Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH

Zero Knowledge Advertising: a new era of privacy-preserving AdTech solutions

Patricia Callejo*[1,2] | Rubén Cuevas[1,2] | Ángel Cuevas[1,2] | Mikko Kotila[3] | Luke Bragg[3] | Michiel Van Roey[3]

[1]Universidad Carlos III de Madrid, Spain
[2]UC3M-Santander Big Data Institute, Spain
[3]Profila GmbH, Switzerland

Correspondence
*Corresponding author Patricia Callejo,
Email: patricia.callejo@uc3m.es

Summary

The new data protection legislation, along with the social pressure, has encouraged the online advertising industry to propose a novel privacy-preserving advertising solution, such as Brave or Google's Topics. While these pioneering solutions represent an undoubtedly important step towards a more ethical form of targeted advertising, they present some limitations. In this paper, we first systematically identify the limitations of the most promising industrial solutions. Armed with such knowledge, we present a new (simple and efficient) privacy-preserving solution for delivering targeted ads, which combines the benefits of state-of-the-art proposals. To confirm the operational viability of our solution in the current online advertising ecosystem, we have performed extensive simulation experiments assessing that: 1) our solution requires minimal use of the device's resources (memory and CPU), making it valid for handheld devices running on limited power batteries; 2) it meets the strict timing requirements for the delivery of ads imposed in the online advertising ecosystem.

KEYWORDS:
Online Advertising, Privacy, Zero Knowledge

1 | INTRODUCTION

An essential distinction of online advertising is its capability to deliver behaviorally targeted ads. To exploit this advantage, online advertising stakeholders have developed a sophisticated tracking ecosystem able to track the activity of users online extensively. Some of the more questionable tracking practices in use have led to numerous scandals related to the potential misuse of users' personal data[1,2,3]. Such scandals have increased legal, and other societal pressure[4,5] on ecosystem companies and other actors, such as trade bodies, to force actors to rethink their business models and adopt privacy-preserving and ethically acceptable solutions.

In this regard, we can find many recent efforts by industry actors such as Brave browser[6], Google's Topics[7] or various Personal Data Platforms (PDPs)[8,9,10,11]. These solutions represent a clear step in the market towards more privacy-preserving ad delivery solutions. While these pioneering solutions represent doubtlessly important contributions to the field, they still have significant limitations. For instance, some of these solutions, such as the Brave browser, operate as walled gardens limiting their operation to specific websites and ad providers. Other approaches, such as Topics or PDPs, fall short in different ways. For example, where Google's Topics does not give users any control over their data, PDPs give users control over their data but do not connect the data meaningfully to the online advertising ecosystem. Besides, other more controversial solutions propose generating a unique and unified ID from personal data[12].

In this paper, we present a novel and comprehensive solution that tries to overcome the limitations of the existing state-of-the-art proposals. We rely on two main principles, simplicity and efficiency, and leverage basic cryptography technologies to propose the first version of our solution. The main functionalities of our solution are: 1) It can be readily integrated with all important online advertising channels (e.g., mobile apps, web browsers, video platforms, social media); 2) It offers two operation modes: (i) the default mode is zero knowledge advertising, where personalized ads are delivered to users without sharing any users' personal data with third parties. (ii) The second mode is referred to as *consent-based advertising*, and it requires the proactive action of the user to activate it. In this mode, the user explicitly selects what personal data items (if any) they are willing to share with each advertiser. To illustrate this mode, let us think of a scenario where a user is willing to share her height and weight with an advertiser A that sells clothes, her location with an advertiser B that sends last-minute restaurant discounts, and her interest in rock music with an advertiser C that promotes rock music concerts. Under this mode, an advertiser, with the user's consent, can keep track of the specific interactions of the user with its ads; 3) In our solution, users are rewarded for both their interaction (e.g., watching or clicking) with ads as well as for the data they voluntarily decide to share with advertisers; 4) The current Ad Tech industry builds profiles of users based on inference techniques that use for instance the browsing history of the user. The profiles resulting from these inference techniques have been proven inaccurate [13,14]. Also, privacy-preserving solutions like Brave or Topics stick to the use of such inference algorithms. Instead, our solution leverages self-declared information by users. This implies strong guarantees that ads delivered to a user are very likely aligned with their preferences; 5) The use of cryptography techniques generates irrefutable proofs of the actions performed by the involved stakeholders in our solution. In other words, our solution natively integrates auditing capabilities to provide safety to users, publishers, advertisers, and other Ad Tech stakeholders.

To the best of the authors' knowledge, there is no other privacy-preserving advertising solution offering the described set of features. Despite this, our solution does not aim at substituting existing ones but to provide people and advertising stakeholders with an alternative solution that aims to increase user privacy and, at the same time, improve the efficiency of advertising by targeting users with ads that are more relevant to them based on their self-declared information.

While conceptual solutions may be theoretically superior to the state-of-the-art, practical requirements make such solutions non-operational in practice. These practical requirements in online advertising can be roughly divided into two types. On the one hand, the online advertising market imposes a rigorous time restriction; the ad delivery process must be completed in the order of a few hundred milliseconds. On the other hand, ad delivery takes place mostly in handheld devices (e.g., smartphones), which have limited computational resources (e.g., CPU or memory) and run on limited-capacity batteries. The misuse of such resources may severely affect the user experience. To confirm that our solution is operational, we have run extensive simulations to measure the delay in the ad delivery process and the consumption of CPU and memory resources. Our results show that the ad delivery process is limited to tens of milliseconds for ads of standard size. Moreover, the CPU and memory consumption remains below 2.5% and 0.1%, respectively, in all considered scenarios.

2 | BACKGROUND

In this section, we first describe the most commonly used pattern of online advertising, programmatic advertising, and discuss the most apparent and inherent problems thereof. Moreover, we describe the most relevant privacy-preserving advertising solutions proposed by the industry in recent years. The research community has also contributed academic proposals in this field, e.g., [15,16,17,18,19,20,21,22], which to the best of the authors' knowledge have not been implemented as part of industrial/commercial solutions so far.

2.1 | Programmatic Advertising

Programmatic Advertising is the common term used to refer to the modern online advertising ecosystem. In programmatic advertising, ads to be shown in the publisher's venue (webpage or mobile app) is chosen in real-time from a pool of active ad campaigns. We distinguish two types of programmatic systems: *open-ecosystem* and *walled-gardens*. We explain the functionality of each of these systems next.

Open Ecosystem

Digital ads are shown in *publisher* venues (e.g., mobile apps or web pages) with ad spaces. A programmatic ad delivery process is triggered each time an ad space is available. Ad spaces are managed by the publisher's ad server, which (if available) loads a pre-configured ad in the ad space. Otherwise, it sells the ad either in a private or the open market[1]. Specifically, the publisher's ad server offers the ad space to an entity referred to as an ad exchange, sometimes also called Supply Side Platform (SSP), through an ad request. This ad request includes information about: 1) the ad space (the venue where it belongs, allowed type of ads, size, etc.); 2) the user (e.g., age, gender, location, interests, etc.); 3) Other related information. The information about the user is typically inferred through different tracking strategies (e.g., cookies, fingerprinting, etc.).

The ad exchange gathers the information from the ad request and translates it into an OpenRTB protocol's bid request[23], which is a typical scenario sent to one or more Demand Side Platforms (DSPs). The DSPs are technology platforms where advertisers (or their agencies) configure their ad campaigns. Advertisers, their agencies, and DSPs form the *buy side* in programmatic advertising, whereas ad exchanges and publishers form the *sell side* in programmatic advertising. Upon the reception of a *bid request*, a DSP checks if the information it includes about the ad space, the user, and the device matches any of its configured ad campaigns. If so, it replies with a *bid response* message, including a bid for the offered ad space. The ad exchange collects all the *bid responses* from different DSPs and conducts a real-time second-bid wins auction to select the winning bid. The winning DSP is informed with a *win-notice* message. The DSP then provides the URL from where the actual ad to be displayed can be retrieved. Ads are typically stored in an ad server controlled by the advertiser or the DSP. Upon the reception of the ad's URL, the user's device retrieves the ad and renders it visible on the corresponding ad space. The advertiser whose ad was delivered then pays a fee shared among all the involved players (publisher, ad exchange, and DSP).

The ad exchange gathers the information from the ad request and translates it into an OpenRTB protocol's bid request[23], which is a typical scenario sent to one or more Demand Side Platforms (DSPs). The DSPs are technology platforms where advertisers (or their agencies) configure their ad campaigns. Advertisers, their agencies, and DSPs form the *buy side* in programmatic advertising, whereas ad exchanges and publishers form the *sell side* in programmatic advertising. Upon the reception of a *bid request*, a DSP checks if the information it includes about the ad space, the user, and the device matches any of its configured ad campaigns. If so, it replies with a *bid response* message, including a bid for the offered ad space. The ad exchange collects all the *bid responses* from different DSPs and conducts a real-time second-bid wins auction to select the winning bid. The winning DSP is informed with a *win-notice* message. The DSP then provides the URL from where the actual ad to be displayed can be retrieved. Ads are typically stored in an ad server controlled either by the advertiser or the DSP. Upon the reception of the ad's URL, the user's device retrieves the ad and renders it visible on the corresponding ad space. The advertiser whose ad was delivered then pays a fee shared among all the involved players (publisher, ad exchange, and DSP).

Finally, note that the programmatic ad delivery process sets strict time constraints in the order of hundreds of milliseconds[24]. It is desired that the ad is rendered as soon as possible after the webpage or mobile app loading process is triggered, to maximize the marketing effect of ads.

Interested readers can find a more detailed description of the functionality of the open programmatic advertising ecosystem in[25].

Walled-Garden

We refer to walled gardens as those advertising platforms where a single player controls the full ad delivery process. Examples of relevant walled-garden players in the online advertising ecosystem are Google Search and Facebook.

The essence of the ad delivery process is the same as in the case of the open ecosystem. In particular, the walled-garden platform owns the venue where ad spaces are shown (e.g., Facebook's application), so they play an equivalent role to publishers in the open ecosystem. In addition, walled-garden platforms allow advertisers to configure their advertising campaigns based on the type of ad and the targeted audience, the device type, and other criteria, so they also play an equivalent role to DSPs in the open ecosystem when it comes to ad inventory on their own venue.

When an ad space is available, the walled-garden platform gathers information about the ad space, the user, and the device. It runs a real-time auction among those ad campaigns matching the collected data. The auction algorithm selects a winning ad campaign whose ad is delivered to the user. Walled gardens typically force advertisers to store ads in their own platform so that they play the equivalent role of advertiser ad server in the open ecosystem. The advertiser whose ad is delivered pays the fee to the walled-garden platform.

Privacy considerations

The online advertising ecosystem offers a clear advantage over traditional advertising channels like TV, press, or radio stations: *personalization*. People's online activity can be tracked and afterward processed to create or update a user profile focused on

[1] Both markets operate similarly. The only difference is that private markets are formed by a selected group of stakeholders.

revealing their preferences and interests. Conceptually, this is a powerful idea for advertising because knowing the interests of someone allows showing them ads aligned with their interests. However, the *obsession* for improving personalization has led to the development of a sophisticated and invasive tracking ecosystem mainly motivated by the digital advertising business model. The prevailing tracking and targeting practices of this ecosystem go further than just the online activity of users; with the proliferation of smartphones, users can be tracked concerning their physical mobility, places visited[26,27,28,29,30], and other aspects.

This sophisticated tracking ecosystem has led to various scandals[1,2,3] that have triggered a reaction by citizens, lawmakers, and others. On the one hand, despite the described tracking ecosystem, online advertising stakeholders still need to show people ads that are actually aligned with their interests[13,14], which translates into many users considering online ads annoying and useless. This user dissatisfaction, along with the perception of privacy intrusion, has led people to increasingly install ad blocker solutions[31,32]. On the other hand, the numerous scandals related to personal data misuse have led some administrations to develop modern data protection legislation to ensure that personal data is collected, stored, and processed under clear and strict conditions. Under these new regulations, some standard practices implemented in the Ad Tech ecosystem may be considered illegal. Examples of these regulations are the GDPR in the EU[4]; the CCPA in California[5]; the LGDP in Brazil[33]; POPI in South-Africa[34]; etc.

The described events have motivated/forced the Ad Tech ecosystem to take action and propose various privacy-preserving alternatives. Some of which we introduce in the following subsection. In addition to these solutions, many voices are asking to eliminate third-party cookies, which is currently the most widespread technique to conduct tracking and targeting online.

2.2 | Privacy Preserving Advertising Solutions

In this subsection, we present the most relevant privacy-preserving advertising solutions proposed so far: Passive Zero-Knowledge Advertising solutions (whose most prominent representative is Brave), Topics (proposed by Google), and Consent-Based Personal Data Platforms (PDPs).

2.2.1 | Brave: Passive Zero-Knowledge Advertising

Brave is a company whose main product[2] is the web browser with the same name. Brave browser was first released in November 2019.

The differentiating feature offered by Brave compared to major browsers is that it is a privacy-preserving browser. It blocks, by default, all ads, and third-party trackers without affecting the end-user online experience otherwise.

The business model of Brave is advertising. It offers its users an opt-in option to activate ads in Brave. Users opting in will receive ads. The way in which this works is that: 1) Brave compensates each individual Brave browser user for each delivered ad with its own token named *Basic Attention Token* (BAT); 2) the type of ads offered by Brave are non-invasive ads in the form of notifications that appear in the right upper corner of the screen. While significantly less intrusive than display or video ads, it is not clear the marketing efficiency of this type of ad.

Brave ads are targeted ads. The browser uses the websites a person visits to infer their interests and preferences. However, this information stays local in the browser and is neither shared with third parties nor even with Brave's own back-end. Instead, Brave collects a pool of ads from the ad campaigns available and sends them to the browser instances. Hence, matching the person's profile to the most suitable campaign is computed locally in the browser instance. This is a change to the current programmatic advertising paradigm in which the user profile is sent through several third-party platforms to reach DSPs where the match between the ad campaigns and the person's profile is executed (See Section 2.1).

Due to the described functionality, we classify Brave as a Passive Zero Knowledge Advertising (pZKA) solution. We consider it Passive since the user's profile is inferred by the browser without self-declared *active* inputs from the user.

Other relevant aspects to highlight from Brave in the context of this paper are the following:

1. Brave operates as a walled garden used as the venue to show ads on its browser.
2. Brave offers users the possibility of proactively (opt-in) deactivating the so-called shields that will allow: (i) trackers and third-party cookies to operate normally, (ii) users to receive regular ads. The user can enable this action for a specific website or for all websites. Users choosing this option will have a similar browsing experience to other browsers such

[2] Brave has recently released its second product, Brave Search, in beta mode[35].

as Google Chrome. Although this is possible, it may be complex for non-skilled users to set up this type of privacy configuration.

3. It cannot be considered a full Zero Knowledge Advertising solution since, in the standard operation of its current version, it still requires revealing the device's IP address in some cases[36]. Note that the GDPR has identified the IP address as Personal Data. Brave claims they do not record or share the IP address with third parties. Brave enables the use of IPFS[37], a p2p DHT-based solution that is still in an early phase of adoption, where little content can be accessed.

2.2.2 | Google's Topics

Google announced that Chrome, which accounts for roughly 2/3 of the browsers market share[38,39,40], would cease the use of third-party cookies[3]. This represents, in practice, the end of the third-party cookies, which has triggered an intense debate concerning targeted online advertising in the *post-cookie* era.

Google's initial proposal, referred to as *Federated Learning of Cohorts* (FLoC)[41,42], has been recently discarded and substituted by a new proposal referred to as *Topics*[7]. In this solution, the web browser (i.e., Google Chrome) computes the top 5 interests of a user every week. These interests are extracted from the user's browsing history, based on the categories assigned by Google to the different websites visited by the user. The interests from the last three weeks are stored. So that, when a user visits a given website, the Topics API will return 3 interests, one from each of the three previous weeks. The solution provides some features to enhance privacy. On the one hand, a third party (referred to as *caller* in the context of Topics) can only receive interest from a user if such a third party has observed the presence of that user in a website classified with such a topic. To clarify this point, let us consider the following toy example: a user U with *sports* as one of the assigned topics. A third party *T* concerning the website www.shoes.com can only receive U's interest *sports* if it previously saw U in a sports website (e.g., www.sports.com). On the other hand, a sixth random topic is assigned to a user every week, and with a probability of 5%, the random topic is returned.

While Topics offers clear and significant privacy improvements compared to the current cookie-based targeting approach, it cannot be considered to provide strong privacy guarantees. For instance, an actor able to fingerprint a user can collect the history of interests over a long period, establishing many interests associated with a user. Moreover, if multiple actors fingerprint a user, they can share the information they have learned about the user through a data-sharing process running in the background. On the other hand, Google should demonstrate the marketing efficiency of Topics. Some questions arise around this: are 3 topics sufficient to target a user properly?; what about demographic characteristics, such as gender or age?; how accurate is the website's topic classification algorithm?

2.2.3 | PDPs: Fine-grained consent-based advertising

Personal Data Platforms (PDPs) allow users to handle their personal data and decide which data and with whom to share it. In the context of online advertising, these platforms allow users to decide the actors they are willing to share data with and which specific data items can be shared. The result is a fine-grained consent-based form of advertising. PDPs typically offer people a user interface (e.g., mobile app) where people can configure their data-sharing preferences (e.g., which data share and with whom). Based on the users' configured privacy preferences, a PDP can offer audiences in bulk or individually (as it would occur in online advertising) to the advertisers providing all required data protection guarantees. Moreover, it is a common design choice among proposed PDPs to offer users explicit rewards in exchange for their data as, for instance, Brave does.

Fine-grained consent-based advertising and PDPs are recent concepts still being covered by research projects[11]. However, several start-ups have already proposed a PDP solution, e.g.,[8,9,10].

In consent-based solutions, the data the user allows is typically sent to a third party. This differs from other approaches, such as those based on access control, where the third party would get access to a repository where the data is stored using some credentials. The former presents fewer security risks since the actual location of data is not revealed, thus making it harder for potential attacks.

[3] Initially. The cessation was announced for the beginning of 2022, but in a later press release, Google postponed it to 2024.

	Opennes	Zero Knowledge	Consent Based	Active vs. Passive	Reward for users	Auditability
Brave	Walled-garden	✓		Passive	✓	✓
Google Topics	✓			Passive		
PDPs	?		✓(fine-grained)	Active	✓	
Our Solution	✓	✓	**✓(fine-grained)**	**Active**	✓	✓

Table 1 Summary of the features offered by state-of-the-art privacy-preserving advertising proposals and our solution (the symbol ✓ indicates the solution offers that property; the symbol "?" means that such solution may or may not provide that feature depending on the specific implementation.

3 | OUR SOLUTION

3.1 | Context

The existing privacy-preserving advertising solutions discussed above provide specific features that serve to pave the ground for the design of new solutions in the field. Using as reference such solutions, we have defined a set of features to frame the design of our solution and contribute a step forward in the context of privacy-preserving targeted digital advertising. In particular, the features we consider in the design of our solution are:

1. *Openness*: We have seen that privacy-preserving as well as traditional advertising solutions can operate either as walled gardens, where ads are shown in a venue controlled by the stakeholder (e.g., Facebook or Youtube) or in the open market, where ads are delivered in third party venues (e.g., Topics or open programmatic market). Hence, it is important to frame any newly proposed solution as *open* vs. *walled-garden*.

2. *Zero Knowledge*: Full privacy is a feature pursued by some new privacy-preserving advertising proposals (e.g., Brave) and requested by some privacy advocates and some societal stakeholders. In the context of this paper, we refer to these solutions as *Zero Knowledge* since they can implement targeted advertising without sharing users' information with any third party.

3. *Consent-based*: Some new privacy-preserving proposals are built on top of the concept of consent. While zero knowledge advertising offers full privacy to the user, it does not allow users to share their data with third parties voluntarily. Instead, consent-based solutions, such as PDPs, allow users to explicitly consent to which information can be shared with third parties. Zero-knowledge and consent-based are the two dominant concepts driving the design of novel privacy-preserving advertising solutions concerning shared data with third parties. Thus it is valuable to frame any new solution in the context of these two concepts.

4. *Active vs. Passive*: Another relevant aspect that drives the design of new privacy-preserving solutions is the manner in which the preferences/interests of users are learned. On the one hand, we refer to *passive* solutions as those relying on inference algorithms to obtain the user's preferences without their direct input (e.g., Google Topics or Brave). On the other hand, in *active* solutions, users take an active role and explicitly declare their preferences (e.g., PDPs). Any new solution should define the method used to collect the preferences/interests of users that allow for delivering personalized advertisements.

5. *User's compensation*: Many new proposed privacy-preserving advertising solutions argue that users should be compensated for receiving ads since they are a fundamental part of advertising revenue (e.g., Brave or PDPs). In contrast, solutions proposed by dominant players (e.g., Google Topics) seem to rely on the traditional incentive scheme in which users are compensated through access to services (e.g., search engine or maps service) rather than direct economic compensation. Therefore, any new privacy-preserving advertising proposal must define a model to compensate users as a key element of the advertising ecosystem.

6. *Auditability*: Data Protection Authorities and privacy advocates have requested the definition of proper auditing mechanisms, which help identify misbehaving advertiser stakeholders. Likewise, advertisers have been demanding the development of auditing mechanisms to assess the performance of their advertising campaigns. Hence, auditing is undoubtedly a growing demand in the advertising sector. New privacy-preserving advertising solutions, such as Brave, are aware of this and rely on cryptography to guarantee the auditability of their processes.

Table 1 summarizes the functionality offered by each of the discussed privacy-preserving solutions in Section 2 across the defined features.

3.2 | Design decisions

Using the 6 features introduced before as reference, our goal is to propose a solution that meets the following requirements:

- *Open* solution is able to deliver ads across any potential venue that operates (now or in the future) in the programmatic ecosystem (web, mobile apps, TV, out-of-home digital, etc.).
- *Active* solution where users explicitly declare their interest, instead of the system relying on machine learning-based inference algorithms[13,14] and other similar approaches.
- It should implement a combination of *Zero Knowledge* and fine-grained *Consent-based* advertising. In particular, it would operate a ZKA protocol by default. However, for those specific cases in which the user explicitly consents to share a specific set of data items with a specific third-party, such information will be shared with the indicated third party.
- It should implement a *compensation* scheme for users in order to share with them the economic benefit resulting from advertising.
- It should be *auditable*.

Table 1 shows the features of our solution and allows comparison with the state-of-the-art solutions introduced in Section 2.

We would like to highlight that the information presented in Table 1 should not be considered as a basis to argue a given solution is better than another. The solutions considered in the table must be understood as alternative solutions offering different capabilities, which can co-exist together (for instance, our solution is not affected by and can co-exist with the current auction-based programmatic advertising based on the openRTB protocol.)

In the context of this paper, the purpose of Table 1 is to show visually and simply that our solution is different from the known state-of-the-art privacy-preserving advertising proposals, both from a technical and functional perspective.

3.3 | Assumptions

We make some assumptions regarding concept definitions as well as functions that we include in our proposal based on solutions that already exist. Next, we describe them:

- **Privacy & Anonymity concepts**: In this paper, we use the concepts of privacy and anonymity in the context of the *zero knowledge*. This is, ads can be delivered to users without leaking any data from the user to any third party (privacy) and without revealing the user's identity to any third party (anonymity). An exception to these concepts is applied in the case of consent-based advertising, where users voluntarily reveal some information to a specific third party upon an agreement.
- **Registration Process**: A user in our system must be associated with a real person. To this end, during the registration process, the user must provide proof of their real identity (e.g., digital identity certificate or physical id document as it is common with the modern know-your-customer approaches)[43]. Alternatively, existing solutions such[44,45] can also be used in the registration process. Note that as a result of this process, an anonymous Avatar representing the user in the system will be generated. This avatar will be assigned a digital certificate with a public/private key pair. Note that a link between the actual user's identity and its avatar will be generated and stored in a repository along with a smart contract between the user and the entity running the system where the system's terms of use are defined. All this information is stored in a repository, for instance, a blockchain. While the anonymous avatar guarantees the anonymity of the user in the system, if there is evidence that a given avatar is breaking the terms of use, it is possible to trace back the identity of the real person linked to such an avatar in order to take the correspondent actions against the misbehaving person. Note that explaining the registration process further than what is described above in terms of the conceptual overview is out of the scope of the paper, and, as mentioned, is expected to be based on already existing solutions such[44,45].
- **Trust Model**: Our solution has three main players. The user, the company running the solution, and the advertiser. The considered trust model makes the following assumptions: 1) Advertisers are considered malicious and thus willing to

access and use any data learned from the user, not paying users for views or clicks of ads, etc. However, in our trust model, advertisers do not collude, i.e., they do not share information with each other. This is a quite realistic assumption since, in the real world, it is unlikely that two advertisers, which are competitors, share information between them; 2) The focus of this paper is guaranteeing users' privacy in online advertising, and thus we consider users to follow the defined protocol and not to misbehave. Note that other trust models in which users misbehave, specifically by committing ad fraud, are left for future work; 3) We consider the company running the solution to be partially malicious, i.e., it tries to learn the identity and data from the user, but follows the protocol and do not cheat on payments. This company does not collude with advertisers or users to perform malicious practices.

- **Non-repudiable proofs**: Each time a transaction between a user and a third party occurs, cryptographic non-repudiable proofs are generated. These provide auditing guarantees for each transaction. For some of these transactions, specific implementations of our solution may consider the use of smart contracts. In this case, we again propose to use existing technology that solves this problem, e.g., Smart Contracts on top of Ethereum[46] or[47]. Note that non-repudiable proofs aim at proactively persuading different players (e.g., malicious users or advertisers) to avoid them systematically misbehaving. Such systematic misbehavior can be unveiled at any point in time with non-repudiable proof.

- **Auditing process**: In this paper, we describe how to create auditable transactions so that any implementation of our solution is subject to auditing not only by the involved stakeholders but also by third parties. The details of where the auditable proofs are stored (most likely a blockchain) and how the auditing process can be implemented are out of the scope of this paper.

- **Compensation to users**: As indicated above, our solution is expected to reward users for their interaction with ads. In particular, the company running our solution will compensate people for sharing their data with advertisers as well as for their interaction with ads. In principle, we assume that this compensation will be done through a cryptocurrency (e.g., ADA, BAT, etc.), but other options could be implemented as well (e.g., Amazon or Netflix gift card).

- As for the rest of the literature, in the context of Zero Knowledge Advertising and Consent-based Advertising, we do not address the issue of advertising fraud in this paper. However, it is worth noting that our solution significantly limits the ability of an attacker to commit fraud. As described above, each avatar in our solution must be linked to a unique real-world identity. Hence, the number of accounts an attacker can create in the proposed system is limited to the number of real-world identities it has access to, which is likely to be reduced to a few accounts. This significantly increases the difficulty of creating botnets or large-scale attack infrastructures. Studying in detail fraud aspects is left for future work.

3.4 | Involved Players

In this subsection, we describe the main stakeholders considered in our solution. Figure 1 shows a scheme representing the described components and their interactions.

- *Avatar*: This is the user's representative within our solution. An avatar must be linked to a real person's identity, but no personal data is associated with the Avatar. Each Avatar is assigned a certificate with an associated public/private key pair. The private key of an Avatar will be used to sign the proofs generated for the transactions associated with an Avatar. This guarantees the non-repudiability of performed actions by users. Moreover, in order to encrypt communications, we propose to use modern solutions such as hierarchical deterministic keys[48,49], which allow generating of an arbitrary number of public/private key pairs from a seed public/private key pair (i.e., the keys associated to the Avatar's certificate). This technique is used to create hierarchical deterministic crypto-currency wallets[50].

- *Ad Delivery Software*: As described above, we envision an open solution that can operate in any third-party publisher willing to show ads in the online advertising ecosystem. Therefore, our solution will be implemented in specific software that third parties can integrate. In particular, in the mobile ecosystem, the ad delivery software is expected to be deployed in an SDK to be integrated by mobile apps, whereas in desktops, the ad delivery software can be either integrated within the web browser or implemented as a web browser extension.

- *Publisher*: The publisher is the owner of the venues offering ad spaces, specifically mobile apps, web pages, or video platforms. The publisher will integrate the ad delivery software described above, and this one will choose the ads to be shown in the publisher's venue.

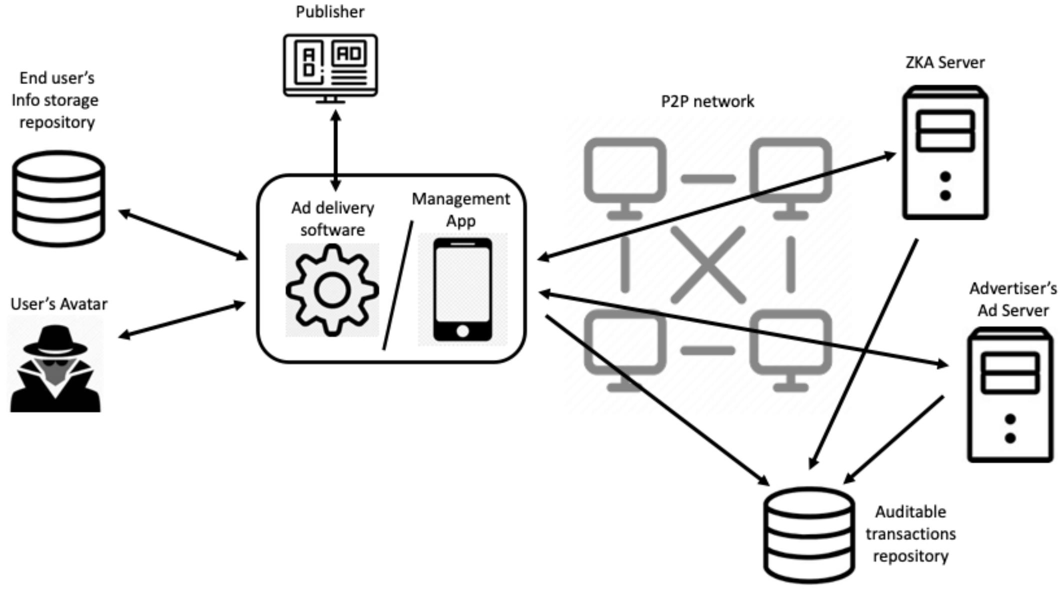


Figure 1 Scheme of our solution including the components and the interaction among them.

- *Advertiser ad server*: In current programmatic advertising, ad campaigns are configured in DSPs. Moreover, ads are served from advertisers' ad servers or, if chosen by the advertiser, from its DSP's ad server. For simplicity, in this paper, we consider a single entity, referred to as the Advertiser ad server, that will take care of all the Advertiser-related interactions. Note that, in a specific implementation of our solution, the functions offered by this Advertiser ad server can be divided across different entities. While this might impose some practical considerations, it does not impose any conceptual or design limitations.

- *Zero Knowledge Advertising Server*: A company using our solution will operate what we refer to as a Zero Knowledge Advertising (ZKA) Server. This server will serve as an intermediary to distribute the available ad inventory reported by advertisers to the *ad delivery software* installed in the user's device. It will receive only the data from the ad delivery software required to implement the billing and accounting process and will not receive any data that could reveal the user's identity. This is why we refer to it as a *zero knowledge* advertising server.

- *P2P network*: Our solution deploys a P2P network formed by users that voluntarily indicate their willingness to participate in such network. This P2P network guarantees that all communications started by the user's avatar A in device D are routed through a peer from the P2P network, acting as a relay toward its final destination. By doing so, we avoid any third party (ZKA Server Provider or the Advertisers Ad Servers) from knowing the actual IP address of the device involved in the communication. We remind you that the IP address is considered personal data by the GDPR. Note that to be selected to relay a communication, a peer must meet a number of conditions: 1) be connected to a high-speed connection (preferably a fixed fiber optics connection) so that the device acting as relay has a low bandwidth overhead; 2) be located topologically close to the device originating the communication to minimize the impact on the communication delay; 3) have a minimum predefined memory and CPU capabilities. Indeed, desktop computers will be preferred over mobile devices to be part of the p2p network.

  An alternative to the P2P network, proposed by Google's privacy sandbox or the IAB Tech Lab[51], would be the utilization of cloud services (e.g., VPN services) so that the messages from a device D are routed through a proxy provided by the cloud service. This option would also hide the IP address of D. However, in this case, the cloud provider providing the proxying service would be able to record all communications from all users. This cloud provider would know which ad servers have accessed each IP address. Hence, the anonymity of the IP address would be, to some extent, exposed to the cloud provider. Instead, the use of the P2P network allows each communication started by a device to be relayed through a different peer. Hence, deanonymizing the activity of a given IP address would require a collusion attack from several nodes within the P2P network. Even if this happens, this collusion attack would be able to cover just a subset of the devices in the system, having access to just partial information from an IP address activity. Based on this, we believe there is no other privacy-preserving advertising solution providing such a level of anonymity for the IP address of a user. Note that in the early phase of the deployment of the proposed system, where the number of registered users is still low, the option of using cloud providers may be recommendable until the user base is sufficiently large to run the P2P-based proxying service.

- *End-users' info storage repository*: End user's data is stored in a storage repository in the cloud where all the info provided by the user, as well as their transactions in the ad ecosystem (through the ad-delivery software), are stored. This repository is protected by a password, and only the user can access it. Note that other access schemes might also be considered, e.g., based on cryptography certificates. We chose password-based protection due to its extensive use and familiarity, even for non-skilled people.

- *Auditable transactions repository*: All executed transactions are cryptographically signed by the involved players, and the signed probes are stored in this repository, which can be implemented using blockchain and smart contracts technology. Since all the players involved in a transaction provide undeniable proof of their agreement on the execution of such a transaction, our solution offers no-reputability guarantees by default.

- *Management App*: This is the main app of the company implementing our solution. This app is the interface that allows users to: 1) configure their Avatar; 2) grant/revoke access to data to specific third parties to implement consent-based advertising; 3) access the list of transactions executed through the advertising platform (ads received, clicks on ads, etc.) and the associated cryptographical probes; 4) have access to its associated wallet to assess the compensations received for their participation in the advertising platform. The management app is also synchronized with the end user's info storage repository. Hence users can manage their information from multiple devices. Indeed it is expected that the management app is instantiated in the form of a mobile app (so the user can use it from their mobile devices) as well as a web app (so the user can use it from any browser).

3.5 | Protocol Description

In this section, we describe the protocol that handles the different operations to grant the desired functionality, i.e., a combination of zero knowledge and fine-grained consent-based advertising guided by the explicit instructions of the user.

We first describe the Zero Knowledge Ads delivery protocol that is the default mode of operation of our solution. Afterward, we explain the processes involved in the fine-grained consent-based ads delivery protocol: 1) the consent granting phase, in which users can grant consent to an advertiser to access some pieces of their data; 2) the consent revoking phase, where users can revoke consent to a given advertiser to access their data; 3) the consent-based ad delivery process. Finally, we discuss the billing&accounting protocol.

3.5.1 | Zero Knowledge Ads delivery

For simplicity, to describe the ZKA ad delivery, let us assume that the ad delivery software used is an SDK installed in a mobile app.

Figure 2 shows the exchange of messages between the different entities involved in the process. Note that, unless other way stated, each message is forwarded through a different peer that is part of a p2p network so that the actual IP address of the user's device is never exposed to third parties (which includes the ZKA server operator and advertisers). In addition, to encrypt communications with third parties, the device (through the ad delivery software or the management app) uses a randomly selected public/private key pair from a large pool of keys generated using a hierarchical deterministic key system. Therefore, each message is encrypted with a different key. These keys do not need to be linked to a certificate since their purpose is encrypting messages and not proving the identity of the device involved in the communication. Note that using a single public key to encrypt all messages from a device would make such a public key an unique identifier of the device. For the sake of readability, we avoid explicitly mentioning this in every step of the protocol. Next, we describe step by step the message exchange depicted in Figure 2.

1. The ZKA Server receives from each Advertiser Ad Server an `AD_LIST` message with a list including for each ad: an *ad id* and the *ad's metadata*. The ad's metadata consists of the following information: the type of ad (e.g., display, video), the size of the ad, the target audience (age, gender, interests, etc.), the IP address or URL of the advertiser ad server

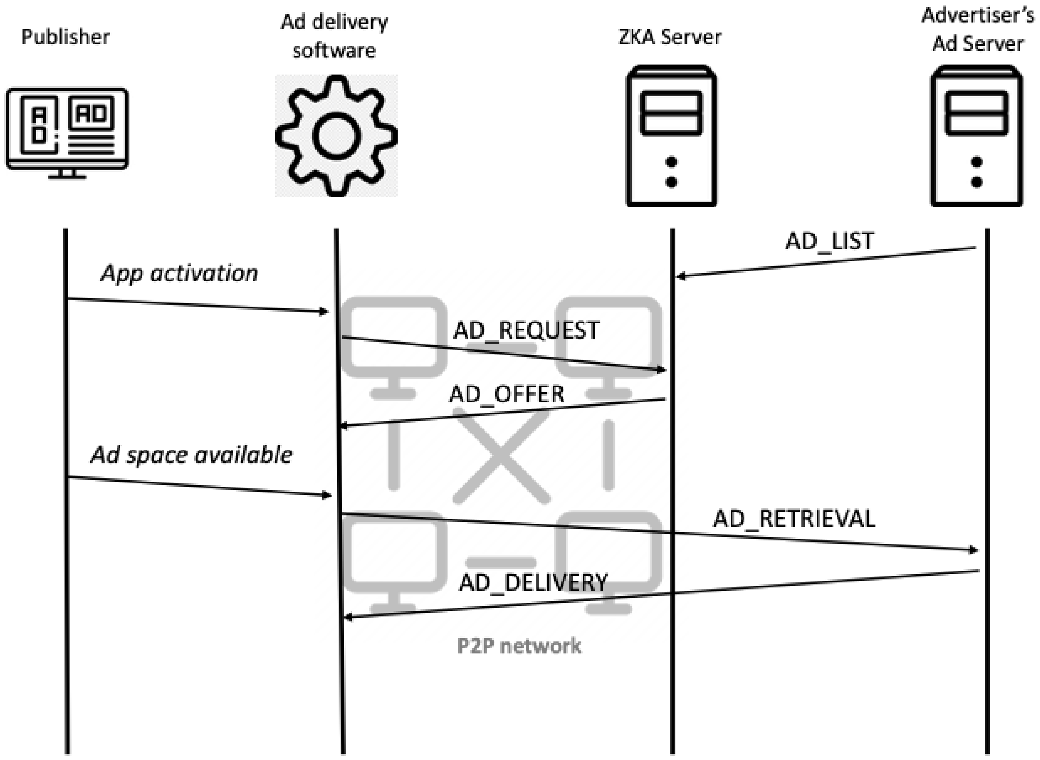


Figure 2 ZKA protocol overview.

from which retrieve the ad, the public key of the advertiser ad server, an expiration time (the deadline to deliver this ad to users), the compensation associated with the different events related to an ad impression (e.g., a view or a click), etc. The ZKA Server collects the meta information of the ad inventory from potentially thousands of advertisers. It is part of the business strategy of the company owning the ZKA Server to decide how to process this ad inventory by aggregating it all together or dividing it into groups depending on different classification criteria (e.g., the historical performance of ad inventory, preferential agreements with certain advertisers, etc.). This is similar to the strategies defined nowadays by Ad Exchanges in the current digital advertising ecosystem. In addition, advertisers are not subject to any new risk. In the current programmatic advertising operation, advertisers share their bid prices with DSPs and AdXs. In our solution, they would share the value of the economic compensation with the ZKA Server and (as we will see) with the ad delivery software.

2. Upon activating the mobile app embedding the ad delivery SDK software, the software creates an `AD_REQUEST` message sent to the ZKA Server. The only parameter included in this message is a public key.

3. Upon the reception of the `AD_REQUEST` message, the ZKA Server generates an `AD_OFFER` message formed by a list including the ad id and ad meta-information for a pool of ads selected by the ZKA Server. The `AD_OFFER` message is encrypted with the public key provided by the ad delivery software in the `AD_REQUEST` message. By doing so, only the ad delivery software will be able to access the list of received ads. Note that the ZKA Server receives neither user's data from the Ad Delivery software nor the IP address that is hidden through the p2p network. Finally, using proper encoding (See Section 4), metadata from thousands of ads can be included in a single `AD_OFFER`. This provides a high probability that ads of interest to a user are included in the list.

4. Once the list of ads is available in the ad delivery software when an ad space is available in the mobile app, the software triggers the process to retrieve an ad to fill the ad space. Note that defining the ad selection algorithm is out of the scope of this paper. However, the ad selection algorithm should be designed based on modern machine learning methods to maximize the probability of the user interacting with the ad. To this end, the selection algorithm should consider the user interaction with ads in the past, the user's preferences, as well as the information regarding the target audience included in

the ads' meta-information. For instance, the algorithm could first select those ads for which the required audience match the profile of the user (demographic properties and interests) as well as the user location (obtained by the ad delivery software from the available location APIs offered by the device OS based for instance in GPS data or GeoIP). Second, the algorithm would rank them using machine learning methods, which use as input the past interactions of the user with ads from every advertiser id, so those ads from brands with which the user has interacted most in the past are ranked first. Note that all the referred information is available in the ad delivery software. Finally, if the algorithm does not find any match among the list of ads, it will send a new `AD_REQUEST` message to gather a new list of ads.

5. Once the algorithm selects an ad, the ad delivery software retrieves the IP address of the advertiser ad server from which retrieves the ad (directly from the ad meta-information or through a DNS resolution of the URL). The ad delivery software sends an `AD_RETRIEVAL` message to the advertiser ad server. This message includes the following data: ad ID encrypted with the public key of the advertiser ad server (by doing so, even an attacker intercepting the message would be unable to know the requested ad), a transaction ID (this will be used as a unique reference of this ad transaction) and a public key associated with the user's avatar certificate.

6. Upon the reception of the `AD_RETRIEVAL` message, the advertiser ad server responds with an `AD_DELIVERY` message, which includes: the ad encrypted with the public key provided by the ad delivery software in the correspondent `AD_RETRIEVAL` message and the transaction ID of the correspondent `AD_RETRIEVAL` message.

7. Finally, the ad delivery software places the ad in the corresponding ad space.

The described process depicts the basic functionality scheme, subject to different improvements/modifications. For instance, the ad selection algorithm could be executed in the background to create a predefined sorted list of ads so that the ad delivery software can prefetch a number of ads (e.g., 4 ads). Upon an ad slot is available, the ad delivery software would immediately deliver one of the prefetched ads to the mobile app for showing it to the user. Using ads prefetching, the ad delivery delay would be negligible.

3.5.2 | Fine-grained consent-based Ad delivery

In this subsection, we provide the details of our protocol for: 1) allowing the user to grant consent to an advertiser to access and process certain user data. We envision a process driven by the advertisers. An advertiser makes an offer in which it indicates the data it requires from the users as well as the compensation it is willing to deliver for the data; 2) allowing the user to revoke the consent to an advertiser; 3) deliver consent-based ads.

Note that from the user's perspective, the consent granting and revoking processes involve the management app, whereas the consent-based ad delivery process requires the participation of the ad delivery software instead.

As in the case of ZKA delivery, unless otherwise stated, every message sent from the management app or the ad delivery software is forwarded through the p2p network, so the IP address of the user's device is hidden.

Consent Granting

Figure 3 shows the exchange of messages we propose to govern the consent-granting process that we describe next.

1. The advertiser ad server sends to the ZKA server a `CONSENT_GRANTING_OFFER` including: the data requested from the user (e.g., age, gender and location or age, gender, and top interests); the offered compensation in the selected crypto-currency; the advertiser-ID and its URL (in case the user want to learn more from the advertiser); the IP address of the advertiser ad server; the offer ID; a public key associated with the advertiser. Obviously, an advertiser may send more than one offer. The ZKA server collects the `CONSENT_GRANTING_OFFERS` from multiple advertisers.

2. Upon the activation of the Management App, it sends an `ADVERTISERS_OFFER_REQUEST` to the ZKA server. This message only includes a public key associated with the user's avatar.

3. Upon the reception of the `ADVERTISERS_OFFER_REQUEST`, the ZKA server sends to the Management app an `ADVERTISERS_OFFER_REPLY` including the information of a pool of advertisers' offers. Note that the algorithm to select which offers to send to each request is out of the scope of the paper. For simplicity, we consider that all offers available at the ZKA server are included in every `ADVERTISERS_OFFER_REPLY` message. The message is encrypted with the management app's public key so that even if an attacker intercepts the message will not be able to read or modify it.

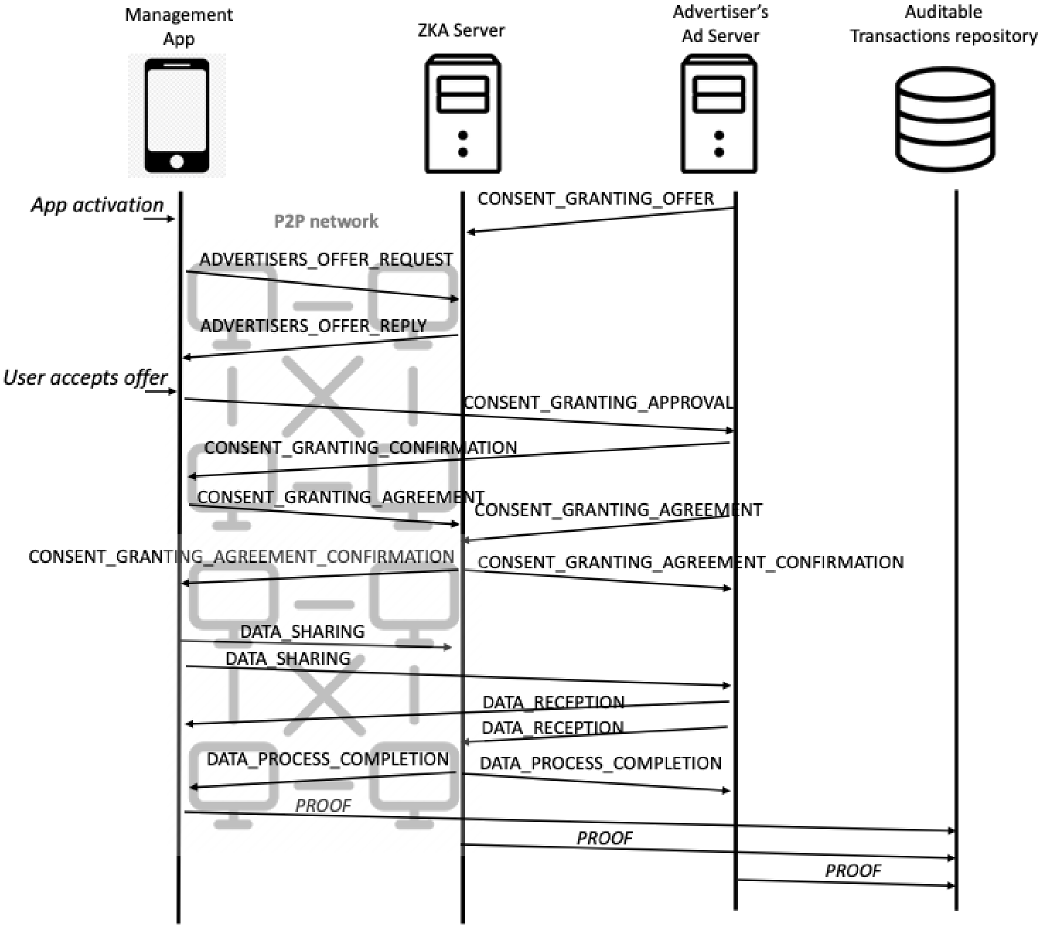


Figure 3 Consent-granting protocol overview.

4. The management app processes the ADVERTISERS_OFFER_REPLY message and presents the different offers in a visual form to the user. If the user accepts an offer, the management app generates a CONSENT_GRANTING_APPROVAL message, which includes the following information: transaction ID (this is a unique ID of the transaction); a tuple formed by the offer-ID, advertiser-ID, user-ID and compensation encrypted with the public key of the advertiser. Note that the user-ID is a random number generated solely for the relation between this advertiser and the user. Note that the user-ID used for different advertisers is different and is not linked to any PII data. Moreover, the management app generates undeniable proof of the consent granted by the user. To this end, it generates a token, a version of the previous tuple (transaction-ID, offer-ID, advertiser-ID, user-ID, and compensation) signed with a private key from the user's avatar. The CONSENT_-GRANTING_APPROVAL is sent to the advertiser ad server.

5. The advertiser ad server, upon the reception of the CONSENT_GRANTING_APPROVAL message, responds with a CONSENT_-GRANTING_CONFIRMATION message. This message includes: the transaction-ID as well as a token which is the signed version of the tuple introduced above (transaction-ID, offer-ID, advertiser-ID and compensation). It is signed with a private key from the advertiser. The confirmation message is sent to the Management App.

6. At this stage, and before proceeding with the data sharing, both entities, the Management App and the advertiser's Ad Server, send a CONSENT_GRANTING_AGREEMENT message to the ZKA Server. This message includes the transaction ID and the compensation information. The Management App signs it with a private key from the user's avatar, and the advertiser ad server signs it with a private key from the advertiser.

7. Upon the reception of both CONSENT_GRANTING_AGREEMENT messages, the ZKA server sends a CONSENT_GRANTING_-AGREEMENT_CONFIRMATION message to both the Management App and the advertiser ad server. This message includes

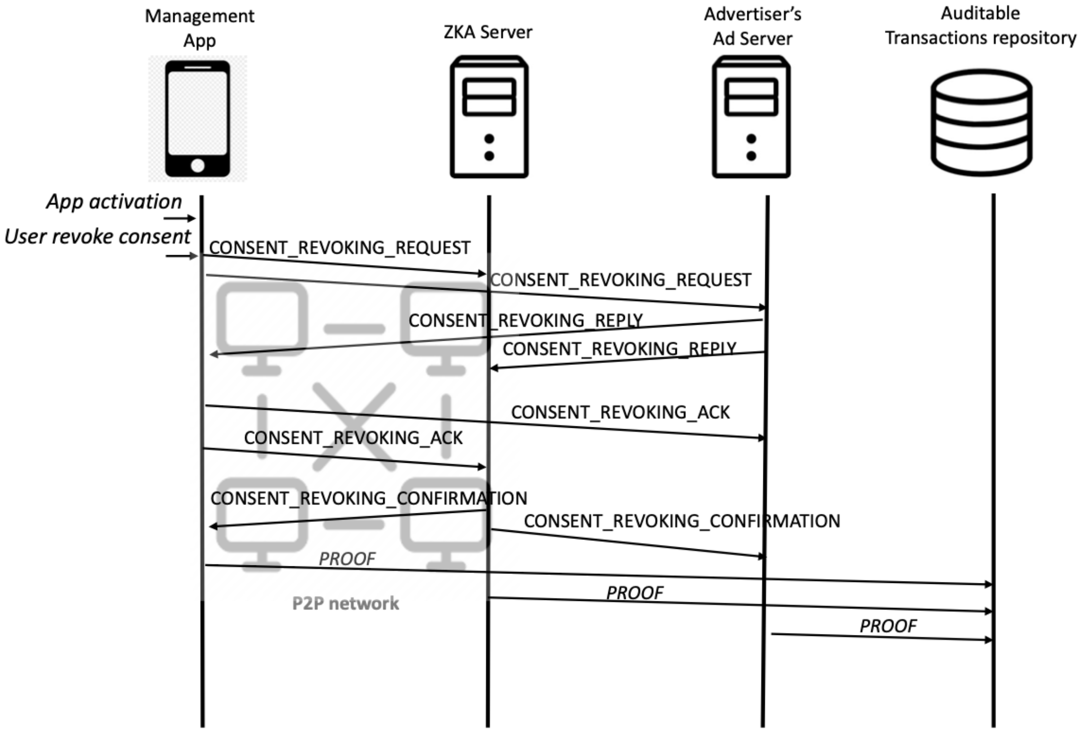


Figure 4 Consent-revoking protocol overview.

the transaction ID and compensation information signed with the company's private key running the ZKA server. At this point, the three involved parties have generated undeniable proof that they are aware of this transaction and approve it.

8. On the reception of the `CONSENT_GRANTING_AGREEMENT_CONFIRMATION`, the Management App sends a `DATA_SHARING` message to the advertiser ad server, that includes the data agreed to be shared with the advertiser. This message includes the transaction ID and the shared data encrypted with the public key of the advertiser ad server. Therefore, only the advertiser would be able to access the shared data. Finally, this message is also signed with the user's avatar's private key, so the user is providing undeniable proof that they shared such data. Hence, if the user is faking the shared data, there will be evidence that can make the user accountable for faking the data. Moreover, the Management App sends a replica of the `DATA_SHARING` to the ZKA server without including the encrypted data to inform the ZKA server that the data has been sent to the advertiser's ad server.

9. Upon the reception of the `DATA_SHARING` message, the advertiser answers with a `DATA_RECEPTION` message informing that the data has been received. This message includes the transaction ID signed with the private key of the advertiser's ad server. This message is sent to both the Ad Manager app and the ZKA server.

10. Upon the reception of both `DATA_SHARING` and the `DATA_RECEPTION` messages, the ZKA server sends a `DATA_PROCESS_COMPLETION` message to both the Management App and the advertiser ad server. This message includes the transaction ID and is signed with the private key of the ZKA server.

Once the described protocol has been executed, the three entities involved in the process have agreed on the existence of the transaction and the completion of the data sharing and have generated a signed proof of their agreement. We have described an option in which the undeniable proofs consists of cryptographic proofs through signed messages. Note that these proofs are signed with the private keys associated with each entity's certificates, which provides non-repudiability guarantees. As we describe later in the paper, these proofs are stored in the *auditable transactions repository*, e.g., a blockchain. However, other alternative approaches, such as smart contracts, can also be used in the described protocol with minor modifications.

Revoking consent

Figure 4 graphically shows the exchange of messages that allows users to revoke the consent granted to an advertiser to use their data.

1. When the user indicates their willingness to revoke the consent granted to an advertiser through the Management app, this one generates a `CONSENT_REVOKING_REQUEST` message. This message includes the transaction ID associated with the corresponding consent-granting process, identifying the specific consent instance to be revoked. The message is signed with a private key from the user's avatar certificate, and it is sent to both the advertiser ad server and the ZKA server.
2. Upon the reception of the `CONSENT_REVOKING_REQUEST` message, the advertiser ad server sends a `CONSENT_REVOKING_REPLY` message to the Management app and the ZKA server. This message is signed with the private key of the advertiser. In the `CONSENT_REVOKING_REPLY`, the advertiser ad server also includes the data it is storing from the user and is planning to remove. This information is encrypted using the public key of the user's avatar, so only the Management app can read it.
3. The Management app verifies whether the data to be removed is correct (it corresponds to the shared data in the consent granting process) and, if that is the case, sends a `CONSENT_REVOKING_ACKNOWLEDGMENT` to both the advertiser ad server and ZKA server. This message includes the transaction ID and is signed with a private key from the user's avatar.
4. Finally, once the `CONSENT_REVOKING_REPLY` and `CONSENT_REVOKING_ACKNOWLEDGMENT` is received by the ZKA server, it generates a `CONSENT_REVOKING_CONFIRMATION` including the transaction ID, which is signed with the private key of the ZKA server. This message is sent to both the Management app and the advertiser ad server.

When the described process is concluded, the three parties have generated cryptographically undeniable proof that they have implemented the consent revoking task. In addition, the advertiser provides proof of the data it had about the user that it must remove due to the consent revoking process. Note that if the advertiser does not delete the data and keeps using it after the consent revoking process has been completed, there is undeniable proof that the advertiser exploits the user's data without the user's consent. This will represent a violation of most modern data protection regulations. Therefore, our solution would allow, for instance, data protection authorities to audit misbehaving advertisers. Alternatively, a malicious advertiser may decide not to follow the defined protocol and not send the `CONSENT_REVOKING_REPLY` or `CONSENT_REVOKING_ACK`. If this occurs, there will be again evidence that the advertiser is misbehaving since both the Management app and the ZKA server would have evidence that the user has requested to revoke the consent granted to the advertiser[4]. In our view, this is an important mechanism to discourage advertisers from ignoring users' consent revocation.

Finally, as in the consent-granting process, the cryptographic proofs can be extended to use smart contracts.

Consent-based Ads delivery

The consent-based ad delivery uses exactly the same message exchange described in the Zero Knowledge Advertising case. The only difference is that the ad delivery software includes the user ID (encrypted with the public key of the advertiser ad server) in the messages shared with the advertiser ad server. By doing so, the advertiser can match this ad with the specific user who granted the consent.

3.5.3 | Billing & Accounting

The final function to implement is the billing & accounting process, which we address in this subsection. Figure 5 shows the messages exchange of our protocol to implement this function. Note that this process is triggered by an event associated with an ad (e.g., a user view or click in an ad). The process is the same, independent of the type of event.

1. Upon the completion of an event, the ad delivery software generates an `EVENT` message including the following information: type of event (e.g., click or view), an event-ID (a single identifier for the current event), the transaction-ID (corresponding to the ad associated with this event), a wallet-ID (the wallet where the cryptocurrency payment should be transferred) and the compensation associated from the event (extracted from the meta information of the ad). The ad delivery software signs the `EVENT` message with a private key from the user's avatar and sends it to both the advertiser ad

[4] Note that in our trust model, we consider that the user and the company running the ZKA server do not collude.

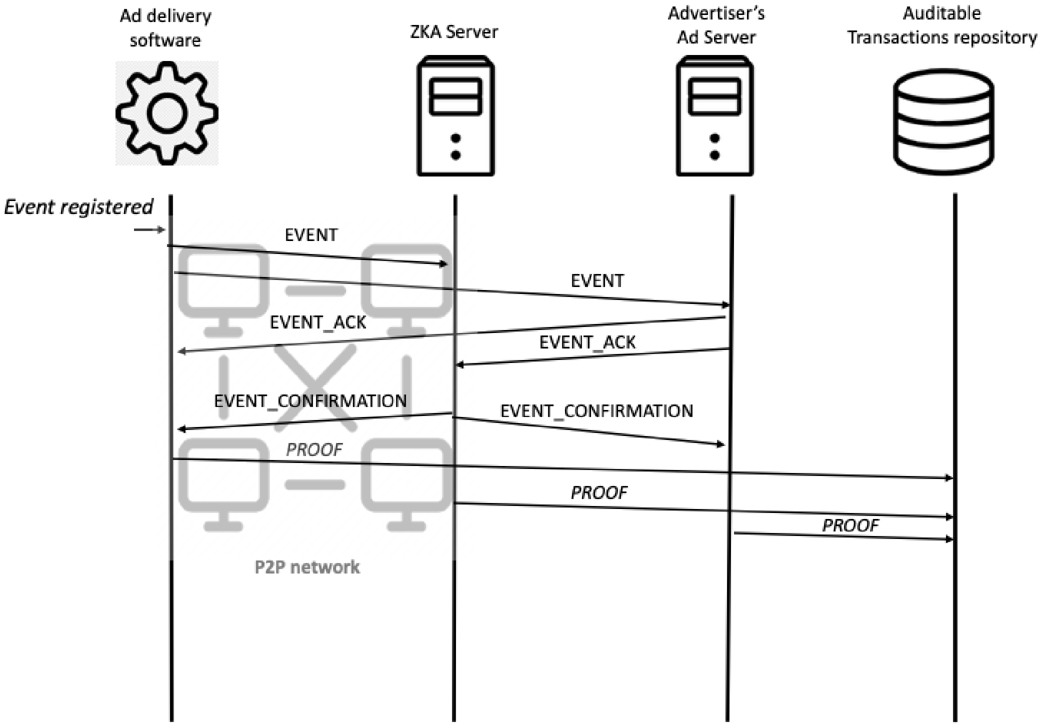


Figure 5 Billing & Accounting protocol overview.

server and the ZKA server. Note that we propose the use of Hierarchical Deterministic wallets[50], so that different transactions are associated with different public/private key pairs from the wallet. Using a single wallet-ID would make this one a potential unique identifier of the users' avatar.

2. Upon the reception of the `EVENT` message, the advertiser ad server creates an `EVENT_ACK` message including: the type of event, the event-ID, the transaction-ID, and the compensation associated with the event. The message is signed with its private key and sent to the ad delivery software and the ZKA server.
3. Finally, once the `EVENT` and `EVENT_ACK` are received by the ZKA server, this one generates an `EVENT_CONFIRMATION` message, including the same information as the `EVENT_ACK` message. This message is signed with the private key of the ZKA server.

It is important to note that consent granting also generates a monetary transaction, and thus it should undergo the billing and accounting process. The process is the same as the one described above with slight modifications: the event type is consent granting, and the transaction ID is the transaction associated with the consent-granting process.

3.6 | Auditability

In the previous section, we have described the protocol governing our solution's functionality. In each of the described processes, there are three involved players: 1) the advertiser ad server; 2) the ZKA server, and 3) the ad delivery software or the management app (depending on the specific process). In every process, each of these entities generates cryptographically signed proofs that confirm their agreement concerning the process. In some cases, these confirmations include data exposing the specific activity of the entity within the process. For instance, in the consent revoking process, the advertiser ad server declares the data that it is going to delete.

The proof generated by an entity is sent to the other two entities, such that the three entities possess for each process its own proof and the proof generated by the other two entities. It is important to note that consent granting also generates a monetary transaction, and thus it should undergo the billing and accounting process. The process is the same as the one described above with slight modifications: the event type is consent granting, and the transaction ID is the transaction associated with the consent-granting process.

These proofs are uploaded to the *auditable transactions repository* so that they can be validated and available in case any dispute between the parties requires so. In particular, we suggest this repository use a blockchain for auditing purposes. The generated cryptographic proofs can be added to blocks of a blockchain as operations that are registered in such blockchain (e.g., Ethereum), so that the blockchain serves as a public (or private) registry of the cryptographic proofs associated with the executed transactions.

Note that upon a dispute, e.g., a user rejecting it granted consent to an advertiser or advertising still using data from a user after the consent revoking process has been executed, the proofs available in the *auditable transactions repository* can be accessed to settle the dispute since they provide undeniable guarantees of the actions taken by each party.

4 | EVALUATION

The discussion presented in Sections 2 and 3 have made clear the main conceptual differences between our proposal and the state-of-the-art alternatives. Note that we claim the design of our solution represents the main contribution of this paper since it provides a new approach to delivering targeted digital ads with full data protection guarantees that extends the portfolio of the already existing solutions (See Table 1).

Hence, since the main motivation of our proposal comes from a conceptual standpoint, and we acknowledge that our proposal is meant to co-exist with other existing approaches, a head-to-head performance comparison of our solution with the existing alternatives does not add much value to the paper. Instead, we must carefully evaluate that our solution qualifies from a technical and performance point of view to operate under the requirements imposed by the current online advertising ecosystem. As described in Section 2, programmatic advertising sets strict time constraints to guarantee that ads are rendered in hundreds of milliseconds. From a quantitative performance perspective, our solution should be able to perform the complete ad delivery process within the established time constraints in the programmatic ecosystem. The equivalent of our solution within the current ad delivery process in programmatic advertising corresponds to steps 4 to 7 of the ZKA protocol described in Sec. 3.5.1 and Figure 2. This is the part subject to timing constraints. For completeness, we also present timing results related to steps 1 to 3 of the ZKA protocol, which is meant to send a list of available ads to the *ad delivery software*.

Moreover, the ad delivery software is expected to run on different types of devices, from desktops to mobile phones. The use of resources (CPU, memory) is important in all types of devices but especially in mobile phones. Mobile phones operate a more limited architecture in terms of CPU and memory and use limited-capacity batteries. Having this in mind, our solution should make limited use of these resources.

Finally, note that the obtained results can be generalized to the consent-based advertising alternative since the process is essentially the same, except for a few small variations in the information added in some of the messages (i.e., the user id), whose impact on the overall delay and resources consumption is negligible.

To assess the performance of our solution in the referred dimensions (time constraints and resource consumption), we have run extensive simulation experiments. In particular, in the rest of the section, we first describe the simulation setup and present the results obtained from the conducted simulation experiments. In particular, we evaluate 3 metrics: delay of the ad delivery process and CPU and memory consumption.

4.1 | Simulation setup

Our simulation considers all the players involved in the ZKA delivery process: the mobile app, the ad delivery software, the ZKA server, and the advertiser's ad server.

The simulator has been implemented in Python. Next, we detail the implementation of the different functions involved in the ad delivery process:

- We have implemented the exchange of messages using standard HTTPS/TCP sockets between the different players.
- The fields of the messages have been implemented using a serialization mechanism. This is a very common approach for encoding data in Internet protocol payloads.

- To implement the cryptographic operations, i.e., encryption and signature functions, we have used the Fernet library [5]. We assume a scheme URL-safe base64-encoded 32-byte key utilizing an implementation of authenticated cryptography, which is commonly used on the web.
- We have emulated the communication between the different players using an infrastructure deployed in Open Stack. Each player (ad delivery software, advertiser ad server, and ZKA ad server) is set up in an independent open stack instance with 4GB RAM and 4 vCPUs. Note that mid-range smartphones are equipped with similar resources to our open-stack instances. Instead, back-end servers offer significantly more powerful resources.
- The ad selection process should happen in the order of a few milliseconds (in the worst case). Hence, we can safely assume that its contribution to the overall delay is negligible.
- We consider the advertiser ad server to be hosted in a data center controlled by the advertiser or by a Content Delivery Network (CDN). In any of these cases, it is expected to account for network connections in the order of Gbps for both up and downstream.
- We assume that devices hosting the ad delivery software are end-user devices (smartphones, tablets, laptops, or desktops) which could then be connected to either the fix or the mobile network access infrastructure. To simulate the download and upload rates associated with these devices, we use as reference the data from the Ookla Speedtest[52] that provides the average upload and download rates for broadband fixed and mobile infrastructure across hundreds of countries. In each simulation run, we select a random value for the device's upload (download) rate from a range defined by the max and min upload (download) rates reported by Ookla for the 100 countries with the fastest networks.
- Devices hosting proxy nodes from the p2p network are end-user devices selected smartly to guarantee good performance. To this end, in our simulation, we assume that p2p proxies are end-user devices connected to broadband networks selected from the same country as the device hosting the ad delivery software. Hence, the upload and download rates of the p2p proxy would be accordingly obtained from the fixed broadband data from Ookla Speedtest.
- To simulate the effect of the p2p network, we consider a worst-case scenario in which the proxy receives the entire message from the source before forwarding it to the destination. This technique is referred to as *Stop-and-Wait* in communication protocols and is known to be inefficient. However, in the context of our paper, it serves the purpose of finding an upper bound for the delay introduced by our protocol.
- We assume that each ad metadata register included in the `AD_OFFER` has a size of 1KB. Note that this is an upper bound of the actual expected size of the metadata from an ad. In our simulation, we consider the `AD_OFFER` message to include metadata for N= [1,10,100,500,1000] ads.
- The common ad size in programmatic advertising is 150KB[53]. In our simulations, we consider the following ad sizes [100, 150, 200, 300, 400, 500, 1000] KB. Note that we consider sizes up to 6,66 times larger than the reference size of 150KB.

4.2 | Results

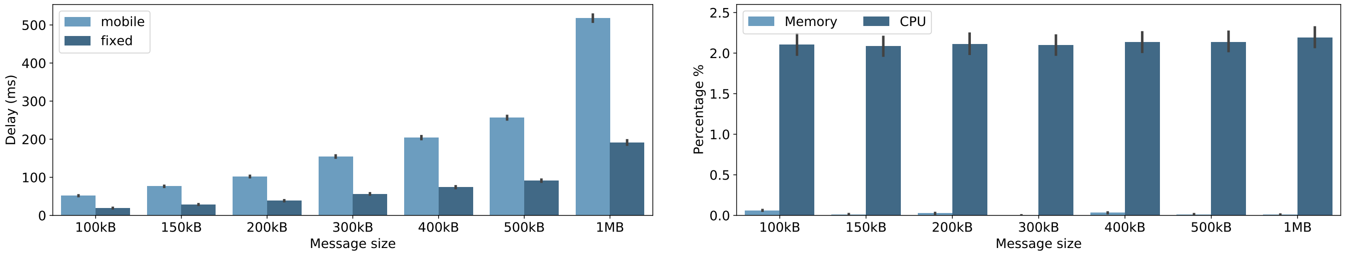
In this subsection, we present the results from the simulations regarding the communication delay, the % of CPU utilization, and the % of memory consumption for the two parts of the ZKA delivery protocol.

On the one hand, Figure 6 shows the results for the first part of the ZKA delivery protocol, involving steps 1 to 3 (See Section 3.5.1), which are meant to deliver a list of available ads to the *ad delivery software*. On the other hand, Figure 7 presents the results for the second part of the ZKA delivery protocol (steps 4 to 7), which is dedicated to the ad delivery process and is subject to delay restrictions set up in programmatic advertising.

Note that for each of the reported scenarios, we have run 500 simulation samples. We report the average value of the considered metric (delay, % of memory consumption, and % of CPU utilization) as a bar and the 95 confident intervals in the form of an error bar.

On the one hand, we observe that our proposed solution meets the expected performance requirements to operate in any device and under the timing restrictions imposed by the online programmatic advertising, even when we consider the p2p proxies

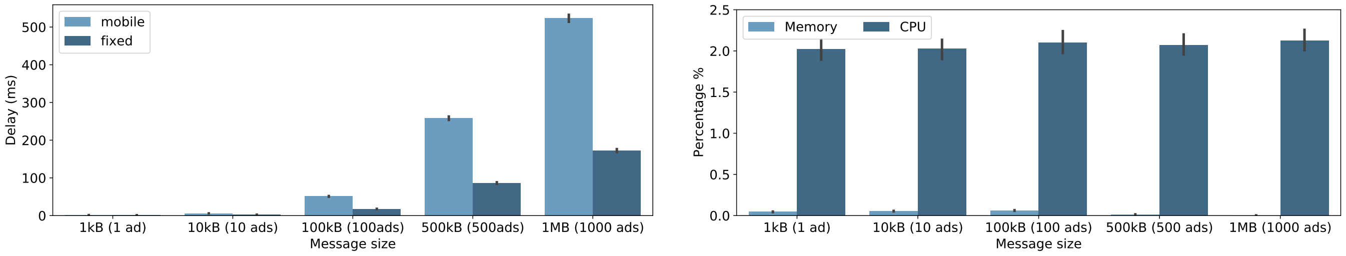
[5] https://cryptography.io/en/latest/fernet/



(a) Delay of communications

(b) Percentage of Memory and CPU consumption

Figure 6 Performance results for steps 1 to 3 of the ad delivery protocol corresponding to the delivery of the metadata of ads from the *ZKA server* to the *ad delivery software*



(a) Delay of communications

(b) Percentage of Memory and CPU consumption

Figure 7 Performance results for steps 4 to 7 of the ad delivery protocol corresponding to the delivery of the ad from the *advertiser ad server* to the *ad delivery software*

implement the inefficient *Stop and Wait* forwarding technique. In particular, for the recommended standard ad size (150kB) the average delay of the ad delivery process in our simulations is 28.51 ms and 76.34 ms for devices connected to fixed and mobile networks, respectively. Even in the considered extreme case with ads of 1MB (6,66 larger than the typical ad size), the average delay is bounded to 191.25 ms, and 518.04 ms for devices with fixed and mobile network connections, still in the range of a few hundred milliseconds. Finally, it is worth mentioning that the use of ad pre-fetching techniques as described in Section 3 would lead to negligible ad delivery delay since ads would have been pre-fetched and thus available locally in the device for each new ad space.

On the other hand, we observe that the resource consumption imposed by our solution into devices running the *ad delivery software* is affordable even for handheld devices for both the ads delivery and the metadata delivery processes. In particular, the CPU utilization and memory consumption are smaller than 2,5% and 0,1%, respectively, in all considered cases.

Finally, the bandwidth overhead generated by our protocol is negligible. Most of the bandwidth consumption is associated with the payload of the messages. First, the size of the `AD_DELIVERY` message is determined by the ad size (typically a few hundred KB), which is common to any ad delivery platform and not specific to ours. Second, the size associated with the `AD_OFFER` message, including the metadata of ads. As we have discussed before, the metadata of an ad is expected to be encoded in registers with a size <1KB. Hence, the total bandwidth consumed by these types of messages is expected to be negligible compared to the overall bandwidth consumed by regular web pages and mobile apps.

4.3 | Other performance considerations

- **P2P relay nodes**: P2P relay nodes have to handle a bandwidth overhead load associated with relaying communications from other peers. As specified above, only devices connected to high-speed connections would be selected to become a member of the p2p network, preferably nodes connected to a symmetric fiber optic connection. Note that such connections are expected to account for upload and download bandwidths in the order of hundreds of Mbps. Assuming that the average size of messages in our protocol is expected to be in the order of hundreds of KB (the largest message to be relayed by peers is the `AD_OFFER`, which is expected to be sent infrequently). Hence, a node in the p2p network in a fiber optic connection is expected to be able to handle tens of messages per second without suffering from a significant bandwidth overhead. Of course, the p2p software will implement the appropriate control techniques to cap the maximum bandwidth used by the device to serve the p2p network.

- **Scalability of the solution**: The online advertising ecosystem is estimated to handle in the order of a trillion ad requests daily. However, a single player does not handle this massive amount of transactions. Our solution has been designed to be operational at different scales. It can be operated by a company with a user base of thousands of users and tens of advertisers up to cases where the user base is in the order of hundreds of millions of millions of users and thousands of advertisers. The former case can be handled with a simple centralized solution with few servers hosted in a data center. The latter case, instead, requires an advanced worldwide scale infrastructure deployed across data centers in different continents and relies on Content Delivery Networks for ads delivery. Note that such infrastructures are common in the current online advertising ecosystem and reusable (probably with some modifications) for our solution. In addition, the p2p network is, by definition, scalable with the size of the system, i.e., the larger the number of users participating, the larger the number of nodes acting as peers. Finally, the auditing process of the information stored in the Auditing transaction repository might represent a challenge if conducted in real-time. However, we believe this is not necessary due to the non-repudiability guarantees offered by our solution, whose goal is to discourage players from misbehaving. Instead, we envision a reactive auditing process in which the transactions of a player are audited upon a properly supported complaint regarding the behavior of such a player. This reactive auditing process represents a reduction of orders of magnitude in the scalability of the auditing scheme.

5 | CONCLUSION

Different social, political, and regulatory actions have urged the online advertising industry to revisit its privacy-related practices. This has led to the development of several privacy-preserving advertising approaches.

In this paper, we propose a novel privacy-preserving advertising solution that presents a combination of functionalities that, to the best of the authors' knowledge, are not currently offered by any solutions in the field. In particular, our solution is designed to operate in multiple venues (webpages, mobile apps, etc.). In addition, it offers a combination of zero-knowledge advertising (which does not share data from users with third parties) with fine-grained consent-based targeting (which shares with those third parties explicitly indicated by a user the specific data selected by the user). Moreover, our solution operates based on explicitly declared information and preferences from users instead of using inference mechanisms and compensates users for sharing their data and interacting with targeted ads. Finally, based on cryptography technology, our solution enables full auditability.

We have described in detail the protocol that will serve as the basis for the implementation of our solution and evaluate its performance through extensive simulations. Our evaluation confirms that the proposed solution can be implemented in practice. On the one hand, our solution meets the delay requirements imposed by the programmatic advertising delivery of ads. On the other hand, it requires minimal resources (memory and CPU) from the devices the ads are delivered on.

In future work, we plan to implement the first prototype of our solution. The prototype will incorporate the feedback received from academia and industry to improve the design presented in this paper.

Conflict of interest

The authors declare no potential conflict of interests.

References

1. Wikipedia. . Cambridge Analytica Scandal. https://en.wikipedia.org/wiki/Facebook%E2%80%93Cambridge_Analytica_data_scandal; 2018.

2. Wall Street Journal. . Google Is Fined $57 Million Under Europe's Data Privacy Law. https://www.nytimes.com/2019/01/21/technology/google-europe-gdpr-fine.html; 2019.

3. BBC . Digital ad industry accused of huge data breach. https://www.bbc.com/news/technology-57232253; 2021.

4. THE EUROPEAN PARLIAMENT AND OF THE COUNCIL . General Data Protection Regulation. https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32016R0679&from=EN; .

5. California State Legislature . California Consumer Privacy Act. https://www.caprivacy.org/; .

6. Brave browser. https://brave.com/; .

7. Google . The Topics API. https://github.com/jkarlin/topics; .

8. WIBSON. https://wibson.io/en; .

9. MyDataMood. https://mydatamood.com/; .

10. Datawallet. https://datawallet.com/; .

11. Program HEU. PIMCITY: Building the next generation of personal data platforms. https://www.pimcity.eu/; .

12. The Trade Desk . Unified ID 2.0. https://www.thetradedesk.com/us/about-us/industry-initiatives/unified-id-solution-2-0; 2022.

13. Bashir MA, Farooq U, Shahid M, Zaffar MF, Wilson C. Quantity vs. Quality: Evaluating User Interest Profiles Using Ad Preference Managers.. In: ; 2019.

14. Reserch. P. Facebook Algorithms and Personal Data. https://www.pewresearch.org/internet/2019/01/16/facebook-algorithms-and-personal-data/; .

15. Toubiana V, Narayanan A, Boneh D, Nissenbaum H, Barocas S. Adnostic: Privacy preserving targeted advertising. In: ; 2010.

16. Guha S, Cheng B, Francis P. Privad: Practical privacy in online advertising. In: ; 2011: 169–182.

17. Backes M, Kate A, Maffei M, Pecina K. Obliviad: Provably secure and practical online behavioral advertising. In: IEEE. ; 2012: 257–271.

18. Helsloot LJ, Tillem G, Erkin Z. AHEad: privacy-preserving online behavioural advertising using homomorphic encryption. In: IEEE. ; 2017: 1–6.

19. Pang Y, Wang B, Wu F, Chen G, Sheng B. PROTA: A Privacy-preserving protocol for real-time Targeted Advertising. In: IEEE. ; 2015: 1–8.

20. Pestana G, Querejeta-Azurmendi I, Papadopoulos P, Livshits B. THEMIS: A Decentralized Privacy-Preserving Ad Platform with Reporting Integrity. *arXiv preprint arXiv:2106.01940* 2021.

21. Boshrooyeh ST, Küpçü A, Özkasap Ö. Privado: Privacy-preserving group-based advertising using multiple independent social network providers. *ACM Transactions on Privacy and Security (TOPS)* 2020; 23(3): 1–36.

22. Boshrooyeh ST, Küpçü A, Özkasap Ö. PPAD: Privacy preserving group-based advertising in online social networks. In: IEEE. ; 2018: 1–9.

23. IAB Tech Lab . OpenRTB Specification v3.0. https://github.com/InteractiveAdvertisingBureau/openrtb/blob/master/OpenRTB%20v3.0%20FINAL.md; 2020.

24. Google . https://developers.google.com/authorized-buyers/rtb/start; .

25. Pastor A, Cuevas R, Cuevas Á, Azcorra A. Establishing trust in online advertising with signed transactions. *IEEE Access* 2020; 9: 2401–2414.

26. Barkhuus L, Dey AK. Location-Based Services for Mobile Telephony: a Study of Users' Privacy Concerns.. In: . 3. Citeseer. ; 2003: 702–712.

27. Sapiezynski P, Stopczynski A, Gatej R, Lehmann S. Tracking human mobility using wifi signals. *PloS one* 2015; 10(7): e0130824.

28. Ilhan A, Fietkiewicz KJ. Data privacy-related behavior and concerns of activity tracking technology users from Germany and the USA. *Aslib Journal of Information Management* 2020.

29. Kröger JL, Raschke P, Bhuiyan TR. Privacy implications of accelerometer data: a review of possible inferences. In: ; 2019: 81–87.

30. Liang Y, Cai Z, Han Q, Li Y. Location privacy leakage through sensory data. *Security and Communication Networks* 2017; 2017.

31. Malloy M, McNamara M, Cahn A, Barford P. Ad Blockers: Global Prevalence and Impact. In: IMC '16. Association for Computing Machinery; 2016; New York, NY, USA: 119–125

32. Ad Blocker Usage and Demographic Statistics in 2021. https://backlinko.com/ad-blockers-users; .

33. Brazilian Government . Lei Geral de Protecao de Dados Pessoais. http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/L13709compilado.htm; .

34. South African Government . Protection of Personal Information Act. https://popia.co.za/; .

35. Brave search. https://brave.com/search/; .

36. Brave Browser Privacy Policy. https://brave.com/privacy/browser/; .

37. IPFS. https://ipfs.io/#how; .

38. Callejo P, Cuevas R, Cuevas Á. An Ad-driven measurement technique for monitoring the browser marketplace. *IEEE Access* 2019; 7: 181339–181347.

39. StatCounter . Browser Market Share Worldwide. https://gs.statcounter.com/browser-market-share; .

40. StatCounter . Browser Market Share Worldwide. https://gs.statcounter.com/browser-market-share/mobile/worldwide; .

41. Google . FLoC whitepaper. https://github.com/google/ads-privacy/blob/master/proposals/FLoC/FLOC-Whitepaper-Google.pdf; .

42. GitHub . FLoC repository. https://github.com/WICG/floc; .

43. Airbnb . Verifying your identity. https://www.airbnb.com/help/article/1237/verifying-your-identity; 2021.

44. IOHK . Atala PRISM. https://atalaprism.io; .

45. NYM. https://nymtech.net/; .

46. Introduction to Smart Contracts. https://ethereum.org/en/developers/docs/smart-contracts/; .

47. Cardano. Making The World Work Better For All. https://cardano.org/; .

48. Khovratovich D, Law J. BIP32-Ed25519: Hierarchical Deterministic Keys over a Non-linear Keyspace. In: ; 2017: 27-31

49. Fan CI, Tseng YF, Su HP, Hsu RH, Kikuchi H. Secure Hierarchical Bitcoin Wallet Scheme Against Privilege Escalation Attacks. In: ; 2018: 1-8

50. Wiki B. Hierarchical Determinisitc Wallet. https://en.bitcoinwiki.org/wiki/Deterministic_wallet; .

51. IAB Tech Lab . EXPLAINING THE PRIVACY SANDBOX EXPLAINERS. https://iabtechlab.com/blog/explaining-the-privacy-sandbox-explainers/; 2021.

52. Ookla Speed Test. https://www.speedtest.net/global-index; .

53. IAB New Ad Portfolio: Advertising Creative Guidelines. https://iabtechlab.com/standards/new-ad-portfolio/; .

□