Journey to the Center of the Cookie Ecosystem: Unraveling Actors' Roles and Relationships

Iskander Sanchez-Rola*, Matteo Dell'Amico*, Davide Balzarotti†,
Pierre-Antoine Vervier*, Leyla Bilge*
*NortonLifeLock Research Group
†EURECOM

***Abstract*—Web pages have been steadily increasing in complexity over time, including code snippets from several distinct origins and organizations. While this may be a known phenomenon, its implications on the panorama of cookie tracking received little attention until now. Our study focuses on filling this gap, through the analysis of crawl results that are both large-scale and fine-grained, encompassing the whole set of events that lead to the creation and sharing of around 138 million cookies from crawling more than 6 million webpages.**

Our analysis lets us paint a highly detailed picture of the cookie ecosystem, discovering an intricate network of connections between players that reciprocally exchange information and include each other's content in web pages whose owners may not even be aware. We discover that, in most webpages, tracking cookies are set and shared by organizations at the end of complex chains that involve several middlemen. We also study the impact of *cookie ghostwriting*, i.e., a common practice where an entity creates cookies in the name of another party, or the webpage.

We attribute and define a set of roles in the cookie ecosystem, related to cookie creation and sharing. We see that organizations can and do follow different patterns, including behaviors that previous studies could not uncover: for example, many cookie ghostwriters send cookies they create to themselves, which makes them able to perform cross-site tracking even for users that deleted third-party cookies in their browsers. While some organizations concentrate the flow of information on themselves, others behave as dispatchers, allowing other organizations to perform tracking on the pages that include their content.

I. Introduction

Web user tracking is at the center of the public attention, both for the key role it plays in the web advertising industry, related to big scandals such as Cambridge Analytica [1], and for being the target of recent major legislation efforts [2, 3, 4].

Because of its impact on the users' privacy, the web tracking panorama has been extensively studied from different angles [5, 6]. For instance, researchers have looked at the technical aspects (such as the problems of cookie sharing and syncing among web actors [7, 8]), but also at the legislation impact and compliance [7], and at the information provided to users and their behavior [9].

Web cookies play a fundamental role in user tracking, but previous measurements of cookie tracking focus mainly on two aspects of the problem, i.e., on identifying and measuring a) which domains are associated to cookies created in the browser, and b) which domains share cookies with other domains. These studies, however, are based on a very simplified model of a web page, which fails to capture the complex and subtle interaction between snippets of code included from many different organizations. In fact, because of the same-origin policy, cookies are traditionally associated with the owner of the webpage (or iframe) they are created in, even when the actual code responsible to set or share the cookie belongs to a third-party entity. To make things worse, this code can be retrieved and executed only at the end of a long chain of inclusions, which may involve several different organizations. This has several important consequences. First, the owner of the website is often unaware of who actually creates first-party cookies in his own pages and of the complex "journeys" these cookies go through when they are later read by and sent to different actors. Second, as a result of this previously simplified model, cookie sharing was only studied in the context of third-party cookies – thus capturing only the tip of the iceberg of the tracking panorama. Finally, by identifying precisely which piece of code creates the cookies, we can discover cases of "*cookie collisions*", in which two different actors included in the same website end up creating cookies with the same name, thus overwriting each other.

Another problem with previous studies is that they considered cookie sharing as a two-party operation, where a cookie from a domain A was sent over an HTTP request to a domain B. However, in reality, the operation involves three actors: who creates the cookie in the first place (which, as we will see later in the paper, might not even be aware of the fact that it is later shared with others), who retrieves it from the browser and includes it in an HTTP request, and who receives the cookie at the end of the chain. In this paper we will see how this finer-grained distinction enables us to discover many interesting phenomena such as, for example, scripts that share with their own domains cookies that were created by different and completely unrelated organizations.

The goal of our study is to look under the hood of a web page and capture the entire *life* of a cookie, from its creation to all the operations into which it is later involved, and uncover the intricate network of relationships between the myriad of actors that take part into the cookie ecosystem. For this purpose, we define the concepts of *cookie trees* and *creation* and *sharing chains*, which enable us to capture the dependencies and relations between entities that act as both end-points and middlemen in the cookie ecosystem. We also introduce the new concept of *cookie ghostwriting*, which

relates to cookies that are set for a party (e.g., the website the user is visiting at the moment), but are actually created by a different entity (e.g., a script loaded from an advertiser).

We then performed a large-scale measurement study, in which we collect fine-grained details of 138M cookie creation events from crawling 6.2M web pages in 1M unique websites. Rather than simply associating events to domains, we devise techniques that enable us to map domains to organizations. Our approach enables us to discover a much larger panorama of relations between actors in the cookie ecosystem, increasing the numbers of relations by more than threefold compared to techniques in the state of the art.

The statistics reported in Section IV show that our approach uncovers a large number of previously not measured behaviors: in particular, cookie ghostwriting is an extremely common practice, showing numbers that are roughly equivalent to those of third-party cookies. For instance, creating first-party cookie from an external library would not pose privacy problems per se, but in out study we found that cookie ghostwriters often send themselves a copy of the first-party cookies they have created, making it possible for them to track even users who only accept first-party cookies (a common feature provided by web browsers). Additionally, some important actors function mostly as cookie ghostwriters, hence being essentially unmeasurable by approaches that only consider third-party cookies. Furthermore, we verify that very often cookie creation and sharing involves complex chains of intermediaries, with several middlemen between the visited website and the entities creating or receiving cookies.

In our behavioral analysis discussed in Section V, we dive deeper into the phenomena we observed in our experiments, providing both a quantitative analysis and a set of empirical examples of the different behaviors we observed in our measurement. In particular, we present cases of cookie sharing, cookie collisions, and discuss the risk brought by including potentially dangerous or malicious cookie actors. Finally, we analyze and compare the cookie ecosystems we encountered in different website categories.

We conclude our analysis of results with an investigation, presented in Section VI, of the network of links between actors of the cookie ecosystem, showing patterns of behavior that often could not be observed with previous approaches. By looking at the fine-grained data collected by our system, we discuss in details how different companies (such as Google, Facebook, RapLeaf, and WPP) often play different roles in the ecosystem. We also emphasize the connections that exist between companies, showing that some (such as AppNexus) act like dispatching hubs that connect many different advertisers.

In conclusion, this paper shows that by pinpointing exactly which piece of code is responsible for the creation and manipulation of each cookie, and how that code ended up included in a web page in the first place, it is possible to draw a much clearer and detailed picture of the cookie ecosystem and of the sharing behavior performed by tracking companies.

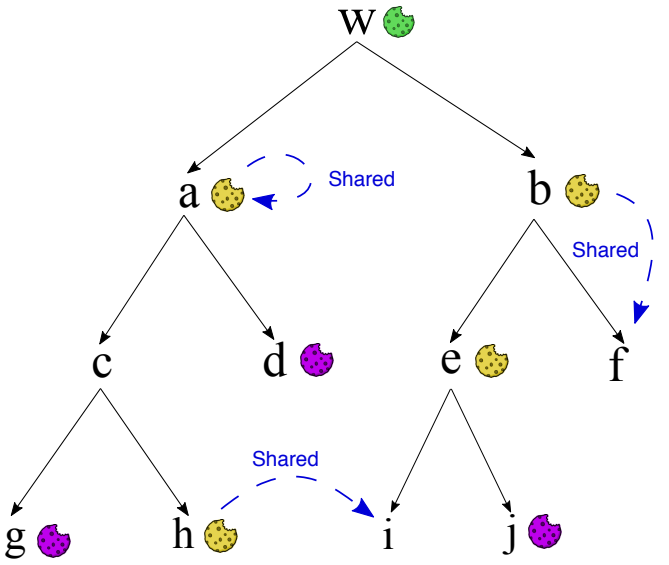

Fig. 1: Example cookie tree. Green cookies are first-party cookies created by `W`, yellow ones are ghosted first-party cookies, and purple ones are third-party cookies.

II. Cookie Trees & Cookie Flows

Traditionally, cookies are divided in just two categories: a) first-party cookies, created for the website's domain, and b) third-party cookies, created by (and for) third party domains that hosted resources requested by the website. This distinction mimics the perspective of the browsers — but as we will clarify later, it is insufficient to fully capture the complex mechanics behind the creation and sharing of cookies among different actors.

To study this phenomenon more accurately, we propose an extended version of resource trees [10, 11]. Resource trees show in a hierarchical way all resources requested by a given page. Up until now, they have been used to identify various security problems, such as inherited resource vulnerabilities [12], malicious content loads [13], or to discover third-party dependencies [14]. For instance, a resource tree can be used to show that a website A loads a JavaScript file from a domain B, which in turn loads code from C, which finally includes an image from a domain D (hereinafter we will call this a *chain* $A \rightarrow B \rightarrow C \rightarrow D$).

For our purpose, we merge information about cookie interactions with resource trees, by marking in the tree nodes that create cookies and by adding special edges to show with whom these cookies are shared. We call the resulting special-focus resource tree a **cookie tree**. For instance, Figure 1 shows a simple cookie tree, in which the main website `W` creates its own cookies (in green in the picture), and three other nodes (`d`, `g` and `j`) set third-party cookies (in purple). However, the tree also shows that other nodes are responsible for the creation of first-party cookies (represented in yellow). For instance, `h` provides some JavaScript code that also creates a cookie *as part of the website* `W`. As an analogy, this is similar to a ghostwriter who writes content that is then published under

the name of a different author. For this reason, to separate these cookies from those created by W itself, we will refer to them as **ghosted** first-party cookies.

This distinction is very important, and constitutes one of the main contributions of our paper. In fact, all previous studies considered all first-party cookies to be identical. Instead, our fine-grained view enables us to clearly understand that first-party cookies are created by a variety of different actors, and often the main website is not even aware of their existence. For instance, if W wants to be compliant with GDPR regulations, it needs to have full control over all its first-party cookies – which is very difficult when those cookies are created by other components that appear deep in the resource tree.

The cookie tree also enables us to emphasize the role of middlemen (such as c) which are not directly involved in cookie-related activities, but are responsible for initiating subsequent requests (such as to g, which in turn sets a third-party cookie). We refer to internal nodes in the tree (i.e., nodes that are neither the root nor leaves) as **intermediaries**.

According to our tree, we define the **creation chain** of a cookie as the sequence of nodes connecting the root to the node that performs a cookie creation, and similarly **sharing chain** the sequence leading to nodes that receive a shared cookie. Moreover, we talk about a **cookie flow** whenever a cookie is explicitly sent to an organization. To better understand and characterize the players involved in a cookie flow, we propose the following four main roles:

- **Cookie creators** are the entities that set cookies, either through JavaScript or through an HTTP header. Depending on the type of the cookie, we further differentiate creators in the main website page (W in our example), **third parties** (d, g and j), and **ghost cookie creators** (a, b, e, and h). It is important to stress that in this example primary and ghost cookies are both treated equally as first-party cookies by web browsers.
- **Intermediaries** are internal nodes in the cookie tree that may not directly create or handle cookies, but include resources which do that (e.g., a and c are intermediaries for the cookie set by h).
- **Senders** retrieve cookies set by other entities and explicitly send them over an HTTP request. We can further break down this role into three categories: **own** senders send their own cookies (i.e., those created by the same node, like a and b), **in-chain** senders send cookies created by one of their child nodes (i.e., for which the sender is part of the creation chain), and **off-chain** senders send cookies without being part of their creation chain (e.g., e, sending the cookie from h to i).
- **Receivers** are instead the entities that explicitly receive cookies as part of an HTTP request. We distinguish **own** receivers that obtain cookies created by themselves (a), **in-chain** receivers that obtain cookies created by their child nodes (i.e., for which the receiver is part of the creation chain), and **off-chain** receivers that obtain cookies for which they were not part of the creation chain (f and i).

A cookie flow always includes three roles: the creator of the cookie, the sender who reads the cookie and includes it in an HTTP request, and the cookie receiver. In the common case in which the receiver is the same as the sender, we say that it is a **self receiver**.

> *Example: Flickr*
> The *flickr.com* homepage includes a s.js JavaScript file from *siftscience.com* (a domain belonging to a company that provides digital trust services). The script uses a popular fingerprinting library (fingerprintjs2) to create a first-party cookie named __ssid. According to our definitions, this is a ghosted cookie, as it is associated to the *flickr.com* domain but it is in fact created by a different entity. Moreover, the same script from *siftscience.com* then takes the newly created cookie and sends it to *hexagon-analytics.com* (which provides analytics services for news and trending topics). In our model, this creates a cookie flow in which *siftscience.com* is both the ghost creator and the own sender, while *hexagon-analytics.com* is an off-chain receiver. The same cookie flow involving the exact same actors appears in several other popular websites, such as *patreon.com* and *shutterstock.com*.

While for simplicity Figure 1 does not represent it, our analysis also covers **ghosted** third-party cookies, that are created or shared by scripts included in the other contexts (e.g., through an iframe).

The roles described here above are intentionally fine-grained in order to capture all different forms of activity an actor might be involved into in the cookie ecosystem. However, the fact that many organizations own several domains open the door to misleading attributions. In fact, if for example Google's Gmail retrieves cookies created by Google Analytics, this may not be very interesting from a privacy perspective as both services belong to the same actor. Therefore, to avoid polluting our results with these less relevant cases, we decided to aggregate the nodes that belong to the same organization. For instance, if a Google-related domain loads resources from a different Google-related domain, we merge the two in a single Google node. In section III-B, we provide details about how this aggregation happens.

III. Methodology

For our study, we assembled a dataset by using the Tranco 1M most accessed domains list [15]. The list is based on the combined ranking of four large data providers: Alexa [16], Cisco Umbrella [17], Majestic [18] and Quantcast [19], thus improving confidence regarding popularity and stability of those domains over time.

A. The Analysis Platform

We implemented our cookie analysis framework on top of a custom crawler which is based on the open-source web browser Chromium (v75.0.3770.90). We used four servers

based in the US, running GNU/Linux 4.15.0-54. Each server had 64 cores with Intel Xeon E5-4600v2 series processors. The experiment was performed in July 2019 and needed 12 days to complete. The crawler is fed with the aforementioned domains list, loads the corresponding web pages, and then recursively accesses three random pages from each domain, up to a distance of three clicks from the initial main page. This results in up to 13 different pages per domain, mimicking the configuration used in other privacy-related studies [20]. To avoid the detection of our automated browser by bot detectors, we implemented the most recently-proposed methods [21, 22, 23].

Our system collects cookie information by using a custom instrumentation developed by using the Chrome debugging protocol (CDP) [24]. By tapping into its network tracing capabilities, we gather all requests and responses performed by the browser, including their headers and bodies. From the `Set-Cookie` header, we can identify the HTTP requests that create cookies. To identify cookies created by JavaScript, we modified the cookie object implementation of the browser to capture the write operations and trace them back to the exact scripts performing the actions. As a result, for each website, we output the complete cookie tree by combining the stack traces of all the specific resources and requests in the entire execution context, including asynchronous JavaScript callbacks, domain/frame redirections, and dynamically embedded scripts (e.g., `document.write`). We identify sender nodes by checking the creators of `POST` and `GET` requests with cookie content. Moreover, our framework also tracks cookie collisions which we define as an event in which a new actor overwrites a previously existing cookie on the page that was created by a different actor.

To accurately detect cookie sharing events, we follow the approaches of the most recent research studies [8, 25] on this domain, analyzing all the requests performed during the access, and attempting to decode (e.g., `BASE64`) and deflate (e.g., `gzip`) the content in multiple layers. Because the main goal of our paper is to study the privacy implications of cookies existing on web pages, we focus only on identifier cookies and therefore, construct the cookie tree only for those. We pre-filter all cookies that are unlikely to be identifiable by using the `zxcvbn` technique proposed in a recent work [26].

B. Mapping Domains to Organizations

Here we describe how we map the domains into unique organizations to attribute the cookie creation or sharing events to unique actors and to accurately identify the roles we defined in the previous sections.

We start with pre-processing the domain names that were involved in cookie related events during our crawling phase. We first find domains that use CNAME cloaking [27], which is a technique to escape ad-blockers through CNAME aliases and disguise third-party trackers first-parties. We then search those identified domain names in the NextDNS blocklist [28] to re-attribute them to existing trackers. Finally, we use `tldExtract` [29] to obtain the *private suffix* of each domain, corresponding to the largest privately-owned suffix of that domain (e.g., *forums.bbc.co.uk* becomes *bbc.co.uk*).

Once the largest privately-owned suffix for each domain is identified, we adopt two strategies to map the domain names to organizations; the first is based on three manually-curated lists (`Disconnect` [30], `WhoTracks.me` [31] and `webxray` [32]), the other is an automated approach we designed to increase the coverage. As discussed in the following, we take care of giving higher priority to manually curated lists.

Automatically finding the owner organization of a domain is a challenging task. The main reason is that the nature of inter-organization relationships are very dynamic and complex, e.g., company merges and acquisitions, or service provider companies acting on behalf of their customers in the online world. Things are further complicated by the domain registration ecosystem that does not aim at providing full transparency on the organizations behind each domain, as exemplified by the myriad of companies providing "anonymized" and "privacy preserving" domain registration services. As a result, such a mapping can only be inferred by combining and cross-checking multiple data sources.

The key idea behind our automation is to find connections among the different nodes of our graph G: *domains*, *IP addresses* and *organization names*. To model these connections, we add links to G when we discover information that associates two different elements. A first source of links is the log of our crawler: we connect domain names with the IP addresses they resolved to when they were crawled. Then, similarly to previous work [33], we furthermore parse Whois records of domain names, extracting e-mail addresses (whose domain is a potential endpoint for a link) and organization names. Finally, we use Internet Routing Registries (IRRs or IP whois [34]) to apply a similar mapping to IP addresses.

The above procedure has the risk of adding to G links that do not represent true connections between resources of the same organization. To minimize this risk, we adopt a conservative approach that vastly reduces those incorrect mappings. A first cause of potential mismatches, as discussed above, are anonymisation services which register domain names. We observe that these services register more domains than regular organizations; hence, we take a dataset of all the domains registered on the Internet [35], manually inspect the top 500 domain registrants (in terms of number of domains registered) to identify Whois privacy protection services, and we remove all nodes and edges associated with them from G. Another cause of noise are cloud hosting and CDN providers, which host on the same network several domains by different companies. Again conservatively, we discard from G all IRR data from organizations hosting 10 or more different domains.

Additionally, organization names are free-form text fields with no enforced validation. For instance, a single organization may appear in the domain Whois and in the IRRs with slightly different names, thus hindering the connection between the two. We address this challenge by performing normalization on the names (e.g., by removing special characters and company extensions) and by matching them using the normalized

Fig. 2: Pervasiveness of identifiers. Most websites and webpages include at least one identifier cookie and almost half of all cookies created are identifiers. Moreover, these identifiers are shared among multiple organizations in more than half of webpages and websites.

Levenshtein distance with a conservative 0.9 threshold. We merge organization names in G according to this approach.

We obtain our final results by returning the groups of domain names in the same community as output by the graph-based label propagation algorithm [36] applied on each connected component extracted from G. We validated our approach by using the manually compiled lists as a "ground truth" reference: we then looked for conflicts, defined as cases where our automated approach was joining two organizations that were identified as separate in one of the three manually curated lists. As a consequence of implementing our automated approach in a very conservative way to minimize false positives, out of the 3,913 domains in any of these lists, we encountered just one single conflict. In this case, a domain was grouped on the Disconnect list to a company with a similar name, while our approach associated it to another company with exactly the same information in the Whois record. Therefore, we filed a bug report to correct the mistake in the Disconnect list; they acknowledged and solved it.

We merge the four mappings to obtain a single one, taking care of solving conflicting domain attributions. We do this by first assigning a priority level to each mapping, based on our assessment of each method's precision: we gave highest priority to the manually compiled datasets, preferring those that have been updated most recently, hence resulting in Disconnect first, WhoTracks.me second, and webxray third. As a fourth data source, we use our automated approach. We then proceed according to a label-propagation algorithm that augments the highest priority mapping with all the others in descending order of priority, siding towards the results of the highest-priority mapping in case of conflicts.

IV. The Cookie Ecosystem at A Glance

We now discuss our experimental results. In this Section, we provide some high-level statistics on the cookie ecosystem; in Section V we go into the details of the cookie-related behavior we observed from the actors involved, while in Section VI we delve into the details of the tangled relationships among the cookie actors we discovered.

TABLE I: Top cookie ghostwriters.

Actor	Websites as ghostwriter	Websites as third-party
Google	175,395	283,778
Facebook	141,708	6,472
AddThis	47,250	52,648
Yandex	41,089	3,782
Baidu	19,965	1,348
Quantcast	17,902	3,039
AdRoll	10,702	1
Criteo	10,486	68
CNZZ	10,052	0
Shopify	9,657	137
ShareThis	9,486	17,589
Hotjar	9,431	3
HubSpot	8,935	372
Zopim	8,557	3
Amazon	7,883	23,358
StatCounter	7,631	2
Mail.Ru	6,573	1,914
Adobe	6,379	30,918
Tawk	5,890	11
Microsoft	5,782	19,200

A. Cookie Statistics

As explained in Section III-A, starting from a seed of 1M domain names, our system explores recursively three random pages up to a depth of three from the main page, resulting in a total of 6.2M pages crawled. During this process, we observed the creation of 137,997,677 cookies, out of which we classify 48% (66.7M) as identifiers with the `zxcvbn` technique described in Section III-A. As many as 76% of the websites we visited did set at least one cookie. After manual inspection, we observed that many of the websites that do not create cookies correspond to either empty/very basic, broken, or abandoned websites. Notwithstanding regulations that require consent before tracking users [2, 3, 4], in Figure 2 we show that almost all of the websites that set cookies do set at least one cookie that is recognized by our system as an identifier. Results presented in the remainder of this paper are based on the total of 738,168 websites that set *at least one* identifier cookie. From Figure 2, we also see that almost half of all the cookies that are set are potential identifiers. Moreover, 387,202 (52.45%) sites exhibited at least one cookie sharing or collision event.

We further determined that, among all identifier cookies, 31.1M (47%) are third-party cookies, 28M are *ghosted* cookies, and only 7.6M are primary first-party cookies that are authored by a resource of the website itself. Primary first-party cookies appear to be globally more prevalent (515K of websites) than third-party (454K) and ghosted (424K) cookies. However, websites that include third-party or ghosted cookies typically include more of these than of primary first-party cookies. On websites that include third-party (resp. ghosted) cookies, these cookies are created by an average of 3.52 (resp. 2.2) actors. This suggests that ghosted cookies are used in

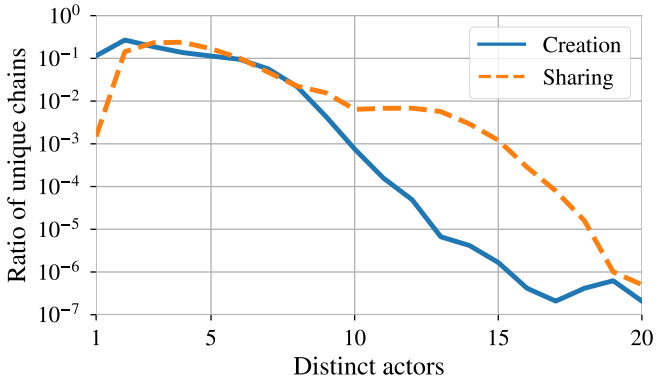

Fig. 3: Number of distinct actors in chains (log scale).

a similar way to third-party ones. However, ghosted cookies might allow for a more stealthy way of tracking: even privacy-conscious users who block third-party cookies could still be tracked through first-party ghosted cookies.

Table I on the preceding page enumerates the top cookie ghostwriters, ranked by the number of websites in which they set ghosted cookies. It is interesting to notice that several organizations mostly or almost exclusively set cookies as ghostwriters, making them harder to detect with previously existing approaches. We speculate that the actors listed in the table want to ensure the tracking can be done even when third-party tracking is blocked.

Of the 138M cookie creation events, we identified 1.43M unique cookie names. Of those cookie names, 793,085 (55%) were associated with an identifier value. Interestingly, for about half of those cookie names we observed both identifiable and non-identifiable values. This suggests that the same name is often used by different websites to store different types of information. We suspect that this might also be done on purpose by some parties to hide identifiers. For example, we discovered many cookies with seemingly innocuous names (such as `lang`) that at times contained identifiable strings like `th5bhut9onq8tnsskqm86` or `6057eb8a86fd83fd24956771`.

During our crawling, we identified 6.24M unique scripts (by content) in the trees. Those scripts were imported from 1.87M unique file names. After excluding scripts that did not perform any cookie-related operation, we were left with 365,277 scripts. Examples in this category include `sharethis.js`, which is used for social media sharing buttons and plugins in websites [37], `ga.js`, which is related to the Google Marketing Platform analytics [38]), and `piwik.js`, which corresponds to the JavaScript tracking client from Matomo [39]. On the other hand, scripts that were not involved in cookie operations were mostly related to well-known frameworks and libraries such as `jquery.js` [40], `bootstrap.min.js` [41] and `modernizr.js` [42].

TABLE II: Inter-actor relationships uncovered with our approach vs. those observable with previous approaches, i.e., solely based on the analysis of sources and destinations of explicit HTTP cookie syncing.

Distinct elements	State-of-the-art	Our approach	Increase
cookie actors	48,285	171,140	+254%
(a_1, a_2) relationships	151,257	809,179	+435%
(s, a_1, a_2) triples	2,025,936	6,490,377	+220%

B. Cookie Tree Statistics

This paper was mainly motivated by the assumption that a deeper investigation into the cookie creation and sharing events would enable the identification of considerably more actors that are part of the cookie ecosystem. This would eventually provide us with a much more comprehensive and accurate picture of the whole ecosystem. To validate this assumption we compare our results with state-of-the-art knowledge on the number of actors, the total number of (inter-actor) relationships, and the number of relationships per website that could be identified through existing methods which only consider explicit HTTP cookie syncing.

Table II shows the number of (s, a_1, a_2) triples – which represent a direct relationship (connection) between an organization a_1 and another a_2 when visiting a website s. This relationship happens when a_1 is linked to a_2 in a cookie tree. The results reported in the table clearly indicates that our approach uncovers significantly more organizations (almost three times more) and relationships (more than four times). This clearly show the added value of performing a deeper investigation of the cookie creation and sharing events. In particular, the analysis of cookie trees enabled us to identify as many as 170K organizations that are involved in the creation and sharing of cookies. Figure 3 instead shows the distribution of the number of actors that are involved in each cookie chain. The number of intermediaries appears to be higher than one would imagine: 62% of all unique creation chains and 86% of all unique sharing chains happen because of at least one intermediary node which is uncovered thanks to our methodology. Furthermore, for 43% of the websites that set identifier cookies we observe that at least three different organizations are involved. In summary, we can observe and study a significantly larger number of actors and relationships between them compared to those that could be obtained using previous methodologies, discovering a large and intricate system of relations that have not been analyzed before.

V. Roles and Behaviors

Here, we analyze the cookie-related behavior we observed in our crawl, discussing cookie sharing, collisions, inclusions of risky domains, and difference between website categories.

A. Cookie Sharing

Cookie sharing is a very common practice, and in our experiments we observed 8.97M cookie sharing events over 387K websites. Worryingly, 505,926 of these events, found

TABLE III: Breakdown, by roles, of cookie sharing events.

Receiver type	*Own* sender			*In-chain* sender			*Off-chain* sender		
	Total websites	Organizations		Total websites	Organizations		Total websites	Organizations	
		Sender	Receiver		Sender	Receiver		Sender	Receiver
Own receiver	351,861	26,898		50,301	14,039	3,335	40,470	2,032	12,410
In-chain (other / self)	60,789	2,592	12,073	26,195 / 67,664	2,444	1,830 / 22,803	20,912	1,538	7,382
Off-chain (other / self)	101,600	14,583	3,910	46,921	14,283	2,548	75,578 / 135,040	3,967	3,240 / 4,940

in 43,748 websites, happened without encryption. Even more worryingly, 37.71% of these cookies that were shared over HTTP were in fact initially created through HTTPS.

Table III provides global statistics about cookie sharing, detailing the different sender/receiver roles. The table shows that the most common phenomenon we observe is *own sender to own receiver*: in 352K websites (48% of those that create identifier cookies), an actor creates a ghosted first-party cookie and then sends it to himself. This practice is likely used by companies to sync ghosted first-party cookies with third-party cookies created when accessing other domains. As a result, the company can keep on tracking users even after they delete or refuse third-party cookies. For example, a user may visit three pages in which a tracking company creates first-party cookies (e.g., *id1*, *id2* and *id3*), which are then send back to the tracker himself. If these IDs are synced with a third-party cookie (e.g., *idGlobal*), even if *idGlobal* is deleted from the browser, the tracking organization still has the information that *id1*, *id2* and *id3* are the same user.

> *Example: Online jewelry store*
> A popular jewelry website offers potential buyers a customer chat. To do so, it loads an external script provided by an online support management company. This script creates a ghosted first-party identifier in the context of the website, and sends it to its own server to match the local id with the general user cookie.

Another common behavior (102K websites, 14% those with identifier cookies) is *own sender to off-chain receiver*. In this case, cookie creators share identifiers with actors that do not appear in the cookie creation trace. This could be due to data sharing agreements, where the intentional action of the cookie creators shows that they willingly share their identifiers.

> *Example: Online gaming store*
> An online gaming store includes a platform that enables advertisers to publish videos. The script of this company creates ghosted first-party identifier in the main page. Then, this same script send the created identifier to another company that offer web user tracking services, with which they probably have a data sharing agreement.

The most suspicious of all roles is *off-chain self receiver*. In this case, an organization that is not involved in the creation of a cookie, retrieves it from the browser and sends it to itself. We observe this behavior on 135K websites, in a significant portion of the websites (18%). This could be explained by the possible off-chain actors that take advantage of scripts that have access to the same cookie storage to be able to read cookies created by other creators (first-party or, more frequently, ghosted ones). This behavior might be consistent with some sort of "cookie stealing", but also with consensual data sharing after a private agreement between actors.

> *Example: Porn video chat website*
> A highly accessed porn video chat website includes a third-party widget to indicate how many users are connected at that moment. This widget adds two additional scripts to the equation; the first is from a bot detection company, and the second is from a marketing company. The marketing company creates an identifier cookie, bot detection script sends it back to himself along with other identifier cookies it did not create, maybe in an attempt to detect suspicious patterns in the cookies. It does so, however, without any direct interaction with the marketing domain.

A couple of more roles may raise the reader's attention, because they involve types of interaction whose purpose may not be intuitively obvious. A first one is *in-chain self receiver* (67K websites, 9% of those with identifiers), which can happen because an actor A may include code from another actor B that will send an identifier back to A.

> *Example: News website*
> A news website includes content from an advertisement company (`x.com`) that includes some recommended/sponsored "articles" at the bottom of the page. This company inserts content from a marketing company in the page (`match.y.com`), which creates an identifier cookie. After the cookie is created, this last company includes a script from the original `x.com` company, which sends the cookie to itself, through the `match.x.com` domain.

A second case worth clarifying is *off-chain sender to own receiver* (40K websites, 5% of those having identifiers), where an entity not involved in the creation of a cookie sends this cookie to the creator itself. This may happen because a third party sends additional data correlated to a user identifier.

TABLE IV: Breakdown by role of dangerous actors included in chains.

Risk	Creator	Sender			Receiver (other / self)			Intermediary	
		Own	*In-chain*	*Off-chain*	*Own*	*In-chain*	*Off-chain*	Creation	Sharing
5–6: Moderate (caution)	60,381	5,194	4,128	4,395	1,527 / 4,290	1,690 / 2,815	5,466 / 2,958	60,729	23,513
7–8: Moderately high (shady)	31,023	6,232	2,947	5,055	1,370 / 5,062	1,254 / 1,931	3,585 / 3,684	26,155	14,245
9–10: High (malicious)	2,943	726	489	259	132 / 560	158 / 311	436 / 185	3,186	2,425

TABLE V: Cookie collisions.

Second writer	Total websites	# creator organizations	
		Original	Overwriters
In original creation chain	1,758	377	880
Not in creation original chain	6,664	3,660	478

Example: Online sports store
A sports retailer creates a first-party identifier cookie with the initial `Set-Cookie` header. The page also includes a script from a e-commerce company that offers behavioral analysis of potential customers; this script collects data about the users' visit, and then sends this information back to the main page (including the mentioned identifier), to an internal path with the `visit-event` name, suggesting that this is used to link the information back to the user.

B. Cookie Collisions

We detected 184,377 cookie collision events in our analysis. In 71% of the cases, an identifier value in the cookie was changed to a different identifier; in the remaining cases, the identifier was changed to a non-identifier value (e.g., an empty string) or vice versa. In most cases (90.16%) the cookies were overwritten by scripts rather than via the `Set-Cookie` headers in requests.

Example: Privacy service
A GDPR compliance control company includes a script that intentionally overwrites cookies for 66 different actors with an empty string, most likely to avoid cases of unrequested tracking. However, due to a race condition, this approach does not always work. In fact, in some cases the value of the cookies was shared though a HTTP request before it was successfully deleted.

While not extremely common, cookie collisions are still an important phenomenon to analyze, due to the potential impact on the proper functioning of the websites and the associated services. In Table V, we provide some overall statistics on the cookie collisions we observed in our dataset. In most cases, these collisions were caused by actors outside of the cookie creation chain (observed in 6.67K websites versus 1.76K where the overwriter was *in* the creation chain). It is interesting to see that only 478 organizations are responsible for overwriting cookies of 3,660 other parties. Collisions may be dangerous for original creators, as a bad overwritten value could break the logic of some software, and may even create security problems if such data is not properly sanitized (e.g., when processed by a server-side application).

Example: Software bug
A company offers a service to dynamically adapt the content of a website based on the referrer value (e.g., to account for users coming from a search engine vs. those that clicked on an advertisement banner). This is implemented by a script that reads all first-party cookies and caches them, performing an encoding phase while processing the data. All cookies are then rewritten in their encoded form, probably due to a bug in the implementation, thus erroneously overwriting all of them.

Example: Copy-pasted code
A US-based web analytics company open sourced all its code for transparency reasons. After manual analysis, it appears that some web analytics companies based in Asia reused the same code in their product without changing the name of the created cookies. Since many websites include code from multiple web analytics providers, all these companies (the original one and the "copycat" ones) end up creating colliding cookies. As a result, likely without realizing it, they are tampering with each other's tracking cookies.

C. Risk

We now consider how often risky domains are involved in the creation or sharing of identifier cookies, as certain reports have shown that multiple targeted attack campaigns employ tracking in their reconnaissance phase to identify target users [43, 44].

To achieve this, we leverage a commercial engine that associates a risk level between 1 (completely safe) and 10 (certainly malicious) to each domain [45, 46]. In 24% of websites there is at least one actor involved in a creation/sharing chain with a risk higher than the risk of the website itself. Worryingly, 9% of websites with domains considered safe (risk level at most 4) include, in a cookie chain, a dangerous domain (risk level 5 or higher), and 4% include likely malicious domains (risk level at least 7).

As shown in Table IV, 60K (8%) websites have at least one cookie creator that is considered moderately dangerous (risk level 5 or 6) while *suspicious* actors create cookies for 31K (4%) websites. Finally, *malicious* actors set cookies in a non-negligible set of 2,943 websites. The other roles (cookie

sender, receiver and intermediary) are less common, but still present at all the risk levels.

> *Example: Cycling news website*
> An eastern European cycling news website includes a script named `tooltip.js` loaded from a local resource, using the `src` field of the `script` tag. This script searches, by `id` value, for a `div` element created by a large Asian online market place which includes an `iframe` for its domain. The `tooltip.js` script changes the inner HTML of that `div`, creating a new iframe pointing to a malicious domain (risk level 10), which then creates a third-party identifier cookie in the user's browser. Since `tooltip.js` is hosted on the website's server, our explanation is that the server was likely infected due to a vulnerability in the server, whose software was not properly updated and was likely affected by a vulnerability (`Apache/2.2.15 (CentOS) modi_rpaf/0.6 PHP/5.4.30`). This particular malicious domain has been related to phishing and ransomware attacks; it is thus possible that we observed the first stage of an attack that lead to a malware infection or phishing attempt using the identifier cookie created here as a trigger (similar to previously reported attacks [43, 44]). At the time of writing the victim website did not seem infected anymore.

D. Website Categories

We now consider how actors in the cookie ecosystem vary among different website categories (using website-to-category data from a commercial engine [45, 46]). For each category, we compute a "bag of words" that includes the number of times links between actors appear in cookie chains. We then use the generalized Jaccard similarity between these bags of words to compute the cross-similarity matrix between website categories. This matrix essentially encodes how similar the cookie ecosystems of categories are.

We tried to use this similarity matrix to perform clustering and discover groups of categories that have similar cookie ecosystems. However, we obtained results showing a single, large cluster centered on the most important actors (see Section VI-A); however, some categories are closer to the center of this cluster, and some are instead quite dissimilar from all other categories. Hence, we computed a centrality score, which is a category's average similarity with each other category, and we ranked all categories for which we crawled at least 1,000 websites according to this centrality score. In Table VI we show the top and bottom categories in this ranking: we can see that the most central categories are those whose business models are heavily reliant on "mainstream" advertising. Conversely, we explain the different cookie ecosystems in the bottom categories with different business models: for example, malware and scam are funded through other means; placeholders websites are essentially unused; charitable organizations are funded through donations; office/business applications and financial services are often paid services that are not based on advertising. Our results for the pornography category confirm those of Wondracek et al. [47], who find that this ecosystem is well separated from that of other web industries, with separate advertisement and different business models.

TABLE VI: Top and bottom website categories, ranked by centrality scores representing their similarity against all other categories (see definition in Section V-D).

Rank	Score	Website category	# websites
1	0.340	Entertainment	36,311
2	0.338	Education	39,864
3	0.336	Reference	15,480
4	0.335	Society/Daily Living	12,184
5	0.333	Art/Culture	3,412
46	0.201	Scam/Questionable/Illegal	1,321
47	0.177	Malicious Sources/Malnets	8,989
48	0.163	Placeholders	6,575
49	0.144	Charitable Organizations	5,049
50	0.135	Office/Business Applications	1,215
51	0.128	Financial Services	19,595
52	0.087	Pornography	18,932

VI. Actors and Relationships

In this section, we take a closer look at the cookie ecosystem to identify the main actors that are responsible for the majority of cookie creation and sharing events.

A. Top Actors

Table VII on the following page presents the list of top companies involved in cookie actions. The list is sorted by the number of websites they appear in and provides information about how often each company plays one of the roles defined in Section II. We associate the organizations to their country through manual investigation based on publicly available data. Among the top 50 organizations, 39 are based in the US and only 11 are based in other countries. Due to space constraints, in Table VII, we include the global top 10 and all other actors that are not based in the US. It is worth pointing out that these 50 organizations alone have a majoritarian weight in the cookie ecosystem, being involved in 58% of the (s, a_1, a_2) triples shown in Table II on page 6.

It is interesting to observe how different actors play different roles in the cookie ecosystem. For instance, while most actors are, among other roles, cookie creators (either as ghostwriters or as third parties), others like RapLeaf, Casale and WPP, rarely create cookies but often act as senders, receivers or intermediaries. With a few exceptions (e.g., Baidu and Mail.ru), most actors frequently behave as intermediaries in both cookie creation and sharing, suggesting a complex panorama of connections between actors. This can be due to a variety of reasons, including *ad exchange* services that facilitate buying and selling advertisements from multiple networks, or the reciprocity patterns discussed in Section VI-B. These indicate

TABLE VII: Top actors, ranked by the number of websites in which they appear; we include the top 10 actors, and actors outside of the US that are in the overall top 50. We report the percentage of websites in which they adopt each role.

#	Actor	CC	Websites	Creator	Sender			Receiver (from other / self)						Intermediary	
					own	*in-chain*	*off-chain*	*own*		*in-chain*		*off-chain*		Creation	Sharing
1	Google	US	415,545	93%	–	2%	14%	– /	–	2% /	1%	7% /	10%	29%	32%
2	Facebook	US	171,699	96%	96%	–	20%	– /	96%	– /	–	3% /	20%	–	20%
3	AddThis	US	62,671	84%	80%	–	13%	4% /	80%	– /	–	14% /	9	1%	4%
4	Yandex	RU	49,036	93%	92%	1%	22%	7% /	92%	– /	–	9% /	22%	2%	92%
5	Adobe	US	45,835	71%	27%	2%	34%	2% /	26%	37% /	1%	36% /	33%	61%	81%
6	RapLeaf	US	42,698	13%	–	1%	75%	– /	–	– /	1%	57% /	73%	10%	68%
7	AppNexus	US	39,991	77%	47%	47%	38%	– /	44%	9% /	46%	46% /	36%	73%	87%
8	Trade Desk	US	39,000	76%	14%	12%	4%	8% /	14%	4% /	11%	51% /	4%	75%	93%
9	Yahoo!	US	38,472	48%	20%	19%	55%	9% /	19%	10% /	19%	71% /	53%	25%	62%
10	Quantcast	US	36,662	68%	62%	–	11%	– /	62%	– /	–	50% /	11%	8%	26%
21	Adform	DK	25,627	24%	1%	–	26%	– /	1%	– /	–	44% /	23%	15%	87%
30	Baidu	CN	22,771	99%	12%	5%	5%	1% /	12%	– /	4%	– /	5%	1%	11%
32	WPP	GB	21,943	–	–	–	28%	– /	–	– /	–	86% /	23%	2%	84%
34	Bidr	AU	20,918	81%	–	1%	6%	74% /	–	63% /	–	3% /	–	90%	94%
35	Horyzon	FR	20,872	89%	1%	11%	9%	1% /	–	73% /	11%	20% /	8%	93%	84%
39	Casale	CA	19,595	32%	19%	–	60%	– /	–	– /	–	57% /	45%	16%	43%
41	Eyereturn	CA	18,616	90%	–	–	–	– /	–	– /	–	17% /	–	–	–
47	Criteo	FR	16,308	79%	64%	–	14%	14% /	64%	– /	–	15% /	12%	5%	39%
48	Mail.ru	RU	16,120	92%	7%	–	36%	2% /	7%	– /	–	10% /	36%	1%	7%
50	Avocet	GB	15,677	51%	3%	–	31%	– /	3%	9% /	–	20% /	31%	50%	86%

that our approach can capture very different *modus operandi* among cookie actors, as we elaborate with some examples.

Example: Facebook, AddThis & Yandex
These actors in the majority of cases share their own cookies with themselves. For example, Facebook creates ghosted cookies with a script served by the *connect.facebook.net* domain and then sends them to himself by including them in the URL of 1x1 pixel GIF image retrieved from *facebook.com*. This behavior implies that whoever receives these cookies (in this case Facebook) will be able to continue tracking even when the third-party cookies for *facebook.com* are deleted. As we discussed earlier, this is a very common phenomenon among other actors as well. Interestingly, Google—one of the main ghostwriters—does not appear to use this technique, and it does not share its ghosted cookies with anyone. This means that it will still be able to track users, but will not able to correlate their visits to different websites if they do not keep third-party cookies.

Example: Eyereturn & Google
Some actors have a relatively constant behavior. For instance, Eyeturn creates cookies but does not participate in other roles in the ecosystem, even if in some cases it receives identifiers created by other actors (i.e., it is an off-chain receiver). On the other hand, Google is rarely a sender or a receiver, but it is quite frequently involved as an intermediary, due to the Google tag manager. Through manual analysis, we found that most cases where Google behaved as a sender were due to website owners embedding copies of Google code rather than downloading them from the official URL.

Example: Rapleaf & Casale
Rapleaf is a marketing company that offers various services, such as identity matching, website visitor identification, and email intelligence. It has an interesting behavioral pattern regarding cookies, being a frequent off-chain cookie sender. Their terms of service [48] help us understand this situation, stating that they *"may share the Information (including PII) with customers, marketing services and platforms, and service providers and vendors that we retain"*, and that they *"may share some or all of the Information with affiliated or subsidiary companies."*. Indeed, we see that they generally use the *idsync.rlcdn.com* domain, whose name suggests that the off-chain cookie sending we observe may happen between willing partners who want to sync their identifiers. We observe a similar pattern with Casale, which is an online ad exchange company.

Example: WPP
Another interesting case is WPP, a company that offers multiple services related to advertising, content, media investment, public relations and public affairs. They are off-chain receivers from multiple different actors in many occasions, indicating a large number of deals with other organizations in order to track web users.

B. Linkage Graph

From the cookie trees we produced, we create a global graph where the nodes are the organizations and the edges connect those organizations that are connected through a cookie chain. The weight of the edges are calculated with the number of websites in which those organizations are connected by an edge in one or more cookie trees. The top nodes, and

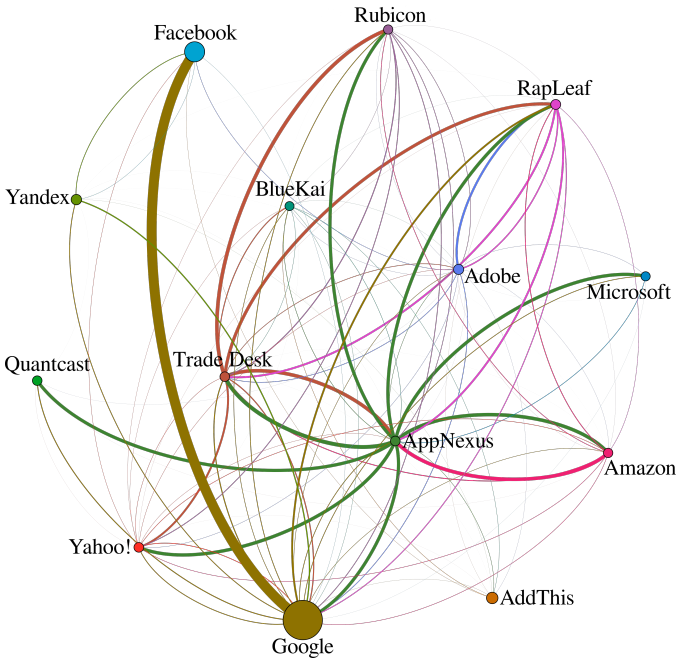

Fig. 4: Linkage subgraph that includes organization appearing in at least 30,000 crawled websites. Edge thickness is proportional to weight: faint lines indicate small weights. Edge direction can be read from curvature (edges follow a clockwise direction) and color (it is the one of the originating edge).

connections between them, are depicted in Figure 4. The curvature of edges (they follow a clockwise direction) can be used to infer their directionality; the single most important edge (thickness is proportional to weight) is the one that connects Google to Facebook, which is principally related to the creation of cookies, as the tag management solution dynamically includes the code of the social network, that ends up creating identifiers on the browser.

Figure 4 also shows that AppNexus has many outgoing links towards other top advertisers, which have roughly the same weight (see also details on those links in Table VIII). Basically, we see that AppNexus works as a *dispatcher*, acting as the hub that connects to many advertisers. By manually analyzing the data we collected we observed the same pattern for other players, such as PubMatic and WPP.

The full linkage graph has 803,410 nodes and 1,874,575 edges; out of all nodes, 171,140 have a non-zero indegree, meaning that they do play a role as cookie actors, while the remaining 632,270 nodes are websites that are not included in any cookie trace of other websites. Figure 5 shows the degree distribution of the graph, displaying the complementary CDF, which is suitable to discern to what extent graphs follow a power-law degree distribution, which would correspond to a straight line in this plot [49]. We see that the degree distribution is well approximated by a power law, with a truncation around a 10^5 degree. Since most nodes have 0 indegree, the indegree and outdegree distributions differ for low values. This truncated power-law degree distribution is typical of scale-free networks, indicating that in this case a few important actors have disproportionate importance.

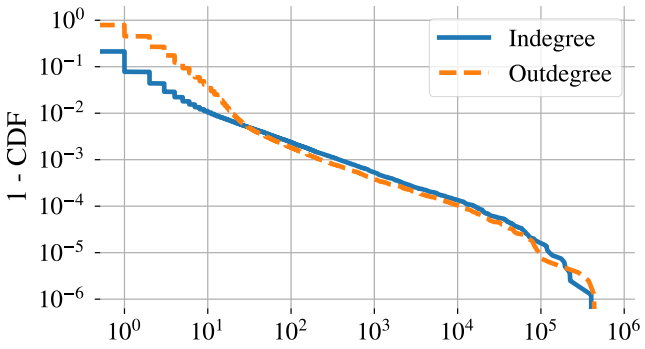

Fig. 5: Degree distribution in the full linkage graph. The complementary CDF on the y axis represents the fraction of nodes having degree of no more than x.

TABLE VIII: The top links, by weight, of the linkage graph. The *SoA* column indicates the ratio of links detectable with previous approaches (see Table II). For simplicity, we do not include collisions and cookie header-based sharing, as they affect a minority of these links.

Origin	Destination	Websites per trace type			
		All	SoA	Creation	Sharing
Google	Facebook	57,261	–	56,886	8,654
AppNexus	Trade Desk	22,235	100%	3,037	22,208
Trade Desk	Rhythm	21,103	95%	15,643	18,462
PubMatic	Drawbridge	21,075	11%	21,068	16,046
Drawbridge	PubMatic	20,847	95%	20,841	15,947
Adobe	Drawbridge	20,743	75%	20,167	15,296
Drawbridge	Adobe	20,671	75%	20,066	15,095
Trade Desk	Rubicon	20,520	99%	3,966	20,493
Amazon	AppNexus	20,252	–	20,162	2,238
Trade Desk	AppNexus	20,079	85%	3,960	20,076
Yahoo!	AOL	19,942	9%	5,131	18,654
AppNexus	Amazon	19,881	100%	18,836	19,124
AppNexus	PubMatic	19,773	98%	15,293	18,861
PubMatic	SiteScout	19,604	2%	19,598	409
Drawbridge	Trade Desk	19,513	–	19,014	15,868
Trade Desk	Drawbridge	19,510	94%	19,015	15,866
Trade Desk	PubMatic	19,002	99%	198	17,114
AppNexus	Yahoo!	18,879	100%	13	18,879
Google	WPP	18,665	–	175	18,508
AppNexus	Microsoft	18,630	99%	18,615	18,326

The top edges of the linkage graph are shown in Table VIII, which better captures the specialization of those links: for example, the link from Google to Facebook is dominated by traces involving cookie creation, while the one from AppNexus to TradeDesk is mostly related to cookie sharing. The *SoA* column refers to the fraction of websites in which the link between actors could be observed using state-of-the-art approaches (see Table II on page 6): we can see that important links between cookie actors (e.g., Google-Facebook, Amazon-AppNexus) would have been invisible, or in other cases grossly underestimated, by using previous approaches based only on third-party sharing. As one may expect, collisions are very rare in the interaction between large players.

Another interesting aspect is the *reciprocity* of some links,

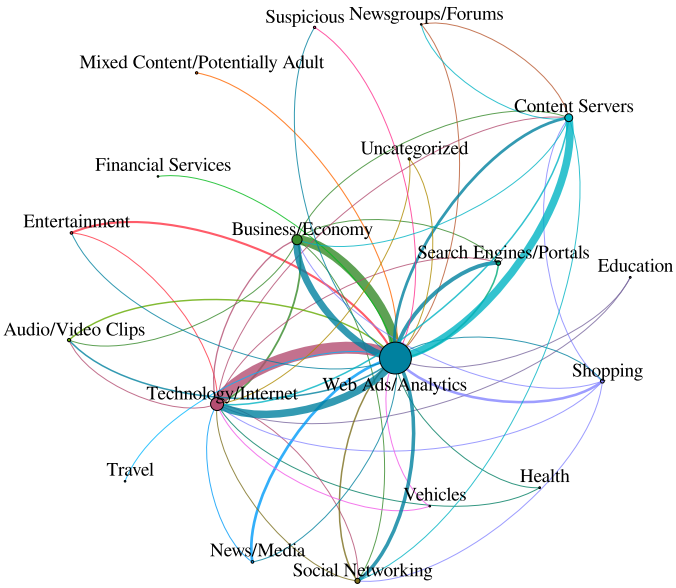

Fig. 6: Subgraph of the per-category linkage graph including nodes with at least 1% degree and the largest edges between them. Edge weight and direction can be read according to the instructions in the caption of Figure 4.

meaning that an edge from A to B is reciprocated with another edge from B to A having about the same weight, or in other words that A includes code from B roughly as often as B includes code from A. For example, in Table VIII, the links from AppNexus to Amazon and TradeDesk have returning links of roughly the same weight, and the same can be said about links from Drawbridge to Trade Desk and PubMatic. We cannot be sure of the reason behind this phenomenon, but it is hard to believe that it is just a coincidence. A possible explanation is that these actors exchange code inclusions to mutually increase each other's coverage. Besides observing these links between big players, we evaluated reciprocity on the whole linkage network, obtaining a value of 0.41 according to the r coefficient of Squartini et al. [50], meaning that 41% of the weight in the whole actor linkage subgraph is reciprocated. This confirms the fact that a reciprocation pattern hence appears to exist and be noticeable in the whole network.

C. Organizations not in trackerlists

To verify how many of the cookie actors that we observed were known to the tracking protection community, we used `git` commit history to retrieve all the domains that were *ever* included in the Disconnect (used by Mozilla Firefox) [51], AdGuard [52] and EasyList [53] trackerlists, for a total of 41,027 filtered domains. The domains that are most active in tracking do appear in trackerlists, but we encountered many small organizations (166,365 in total) involved in the cookie ecosystem that were not included in any trackerlist. The number of websites in which these actors appear (average 4.10) is distributed very unevenly, matching once again a truncated power law similar to the one reported in Figure 5. 45.7K organizations appear in at least two websites, 4.8K in at least 10, and only four are included in 10K websites or more. Overall, existing trackerlists have a good coverage of the most important players (whitelisted or blacklisted), with the largest uncovered one being `zorosrv.com`, which appears in 16,000 websites and ranked 49$^{\text{th}}$ in the list shown in Table VII.

TABLE IX: Highest degree nodes in the per-category graph.

Category	Degree	Indegree	Outdegree
Web Ads/Analytics	31%	41%	22%
Technology/Internet	12%	12%	12%
Business/Economy	10%	9%	10%
Content Servers	7%	6%	9%
Social Networking	4%	7%	2%
Search Engines/Portals	4%	6%	2%
Shopping	3%	2%	5%
Audio/Video Clips	3%	4%	2%
News/Media	2%	1%	3%
Entertainment	2%	1%	3%
Mixed Content	2%	2%	1%
Suspicious	2%	2%	2%
Uncategorized	2%	1%	2%
Education	1%	–	2%
Health	1%	–	2%

D. Linkage between website categories

We now discuss the linkage between cookie actor categories. Cookie actors can and do have domains relevant to multiple categories (e.g., Google/Alphabet owns the `google.com` search engine, the `youtube.com` video website and the `doubleclick.net` advertiser), so we now consider the intermediate chains between domains described in Section III-B, and build a new linkage graph, following the procedure of Section VI-B, based on these chains. As in Section V-D, we use again the dataset provided by a commercial engine [45, 46]—this time to associate categories to each domain in the trace—and collapse this graph by merging nodes of the same categories, obtaining a linkage graph connecting 85 categories. The linkage between the top categories is displayed in Figure 6.

The top categories by degree (defined as sum of in- and out-degree) of the graph are displayed, with their ratio of the total degree, in Table IX. It is clear that most cookie traces are concentrated in just a few categories and, perhaps unsurprisingly, the *ads/analytics* category is by far the most important in this graph. Other important categories are *technology/services*, including domains used for Javascript libraries and web services and *business/economy*, including businesses often related to marketing, e.g., the `gumgum.com` applied computer vision company.

Further down the list, the *mixed content* category includes several providers of user-generated content that may be marked for parental control. The *suspicious* and *uncategorized* categories include a large number of small actors, which are described in more details in Section V-C.

It is interesting to consider the mismatch between indegree and outdegree of nodes: those with a larger indegree, such as *ads/analytics*, *social networking*, and *search engines*, are the ones that perform most tracking; on the other hand, categories

such as *shopping*, *news/media* and *entertainment* include more pages are likely to bring information to trackers when visited.

VII. Discussion

Our study shows that it is common to have complex chains of intermediaries in web pages, and that cookies are written by, and shared among, many different actors. The complexity of interactions that we observed in our experiments have multiple implications in the current web panorama.

From the point of view of website owners and developers, the sheer number of actors involved in each page makes it difficult to guarantee that user privacy is respected. Recent data protection legislations require strict control on user identifiable information, but due to this intricate structure, ensuring compliance with current regulations may be very difficult without advanced methods to check the information flow. Moreover, we have seen that it is unfortunately not rare to encounter content from potentially dangerous sources associated to cookie creation and sharing. On the one hand, we can envision that enforcement of current and upcoming privacy jurisprudence will keep limiting the amount of unwanted data sharing between trackers; on the other hand, we believe that finer-grained approaches such as ours will help in detecting actors who try to skirt the rules.

From a preventive point of view, there are various aspects to take into account. Existing blocking solutions use pre-calculated lists in order to block tracking, and they successfully include most of the larger actors of the cookie ecosystem. However, our study shows that a large number of smaller players are not present in these lists, and that these actors often play important roles in creation and sharing chains. Cookie trees and cookie flows offer a systematic way to better understand each actor, together with its role, in the trackerlists. Basic cookie analysis may erroneously underestimate or misclassify many actors, such as those that create ghosted cookies or work as dispatchers for other players in the ecosystem. By allowing these tools to identify and stop the tracking process from its initial stag, our approach will largely improve their efficiency while avoiding other anti-blocking strategies [54, 55] that can lead to other privacy violations.

VIII. Related Work

Web tracking has been studied from multiple perspectives. Krishnamurthy and Wills [56] were among the first to analyze cookies. Later, Roesner et al. [57] proposed a classification of web tracking behaviors. Recent research [5, 58] performed large-scale analyses of the ecosystem, and found that most of the highly-accessed websites performed tracking.

Web cookies have also been analyzed from different perspectives. Sivakorn et al. [59] performed an extensive study on the privacy threats that users may suffer when an attacker steals their HTTP cookies. They checked a wide range of highly accessed services and found that the corresponding problems are not limited to a specific group of website types. Franken et al. [6] evaluated the effect of third-party requests and cookie policies in various browsers and extensions. They used a framework which automatically checked policy enforcement using a comprehensive set of test cases, and identified multiple flaws in existing policy implementations. Sanchez-Rola et al. [60] presented a timing side-channel attack against server-side request processing schemes that uses cookies to detect the specific state of users in third-party websites.

Cookie sharing was analyzed by multiple studies. Acar et al. [61] were among the first to perform a comprehensive analysis of the topic. They present a technique to analyze identifier flows using the *strace* debugging tool to analyze traffic, and a set of criteria to distinguish and extract pseudonymous identifiers. Their measurements show that more than 11% of the users browser history can be linked this way in the wild. Ghosh et al. [62] analyzed the concept from a business perspective, and have shown that when trackers have similar sizes they will find that cookie sharing is financially beneficial for all; conversely, when they are not homogeneous, cookie sharing will increase the revenue of some at the expense of others. Brookman et al. [9] reviewed a small set of popular websites to check for types of cookie sharing that facilitate cross-device tracking. They demonstrated that many of these websites were sharing extensive data with third-party services, and that user lack information to understand what is actually happening, as the privacy policies do not clarify the specific purpose of sharing. A recent paper by Urban et al. [8] studied cookie sharing before and after the European General Data Protection Regulation (GDPR) came into effect. They detected a statistically significant impact on the number of connections between third-parties (around 40% less). The most recent paper in the topic, authored by Papadopoulos et al. [7], analyzed cookie sharing in a year-long traffic log from 850 real mobile users. They implemented a method to detect these events in real time, and measured the privacy loss on the user side. Using this method, they were able to find that 97% of the web users are sometime exposed to cookie sharing, most of them within the first week.

In summary, a large corpus of work studied the security and privacy implications of cookie sharing and web tracking; however, the set of interdependences between the players in the cookie ecosystem due to the complexities of modern-day webpages—the main topic of this research—still remained largely unexplored. Or main goal has been to fill this gap.

IX. Conclusions

We took an in-depth analysis of the cookie web tracking panorama, and we have shown that the intricate mechanics of interactions between several different authors on the same web page play a key role. Through a large-scale and fine-grained crawl, we obtained a large dataset that allowed us to shine light on large parts of the cookie ecosystem that previously received little attention. We reveal the existence of a convoluted network of connections between players, where each of them follows multiple different roles. For instance, thanks to our methodology, we are able to detect the presence of dispatcher organizations that facilitate cross-site tracking.

References

[1] The Guardian, "A year-long investigation into facebook, data, and influencing elections in the digital age," https://www.theguardian.com/news/series/cambridge-analytica-files, 2018.

[2] European Union, "Directive 2009/136/EC of the European Parliament and of the Council of 25 November 2009," *Official Journal of the European Union*, 2009.

[3] ——, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016," *Official Journal of the European Union*, 2016.

[4] I. Lapowsky, "California unanimously passes historic privacy bill," https://www.wired.com/story/california-unanimously-passes-historic-privacy-bill/, 2018.

[5] S. Englehardt and A. Narayanan, "Online tracking: A 1-million-site measurement and analysis," in *ACM Conference on Computer and Communications Security (CCS)*, 2016.

[6] G. Franken, T. Van Goethem, and W. Joosen, "Who left open the cookie jar? a comprehensive evaluation of third-party cookie policies," in *USENIX Security Symposium (Sec)*, 2018.

[7] P. Papadopoulos, N. Kourtellis, and E. Markatos, "Cookie synchronization: Everything you always wanted to know but were afraid to ask," in *The Web Conference (WWW)*, 2019.

[8] T. Urban, D. Tatang, M. Degeling, T. Holz, and N. Pohlmann, "Measuring the impact of the gdpr on data sharing in ad networks," in *ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, 2020.

[9] J. Brookman, P. Rouge, A. Alva, and C. Yeung, "Cross-device tracking: Measurement and disclosures," *Privacy Enhancing Technologies (PETs)*, no. 2, 2017.

[10] N. Nikiforakis, L. Invernizzi, A. Kapravelos, S. Van Acker, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "You are what you include: large-scale evaluation of remote javascript inclusions," in *ACM Conference on Computer and Communications Security (CCS)*, 2012.

[11] D. Kumar, Z. Ma, Z. Durumeric, A. Mirian, J. Mason, J. A. Halderman, and M. Bailey, "Security challenges in an increasingly tangled web," in *The Web Conference (WWW)*, 2017.

[12] T. Lauinger, A. Chaabane, S. Arshad, W. Robertson, C. Wilson, and E. Kirda, "Thou shalt not depend on me: Analysing the use of outdated javascript libraries on the web," in *Network and Distributed System Security Symposium (NDSS)*, 2017.

[13] S. Arshad, A. Kharraz, and W. Robertson, "Include me out: In-browser detection of malicious third-party content inclusions," in *International Conference on Financial Cryptography and Data Security (FC)*, 2016.

[14] T. Urban, M. Degeling, T. Holz, and N. Pohlmann, "Beyond the Front Page: Measuring Third Party Dynamics in the Field," in *The Web Conference (WWW)*, 2020.

[15] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," in *Network and Distributed System Security Symposium (NDSS)*, 2019.

[16] Alexa, "SEO and Competitive Analysis Software." https://www.alexa.com/, 2019.

[17] Cisco Umbrella, "Umbrella Popularity List." https://umbrella-static.s3-us-west-1.amazonaws.com/index.html, 2019.

[18] Majestic, "The Majestic Million." https://majestic.com/reports/majestic-million, 2019.

[19] Quantcast, "Audience Insights That Help You Tell Better Stories." https://www.quantcast.com/top-sites/, 2019.

[20] I. Sanchez-Rola, D. Balzarotti, C. Kruegel, G. Vigna, and I. Santos, "Dirty Clicks: a Study of the Usability and Security Implications of Click-related Behaviors on the Web," in *The Web Conference (WWW)*, 2020.

[21] E. Sangaline, "Making Chrome Headless Undetectable," https://intoli.com/blog/making-chrome-headless-undetectable/, 2017.

[22] ——, "It is Not Possible to Detect and Block Chrome Headless," https://intoli.com/blog/not-possible-to-block-chrome-headless/, 2018.

[23] Dymo, "Missing Accept_languages in Request for Headless Mode," https://bugs.chromium.org/p/chromium/issues/detail?id=775911, 2017.

[24] ChromeDevTools, "DevTools Protocol API," https://github.com/ChromeDevTools/debugger-protocol-viewer, 2019.

[25] O. Starov and N. Nikiforakis, "Extended tracking powers: Measuring the privacy diffusion enabled by browser extensions," in *The Web Conference (WWW)*, 2017.

[26] I. Sanchez-Rola, M. Dell'Amico, P. Kotzias, D. Balzarotti, L. Bilge, P.-A. Vervier, and I. Santos, "Can I opt out yet?: GDPR and the global illusion of cookie control," in *ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, 2019.

[27] R. Cointepas, "CNAME cloaking, the dangerous disguise of third-party trackers," https://medium.com/nextdns/cname-cloaking-the-dangerous-disguise-of-third-party-trackers-195205dc522a, 2019.

[28] NextDNS, "NextDNS CNAME cloaking blocklist," https://github.com/nextdns/cname-cloakin-blocklist, 2019.

[29] J. Kurkowski, "tldExtract: Accurately separate the TLD from the registered domain and subdomains of a URL, using the Public Suffix List," https://github.com/john-kurkowski/tldextract, 2019.

[30] Disconnect, "Make the web faster, more private, and more secure," https://github.com/disconnectme, 2019.

[31] Cliqz GmbH, "WhoTracks.me: Bringing Transparency to Online Tracking," https://github.com/cliqz-oss/whotracks.me, 2019.

[32] T. Libert, "Webxray, a tool for analyzing third-party content on webpages and identifying the companies which collect user data." https://github.com/timlib/webXray, 2019.
[33] P. Kotzias, L. Bilge, P.-A. Vervier, and J. Caballero, "Mind your own business: A longitudinal study of threats and vulnerabilities in enterprises." in *Network and Distributed System Security Symposium (NDSS)*, 2019.
[34] Merit Network Inc., "IRR - Internet Routing Registry," http://www.irr.net/.
[35] "WhoisXML API," https://www.whoisxmlapi.com/.
[36] N. Raghavan, R. Albert, and S. Kumara, "Near Linear Time Algorithm to Detect Community Structures in Large-Scale Networks," *Physical review. E, Statistical, nonlinear, and soft matter physics*, 10 2007.
[37] ShareThis, "Social Share Buttons and Plugins for Websites." https://sharethis.com/, 2019.
[38] Google Marketing Platform, "A unified advertising and analytics platform." https://marketingplatform.google.com, 2019.
[39] Matomo, "Secure Open Web Analytics Platform." https://matomo.org/, 2019.
[40] jQuery, "Fast, small, and feature-rich JavaScript library." https://jquery.com/, 2019.
[41] Bootstrap, "Open source toolkit for developing with HTML, CSS, and JS." https://getbootstrap.com/, 2019.
[42] Modernizr, "Feature detection library for HTML5/CSS3." https://modernizr.com/, 2019.
[43] Threat Intelligence, FireEye, "Pinpointing Targets: Exploiting Web Analytics to Ensnare Victims," https://www2.fireeye.com/rs/848-DID-242/images/rpt-witchcoven.pdf, 2015.
[44] Security Response, Symantec, "The Waterbug attack group," http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/waterbug-attack-group.pdf, 2015.
[45] Symantec, "The Need for Threat Risk Levels in Secure Web Gateways," https://www.symantec.com/content/dam/symantec/docs/white-papers/need-for-threat-tisk-Levels-in-secure-web-gateways-en.pdf, 2017.
[46] ——, "WebPulse," https://www.symantec.com/content/dam/symantec/docs/white-papers/webpulse-en.pdf, 2017.
[47] G. Wondracek, T. Holz, C. Platzer, E. Kirda, and C. Kruegel, "Is the internet for porn? an insight into the online adult industry." in *Workshop on the Economics of Information Security (WEIS)*, 2010.
[48] Towerdata, "Privacy policy," https://www.towerdata.com/privacy-policy, 2019.
[49] A. Clauset, C. R. Shalizi, and M. E. Newman, "Power-law distributions in empirical data," *SIAM review*, vol. 51, no. 4, pp. 661–703, 2009.
[50] T. Squartini, F. Picciolo, F. Ruzzenenti, and D. Garlaschelli, "Reciprocity of weighted networks," *Scientific Reports*, vol. 3, no. 1, pp. 1–9, Sep 2013.
[51] Mozilla Foundation, "Security/Tracking protection," https://wiki.mozilla.org/Security/Tracking_protection, 2019.
[52] AdGuard, "Adguard content blocking filters," https://github.com/AdguardTeam, 2019.
[53] EasyList, "Easylist filter subscription," https://github.com/easylist, 2019.
[54] U. Iqbal, Z. Shafiq, and Z. Qian, "The ad wars: retrospective measurement and analysis of anti-adblock filter lists," in *Internet Measurement Conference (IMC)*, 2017.
[55] S. Zhu, X. Hu, Z. Qian, Z. Shafiq, and H. Yin, "Measuring and disrupting anti-adblockers using differential execution analysis," in *Network and Distributed System Security Symposium (NDSS)*, 2018.
[56] B. Krishnamurthy and C. Wills, "Privacy diffusion on the web: a longitudinal perspective," in *The Web Conference (WWW)*, 2009.
[57] F. Roesner, T. Kohno, and D. Wetherall, "Detecting and defending against third-party tracking on the web," in *Networked Systems Design and Implementation (NSDI)*, 2012.
[58] I. Sanchez-Rola and I. Santos, "Knockin' on trackers' door: Large-scale automatic analysis of web tracking," in *Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*, 2018.
[59] S. Sivakorn, I. Polakis, and A. D. Keromytis, "The cracked cookie jar: Http cookie hijacking and the exposure of private information," in *IEEE Symposium on Security and Privacy (SP)*, 2016.
[60] I. Sanchez-Rola, D. Balzarotti, and I. Santos, "BakingTimer: Privacy Analysis of Server-Side Request Processing Time," in *Annual Computer Security Applications Conference (ACSAC)*, 2019.
[61] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The web never forgets: Persistent tracking mechanisms in the wild," in *ACM Conference on Computer and Communications Security (CCS)*. ACM, 2014.
[62] A. Ghosh, M. Mahdian, R. P. McAfee, and S. Vassilvitskii, "To match or not to match: Economics of cookie matching in online advertising," *ACM Transactions on Economics and Computation (TEAC)*, vol. 3, no. 2, pp. 1–18, 2015.