Politecnico di Torino

Politecnico di Torino

Master's Degree Thesis in Cybersecurity Engineering

A.a. 2024/2025

Graduation Session December, 2025

A Dual-Vantage Measurement of the Privacy Sandbox in the Real World

Advisors:

Prof. Marco MELLIA
Prof. Martino TREVISAN
Prof. Nikhil JHA
Dr. Alberto VERNA

Candidate:
Rachid Youssef GRIB

Abstract

Recently, the web industry has opened the possibility to phase out third-party cookies. Following the lead of the Safari and Firefox browsers, which block them by default, in 2019 Google announced the Privacy Sandbox initiative, a set of APIs designed to replace the functionality of third-party cookies in the Chrome browser. These APIs are designed to support advertising functionality, fight fraud and strengthen cross-site privacy boundaries, improving user privacy. This thesis investigates the Privacy Sandbox, with a particular focus on the Protected Audience API, which enables interest-based advertising and ad retargeting. The research objective is to evaluate the adoption and implementation in practice. To this end, two complementary methods are applied. First, a Chromium-based web crawler analyzes the top 10,000 websites according to the Tranco list. The crawler collects JavaScript calls and Chrome DevTools protocol events related to the Privacy Sandbox APIs, both after accepting and after denying cookie consent banners. Second, a custom Chrome extension intercepts and analyzes JavaScript calls to the Privacy Sandbox APIs during real browsing sessions. This method enables the collection of authentic user data, providing insights into actual usage patterns and behaviors. The collected data highlights cases where the Privacy Sandbox APIs are not configured as intended by third parties that implement them, suggesting potential areas where privacy concerns could emerge from their usage. While the Privacy Sandbox aims to improve user privacy and provide an alternative to third-party cookies, the findings indicate that there are still challenges to be addressed for its broader and more effective adoption.

Acknowledgements

Un sincero ringraziamanto ai miei relatori Prof. Marco Mellia , Prof. Martino Trevisan, Prof. Nikhil Jha e ad Dr. Alberto Verna, per il supporto e i loro consigli, fondamentali durante lo sviluppo di questo lavoro di tesi e per avermi trasmesso la loro passione per la ricerca.
Un ringraziamento speciale va ai miei genitori, a mio papà Jama e mia mamma Ijjou, per il loro amore incodizionato e continuo supporto, e a mio fratello Samir per essere sempre stato sempre al mio fianco incoraggiandomi a dare il meglio di me.

Table of Contents

List of Figures

Acronyms

API
Application Programming Interface

DSP
Demand-Side Platform

SSP
Supply-Side Platform

FedCM
Federated Credential Management

SLD
Second-Level Domain

GDPR
General Data Protection Regulation

CCPA
California Consumer Privacy Act

CHIPS
Cookies Having Independent Partitioned State

CMP
Consent Management Platform

CP
Calling Party

GTM

Google Tag Manager

Chapter 1

Introduction

In recent years, privacy concerns related to web browsing have received increasing attention from users, browsers and regulators. Web browsing activities rely on small pieces of data known as cookies, which websites store on users' devices to remember their preferences, login information, but over time they have also been used for tracking users across different websites for advertising and analytics purposes. While cookies can enhance user experience, third party cookies that are cookies set by domains other than the one the user is visiting, have raised significant privacy concerns due to the fact that allow advertisers and other third parties to track users' browsing behavior. These practices have raised regulatory concernes especially with the introduction of regulations such as the General Data Protection Regulation (GDPR) in Europe and the California Consumer Privacy Act (CCPA) in the United States. In response to these concerns, web browsers like Safari and Firefox have implemented measures to block or limit the use of third-party cookies. To address these challenges, Google introduced the Privacy Sandbox initiative: a set of APIs and technologies designed to enhance security and privacy on the web with the goal of replacing third-party cookies in Chrome. These APIs aim to limit cross-site tracking, combat fraud, and provide privacy-preserving alternatives to traditional tracking mechanisms. Google Chrome plans to phase out third-party cookies in favor of the Privacy Sandbox. However, in the October 17th, 2025 update, Google announced that some Privacy Sandbox APIs will be deprecated after evaluating community feedback [1]. This reflects the ongoing evolution of the initiative and the need to balance advertising functionality with improved user privacy.

1.1 Motivation

The adoption of these new technologies raises several questions about their use in real-world scenarios. It is therefore important to investigate how these APIs are used and whether third parties that implement them are legitimate and compliant with applicable regulations. This thesis investigates the adoption and implementation of the Privacy Sandbox APIs in practice, with a particular focus on the Protected Audience API, which enables interest-based advertising and ad retargeting.

1.2 Methodology

To evaluate the adoption and implementation of the Privacy Sandbox APIs, two complementary methods are applied. First, a Chromium-based web crawler analyzes the top 10,000 websites according to the Tranco list [2]. The crawler collects JavaScript calls and Chrome DevTools Protocol events related to the Privacy Sandbox APIs, both after accepting and after denying cookie consent banners. Second, a custom Chrome extension intercepts and analyzes JavaScript calls to the Privacy Sandbox APIs during real browsing sessions. This method enables the collection of authentic user data, providing insights into real usage patterns and behaviors. Whereas the first method enables collection of a large amount of data from a wide range of websites, the second method provides a more in-depth analysis because the user visits internal pages, clicks advertising links, and interacts with the website in a more natural way.

1.3 Thesis structure

The thesis is structured as follows:

- **Chapter 2** provides an overview of cookies, regulations, browser responses, and cookie blocking mechanisms. It also presents an overview of the Privacy Sandbox APIs, their functionalities, and their usage. Furthermore, it discusses who is allowed to use these APIs by explaining the concepts of allowed and attested third parties. Furthermore presents the Protected Audience API, describing the algorithm , the main calls, usage types and possible privacy concerns. Finally it presents related work on the Privacy Sandbox.
- **Chapter 3** describes the methodology used to investigate the adoption and implementation of the Privacy Sandbox APIs, including the web crawler and the Chrome extension.
- **Chapter 4** presents the results of the data collection and analysis of the crawler, highlighting key findings and possible privacy concerns.

- **Chapter 5** presents the results of the data collection and analysis of the Chrome extension, highlighting key findings.
- **Chapter 6** discusses the main findings , contributions and limitations of the thesis.

Chapter 2

Background and Privacy Sandbox

This section provides a comprehensive overview of cookies, regulatory responses, browser behavior, and introduces the Privacy Sandbox initiative.

2.1 Cookies and regulations

In the context of web browsing and online advertising, cookies are small pieces of text data that are stored on a user's device by a web browser at the request of a web server. Initially, cookies were designed to remember user preferences (e.g., which items a user wants), but today they are also used by advertisers to collect personal information and by web developers to track user behavior across multiple websites.

Cookies can be classified based on their origin:

- **First-party cookies**: These cookies are set by the website that the user is currently visiting. They are typically used to enhance user experience by remembering login details, language preferences, and other settings specific to that site. First-party cookies are generally considered less intrusive in terms of privacy since they are only accessible by the site that created them. Furthermore, in most cases, they are essential for the proper functioning of many websites.
- **Third-party cookies**: These cookies are set by domains other than the one the user is currently visiting. They are usually used by advertising companies to track users across multiple websites, building a profile of their browsing habits and interests. This cross-site tracking raises significant privacy concerns, as users may not be aware that their activities are monitored.

In practice, cookies are used for different purposes, such as:

- For example, if a user shops online for books, the bookstore can save information about which books are viewed and purchased. This enables the bookstore to provide suggestions for other books that might be of interest to that user.
- Cookies can save users' geographic locations, which allows websites to personalize information about weather, movie theater times, television listings, or local events.
- Cookies can also be used to remember users' language preferences, ensuring that they see content in their preferred language when they return to a website.
- A more controversial use of cookies involves tracking users across multiple websites. In this case, a third-party domain (for example, `advertiser.com`) places a cookie on the user's browser when they visit a website that includes its content or advertisements (e.g., `website1.com`). Later, when the same user visits another site that also integrates content from the same advertiser (e.g., `website2.com`), the advertiser can read the previously stored cookie. By correlating the information retrieved from multiple sites, the third party can reconstruct a profile of the user's browsing behavior, effectively creating a cross-site *UserHistory*.

Cookies can be used for legitimate purposes, but they can also enable cross-site tracking as illustrated in the example 2.1 which raises significant privacy concerns.

Furthermore, cookies can be classified based on their lifespan:

- **Session cookies**: These cookies are temporary and are deleted once the user closes their browser. They are primarily used for session management, such as keeping users logged in during a browsing session.
- **Persistent cookies**: These cookies remain on the user's device for a specified period or until they are manually deleted. They are used for remembering user preferences and login information across multiple sessions.

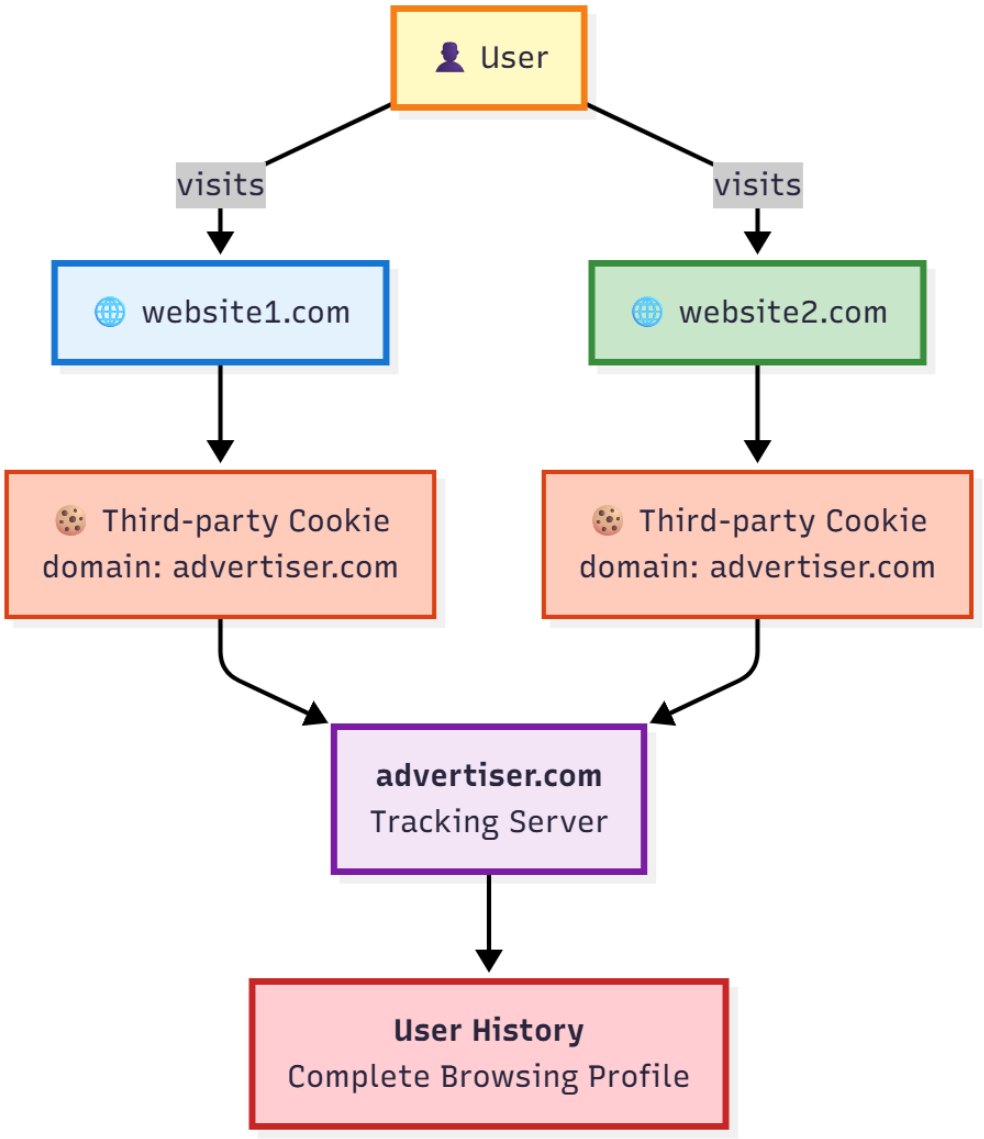

Figure 2.1: Example of cookie tracking.

2.1.1 Privacy concerns and regulations

As shown in Figure 2.1, cookies can be used to track users across multiple websites, enabling the creation of a detailed profile of a user's browsing history, interests, habits, and personal information. In response to these concerns, multiple regulations have been introduced. First, the General Data Protection Regulation (GDPR) [3], which came into effect on 25 May 2018 in the European Union, is one of the most comprehensive privacy and data-protection laws in the world. It imposes

obligations on organizations anywhere, so long as they target or collect data related to EU citizens. The GDPR defines several key legal terms; the most important are the following:

- **Personal data**: Any information that relates to an individual who can be directly or indirectly identified.
- **Data processing**: Any action performed on personal data.
- **Data controller**: The person or entity that is responsible for deciding why and how personal data will be processed.
- **Data subject**: The person whose data is being collected, stored, or processed.
- **Data processor**: A third party that processes data on behalf of the data controller. The GDPR has additional rules for data processors.

The GDPR requires organizations to obtain explicit consent from users before collecting or processing their personal data, including through the use of cookies. Another important regulation is the California Consumer Privacy Act (CCPA) [4], which gives California consumers more control over the personal information that businesses collect about them. The CCPA provides guidance on how to implement these rights. The law gives new privacy rights to California consumers, including:

- The right to know what personal information is being collected about them.
- The right to delete personal information collected from them (with some exceptions).
- The right to opt-out of the sale of their personal information.
- The right to non-discrimination for exercising their CCPA rights.

In 2020 the California Privacy Rights Act (CPRA) was approved; it amended the CCPA and went into effect on 1st January 2023, adding new privacy protections, including:

- The right to correct inaccurate personal information that a business has about them.
- The right to limit the use and disclosure of sensitive personal information.

Businesses that are subject to the CCPA have several responsibilities, including responding to consumer requests to exercise their rights.

2.1.2 Browser responses and cookie blocking

After the introduction of privacy laws such as the GDPR and CCPA, and the growing public awareness of online privacy issues, web browsers have increasingly implemented measures to limit or block third-party cookies by default.

Safari

Apple's Safari browser was among the first to take significant action against cross-site tracking. Since 2017, Safari has implemented *Intelligent Tracking Prevention* (ITP), a technology that uses on-device machine learning to detect and limit tracking behavior. ITP progressively enhanced restrictions on third-party cookies, culminating in their complete blocking by default starting from Safari 13.1 (released in March 2020) [5, 6].

Brave

Brave is a privacy-focused browser that blocks third-party cookies and trackers by default through its *Brave Shields* feature. Brave Shields also includes protections against fingerprinting, intrusive ads, and cross-site tracking, positioning Brave as one of the most privacy-aggressive browsers [7, 8].

Firefox

Firefox introduced *Enhanced Tracking Protection* (ETP) in 2019, which blocks third-party cookies from known tracking domains listed by Disconnect (Tracker protection list). Users can select between "Standard," "Strict," and "Custom" modes, providing varying levels of protection. ETP is enabled by default for all users since Firefox 69 [9, 10].

Google Chrome

Google Chrome, the most widely used browser, has historically allowed third-party cookies by default. However, Google announced plans to phase them out in favor of its *Privacy Sandbox* initiative, which seeks to balance user privacy with the needs of advertisers through new privacy-preserving APIs such as the *Topics API* and the *Protected Audience API* [11, 12].

2.2 The Google's Privacy Sandbox

The Privacy Sandbox is a set of APIs developed by Google to replace third-party cookies in Chrome. In this chapter we present a general overview of the Privacy

Sandbox APIs, their functionality and usage. Furthermore, we describe who can access these APIs by discussing the concepts of "allowed" and "attested" third parties.

2.2.1 Overview of the Privacy Sandbox

The Privacy Sandbox initiative is a initiative of Google to develop web technologies that enhance user privacy while enabling online advertising. The goal of the Privacy Sandbox is to phase out third-party cookies and other forms of cross-site tracking, which have raised significant privacy concerns among users and regulators.

- **Protected Audience API**[13]: enables privacy-preserving ad re-targeting without relying on cross-site tracking mechanisms like third-party cookies. The API allows advertisers to show ads to users based on websites prevously visited.
- **Attribution Reporting API**[14]: allows measuring the convesion by an ad-click or an ad-view to a subsequent action on the advertiser's website (e.g a purchase). Instead of relying on cross-site identifiers, the API generates reports that attribute conversions to ad interactions in a privacy-preserving manner.
- **Topics API**[15]: provides a privacy-preserving mechanism for interest-based advertising. The Topics API can then give API callers (e. g. ad tech platforms) access to a user's topics of interest, but without revealing additional information about the user's activity.
- **Shared Storage API**[16]: introduces a new form of shared browser storage that can be accessed across multiple domains. The shared storage can be accessed only from a javaScript secure environments (worklets).
- **Private Aggregation API**[17]: enables the generation of aggregate statistics based on client-side data, without revealing individual user information. It is designed to be used in combination with the Protected Audience and Shared Storage APIs to compute metrics such as ad reach and auction winning rate while preserving user anonymity.
- **Private State Tokens API**[18]: Private State Tokens allows websites to issue cryptographically signed tokens that can be redeemed later to prove certain actions were taken, such as completing a CAPTCHA or logging in. This helps prevent fraud and abuse while preserving user privacy.
- **CHIPS API**[19]: Cookies Having Independent Partitioned State (CHIPS) allows developers to opt a cookie into partitioned storage, with separate

cookie jars per top-level site, improving user privacy and security. Without partitioning, third-party cookies can enable services to track users and join their information from across many unrelated top-level sites.

- **Fenced Frames API**[20]: Fenced Frames API allows ads to be shown in isolated and secure areas of the page, without access to your personal data. The Fenced Frames API introduce the *<fencedframe>* HTML element, which prevents data sharing between the top-level site and the embedded content (e.g., an ad banner).

- **Federated Credential Management API**[21]: Federated Credential Management (FedCM) allows users to log into sites using their federated identity (e.g., "Sign in with...") without relying on third-party cookies or navigational redirects, normally used by protocols such as OAuth.

- **Related Website Sets API**[22]: Related Website Sets (RWS) provides a way for organizations to declare multiple domains as part of a single related group. This allows browsers to apply more permissive privacy boundaries within it.

2.2.2 Privacy Sandbox Allowed and Attested Third Parties

In order to use the Privacy Sandbox APIs, third parties need to be either allowed or attested. In fact this is a way to ensure that only trusted parties can access these APIs, helping to protect user privacy and prevent abuse.

Allowed third parties

Allowed third parties are authorized to use certain Privacy Sandbox APIs. This typically means that Google has included them in a predefined list that can access the APIs. Being "allowed" is a status granted directly by Google, and it usually applies to large, well-established services or to cases where the authorization is essential for compatibility or basic functionality.

Attested third parties

Attested third parties, on the other hand, are parties that have gone an attestation process to prove their compliance with specific technical and policy requirements. Instead of being directly whitelisted by Google, they must demonstrate that they meet the criteria established by the Privacy Sandbox framework. The attestation process can involve independent auditors that verify the party's identity, business

practices, and intended use of the APIs. Tipically attested third paries are companies that want to use the APIs for advertising purposes so they must prove to be compliant with privacy requirements of the Privacy Sandbox.

Allowed vs Attested

The key difference between allowed and attested parties lies in how trust is established:

- **Allowed**: trusted by default through a direct inclusion in an official list maintained by Google.
- **Attested**: trusted because of a formal verification process that confirms compliance with privacy-preserving requirements.

In summary, "allowed" third parties are trusted because of their predefined status, while "attested" third parties are trusted because they have proven to meet the standards required for responsible use of the Privacy Sandbox APIs.

2.3 Protected Audience API

The Protected Audience API [13] is a Privacy Sandbox technology that enables interest-based advertising using interest groups and on-device ad auctions to choose which ad to show based on interest groups from websites the user has previously visited. In this chapter we explore the key features and functionality of the Protected Audience API, focusing on the main API calls and their usage.

2.3.1 Motivation

Traditionally, online advertising relied on third-party cookies to track users across different websites and build detailed profiles of their interests and behaviors. Browsers need a way to enable ad platforms to select relevant ads to show to users so publishers can monetize their content while protecting user privacy. The Protected Audience API aims to provide a privacy-preserving alternative to third-party cookies for interest-based advertising. With these APIs, which enable on-device ad auctions and interest groups, advertisers can still deliver relevant ads without holding information about individual users.

2.3.2 Glossary of Protected Audience Terminology

The terminology used by Google in the Privacy Sandbox in particular in the context of the Protected Audience API can be confusing or difficult to understanding by

some readers, because some terms have a different meaning in real-world. This glossary clarifies the most important terms used throughout this thesis.

Interest Group A collection of users who share similar interests, defined by the advertiser or DSP. Users are added to interest groups based on their browsing behavior, such as visiting specific websites (eg. customers interested in buying bikes).

Bidding Logic JavaScript code executed in a worklet environment on the user's device to determine the bid that an advertiser is willing to pay for showing an ad.

Seller The entity running the on-device auction (the SSP or the publisher).

Buyer The entity participating in the auction with one or more interest groups (DSP or advertiser).

Auction Worklet The worklet environment in which the seller's or SSP's auction logic runs.

2.3.3 Protected Audience Lifecycle

The Protected Audience lifecycle consists of four main phases:

- Interest group joining: When a user visits a website, the DSP or the advertiser can register the user with an interest group.
- Ad auction run: When a user visits a page that contains an ad space, the publisher or the SSP runs an on-device ad auction to select the most relevant ad.
- Ad displaying: The winning ad from the auction is displayed to the user.
- Reporting: The publisher's or SSP's code can include a `reportResult()` function to report the outcome of the ad auction.

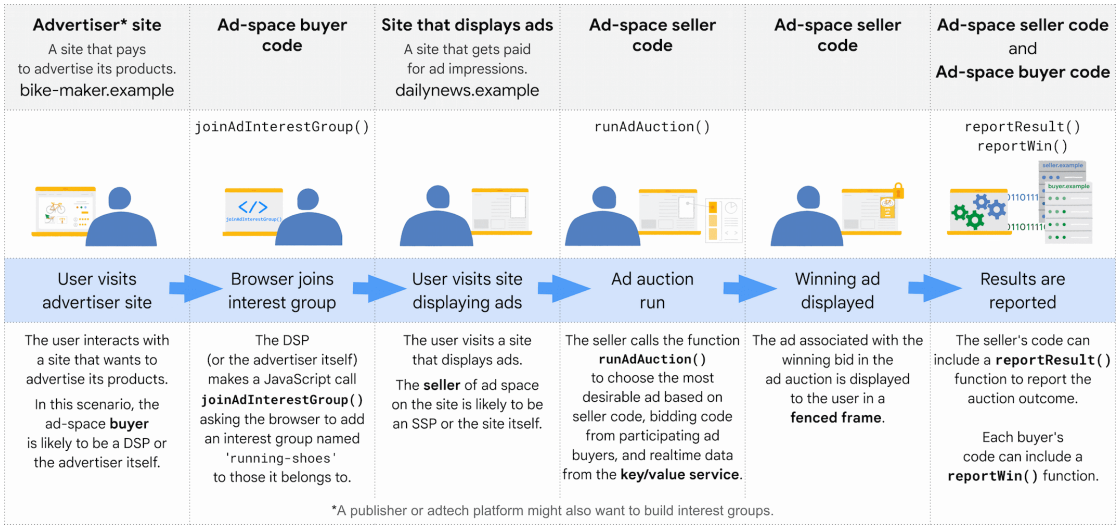

Figure 2.2: Protected Audience API Lifecycle [13].

2.3.4 Protected Audience algorithm

Let's dive deeper into how it works and how this technology preserves user privacy. The following picture shows that the code of the auction and the bidding logic are executed in isolated worklets, which are sandboxed environments that run in the user's browser. This means that even the bidding logic and the `scoreAd` function cannot access any user data directly; they can only access the data that is passed to them as arguments. This is a key feature of the Protected Audience API that helps preserve user privacy by limiting the amount of data that third parties can access. Overall, this mechanism permits running the auction and the bidding logic on the user's device without exposing user data to third parties, while still enabling interest-based advertising.

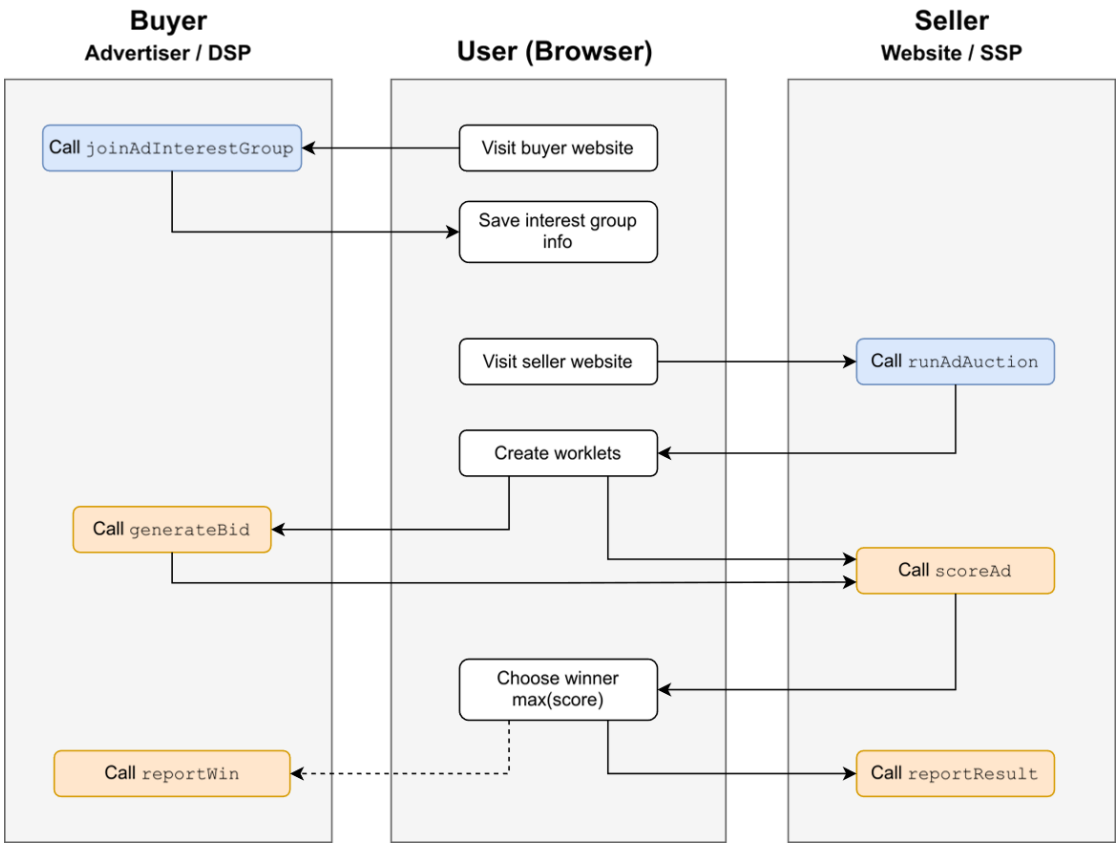

Figure 2.3: Protected Audience Worklets.

2.3.5 K-anonymity

The privacy concept known as "K-anonymity" seeks to make every individual in a dataset indistinguishable from at least $k-1$ other individuals. To put it another way, among a group of at least k users, the data of every particular user cannot be individually recognized. Because it lowers the possibility of connecting data to a particular individual, this feature is frequently regarded as a helpful privacy assurance. To protect user privacy the protected Audience API design includes k-anonymity checks with the aim of preventing tracking through user-specific interest groups. To be considered k-anonymous, an interest group must have at least 50 members over a period of 30 days , meaning that at least 50 different users must have joined the same interest group within the last 30 days. If an interest group does not meet this k-anonymity requirement, the browser will not allow it to participate in ad auctions. This k-anonymity check will be implemented by the browser in the future, when implemented it will help ensure that interest groups cannot be used to uniquely identify or track individual users, thereby enhancing

user privacy.

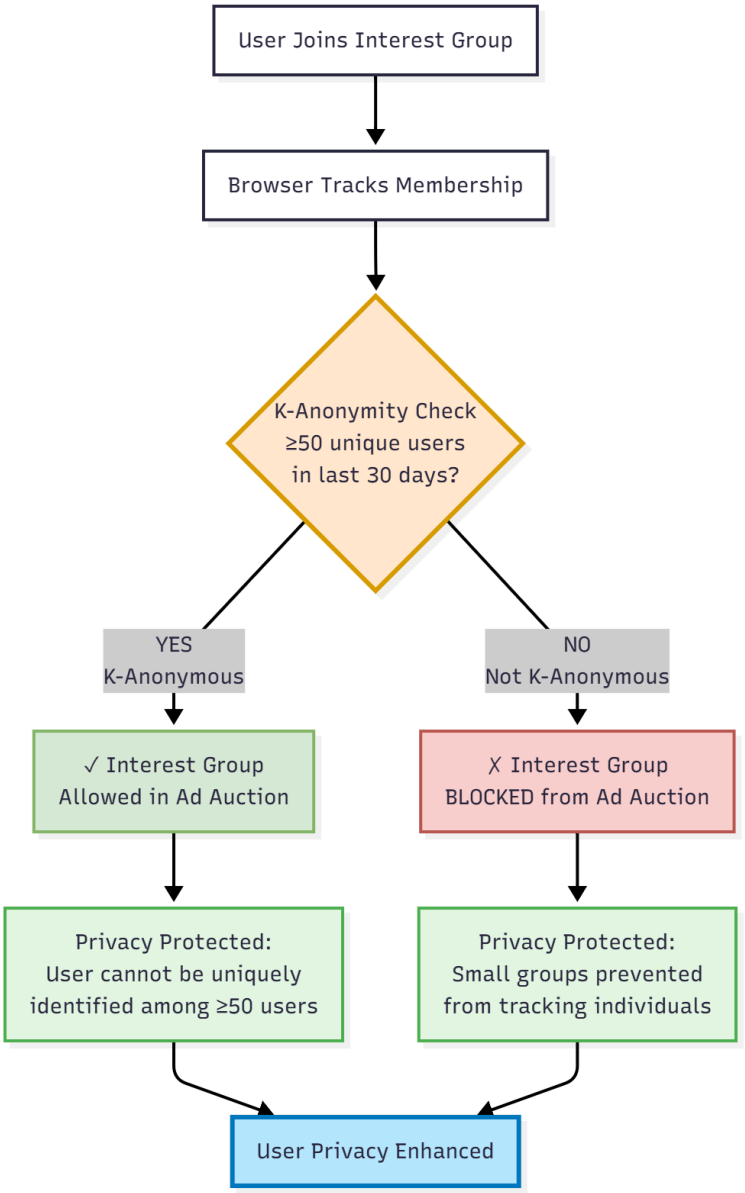

Figure 2.4: Protected Audience K-anonymity.

2.3.6 Main API calls

Joining an interest group

To join an interest group, the advertiser or the DSP can ask the user's browser to add the user to their interest group by calling the `joinAdInterestGroup()` JavaScript function. The function takes two arguments. The first is an object that contains fields such as the owner (the origin of the interest group) and the name of the interest group (these fields are mandatory). Other fields are optional. The most relevant optional fields are `biddingLogicUrl`, which is the URL for bidding logic run in a worklet during the auction phase; `updateUrl`, which is a URL that returns JSON to update interest group attributes; and `ads`, which are the ads that can be shown to the user. Although bidding logic and ads are optional fields, without them it is not possible to participate in the auction. The second argument is the duration of the interest group in seconds.

Here is an example from the Protected Audience API demo showing the structure of an interest group:

```
[{
    "owner": "https://protected-audience-demo-dsp.web.app",
    "name": "tv",
    "biddingLogicUrl": "https://protected-audience-demo-dsp.web.app/bid.js",
    "ads": [{
        "renderUrl": "https://protected-audience-demo-dsp.web.app/ads/default-ad.html",
        "metadata": {
            "adName": "default-ad"
        },
        "buyerReportingId": "brid123",
        "buyerAndSellerReportingId": "bsrid123",
        "selectableBuyerAndSellerReportingIds": [
            "sbsrid123",
            "sbsrid456",
            "sbsrid789"
        ]
    }]
}, 604800]
```

Listing 2.1: Interest Group Example from Protected Audience Demo

Running the ad auction

When a user visits a page that contains an ad space, the publisher or the SSP can run an on-device ad auction to select the most relevant ad to show to the user. This is done by calling the `runAdAuction()` JavaScript function, which takes an

object as its argument. The most relevant fields are `seller` (the origin of the seller running the auction), `decisionLogicUrl` (the URL for the auction worklet), and `interestGroupBuyers` (an array of origins of buyers that can participate in the auction). Here is an example from the Protected Audience API demo showing the structure of an ad auction:

```
[{
    "seller": "https://protected-audience-demo-ssp.web.app",
    "decisionLogicUrl": "https://protected-audience-demo-ssp.web.app/decision-logic.js",
    "interestGroupBuyers": ["https://protected-audience-demo-dsp.web.app"],
    "auctionSignals": {
        "isControversial": true
    },
    "sellerSignals": {
        "key": "value"
    },
    "sellerTimeout": 100,
    "perBuyerSignals": {
        "https://protected-audience-demo-dsp.web.app": {
            "windowInnerHeight": 551
        }
    },
    "perBuyerTimeouts": {
        "*": 50
    },
    "resolveToConfig": true
}]
```

Listing 2.2: Ad config Example from Protected Audience Demo

Usage types

The Protected Audience API can be called in two main contexts: iframes or fenced frames. Each is described in this section.

Iframes

The Protected Audience API can be invoked from a website's HTML using an iframe element. Here is an example of how an iframe can be used to load content from the Protected Audience demo:

```
<iframe id="dsp-pixel" src="https://protected-audience-demo-dsp.web.app">
    #document (https://protected-audience-demo-dsp.web.app/)
</iframe>
```

Listing 2.3: Iframe Example for Protected Audience Demo

Fenced frames

The Protected Audience API can also be invoked in fenced frames. Unlike iframes which normally permit some sharing of document context between the embedding page and the embedded content fenced frames prevent this sharing, so data from different websites cannot be exchanged. The Fenced Frames API is not yet mandatory for implementing the Protected Audience API, but according to the Privacy Sandbox documentation it will become mandatory in 2026. Here is an example of how a fenced frame can be used to load content from the Protected Audience demo:

```
1 <fencedframe id="protected-audience-ad" mode="opaque-ads"></
    fencedframe>
```

Listing 2.4: Fenced Frame Example for Protected Audience Demo

2.3.7 Discussion and possible privacy issues

There are some possible privacy issues resulting from the use of the Protected Audience API. As will be discussed in Chapter 4, the usage of the Protected Audience API has declined over time. However, should its adoption increase again in the future, it would be valuable to further investigate and highlight the privacy implications associated with this API. Exploring these concerns in greater depth could contribute to a more comprehensive understanding of its impact and inform the development of more robust privacy-preserving solutions.

Lets see the `scoreAd` for perform auction operations from lucead.com.

```
1     const allowed_buyers=['https://lucead.com','https://adapting-
      opossum-stunning.ngrok-free.app','https://exploreFledge.com'];
2
3 function scoreAd(adMetadata,bid,auctionConfig,
      trustedScoringSignals,browserSignals)
4 {
5   let desirability=bid;
6   const floor=auctionConfig?.auctionSignals?.lucead?.floor || .01;
7
8   //debugger;
9   if(browserSignals.bidCurrency!=='USD')
10  {
11    if(browserSignals.bidCurrency==='EUR')
12      desirability=bid/.92;
13    else
```

```
14        desirability=0;
15    }
16
17    if(bid<floor || !allowed_buyers?.includes(browserSignals.
       interestGroupOwner))
18      desirability=0;
19
20    return {
21      desirability,
22      allowComponentAuction:true,
23    };
24  }
```

Listing 2.5: Auction code snippet from lucead.com

As shown in Listing 2.5, the auction code from `lucead.com` includes a list of allowed buyers. Maybe this company is still testing the API and has not yet fully implemented it. That can be reasonable due to the fact that, the code is available and is not obfuscated, unlike the other code snippets from other big companies.

However, it is interesting to point out that, since it is not written in the documentation that there is a check by "someone" to verify if the auction code respects the rules and permits bidding to everyone without setting a whitelist, this can be a problem for the adoption of the API.

In fact, as we can see in Listing 2.5, the presence of a whitelist in the auction logic means that the seller (Lucead) can restrict which buyers are allowed to participate in the auction. This is against what the Protected Audience API aims to provide, and could lead to several issues:

- **Reduced market competition:** Sellers could favor their own DSP or selected partners, excluding other legitimate buyers and distorting the auction outcome.
- **Potential privacy risks:** If the same entity acts as both DSP and SSP, and controls the whitelist, it can correlate data from both sides of the transaction, increasing the risk of user profiling and cross-site tracking, as discussed in the previous section.
- **Lack of transparency:** Without external audits or enforcement mechanisms, there is no guarantee that sellers will implement the API in a fair and privacy-preserving way. This could destroy trust in the Privacy Sandbox ecosystem and slow down adoption by publishers and advertisers.

The documentation for the Protected Audience API does not currently specify any mandatory external checks or certification for auction code implementations. This means that, in practice, each seller can implement their own logic, including

whitelists, without oversight unless publishers or buyers themselves audit the code or the browser enforces stricter requirements.

This situation highlights a potential privacy issue in the the Protected Audience design, in fact without a proper verification mechanism, sellers could implement auction logic that undermines the privacy goals of the API.

2.4 Related Work

From its introduction, the Privacy Sandbox has attracted the interest of academic and industry researchers, who explored both its deployment and the privacy implications of their usage.

Most studies are focused on the Topics API. Several studies have questioned its ability to prevent re-identification. On the contrary, they have shown that users can be probabilistically linked across websites by collecting a large amount of topics over time, even when category selection and randomization mechanisms are applied [23, 24, 25, 26, 27]. Large-scale web measurements revealed that the Topics API's adoption, although constant at 30–40% of websites, remains fragmented and experimental, with evidence of A/B tests taking place [28, 29, 30]. The same studies highlight how many websites invoke the API inconsistently or even before consent is granted, often through unauthorized or misconfigured third-party scripts.

The Protected Audience has received a similar amount of analysis. Similar to the Topics API, this solution was also proven to be susceptible to re-identification attacks by taking advantage of covert channels to create unique identifiers and leak user information [31, 32, 33, 34]. Large-scale measurements studies show that the Protected Audience is much less adopted compared to the Topics API, accounting for ~1% of websites during early deployment [35] and reaching ~25% usage at its peak [30]. A deep dive on the players shows that the adoption remains heavily concentrated on Google's own infrastructure, as most auctions in particular up to 99% appear to be started by Google-owned domains [33].

Other Privacy Sandbox APIs have so far attracted more limited research attention. Recent works have demonstrated that the reports generated by the Attribution Reporting and Private Aggregation APIs satisfy differential privacy even in highly interactive settings that include proposed extensions like key discovery and requerying [36]. On the other hand, the Shared Storage API approximately used by 10% of websites was found to be vulnerable to multiple covert channels [37]. Similarly, the foundational assumption underlying the Related Website Sets proposal was challenged by findings indicating that users frequently fail to accurately determine if two sites are affiliated, suggesting potential privacy harms by excessively broad declared relations [38]. Lastly, implementation focused research into Cookies Having Independent Partitioned State (CHIPS) revealed that adoption has been slow,

especially among third-party tracking domains, which still predominantly rely on non-partitioned cookies [39].

So far, no significant works focusing on the Fenced Frames and FedCM APIs have been produced.

Overall, this researchs highlights both the potential and the challenges of the Privacy Sandbox initiative. While it offers an innovative approach to sobstitute third-party cookies, significant concerns remain regarding its effectiveness in preserving user privacy and its adoption across the web ecosystem.

Chapter 3

Dual Vantage Measurement Methodology

In this section we deep dive into the dual vantage measurement methodology that we use to measure the adoption and to evaluate the implementation of the Privacy Sandbox APIs in practice. This dual-vantage methodology allow us to have two different perspectives on the adoption of the Privacy Sandbox APIs, the first one is the web crawler vantage, that allow us to have a large scale measurement of the adoption of the Privacy Sandbox APIs across a wide range of websites, while the second one is the Chrome Extension vantage, that allow us to have a more realistic measurement of the adoption of the Privacy Sandbox APIs during real browsing sessions.

3.1 Web Crawler

To evaluate the adoption of the Privacy Sandbox APIs, we developed a web crawler based on a Chromium browser. The crawler is built on a previous research project [40]. It visits URLs and click the "accept" button on privacy-policy banners. We adapted the crawler to detect the use of the Privacy Sandbox APIs on websites after accepting and after denying cookie consent. To do so, we first developed a wordlist of keywords for detecting deny actions on cookie-consent banners.

3.1.1 Strategy to deny the cookie consent

To detect the deny button we used the Priv-Accept framework, which has built-in screenshot functionality. We crawled the top 100 websites from the UK, France, Germany, Spain, and Italy according to SimilarWeb [41] and manually reviewed the screenshots to add keywords to the wordlist following the same strategy used

in Priv-Accept [40] to detect the accept button. The final wordlist contains 166 keywords in five languages (English, French, German, Spanish, and Italian). We also handled cookie-consent banners that do not have a deny button but provide an "options" button that opens a popup where the user can disable non-essential cookies. In this case, we modified the original Priv-Accept code to allow the crawler to perform two clicks: first on the "options" button and then on the "deny" button. The final options wordlist created using the same strategy contains 143 keywords in five languages (English, French, German, Spanish, and Italian). Finally, we implemented functionality to click buttons even when they are inside iframes by iterating over all iframes on the page and searching for the buttons inside them.

The following code shows the implementation of the enhanced banner search functionality that handles both regular elements and iframe-embedded banners:

```
def search_iframe_banner(driver, wordlist_file=deny_words if deny else accept_words,
                         screenshot_name="clicked_element"):
    banner_data = click_banner(driver, wordlist_file, screenshot_name=screenshot_name)
    if banner_data.get("clicked_element"):
        driver.switch_to.default_content()
        return banner_data
    iframes = driver.find_elements(By.TAG_NAME, "iframe")
    for iframe in iframes:
        try:
            logger.info(f"Searching for banner in iframe: {iframe.id}")
            driver.switch_to.frame(iframe)
            internal_banner_data = search_iframe_banner(driver, wordlist_file,
            screenshot_name=screenshot_name)
            if internal_banner_data:
                return internal_banner_data
        except:
            logger.info("Exception while searching banner in iframe: {}".format(iframe.id))
        finally:
            driver.switch_to.default_content()
    return None


def double_click_banner(driver):
    # First click: option_words
    logger.info("Searching for Options button")
    first_result = search_iframe_banner(driver, wordlist_file=option_words,
```

```
27                                          screenshot_name="
    option_button")
28      if first_result is None or not first_result.get("
    clicked_element"):
29          return first_result, None
30      time.sleep(timeout)
31      logger.info("Searching for {} button".format("Deny" if deny
    else "Accept"))
32      second_result = search_iframe_banner(driver,
33                                           screenshot_name="
    deny_button" if deny else "accept_button")
34      return first_result, second_result
```

Listing 3.1: Enhanced Banner Search with Iframe Support

3.1.2 Detecting Privacy Sandbox API usage

Subsequently, we implemented functionality to detect Privacy Sandbox API usage by listening to Chrome DevTools Protocol (CDP) events and intercepting JavaScript calls to the Privacy Sandbox APIs.

This is achieved through a modular architecture composed of a central *APICallInterceptor* and several specialized *ApiCallCollector* components. The interceptor establishes a CDP connection with the browser and injects a monitoring script into each page and frame to capture function calls (e.g., *document.browsingTopics()*, *navigator.joinAdInterestGroup()*, *navigator.runAdAuction()*). In parallel, intercept also CDP events (e.g., *Storage.interestGroupAccessed*).

Each Privacy Sandbox API is handled by a dedicated collector, which implements the logic to record JavaScript calls, CDP events, and, where applicable, data extracted from Chrome's internal databases. For example, the *Protected Audience API* collector registers calls to *navigator.joinAdInterestGroup()* and *navigator.runAdAuction()*, while also listening for storage events associated with ad auctions and interest-group. Similarly, collectors have been designed for the Topics API, the Attribution Reporting API, the Fenced Frames API, and other Privacy Sandbox APIs. Subsequently, the *Priv-Accept* extract two sets of domains for each crawl session.

The first set corresponds to the **attested domains**, which are obtained by querying the `.well-known/` endpoints of the contacted sites and parsing their published *Privacy Sandbox attestations*. This information describes the APIs that a given site claims to use, together with its attestation metadata.

The second set corresponds to the **allowed domains**, which are derived from Chrome's local attestations database (`privacy-sandbox-attestations.dat`). This file, stored within the browser profile, contains the origins that Chrome internally recognizes as eligible to access specific Privacy Sandbox APIs.

Finally, the resulting datasets including verified domains, API-usage metrics, and topic-analysis outputs are periodically transferred to the **Smart Data Cluster**, where longitudinal analyses are carried out using `Jupyter Notebooks` to monitor the evolution of Privacy Sandbox adoption over time.

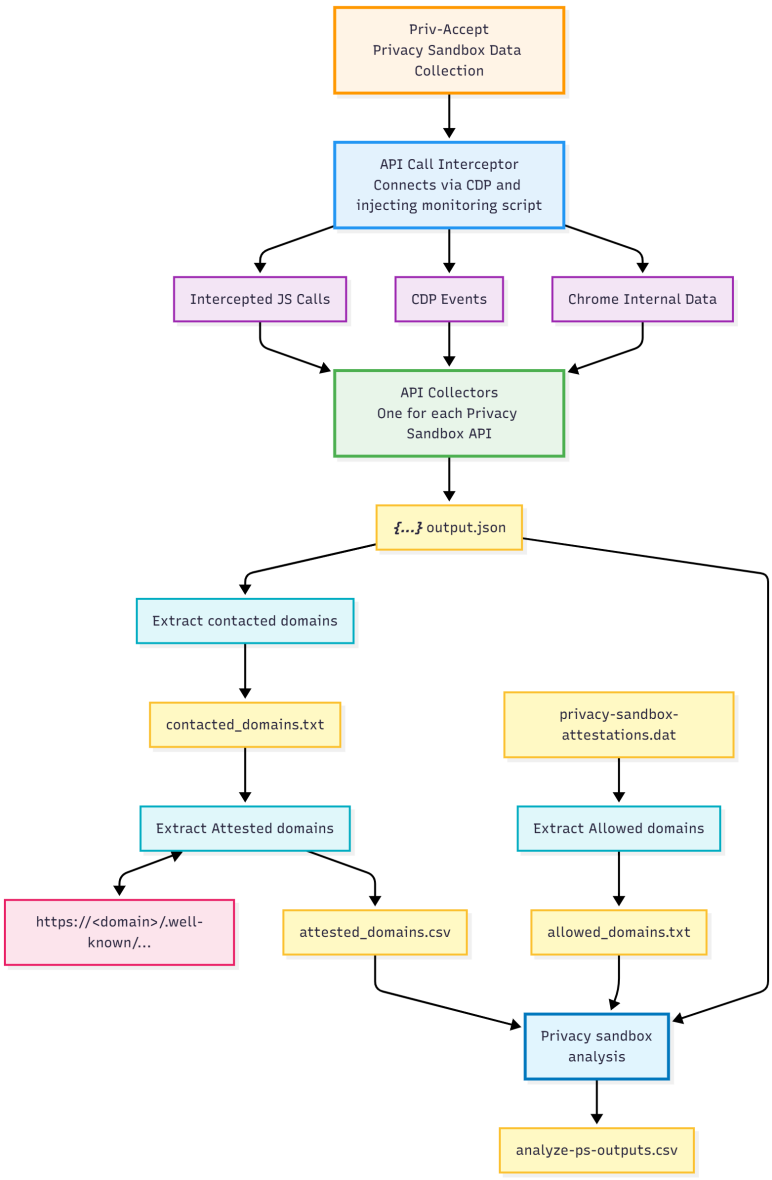

Figure 3.1: Architecture of the Web Crawler.
Legend: Orange: Priv-accept tasks; Blue: Processing steps; Purple: JS calls, CDP events,internal data monitoring; Green: Privacy Sandbox API collectors; Yellow: Data outputs (json, txt, dat, csv); Red: Attestation domain requests.

3.2 Chrome Extension

In addition to the web crawler, which collects large amounts of data from a wide range of websites, we developed a Chrome extension that intercepts and analyzes JavaScript calls to the Privacy Sandbox APIs during real browsing sessions.

3.2.1 Strategy and implementation

The extension is built using Manifest V3. The extension has several components that work together to monitor and log the usage of the Privacy Sandbox APIs. First, the `availabilityCheck.js` script checks if the Privacy Sandbox APIs are available in Chrome. If they are not available, the extension sends a notification via the `chrome.notifications` API to inform the user. Second, `injected.js` is injected into every mainframe and iframe by the content script `injector.js` at document start. The main purpose of `injected.js` is to override Privacy Sandbox API functions to log their usage and parameters. Furthermore, it also records all visited URLs and cookies that are set during the browsing session. This information is sent to the content script `injector.js`, which sends it to the background script `background.js` via `window.postMessage()`. Finally, the background script sends the data via fetch to a server hosted by the Polytechnic University of Turin, where the data are stored in a database for later analysis.

Here the main code snippets for the hooking of the Privacy Sandbox APIs in `injected.js`:

```
1    function hookAPI(obj, methodName, funcFilter, description) {
2      try {
3        const original = obj[methodName];
4        if (typeof original !== "function") return;
5
6        obj[methodName] = new Proxy(original, {
7          apply(target, thisArg, argumentsList) {
8            logApiCall(`${obj.constructor?.name || "Unknown"}.${
   methodName}`, argumentsList, funcFilter, description);
9            window.__paiAnyHooked = true;
10           return Reflect.apply(target, thisArg, argumentsList);
11         }
12       });
13     } catch (e) {
14       console.warn(`[PAI Tracker] Failed to hook ${methodName
   }:`, e);
15       }
16    }
```

Listing 3.2: Hooking of the Privacy Sandbox APIs in injected.js

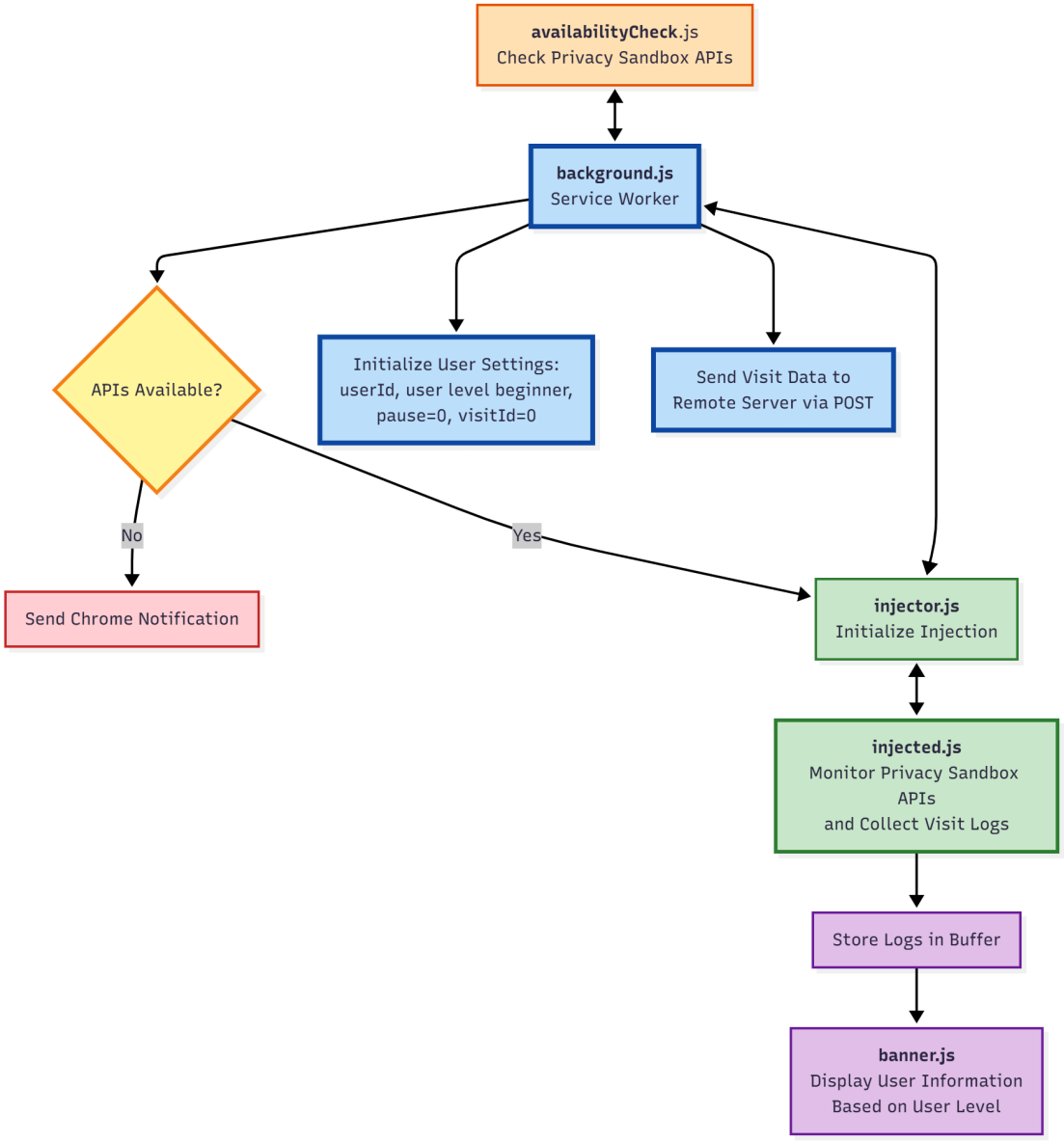

Figure 3.2: Architecture of the Chrome Extension.
Legend: Orange: API availability check; Blue: Service worker tasks; Yellow: Decision process; Red: Notification to user; Green: Injection and monitoring of APIs; Purple: User info display.

3.2.2 APIs Used for Data Collection

To better understand the following sections that describe how the extension collects data, here is a brief overview of the main APIs used. We start with the Chrome

APIs:

- `chrome.storage`: This API is used to store and retrieve data locally within the extension. The extension uses both *chrome.storage.local* for data that should remain on the local device (for example, `visitId`) and `chrome.storage.sync` for data that should be synchronized across devices (for example, `userId`).
- `chrome.cookies.getAll()`: This API is used to retrieve all cookies associated with a specific URL (with null value matches all URLs). The extension uses this function to collect information about cookies set during the browsing session, including their names, values (hashed for privacy), domains, expiration dates, and flags (e.g., HttpOnly, Secure).
- `chrome.notifications`: This API is used to create and manage notifications within the Chrome browser. The extension uses this API to inform users about important events, such as the availability of the Privacy Sandbox APIs in their browser.

The extension also uses the following Web APIs:

- `window.postMessage()`: This API is used to facilitate communication between different contexts, such as between the injected script and the content script.
- `fetch()`: This API is used to make network requests to send the collected data to a remote server for storage and analysis.

3.2.3 Data Collected

The extension collects data via two fetch requests during the browsing session to a server hosted by the Polytechnic University of Turin:

- API calls: The extension logs all calls to the Privacy Sandbox APIs.
- Browsing history and cookies: The extension records all visited URLs and cookies set during the browsing session.

Each API call is logged with the following information. The code snippet 3.3 shows the log-object structure that is collected for each Privacy Sandbox API call:

```
1 isCollectionPaused().then((paused) => {
2         if (paused) {
3           console.log("[PAI Tracker] Collezione temporaneamente
    disabilitata");
4           return;
5         }
```

```
6
7         Promise.all([
8           fetchUserId(),
9           getVisitIdCached(),
10          hashUrlParameters(location.href),
11          hashUrlParameters(document.referrer)
12        ])
13          .then(([clientId, visitId, hashedUrl, hashedReferer]) =>
    {
14            const log = {
15              clientId,
16              api: name,
17              timestamp: new Date().toISOString(),
18              args,
19              url: hashedUrl,
20              referer: hashedReferer,
21              caller: caller,
22              visitId
23            };
24
25
26            if (window === window.top) {
27              addLogEntry_buffer(log);
28            } else {
29              notifyParent(log);
30            }
31
32            try {
33              const event = new CustomEvent("__pai_log_event", {
    detail: log });
34              window.dispatchEvent(event);
35            } catch (err) {
36              console.warn("Errore invio log al content script:",
    err);
37            }
38          });
39      });
```

Listing 3.3: API Call Log Data Structure

This log object contains the following fields:

- `clientId`: Unique identifier for the user of the extension (stored in *syncStorage*). This ID is generated when the extension is installed and remains constant across browsing sessions.
- `api`: Name of the Privacy Sandbox API being called.
- `timestamp`: ISO timestamp of when the API call occurred.

- `args`: Arguments passed to the API call.
- `url`: URL of the page where the API call was invoked. Query-parameter keys are kept in cleartext, while their values are hashed using SHA-256 and truncated to eight characters. This preserves some information about the URL while helping to ensure user privacy.
- `referer`: URL of the referer page, if available. It is hashed in the same way as `url`.
- `caller`: Information about the caller of the API, if available (e.g., the third-party script that invoked the API).
- `visitId`: Unique identifier for the current page visit. This ID is stored in `localStorage` and is updated whenever the user navigates to a new page.

The other fetch request sends the browsing history and cookies set during the browsing session.

```
1  if (window === window.top) {
2      isCollectionPaused().then((paused) => {
3        if (paused) {
4          return;
5        }
6
7        Promise.all([
8          fetchUserId(),
9          getVisitIdCached(),
10         fetchCookies(location.href),
11         hashUrlParameters(location.href),
12         hashUrlParameters(document.referrer)
13       ])
14         .then(([clientId, visitId, cookies, hashedUrl,
     hashedReferer]) => {
15           const siteLog = {
16             clientId,
17             url: hashedUrl,
18             referer: hashedReferer,
19             timestamp: new Date().toISOString(),
20             cookies,
21             visitId
22           };
23
24           try {
25             const event = new CustomEvent("__pai_site_event", {
     detail: siteLog });
26             window.dispatchEvent(event);
27           } catch (err) {
```

```
28              console.warn("Errore dispatch site event:", err);
29            }
30          });
31      });
32  }
```

Listing 3.4: Browsing History and Cookies Log Data Structure

This log object contains the following fields:

- `clientId`: Unique identifier for the user of the extension (stored in *syncStorage*). This ID is generated when the extension is installed and remains constant across browsing sessions.
- `url`: URL of the page where the cookies were collected. Query-parameter keys are kept in cleartext, while their values are hashed using SHA-256 and truncated to eight characters. This preserves some information about the URL while helping to ensure user privacy.
- `referer`: URL of the referer page, if available. It is hashed in the same way as `url`.
- `timestamp`: ISO timestamp of when the cookies were collected.
- `cookies`: List of cookies set on the page, including their name, hashed value, domain, expiration date, partitionKey, and flags (e.g., HttpOnly, Secure).
- `visitId`: Unique identifier for the current page visit. This ID is stored in `localStorage` and is updated whenever the user navigates to a new page.

3.2.4 User Interface

The extension includes a popup that is displayed when the user clicks the extension icon. The popup displays information according to the user level, which can be either “beginner” or “expert” (the user level is stored in `localStorage`). The default level is “beginner”, but the user can switch to “expert” mode by selecting the expert option in the popup window. The user level determines the amount of information displayed about API calls.

- In `beginner mode`, the extension displays a simplified view, showing only a brief explanation with a ’More Info’ button that provides a higher-level description of the Privacy Sandbox API and its caller (if available).
- In `expert mode`, the extension provides a detailed view of API calls, including the exact JavaScript API name and the caller of that API (if available).

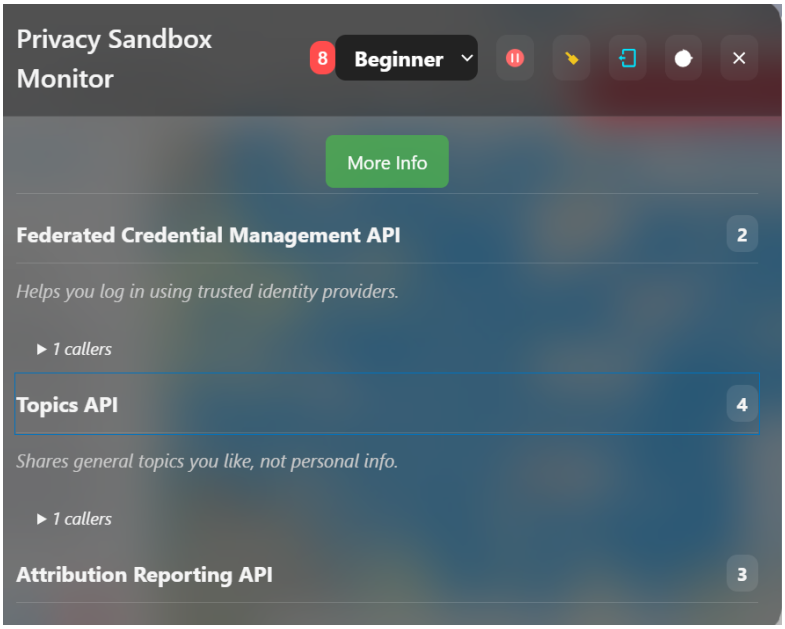

Figure 3.3: Beginner Mode.

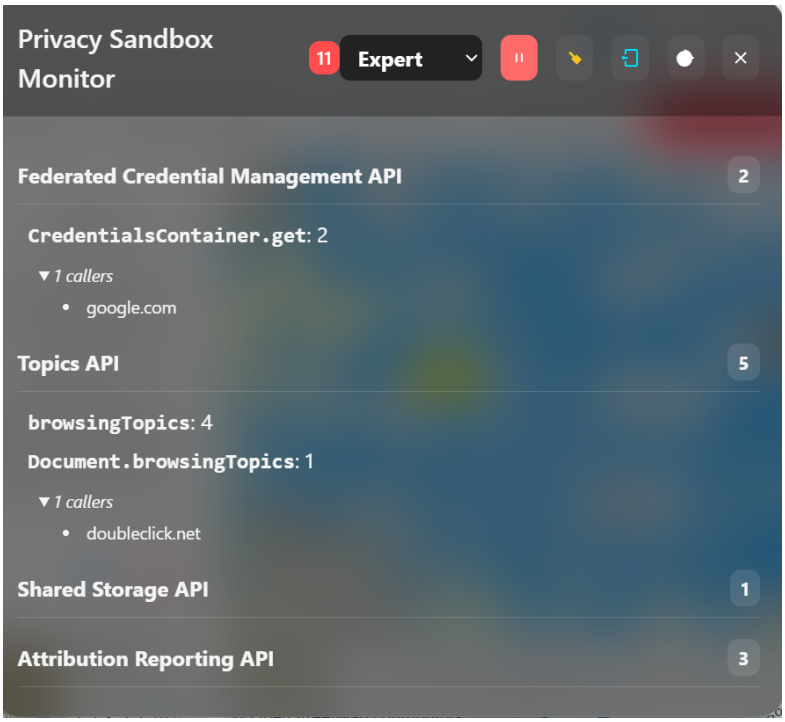

Figure 3.4: Expert Mode.

3.3 Extension distribution

The extension distribution is managed through a Google Form [42] that collects user consent. The form explains the purpose of the study in the attached information sheet, other than collect some information related to the user's browsing habits :

- If the user uses or plan to use ad blockers
- if the user uses or plan to use chrome as main browser for the purpose of the study

The form also provides an attached privacy policy pursuig the article 13 of the GDPR regulation. After submitting the form, users receive a link to a ZIP file containing the extension and an installation guide, provided in both English and Italian for clarity. See Appendix A for the installation instructions.

Chapter 4

Crawler Results and Analysis

In this chapter we present the results obtained from the crawler and analyze them.

4.1 Crawler Performance

First, we present the performance of the crawler in terms of the number of successful visits to websites and the number of accept and deny buttons found by the crawler.

As shown in Figure 4.1, the crawler visited around 8,500 websites successfully and found about 3,700 accept buttons and 2,300 deny buttons. The number of successful first visits for the deny container is consistently higher than for the accept container. This is probably because, when we start the two containers in parallel, the "deny" container starts slightly earlier than the "accept" container; thus, the second visit may be recognized as a bot by some websites and blocked. The number of accept buttons found is approximately 44% of successful visits, while deny buttons are found in about 27% of successful visits. As discussed in Chapter 3, the methodology used to detect the deny button is the same as for the accept button. However, denying a cookie is typically more difficult than accepting it, because websites often place deny controls behind paywalls or implement deny actions using checkboxes or other elements that are harder to detect.

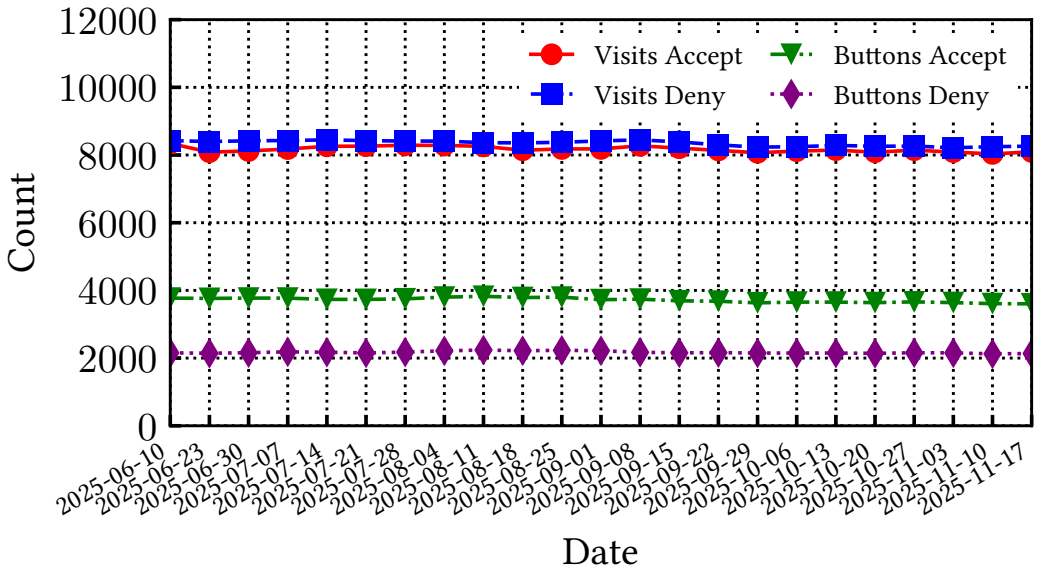

Figure 4.1: Number of successful visits and buttons found over time.

4.1.1 Privacy Sandbox - General Overview

In this section we present a general overview of websites' usage of the Privacy Sandbox.

The two figures 4.2 and 4.3 report the adoption of Privacy Sandbox APIs during the first and the last week of measurements. In the first week (Figure 4.2), API usage generally increases after accept button clicks and decreases after deny button clicks. This pattern is consistent across all Privacy Sandbox APIs, indicating that websites' use of these APIs is influenced by the cookie consent choice. The CHIPS, Topics API, Shared Storage API, and Attribution Reporting API are the most widely adopted APIs, reflecting their central role in the Privacy Sandbox ecosystem.

In the last week (Figure 4.3), the same pattern persists the APIs usage is generally the same of the first week , with a significant decrease in the usage of the Protected Audience API.

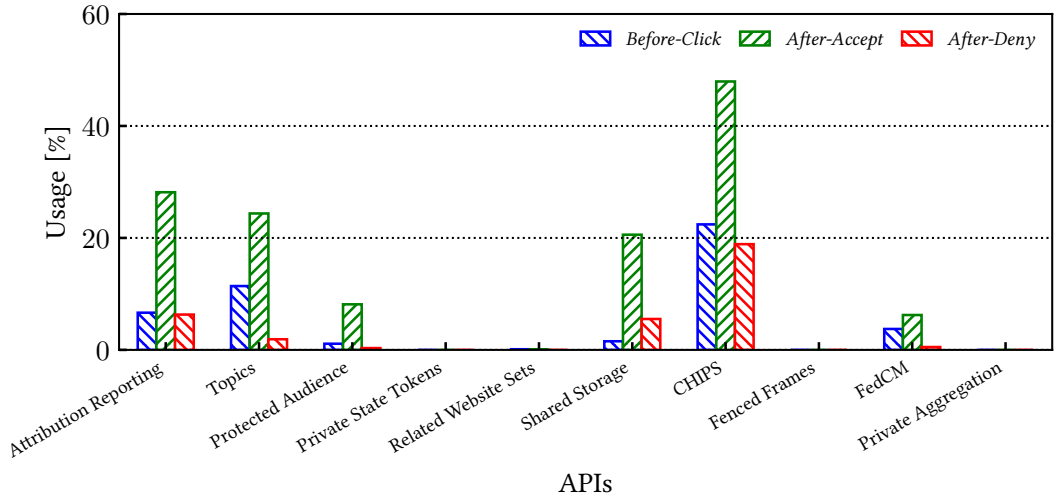

Figure 4.2: Usage of Privacy Sandbox APIs over the first week of measurement after accept/deny button is clicked.

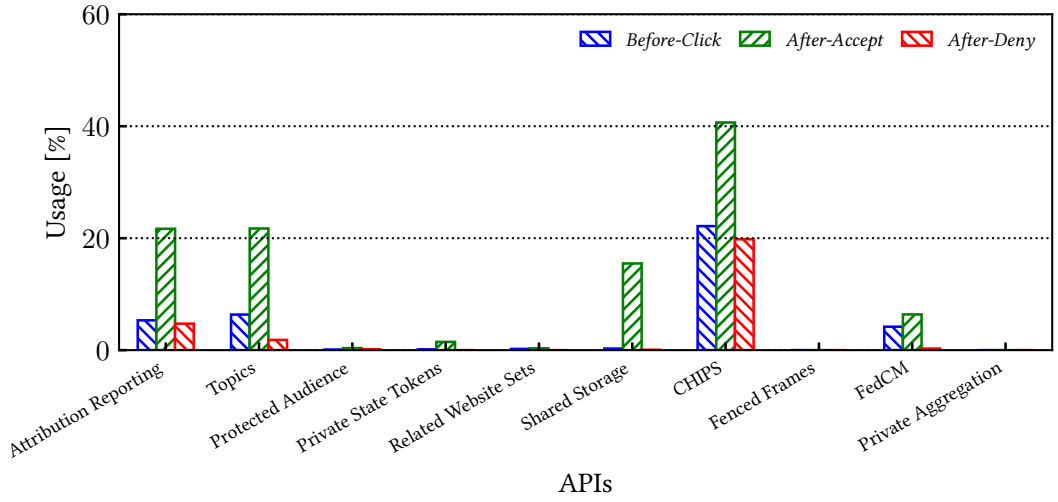

Figure 4.3: Usage of Privacy Sandbox APIs over the last week of measurement after accept/deny button is clicked.

4.1.2 Third party cookie vs CHIPS

Let's now compare the usage of third-party cookies and CHIPS. As shown in the figure 4.4, the usage of third-party cookies is significantly lower than that

of CHIPS after the accept button is clicked. This is due to the fact that many websites have already started migrating from third-party cookies to CHIPS, and in particular after the week of 21st of July 2025 there is a significant drop in the usage of third-party cookies. This trend is due to the fact that the big players in

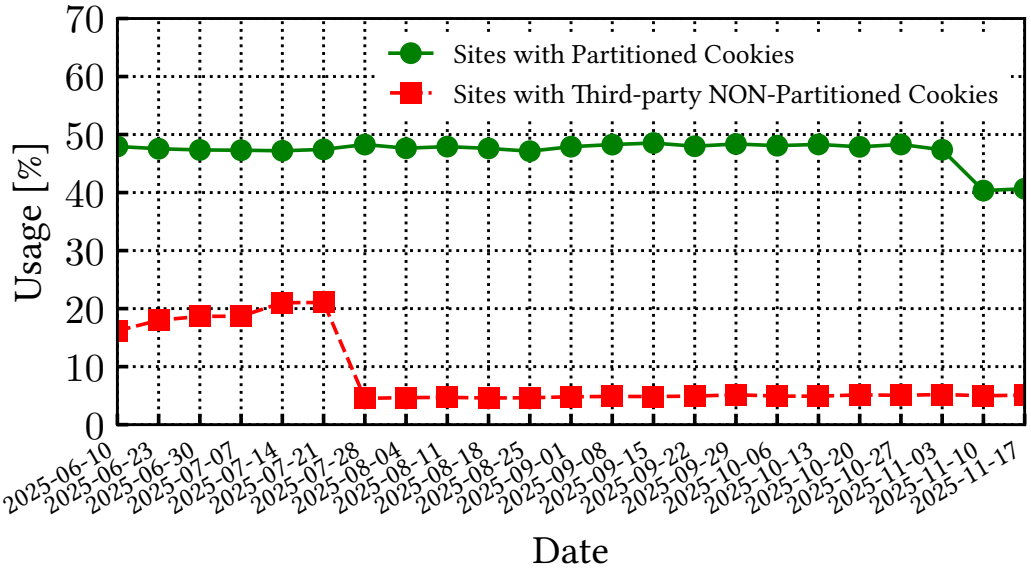

Figure 4.4: Comparison between third-party cookies and CHIPS after accept button is clicked.

the advertising ecosystem, such as Google and Microsoft, have started to do not use third-party cookies as shown in the figure 4.5.

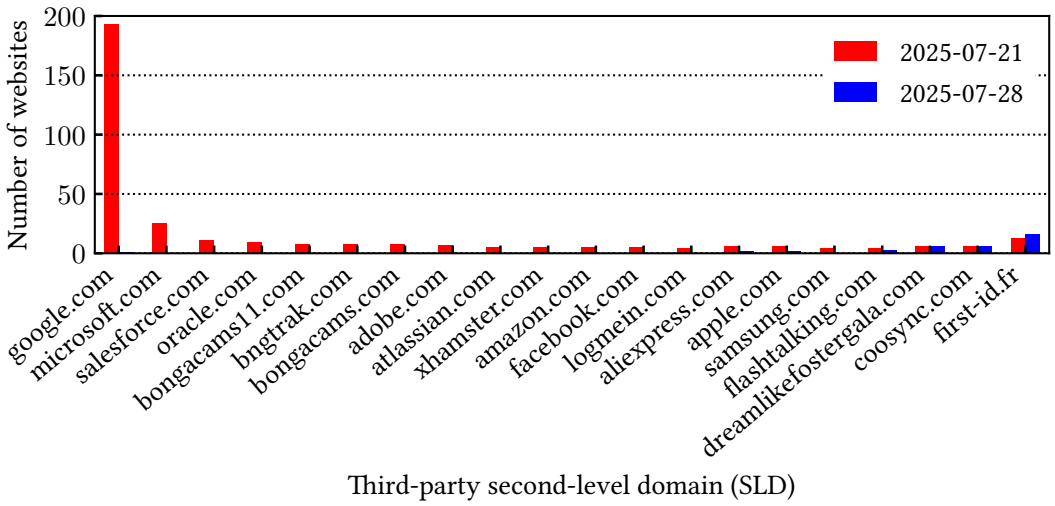

Figure 4.5: Analysis of drop in third-party cookies usage by big players.

4.1.3 Co-utilization of Privacy Sandbox APIs

In this section we analyze the co-utilization of different Privacy Sandbox APIs by websites. In the picture 4.6 we analyse the co-utilization of privacy sandbox APIs that is, the probability that a visit which triggers a first API (API1, on the rows) also triggers a second API (API2, on the columns) in relation to their support means that some APIs since are called much more less frequently than others so the co-utilization matrix may be not expressive. The more frequent co-utilization is between Shared Storage and Topics APIs. While Federated Credential Management, Attribution Reporting APIs and CHIPS are co-utilized with multiple other APIs meaning that they utilization is transversal to different use cases. This figure highlights that current deployments tend to cluster around a few core APIs (CHIPS,Topics, Shared Storage, Attribution Reporting, Federated Credential Management), while other components of the Privacy Sandbox ecosystem are still rarely combined in real-world usage.

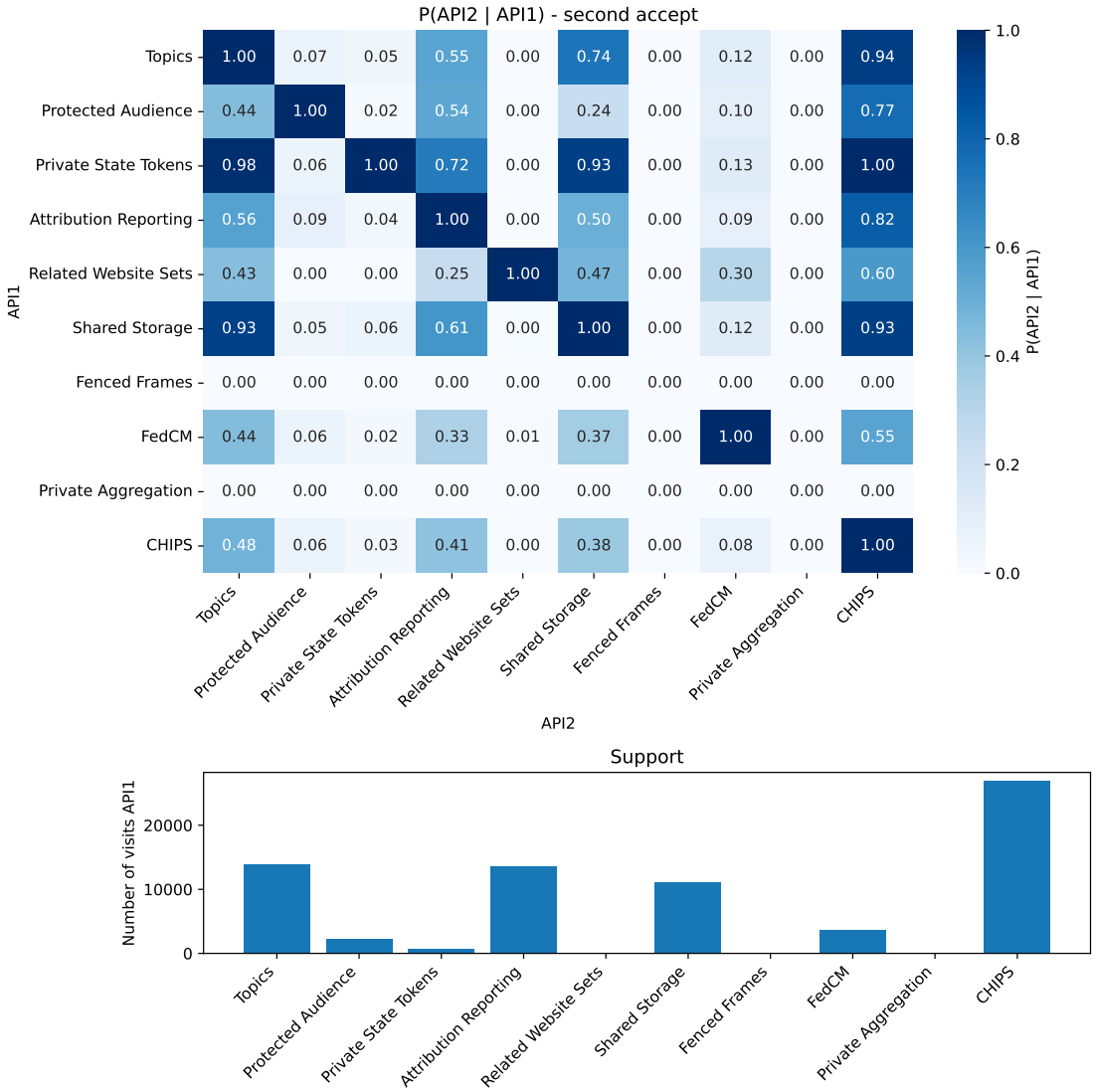

Figure 4.6: Co-utilization of Privacy Sandbox APIs from crawler data.

4.2 Protected Audience API

In this section we focus on the Protected Audience API, analysing API usage and adoption over time. As shown in the figures 4.7, usage of the Protected Audience API is around 6% after the accept button is clicked and around 1% after the deny button is clicked. We also observe that API usage generally decreases over time. This may be because parties that were using the API have reduced their usage or switched to other APIs. Following Google's announcement in May 2025 that cookies would not be deprecated immediately, some parties may have stopped

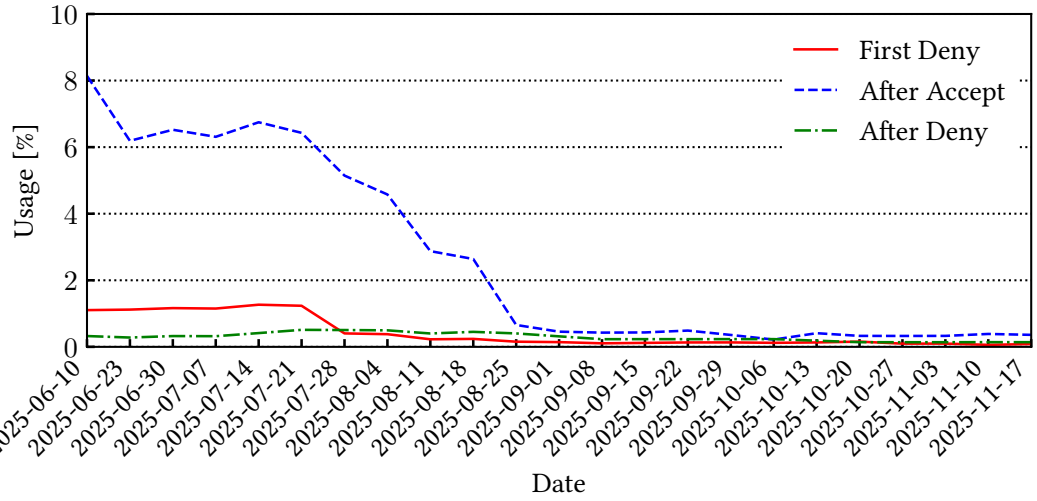

Figure 4.7: Percentage of websites using Protected Audience API after accept button is clicked.

investing in the Protected Audience API. This effect is particularly evident in the week of 18–25 August 2025, when usage after accept-button clicks dropped from 2.6% to 0.6%.

4.2.1 How the API usage drops over time

In this section we will analyze how the usage of the Protected Audience API drops over time after the accept or deny button is clicked.

As shown in Figures 4.8 and 4.9, usage of the Protected Audience API drops significantly; the decrease is particularly marked for the *navigator.joinAdInterestGroup* function. As noted 4.8 and 4.9, this decline may reflect parties decreasing their use of the API or switching to other solutions. The drop is observed after both accept and deny button clicks.

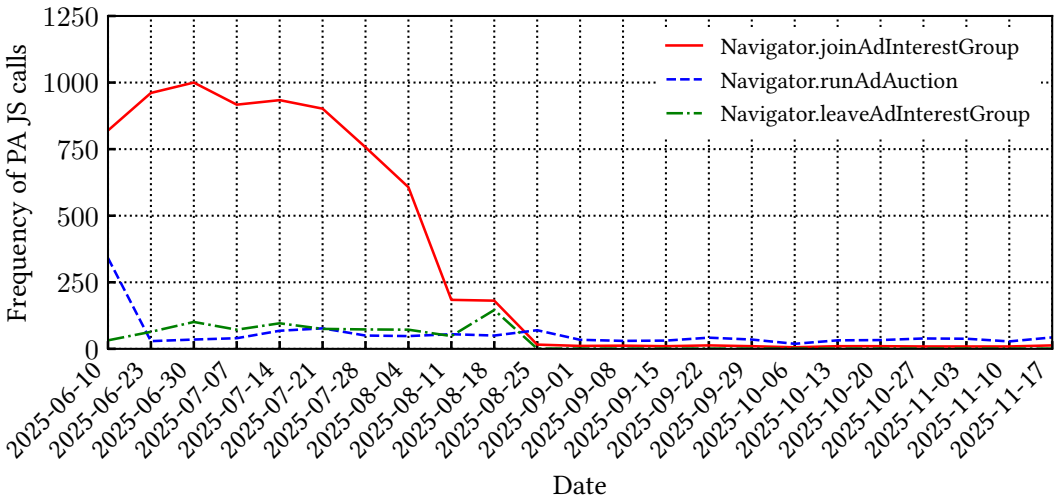

Figure 4.8: Drop of Protected Audience API usage over time after accept button is clicked.

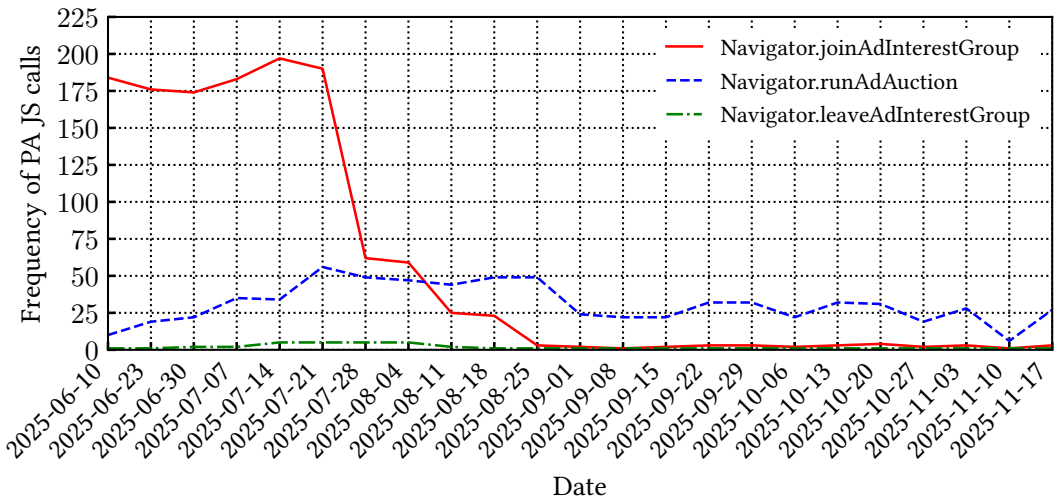

Figure 4.9: Drop of Protected Audience API usage over time after deny button is clicked.

4.2.2 Analysis of parties that drop the usage of the API

In this section we analyze which parties stopped using the API over time.

These two figures 4.10 and 4.11 clearly show that the parties that dropped usage

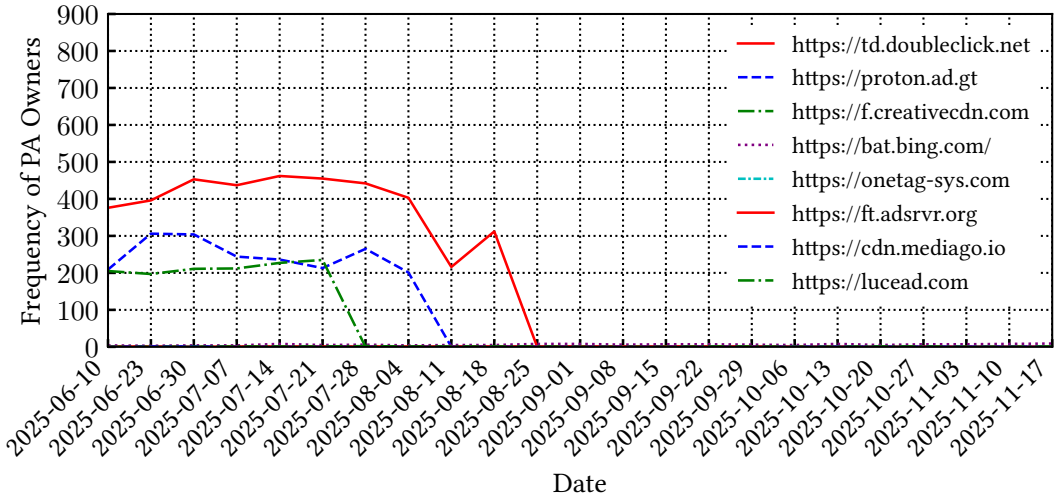

Figure 4.10: Parties that drop the usage of Protected Audience API after accept button is clicked.

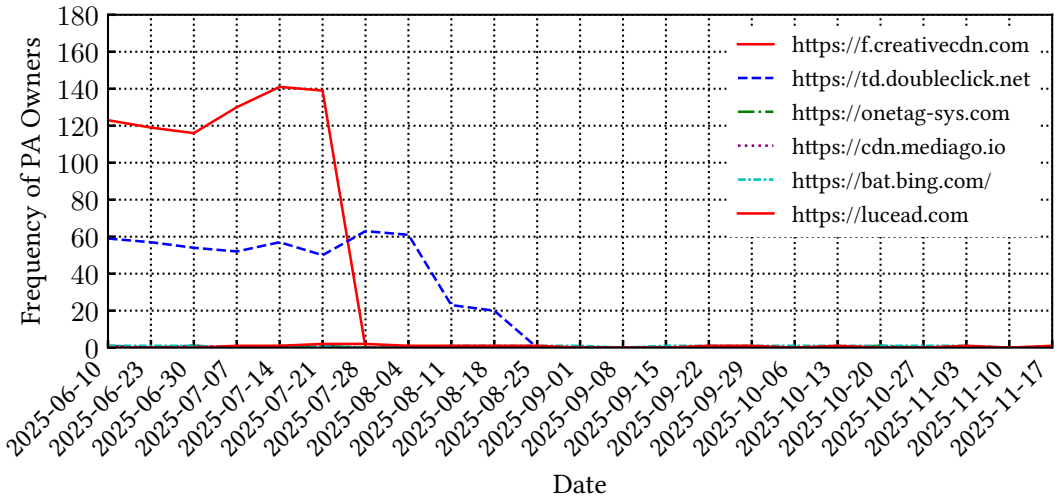

Figure 4.11: Parties that drop the usage of Protected Audience API after deny button is clicked.

of the API are mainly creative.cdn, doubleclick.net and proton.ad.gt. This likely reflects a decision by major parties to stop using the API and to cease further investment.

4.2.3 Protected Audience API: In-depth Analysis of a Week

In this section we analyse the Protected Audience API usage during a specific week, examining the number of ads in each interest group, the duration of interest groups, the `biddingLogicUrl`, and the parties involved in bidding.

Number of ads per Interest group owner

As discussed in Chapter 2, interest groups include ads that can be shown to users. From the two figures 4.12 and 4.13, the number of ads per interest group is generally low except for doubleclick.net, which shows a high number of ads per interest group. This is probably because other parties were still testing the API during that week and therefore created interest groups with a small number of ads, while doubleclick.net included multiple ads per group to provide different sizes for different ad spaces.

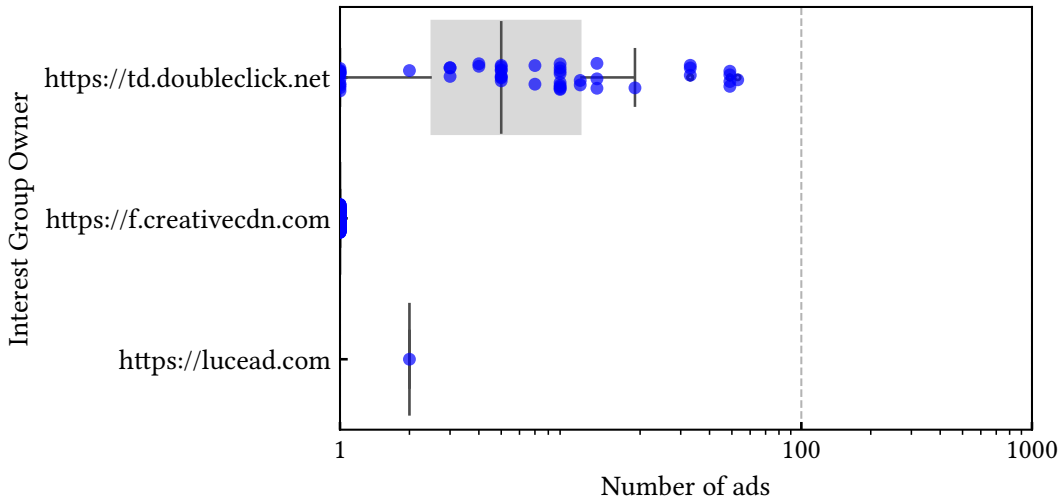

Figure 4.12: Average number of ads per interest group owner after accept button is clicked.

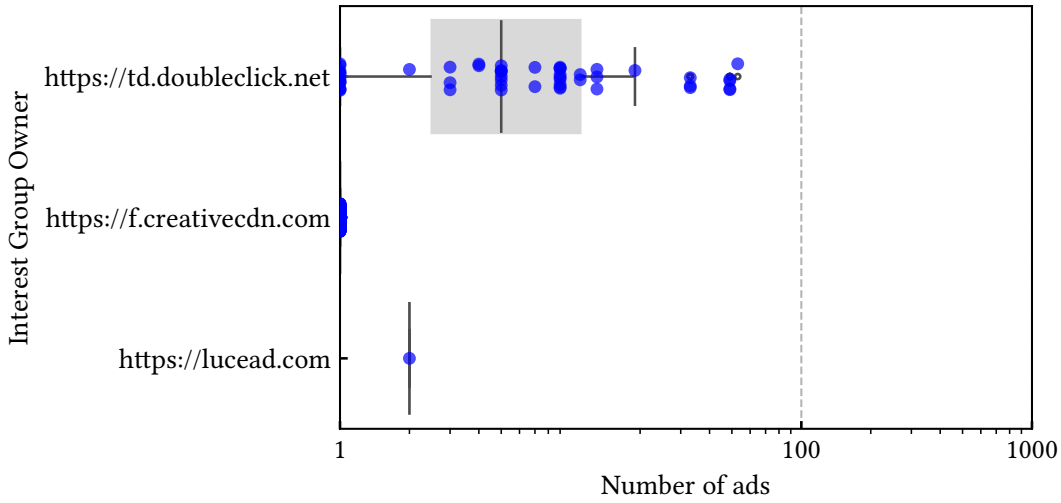

Figure 4.13: Average number of ads per interest group owner after deny button is clicked.

4.2.4 Duration of interest groups

As discussed in Chapter 2, interest groups have a duration that can be set by the party that creates the interest group. The Google documentation [13] suggests that the duration should be set to a maximum of 30 days. From the two figures 4.14 and 4.15, the duration is respected only by adsrvr.org, while other parties set durations longer than 30 days. In fact doubleclick.net sets a interst-group duration slightly above 30 days, other parties such as creativecdn.com and bat.bing.com set a duration of 30,000 days. This suggests that these parties specify the duration in milliseconds instead of seconds, resulting in an unintentionally long duration. This is a potential privacy issue because interest groups kept on the user's device for long periods allow parties to participate in bidding for an extended time.

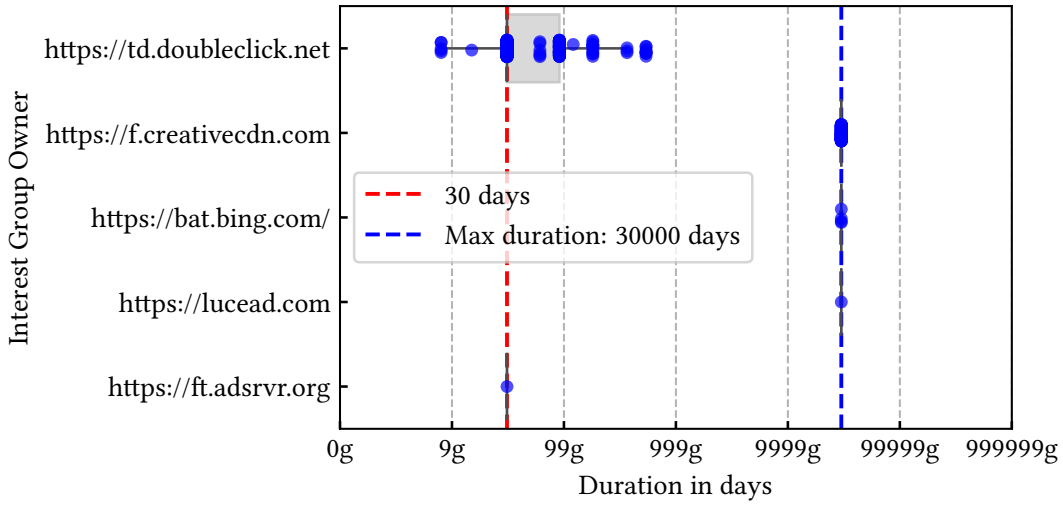

Figure 4.14: Average duration of interest group owner after accept button is clicked.

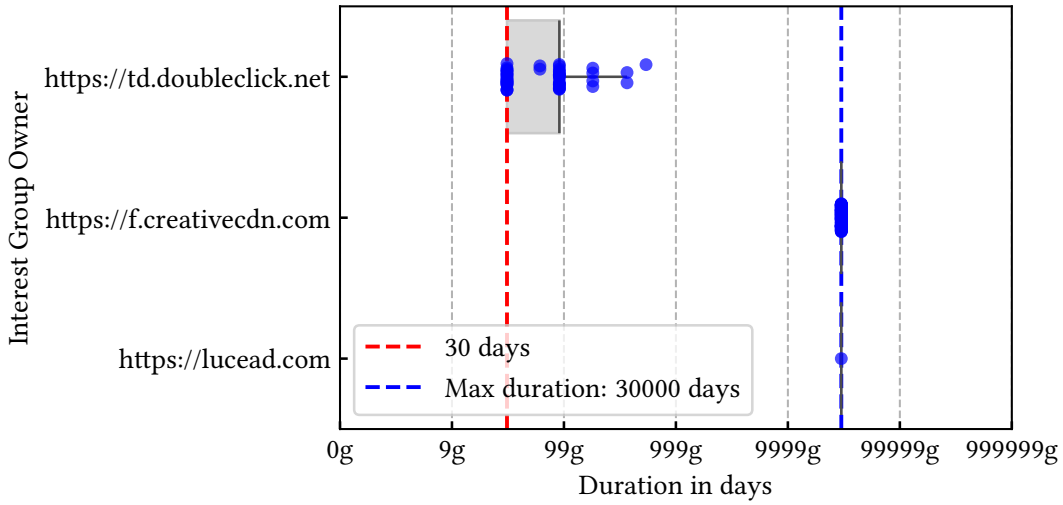

Figure 4.15: Average duration of interest group owner after deny button is clicked.

4.2.5 Most Frequent Bidding Logic Url

As discussed in Chapter 2, interest groups have a `biddingLogicUrl` used for bidding. Analyzing these URLs, we find that the most used `biddingLogicUrl` originates from doubleclick.net. This statistic suggests that doubleclick.net is one of the most

active parties in bidding. The bidding scripts cannot be inspected further because they are obfuscated, which prevents a detailed analysis of the auction logic used to generate bids. This obfuscation is a common practice of advertising platforms like doubleclick.net, where the bidding logic is hidden to protect algorithms and prevent competitors or third parties from reverse-engineering the bidding process.

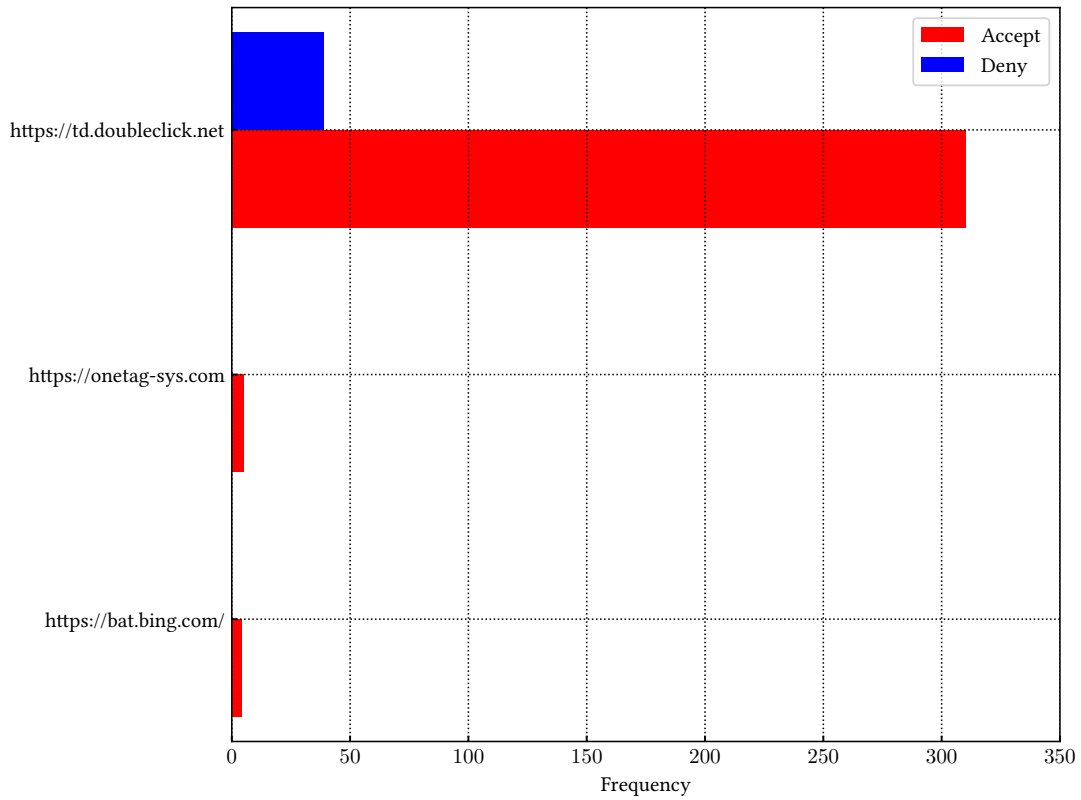

Figure 4.16: Most frequent Bidding Logic Url after accept button is clicked vs deny button is clicked.

Chapter 5

Chrome Extension result and analysis

In this chapter, we present the results obtained from the Chrome Extension. We analyze the data collected during our experiments and discuss the implications of our findings.

5.1 Users Statistics

During the data collection phase using the chrome extension, we had a total of 25 users installing and using the extension per 2 weeks. The form was shared moslty with Politecnico di Torino students and professors, so the majority of users are from this institution. As we said in Chapter 3, the users were asked to repond to a form with some utilization questions.

From the form responses we can see that the 61.9% of the users uses an ad-blocker during their browsing sessions, this could have an impact on the data collected, since ad-blockers can block some of the Privacy Sandbox APIs, in particular the ones related to advertising (e.g. Protected Audience API, Topics API), so the data collected could be lower than the real usage of the APIs. Moreover the 85.7% of the user plan to use or currently use Google Chrome as their main browser.

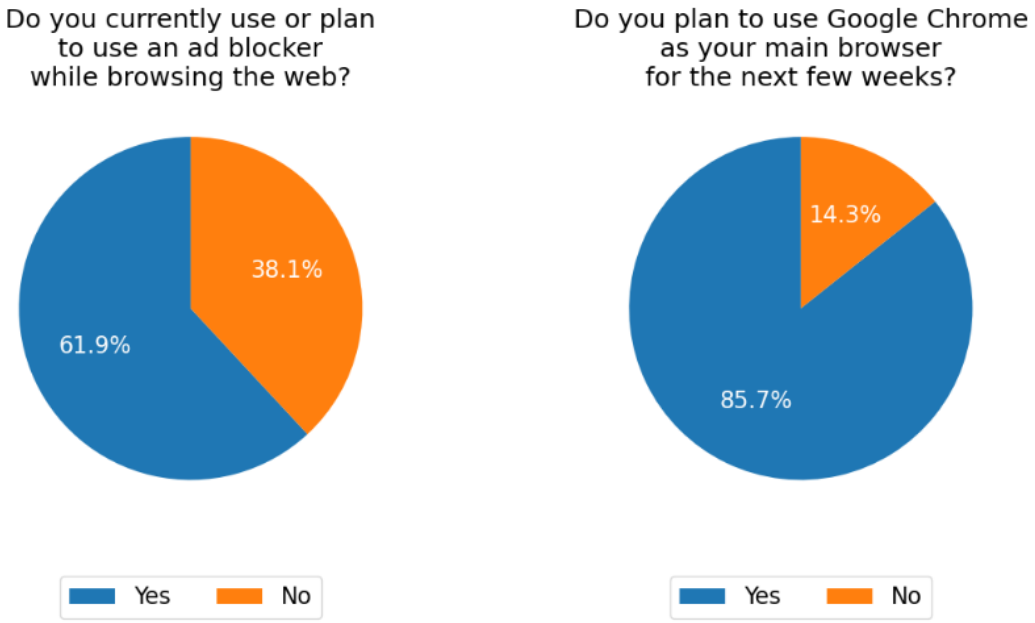

Figure 5.1: Users form responses.

5.2 General Statistics

In this section we present general statistics about the data collected by the Chrome Extension. Differently from the prevoius method based on web crawling, here we have data collected from real users browsing the web. So the data contains also url that are internal pages of websites that are not investigated with the first method. So the picture 5.2 describe the number of internal vs main pages visited during the data collection.

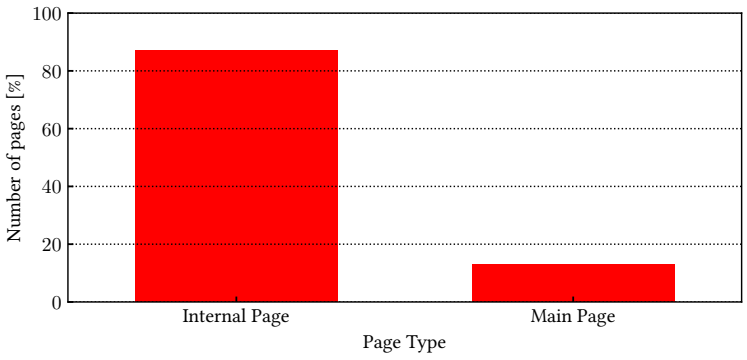

Figure 5.2: Internal vs Main pages visited.

Moreover deeper analysis on domains visited are jupyter.polito.it and google.com other popular domains are outlook.com and github.com, that are mostly used for work related activities, this describes the fact that the chrome estension was used mostly by polito students and professors. In the figure 5.3 we can see the top 20 domains visited during the data collection.

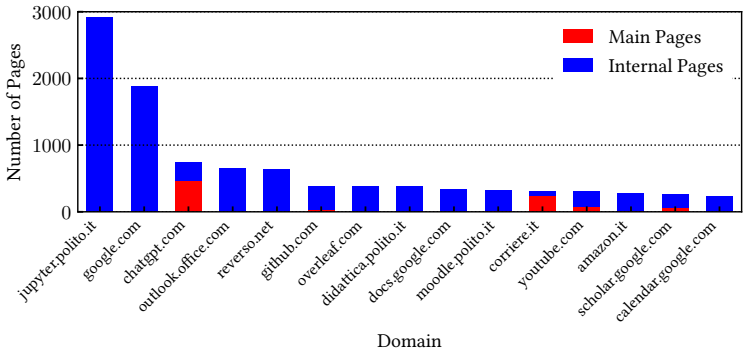

Figure 5.3: Top 20 domains internal vs main pages visited.

Furthermore we will present general statistics evaluating the visits to the different websites and the number of API calls recorded. In fact as we can see from the figure 5.4, the number of websites with API calls recorded is less than the number of websites visited. This is due to the fact that not all websites use the Privacy Sandbox APIs, so some visits do not generate any API calls, moreover as we see in the chapter 4 from the crawler vantage, the use of Privacy Sandbox APIs increase after cookie consent acceptance, so if the user deny the cookie consent, the number of API calls recorded will be lower.

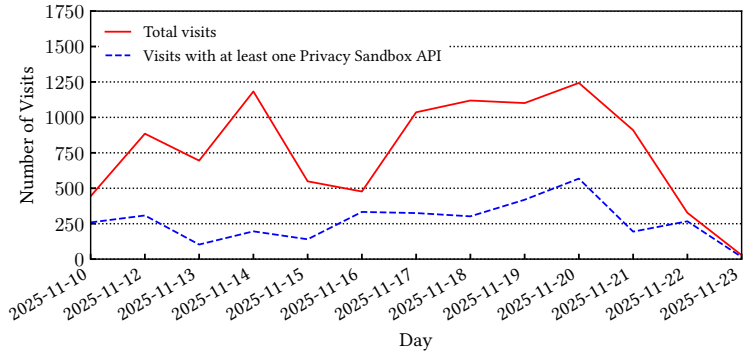

Figure 5.4: Number of API calls vs Number of websites visited.

5.2.1 Referred vs direct traffic

In this section we analyze the traffic sources of the visits. As we can see from the figure 5.5, most of the API calls are generated by referred traffic, meaning that the user was referred to the website by another website. The direct traffic intead i sthe traffic generated by users who directly type the URL in the browser or use a bookmark.

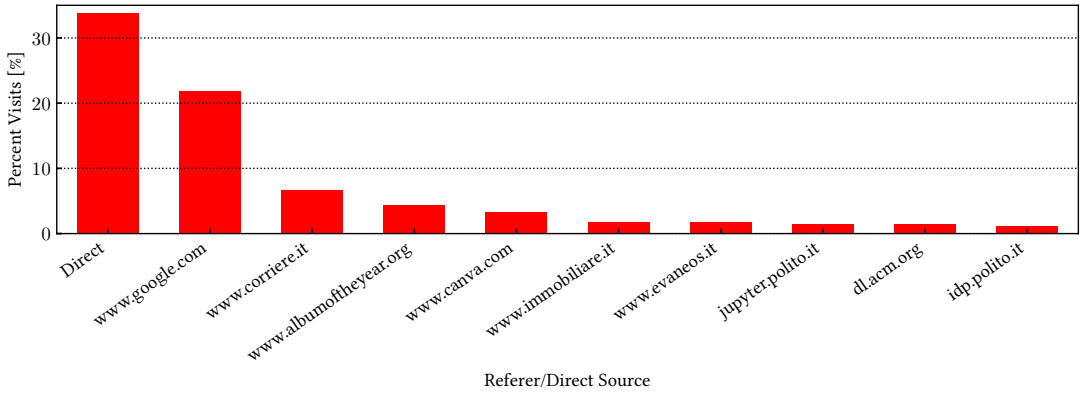

Figure 5.5: Referred vs Direct traffic.

5.3 Privacy Sandbox APIs measurements and analysis

In this section we analyze the usage of the different Privacy Sandbox APIs. As we can see from the figure 5.6, the Attribution Reporting API is the most used API, followed by the topics API and the Shared Storage API.

Let's analyse now the third party that use the Privacy Sandbox APIs. As we can see from the figure 5.7, doubleclick.net is the most used third party, followed by rubiconproject.com and google.com.

Moreover we analyze the different APIs in more detail and the relationship between them. In the figure 5.8 we can see the top third party appearance in different visited sites and how many different APIs of the Privacy Sandbox framework they use. The analysis shows that double click is the third party that appears in more different sites, in particular usually appears in the same sites of the rubiconproject.com.

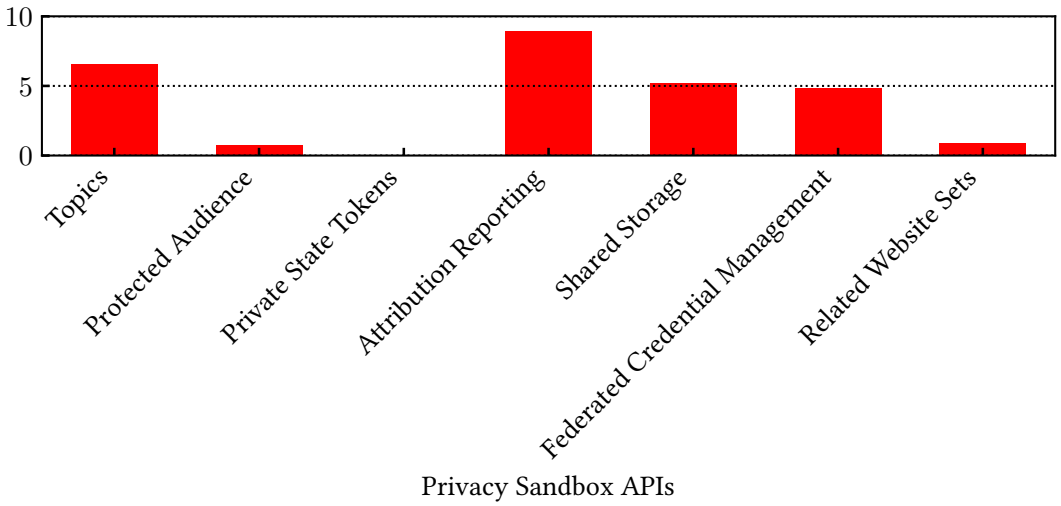

Figure 5.6: Usage of Privacy Sandbox APIs.

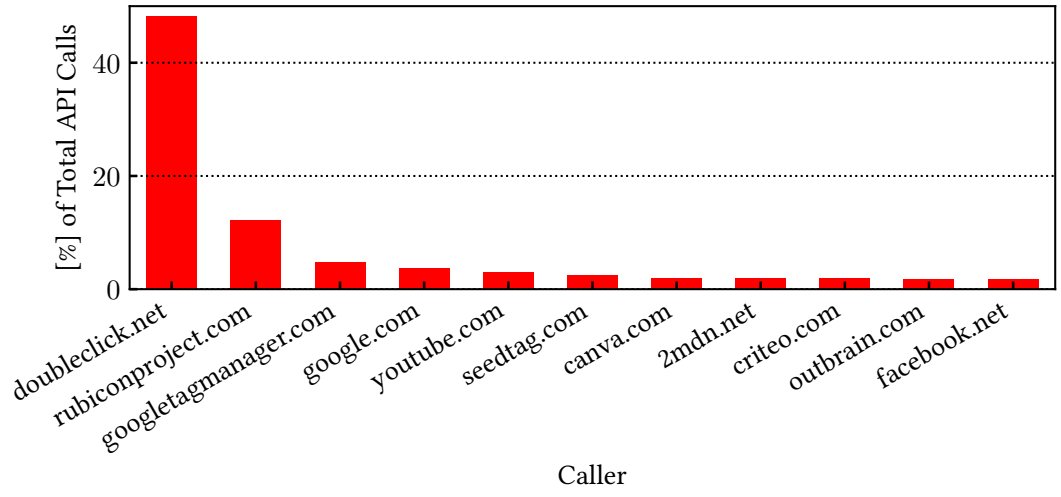

Figure 5.7: Third party usage of Privacy Sandbox APIs.

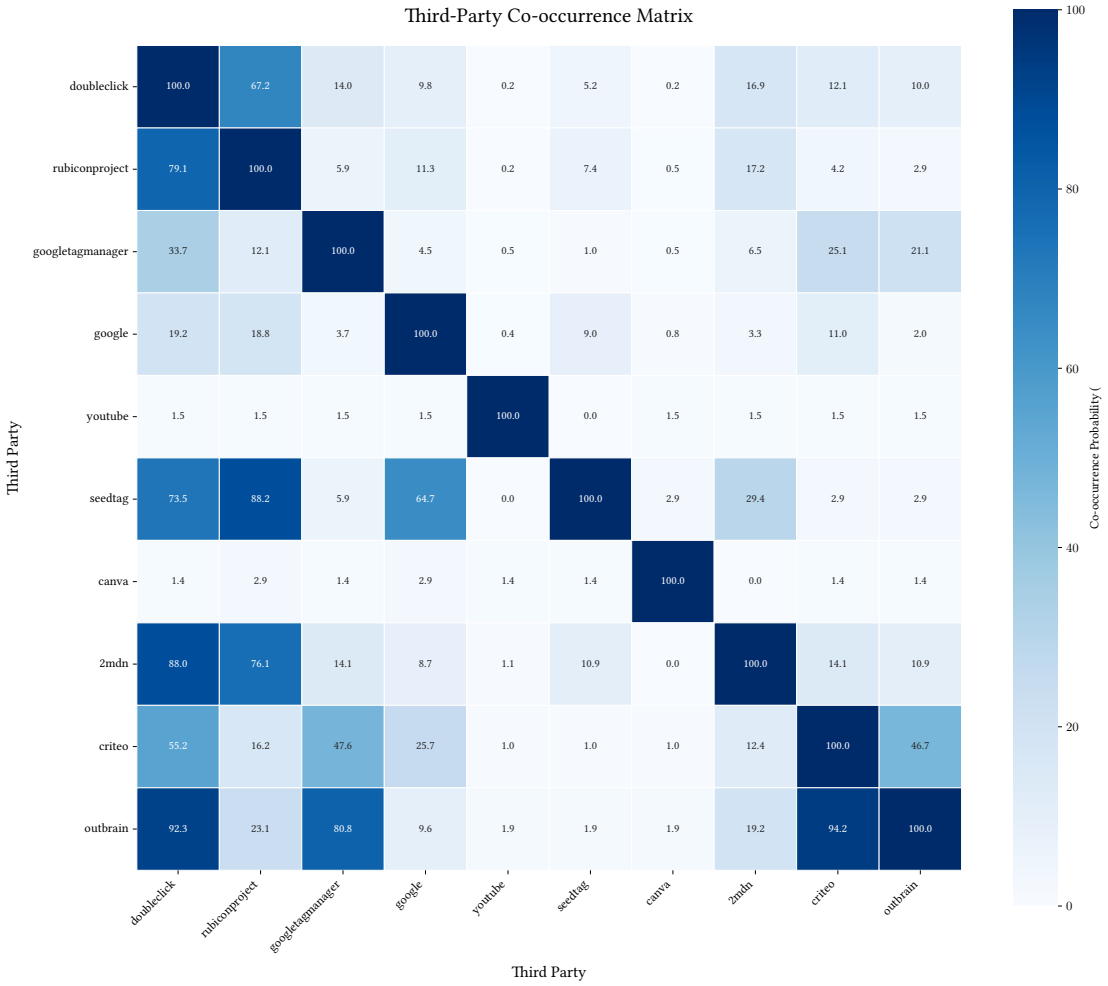

Figure 5.8: Third party appearance in different visited sites.

5.3.1 Co-utilization of APIs

In the figure 5.9 we can see the co-utilization of different APIs by web sites. The matrix shows the probability that a website using a API1 also use API2.

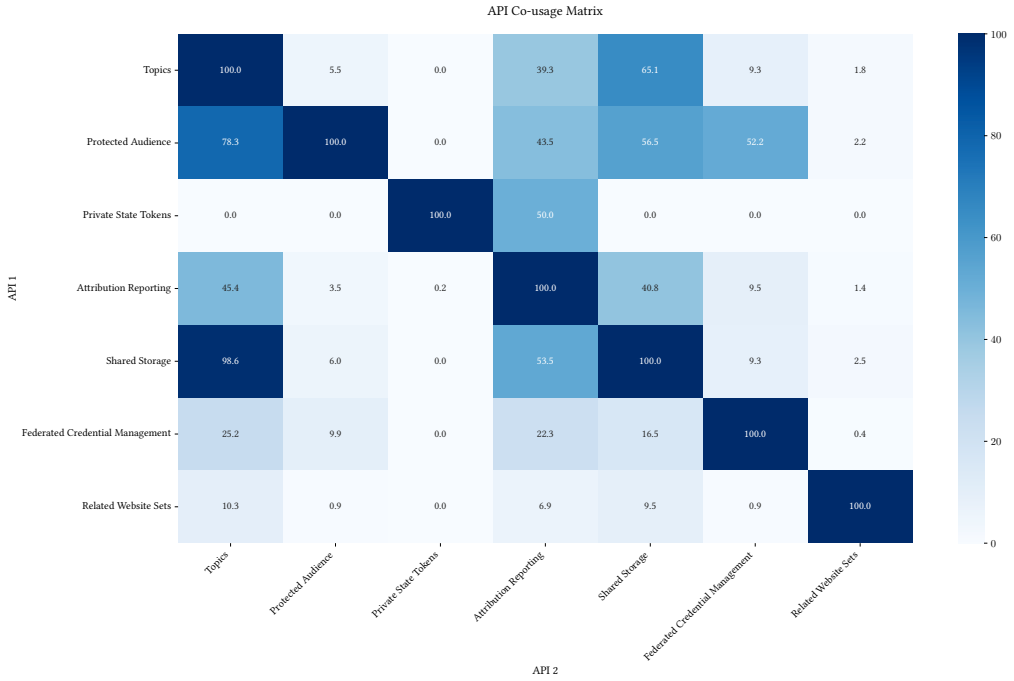

Figure 5.9: Co-utilization of Privacy Sandbox APIs.

The relationship between Topics and Shared Storage in the data shows an interesting implementation pattern. When we look at the 98.6% co-occurrence rate, it becomes clear that websites deploying Shared Storage are almost always using Topics alongside it.

What really stands out in the matrix is how Attribution Reporting behave differently from the other APIs. While most APIs show strong correlations with specific partners, Attribution Reporting appears active across multiple scenarios it co-occurs with Topics in 45.4% of visits, with Shared Storage in 40.8%, and even shows up alongside Protected Audience and Federated Credential Management.

Chapter 6

Conclusion

In this thesis, we presented a dual-vantage measurement of the adoption and implementation of the Privacy Sandbox APIs in the wild, combining a Chromium-based web crawler on the Tranco Top-10k with a Chrome extension deployed to real users. The goal was to understand how these APIs, and in particular the Protected Audience API, are used in practice and whether their implementation and use aligns with the privacy and governance guarantees described in the Google documentation.

6.1 Main findings

From the crawler vantage, we showed that Privacy Sandbox adoption is driven primarily by the CHIPS, the Topics API, the Attribution Reporting API, and the Shared Storage API, all of which are more frequently used after cookie-consent acceptance than after denial. We observed that CHIPS is rapidly replacing traditional third-party cookies for big third parties with a significant drop in third-party cookie usage after July 2025, especially for large advertising platforms such as Google and Microsoft. In contrast, the Protected Audience API exhibits relatively limited adoption (around 7% of sites after consent acceptance and about 1% after denial) with usage dropping sharply. By inspecting interest-group parameters, we highlighted how many actors configure long-lived interest groups that exceed the 30-day duration (max lifetime suggested by Google), and how `doubleclick.net` dominates both the number of ads per interest group and the most frequent `biddingLogicUrl`. Finally, the analysis of non-obfuscated auction code from `lucead.com` revealed the presence of a whitelist of allowed buyers, exposing a governance issue: nothing in the current implementation prevents sellers from restricting competition beyond the purpose of the Protected Audience design.

From the extension vantage, we collected data from 25 users, mostly students

and professors at Politecnico di Torino, whose browsing activity is characterized by intensive use of work-related services and a high adoption of ad blockers. This dataset complements the crawler by capturing internal pages and real interaction paths, showing that many Privacy Sandbox invocations occur beyond landing pages and that not all visited sites rely on these APIs. The extension measurements points out that the Attribution Reporting is the most widely used API, followed by the Topics API and the Shared Storage API, and that third parties such as `doubleclick.net`, `rubiconproject.com`, and `google.com` are central actors in the ecosystem. The co-utilization matrix indicates that sites using Shared Storage almost always pair it with Topics, while Attribution Reporting plays a transversal role, co-occurring with multiple APIs as a general measurement layer, and most API calls are generated by referred rather than direct traffic.

Together these two vantage points provide a picture of a web ecosystem that is experimenting with the Privacy Sandbox but still dependent on cookies and traditional tracking mechanisms, with potential privacy and governance issues that need to be addressed to ensure that the Privacy Sandbox respects its original goals.

6.2 Contributions

This thesis makes three main contributions.

- It introduces a dual-vantage measurement methodology that combines large-scale automated crawling with a Chrome extension used during real user browsing sessions, enabling a more complete view of Privacy Sandbox usage across both top-ranked sites and real browsing sessions.
- It provides a longitudinal analysis,from the crawler vantage, of how Privacy Sandbox APIs are adopted and distinguishing behaviour before and after cookie-consent acceptance and denial.
- It provides a complementary perspective from the extension vantage, capturing real browsing sessions and internal pages to reveal usage patterns not observable with the crawler vantage.
- It uncovers concrete implementation and governance issues, including long-lived interest groups and auction code that include buyer whitelists, highlighting how these practices may undermine both privacy and competition goals.

6.3 Limitations and future work

Despite these contributions, the work has some limitations that suggest directions for future research. First, the crawler is limited to the Tranco Top-10k , while the

extension dataset is restricted to a small, non-representative cohort of users with high ad-blocker adoption, which may underestimate actual API usage. Second some privacy mechanisms, such as k-anonymity enforcement for interest groups, are not deployed in the browser, so the analysis cannot assess their effectiveness in practice.

Future work could extend the longitudinal measurements to a longer period and additional regions, tracking how adoption evolves. The extension could be deployed to a larger and more diverse user base to improve representativeness, this could be achieved through a collaboration with the Politecnico di Torino , other university and research institutions. The dual vantage metology could be applied to future privacy technologies to assess their real-world adoption and implementation. Finally, more collaboration between measurement researchers, browser vendors, and regulators could help to define governance rules for auction code, attestation processes, and consent handling, ensuring that the technical design of the Privacy Sandbox is matched by compliant implementations in the wild.

Appendix A

Installation Guide / Guida all' Installazione

A.1 Versione Italiana

A.1.1 Guida all'Installazione (Primi Passi)

Prima di procedere con l'installazione, assicurati di aver installato l'ultima versione di Google Chrome. In caso contrario, verifica di avere una versione almeno pari alla **135**. Puoi controllare la versione attuale di Chrome aprendo la pagina:

`chrome://settings/help`

Nel caso in cui l'aggiornamento di Chrome non sia possibile o la versione in uso sia inferiore alla 135, procedi ad abilitare manualmente le seguenti opzioni. Se invece disponi di una versione aggiornata, puoi passare direttamente al secondo step (A.1.2)

`chrome://flags/`

- `Privacy Sandbox Internal Pages`
- `Privacy Sandbox Privacy Policy`
- `Privacy Sandbox Ads API UX Enhancements`
- `Privacy Sandbox Equalized Prompt Buttons`
- `Privacy Sandbox Ad Topics Content Parity`

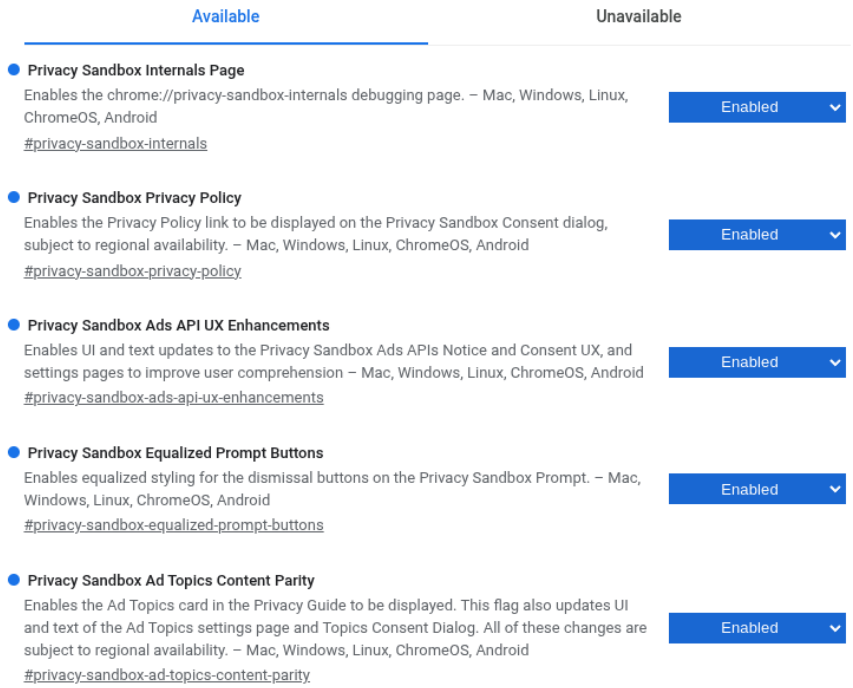

Figure A.1: Abilitazione delle opzioni Privacy Sandbox

A seconda della versione di Chrome installata, potresti visualizzare solo alcune delle opzioni elencate. Dopo averle abilitate, **riavvia Chrome** per applicare le modifiche e procedi con i passaggi successivi per installare l'estensione *Privacy Sandbox Monitor*.

Se non attivi queste impostazioni, l'estensione mostrerà una notifica di promemoria per invitarti ad abilitarle.

Figure A.2: Notifica per l'abilitazione delle opzioni Privacy Sandbox

A.1.2 Guida all'Installazione (Secondo Step)

Assicurati di aver completato correttamente il primo step prima di procedere.

Secondo Step Dopo aver terminato il passaggio precedente, abilita le seguenti impostazioni in Chrome:

- **Abilita le misurazioni della Privacy per gli annunci**: Vai su `Impostazioni > Privacy e sicurezza > Privacy per gli annunci`, oppure apri direttamente la pagina:

  `chrome://settings/adPrivacy`

- **Abilita i cookie di terze parti**: Vai su `Impostazioni > Privacy e sicurezza > Cookie di terze parti`, quindi imposta su:

  – "Consenti tutti i cookie", **oppure**

  – "Blocca i cookie di terze parti in modalità Incognito".

  Puoi accedere direttamente alla pagina tramite:

  `chrome://settings/cookies`

Installa l'estensione Segui questi passaggi per installare l'estensione Privacy Sandbox Monitor:

- Scarica il file ZIP `privacy-sandbox-monitor.zip`

- Estrai il contenuto del file ZIP in una cartella sul tuo computer.
- Apri Google Chrome e vai su `chrome://extensions/`.
- Abilita la "Modalità sviluppatore" attivando l'interruttore in alto a destra.

Figure A.3: Abilita la Modalità Sviluppatore

- Clicca sul pulsante "Carica estensione non pacchettizzata" e seleziona la cartella dove hai estratto il file

Figure A.4: Carica Estensione Non Pacchettizzata

A.1.3 Pagina di Benvenuto

Al termine dell'installazione, l'estensione aprirà automaticamente una nuova scheda con la pagina di benvenuto che fornisce una panoramica delle funzionalità e spiega la Privacy Sandbox con link alla documentazione ufficiale Google.

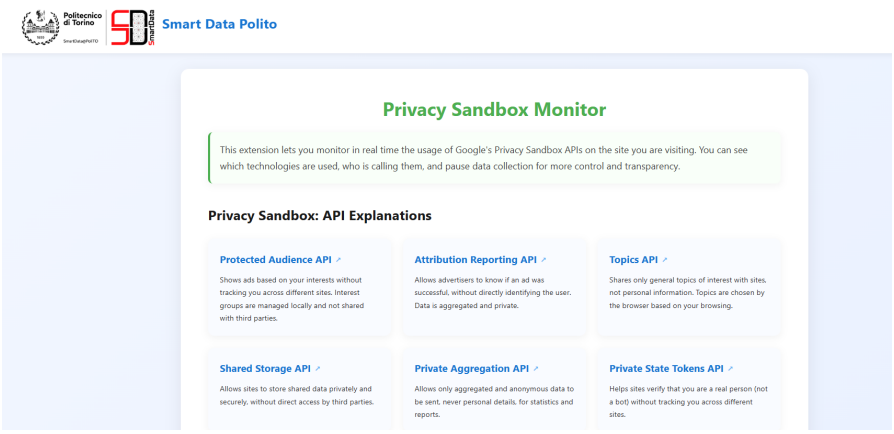

Figure A.5: Pagina di Benvenuto

A.1.4 Utilizzo dell'Estensione

Finestra Popup

Il modo migliore per utilizzare questa estensione è fissarla nella barra degli strumenti di Chrome, poi clicca sull'icona per visualizzare informazioni in tempo reale sulle attività della Privacy Sandbox.

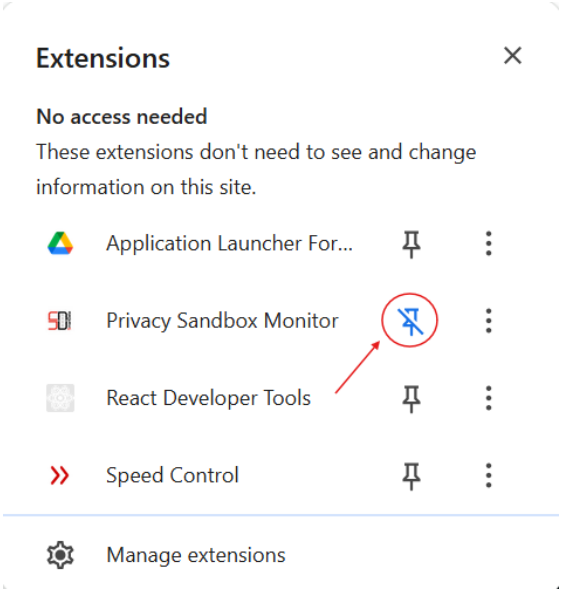

Figure A.6: Fissa l'estensione nella barra degli strumenti

La finestra popup contiene le seguenti sezioni:

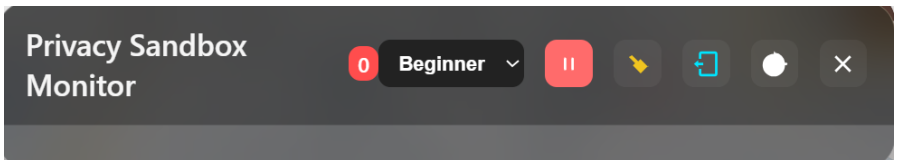

Figure A.7: Finestra Popup

- Il numero totale di chiamate API Privacy Sandbox effettuate dal sito web corrente.
- Il livello dell'utente (principiante o esperto), vedi la sezione successiva per maggiori dettagli A.1.4.
- Il pulsante pausa/play per interrompere o riprendere il monitoraggio. Quando è in pausa, l'estensione non registra alcuna APIs. La durata massima della

pausa è di 24 ore.

- Il pulsante per cancellare i log, che elimina tutte le APIs registrate nel sito web corrente.
- Il pulsante per esportare in CSV, che esporta tutte le APIs registrate nel sito web corrente in un file CSV.
- Il pulsante per passare dalla modalità chiara a quella scura.

Livelli Utente

L'estensione offre due livelli utente: principiante ed esperto. Il livello predefinito è principiante, ma gli utenti possono passare alla modalità esperto selezionando l'opzione corrispondente nella finestra popup. Il livello utente determina la quantità di informazioni mostrate sulle APIs.

- In modalità principiante, l'estensione mostra una vista semplificata delle APIs, visualizzando solo una breve descrizione delle chiamate con il bottone "Maggiori informazioni" per una migliore comprensione delle APIs e il chiamante di tali APIs (se disponibile).
- In modalità esperto, l'estensione fornisce una vista dettagliata delle APIs, includendo il nome completo dell'API e il chiamante di tali APIs (se disponibile).

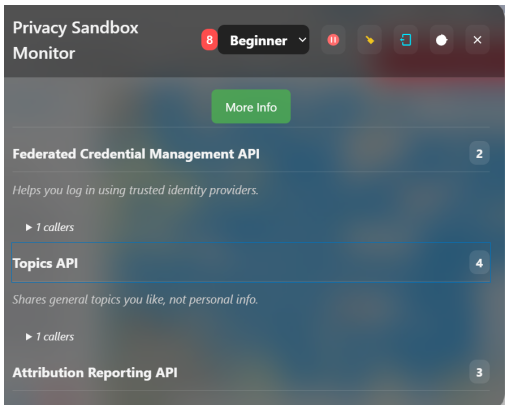

Figure A.8: Modalità Principiante

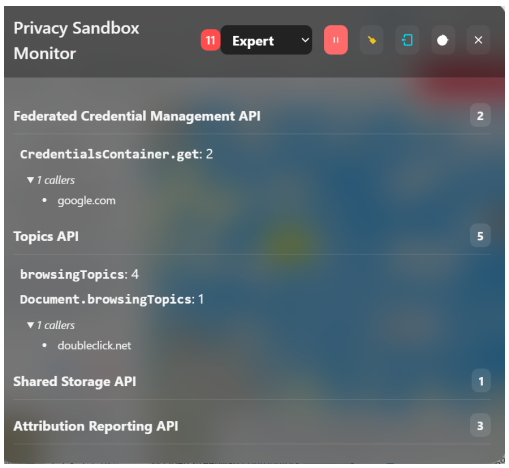

Figure A.9: Modalità Esperto

A.1.5 Visita la pagina demo per verificare il funzionamento

Per verificare il corretto funzionamento dell'estensione, puoi visitare la pagina demo all'indirizzo **https://protected-audience-demo-advertiser.web.app/**. Controlla che l'estensione rilevi una chiamata alle Protected Audience API, come mostrato nell'immagine sottostante.

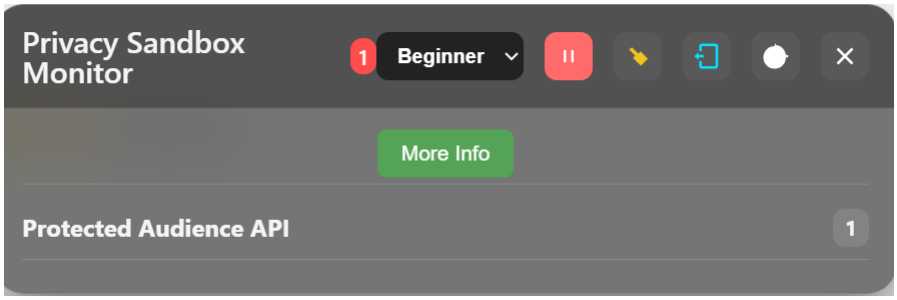

Figure A.10: Pagina demo

A.1.6 Verifica dell' integrità

Per assicurarti che il file ZIP non sia stato manomesso, puoi verificarne l'hash SHA-256 usando il sito web:

- `https://emn178.github.io/online-tools/sha256_checksum.html`
- Seleziona l'opzione `SHA256 file` dal menu Online Tools nella barra laterale sinistra.
- Trascina o seleziona il file `privacy-sandbox-monitor.zip`
- Confronta l'hash calcolato con il seguente valore:

```
7d1d375d444f1d1c5a602d644d7b55bfa6da3221e6fd8a1d6f387566c3c9c1dd
```

Se gli hash coincidono, il file è integro e sicuro da usare. Se non coincidono, non procedere con l'installazione e contatta il fornitore del file ZIP.

A.2 English Version

A.2.1 Installation Guide (First Steps)

Before proceeding with the installation, make sure you have installed the latest version of Google Chrome. If not, verify that your current version is at least **135**. You can check your Chrome version by opening the following page:

`chrome://settings/help`

If updating Chrome is not possible or your current version is lower than 135, proceed to manually enable the following options. If you are already using an updated version, you can skip this step and go directly to the **Second Step** (A.2.2).

`chrome://flags/`

- `Privacy Sandbox Internal Pages`
- `Privacy Sandbox Privacy Policy`
- `Privacy Sandbox Ads API UX Enhancements`
- `Privacy Sandbox Equalized Prompt Buttons`
- `Privacy Sandbox Ad Topics Content Parity`

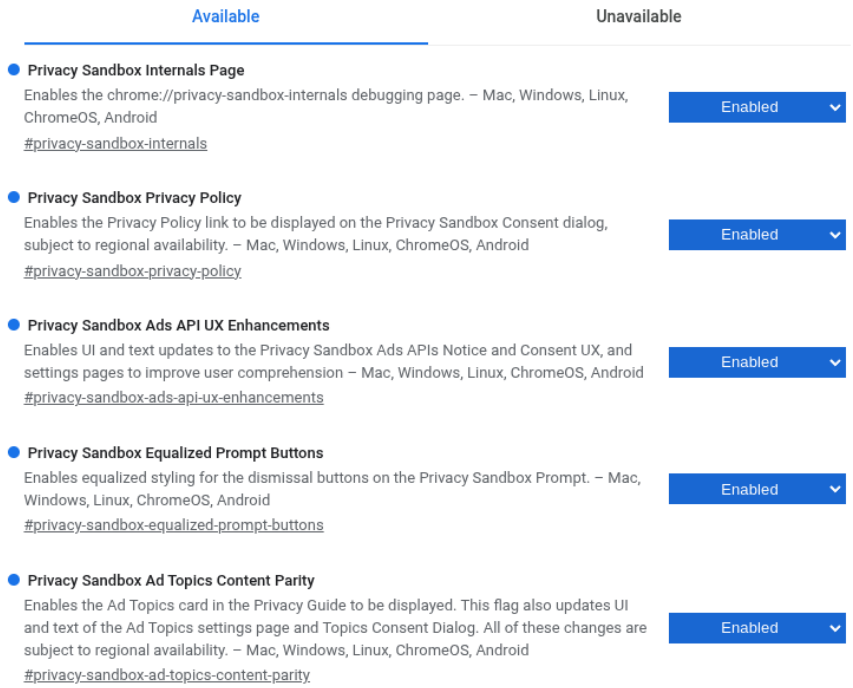

Figure A.11: Enable the Privacy Sandbox options

Depending on your Chrome version, you may see only some of the options listed above. After enabling them, **restart Chrome** to apply the changes and proceed with the next steps to install the *Privacy Sandbox Monitor* extension.

If these options are not activated, the extension will display a notification reminding you to enable them.

Figure A.12: Notification prompting to enable Privacy Sandbox options

A.2.2 Installation Guide (Second Step)

Make sure you have successfully completed the first step before continuing.

Second Step After completing the previous step, enable the following settings in Chrome:

- **Enable the ad Privacy measurements**: Go to `Settings > Security and Privacy > Ad Privacy`, or open the following page directly:

  `chrome://settings/adPrivacy`

- **Enable third-party cookies**: Go to `Settings > Security and Privacy > Third-party cookies`, then select:

  – "Allow all cookies", **or**

  – "Block third-party cookies in Incognito mode".

  You can also access the page directly at:

  `chrome://settings/cookies`

Install the Extension Follow these steps to install the *Privacy Sandbox Monitor* extension:

- Download the ZIP file `privacy-sandbox-monitor.zip`.

- Extract the contents of the ZIP file to a folder on your computer.
- Open Google Chrome and navigate to `chrome://extensions/`.
- Enable "Developer mode" by toggling the switch in the top right corner.

Figure A.13: Enable Developer Mode

- Click on the "Load unpacked" button and select the folder where you extracted the ZIP file.

Figure A.14: Load Unpacked Extension

A.2.3 Onboarding

Upon installation, the extension will automatically open a new tab with the onboarding page, which provides an overview of the extension's functionality and explains the Privacy Sandbox with links to Google's official documentation.

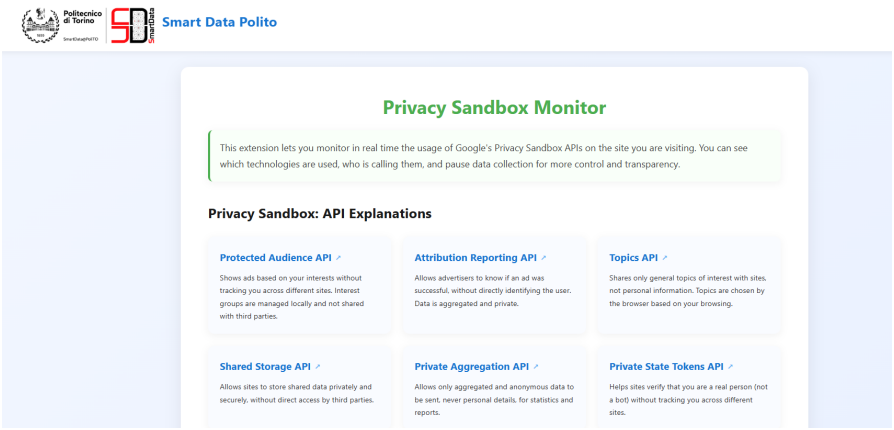

Figure A.15: Onboarding Page

A.2.4 Using the Extension

Popup Window

The best way to use this extension is to pin it to the Chrome toolbar. After that, click the icon, which displays real-time information about Privacy Sandbox API calls.

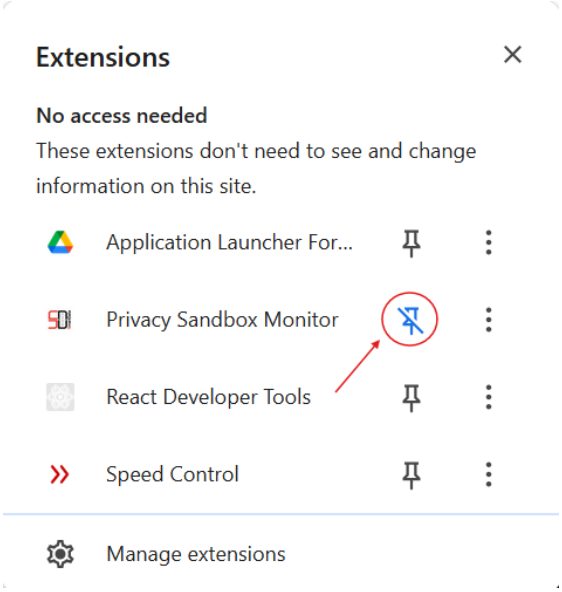

Figure A.16: Pin the extension to the toolbar

The popup window contains the following sections:

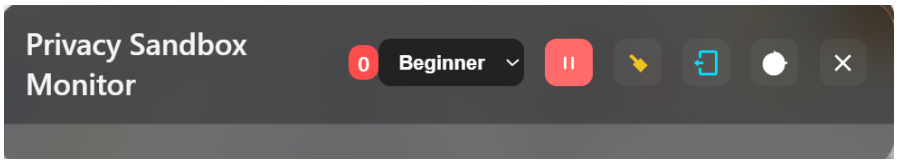

Figure A.17: Popup Window

- The total number of Privacy Sandbox API calls made by the current website.
- The current user level (beginner or expert), which determines the amount of information shown. For more details, see Section A.2.4.
- The pause/play button to stop or resume monitoring. When paused, the extension will not log any API calls. The maximum pause duration is 24 hours.

- The clear logs button to clear all recorded API calls on the current website.
- The export CSV button to export all recorded API calls on the current website to a CSV file.
- The toggle button to switch between light and dark mode.

User Levels

The extension offers two user levels: beginner and expert. The default level is beginner, but the user can switch to expert mode by selecting the expert option in the popup window. The user level determines the amount of information displayed about the API calls.

- In `beginner mode`, the extension displays a simplified view, showing only a brief explanation with a 'More Info' button that provides a better understanding of the Privacy Sandbox API and its caller (if available).
- In `expert mode`, the extension provides a detailed view of the API calls, including the exact JavaScript API name and the caller of that API (if available).

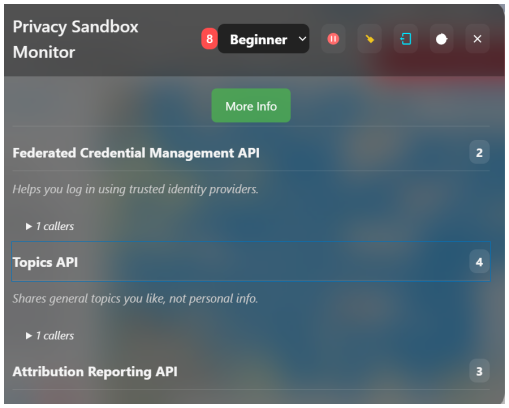

Figure A.18: Beginner Mode

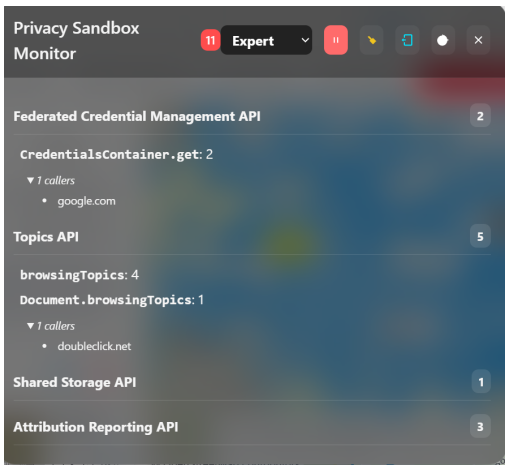

Figure A.19: Expert Mode

A.2.5 Visit the demo page to verify functionality

To verify that the extension is working correctly, visit the demo page at **https://protected-audience-demo-advertiser.web.app/**. Check that the extension detects a call to the Protected Audience API, as shown in the image below.

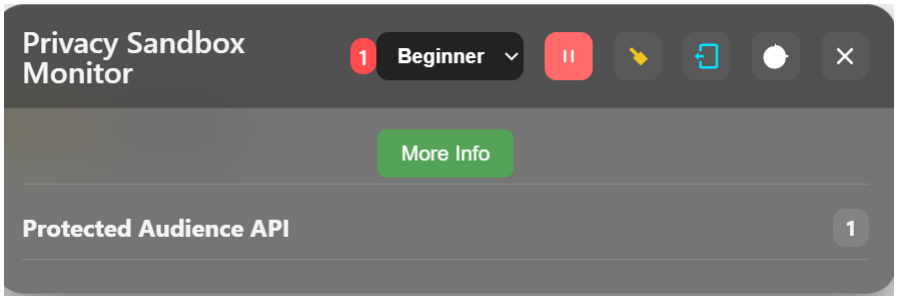

Figure A.20: Demo page

A.2.6 Integrity Check

To make sure the ZIP file hasn't been tampered with, you can verify its SHA-256 hash using a website:

- `https://emn178.github.io/online-tools/sha256_checksum.html`
- Select the `SHA256 file` option from the Online Tools menu on the left sidebar.
- Drag or select the `privacy-sandbox-monitor.zip` file.
- Compare the calculated hash with the following value:

```
7d1d375d444f1d1c5a602d644d7b55bfa6da3221e6fd8a1d6f387566c3c9c1dd
```

If the hashes match, the file is intact and safe to use. If they don't, do not proceed with the installation and contact the provider of the ZIP file.

Bibliography

[1] Anthony Chavez. *Update on Plans for Privacy Sandbox Technologies*. Accessed: 2025-10-20. Oct. 17, 2025. URL: https://privacysandbox.com/news/update-on-plans-for-privacy-sandbox-technologies/ (cit. on p. 1).

[2] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. *Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation*. 2018 (cit. on p. 2).

[3] European Union. *Regulation (EU) 2016/679 (General Data Protection Regulation)*. 2018 (cit. on p. 6).

[4] California Legislative Information. *California Consumer Privacy Act (CCPA)*. 2020 (cit. on p. 7).

[5] Apple Inc. *Intelligent Tracking Prevention*. https://webkit.org/blog/7675/intelligent-tracking-prevention/. Accessed: 2025-10-20. 2017 (cit. on p. 8).

[6] Apple Inc. *Full Third-Party Cookie Blocking and More*. https://webkit.org/blog/10218/full-third-party-cookie-blocking-and-more/. Accessed: 2025-10-20. 2020 (cit. on p. 8).

[7] Brave Software. *Brave Shields: How It Works*. https://support.brave.com/hc/en-us/articles/360022806212-What-is-Brave-Shields-. Accessed: 2025-10-20. 2024 (cit. on p. 8).

[8] Brave Software. *Privacy Features Overview*. https://brave.com/privacy-features/. Accessed: 2025-10-20. 2024 (cit. on p. 8).

[9] Mozilla Foundation. *Enhanced Tracking Protection in Firefox*. https://support.mozilla.org/en-US/kb/enhanced-tracking-protection-firefox-desktop. Accessed: 2025-10-20. 2019 (cit. on p. 8).

[10] Mozilla Foundation. *Firefox Privacy Features*. https://www.mozilla.org/en-US/firefox/privacy/products/. Accessed: 2025-10-20. 2024 (cit. on p. 8).

[11] Google LLC. *The Privacy Sandbox*. https://privacysandbox.com/. Accessed: 2025-10-20. 2024 (cit. on p. 8).

[12] Google Chrome Developers. *Preparing for Third-Party Cookie Phaseout*. https://developer.chrome.com/docs/privacy-sandbox/third-party-cookie-phaseout/. Accessed: 2025-10-20. 2024 (cit. on p. 8).

[13] Google Chrome Developers. *Protected Audience API (FLEDGE)*. https://developer.chrome.com/docs/privacy-sandbox/protected-audience/. Accessed: 2025-10-24. 2024 (cit. on pp. 9, 11, 13, 44).

[14] Google Chrome Developers. *Attribution Reporting API*. https://developer.chrome.com/docs/privacy-sandbox/attribution-reporting/. Accessed: 2025-10-24. 2024 (cit. on p. 9).

[15] Google Chrome Developers. *Topics API*. https://developer.chrome.com/docs/privacy-sandbox/topics/. Accessed: 2025-10-24. 2024 (cit. on p. 9).

[16] Google Chrome Developers. *Shared Storage API*. https://developer.chrome.com/docs/privacy-sandbox/shared-storage/. Accessed: 2025-10-24. 2024 (cit. on p. 9).

[17] Google Chrome Developers. *Private Aggregation API*. https://developer.chrome.com/docs/privacy-sandbox/private-aggregation/. Accessed: 2025-10-24. 2024 (cit. on p. 9).

[18] Google Chrome Developers. *Private State Tokens*. https://developer.chrome.com/docs/privacy-sandbox/private-state-tokens/. Accessed: 2025-10-24. 2024 (cit. on p. 9).

[19] Google Chrome Developers. *CHIPS: Cookies Have Individual Partitioned State*. https://developer.chrome.com/docs/privacy-sandbox/chips/. Accessed: 2025-10-24. 2024 (cit. on p. 9).

[20] Google Chrome Developers. *Fenced Frames*. https://developer.chrome.com/docs/privacy-sandbox/fenced-frames/. Accessed: 2025-10-24. 2024 (cit. on p. 10).

[21] Google Chrome Developers. *Federated Credential Management (FedCM)*. https://developer.chrome.com/docs/privacy-sandbox/fedcm/. Accessed: 2025-10-24. 2024 (cit. on p. 10).

[22] Google Chrome Developers. *Related Website Sets*. https://developer.chrome.com/docs/privacy-sandbox/related-website-sets/. Accessed: 2025-10-24. 2024 (cit. on p. 10).

[23] Nikhil Jha, Martino Trevisan, Emilio Leonardi, and Marco Mellia. «On the robustness of topics API to a Re-identification attack». In: *arXiv preprint arXiv:2306.05094* (2023) (cit. on p. 20).

[24] Nikhil Jha, Martino Trevisan, Emilio Leonardi, and Marco Mellia. «Re-identification attacks against the topics API». In: *ACM Transactions on the Web* 18.3 (2024), pp. 1–24 (cit. on p. 20).

[25] Yohan Beugin and Patrick McDaniel. «Interest-disclosing Mechanisms for Advertising are Privacy-Exposing (not Preserving)». In: *Proceedings on Privacy Enhancing Technologies* 1 (2024), pp. 41–57 (cit. on p. 20).

[26] Martin Thomson. *A Privacy Analysis of Google's Topics Proposal*. Tech. rep. Mozilla, 2023 (cit. on p. 20).

[27] Mário S Alvim, Natasha Fernandes, Annabelle McIver, and Gabriel H Nunes. «The privacy-utility trade-off in the topics API». In: *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2024, pp. 1106–1120 (cit. on p. 20).

[28] Alberto Verna, Nikhil Jha, Martino Trevisan, and Marco Mellia. «A First View of Topics API Usage in the Wild». In: *Proceedings of the 20th International Conference on emerging Networking EXperiments and Technologies*. 2024, pp. 48–54 (cit. on p. 20).

[29] Alberto Verna, Nikhil Jha, Martino Trevisan, and Marco Mellia. «Understanding Topics API in the Wild: Dubious Usage and Stale Adoption». In: *IEEE Transactions on Privacy* (2025) (cit. on p. 20).

[30] Garrett Johnson. «Unearthing Privacy-Enhancing Ad Technologies (PEAT): The Adoption of Google's Privacy Sandbox». In: *Available at SSRN* (2024). DOI: 10.2139/ssrn.4983927. URL: https://dx.doi.org/10.2139/ssrn.4983927 (cit. on p. 20).

[31] Minjun Long and David Evans. «Evaluating Google's Protected Audience Protocol». In: *arXiv preprint arXiv:2405.08102* (2024) (cit. on p. 20).

[32] Mir Masood Ali, Binoy Chitale, Mohammad Ghasemisharif, Chris Kanich, Nick Nikiforakis, and Jason Polakis. «z». In: *Network and Distributed System Security (NDSS) Symposium*. 2023 (cit. on p. 20).

[33] Giuseppe Calderonio, Mir Masood Ali, and Jason Polakis. «Fledging Will Continue Until Privacy Improves: Empirical Analysis of Google's {Privacy-Preserving} Targeted Advertising». In: *33rd USENIX Security Symposium (USENIX Security 24)*. 2024, pp. 4121–4138 (cit. on p. 20).

[34] Martin Thomson. *Protected Audience Privacy Analysis*. Tech. rep. Mozilla, 2024 (cit. on p. 20).

[35] Michiel Philipse, Güne Acar, and Christine Utz. «Post-Third-Party Cookies: Analyzing Google's Protected Audience API». MA thesis. Nijmegen, NL: Radboud University, 2024. URL: https://www.cs.ru.nl/masters-theses/2024/M_Philipse___Post-Third-Party_Cookies_Analyzing_Google's_Protected_Audience_API..pdf (cit. on p. 20).

[36] Badih Ghazi et al. «On the Differential Privacy and Interactivity of Privacy Sandbox Reports». In: *Proceedings on Privacy Enhancing Technologies* (2025) (cit. on p. 20).

[37] Alexandra Nisenoff, Deian Stefan, and Nicolas Christin. «Exploiting the Shared Storage API». In: (2025) (cit. on p. 20).

[38] Stephen McQuistin, Peter Snyder, Hamed Haddadi, and Gareth Tyson. «A First Look at Related Website Sets». In: *Proceedings of the 2024 ACM on Internet Measurement Conference*. 2024, pp. 107–113 (cit. on p. 20).

[39] Maximilian Zöllner, Anja Feldmann, and Ha Dao. «A First Look at Cookies Having Independent Partitioned State». In: *International Conference on Passive and Active Network Measurement*. Springer. 2025, pp. 182–196 (cit. on p. 21).

[40] Martino Trevisan, Nikhil Jha, and Antonino Musumeci. *Priv Accept*. 2020 (cit. on pp. 22, 23).

[41] SimilarWeb. *SimilarWeb top websites*. 2025 (cit. on p. 22).

[42] Privacy Sandbox Monitor. *Privacy Sandbox Monitor*. `https://forms.gle/Dg1i6mV6YBTDYR9h6`. Accessed 15 Nov. 2025. 2025 (cit. on p. 33).