AdVeil: A Private Targeted Advertising Ecosystem

Sacha Servan-Schreiber
MIT CSAIL
3s@mit.edu

Kyle Hogan
MIT CSAIL
klhogan@mit.edu

Srinivas Devadas
MIT CSAIL
devadas@mit.edu

Abstract

This paper presents AdVeil, a private targeted advertising ecosystem with strong security guarantees for end users. AdVeil is built around an untrusted advertising network which targets relevant ads to users and processes metrics *without* learning any of the users' personal information in the process.

Our targeting protocol combines private information retrieval with locality-sensitive hashing for nearest neighbor search. User data is kept locally on the client, giving users full control over and transparency into the contents of their targeting profiles.

AdVeil supports private billing metrics, allowing the ad network to correctly charge advertisers and pay websites for publishing ads. This is done *without* the ad network learning which user interacted with which ads. AdVeil achieves this using an anonymizing proxy (e.g., Tor) along with unlinkable anonymous tokens to identify and prevent fraud.

We build a prototype implementation of AdVeil to demonstrate its potential for real-world deployment. Our evaluation shows that AdVeil scales to ad networks with millions of targeting categories. Targeting from a set of 1 million possible categories takes roughly 1.6 seconds with a single 16-core server and is highly parallelizable. Targeting is performed out-of-band (e.g., on a daily basis) while ad delivery happens in real time as users browse the web. Verifying reports (for fraud prevention) requires less than 300 microseconds per report.

1 Introduction

Internet advertising is a $140 billion industry that relies on pervasive surveillance of people for the purpose of serving them relevant advertisements [30]. Termed *targeted* advertising, this process has been the focus of recent controversy due to the often highly personal nature of the data being used and the invasive collection practices [50, 85, 97]. Proposals for moving away from targeted advertising often tout *contextual* advertising as a private alternative [62, 72, 94]. Contextual ads are chosen based *only* on the website they will be displayed on, and *not* on personal data, preserving user privacy.

However, relevancy of advertisements for end users is believed to increase user engagement which, consequently, increases profit [66, 79]. Alphabet and Meta, the most prominent ad-tech companies today, respectively earned 93% and 97% of their 2021 revenue from advertising alone [14, 83]. Because of this, ad networks have proven unwilling to move away from targeted advertising and the associated surveillance. Recent proposals from Google for a privacy preserving alternative [114, 115] fall short, with privacy advocates such as the Electronic Frontier Foundation (EFF) referring to them as "the opposite of privacy-preserving technology" [38]. Other solutions for privacy-preserving targeted advertising have practical performance limitations or fail to achieve accuracy comparable to non-private advertising [19, 24, 53, 55–57, 63, 87, 107, 108, 114, 115].

AdVeil is a low-latency, scalable solution for targeted advertising with clear-cut privacy guarantees. The main goal of AdVeil is to provide *unlinkability* between users and their personal data. We define *personal data* to mean any information about the user, including which ads they interact with. The interests and demographics contained in users' targeting profiles are not sent in the clear—even anonymized—to the ad network as part of the targeting process. AdVeil ensures that the ad network learns **only** which ads are viewed (and clicked on) by users, but not which user saw any given ad.

AdVeil is (trivially) compliant with existing data privacy legislation (e.g., GDPR [6] and CCPA [7]). User profiles are held locally, with users themselves having full control over and transparency into *which* of their features are used for targeting. They may even opt out of targeted advertising entirely, in which case websites can cleanly fall back to serving contextual ads. Giving users control over their own data is important, since companies have been found to use protected demographics like gender, race, or religion to target advertisements in a discriminatory manner [13, 31, 58, 116]. This practice can limit the visibility of needed services, such as education, to

people who might have benefited from them [41, 88].

Finally, addressing ad fraud is a crucial requirement for online advertising [99, 122]. Malicious parties may attempt to generate false ad interactions to artificially deplete an advertiser's budget or increase revenue for a website displaying ads. AdVeil provides integrated fraud-prevention based on unlinkable anonymous tokens, preventing abuse by botnets.

Components. AdVeil is realized using well-established building blocks which we combine to provide a scalable, accurate, and privacy-preserving advertising ecosystem. We use single-server private information retrieval (PIR) in conjunction with nearest neighbor search to achieve private ad targeting. We note that, while faster, the trust and availability requirements for multi-server PIR make it an impractical choice in this setting (Section 7.1.2). To the best of our knowledge, we are the first to use PIR to privately *target* ads to users. Our fraud-prevention mechanism is built using unlinkable tokens with metadata [67, 96, 110], which enable anonymous flagging of suspicious requests. A generic anonymizing proxy (e.g., Tor [43]) is used during delivery and reporting to hide user identities. Targeting (unlike delivery and reporting) does not run over the anonymizing proxy because targeting operates on personal data which, without PIR, would need to be sent to the Broker. See Section 5.2 for details.

Contributions:

1. A novel and efficient protocol for ad targeting that reveals no information about the user profile.
2. A fraud prevention mechanism based on anonymous tokens that is fully compatible with private browsing and existing ad-fraud countermeasures.
3. Out-of-the-box compliance with privacy regulations such as GDPR and CCPA, in addition to full compatibility with existing anti-tracking and privacy-enhancing web technologies.
4. An open-source prototype implementation [4] which we empirically evaluate to demonstrate applicability to a real-world deployment.

Limitations:

1. AdVeil imposes a computational overhead on the ad network, which translates to higher operational costs. However, we provide concrete cost estimates and show that AdVeil can remain profitable, even when deployed on commodity hardware (Section 7).
2. AdVeil is intended to *disincentivize* web tracking by ad networks, but is not itself a defense against browser fingerprinting or user tracking.
3. AdVeil requires cooperation of browsers. However, major browser vendors already support aspects of privacy-preserving advertising [87, 105, 114, 117].

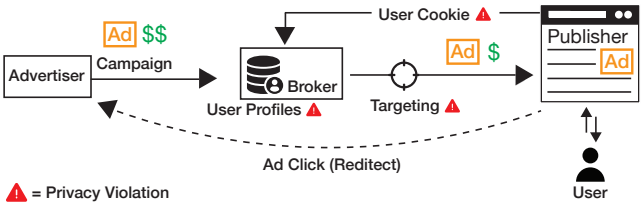


Figure 1: A non-private targeted advertising pipeline.

2 Background

In this section, we introduce the existing, non-private advertising ecosystem. This ecosystem consists of a set of parties who engage in different stages of an *advertising pipeline*, which supports targeted advertising.

2.1 Participants

We use standard terminology for parties that comprise the advertising pipeline [19, 53, 55, 63, 107]:

- **Users**: people browsing the web and viewing ads;
- **Clients**: web browsers (e.g., Firefox or Chrome) controlled by users;
- **Advertisers**: companies with products and services to advertise (e.g., Squarespace or Bose);
- **Publishers**: websites that display ads to users viewing the webpage (e.g., `wired.com` or mobile apps);
- **Broker**: an ad-tech company (e.g., Google) responsible for matching users to ads, billing advertisers, and compensating publishers for user interactions.

The Broker, or ad network, can be thought of as the governing body of the advertising ecosystem. The Broker's primary goal is matching ads from an Advertiser to the users most likely to engage with the ad. This is done via Publishers which generate revenue by displaying ads to users visiting their webpage.

2.2 Advertising pipeline

The advertising pipeline, shown in Figure 1, begins when an Advertiser creates a new ad campaign with the Broker, specifying a target audience and allocating a campaign budget. Abstractly, the target audience can be specified as a *feature vector* which contains information about the contents of the ad, e.g., tags such as `mechanical keyboards` or `outdoor gear`. In addition to the feature vectors provided by Advertisers, the Broker possesses *user profiles* containing demographics and interests, e.g., `[woman, computers, high-income]`, about users. These profiles are constructed by pervasively tracking users on the web [116]. The Broker then *targets* ads to users using their personal profile [90, 114]. This can be abstracted as a *nearest neighbor search* problem between vectors [16].

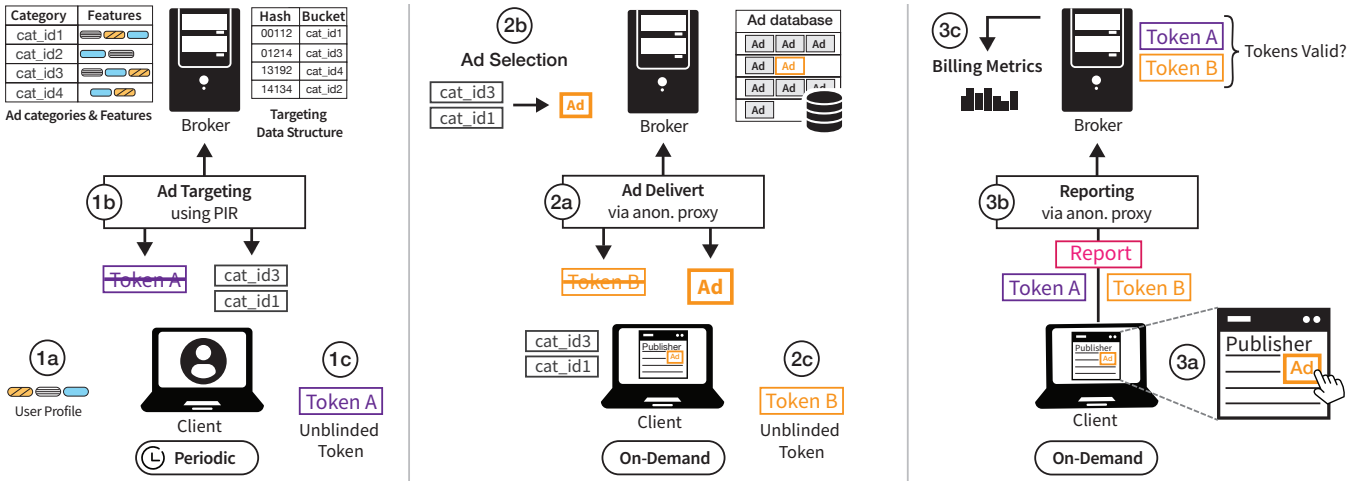


Figure 2: Overview of AdVeil. Description of the components is provided in Section 5.

When users are shown an ad on a publisher's webpage, a browser script generates an *impression* report for the Broker. This is by far the most common type of user engagement, followed by *clicks*, which are redirected through the Broker to measure engagement. With impression reports and click redirects, the Broker obtains necessary metrics for billing Advertisers, compensating Publishers, and updating targeting algorithms. Crucially, the Broker eliminates all fraudulent views and clicks that it deems to be generated by bots, or malicious Publishers and clients, to artificially skew perceived user interactions with ads [99].

2.3 Styles of online advertising.

Online advertising consists of two main targeting strategies for matching ads to users: *contextual* and *behavioral*.

Contextual advertising is done *independently of the user's profile*. Publishers display ads that are relevant to their own content, which is presumably of interest to visiting users based on their choice to access the Publisher's site or app.

Behavioral ad targeting matches a user to a set of ads based on the user's *profile*. Each profile consists of the user's browsing behavior obtained by the Broker through user tracking. Retargeting is a common type of behavioral advertising in which the Broker may preferentially display an Advertiser's ads to users who have previously interacted with the Advertiser.

Real-time Bidding is a mechanism associated with both behavioral and contextual advertising. It occurs on-demand when a user requests an ad and auctions the targeted category to the highest bidding Advertiser.

2.4 Ad Fraud

In online advertising, fraud refers to ad interactions that never involved a real user. Often, this is perpetrated by malicious Publishers to increase their ad revenue, using botnets to generate fake interactions. Defense strategies thus rely on identifying traffic that originates from a botnet or otherwise does not involve a human and *covertly* demonetizing it [67]. This is because fraud detection strategies are heuristic and providing any feedback on whether an interaction was classified as fraudulent would allow the perpetrators to refine their strategy and evade detection in the future.

3 The AdVeil system

AdVeil is a general-purpose ecosystem for targeted advertising that provides privacy to users while retaining comparable targeting accuracy, performance, metrics, and fraud prevention for the Broker.

3.1 Overview

AdVeil follows the steps described in Figure 2, which transform the advertising pipeline of Figure 1 into a private ecosystem. AdVeil is designed around *targeting*, *delivery*, and *reporting*. These three stages abstract the parts of the advertising pipeline where personal user information is involved (see Figure 1).

Targeting (steps (1a) to (1c) in Figure 2) assigns relevant ads to a user. The client holds a user profile (step (1a)) described as a high-dimensional feature vector [90]. Targeting categories are likewise associated with a feature vector describing attributes of the ads in each category. The targeting protocol outputs a selection of ad categories to the user's client. In addition, for each targeted category, the Broker provides a blind signature on a special anonymous token with an embedded private metadata bit [67] indicating whether the client is identified as a "human" or a "bot." The client later returns the unblinded token (step (1c)) to the Broker in reporting, allowing the Broker to discard "bot" reports.

Delivery (steps ⓶ⓐ to ⓶ⓒ in Figure 2) retrieves a new ad from a targeted category to display to the user. The client first sends the category to the Broker via the anonymizing proxy to hide its identity (step ⓶ⓐ). The Broker selects an ad for the given targeting category in real-time (step ⓶ⓑ). This mechanism provides support for real-time bidding and other on-demand delivery logic (see Section 5.3). As in Targeting, the Broker provides a blind signature on a fresh anonymous token with embedded *public* metadata bits [96], binding the token signature to the returned ad. In reporting, the unblinded token (step ⓶ⓒ) ensures that the report is uniquely associated to a delivered ad. The anonymizing proxy prevents the Broker from linking the ad identifier to the client.

Reporting (steps ⓷ⓐ to ⓷ⓒ in Figure 2) provides interaction reports (e.g., views and clicks; step ⓷ⓐ) to the Broker. To do so privately, the client batches and sends all reports to the Broker at fixed time intervals through the anonymizing proxy. The client also attaches the signed tokens obtained in the targeting and delivery protocols. All reports with invalid tokens are deemed fraudulent and discarded by the Broker. All remaining (valid) reports are used by the Broker for metrics and billing purposes (step ⓷ⓒ).

3.2 Threat model and requirements

AdVeil has two independent security concerns: user privacy, which we define as *unlinkability* between users and their personal data, and *fraud prevention*, which we define as reporting statistics that are resilient to botnets.

Personal data. Concretely, we define *personal data* to be any information that is related, either directly or indirectly, to a user's targeting profile. This includes elements of the user's feature vector used for targeting, which may contain demographics and interests, and are thus *directly* related to the user's profile. It also includes any ads or ad categories the user sees or interacts with. This is because, given the nature of targeting, ads are *indirectly* related to the user's profile through the targeting algorithm. While the Broker may indirectly infer profile features from the ad reports, they remain unlinked from the identity of any users (Section 6). Importantly, ad reports are also not linkable to each other, even those that are generated by the same user. As such, the Broker never learns whether any two ads were seen by the same or different users, limiting its ability to even indirectly inferring personal data.

Adversarial model. AdVeil assumes a rational [48, 54, 119] Broker that may try to link the identity of users to their personal data. A **rational** Broker is incentivized to provide correct functionality for its advertising ecosystem. We make two assumptions about the Broker's behavior in order to provide user unlinkability:

1. the Broker does not deny service to honest users during ad targeting or delivery and,
2. the Broker covertly excludes all fraudulent reports.

These assumptions mirror those made in online advertising today and are upheld by the financial incentives of the Broker. If the Broker refuses to serve ads, then it cannot collect revenue from advertisers. Similarly, if fraudulent reports are included in metrics, then the Broker violates the billing contract it has with Publishers and Advertisers. Overtly excluding fraudulent reports by denying service rather than using AdVeil's covert fraud prevention mechanism would alert bots of detection and allow them to evolve their strategies.

Unlinkability. Privacy in AdVeil is defined as *unlinkability* between users and their personal data. AdVeil guarantees that any data that users report on cannot be tied back to their identity. For accurate billing, it suffices to report the interaction type (e.g., view or click) and the ad involved; reports do **not** need to contain any information about the user.

Unlinkability is defined on a per-epoch basis, where each epoch is the time period from the start of the targeting protocol through the end of reporting. Epochs hide timing correlations between AdVeil's protocols and ensure that all users participating in a given epoch are part of an *anonymity set*. The Broker cannot determine which user within an anonymity set was responsible for any given report. We discuss potential cross-epoch leakage resulting from intersection attacks [39, 40, 75, 77, 118] in Section 6.1. To achieve unlinkability, we require that users make some requests via an anonymizing proxy such as Tor [43] (any anonymizing proxy hiding the sender identity is sufficient); see Section 4. The anonymizing proxy ensures that, while the Broker learns which ads are delivered, seen, and reported on, it does not learn the identity of the user involved.

Fraud Prevention. AdVeil expects users of the system to arbitrarily misbehave or be impersonated by large scale bot networks. Users may try to avoid seeing ads entirely or incur billing for ads that were never delivered.

AdVeil integrates a fraud-prevention mechanism using anonymous tokens [67, 96] that cleanly composes with existing bot detection mechanisms. Importantly, this is a *covert* method of fraud prevention. That is, while the Broker can recognize reports with invalid tokens, no other user can. This prevents bots from using AdVeil's tokens to learn which of their evasion strategies were successful [67]. In addition, AdVeil ensures that users only obtain tokens for the exact ads they were delivered, preventing them from generating valid reports on arbitrary ads. In AdVeil, fraud-prevention guarantees that:

1. (only) the Broker can identify fraudulent reports,
2. reports are never double-counted in billing, and
3. only delivered ads can be reported on.

We note that AdVeil does not make it easier for users to block ads. If users opt out of targeted advertising by not participating, then Publishers can automatically fall back to *contextual* ads which do not require any user data or participation.

Non-goals. While AdVeil supports the return of arbitrary data during the reporting phase, it only *requires* reports to contain the ad identifier. Determining what, if any, user features can be reported on privately is an orthogonal problem. AdVeil is compatible with any choice of data privacy mechanism such as differential privacy [44] or k-anonymity [103] if any user data beyond ad interactions is to be reported.

4 Building blocks

We design AdVeil around several cryptographic and data structure building blocks. Overall, we aim to make AdVeil general and modular, so as to fit a wide array of potential deployment scenarios.

For private targeting, we use single-server PIR [17, 69] in conjunction with a standard similarity search data structure based on locality-sensitive hashing (LSH) [16, 49]. We emphasize that using an LSH-based data structure ensures that AdVeil is compatible with targeting techniques used in practice [90, 114].

For private delivery and reporting, we use an anonymizing proxy (e.g., Tor [43]), which reveals which ad was retrieved or reported on but not which client downloaded the ad.

For data integrity in both targeting and delivery, the Broker commits to all data structures using a vector commitment (e.g., Merkle tree) that can be efficiently verified by the client.

For fraud prevention, we use anonymous one-time-use tokens with public metadata. These tokens enforce a "one delivery; one report" policy and ensure that each report is uniquely associated with a prior ad delivery.

4.1 Private information retrieval

Private information retrieval (PIR) is a protocol for retrieving items from a remote database or data structure without revealing *which* item was retrieved [25, 34, 35]. PIR in the single-server setting [69] with a database of N items requires $O(N)$ work on the server and sublinear communication (in practice $O(\sqrt{N})$ communication [17]) between the client and database. In AdVeil, PIR is used by the client to privately query the targeting data structure and obtain a set of ad categories.

Definition 1 (PIR [69]; informal)**.** *Let M be scheme-specific public parameters. For a database (or a key-value store [35]), PIR has the following functionality:*

$\underline{\mathsf{Query}}(M, \mathsf{idx}) \to (s, Q)$. *Takes as input public parameters and an index. Outputs secret state s and query Q.*

$\underline{\mathsf{Answer}}(\mathcal{D}, Q) \to A$. *Takes as input a database $\mathcal{D}$ and query Q. Outputs answer A.*

$\underline{\mathsf{Recover}}(s, A) \to \mathcal{D}[\mathsf{idx}]$. *Takes as input secret state s and query answer A. Outputs recovered database item $\mathcal{D}[\mathsf{idx}]$.*

The functionality must satisfy correctness *and* privacy. *Informally, correctness holds if the computed answer produces the item at index* idx *in the database $\mathcal{D}$ when fed through* Recover. *Privacy for the client holds if the (potentially malicious) server learns no information on the query. We refer to Kushilevitz and Ostrovsky [69] and Angel et al. [17] for more formal definitions.*

4.2 Anonymous one-time-use tokens

We make use of anonymous one-time-use tokens [67, 96, 110] for fraud prevention. These tokens are instantiated between a prover and a verifier. The Broker (i.e., acting as the verifier) signs a blinded token generated by the client (e.g., acting as the prover). During signing, the Broker can embed public metadata [96, 110] into the signature. Later, the unblinded token and the signature is redeemed by the Broker as part of a report. The redeemed token is unlinkable to the originally signed token. When redeeming a token, the Broker can:

1. check if the signed token is valid,
2. check if the token was previously redeemed, and
3. read the public metadata from the token signature.

We use the randomized anonymous tokens of Kreuter et al. [67] based on Okomoto-Schnorr blind signatures [86] to achieve fraud prevention and rate limiting in AdVeil. We use the public metadata extension of Tyagi et al. [110] when required. Specifically, AdVeil uses two distinct token schemes: one that supports valid/invalid tokens *and* one that supports embedded public metadata. For simplicity, we present a single interface that captures both functionalities.

Definition 2 (Anonymous tokens; informal)**.**
An anonymous token scheme [42, 67, 96, 110] consists of efficient algorithms Setup, KeyGen, TokenGen, Sign, Unblind *and* Redeem *with the following functionality:*

$\underline{\mathsf{Setup}}(1^\lambda) \to \mathsf{crs}$. *Outputs a common reference string* crs.

$\underline{\mathsf{KeyGen}}(\mathsf{crs}) \to (\mathsf{pk}, \mathsf{sk})$. *Outputs a new public key* pk *and secret token signing key* sk.

$\underline{\mathsf{TokenGen}}(\mathsf{pk}) \to (\tau, \tilde{\tau}, r)$. *Outputs a new token τ, blinded token $\tilde{\tau}$, and blinding randomness r using the public key.*

$\underline{\mathsf{Sign}}(\mathsf{sk}, \tilde{\tau}, \mathsf{md}) \to \tilde{\sigma}$. *Takes as input the secret signing key, blinded token $\tilde{\tau}$, and (optional) public metadata. Outputs signature $\tilde{\sigma}$ on the blind token $\tilde{\tau}$ containing* md.

$\underline{\mathsf{Verify}}(\mathsf{pk}, \tau, \sigma, \mathsf{md}) \to \mathsf{yes}/\mathsf{no}$. *Takes as input the token, signature, and public metadata. Outputs* yes *if and only if σ contains embedded public metadata* md.

$\underline{\mathsf{Unblind}}(\mathsf{pk}, \tilde{\tau}, \tilde{\sigma}, r) \to (\tau, \sigma)$. *Takes as input the public key, blinded token $\tilde{\tau}$, blind signature $\tilde{\sigma}$, and blinding randomness r. Outputs unblinded token τ and signature σ.*

$\underline{\mathsf{Redeem}}(\mathsf{sk}, \ell, \tau, \sigma) \to (\mathsf{md}, \ell')$. *Takes as input a secret key* sk*, spent token list* ℓ*, token* τ*, and signature* σ*. Outputs whether the token is valid with respect to the spent list and signature, public metadata* md*, and updated spent list* ℓ'*. For notational convenience, we let* $\mathsf{md} = \mathsf{invalid}$ *if the token* τ *or signature* σ *is invalid.*

The functionality must satisfy completeness, unlinkability, and unforgeability. A token may have no embedded metadata, in which case the metadata is denoted by $\perp$*.*

Properties of one-time-use anonymous tokens. Completeness states that a verifier always accepts valid tokens using Redeem. Unforgeability states that a prover cannot forge a valid token or change the embedded metadata. Unlinkability requires that the verifier learns nothing beyond the public metadata and token validity.

A new construction. We make a new observation on the Okomoto-Schnorr Privacy Pass (OSPP) token construction described by Kreuter et al. [67]. Our insight is that the OSPP-without-NIZK construction [67] is ideally suited to fraud-prevention. Using OSPP-without-NIZK, the Broker can prevent fraud by giving *invalid* token signatures to bots and *valid* signatures to honest clients during targeting. By the randomization of the token, clients cannot tell whether they received a valid or invalid token (thus not alerting bots to detection). Such tokens satisfy 2-unlinkability [67], meaning that all valid and invalid tokens are unlinkable within their respective anonymity sets. See Appendix C for details.

4.3 Other tools

Locality-sensitive hashing (LSH) [16, 49, 59] is a tool used in recommendation engines for efficiently matching users to products. More concretely, LSH is useful for solving the nearest neighbor search problem. Neighboring (i.e., similar) feature vectors can indicate potential relevancy of a specific ad to a user (e.g., Google's FLoC [114] proposal uses LSH to group users to interest categories). We use LSH as the building block in our private targeting mechanism, described in Section 5.1.

Vector commitments [12, 23, 32, 82, 106] (e.g., Merkle trees) are used for proving correctness of a value at a given index in a vector, relative to a short commitment. In AdVeil, vector commitments guarantee targeting consistency across clients, which prevents "tagging" clients through ad targeting. The two properties that are important for understanding AdVeil are that each proof is (1) of logarithmic (or constant [12, 23, 32, 106]) size relative to the size of the vector and (2) can be efficiently verified given only the commitment, value, and proof.

Anonymizing proxies [10, 43, 112] serve as an intermediary, or *proxy*, for communications between a client and server to hide the relationship between the sender and recipient of messages. AdVeil only requires *client anonymity* [20] from the proxy. That is, the identity (IP address) of the client should be hidden by the proxy. The Onion Router (Tor) [43], Invisible Internet Project (I2P) [10], and VPN0 [112] all provide this property. Tor is the most widely deployed anonymizing proxy and is bundled by default in the Brave browser [1] and available through add-ons for Firefox [9] and Chrome [8].

5 System architecture

This section introduces our construction of AdVeil. We describe protocols for each stage of the advertising pipeline, using the tools of Section 4. We start by explaining a mechanism for *non-private* targeting, which forms the foundation of our private targeting protocol.

5.1 Targeting and nearest neighbor search

Hashing-based data structures solve *approximate* nearest neighbor (ANN) search. ANN search is at the core of recommendation systems, including ad targeting systems [90, 92, 114, 121]. We note that ANN search is a general problem and can be applied to many definitions of "similarity" [16]. ANN search is defined over a set S of high-dimensional feature vectors and a query vector $\boldsymbol{q}$. For a fixed similarity metric, ANN search returns the approximate nearest neighbor(s) of $\boldsymbol{q}$ in S.

Ad targeting can be realized using ANN search [114]. Let $\mathcal{D}$ be a set of N ad categories and corresponding feature vectors. Each $(\mathsf{id}_i, \boldsymbol{v_i}) \in \mathcal{D}$ has id_i representing the ith category and $\boldsymbol{v_i}$ describing the targeting attributes for the category. To find the most relevant ad category for a user profile $\boldsymbol{q}$, we can find the ANN of $\boldsymbol{q}$, which we denote by $\boldsymbol{v_j}$, and output the corresponding id_j.

A common data structure used for solving ANN search efficiently is based on LSH (e.g., MinHash [27] or SimHash [33]). In the context of ad targeting, we define the data structure by two algorithms: Build and Target. We present a concrete instantiation of Build and Target which follow the ANN search data structure of Gionis et al. [49]. In Section 5.2, we transform Target into a privacy-preserving protocol between the client and the Broker using PIR. See the survey of Andoni et al. [16] for many details on ANN search and parameter selection.

$\underline{\mathsf{Targeting.Build}}(\mathcal{D}, \mathcal{H}, L) \to \mathcal{S}$.

1: Build L hash tables $T_1, \ldots, T_L$ using LSH functions $\mathsf{h}_1, \ldots, \mathsf{h}_L$ sampled i.i.d. from LSH family $\mathcal{H}$;
2: For each $\boldsymbol{v_i} \in \mathcal{D}$, compute $k_j \leftarrow \mathsf{h}_j(\boldsymbol{v_i})$, and append $(\mathsf{id}_i, \boldsymbol{v_i})$ to the bucket in hash table T_j with key k_j;
3: Output $\mathcal{S} = \{T_1, \ldots, T_L, \mathsf{h}_1, \ldots, \mathsf{h}_L\}$.

$\underline{\mathsf{Targeting.Query}}(\mathcal{S}, \boldsymbol{q}) \to \mathsf{id}$.

1: Compute $k_j \leftarrow h_j(\boldsymbol{q})$ for $j \in [L]$;
2: Set $C := B_1 \cup \cdots \cup B_L$ where each B_j is the bucket in hash table T_j with key k_j;
3: Output the id of the most similar vector to $\boldsymbol{q}$ in C.

5.2 AdVeil protocols

AdVeil realizes each stage of the advertising pipeline via separate and modular protocols. We first describe the context in which the protocols are instantiated. With the exception of Protocol 1 (Setup), the protocols describe the steady state of AdVeil.[1] We begin by describing the background setting and assumptions.

Epochs. AdVeil divides time into discretized epochs. Aside from Protocol 1, which runs only once, all protocols run within the context of these epochs. Epoch duration affects the frequency of ad targeting and reporting, but not *delivery*, which remains on-demand.

Ad feature vectors. As a starting point, we assume that the Broker has a set of targeting categories and a database of ads within these categories. AdVeil is agnostic to how the Broker defines the categories, making it compatible with a variety of strategies (mapping of user features to ad category features [92]).

User profile. The user's profile feature vector is constructed locally by the client using any "profile building" function $\mathcal{F}_{\text{profile}}$, provided as part of the public parameters generated in setup (Protocol 1). For example, $\mathcal{F}_{\text{profile}}$ can map user browsing history to a profile feature vector. It can also include information about websites they visit and searches performed [64, 68, 91, 114]. However, because AdVeil depends on $\mathcal{F}_{\text{profile}}$ being *local to the client*, the choice of what inputs $\mathcal{F}_{\text{profile}}$ is evaluated on depends entirely on the client. A malicious client can choose to not evaluate $\mathcal{F}_{\text{profile}}$. However, this is a feature not a bug: it guarantees that users have full control over their personal data and makes AdVeil immediately compatible with privacy regulations such as the GDPR [6].

5.2.1 Setup (Protocol 1).

The setup for the Broker involves publicly committing to all the parameters required for targeting, delivery, and reporting such that it cannot equivocate. This commitment can be posted to a public bulletin board [21, 36] or disseminated through a gossip network [70, 81, 84]. The Broker commits to the targeting hash tables (using vector commitments [32, 52, 71]). The Broker also generates a new token keypair (Definition 2) and posts the public key. For simplicity, we assume that all commitments and public parameters are consistent and accessible to all clients via the public bulletin board.

[1] Protocol 1 can be re-run to update targeting parameters.

Protocol 1: AdVeil Setup

Step 1 (Broker)

// $\mathcal{D}$: feature vectors for each targeting category
1: $\mathcal{S} \leftarrow$ Targeting.Build($\mathcal{D}$, params). // See Section 5.1
2: $C_{\mathcal{S}} \leftarrow$ VC.Commit($\mathcal{S}$). // Commitment to hash tables
3: (pk, sk) $\leftarrow$ AT.KeyGen(crs). // Anonymous token keys
4: **Publish**: common reference string crs, public key pk, params, commitment $C_{\mathcal{S}}$, profile building function $\mathcal{F}_{\text{profile}}$, and LSH functions $h_1, \ldots, h_L$ on a public bulletin board.

5.2.2 Targeting (Protocol 2)

Targeting occurs at the beginning of each epoch (e.g., once a day) and runs between the client and the Broker. The client obtains the relevant ad category while the Broker learns **nothing** about which category was targeted in the process. Protocol 2 trivially extends to output the top-k [49] ad categories with minimal overhead. Along with the categories, the client receives the Broker's blind signature on the provided anonymous token(s). Each token signature is either valid or invalid, indicating whether the client is believed to be a "bot" or a "human" and is later used in Protocol 4 to discard bot reports.[2] To ensure that the Broker returns the correct set of targeted categories, the client checks the vector commitment $C_{\mathcal{S}}$. To facilitate this, we assume the Broker concatenates the proof string for each hash table bucket to the bucket contents. Thus, the client can retrieve the proof along with the contents when privately querying the targeting hash tables through PIR in Protocol 2. The client aborts if any of the proofs fail to verify.

5.2.3 Delivery (Protocol 3)

Clients use Protocol 3 on-demand within epochs to retrieve ads corresponding to the targeted categories obtained from Protocol 2. The client fetches ads for the targeted category from the Broker via the anonymizing proxy. Because of this, the Broker has the ability to decide *which* ad to deliver for the requested category. Broker-side selection logic such as real-time bidding can be applied to decide which ad to serve, on demand. We discuss the use of PIR as an alternate delivery mechanism in Section 7, but note that it does not provide the required performance for on-demand delivery nor the ability to support Broker-side ad selection.

5.2.4 Reporting (Protocol 4)

Reporting occurs at the end of an epoch. When the user interacts with a previously retrieved ad displayed on a Publisher's webpage, such as seeing (or clicking on) an ad, an impression

[2] We note that the clients *cannot* determine the validity of the token signature (Appendix C) which prevents alerting malicious clients or bots of detection.

Protocol 2: Targeting

Step 1 (Client)

// $\boldsymbol{q}$: user profile feature vector held by the client
// h_i: LSH functions in the public parameters

1: $k_i \leftarrow \mathsf{h}_i(\boldsymbol{q})$ for $i \in [L]$. // Compute profile hash keys
2: $Q_i \leftarrow \mathsf{PIR.Query}(M, k_i)$ for $i \in [L]$. // Query for bucket
3: $(\tau_t, \widetilde{\tau_t}, r) \leftarrow \mathsf{AT.GenToken}(\mathsf{pk})$. // New reporting token
4: Send queries $Q_1 \ldots Q_L$ and $\widetilde{\tau_t}$ to the Broker.

Step 2 (Broker)

1: $A_i \leftarrow \mathsf{PIR.Answer}(T_i, Q_i)$ for $i \in [L]$. // Query answer
2: $\widehat{\mathsf{sk}} \leftarrow \begin{cases} \mathsf{sk}^* \text{ if client identified as } \textbf{bot}. & \text{// invalid signature} \\ \mathsf{sk} \text{ if client identified as } \textbf{human}. & \text{// valid signature} \end{cases}$
3: $\widetilde{\sigma_t} \leftarrow \mathsf{AT.Sign}(\widehat{\mathsf{sk}}, \widetilde{\tau_t}, \perp)$.
4: Send $A_1, \ldots, A_L$ and $\widetilde{\sigma_t}$ to the client.

Step 3 (Client)

1: $(B_1 \| \pi_1, \ldots, B_L \| \pi_L) \leftarrow \mathsf{PIR.Recover}(A_1, \ldots, A_L)$.
2: If there exists i such that $\mathsf{VC.Verify}(C_{\mathcal{S}}, B_i, \pi_i) = \text{no}$, abort.
3: $\mathsf{id} \leftarrow$ nearest neighbor to $\boldsymbol{q}$ in C where $C := B_1 \cup \cdots \cup B_L$.
4: $(\tau_t, \sigma_t) \leftarrow \mathsf{AT.Unblind}(\mathsf{pk}, \widetilde{\tau_t}, \widetilde{\sigma_t}, r)$. // Unblind signature
5: Output $(\mathsf{id}, \tau_t, \sigma_t)$.

Protocol 3: Delivery

Step 1 (Client)

// id: targeted category obtained from Protocol 2

1: $(\tau_d, \widetilde{\tau_d}, r) \leftarrow \mathsf{AT.GenToken}(\mathsf{pk})$. // New delivery token
2: Send $(\mathsf{id}, \widetilde{\tau_d})$ to the Broker, via the **proxy**.

Step 2 (Broker)

// $\mathcal{D}$: database of ads

1: $\mathsf{ad_{id}} \leftarrow \mathsf{SelectAd}(\mathcal{D}, \mathsf{id})$. // real-time selection logic
2: $\widetilde{\sigma_d} \leftarrow \mathsf{AT.Sign}(\mathsf{sk}, \widetilde{\tau_d}, \mathsf{ad_{id}})$. // Ad ID as public metadata
3: Send $(\mathsf{ad}, \mathsf{ad_{id}}, \widetilde{\sigma_d})$ to the client, via the **proxy**.

Step 3 (Client)

1: $(\tau_d, \sigma_d) \leftarrow \mathsf{AT.Unblind}(\mathsf{pk}, \widetilde{\tau_d}, \widetilde{\sigma_d}, r)$.
2: If $\mathsf{AT.Verify}(\mathsf{pk}, \sigma_d, \mathsf{ad_{id}}) = \text{no}$, abort. // Invalid metadata
3: Output $(\mathsf{ad}, \tau_d, \sigma_d)$.

Protocol 4: Reporting

Step 1 (Client)

// report: report payload
// (τ_t, σ_t): token and signature from Protocol 2
// (τ_d, σ_d): token and signature from Protocol 3

1: Send $(\text{report}, \tau_t, \sigma_t, \tau_d, \sigma_d)$ to the Broker, via the **proxy**.

Step 2 (Broker)

// (ℓ_t, ℓ_d): lists of all redeemed tokens

1: $(\mathsf{md}_t, \ell_t') \leftarrow \mathsf{AT.Redeem}(\mathsf{sk}, \ell_t, \tau_t, \sigma_t)$.
2: $(\mathsf{md}_d, \ell_d') \leftarrow \mathsf{AT.Redeem}(\mathsf{sk}, \ell_d, \tau_d, \sigma_d)$.
3: If $\mathsf{md}_d = \text{valid}$ or $\mathsf{md}_d = \text{invalid}$: reject. // Invalid token
4: If $\mathsf{md}_d = \perp$: reject. // No metadata in delivery token
5: Else, set $\ell_t \leftarrow \ell_t'$ and $\ell_d \leftarrow \ell_d'$ and accept report.

report is generated and stored by the client. At the end of the epoch, the client sends all stored reports to the Broker via the anonymizing proxy, as specified in Protocol 4. The client includes the signed tokens obtained in Protocols 2 & 3. Upon receiving a report, the Broker verifies the tokens, discarding all reports with invalid tokens. All reports with valid and a yet-unspent token are processed by the Broker.

5.3 The AdVeil ecosystem

AdVeil focuses on *general* targeted advertising, with support for multiple targeting and metrics strategies.

Targeting strategies. By giving the Broker control over the choice of ANN search data structure and real-time ad selection logic, AdVeil does not restrict the Broker to a specific targeting strategy. Types of targeting that can be supported in AdVeil include:

- **Contextual** targeting can bypass clients entirely by having the Publisher engage in the protocols. Users still see and interact with ads, but are not involved in any other aspect of the pipeline.
- **Behavioral** targeting is supported via the profile building function $\mathcal{F}_{\mathsf{profile}}$, which can locally observe information about users' browsing habits to build their profile feature vector. Protocol 2 can be run periodically to retrieve a new set of targeted ads following changes to the user's profile. Retargeting, or preferentially displaying ads to users who have had prior interactions with an Advertiser, is one example of behavioral targeting that AdVeil can support.

The only requirement for targeting strategies is that they are fully local. Users must not make any requests other than those specified in Protocol 2, Protocol 3, and Protocol 4 as part of any targeting mechanism. Making such requests can reveal information that might link a user to personal data or the ads they see.

Real-time delivery. AdVeil delivers ads over any anonymizing proxy. We require delivery to run on demand, rather than periodically as is done with targeting, primarily to support real-time bidding. For example, using Tor, AdVeil's users can—in real time—request an ad from the targeted category and have advertisers bid for the slot. Because the delivery and reporting requests are unlinkable from the targeting request, delivering different ads for the same category does not

compromise unlinkability (we elaborate in Section 6).

Measurement strategies. Reports in AdVeil are *individual* without being *linkable*. The Broker learns precisely which ads were seen and how often, without ever learning who saw them. This allows the Broker to support a variety of measurement strategies including:

- **Impressions** are reported when users *see* an ad.
- **Clicks** are reported when users *click* on an ad.
- **Conversions** are reported when a user engages with the Advertiser after clicking on an ad. The local client is capable of observing when an ad click generates a conversion and creates the resulting report (similar to conversion tracking in Safari [117]).

All reports in AdVeil contain the ad delivered and interaction type (e.g., view or click). AdVeil supports reporting on arbitrary additional data but, as discussed in Section 3.2, determining what additional data can be reported privately is an orthogonal problem.

6 Security arguments

Here, we analyze the security of AdVeil. We frame our analysis in terms of *user unlinkability* and *fraud prevention*, the requirements from our threat model.

6.1 User unlinkability

We recall that the security guarantee of AdVeil, as established in Section 3.2, is *unlinkability* between users and their personal data, including which ads they see and interact with. The crux of the unlinkability argument lies in analyzing the combination of targeting (where the Broker learns the client's identity; required for effective fraud prevention) and delivery/reporting (where the client's identity is hidden by the anonymizing proxy). Concretely, we must ensure that the Broker cannot deviate in targeting to link the client in delivery or reporting, without also compromising its own goals (and violating the *rationality* assumption; see Section 3.2).

Theorem 1 (Unlinkability)**.** *Fix a reporting epoch. The set of reports recovered through Protocol 4 is unlinkable from the set of clients that submitted them—as well as prior executions of Protocol 2 and Protocol 3—conditioned on:*

1. *the user and client not explicitly or implicitly leaking personally identifying information to any party ,*
2. *the Broker being rational and not denying service to honest users or sabotaging fraud prevention, and*
3. *the privacy of PIR [69], soundness of the vector commitment [32, 71, 82], unlinkability of one-time-use tokens [67, 96, 110], and the client anonymity property of the anonymizing proxy [20], being true under their respective security assumptions.*

Proof (sketch). See Appendix A.2 for full proof. At a high level, we show that a malicious Broker can only partition users into two sets: those with valid tokens and those with invalid tokens. If the Broker exploits tokens to separate a small subset of victim users, rather than for fraud prevention, it must necessarily compromise either its billing metrics or the covertness of its fraud prevention. By using one type of token for victims, the Broker must either (1) deny service to bots entirely, (2) deny service to all honest users not in its victim subset, or (3) issue valid tokens to *both* bots and all honest users excepting the users in its victim subset. In case (1), the Broker loses covertness of fraud detection allowing bots to rapidly evolve their evasion strategies. In case (2), the Broker loses the ability to bill for all non-victim users. In case (3), the Broker can no longer distinguish honest reports from fraudulent reports, resulting in invalid billing. As a result, a rational (monetarily-driven) Broker is incentivized to follow the protocol ensuring the unlinkability property of AdVeil. □

Cross-epoch unlinkability If only a subset of users participate in each epoch, then AdVeil cannot provide full unlinkability across epochs due to intersection attacks [39, 40, 75, 77, 118]. Intersection attacks across epochs are possible because users are correlated with the ads they see. Over time, the Broker can infer the relationship between a user and the ads they are shown by observing the *intersection* of epochs in which the user participated, even though the unlinkability property is satisfied *within* each epoch. That is, ads that appear most frequently alongside a certain user are likely to be the ads that the user reported on. This leakage is small, but is not resolvable without holding either the set of users or ads constant across all epochs—neither of which is reasonable for an internet-scale system.

6.2 Fraud prevention

Recall that the requirements for fraud prevention (detailed in Section 3.2), are that the Broker (1) flags all targeting requests it believes to be fraudulent and (2) only includes unique, valid reports in Protocol 4.

Theorem 2 (Fraud Prevention)**.** *Assuming the unforgeability property of anonymous tokens [67, 96, 110], each report recovered by the Broker through Protocol 4:*

1. *includes a valid token signature output in Protocol 2,*
2. *is unique in the set of all reports across epochs, and*
3. *is associated with an ad delivered in Protocol 3.*

Proof. (1) comes from the unforgeability property of the anonymous tokens; no client can forge a valid token signature without knowledge of the secret signing key held by the Broker [67], thus clients may only obtain valid token signatures via Protocol 2. (2) is likewise guaranteed by the anonymous

token unforgeability property and the list ℓ of all redeemed tokens maintained in Protocol 4 (see Privacy Pass [42] for more details and optimizations). Similarly, (3) is due to the *public* metadata issued by the Broker with the token signature in Protocol 3. Unforgeability [96, 110] guarantees that no client can produce a valid token with metadata indicating an ad other than the one that was delivered through Protocol 3. To expand on token validity, only clients marked as "human" in Protocol 2 are given valid signatures on the token. However, the clients themselves cannot determine the validity of their token signature (see Appendix C) which provides covert fraud detection. □

7 Evaluation

We implement a prototype of AdVeil and measure its end-to-end performance in a networked deployment. We evaluate the computational overhead of the Broker when serving client requests and the end-to-end latency of targeting and delivery on the client. We note that we do not compare AdVeil *quantitatively* with existing work in privacy-preserving advertising. Prior proposals are either of a theoretical/qualitative nature [63, 115], only solve one aspect of the pipeline under different assumptions [53, 57, 108], or have incomparable approaches [55, 57, 107]. We *qualitatively* compare all these systems with AdVeil in Section 8. In particular, as highlighted by Table 8, prior designs provide privacy by relying on a trusted third party in a way that limits the scalability of the system to the capabilities of the third party, while AdVeil's scalability depends only on the Broker and the anonymizing proxy. The anonymizing proxy can be instantiated by any variety of scalable systems such as Tor [43] or I2P [10] (both of which are already widely deployed, with millions of users [11, 73]).

Implementation. We implement AdVeil in Go (v1.13) and C++17 in approximately 2,000 lines of code. Our implementation is open source [4]. We use the Microsoft SealPIR library [17] to instantiate single-server PIR. Our implementation of anonymous one-time-use tokens is partially based on Cloudflare's Privacy Pass code [42]. We assume the Broker commits to the targeting data structure hash table using the vector commitment of Libert and Yung [71], which requires a one-time trusted setup but has very succinct opening proofs of 48 B. Succinct proofs improve the performance of targeting given that the user retrieves the proof in conjunction with the value from the hash table via PIR.

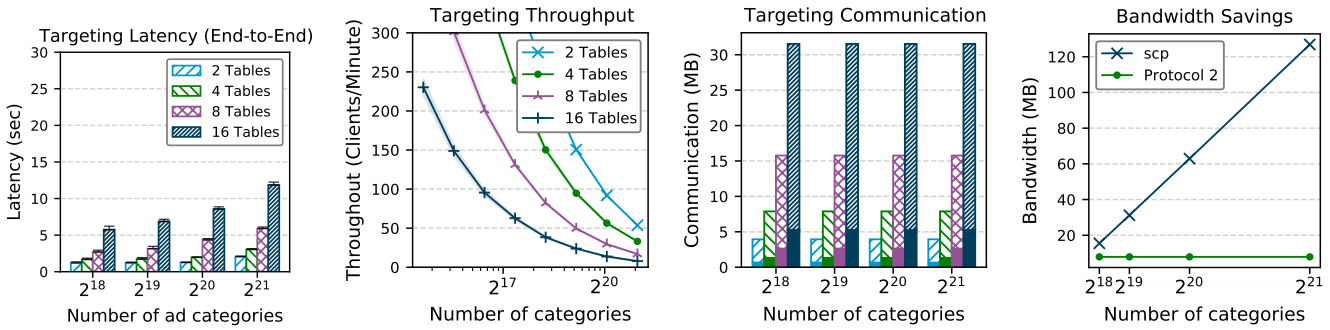
Environment. We deploy AdVeil on an Amazon EC2 server instance for the Broker and a MacBook Pro for the client. The Broker's server runs on `c4.4xlarge` VM (16 vCPUs @ 2.9 GHz; 32 GB RAM) with an hourly cost of $0.80 as of January 2022 [15]. We run the client on a MacBook Pro (8 CPUs; 32 GB RAM). We measure a throughput of 85 Mbit/s download and 33 Mbit/s upload using the `speedtest-cli` tool [98].

Parameters. The tunable parameters in AdVeil include the number of unique ad targeting categories, the size (in kilobytes) of each ad, and the number of hash tables used in the targeting data structure (Section 5.1). The total number of categories has a direct impact on network bandwidth and server processing time due to PIR (see Section 4.1). However, the total number of *ads* does not impact AdVeil's performance. Increasing the number hash tables in the targeting data structure improves accuracy (Section 7.1.1), but also increases the total number of parallel PIR queries required (Protocol 2). To evaluate AdVeil, we select parameters to mirror those of other ANN search data structure accuracy evaluations [16, 74, 95]. We vary the number of hash tables from $L = 2$ to 16, set each hash table to have N keys (i.e., total number of unique ad categories), and cap the size of each hash table bucket to at most one category. We use the LSH-multi-probing [74] optimization (which allows for retrieval of multiple candidates from the same hash table) in conjunction with batch-PIR [60, 95], following a similar evaluation performed in Servan-Schreiber et al. [95]. This setup and techniques are standard and are **not** novel to AdVeil.

Data. Real-world ad features are proprietary. Instead, we evaluate AdVeil using four open-source datasets frequently used in ANN search benchmarks [2]. (A similar compromise is made in the evaluation of Google's FLoC proposal [114].) We use the DEEP1B, GIST, MNIST, and SIFT) datasets, which have 10,000,000, 1,000,000, 64,000, and 1,000,000 unique feature vectors (i.e., "categories"), respectively. The performance of PIR (which is used to query the hash tables) is not impacted by the underlying data distribution. These datasets serve only to determine the number of hash tables (L) required for good accuracy. We set the number of LSH multi-probes [74, 95] (keys queried per hash table) to 5 for all experiments. We use dimensionality reduction [61] to map all feature vectors to $d = 50$ dimensions, so as to have a universal performance evaluation. We find that on real world datasets, it suffices to have $L = 4$ to achieve over 95% accuracy of finding the approximate nearest neighbor (i.e., the closest targeting category). We set the approximation factor to 2 (the result is within 2× of the true nearest neighbor [16]).

Ad sizes. Common online banner ads [51] (which account for 89% of ads on some platforms [76, 80]) are typically under 150 kB in size. Video ads are typically under 4 MB in size [80]. Ad size only impacts delivery as all other protocols operate on fixed-size category IDs.

Methodology. Unless otherwise stated, we run each experiment 10 times and report mean and 95% confidence interval over the trials. We parallelize computation on the servers when possible.



(a) Client end-to-end latency (seconds) for a targeting query. Latency is primarily impacted by Broker computation time (computing query answers), but also affected by ping time and client network bandwidth.

(b) Broker targeting throughput (clients per minute) with 16 cores. Throughput is only impacted by PIR query answer computation (Broker computation is linear in the number of ads in the database).

(c) Client-Broker communication for targeting (PIR query and response size). Bottom solid areas of each bar indicate fraction of upload communication. Same legend as Figure 3a.

(d) Communication comparison between "naive" PIR (retrieving the entire ANN search data structure using `scp`) and Protocol 2, which uses SealPIR to reduce communication. We set $L = 4$.

Figure 3: Evaluation of ad targeting in AdVeil. We report end-to-end latency as measured on the client machine. Communication is measured as the network overhead between the client and Broker. Throughput measures the raw computational processing throughput of the Broker's server (parallelized across 16 cores) to answer client targeting requests. Shaded regions and error bars represent a 95% confidence.

Protocol	Timing	Latency	Communication
Targeting	Periodic	1.6	8 MB
Delivery	On-demand	3 s	7 kB
Reporting	On-demand	—	500 B

Table 4: Summary of our prototype evaluation. For the targeting protocol, we have $L = 4$ hash tables, 5 LSH multiprobes per table, and one million targeting categories. For delivery, we use 5 kB ads.

Dataset	Num. categories	Accuracy	Num. tables (L)
DEEP1B	10,000,000	98 %	4
GIST	1,000,000	95 %	4
MNIST	64,000	97 %	3
SIFT	1,000,000	95 %	4

Table 5: Number of hash tables (L) for $\geq 95\%$ accuracy.

7.1 Results

In this section, we describe our evaluation results for targeting, delivery, and metric recovery in terms of processing time, latency, and communication. We summarize the results of our evaluation in Table 4.

7.1.1 Targeting accuracy

ANN search accuracy is typically measured through *recall*: the fraction of approximate nearest neighbors found over the total number of queries performed. While AdVeil is agnostic to the ANN search parameters (e.g., the LSH family used) we measure the accuracy of nearest neighbor queries as a function of the number of hash tables used in the ANN search. In Table 5, we report the number of hash tables required to achieve $\geq 95\%$ accuracy for each dataset. We find that $L \geq 4$ hash tables (and 5 multi-probes [74]) is sufficient.

7.1.2 Client latency.

Targeting (Protocol 2) latency. Figure 3a shows the impact that the number of ad categories and targeting hash tables has on client latency. Latency ranges from a couple seconds with around 300,000 categories to around ten seconds when there are over 2,000,000 categories. Network throughput accounted for one to five seconds of end-to-end latency on the client (which has throughput comparable to a home WiFi network).

Delivery (Protocol 3) latency. Figure 6 shows the relative performance of Tor vs. single and two-server PIR for delivering ad payloads. We take the mean latency for downloading a 50 kB and 1 MB file from Tor metrics [73] data for February-May 2021. Delivery latency for a 50 kB ad (e.g., a small image) through Tor is approximately one second while latency for a 1 MB ad (e.g., 5 second 480p video) is approximately three seconds. To illustrate the impracticality of using PIR for delivering ads, we evaluate SealPIR over 500 B to 1 kB ads while varying the total number of ads in the database. We also consider two-server PIR, which is more concretely efficient in practice [25, 113]. Our results show that both single-server and two-server (requiring non-colluding servers) PIR impose a high overhead for delivery, even when only considering small ads. As such, we choose to focus on the use of the anonymizing proxy for ad delivery in AdVeil. We stress that *targeting* must use PIR as replacing it with Tor (or

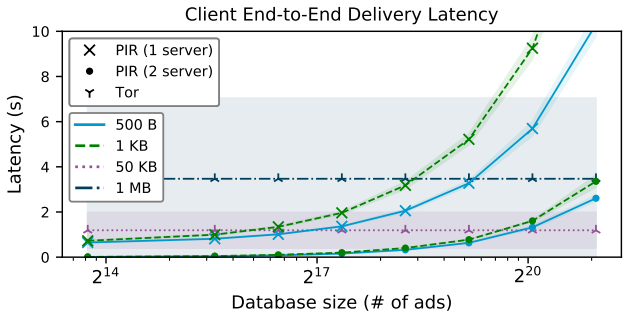


Figure 6: Client end-to-end ad delivery latency (seconds) using PIR, two-server PIR, and Tor. Latency for PIR queries is proportional to the size of the ad database. Latency for Tor is only dependent on individual ad size. Shaded region represents a 95% confidence interval (Tor has high variance in latency).

any anonymizing proxy) would require user data to be sent to the Broker during targeting, directly revealing PII.

7.1.3 Communication overhead

We report the total communication (in MB) exchanged between the Broker and the client when targeting ads in Figure 3c. Communication depends on database size and number of tables. The total communication remains under 32 MB with 16 tables and under 10 MB with 4 tables. We find that the majority of communication is from the PIR query response since the PIR queries themselves are of constant size with respect to the number of categories (due to SealPIR query compression [17]). We believe this communication to be reasonable for the average internet client, especially considering targeting happens once per epoch (e.g., once a day) and is "download-heavy," which aligns well with real-world networks [5]. Moreover, 10 MB of communication is approximately equivalent to visiting four mobile websites [104]. In Figure 3d, we contrast the bandwidth usage of SealPIR with "naive PIR" where the entire ANN search data structure is sent to the client.

7.1.4 Targeting throughput

We report the targeting throughput (in terms of clients per minute) of the Broker in Figure 3b. Throughput is limited by the computational overhead of processing the PIR queries over the targeting hash tables. With $L = 4$ hash tables and one million ad categories, targeting throughput was approximately 50 clients per minute. We note that the throughput is massively parallelizable; increasing linearly with the Broker's computational capacity. For example, with 100 million active users in the ecosystem (and one million unique categories), the Broker would need approximately 1,500 commodity virtual machines to refresh user targeting on a daily basis.

7.1.5 Reporting

Per-report computation on the Broker is light and consists of redeeming the two tokens attached to each report. In total, redeeming the tokens for a single report requires under 300 μs of processing time on one core (see Table 7). Thus, a single 16-core server can process upwards of 50,000 reports per second when parallelized.

7.1.6 Client and Broker microbenchmarks

We measure the client processing time for PIR queries and the anonymous tokens. Generating the PIR queries requires under one millisecond. Decryption of the PIR query response takes approximately two milliseconds. All client-side token processing (generating and unblinding) is under 300 μs. Server-side processing of tokens requires under 300 μs for all operations.

Client			
879 μs PIR.Query	2007 μs PIR.Recover	252 μs AT.GenToken	328 μs AT.Unblind
Broker			
278 μs AT.Sign (no metadata)	167 μs AT.Redeem (no metadata)	178 μs AT.Sign (pub metadata)	151 μs AT.Redeem (pub metadata)

Table 7: Microbenchmarks (in microseconds) for SealPIR (query and answer recovery) and anonymous one-time-use tokens (with and without public metadata).

7.1.7 Operational costs

We estimate the operational costs of running AdVeil. Our estimate focuses on an ad database and reports-per-epoch size of one million ad categories. We assume the Broker uses 4 LSH tables and use the AWS costs from our own evaluation – \$0.80/HR for the Broker's machine. Given this setup, the processing time of the Broker is approximately 1100 ms per execution of the targeting protocol (Protocol 2). (Note that Figure 3a includes network latency in addition to processing time.) The total costs to target and recover a report for one ($k = 1$) ad category is 0.02¢, as computed using Equation (1).

$$\underbrace{(\$0.80/\text{HOUR}) \cdot 1100 \text{ ms}}_{\text{targeting } k \text{ categories}} + \underbrace{k \cdot (\$0.80/\text{HOUR} \cdot 0.5 \text{ ms})}_{\text{token signing and redemption}} \quad (1)$$

Using the average revenue generated by an ad impression (≈ 0.20¢ [100]), we find that AdVeil's expected revenue is roughly 10× the cost of serving a single ad (on a non-enterprise server). Amortized over $k = 10$ ad categories targeted simultaneously (see Gionis et al. [49] for how ANN search trivially extends to k-nearest neighbors), the expected revenue is over 100× the cost of targeting. Hence, AdVeil can be deployed (with off-the-shelf infrastructure) and *still* result in net profit for the Broker.

Table 8: Comparison of AdVeil to related work on targeted advertising.

	Privacy		Correctness			Scalability		
	Trusted Third Party (TTP)	Targeting Data Sent to Broker	Targeting Accuracy	Fraud Prevention	Report Granularity	TTP Workload	TTP Availability	Simultaneous Requests
Adnostic [107]	Decryption Oracle	Contextual	Contextual+	Limited [i]	Aggregate	≪ Broker	≪ Broker	✓
ObliviAd [19]	TEE	None[ii]	Full Targeted	Full	Individual	N/A [iii]	N/A	✗
Privad [55]	Dealer	Broad Categories	Limited Targeted	Full	Individual	≈ Broker	= Broker [iv]	✓
Themis [87]	PoA Blockchain	None	Full Targeted	Full	Individual	≈ Broker [v]	≈ Broker [v]	✓
AdVeil	None	None	Full Targeted	Full	Individual	N/A	N/A	✓

[i] Adnostic does not perform fraud detection beyond guaranteeing that an individual report is for a single ad.
[ii] Complete targeting data is sent to the TEE held by the Broker.
[iii] Oblivad's TTP is a TEE held by the Broker. It does not correspond to a physically or administratively separate entity.
[iv] The TTP participates in processing every targeting request, on demand. In Privad it also participates in reporting.
[v] The blockchain authorities must validate every report on demand and selected users must participate in MPC to compute metrics.

8 Related work

There exists a large body of work on privacy-preserving advertising [19, 24, 53, 55–57, 63, 87, 107, 108, 114, 115]. We focus here on providing a comparison to other systems like AdVeil that cover the *complete* advertising pipeline. Table 8 compares the privacy, correctness, and scalability of AdVeil to other full-pipeline proposals.

Juels [63] is the first proposal for supporting privacy-preserving targeted advertising on the internet. Juels preserves user privacy when delivering ads in two primary ways: using a relaxed version of a mixnet (similar to AdVeil's use of an anonymizing proxy) and grouping users to hide which user from the group requested an ad. However, the work is theoretical and does not account for all aspects of online advertising or the complexity of today's advertising ecosystems.

Privad [55] provides a targeting model based on broad interest categories that are narrowed locally by the client. Privad relies on a *centralized* anonymizing proxy, referred to as the Dealer, to provide user privacy and enforce fraud prevention. The Dealer is assumed not to collude with the Broker and provides user privacy by mediating *all user communication*. Thus, the Broker learns the contents, but not the origin, of each request. To do this, the Dealer must sustain the load of the entire ad network as a non-colluding third party. Privad cannot trivally solve this issue by replacing the Dealer with an existing, distributed anonymizing proxy as this would leave their system without fraud prevention.

ObliviAd [19] relies heavily on a Trusted Execution Environment (TEE) to provide user privacy. The TEE is single-handedly responsible for *all stages of the advertising pipeline*, from ad targeting to unlinkability of reports and fraud prevention. To prevent the Broker from learning which ads are delivered to which clients, the TEE uses ORAM [101] to retrieve ads from the ad database. ORAM can only serve a single client request at a time; concurrent accesses would require a replica of the ORAM instance per-request [18, 101]. In addition, TEEs have seen a series of powerful attacks since ObliviAd was published [26, 29, 37, 111]. As a result, we do not believe that either issue is surmountable with today's instantiations of TEEs or ORAM [18, 37, 101].

Adnostic [107] primarily focuses on the targeting and reporting stages. Similarly to Privad, Adnostic performs targeting locally on the client. However, Adnostic only uses *contextual* features during targeting and does not make an effort to hide which ads are delivered to a user. Adnostic attempts to prevent the Broker from linking the delivery and reporting phases by aggregating reports using homomorphic encryption and zero-knowledge proofs rather than revealing them individually. Decryption of aggregate reports is handled by a TTP that checks the size of reports prior to decryption to ensure that only large groups of users are reported on. However, Adnostic makes no effort to hide users' browsing or click behavior from the Broker, leading us to agree with Privad's description of their Broker model as "honest-and-*not*-curious" [55].

Themis [87] is the Brave browser's contribution to private targeted advertising. It attempts to replace the role of the Broker with a permissioned blockchain, run by Publishers or foundations such as the EFF. Themis additionally supports payment to users for their interaction with ads. Privacy for payments, metrics, and auditing of the blockchain is based on a Proof of Authority protocol [3]. The attempted removal of the Broker, auditability, and user compensation are all interesting directions for the future of private advertising. Unfortunately, Themis achieves them by entirely offloading both targeting and delivery to clients. Each user must download both the targeting model and the *entire* database of ads and ad features to their local device.

9 Discussion and conclusions

In recent years, a fundamental conflict between privacy and advertising has emerged. AdVeil provides full compliance with technological and legislative best practices for user privacy *without* limiting the relevance of advertisements shown on the internet. Ultimately, we believe that AdVeil is a viable alternative to existing, non-private targeted advertising schemes that meets the needs of both users and ad networks.

Targeted Advertising can have serious negative side effects, from web tracking to discriminatory or manipulative practices [45]. In light of this, it seems challenging to advocate for even privacy-preserving targeted advertising. However, current legislative and technological efforts to prevent targeted ads or their ill effects have been markedly unsuccessful. Ad tech companies are more willing to fight legal battles and pay fines than they are to stop collecting user data [22, 65, 65, 102] and, despite anti-tracking measures, most peoples' browsers remain uniquely identifiable [28, 47, 78, 89, 120].

In AdVeil, users have transparency into the ad targeting process and control over any features they deem sensitive. It does not require ad-tech companies to track users for targeting data, instead giving control over the collection and use of this data back to the users themselves. While AdVeil is not a complete solution to discriminatory or manipulative advertising, we believe it provides a platform for future work in this direction.

Practicality motivated our decision to rely on anonymous communication over Tor for delivery of ads. PIR, while effective for private targeting, fails to support necessary components of advertising such as real-time bidding and cannot achieve the performance required for on-demand delivery of ads. This is true even for muti-server PIR which also introduces unrealistic trust and availability assumptions. We believe that requiring a separate entity with the same computational power and availability requirements as the ad network—a necessity for multi-server PIR—is unreasonable, especially considering the lack of support for real-time bidding.

Similarly, the ad network requires that any computational overhead imposed by AdVeil be *sustainable*. While AdVeil is more computationally expensive than non-private alternatives, we showed in Section 7.1.7 that it remains profitable. We additionally note that current, non-private advertising, while less expensive to run, carries a high cost in fines [93, 97] and damage to reputation [46] associated with privacy violations.

Conclusions. We presented AdVeil, a system for privacy-preserving targeted advertising that addresses usability and privacy concerns of existing work. In doing so, we introduce a novel method of targeting ads that provides strong privacy guarantees for users, while ensuring targeting accuracy that is on-par with existing systems. We provide an open-source prototype implementation with an end-to-end evaluation. Our results show the practical and economic feasibility of deploying AdVeil to provide user privacy, without compromising on the needs of the ad network.

10 Acknowledgements

We thank Peter Deutsch, Jules Drean, and Ben Murphy for for helpful discussion and feedback on this paper. In particular we would like to thank Patrick Zhang for his contributions that led to early ideas for this work. We also gratefully acknowledge Michael Specter and Nirvan Tyagi for pointing out some subtleties in secure use of ad targeting and anonymous tokens, respectively.

References

[1] Brave: The browser reimagined. https://brave.com/. Accessed March 2022.

[2] ANN benchmark datasets. https://github.com/erikbern/ann-benchmarks/. Accessed March 2022.

[3] Consensys Quorum. https://consensys.net/quorum/. Accessed March 2022.

[4] AdVeil prototype source code. https://github.com/sachaservan/adveil. Accessed March 2022.

[5] Speedtest global index. https://www.speedtest.net/global-index. Accessed March 2022.

[6] General Data Protection Regulation. https://gdpr-info.eu/, 2016. Accessed March 2022.

[7] California Consumer Privacy Act of 2018. https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?division=3.&part=4.&lawCode=CIV&title=1.81.5, 2018. Accessed March 2022.

[8] Onion Browser Button for Chrome. https://chrome.google.com/webstore/detail/onion-browser-button/fockhhgebmfjljjmjhbdgibcmofjbpca, 2021. Accessed March 2022.

[9] Onion Browser Button for Firefox. https://addons.mozilla.org/en-US/firefox/addon/tortm-browser-button/, 2021. Accessed March 2022.

[10] I2P: The Invisible Internet Project. https://geti2p.net/en/, 2022. Accessed March 2022.

[11] I2P metrics. https://i2p-metrics.np-tokumei.net/overview, 2022. Accessed March 2022.

[12] Shashank Agrawal and Srinivasan Raghuraman. KVaC: Key-value commitments for blockchains and beyond. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 839–869. Springer, 2020.

[13] Muhammad Ali, Piotr Sapiezynski, Aleksandra Korolova, Alan Mislove, and Aaron Rieke. Ad delivery algorithms: The hidden arbiters of political messaging. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 13–21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450382977. doi: 10.1145/3437963.3441801. URL https://doi.org/10.1145/3437963.3441801.

[14] Alphabet. Alphabet earnings report. https://abc.xyz/investor/, 2021. Accessed March 2022.

[15] Amazon. Amazon EC2 spot instances pricing. https://aws.amazon.com/ec2/spot/pricing/. Accessed March 2022.

[16] Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. Approximate nearest neighbor search in high dimensions. *arXiv preprint arXiv:1806.09823*, 7, 2018.

[17] Sebastian Angel, Hao Chen, Kim Laine, and Srinath Setty. PIR with compressed queries and amortized query processing. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 962–979. IEEE, 2018.

[18] Gilad Asharov, Ilan Komargodski, Wei-Kai Lin, Kartik Nayak, Enoch Peserico, and Elaine Shi. Optorama: Optimal oblivious RAM. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 403–432. Springer, 2020.

[19] Michael Backes, Aniket Kate, Matteo Maffei, and Kim Pecina. Obliviad: Provably secure and practical online behavioral advertising. In *2012 IEEE Symposium on Security and Privacy*, pages 257–271. IEEE, 2012.

[20] Michael Backes, Aniket Kate, Praveen Manoharan, Sebastian Meiser, and Esfandiar Mohammadi. AnoA: A framework for analyzing anonymous communication protocols. In *2013 IEEE 26th Computer Security Foundations Symposium*, pages 163–178. IEEE, 2013.

[21] Josh Daniel Cohen Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Yale University, 1987.

[22] Stephanie Bodoni. Facebook to fight Belgian ban on tracking users (and even non-users). https://www.bloomberg.com/news/articles/2019-03-27/facebook-attack-of-belgian-order-on-user-tracking-gets-hearing, 2019. Bloomberg.

[23] Dan Boneh, Benedikt Bünz, and Ben Fisch. Batching techniques for accumulators with applications to IOPs and stateless blockchains. In *Annual International Cryptology Conference*, pages 561–586. Springer, 2019.

[24] Sanaz Taheri Boshrooyeh, Alptekin Küpçü, and Öznur Özkasap. PPAD: Privacy preserving group-based advertising in online social networks. In *2018 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 1–9. IEEE, 2018.

[25] Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 337–367. Springer, 2015.

[26] Ferdinand Brasser, Urs Müller, Alexandra Dmitrienko, Kari Kostiainen, Srdjan Capkun, and Ahmad-Reza Sadeghi. Software grand exposure: SGX cache attacks are practical. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, Vancouver, BC, August 2017. USENIX Association.

[27] Andrei Z Broder. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pages 21–29. IEEE, 1997.

[28] Bill Budington. Panopticlick: Fingerprinting your web presence. San Francisco, CA, January 2016. USENIX Association.

[29] Jo Van Bulck, Marina Minkin, Ofir Weisse, Daniel Genkin, Baris Kasikci, Frank Piessens, Mark Silberstein, Thomas F. Wenisch, Yuval Yarom, and Raoul Strackx. Foreshadow: Extracting the keys to the intel SGX kingdom with transient out-of-order execution. In *27th USENIX Security Symposium (USENIX Security 18)*, page 991–1008, Baltimore, MD, August 2018. USENIX Association. ISBN 978-1-939133-04-5.

[30] Interactive Advertising Bureau. Internet advertising revenue report. https://www.iab.com/insights/internet-advertising-revenue-report/, April 2021. Accessed March 2022.

[31] José González Cabañas, Ángel Cuevas, and Rubén Cuevas. Unveiling and quantifying Facebook exploitation of sensitive personal data for advertising purposes. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 479–495, Baltimore, MD, August 2018. USENIX Association. ISBN 978-1-939133-04-5.

[32] Dario Catalano and Dario Fiore. Vector commitments and their applications. In *International Workshop on Public Key Cryptography*, pages 55–72. Springer, 2013.

[33] Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388, 2002.

[34] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private Information Retrieval. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 41–50. IEEE, 1995.

[35] Benny Chor, Niv Gilboa, and Moni Naor. *Private information retrieval by keywords*. Citeseer, 1997.

[36] Arka Rai Choudhuri, Matthew Green, Abhishek Jain, Gabriel Kaptchuk, and Ian Miers. Fairness in an unfair world: Fair multiparty computation from public bulletin boards. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 719–728, 2017.

[37] Victor Costan and Srinivas Devadas. Intel SGX explained. *IACR Cryptol. ePrint Arch.*, 2016(86):1–118, 2016.

[38] Bennett Cyphers. Google's FLoC is a terrible idea. https://www.eff.org/deeplinks/2021/03/googles-floc-terrible-idea, 2021. Accessed March 2022.

[39] George Danezis and Andrei Serjantov. Statistical disclosure or intersection attacks on anonymity systems. In *International Workshop on Information Hiding*, pages 293–308. Springer, 2004.

[40] George Danezis, Claudia Diaz, and Carmela Troncoso. Two-sided statistical disclosure attack. In *International Workshop on Privacy Enhancing Technologies*, pages 30–44. Springer, 2007.

[41] Amit Datta, Anupam Datta, Jael Makagon, Deirdre K Mulligan, and Michael Carl Tschantz. Discrimination in online advertising: A multidisciplinary inquiry. *Conference on Fairness, Accountability, and Transparency*, 81:20–34, 2018.

[42] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *Proceedings on Privacy Enhancing Technologies*, 2018(3):164–180, 2018.

[43] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004.

[44] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[45] Gilad Edelman. Why don't we just ban targeted advertising? https://www.wired.com/story/why-dont-we-just-ban-targeted-advertising/, 2020. Accessed March 2022.

[46] Peter Elkind, Jack Gillum, and Craig Silverman. How Facebook undermines privacy protections for its 2 billion WhatsApp users. https://www.propublica.org/article/how-facebook-undermines-privacy-protections-for-its-2-billion-whatsapp-users, 2021. Accessed October 2021.

[47] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 1388–1401, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341394. doi: 10.1145/2976749.2978313.

[48] Juan Garay, Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Rational protocol design: Cryptography against incentive-driven adversaries. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 648–657. IEEE, 2013.

[49] Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *Vldb*, volume 99, pages 518–529, 1999.

[50] Bill Goodwin and Sebastian Klovig Skelton. Facebook's privacy game – how Zuckerberg backtracked on promises to protect personal data. https://www.computerweekly.com/feature/Facebooks-privacy-U-turn-how-Zuckerberg-backtracked-on-promises-to-protect-personal-data, 2019. Accessed March 2022.

[51] Google AdSense. Guide to ad sizes. https://support.google.com/adsense/answer/6002621. Accessed March 2022.

[52] Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: aggregating proofs for multiple vector commitments. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 2007–2023, 2020.

[53] Matthew Green, Watson Ladd, and Ian Miers. A protocol for privately reporting ad impressions at scale. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1591–1601, 2016.

[54] Adam Groce, Jonathan Katz, Aishwarya Thiruvengadam, and Vassilis Zikas. Byzantine agreement with a

rational adversary. In *International Colloquium on Automata, Languages, and Programming*, pages 561–572. Springer, 2012.

[55] Saikat Guha, Bin Cheng, and Paul Francis. Privad: Practical privacy in online advertising. In *USENIX conference on Networked systems design and implementation*, pages 169–182, 2011.

[56] Leon J Helsloot, Gamze Tillem, and Zekeriya Erkin. AHEad: privacy-preserving online behavioural advertising using homomorphic encryption. In *2017 IEEE Workshop on Information Forensics and Security (WIFS)*, pages 1–6. IEEE, 2017.

[57] Leon J Helsloot, Gamze Tillem, and Zekeriya Erkin. BAdASS: Preserving privacy in behavioural advertising with applied secret sharing. In *International Conference on Provable Security*, pages 397–405. Springer, 2018.

[58] Basileal Imana, Aleksandra Korolova, and John Heidemann. Auditing for discrimination in algorithms delivering job ads. WWW '21, page 3767–3778, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383127. doi: 10.1145/3442381.3450077. URL https://doi.org/10.1145/3442381.3450077.

[59] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, 1998.

[60] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Batch codes and their applications. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 262–271, 2004.

[61] William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space 26. *Contemporary mathematics*, 26, 1984.

[62] Seb Joseph. 'contextual targeting is going to be the new black': As IDFA restrictions loom, advertisers brace for the fallout. https://digiday.com/media/contextual-targeting-is-going-to-be-the-new-black-as-idfa-restrictions-loom-advertisers-brace-for-the-fallout/, 2020. Accessed March 2022.

[63] Ari Juels. Targeted advertising... and privacy too. In *Cryptographers' Track at the RSA Conference*, pages 408–424. Springer, 2001.

[64] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. GAZELLE: A low latency framework for secure neural network inference. In *27th USENIX Security Symposium (USENIX) Security 18)*, pages 1651–1669, 2018.

[65] Michael Kaplan. Facebook and Google are already facing lawsuits under new data rules. https://money.cnn.com/2018/05/25/technology/gdpr-compliance-facebook-google/index.html, 2018. Accessed March 2022.

[66] Kai Kaspar, Sarah Lucia Weber, and Anne-Kathrin Wilbers. Personally relevant online advertisements: Effects of demographic targeting on visual attention and brand evaluation. *PloS one*, 14(2):e0212419, 2019.

[67] Ben Kreuter, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. Anonymous tokens with private metadata bit. In *Annual International Cryptology Conference*, pages 308–336. Springer, 2020.

[68] Nishant Kumar, Mayank Rathee, Nishanth Chandran, Divya Gupta, Aseem Rastogi, and Rahul Sharma. Cryptflow: Secure tensorflow inference. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 336–353. IEEE, 2020.

[69] Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proceedings 38th annual symposium on foundations of computer science*, pages 364–373. IEEE, 1997.

[70] Ben Laurie. Certificate transparency. *Communications of the ACM*, 57(10):40–46, 2014.

[71] Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In *Theory of Cryptography Conference*, pages 499–517. Springer, 2010.

[72] Robinson & Cole LLP. California's Consumer Privacy Rights Act: Opt-out rights and data profiling. https://www.natlawreview.com/article/california-s-consumer-privacy-rights-act-opt-out-rights-and-data-profiling, 2021. Accessed March 2022.

[73] Karsten Loesing, Steven J. Murdoch, and Roger Dingledine. A case study on measuring statistical data in the Tor anonymity network. In *Proceedings of the Workshop on Ethics in Computer Security Research (WECSR 2010)*, LNCS. Springer, January 2010.

[74] Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. Multi-probe lsh: efficient indexing for high-dimensional similarity search. In *33rd International*

Conference on Very Large Data Bases, VLDB 2007, pages 950–961. Association for Computing Machinery, Inc, 2007.

[75] Nayantara Mallesh and Matthew Wright. The reverse statistical disclosure attack. In *International Workshop on Information Hiding*, pages 221–234. Springer, 2010.

[76] Match2One. Top banner sizes: The most effective banners of 2020. https://www.match2one.com/blog/standard-banner-sizes/, July 2020. Accessed March 2022.

[77] Nick Mathewson and Roger Dingledine. Practical traffic analysis: Extending and resisting statistical disclosure. In *International Workshop on Privacy Enhancing Technologies*, pages 17–34. Springer, 2004.

[78] Arunesh Mathur, Jessica Vitak, Arvind Narayanan, and Marshini Chetty. Characterizing the use of browser-based blocking extensions to prevent online tracking. In *Fourteenth Symposium on Usable Privacy and Security (SOUPS 2018)*, pages 103–116, Baltimore, MD, August 2018. USENIX Association. ISBN 978-1-939133-10-6.

[79] S. C. Matz, M. Kosinski, G. Nave, and D. J. Stillwell. Psychological targeting as an effective approach to digital mass persuasion. *Proceedings of the National Academy of Sciences*, 114(48):12714–12719, 2017. ISSN 0027-8424. doi: 10.1073/pnas.1710966114.

[80] Advance Media. Digital ad specs. https://www.advancemediany.com/wp-content/uploads/2018/11/DigitalAdSpecs_20181108.pdf, May 2021. Accessed March 2022.

[81] Marcela S Melara, Aaron Blankstein, Joseph Bonneau, Edward W Felten, and Michael J Freedman. CONIKS: Bringing key transparency to end users. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 383–398, 2015.

[82] Ralph C Merkle. A digital signature based on a conventional encryption function. In *Conference on the theory and application of cryptographic techniques*, pages 369–378. Springer, 1987.

[83] Meta. Meta earnings report. https://investor.fb.com/investor-news/press-release-details/2022/Meta-Reports-Fourth-Quarter-and-Full-Year-2021-Results/default.aspx, 2021. Accessed March 2022.

[84] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[85] Kate O'Flaherty. This is why people no longer trust Google and Facebook with their data. https://www.forbes.com/sites/kateoflahertyuk/2018/10/10/this-is-why-people-no-longer-trust-google-and-facebook-with-their-data, 2018. Accessed March 2022.

[86] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *Annual international cryptology conference*, pages 31–53. Springer, 1992.

[87] Gonçalo Pestana, Iñigo Querejeta-Azurmendi, Panagiotis Papadopoulos, and Benjamin Livshits. THEMIS: Decentralized and trustless ad platform with reporting integrity, 2020.

[88] Angelisa C. Plane, Elissa M. Redmiles, Michelle L. Mazurek, and Michael Carl Tschantz. Exploring user perceptions of discrimination in online targeted advertising. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 935–951, Vancouver, BC, August 2017. USENIX Association. ISBN 978-1-931971-40-9.

[89] Gaston Pugliese, Christian Riess, Freya Gassmann, and Zinaida Benenson. Long-term observation on browser fingerprinting: Users' trackability and perspective. *Proceedings on Privacy Enhancing Technologies*, 2020 (2):558 – 577, 01 Apr. 2020. doi: https://doi.org/10.2478/popets-2020-0041.

[90] Anand Rajaraman and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2011.

[91] Deevashwer Rathee, Mayank Rathee, Nishant Kumar, Nishanth Chandran, Divya Gupta, Aseem Rastogi, and Rahul Sharma. Cryptflow2: Practical 2-party secure inference. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 325–342, 2020.

[92] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

[93] Sam Schechner. Google, Amazon fined $163 million as France takes hard line on privacy. https://www.wsj.com/articles/google-amazon-fined-163-million-as-france-takes-hard-line-on-privacy-11607601278, 2020. Accessed October 2021.

[94] Phil Schraeder. Contextual advertising leads the way in a post cookie reality. https://adage.com/article/gumgum/contextual-advertising-leads-way-

post-cookie-reality/2293976, 2020. Accessed March 2022.

[95] Sacha Servan-Schreiber, Simon Langowski, and Srinivas Devadas. Private nearest neighbor search with sublinear communication and malicious security. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022.

[96] Tjerand Silde and Martin Strand. Anonymous tokens with public metadata and applications to private contact tracing. *IACR Cryptol. ePrint Arch.*, 2021:203, 2021.

[97] Natasha Singer. What you don't know about how Facebook uses your data. https://www.nytimes.com/2018/04/11/technology/facebook-privacy-hearings.html, 2018. Accessed March 2022.

[98] Speedtest. Speedtest CLI. https://www.speedtest.net/apps/cli. Accessed March 2022.

[99] Kevin Springborn and Paul Barford. Impression fraud in on-line advertising via pay-per-view networks. In *22nd USENIX Security Symposium (USENIX Security 13)*, pages 211–226, 2013.

[100] Statista. Social media advertising cost-per-mille (CPM) worldwide from 2nd quarter 2018 to 4th quarter 2019. https://www.statista.com/statistics/873631/social-media-advertising-cpm/, 2020. Accessed March 2022.

[101] Emil Stefanov, Marten van Dijk, Elaine Shi, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. Path ORAM: An extremely simple oblivious RAM protocol. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, page 299–310, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450324779. doi: 10.1145/2508859.2516660.

[102] Jonathan Stempel. Google faces $5 billion lawsuit in U.S. for tracking "private" internet use. https://www.reuters.com/article/us-alphabet-google-privacy-lawsuit/google-faces-5-billion-lawsuit-in-u-s-for-tracking-private-internet-use-idUSKBN23933H, 2020. Accessed March 2022.

[103] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[104] John Teague. Page weight: 2021: The web almanac by http archive. https://almanac.httparchive.org/en/2021/page-weight, 2021. HTTP Archive.

[105] Martin Thomson. Privacy preserving attribution for advertising. https://blog.mozilla.org/en/mozilla/privacy-preserving-attribution-for-advertising/, 02-22-2022. Accessed March 2022.

[106] Alin Tomescu, Yu Xia, and Zachary Newman. Authenticated dictionaries with cross-incremental proof (dis) aggregation. *IACR Cryptol. ePrint Arch.*, 2020:1239, 2020.

[107] Vincent Toubiana, Arvind Narayanan, Dan Boneh, Helen Nissenbaum, and Solon Barocas. Adnostic: Privacy preserving targeted advertising. In *Proceedings Network and Distributed System Symposium*, 2010.

[108] Minh-Dung Tran, Gergely Acs, and Claude Castelluccia. Retargeting without tracking. *arXiv preprint arXiv:1404.4533*, 2014.

[109] Theja Tulabandhula, Shailesh Vaya, and Aritra Dhar. Privacy-preserving targeted advertising. *arXiv preprint arXiv:1710.03275*, 2017.

[110] Nirvan Tyagi, Sofıa Celi, Thomas Ristenpart, Nick Sullivan, Stefano Tessaro, and Christopher A Wood. A fast and simple partially oblivious prf, with applications.

[111] Stephan van Schaik, Andrew Kwong, Daniel Genkin, and Yuval Yarom. SGAxe: How SGX fails in practice. https://sgaxe.com/files/SGAxe.pdf, 2020. Accessed March 2022.

[112] Matteo Varvello, Iñigo Querejeta Azurmendi, Antonio Nappa, Panagiotis Papadopoulos, Goncalo Pestana, and Ben Livshits. VPN0: A privacy-preserving decentralized virtual private network. *arXiv preprint arXiv:1910.00159*, 2019.

[113] Frank Wang, Catherine Yun, Shafi Goldwasser, Vinod Vaikuntanathan, and Matei Zaharia. Splinter: Practical private queries on public data. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pages 299–313, 2017.

[114] Web Incubator CG. FLOC. https://github.com/WICG/floc, 2021. Accessed March 2022.

[115] Web Incubator CG. TURTLEDOVE. https://github.com/WICG/turtledove, 2021. Accessed March 2022.

[116] Miranda Wei, Madison Stamos, Sophie Veys, Nathan Reitinger, Justin Goodman, Margot Herman, Dorota Filipczuk, Ben Weinshel, Michelle L. Mazurek, and Blase Ur. What Twitter knows: Characterizing ad targeting practices, user perceptions, and ad explanations through users' own twitter data. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 145–162. USENIX Association, August 2020. ISBN 978-1-939133-17-5.

[117] John Wilander. Privacy preserving ad click attribution for the web. https://webkit.org/blog/8943/privacy-preserving-ad-click-attribution-for-the-web/, 2019. Accessed March 2022.

[118] David Isaac Wolinsky, Ewa Syta, and Bryan Ford. Hang with your buddies to resist intersection attacks. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 1153–1166, 2013.

[119] Charles Wright and Mayank Varia. Crypto crumple zones: Enabling limited access without mass surveillance. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 288–306. IEEE, 2018.

[120] Ting-Fang Yen, Yinglian Xie, Fang Yu, Roger Peng Yu, and Martin Abadi. Host fingerprinting and tracking on the web: Privacy and security implications. In *NDSS*, volume 62, page 66, 2012.

[121] Ruofei Zhang and Ying Cui. Similarity function in online advertising bid optimization, August 4 2011. US Patent App. 12/698,463.

[122] Xingquan Zhu, Haicheng Tao, Zhiang Wu, Jie Cao, Kristopher Kalish, and Jeremy Kayne. *Fraud prevention in online digital advertising*. Springer, 2017.

A Proofs

A.1 Privacy of individual protocols

We first show that the targeting, delivery, and reporting protocols, *individually* provide unlinkability from the user (specifically, the user's targeting profile).

Claim 1. *Protocol 2 (targeting) reveals the identity of the client but no other information to the Broker, conditioned on the privacy requirement of the PIR scheme.*

Proof. The claim follows immediately from protocol inspection. Specifically, in Protocol 2 (targeting), the client only interacts with the Broker through a series of PIR queries which, by definition, hide the user's query (i.e., profile) from the Broker. The anonymous token issued in Protocol 2 is generated by the client independently of all user data and hence reveals no information on its own. □

Claim 2. *Protocol 3 (delivery) and Protocol 4 (reporting) reveal the ad category, ad delivered for the category, and report contents to the Broker, but not the identity of the client, conditioned on the client anonymity property of the anonymizing proxy [20].*

Proof. In Protocols 3 & 4, the client interacts with the Broker through the anonymizing proxy, which reveals the targeted category and which ad was served (resp. the contents of the report) but not which client it was served to (resp. which client submitted the report). This follows directly from the client anonymity property of the anonymizing proxy [20]. The anonymous tokens generated and sent by the client in both the delivery and the reporting protocols contain no identifiable information as they are computed independently of the user data (and are uniformly random). □

A.2 Proof of Theorem 1.

The unlinkability argument hinges on showing that a malicious (but rational) Broker cannot deviate in the targeting (Protocol 2) to link the client in delivery (Protocol 3), or reporting (Protocol 4).

At a high level, this follows from the Broker committing to the targeting hash tables in Protocol 1 (setup) and the total number of targeting categories, coupled with the unlinkability property of anonymous tokens. More formally, we must individually examine interactions taken by the client and the Broker in an epoch.

First, because the targeting hash tables are committed to in Protocol 1, a malicious Broker cannot change the targeting data structure between clients, which ensures that all clients retrieve the targeted category from the *same* targeting data structure, in each epoch. That is, the contents of the targeting data structure must be *consistent* across all clients. This prevents a malicious Broker from "tagging" a client by evaluating the PIR queries on different targeting data structures, depending on the client. This guarantee is upheld by the vector commitment proof returned to the clients with their PIR queries. If the Broker is capable of answering the PIR query with a valid proof (with respect to the commitment C_S) for a bucket value that is *not* in the queried hash table, with better than negligible probability in a security parameter, then the Broker is also capable of breaking the soundness property of the vector commitment with non-negligible probability [32, 71, 82].

Second, we examine the anonymous token (and the embedded public metadata) as a means for linking a client to a report. We show that exploiting anonymous tokens for linking clients to reports, while possible, is monetarily disincentivized and hence falls under irrational behavior. Specifically, because each token issued in Protocol 2 is set up to reveal *one* bit of information (valid or invalid), the Broker can only partition the anonymity set into two groups: clients with valid tokens and clients with invalid tokens (see Appendix C for additional details). This division is necessary for any scheme that allows the Broker to silently tag fraudulent requests for later identification. However, all tokens within these two sets are unlinkable from other tokens in their respective set by the properties of anonymous tokens [67] (also Appendix C); hence the Broker cannot link a valid (resp. invalid) token to a

prior targeting request.

A malicious Broker can still split a single client into their own group (by issuing only one valid token), allowing it to directly link their identity to the contents of their report. However, this sabotages fraud prevention, as the Broker must group all other users **and** all bots into the other set (invalid tokens), losing its ability to distinguish fraudulent reports and causing it to violate its billing arrangement with Advertisers and Publishers. Such a strategy is equivalent to denying service to all-but-one user, which is diametrically opposed to the Broker's goal of running an advertising ecosystem. Specifically, such a strategy results in significant monetary losses for the Broker and is thus *irrational*.

B Extended related work

FLoC [114], which ran on Chrome during 2021, focuses specifically on the targeting phase of the advertising pipeline. It assigns users to groups based on their browsing behavior – users with similar browsing habits and, presumably, interests are assigned to the same group. FLoC users then receive ads based on their group rather than their individual features. However, it is unclear what, if any, privacy FLoC provides as the process by which users are formed into groups is entirely opaque and these groups could be arbitrarily specific.

Turtledove [115] takes a similar approach to Privad where users subscribe to interest groups and can participate in local auctions to receive an ad based on these interest groups. However, Turtledove does not make any assumptions about the specificity of these groups or what metadata may be included in the auction. It only requires that group size meets an unspecified k-anonymity threshold. Additionally, Turtledove requires a trusted server to support the inclusion of real time external data into the local auction. User requests to this server are *not* required to conform to any k-anonymity bound. Between the specificity of interest groups, the blind trust in this server, and the lax restrictions on reporting, it is not clear that Turtledove provides its users with any more privacy than traditional advertising.

PPAD [24] is a group-based ad targeting system where the privacy for users is guaranteed relative to the *group* that a user belongs to. Each user is assigned to a group based on their attributes and hence reveals *coarse grained* interests of users. The Broker and a semi-trusted third party run a variant of a private-set intersection protocol (using shares of user-provided feature vectors) to determine which ads should be displayed to a group. The advantage of PPAD is that the Broker can target and deliver ads at a group level and can evaluate set intersection while the user is "offline" in order to serve ads when the user goes back online. However, the model of PPAD provides a tradeoff between targeting accuracy and privacy. Specifically, the more fine-grained the group selection, the more privacy leakage it incurs; PPAD does not provide an analysis of this leakage.

BAdASS [57] (and its precursor AHEAd [56]) are designed for online targeting and do not delve into other aspects of the advertising ecosystem such as reporting and fraud prevention. BAdASS uses a multi-party computation protocol executed between a group of honest-but-curious parties. Moreover, BAdASS requires splitting trust among a set of Demand-Side Platforms (DSPs) which manage content targeting in an ad network. Even a single malicious DSP can disrupt the correctness of the protocol (or worse yet de-anonymize the user) unless expensive zero-knowledge proofs are added to prevent deviations from protocol.

Tran *et al.* [108] is designed to address the *retargeting* aspect of online advertising. Their primary assumption is that the Broker (or ad exchange) will not collude with retargeting services. Clients engage in a protocol between the Broker and retargeter to fetch ads for products they have previously shown interest in (e.g., by adding a product to a shopping cart). The retargeter learns which ad was displayed but not which user requested it, while the Broker learns which user requested the ad but not which ad was retrieved via the retargeter. User privacy is ensured if the Broker and retargeter do not collude.

Tulabandhula *et al.* [109] propose a collection of functions for privacy-preserving association rule mining and recommendations. Their protocols work between a client and server, where the client stores the feature vector locally. Their results can be applied to AdVeil in the targeting process and may be useful for certain deployments.

AdScale [53] improves the reporting scheme proposed in Adnostic, but does not address other aspects of the pipeline. Users in Adscale respond with homomorphically encrypted reports that are aggregated by the Broker, but can only be decrypted by a designated trusted third party (TTP). This is intended to ensure that neither the Broker nor the TTP are individually capable of learning the plaintext responses of a single user. As a consequence, only aggregate information can be used for billing and targeting purposes.

C Anonymous token construction

For completeness, we describe the OSPP-without-NIZK construction of one-time-use anonymous tokens. Kreuter et al. [67] do not explicitly describe the construction that we present below, however, we note that it is a direct corollary of two constructions they do present (and is explicitly alluded to in their paper).

Motivation. In AdVeil, we require that only the verifier (i.e., the Broker) can generate a valid token, while anyone can generate an invalid token. Crucially, we must also guarantee that recipients of tokens *cannot tell whether the token is valid*

KeyGen(crs)

1 : **parse** crs = $(\mathbb{G}, g, h)$
2 : $(x, y) \leftarrow \mathbb{Z}_p^* \times \mathbb{Z}_p^*$
3 : pk := $(\mathbb{G}, g, h, g^x, h^y)$
4 : sk := (x, y)
5 : **return** (pk, sk)

Sign(sk, τ)

1 : **parse** sk = (x, y)
2 : $z \xleftarrow{R} \mathbb{Z}_p^*$
3 : $s \leftarrow H_z(\tau, z)$
4 : $w \leftarrow \tau^x s^y$
5 : **return** $\sigma = (z, w)$

TokenGen(pk)

1 : **parse** pk = $(\mathbb{G}, g, h, g^x)$
2 : $\tau \leftarrow \{0,1\}^\lambda$
3 : $t \leftarrow H(\tau)$
4 : $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p^*$
5 : $\tau \leftarrow (tg^{-\beta})^{\alpha^{-1}}$
6 : $r := (\alpha, \beta)$
7 : **return** (τ, τ, r)

Setup(1^λ)

1 : **return** crs := $(\mathbb{G}, g, h)$

Unblind(pk, τ, σ, r)

1 : **parse** $r = (\alpha, \beta)$,
$\sigma = (z, w)$,
pk = $(\mathbb{G}, g, h, g^x, h^y)$
2 : $u \leftarrow H_z(\tau, z)^\alpha h^\beta$
3 : $v \leftarrow w^\alpha (g^x h^y)^\beta$
4 : **return** $\sigma = (u, v)$

Redeem(sk, ℓ, τ, σ)

1 : **parse** sk = (x, y)
$\sigma = (u, v)$
2 : $t \leftarrow H_\tau(\tau)$
3 : $w \leftarrow t^x u^y$
4 : **if** $w = v$ **and** $\tau \notin \ell$ **then**
5 : $\quad \ell' \leftarrow \ell \cup \tau$
6 : $\quad$ **return** (valid, ℓ')
7 : **else return** (invalid, ℓ)

Figure 9: OSPP-without-NIZK anonymous one-time-use token construction providing 2-unlinkability with one bit of private metadata.

or invalid. This covertness property is required to avoid alerting bots that they have been detected by the Broker.

The challenge is that anonymous token constructions [42, 110] *explicitly* reveal token validity to the prover (specifically, in [42, 110] the verifier provides a NIZK proof of correct token signing). Kreuter et al. [67] provide a construction for an anonymous token with a *private* metadata bit (only readable to the verifier) and which does not require a NIZK of correct signing. While this does allow the Broker to identify and exclude "bot" reports, it does so using a metadata bit $b \in \{0, 1\}$ included in *valid* tokens. That is, there are *three* types of tokens, and thus three possible ways to divide clients, under this construction:

1. valid token signatures with $b = 0$,
2. valid token signatures with $b = 1$ and,
3. invalid token signatures.

This allows a malicious Broker to target a specific user by issuing only a single token with $b = 1$ while still excluding fraudulent requests using invalid tokens. In this way, the Broker is able to link the identity of a targeted user to their report contents without compromising on the accuracy of either metrics or fraud prevention.

To prevent this attack, we require a construction with at most *two* sets: clients with valid tokens and clients with invalid tokens. We observe that an implicit construction alluded to but not described by Kreuter et al. [67] gives us the required properties:

1. any party can generate invalid token signatures,
2. only the Broker can generate valid token signatures,
3. clients do not learn the validity of the token signature received from the verifier.

We call the construction OSPP-without-NIZK, following the naming convention of Kreuter et al. [67]. For simplicity, in Figure 9, we present the OSPP-without-NIZK construction *without* embedded public metadata but hypothesize that the construction is extendable to that regime following the transformation of Silde and Strand [96]. We note the resemblance to the OSPP [67, Construction 2] and PMBT-without-NIZK [67, Construction 5]. Let $\mathbb{G}$ be any prime order $p > 2^\lambda$ with generators g and h. For security, we assume that the Chosen-target Diffie–Hellman assumption holds in $\mathbb{G}$ [67].

Correctness of OSPP-without-NIZK. To see correctness, observe that

$$
\begin{aligned}
w &= t^x u^y \\
&= t^x (H_z(\tau, z)^\alpha h^\beta)^y \\
&= t^x H_z(\tau, z)^{\alpha y} h^{\beta y} \\
&= t^x g^{-\beta x} H_z(\tau, z)^{\alpha y} g^{\beta x} h^{\beta y} \\
&= t^x g^{-\beta x} H_z(\tau, z)^{\alpha y} (g^x h^y)^\beta \\
&= (t^{\alpha^{-1} x} g^{-\beta \alpha^{-1} x} H_z(\tau, z)^y)^\alpha (g^x h^y)^\beta \\
&= ((t^{\alpha^{-1}} g^{-\beta \alpha^{-1}})^x H_z(\tau, z)^y)^\alpha (g^x h^y)^\beta \\
&= (((tg^{-\beta})^{\alpha^{-1}})^x H_z(\tau, z)^y)^\alpha (g^x h^y)^\beta \\
&= (\tau^x H_z(\tau, z)^y)^\alpha (g^x h^y)^\beta \\
&= w^\alpha (g^x h^y)^\beta \\
&= v
\end{aligned}
$$

For security, we point to the definitions and proofs of Kreuter et al. [67]. The unlinkability (specifically 2-unlinkability [67, Definition 2]), unforgeability, and privacy of the token validity [67, Theorem 15] follow from the security analysis OSPP and PMBT-without-NIZK constructions of Kreuter et al. [67].

Application of OSPP-without-NIZK to AdVeil. Following the one-time setup and key generation, a client runs TokenGen to generate a new token with a targeting or delivery request. The blind token is sent to the Broker for signing. If the Broker believes the client to be a "human," then it generates a valid signature using Unblind. Otherwise, the Broker returns $\sigma = (z, w)$ with random $z, w \xleftarrow{R} \mathbb{Z}_p^*$ (invalid signature). The client

cannot differentiate between a valid and invalid signature [67, Theorem 15]. The client unblinds the signature using Unblind. The unblinded (and unlinkable) token and signature is given to the Broker in reporting. The Broker rejects all reports where Redeem outputs invalid. As mentioned earlier, the Broker can only partition clients based on the token validity, partitioning all clients into at most two anonymity sets.