Data flows

Hari Sundaram

Associate Professor (CS, ADV)

hs1@illinois.edu

thanks: Panagiotis Papadopoulos

Introduction

Web search

Game Theory

Auctions

Data flows

Privacy

Text Ads

Display Ads

NETFLIX

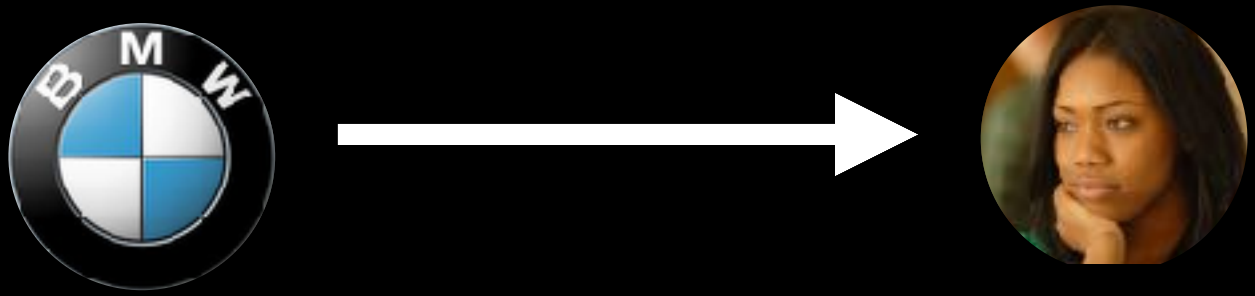
Recommender systems

Behavioral targeting

Emerging areas

Final Presentations

Google search: "**Car Sales**"



$ 1.72

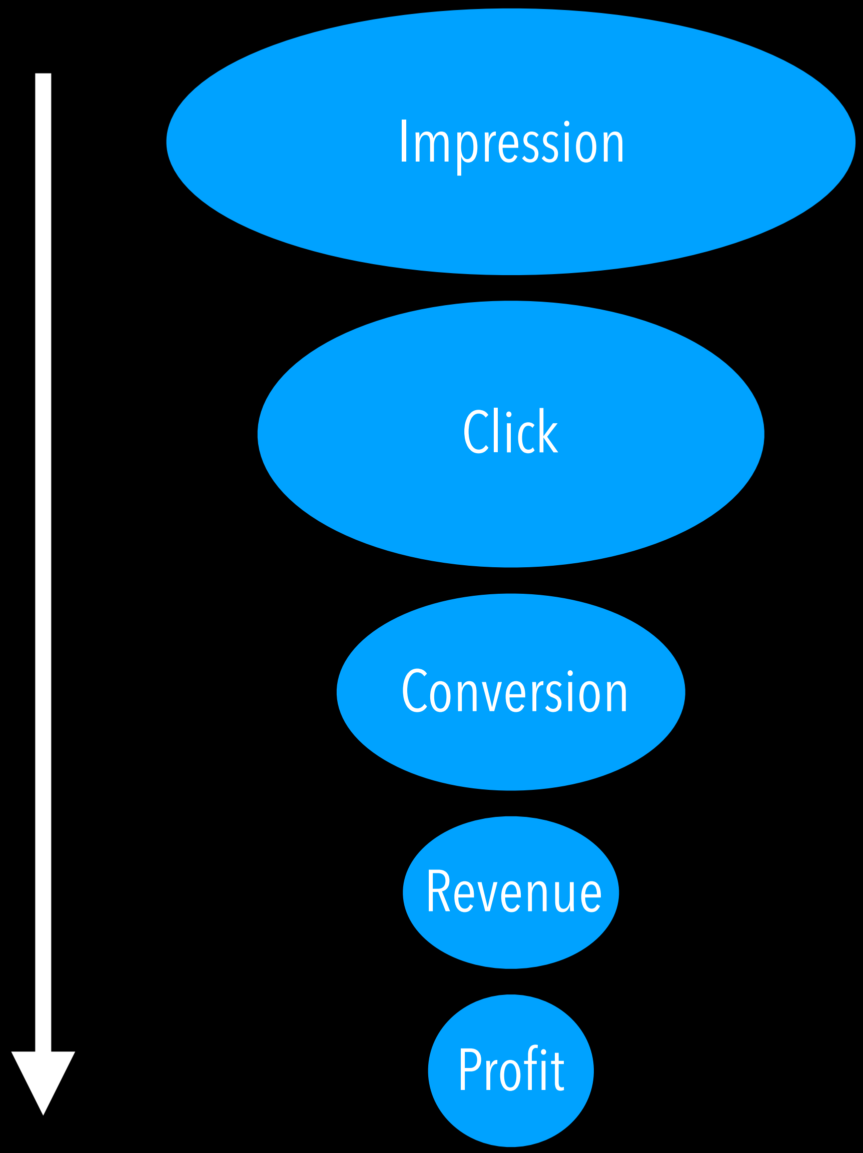
how does BMW know how much Jane Doe is worth?

$ 1.72

to bid **correctly** on an ad auction, advertisers need to build profiles

else, they might bid an amount that causes them to lose the auction

one way is to model a funnel



This is pretty coarse grained

but web-search isn’t the only place for ads!

display ads within web-sites, mobile apps etc.

how do advertisers build profiles about us?

enter the cookie

1st party:

cookies were invented to maintain state of the connection on the client;

often used to maintain login credentials at client

origins

1st party:

cookies were invented to maintain state of the connection on the client;

often used to maintain login credentials at client

use

3rd party:

cookies were invented to track users across websites

Single Origin Policy:

origin = protocol://host:port

Network access, Read/write DOM, Storage (cookies)

all **three** have to match for pages to exchange data

simple in principle, but lots of corner cases; browser implementation dependent how does an advertiser still know anything about you?

Trackers, data brokers (e.g., Axciom) and data management platforms (e.g., Cambridge Analytica, Turn) collect and process user data to form user profiles

AXICOM

TURN

enter the tracker

User profiles may contain information not only from online but **also from the offline world**:

phone number, city/state, email address, SSN, bankruptcy/education information, employment details, information on marriage/divorce, property records, etc.

Profiles are sold in data markets to advertisers for targeted advertising.

to be useful, advertisers need attribution of collected data

universal ID

gender

birthdate

browsing history

interests

sexual preferences

"ade87e60-5336-4dd9-9a2a- 763e85516f6d-tuct150ff6a"

identifying **users**

data broker id’s the user as “userABC”

the advertiser may know that same user as “user123”

how do they figure out that “userABC” and “user123” are the **same** person?

a mechanism to bypass the single-origin policy

allows web companies to share cookies, and match the different IDs they assign for the same user.

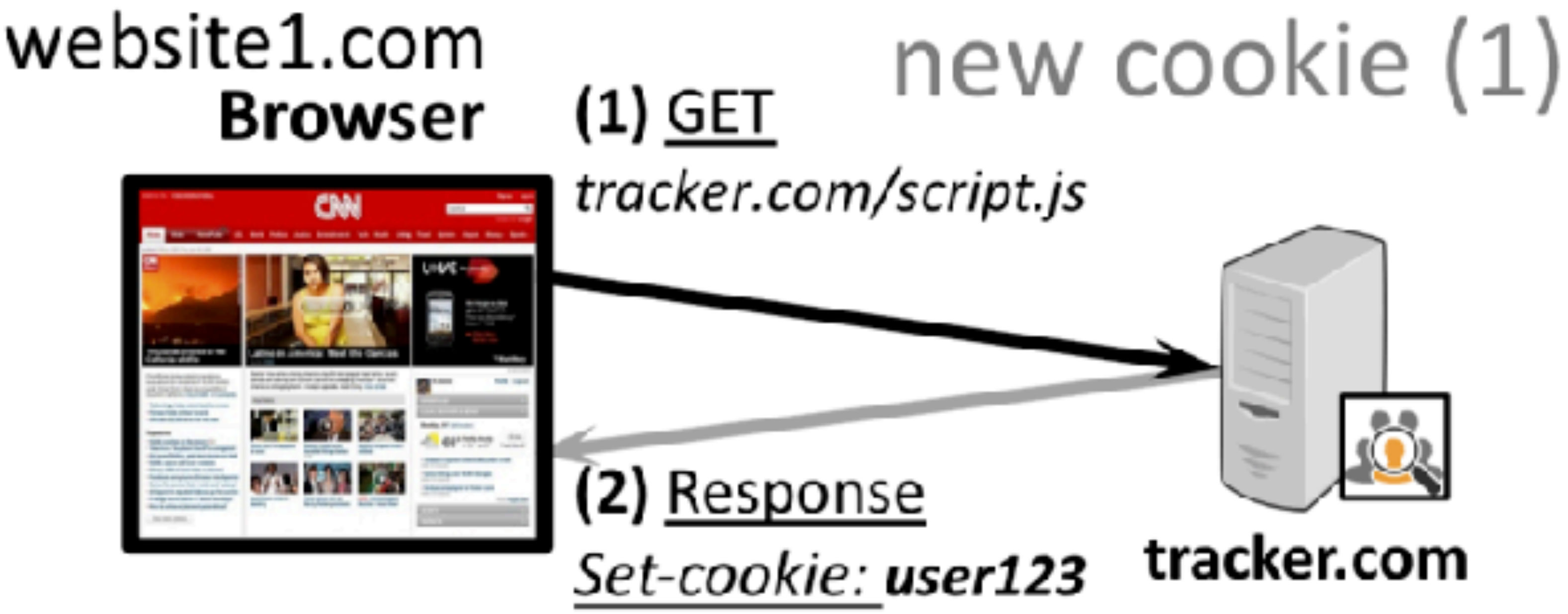
cookie **synchronization**

157 of top 200 websites (i.e. 78%) have 3rd parties which synchronize cookies with at least one other 3rd party

- they can reconstruct 62-73% of a user's browsing history*

Steven Englehardt and Arvind Narayanan. Online Tracking: A 1-million-site Measurement and Analysis. (ACM CCS '16).

cookie **synchronization**



website1.com
Browser

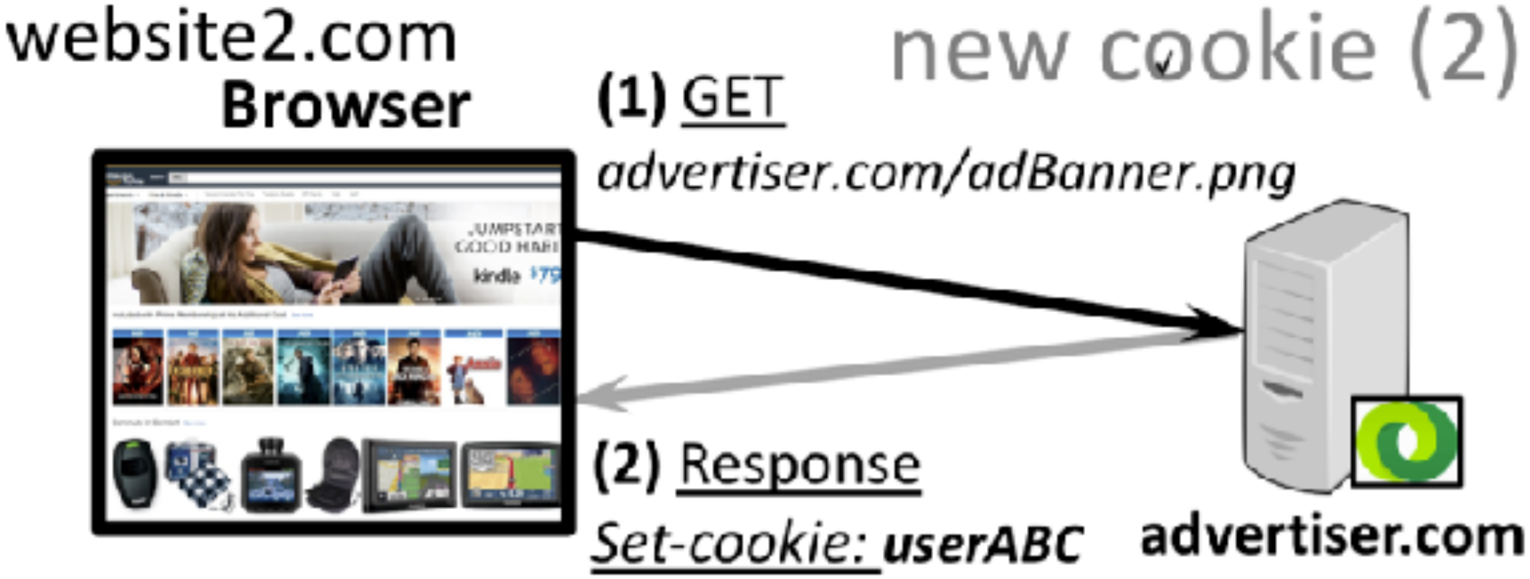
new cookie (1)

(1) GET
tracker.com/script.js

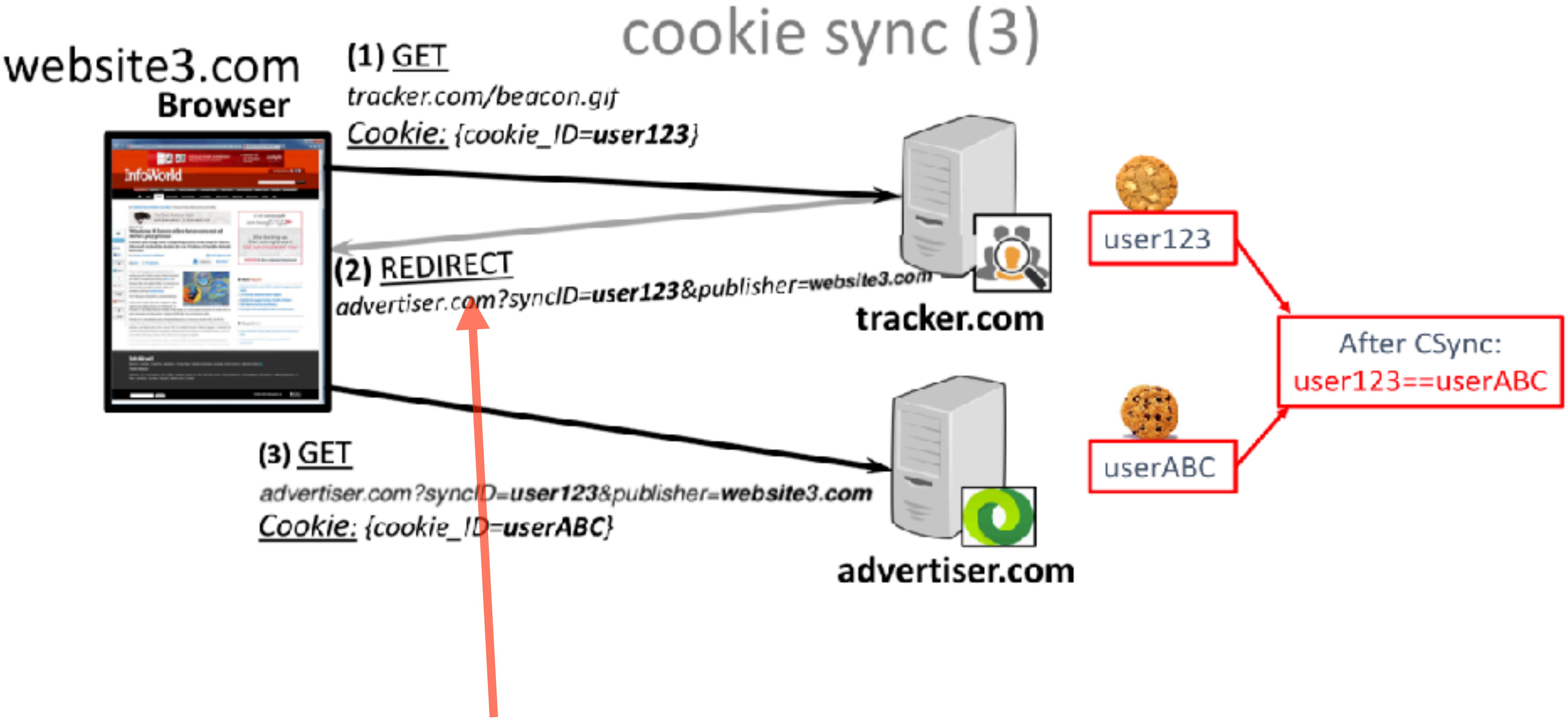
(2) Response
Set-cookie: ***user123***

tracker.com

cookie **synchronization**



cookie **synchronization**



cookie sync (3)

website3.com
Browser

(1) GET
tracker.com/beacon.gif
*Cookie: {cookie_ID=**user123**}*

(2) REDIRECT
*advertiser.com?syncID=**user123**&publisher=**website3.com***

tracker.com

(3) GET
*advertiser.com?syncID=**user123**&publisher=**website3.com***
*Cookie: {cookie_ID=**userABC**}*

advertiser.com

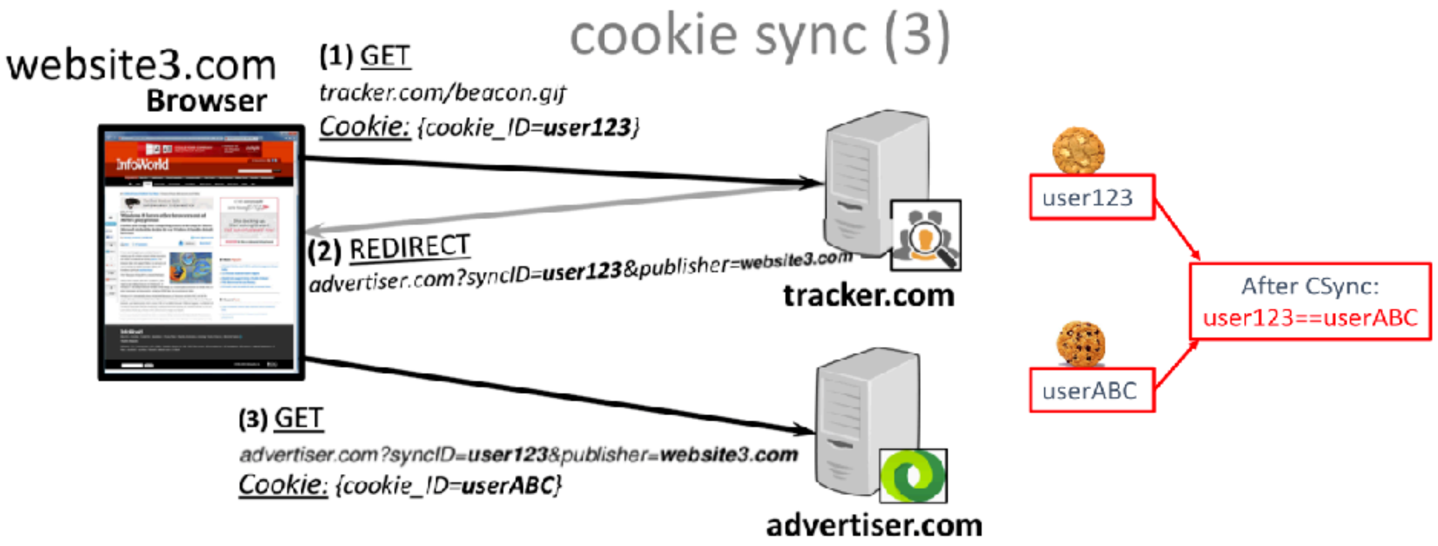
user123

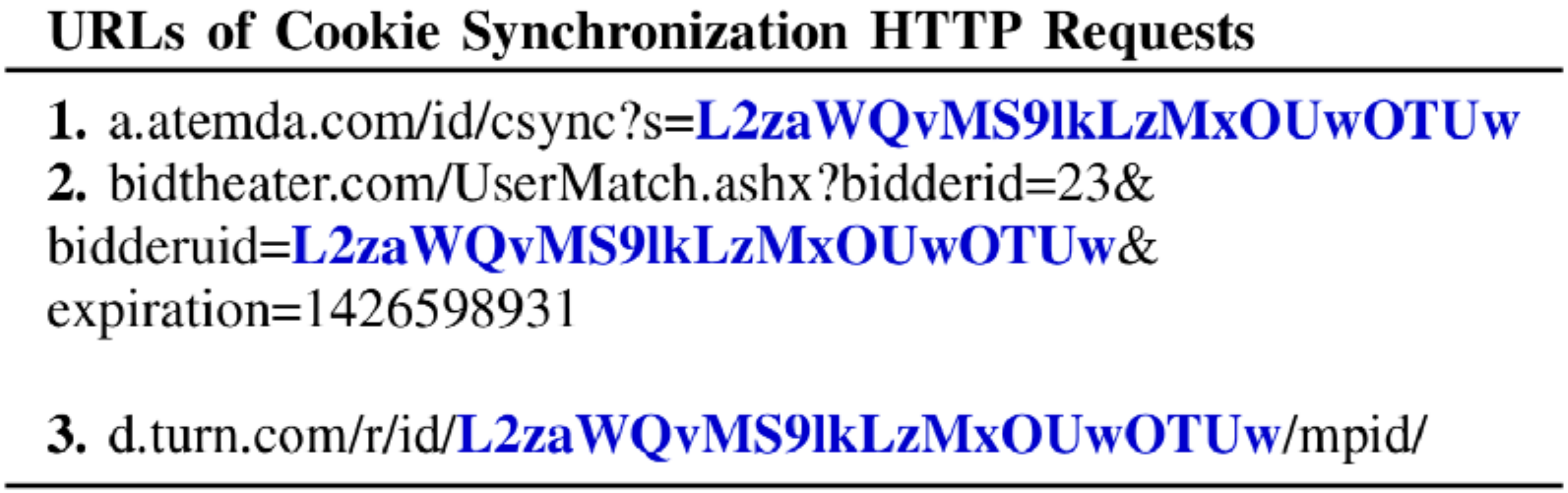
userABC

After CSync:
user123==userABC

Notice the **redirect**–why is this needed?

cookie **synchronization**


cookie sync (3)
website3.com
Browser
(1) GET
tracker.com/beacon.gif
Cookie: {cookie_ID=user123}
(2) REDIRECT
advertiser.com?syncID=user123&publisher=website3.com
tracker.com
(3) GET
advertiser.com?syncID=user123&publisher=website3.com
Cookie: {cookie_ID=userABC}
advertiser.com
user123
After CSync:
user123==userABC
userABC



URLs of Cookie Synchronization HTTP Requests

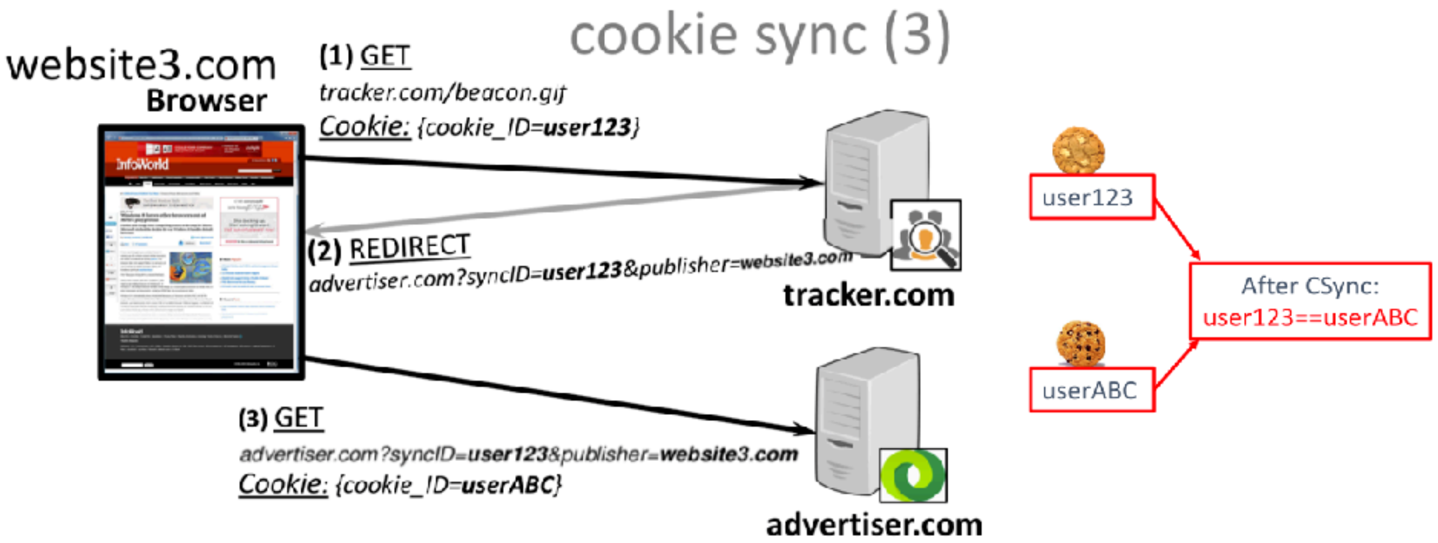
1. a.atemda.com/id/csync?s=**L2zaWQvMS9lkLzMxOUwOTUw**
2. bidtheater.com/UserMatch.ashx?bidderid=23&
bidderuid=**L2zaWQvMS9lkLzMxOUwOTUw**&
expiration=1426598931

3. d.turn.com/r/id/**L2zaWQvMS9lkLzMxOUwOTUw**/mpid/

Example real-world 3rd party synchronizations

privacy implications

cookie synchronization


cookie sync (3)
website3.com
Browser
(1) GET
tracker.com/beacon.gif
Cookie: {cookie_ID=user123}
(2) REDIRECT
advertiser.com?syncID=user123&publisher=website3.com
tracker.com
(3) GET
advertiser.com?syncID=user123&publisher=website3.com
Cookie: {cookie_ID=userABC}
advertiser.com
user123
userABC
After CSync:
user123==userABC

advertiser.com has learnt:

- the user has visited website3.com
- that the person it knew as user123 is identified as userABC on tracker.com
- Server-to-server data merges result in slow loss of anonymity

why can't we delete cookies?

coupled with **evercookie**, or user fingerprinting, CSync allows re-identification of users even after they delete their cookies

https://github.com/samyk/evercookie

Cookie synchronization in the wild

P. Papadopoulos, N. Kourtellis, and E. Markatos. **Cookie synchronization: Everything you always wanted to know but were afraid to ask**. In The World Wide Web Conference, WWW '19, pages 1432–1442, New York, NY, USA, 2019. ACM.

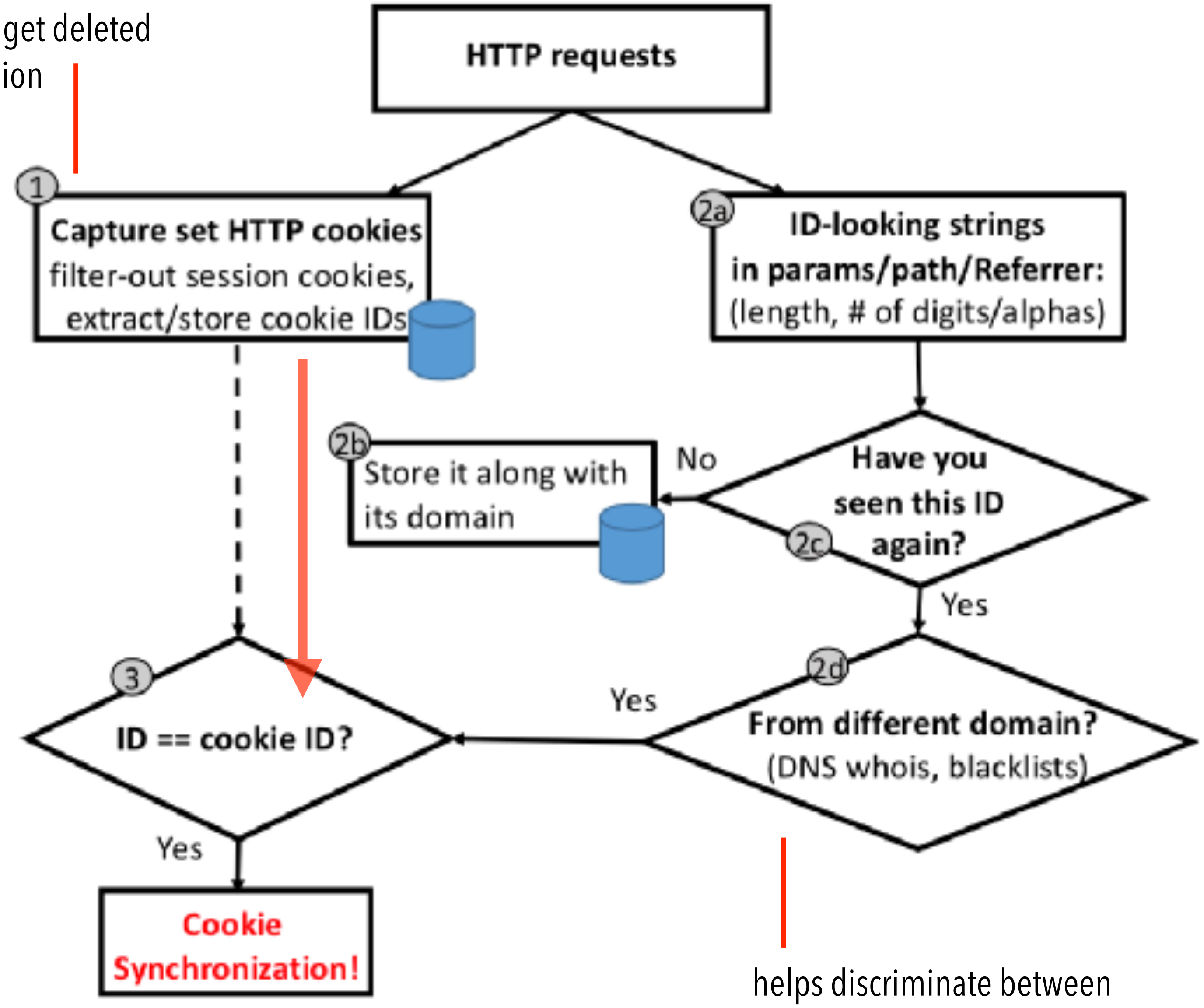
looking for cookie **synchronization**

179M HTTP requests from mobile devices of **850** volunteering users across 2016

web traffic redirection through a set of proxies

use heuristics to detect CSync.

filter out session cookies that get deleted after the session



helps discriminate between intentional ID leaking and legitimate cases of internal ID-sharing, thus avoiding false positives.

encrypted cookie **synchronization**

the previous method relies on IDs being synced in plaintext

However, major web companies such as DoubleClick have started **encrypting the cookie ID** in an attempt to protect the actual cookie from being revealed to unwanted parties that may snoop the user’s traffic (plugins or even ISPs).

encrypted cookie **synchronization**

the previous method relies on IDs being synced in plaintext

However, major web companies such as DoubleClick have started **encrypting the cookie ID** in an attempt to protect the actual cookie from being revealed to unwanted parties that may snoop the user's traffic (plugins or even ISPs).

Under the traditional plaintext case of cookie ID syncing, the **same source** company can sync **independently** with **multiple 3rd-parties** for the same user cookie ID.

But, nothing prevents these 3rd party companies from **syncing IDs with each other**, and determine that they have information about the same user!

gradual loss of anonymity

with hashing or encryption, in principle, cookie syncing becomes hard and thus may go undetected.

To build this mechanism, they employ machine learning methods (e.g. **a decision tree**), which they train on the ground truth datasets created with the previous, heuristic-based technique.

detecting encrypted CSync

They analyze various features extracted from the web traffic due to CSync, and train a machine learning classifier to automatically classify a new HTTP connection as being a CSync event or not.

They make the assumption that the various features used to characterize, and eventually detect, CSync with plaintext IDs, are equally used, and have the same distributions and variability as in the CSync with encrypted IDs.

A reasonable assumption, since the companies employing encrypted IDs are not expected to change the rest of their mechanism which delivers these IDs and triggers CSync with their partners; these companies only want to obfuscate the IDs to avoid further, and unwanted, CSync.

detecting encrypted CSync

They analyze various **features** extracted from the web traffic due to CSync, and train a machine learning classifier to automatically classify a new HTTP connection as being a CSync event or not.

> They make the assumption that the various features used to characterize, and eventually detect, CSync with plaintext IDs, are equally used, and have the same distributions and variability as in the CSync with encrypted IDs.

features

EntityName: {domain of recipient company}

TypeOfEntity {Content, Social, Advertising, Analytics, Other} ParamName: {aid, u, guidm, subuid, tuid, etc.}

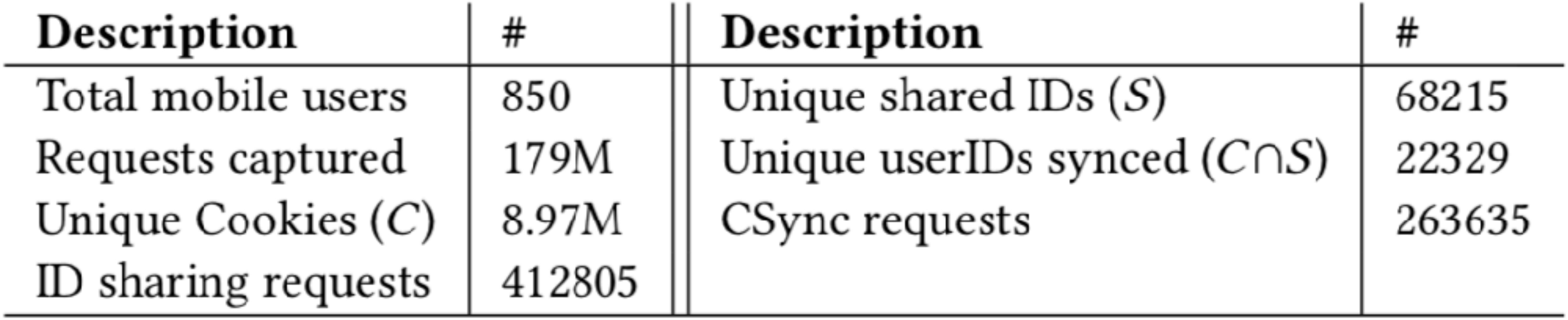
WhereFound: {parameter in URL, parameter in Referrer, in the URL path}

StatusCode: {200, 201, 202, 204, etc.}

Browser: {Firefox, Chrome, Internet Explorer, etc.}

NoOfParams: {0, 1, 2, ..., etc.}

dataset characteristics



Description	#	Description	#
Total mobile users	850	Unique shared IDs (S)	68215
Requests captured	179M	Unique userIDs synced ($C \cap S$)	22329
Unique Cookies (C)	8.97M	CSync requests	263635
ID sharing requests	412805		

dataset characteristics

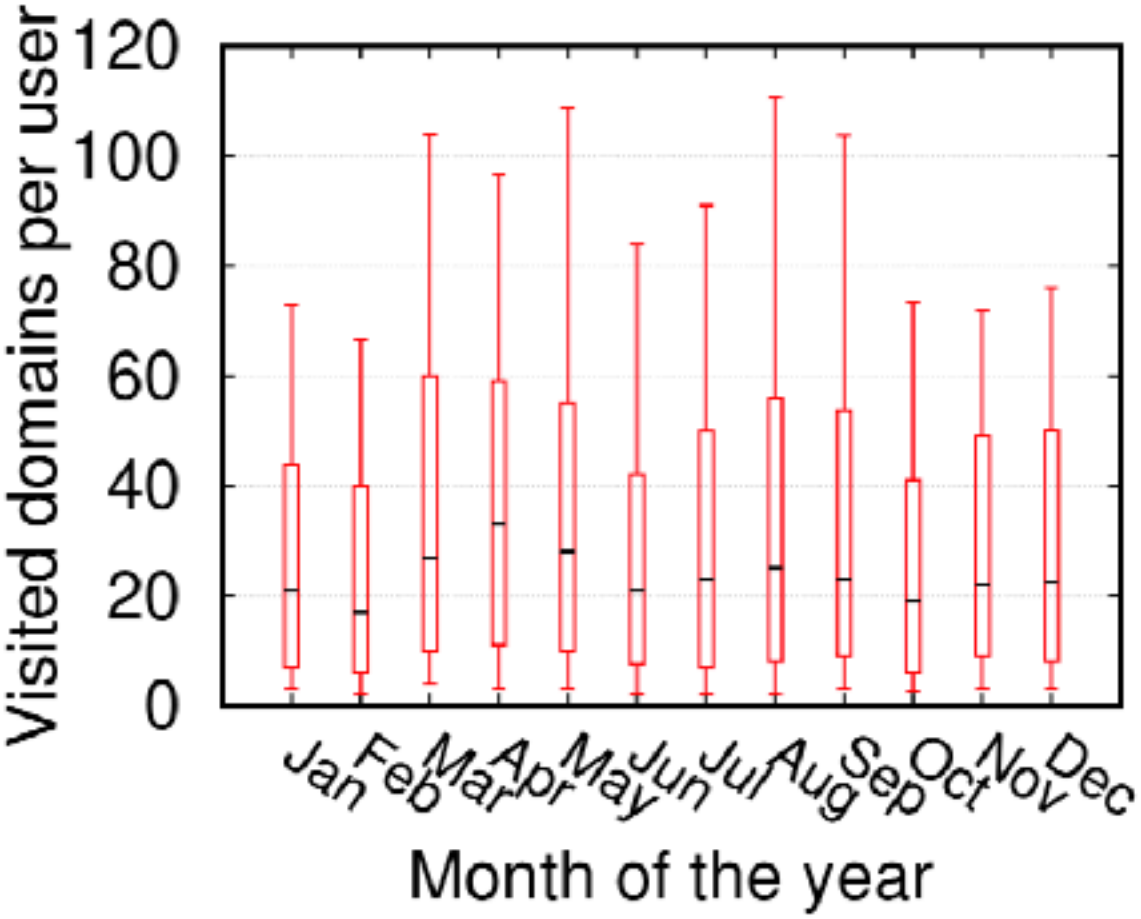


Figure 4: Distribution of number of unique domains visited per user, per month. The median user in our dataset visits 20 - 30 different domains per month.

dataset characteristics

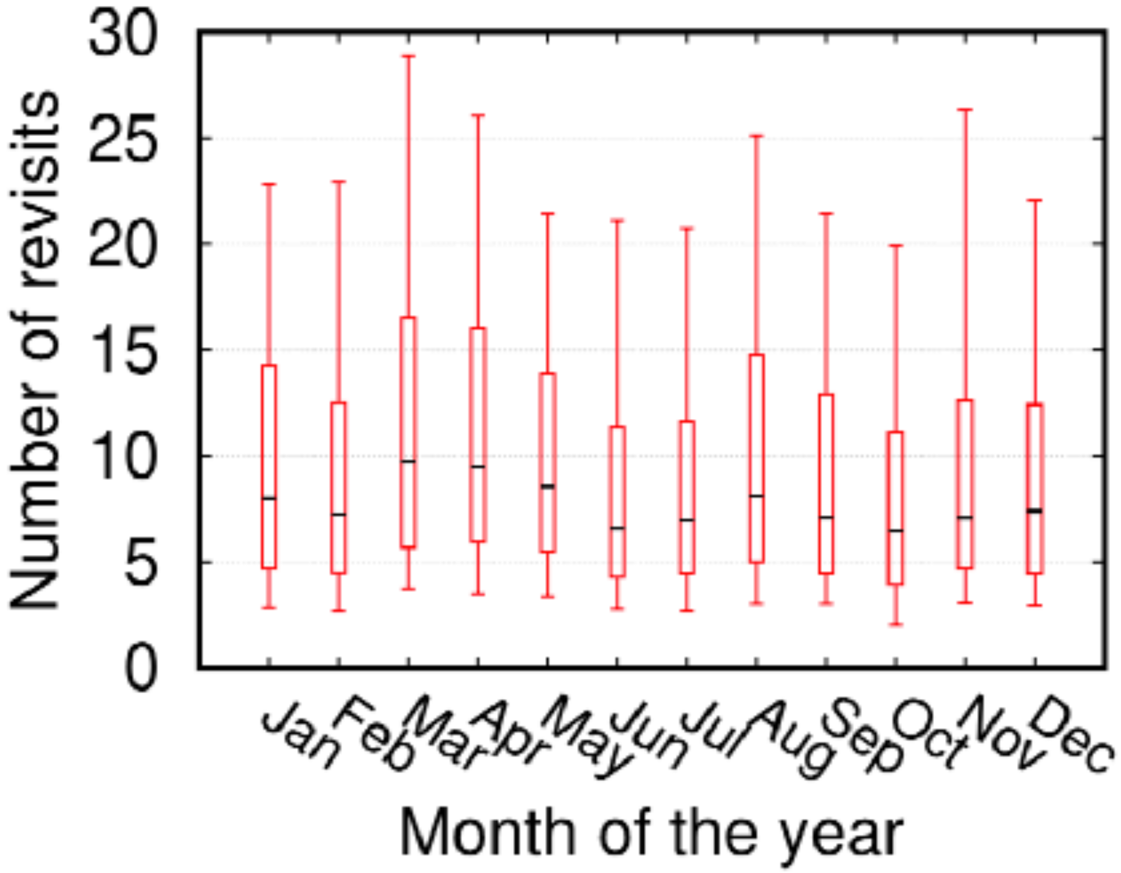


Figure 5: Distribution of number of times a user revisits the same domain per month. The median user revisits a domain around 7-10 times per month.

dataset characteristics

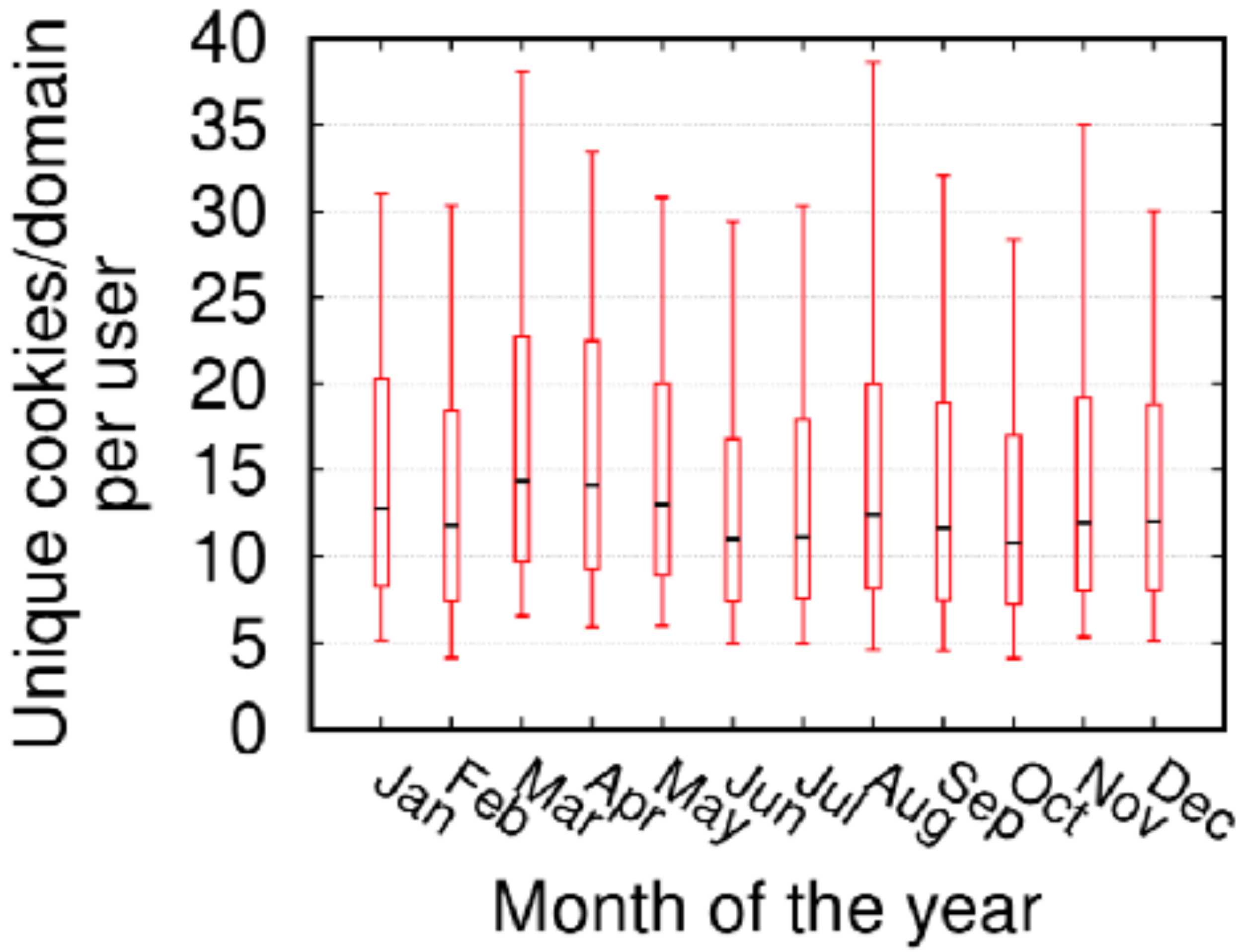
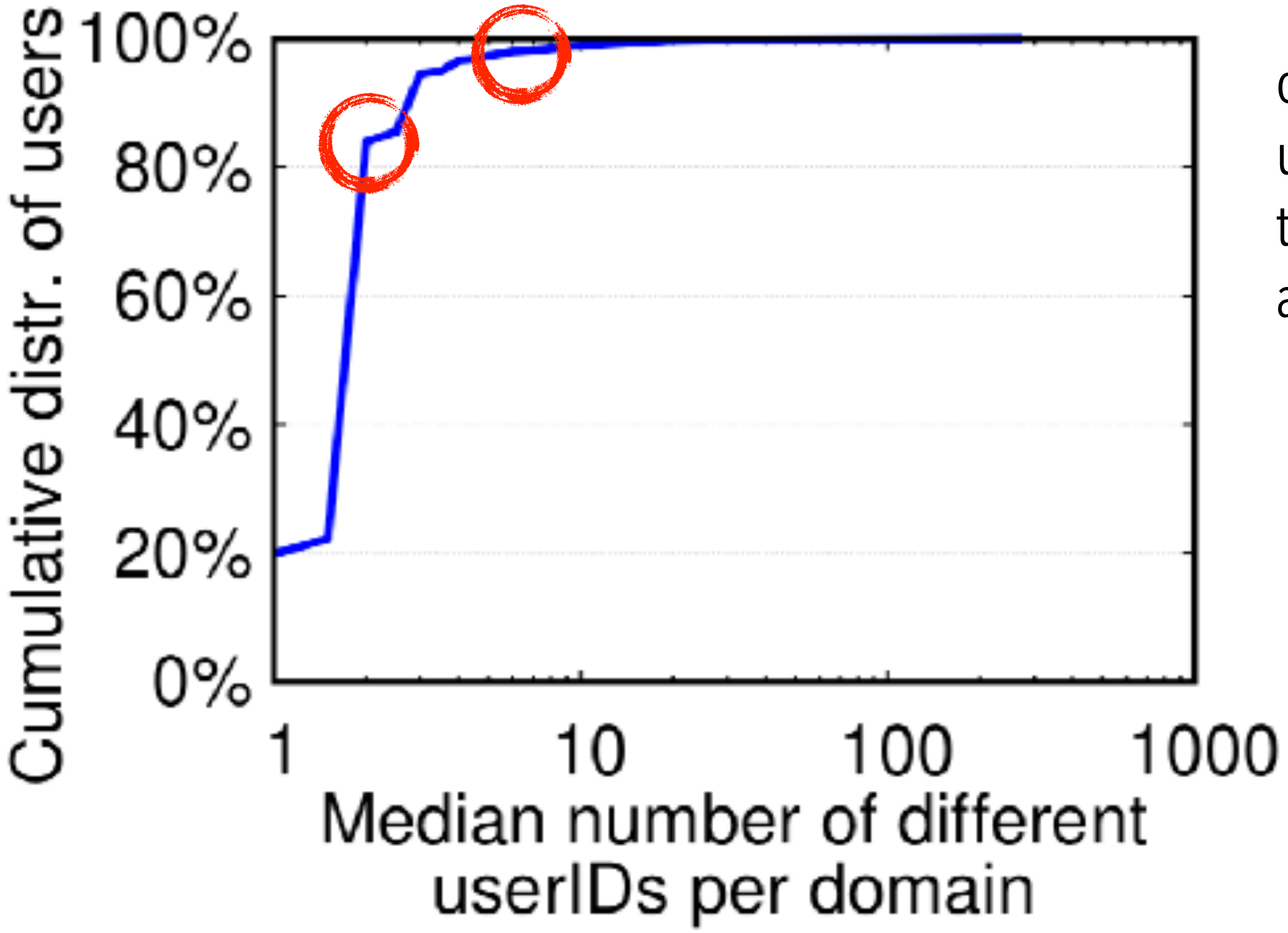


Figure 6: Number of (first and 3rd-party) cookies per domain per user. We see that the median user receives 12.25 cookies, on average, per visited website.

dataset characteristics



only **1.13%** of the users receive more than 9.5 IDs on average

Figure 7: Unique userIDs set per domain, across the year. 80% of users are known to a single domain with only ~2 aliases, on average.

this implies that most users **don't erase their cookies regularly**

# affected users

97% of users with regular activity the web (>10 HTTP reqs/day) affected

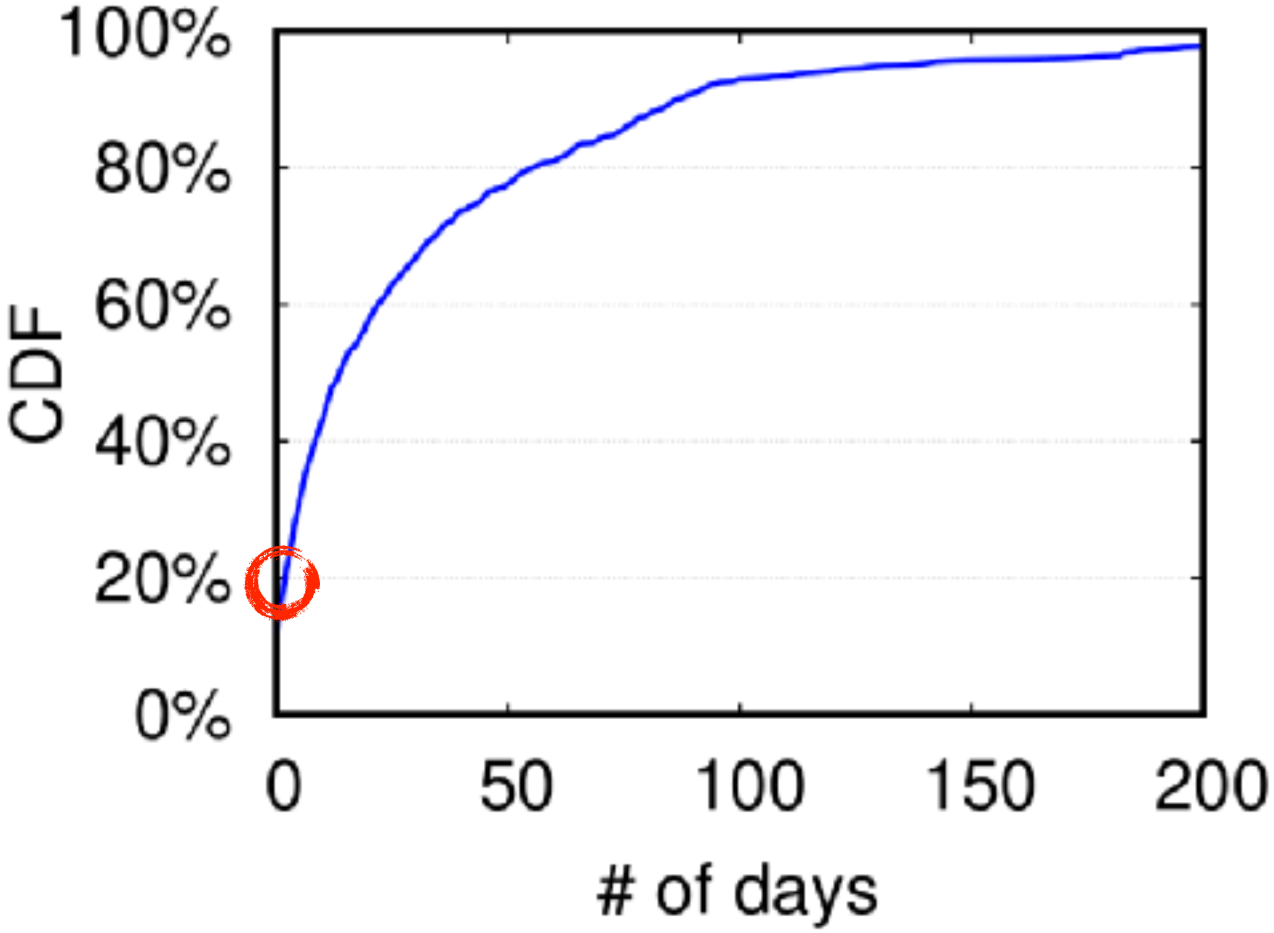
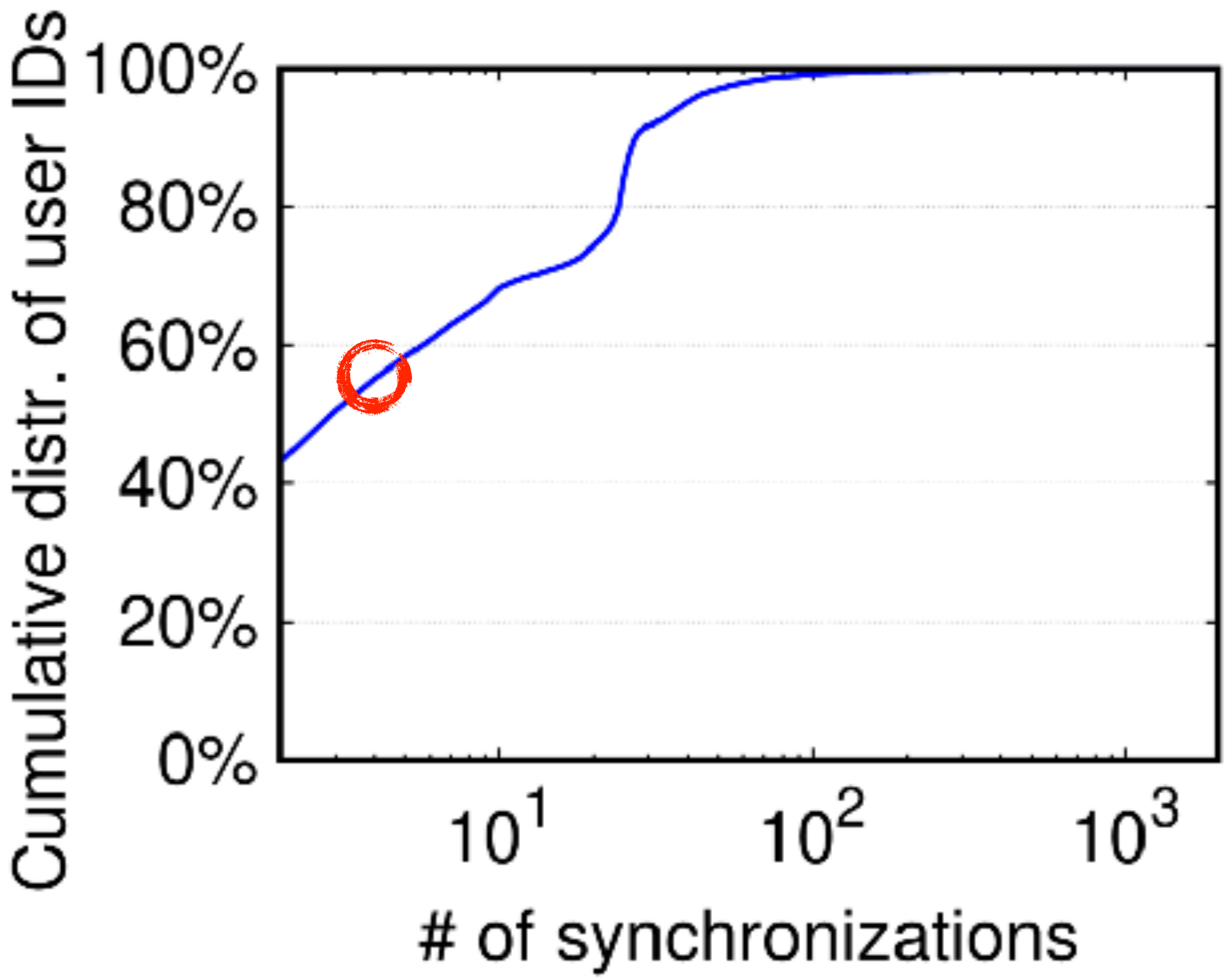


Figure 8: Distribution of time taken for first CSync to appear per user. 20% of users get their first userID synced in 1 day or less.

synchronizations per ID



the average user receives around **1** synchronization per **68** requests.

a **median** user gets up to **6.5** userIDs synced, and **3%** of users has up to **100** userIDs synced.

Figure 11: Distribution of synchronizations per userID. The median userID gets synced with 3.5 different domains.

who initiates?

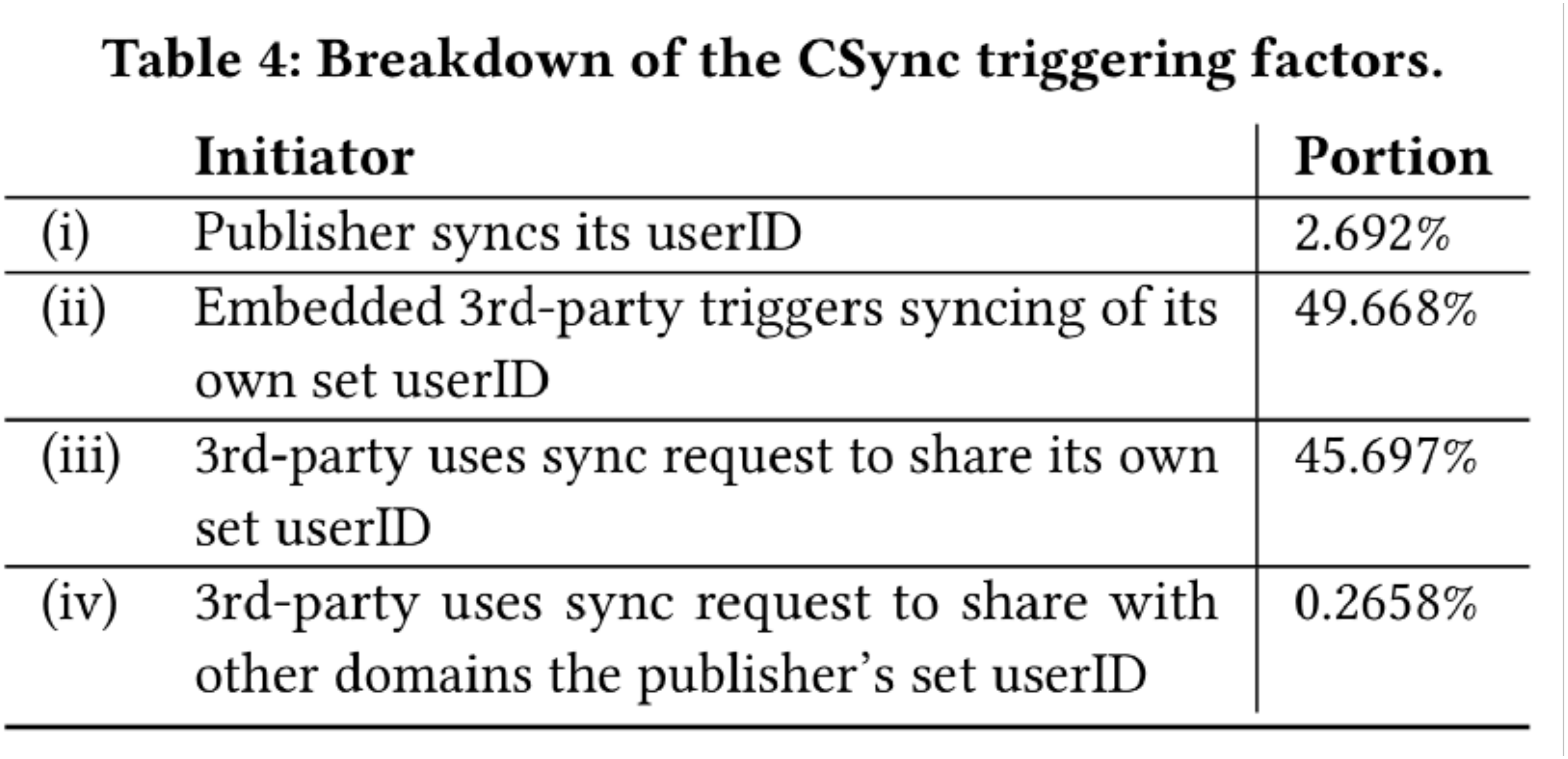
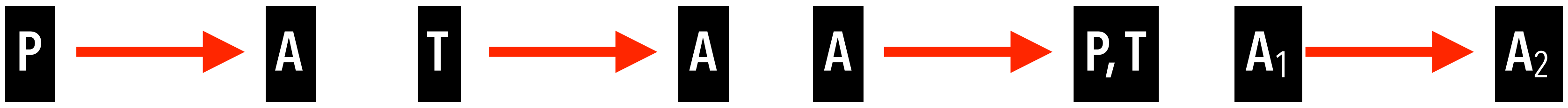


Table 4: Breakdown of the CSync triggering factors.

	Initiator	Portion
(i)	Publisher syncs its userID	2.692%
(ii)	Embedded 3rd-party triggers syncing of its own set userID	49.668%
(iii)	3rd-party uses sync request to share its own set userID	45.697%
(iv)	3rd-party uses sync request to share with other domains the publisher's set userID	0.2658%



P → A

T → A

A → P, T

A_1 → A_2

Cookie ID re-use

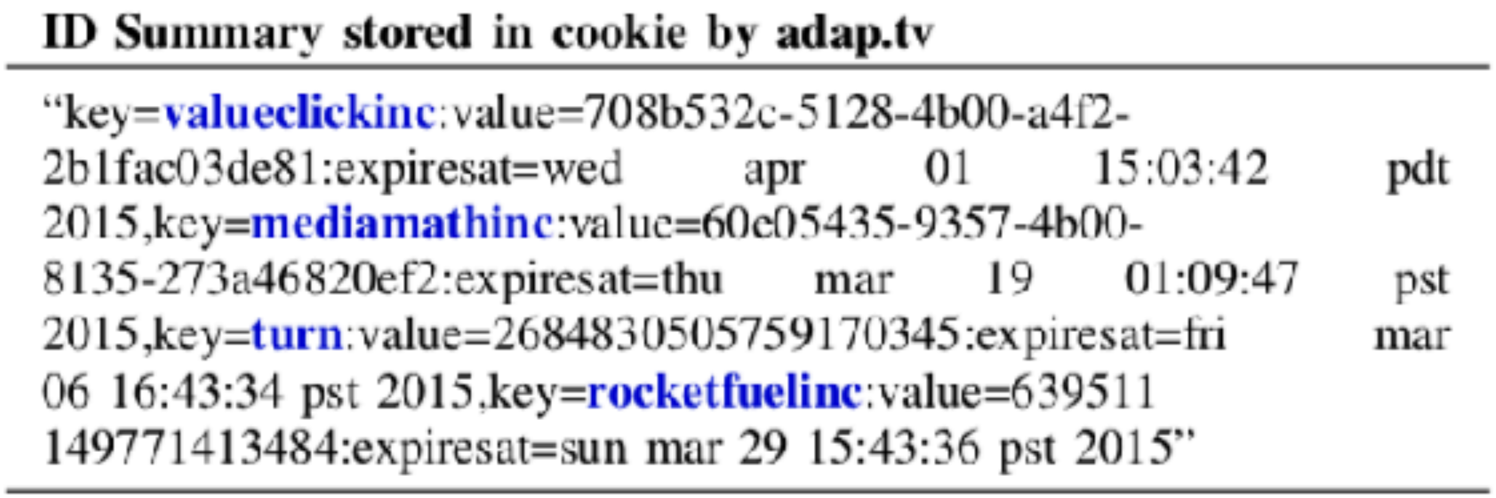
Cases of domains setting cookies using userIDs previously used by **other** domains.

Example:

baidu.com sets cookie **baiduid = {idA}**

Later, different domains set **their own** cookies by using **baiduid = {idA}**

cookies to summarize



ID Summary stored in cookie by adap.tv
“key=**valueclickinc**:value=708b532c-5128-4b00-a4f2-2b1fac03de81:expiresat=wed apr 01 15:03:42 pdt 2015,key=**mediamathinc**:value=60c05435-9357-4b00-8135-273a46820ef2:expiresat=thu mar 19 01:09:47 pst 2015,key=**turn**:value=2684830505759170345:expiresat=fri mar 06 16:43:34 pst 2015,key=**rocketfuelinc**:value=639511 149771413484:expiresat=sun mar 29 15:43:36 pst 2015”

It includes (previously synced) userIDs and expiration dates as set by 4 different domains.

how can this be possible, when the connection is HTTPS?

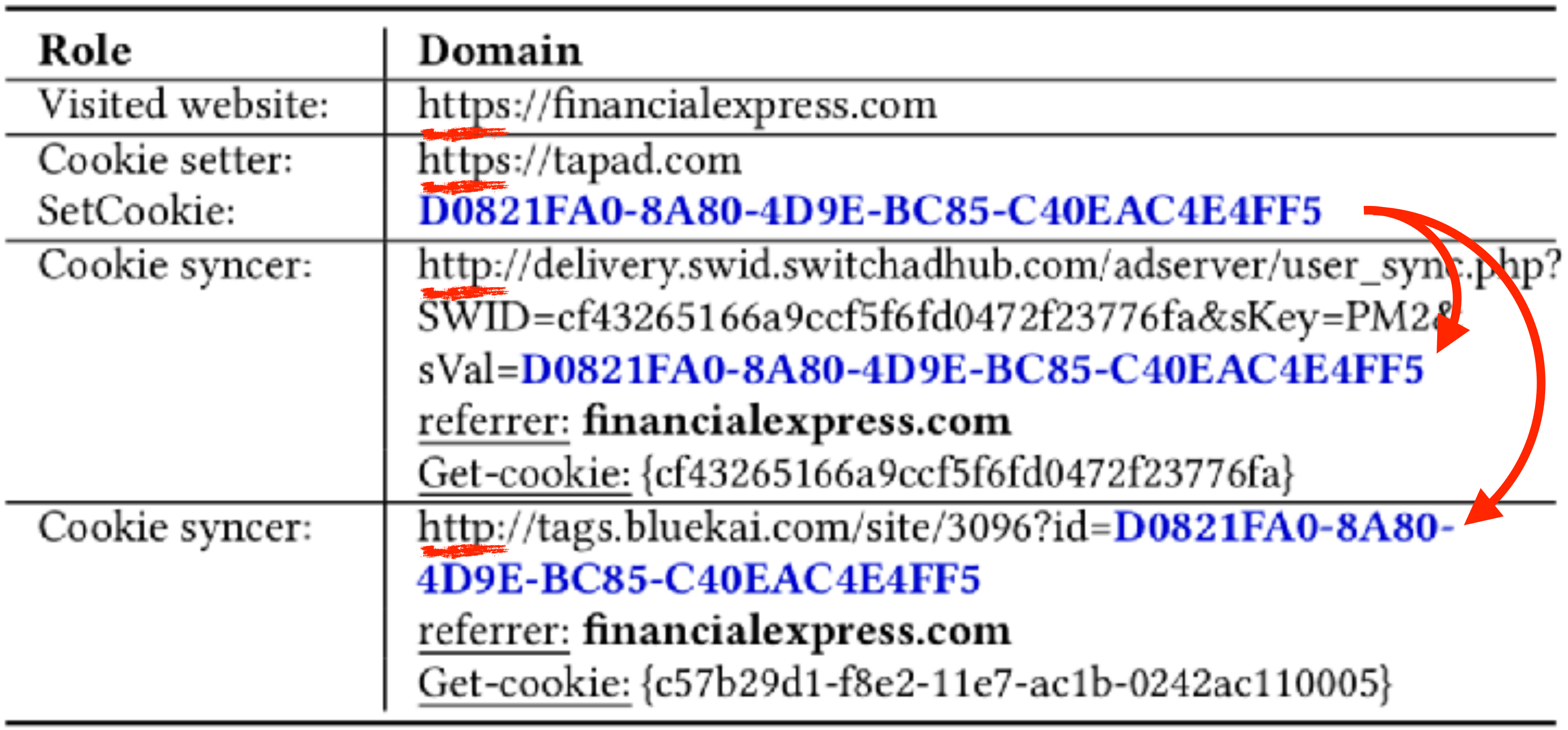
User IDs spill out of TLS!

It is caused by CSync events that sync a userID from a TLS cookie with non-TLS 3rd parties.

mixing HTTP (un-encrypted) and HTTPS (**encrypted**) is a bad idea

A snooping ISP can eavesdrop

Table 6: Example of ID-spill from SSL in our dataset.



Role	**Domain**
Visited website:	https://financialexpress.com
Cookie setter: SetCookie:	https://tapad.com **D0821FA0-8A80-4D9E-BC85-C40EAC4E4FF5**
Cookie syncer:	http://delivery.swid.switchadhub.com/adserver/user_sync.php?SWID=cf43265166a9ccf5f6fd0472f23776fa&sKey=PM2&sVal=**D0821FA0-8A80-4D9E-BC85-C40EAC4E4FF5** referrer: **financialexpress.com** Get-cookie: {cf43265166a9ccf5f6fd0472f23776fa}
Cookie syncer:	http://tags.bluekai.com/site/3096?id=**D0821FA0-8A80-4D9E-BC85-C40EAC4E4FF5** referrer: **financialexpress.com** Get-cookie: {c57b29d1-f8e2-11e7-ac1b-0242ac110005}

Top syncing categories

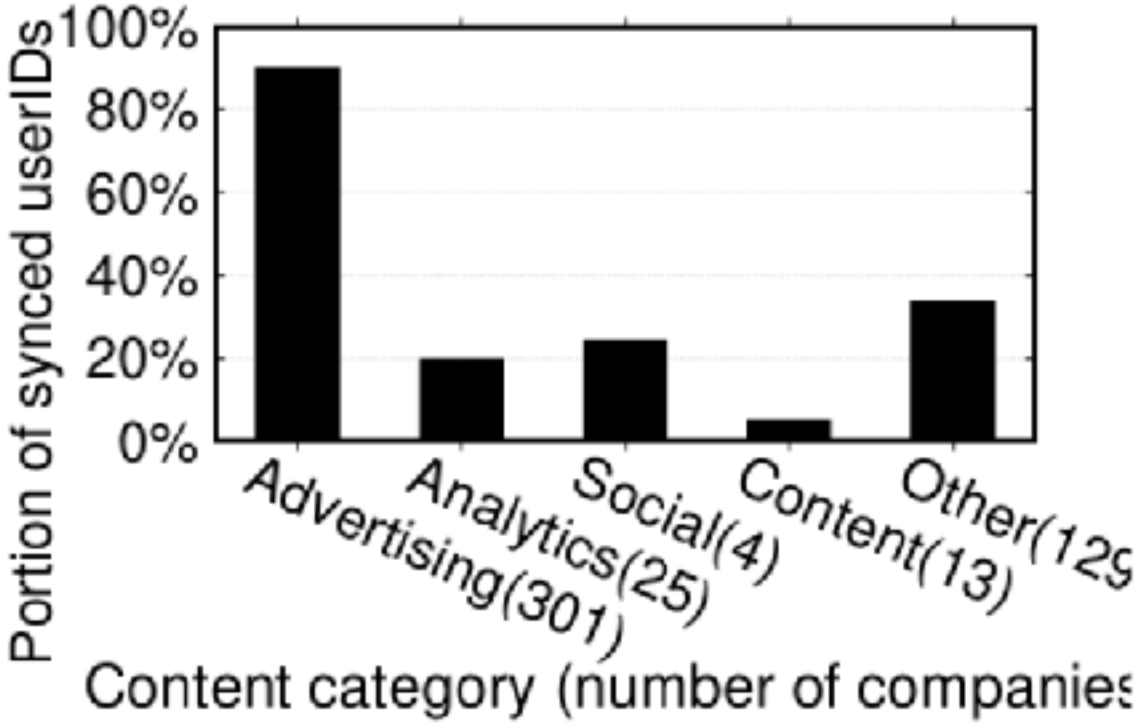
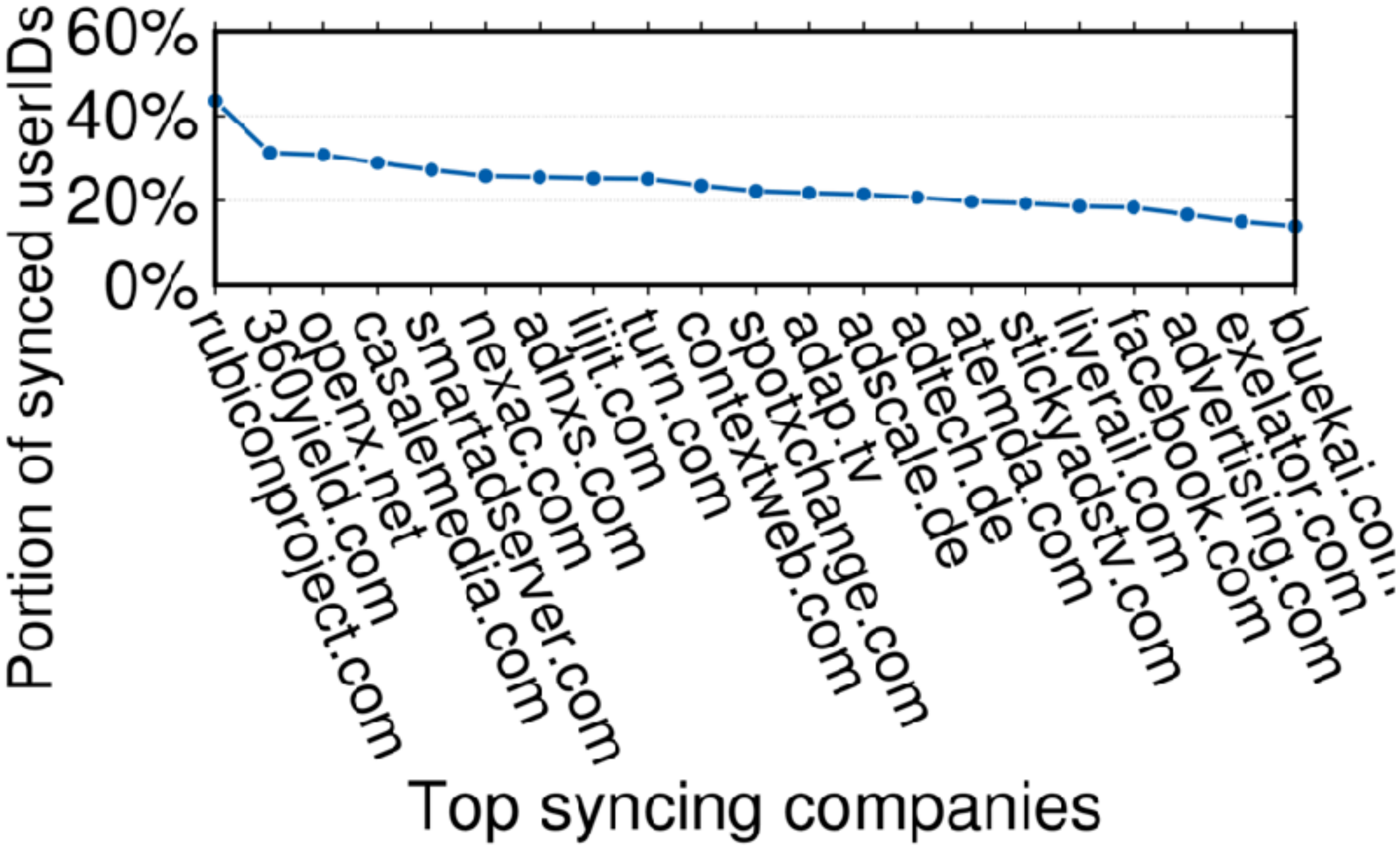


Figure 14: Portion of synced userIDs learned per content category. As expected, ad-related companies learned the vast majority (90%) of the total synced userIDs in our dataset.

Top syncing companies



Leak of sensitive information

13 syncs leaking the user’s city level location

2 syncs leaking the user’s registered phone number

10 syncs leaking the user’s gender

9 syncs leaking the exact user’s age

3 syncs leaking the user’s full birth date

2 syncs leaking the user’s first and last name

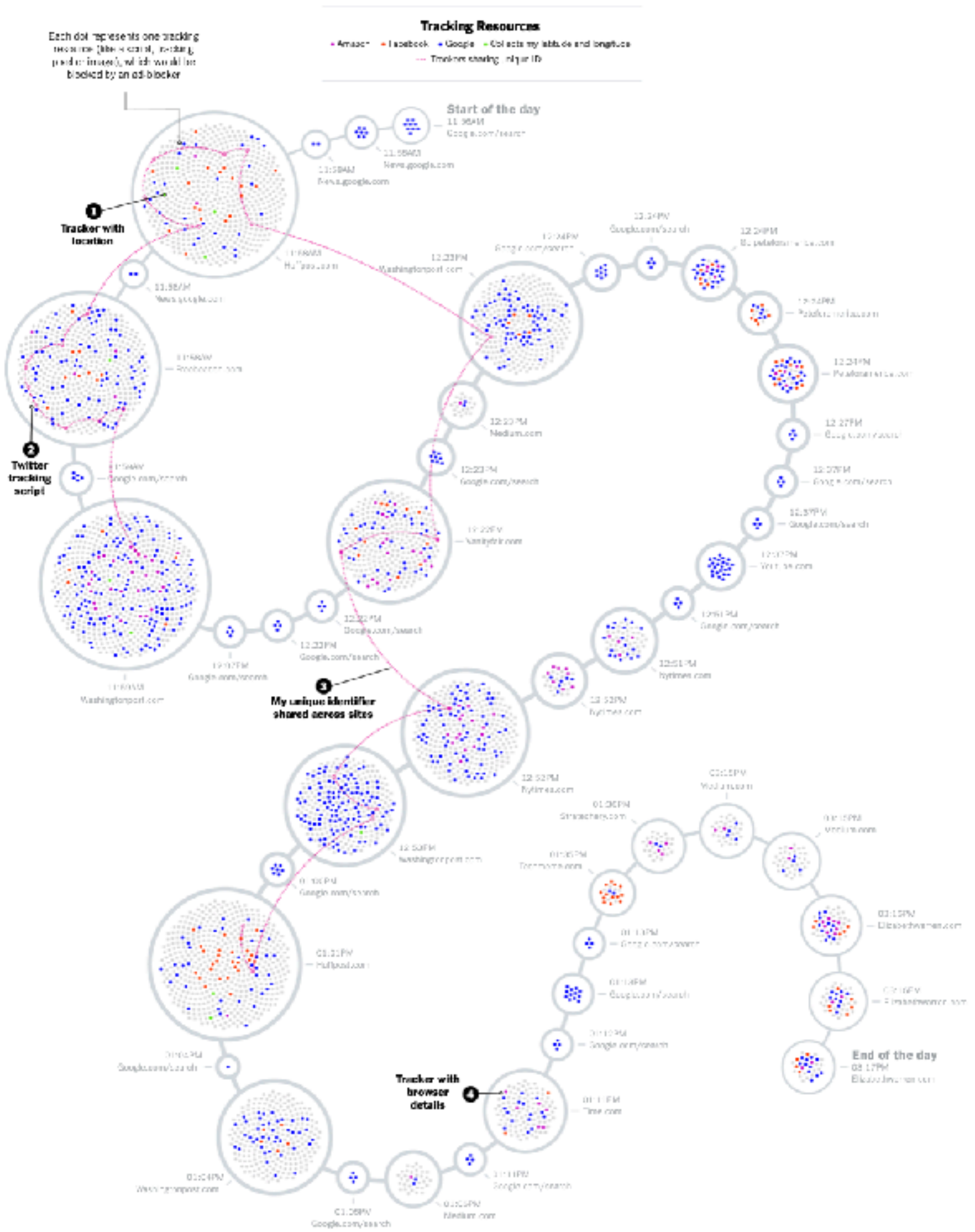
16 syncs leaking the user’s email address

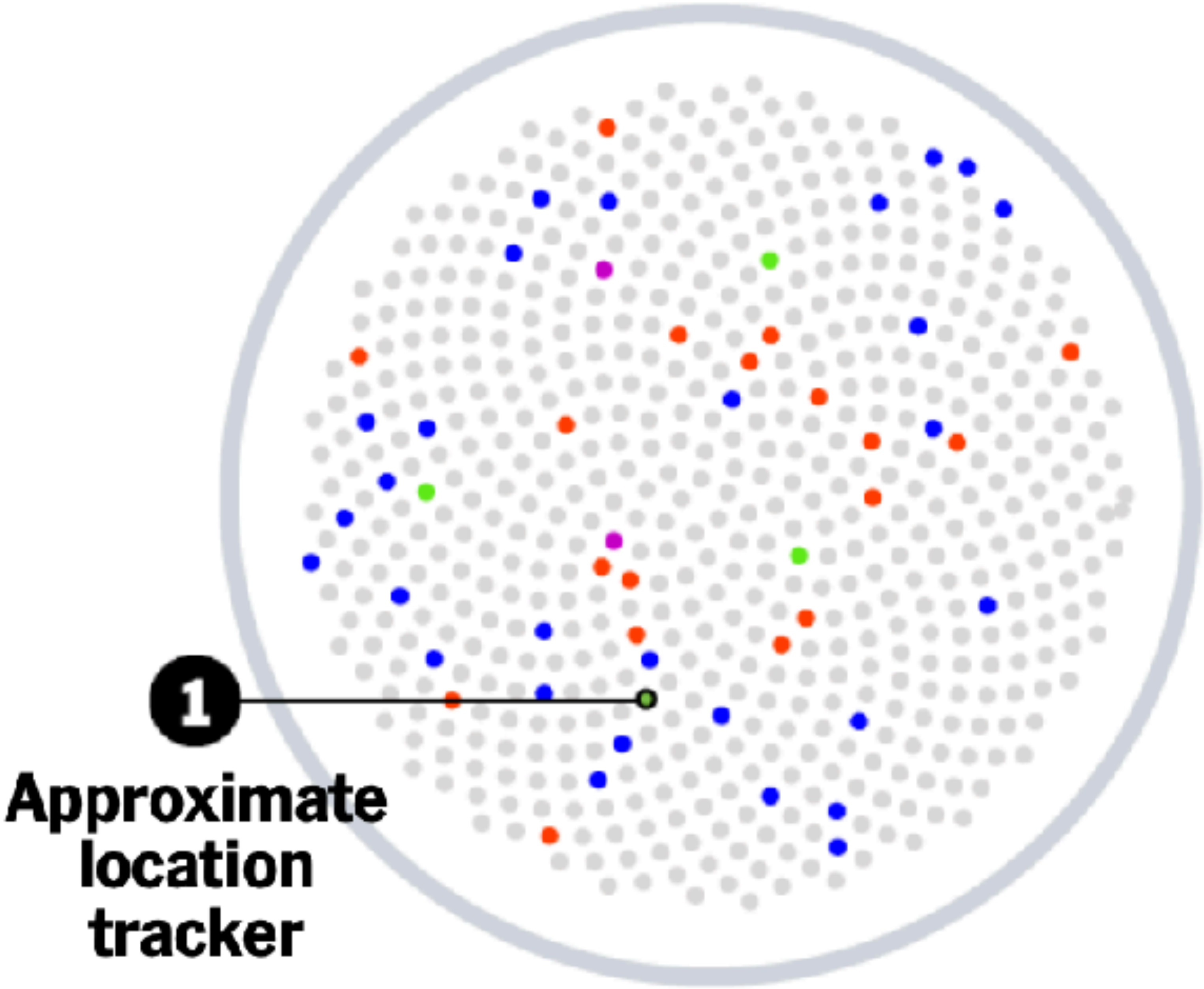
4 syncs leaking user login credentials: username/password

An example

A New York Times Columnist uses OpenWPM

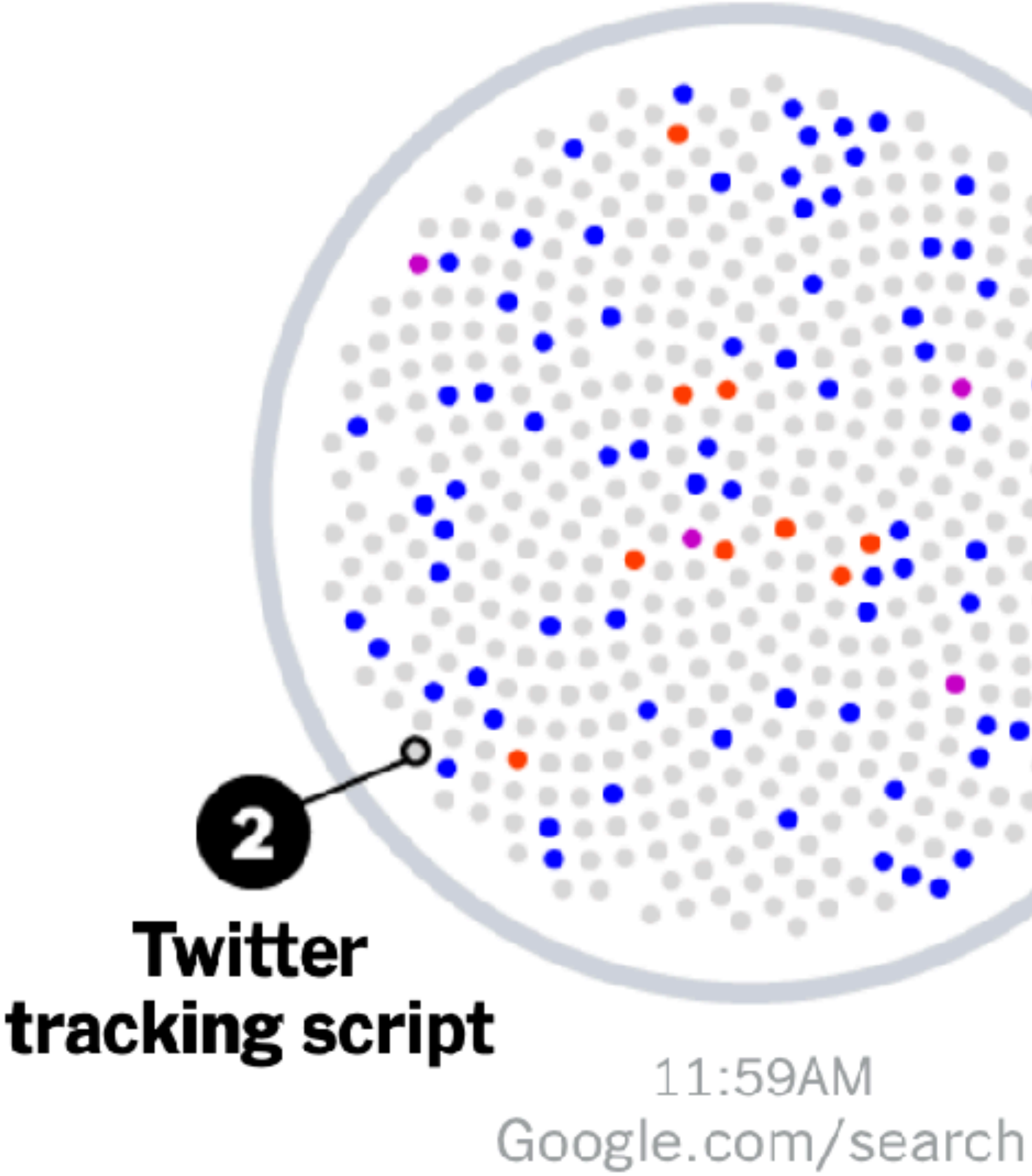
https://www.nytimes.com/interactive/2019/08/23/opinion/data-internet-privacy-tracking.html


Tracking Resources
● Amazon ● Facebook ● Google ● Collects my latitude and longitude
--- Trackers sharing unique ID
Each dot represents one tracking resource (like a script, tracking pixel or image), which would be blocked by an ad-blocker
Start of the day
11:58AM
Google.com/search
11:58AM
News.google.com
11:58AM
News.google.com
11:58AM
Huffpost.com
1
Tracker with location
11:58AM
News.google.com
11:59AM
Facebook.com
2
Twitter tracking script
11:59AM
Google.com/search
Washingtonpost.com
Google.com/search
12:22PM
Google.com/search
12:23PM
Google.com/search
Vanityfair.com
12:23PM
Google.com/search
12:20PM
Medium.com
12:23PM
Washingtonpost.com
Google.com/search
Google.com/search
12:24PM
Google.com/search
12:27PM
Google.com/search
12:37PM
Google.com/search
Youtube.com
Google.com/search
12:51PM
Nytimes.com
12:52PM
Nytimes.com
3
My unique identifier shared across sites
12:52PM
Nytimes.com
12:53PM
Washingtonpost.com
Google.com/search
01:21PM
Huffpost.com
Google.com/search
01:04PM
Washingtonpost.com
01:08PM
Google.com/search
01:09PM
Medium.com
01:11PM
Google.com/search
4
Tracker with browser details
Time.com
Google.com/search
Google.com/search
Google.com/search
Techcrunch.com
Stratechery.com
Medium.com
Medium.com
End of the day
03:17PM



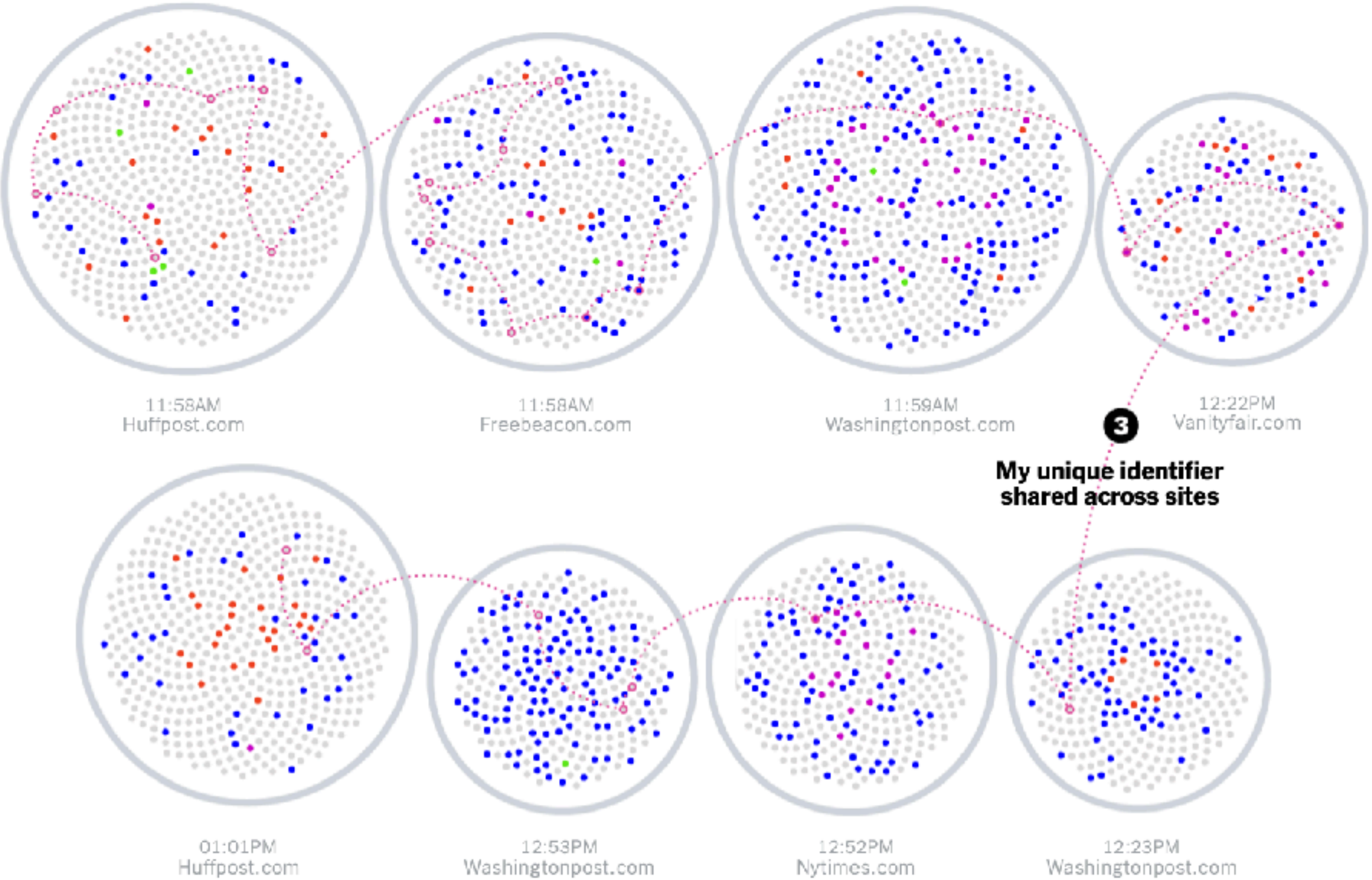
11:58AM
Huffpost.com

Where I live



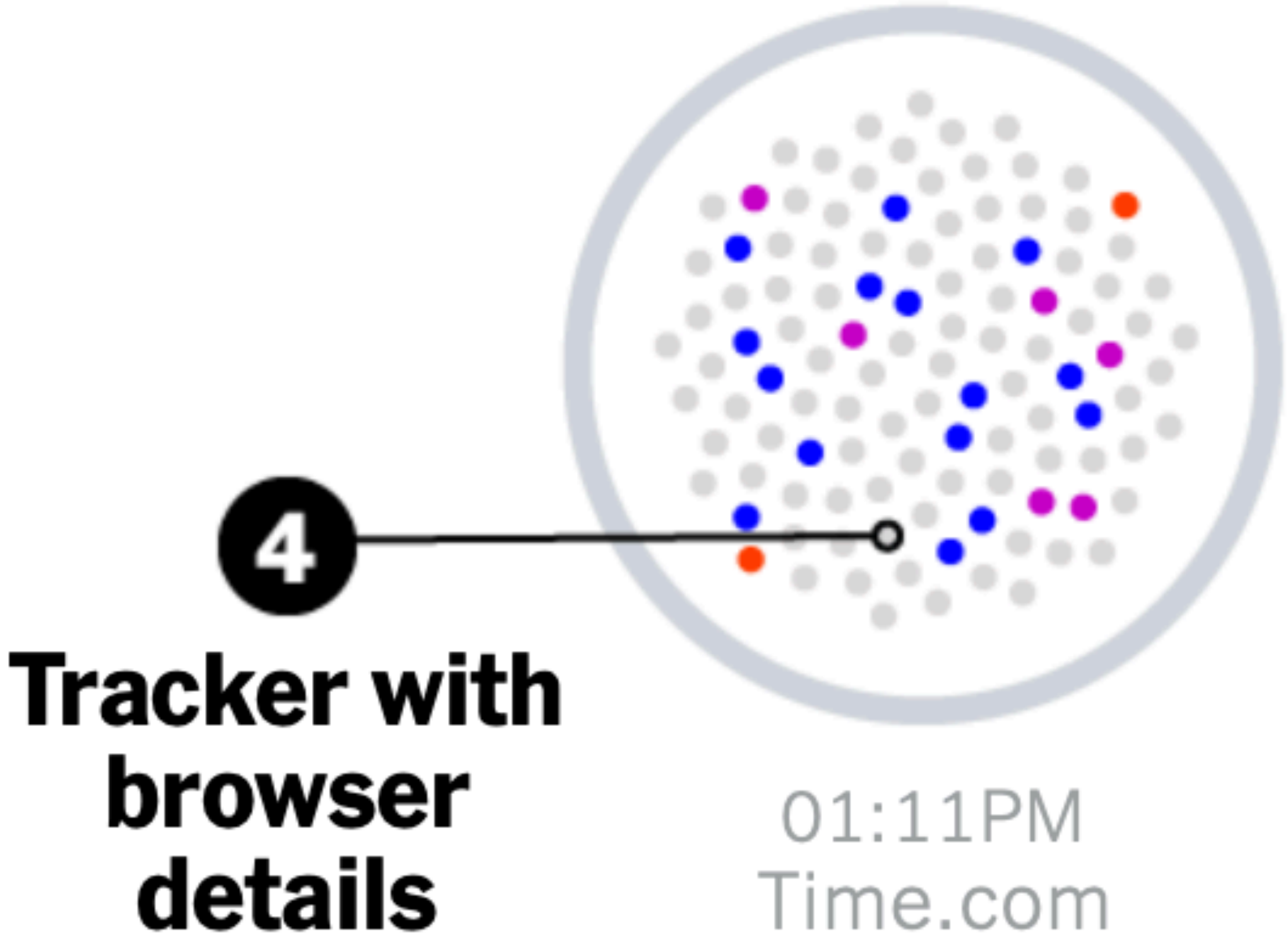
“Tracking scripts like this one for Twitter allow websites to add useful features like share buttons. But the scripts often double as trackers meant to record site visits and build profiles about users. In this case, Twitter can use the information about this page to suggest new followers or sell more targeted advertising on its platform.”

Widgets or trackers?



My unique identifier: 5535203407606041218

Fingerprinting



“Even when companies don’t have an ID to track me, they can use signals from my computer to guess who I am across sites. That’s partly why trackers like this one received more information about my computer than you could imagine being useful, like my precise screen size. Other trackers received my screen resolution, browser information, operating system details, and more.”

Counter-measures?

Ad blockers!

97% of users are exposed to CSync at least once. The median user is synced at least once within the first week of browsing.

The average user receives around 1 synchronization per 68 HTTP requests, and gets up to 6.5 of their userIDs synced.

The number of domains that learn about the median user after CSyncs grows by a factor of 6.75.

The median userID gets leaked to 3.5 domains, on average.

Summary:

Cookie synchronization is the basis of much of the data collected by Ad companies

Ad-related domains participate in more than7 5% of all CSync through the year, learning as much as 90% of all synced userIDs.

Sensitive information (e.g., gender, birth dates) is sometimes passed to the syncing domain along with the userID.