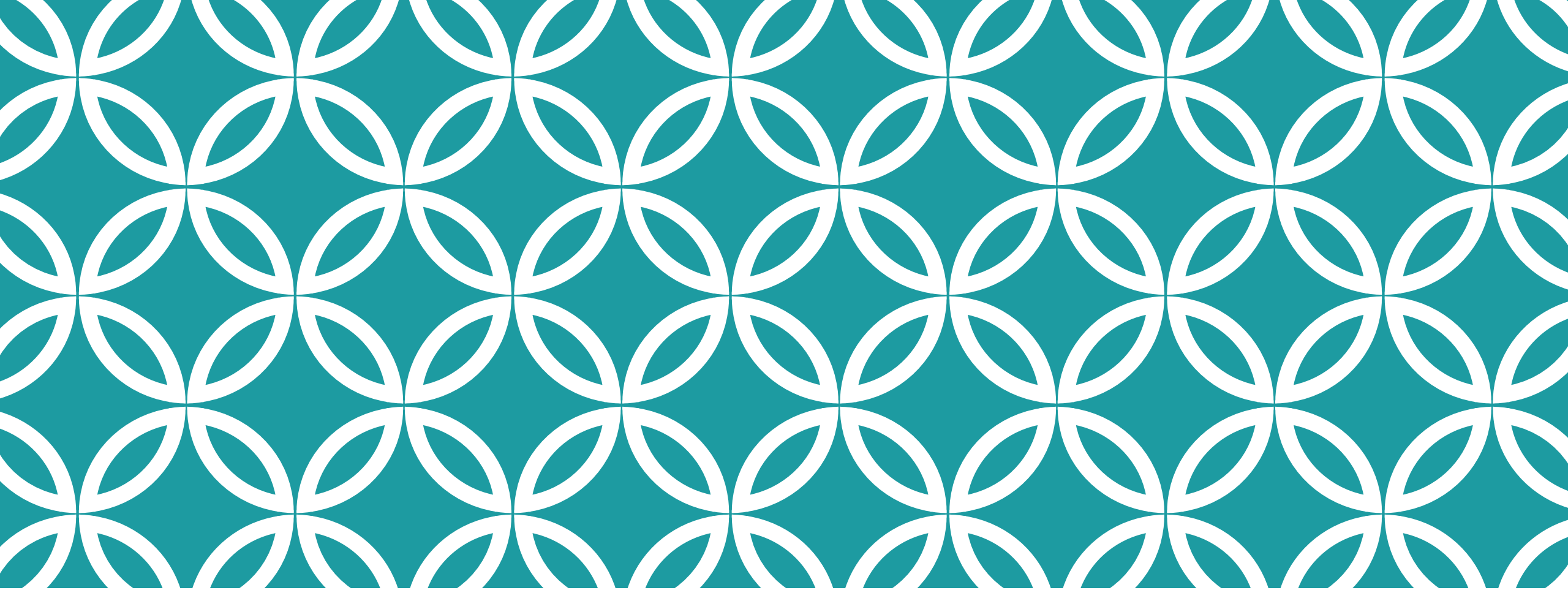
Note, however, that as long as either plugins or fonts are included as part of a user's fingerprint and relied upon to provide meaningful information to the fingerprinting party, the current version of PriVaricator is likely to provide adequate randomization

Updating policies

Fluid browser updates enable changing PriVaricator policies

Note that similar updates are shipped to other browser-hosted security mechanisms such as XSS filters, malware filters, and tracking protection lists (TPLs)

Extensions such as ad blockers also update their blacklists on a regular basis. As such, we feel that PriVaricator provides an extensible platform for stateless fingerprinting defenses



CONCLUSIONS

CONCLUSIONS

PriVaricator: an addition to the browser private mode

Designed to combat stateless tracking or fingerprinting

Negligible performance overhead

Effective for a range of policy parameter values

Breaks quite little (only a handful of sites) in our evaluation