Protected Audience Privacy Analysis

Martin Thomson, Mozilla, 2024-03

Google's Privacy Sandbox includes a number of advertising features. The most ambitious of these is Protected Audience. This proposal aims to support the use of cross-site information about user activity in targeted advertising without enabling the collection of that information. This analysis looks at whether Protected Audience achieves the privacy objectives it sets.

Summary

Protected Audience seeks to enable targeted advertising without the privacy costs that current approaches have.

The ideas behind Protected Audience have tremendous potential. The proposal seeks to insulate the advertising industry from information about browsing activity, while providing the means to use that information for targeted advertising. This idea, part of a general trend toward the use of privacy enhancing technology (PET), is one that is gaining recognition as a way of realizing the benefits of data use without the toxic effects of unconstrained data collection. As the advertising industry is responsible for some of the more objectionable data collection practices today, having a proposal that applies PET to advertising is very appealing.

Protected Audience falls short largely due to the magnitude of its ambitions. The Web is an extremely challenging deployment environment. Deploying this sort of technology to the Web means dealing with a bunch of very hard constraints. When combined with a need to produce a design that will support a profitable business for the advertising industry, the result is a pile of privacy compromises. Those compromises might be enough for some advertising businesses to turn a profit, but the resulting privacy properties are not good.

Protected Audience is an ambitious project on multiple levels. It is technically ambitious, relying on new mechanisms and technology, some of which has not been proven on the web. The number of components to the design and the complexity of their interactions is similarly ambitious. Where Protected Audience is most ambitious is in its aspirations of acceptance by both the advertising industry and Web users.

Unfortunately, the design of Protected Audience is riddled with concessions that are the result of a consistent desire to have the feature appeal to the advertising industry. The result of these concessions is a design that — at best — provides only very weak technical privacy protection.

The current state of Protected Audience includes a number of temporary measures, most of which negate the privacy protections in the proposal. These temporary measures are included so that the advertising industry is able to learn how to use Protected Audience. This is an understandable approach because Protected Audience is not only massively complicated, but innately very difficult to use due to the opacity of its functions. Allowing sites to remove opacity in key places aids them in using Protected Audience. However, this also negates the core privacy protections that the proposal aims to provide.

Even if Google is able to remove temporary measures on its planned schedule, the ideal state of Protected Audience still requires considerable work to achieve its privacy goals. There are several weaknesses in the design that are difficult to address. For some of these, we are not aware of any feasible solutions. Though it is possible that clever new techniques might be discovered eventually, deploying Protected Audience without even a general strategy for addressing these shortcomings means accepting the possibility of perpetual privacy issues.

In the near term, the design ends up providing people with no meaningful privacy protection. Over time, the removal of temporary affordances might improve the privacy situation, but the overall design contains weaknesses that have no clear solutions. Those mean that a promise of privacy could be a mirage: something that might seem close, but never ends up materializing.

The purpose to which this technology is applied — primarily retargeted advertising — is controversial. Even if the privacy shortcomings of Protected Audience were to be addressed, the question remains as to whether the application to advertising justifies this use of technology.

Whether an idealized version of Protected Audience might be justified is largely an academic exercise. In its current form, Protected Audience falls a long way short of its own privacy goals. While some of these flaws are likely temporary, some problems have no known solution. We cannot accept that improved ad targeting is worth that cost.

Table of Contents

Preliminaries

This is a high-level overview and analysis of Protected Audience. The goal of this analysis is to understand the implications that might arise from deployment of Protected Audience on the Web. Our primary focus is in determining whether it meets its stated objectives, which is to support a particular mode of targeted advertising on the Web. Of particular interest is testing the claim that Protected Audience will protect the private information of people who use the Web.

We acknowledge that a broader notion of privacy encompasses more than just giving people control over information about themselves. Protected Audience is specifically designed to enable cross-site remarketing. That people will observe consequences of their actions on other sites undermines the idea that each site is a unique space in which they can present a different persona. A more thorough discussion about the risks and benefits of enabling the use of cross-site or cross-context information for the purposes of advertising is a worthy subject, but this document is primarily technical.

Baseline

We start by accepting the basic privacy model that Protected Audience assumes, as outlined in "A Potential Privacy Model for the Web". In short, this model states that the Web achieves an important privacy goal if details of the interactions between a person and a website cannot be learned by any other website. This goal is achieved only when sites cannot collude to obtain this information, though it allows people to release the information if they choose to.

This goal effectively prohibits cross-site information flow about a person. However, we accept that there are conditions under which information exchange is advantageous. Deliberate actions on the part of a person are one such case where one site might be granted information about actions on another website. However, the goal is to require that no website be able to obtain cross-site information without assistance.

Protected Audience is an example of where a user agent might allow limited information flow about cross-site activity. We will look at the extent of the information flows in

Protected Audience and seek to establish the extent to which the cross-site information that flows is controlled. If the intent is to achieve the stated privacy goals, the API needs to ensure that what a site learns either cannot be attributable to any specific person or that the information gain is quantified and bounded. Understanding privacy loss makes it possible to make judgments about whether the privacy loss is justified, though that is ultimately a subjective judgment.

We start from an assumption that the Web largely achieves the stated privacy goal. There are a number of places where this is not necessarily the case today. There are also some ways in which it might not ever be possible to attain this ideal state. We will discuss some of the relevant limitations as part of this analysis.

However, it is worth noting that this baseline differs significantly from the assumptions of the authors of Protected Audience. Google Chrome is, in effect, the only browser that currently permits the use of third party cookies. Cookies make it trivial for different sites to correlate the activity of the same person and violate these goals. This cannot be the baseline from which to compare this capability from a privacy standpoint because all other browsers block this use of third-party cookies and Google has well-documented plans to do the same.

Ultimately, the question we seek to answer is what effect Protected Audience might have on privacy relative to a browser with these basic privacy protections implemented. That is, it is most appropriate to consider the effect Protected Audience has on privacy when cross-site cookies are already blocked.

Temporary Exclusions

Protected Audience also includes a number of temporary measures. These measures are generally aimed at making it easier for sites to adopt Protected Audience, which often comes with a severe cost to privacy.

The bulk of this analysis considers the effect of Protected Audience without these temporary measures in place. This alters the analysis as far as the complexity of implementation, both for browser makers and for sites who might use Protected Audience.

We include a section on Temporary Measures that discusses the effect of each of these measures in a little more detail.

Threat Model and Actors

Our threat model has only a single honest entity: the user agent. That is, we assume that people choose a user agent that they trust to represent their interests. For the purposes of this analysis we do not consider the possibility that someone might be convinced — or forced — to use a user agent that does not adequately fulfill this role.

Protected Audience is very likely to be used by entities in the advertising industry, who seek only to use the capabilities to advertise. We are interested in what the privacy consequences are for those entities that participate fairly and with all proper respect for privacy, but this sort of usage has little bearing on our conclusions. We are interested in understanding the extent of the risks involved, which means considering what advantage a dishonest actor might be able to obtain.

One important property of the Web is that it is possible to participate without gaining permission. Open participation means admitting the possibility of bad actors. Therefore, we have to look at the privacy protections that might be offered to people when the sites they interact with are malicious. To that end, our model assumes that sites might be controlled by the same adversary or by fully cooperating adversaries. This also extends to any companies that might act on behalf of a site.

An adversary is assumed to be able to cause at least some people to visit their site and then interact with the content that they display. Malicious sites are assumed to be able to enlist the help of any number of other malicious sites that are willing to share any information.

The consequences of open participation also extends to participation as a user agent. An adversary is considered capable of obtaining and running a user agent. Similarly, we consider it to be easy for an adversary to create websites. Adversaries can therefore be assumed to control any number of user agents. Only where specific protections are in place do we allow honest actors to detect or block such a user agent.

A design might specifically designate certain elements as trusted. These require analysis to support any claim regarding trustworthiness. Trusted elements also invite the possibility of threats other than our primary privacy concerns, so we will also consider factors like security, centralization, and equitable access.

Technical Protection

This document concentrates on the *technical* privacy protections in Protected Audience. We acknowledge that non-technical safeguards have some role to play in complementing technical protections. There are many aspects of privacy that do not submit to purely technical controls. However, we also believe that technical protections need to provide some assurances to those who might not be able to rely on any non-technical protections.

Google does not make Protected Audience available unconditionally. Sites are required to make a public commitment not to use Protected Audience to identify users across sites before the API is enabled. Firstly, we do not believe that this will provide a strong enough guarantee against bad behavior. Secondly, this creates the unwelcome precedent that a browser maker can decide whether or not a website is able to use a certain capability, which is contrary to our guiding principle of enabling open participation.

We do not further consider this and other non-technical safeguards in our analysis.

Proposal Documentation

The Protected Audience documentation is under active development, so there are gaps and mismatches between the intent of the proponents, what is implemented, and what the various documents capture.

Protected Audience is a proposal that is described largely through an explainer, along with a number of supplementary documents. A draft specification exists, but is not a very useful reference for the purposes of understanding the goals and intent of different design features in the proposal. Some of the information in this document is also derived from open issue discussion and source code.

Some Details Elided

This document is an attempt to analyze the high-level structure of the proposal. To this end, a lot of detail is not included. A number of features in the proposal are outright ignored. For instance, component auctions are a major feature that we do not examine in detail. This is because, for the most part, component auctions do not substantially affect the privacy analysis.

As much as possible, any assumptions will be pointed out, but this is a complex proposal that comprises many interacting components. Errors of interpretation are likely given the size, scope, and nature of the proposal.

Protected Audience Overview

Protected Audience is complicated. However, many of the complications in the design have no material impact on privacy analysis. Therefore, it is possible to describe the system in an abstracted form in order to understand it better.

Protected Audience allows sites to create a marking for visitors. These markings — called interest groups in the proposal — are created by any site and can be used on any other site, for limited purposes.

How markings are applied is not constrained. Sites can apply many markings with no privacy-relevant constraints on the information that is attached to each marking. Markings can also have any amount of additional added to them by sites. Any privacy provided comes from very tightly controls on the use of markings.

The goal of Protected Audience is to ensure that markings cannot escape the browser. Markings can only be used to select and display an advertisement. Chosen advertisements are displayed by the browser, but the actors involved in supplying that advertisement are supposed to remain completely ignorant of what was displayed. Sites are not supposed to be able to learn whether markings exist, what markings exist, who applied the markings, and which markings were used in the ad selection process.

In effect, Protected Audience follows a label-based information flow control design. Access to markings is conditional on not being able to write to any context with a less restrictive label; that is, anything that can read markings cannot be allowed to communicate with ordinary web content.

Markings can be applied by any site that a person visits. As a person visits different sites, those sites might observe activity that an advertiser considers to be of some interest. For each such event, the site requests that the browser store a marking.

The site includes arbitrary information with each marking that can capture details of the activity of interest or any information that might be useful to the site when it later uses the marking, as below. Markings are owned by a single site that is nominated when the

marking is created. The owning site can be different to the one where the marking is created.

At a high level, the process for reading markings operates as follows:

1. Any website can create a processing context that has both elevated access to markings and reduced access to communication. The site decides which sites will be able to execute a program in this context. In creating the context, the site supplies some basic information that is made available to all of the programs.
2. Programs from the identified sites are executed. Only sites that own applied markings have their program executed. Each program is given access to their own markings and the contextual information supplied at the time the marking was created. The output of each program is a URL, a rating for that URL, plus arbitrary supplementary information.
3. The initiating site is then able to execute its own program. Separate instances of this program receive the output from each program along with the same contextual information. Each instance of this program is not permitted to communicate with other instances or the site. The output of each execution of this program is an adjusted rating.
4. The user agent selects the URL that received the highest adjusted rating. The value of any rating and any supplementary information are discarded.
5. The selected URL is displayed in such a way that prevents any of the involved sites from knowing what URL was chosen.

Programs are executed independently. The same program might be instantiated multiple times, once for each marking. Those instances are not permitted to communicate with each other. Programs are ephemeral and have no means of remembering the information they receive; they act as pure functions that translate inputs into outputs.

In practice the site is given an opaque handle to the selected URL. The site can give the browser this handle and ask the browser to display it. The browser is able to use the handle to obtain the selected URL and display the referenced content. When the site asks the browser to display the referenced content, the browser does so in a way that ensures that the site cannot learn which URL was selected.

Privacy Analysis Overview

The description above is idealized. There are several ways in which ideal functionality is not achieved. The privacy analysis in this document explores some of those gaps.

If the user agent has to fetch a URL to display the content, that reveals that this URL was chosen to the server that the URL refers to. Programs could encode the information they received into the URL and effectively circumvent any restrictions on communication. Use of entirely self-contained data: or blob: URLs is one possible approach to eliminating this concern, as those URLs do not require communication with a server. However, Protected Audience does not presently limit use of URLs in this way. Fetching Ad Content discusses how Protected Audience deals with this problem in more detail.

People interacting with advertisements also creates further opportunities for the information that contributed to their selection to be leaked. We also consider information leaks that occur as a result of Interacting with Ads.

This design has significant challenges for browsers in terms of providing Isolation for Auction Processing. Additionally, in order to make Protected Audience practical for deployment – both in terms of usefulness and in terms of complexity – there are a number of places where information leaks. The safeguards that are placed around each of these leaks each require its own analysis. Of particular note here is the potential for Real-time Updates from servers and the system of Reporting which advertisements were selected.

The use of *k*-anonymity fills a peculiar role in the design, acting as something of a backstop for privacy leaks in other parts of the design.

Advertising with Protected Audience

Protected Audience is designed for the very specific use case of selecting advertisements. In theory, a narrow application domain like this is more conducive to a tightly constrained design. That is, the design could provide sufficient flexibility to address identified, but avoid the true generality that might result in unexpected or unwanted uses.

The structure of the processing in Protected Audience matches that of an ad auction, where inventory – or spaces where ads might be displayed – is made available to the highest bidder.

1. People visit sites and do things on those sites.
2. Entities present on those sites observe what people do and request that the browser save markings, to which they attach the details of the people or what they did.
3. A site that decides to auction a space for use by an advertiser. The site — or their delegated Supply Side Platform (SSP) — initiates an auction using Protected Audience.
4. The auction runner chooses what information they want to pass to potential buyers.
5. Each buyer — or their delegated Demand Side Platform (DSP) — supplies a program that can make a bid.
6. The bidding program is given the information provided by the auction runner and the information recorded in a single marking that is owned by the buyer. The same bidding program is run once for each marking, so multiple bids can be made.
7. The auction runner screens each bid that was made to apply brand safety or other checks, which can result in adjustments to bids.
8. The browser selects the ad that has the highest bid, after any adjustments made by the runner.
9. The auction runner asks the browser to display the selected bid.

The simple system described here has no outputs other than content that is displayed on screen. A system that only performed this function might be difficult to build, but under the right constraints this goal is technically feasible.

The complete process is considerably more complex than the one described here. Later sections of the document will introduce variations to this process as they become relevant.

Privacy Analysis

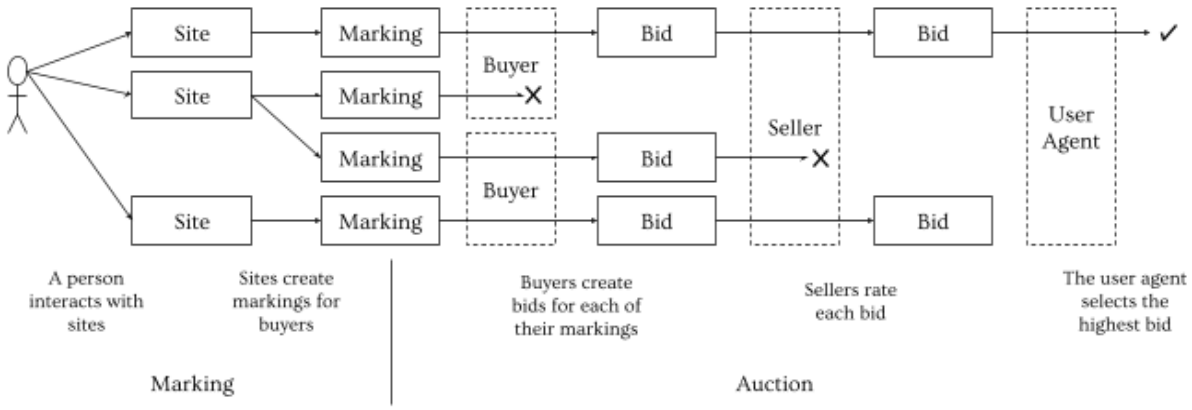
Markings applied by sites are expected to carry information that a DSP can use to refine bids. As such, markings are expected to contain attributes that are relevant for advertising purposes, such as whether the person exhibited interest in a given product or topic. As such, the API accepts structured data with field labels like “ads” or “trustedBiddingSignalsKeys”.

There is no real limit placed on the information that can be attached to a marking. Similarly, the auction runner can include arbitrary information that is passed to all of the programs that run. While many of the fields in the proposal are processed in very specific ways that might constrain what they can carry, there is effectively no limit on the quantity of information that can be passed between sites at this stage.

The primary constraint on the availability of information is that each marking is processed separately.

- Markings can be applied or updated by any site, but each marking contains information from a single site.
- Bidding programs can only access one marking at a time and make a single bid.
- Screening programs can only access one bid at a time.

This means that the information about browsing activity that is passed to each bidding program comes from exactly two sites: the site where the marking was applied and the site where the auction is running.



The screening program run by the auction runner effectively has the same constraint. This program is run separately for each bid made by the bidding program. Though a screening program cannot access markings directly, a bidding program can copy markings to the supplementary data it provides in support of its bid.

Our threat model admits the possibility of collusion between the provider of the bidding and screening programs, so we assume that screening programs have full access to markings.

The information that is available to bidding and screening programs is sufficient to join the identity of a person across the two involved sites. That is, the current site and the one on which the marking was applied. For privacy reasons, this is an outcome we wish to prevent, there are two areas to focus on:

1. The process of running bidding and screening programs. Any information leakage from this process could be used to recover the cross-site information available to these programs.
2. The handling of the output of any bidding process. Cross-site information could be encoded in the output URL, so its handling needs to ensure that this information does not leak.

Achieving the first depends on isolation in processing markings (making bids and screening them); the second on isolation of all of the actions that follow, including fetching ad content, displaying it, reporting, and ad interaction. We'll look at each in some detail.

Isolation for Auction Processing

Protected Audience offers two modes of operation for executing the bidding and screening programs. The first executes these programs in the browser, the second is executed by a trusted server.

If breaking the isolation protections were the goal of a malicious site, then the weaker of these two options is relevant, as sites are able to choose which method is used for each auction. We will examine each in turn as each offers different opportunities for an attacker to exploit information leaks.

In-Browser Auction

A system that isolates processing in this manner is already fairly challenging to build. A browser executes code of hostile origin (websites) both with and without the elevated privileges necessary to access markings.

Browsers largely depend on the cooperation of sites to maintain isolation. That is, isolation between different contexts is maintained with the aid of the entity that is being isolated. Isolation primarily exists to ensure that secrets shared between a user and one site cannot be accessed by another site. For instance, the password used to login with a bank site is protected from being read by other sites.

Sites cooperate with this practice because it is in their interest to ensure that their secrets are adequately protected from leaking to other sites. To strengthen isolation, browsers implement a range of technical measures that either work without site involvement (like process isolation or CORS) or that add protections above the baseline when actively enabled by a site (like CSP, COOP, or COEP).

> **Note:** There is an important exception to this notion of cooperative isolation. That is, browsers aim to prevent sites from correlating activity from the same person across different sites. That is highly relevant to Protected Audience, as this is its primary goal.
>
> Being able to correlate activity across sites would violate contextual integrity, a powerful privacy concept that browsers aim to uphold for interactions with sites. When it comes to this goal, sites are viewed as adversarial to the goals of the browser.
>
> Modern browsers include a lot of functionality that can be abused by sites for cross-site recognition. Fingerprinting can provide a means of reidentifying the same browser on different sites. Navigation and bounce tracking can be used to pass information between sites and enable reidentification. However, most relevant to discussion of Protected Audience are imperfections in the browser sandbox.
>
> Sandbox imperfections mean that there can be ways to establish cross-site communication between sites using side channels, or leaks in the isolation that the browser enforces. To create a side channel requires that sites have a means of generating a signal and a complementary means of observing that signal. For instance, pool party describes how global limits on access to shared resources can be used to establish cross-site communication channels by consuming resources in one context and observing the effect that has on reduced capacity in another context.
>
> Recognizing that addressing these shortcomings in the platform is inherently difficult does not also justify the addition of new means of breaking site isolation. The Web community has broad agreement that new capabilities cannot worsen this situation. Tacit in this agreement is an acknowledgment that solutions to existing problems are difficult to build and deploy, but that doing that work is worthwhile. To ensure that efforts to improve isolation do not become futile in

addition to being difficult, adding new means of breaking isolation is strongly discouraged.

Protected Audience relies on having strong, technical isolation in order to maintain privacy.

Code from the same site is concurrently executed in two contexts with very different privilege levels. Code that has access to markings has access to information from two different sites: the one where the marking was made and the one where the auction is running. Though this code is unable to send messages directly, preventing data exfiltration in this setting is extremely challenging.

Bidding and screening programs execute in a shared computing environment. Even without any direct means of communication, side channels that exploit shared resources might be used to exfiltrate information.

Process isolation is generally seen as being effective at preventing some types of accidental information leakage; however, process isolation does not necessarily prevent leaks in an adversarial setting. Furthermore, recent research has shown that gross differences in execution patterns can be observed across processes.

The shared resources in a browser include CPU, memory, and cache. Attacks on isolation mechanisms in modern hardware — such as those that exploit weaknesses in implementation related to speculative execution — show that it is possible to access secrets from programs, even when an effort has been made to prevent those secrets from leaking. Protected Audience executes programs that have an incentive to leak their secrets. Then it provides those programs with many options for carrying out those leaks.

For instance, bidding and screening programs are denied access to a real-time clock, but the code that runs outside of the restricted sandbox can access high resolution timers. This site could observe the time that an auction takes to execute. If programs deliberately take varying amounts of time, they could use their running time to leak information.

A variable number of bidding and screening programs execute. The auction runner chooses a maximum number of bids, however the maximum number of programs that are executed is based on how many markings were made by sites. If the number of programs that are executed could leak, this could reveal how many markings were selected.

The auction runner is also able to selectively alter which markings are considered:

- The auction runner specifies which sites can be involved.
- The auction runner can select which markings are considered using a special query language that can interrogate arbitrary properties of the marking that are set when creating or updating the marking.

These factors make preventing information leakage through side channels quite difficult. A typical approach to preventing leaks through the execution time of isolated code is to ensure that it takes a fixed time to run (or, more precisely, that the running time does not depend on information that should not be leaked). Performance and flexibility requirements of advertisers might be incompatible with that sort of defense.

Bidding and Auction Server

Google also describes a variant of the API where bidding and screening programs are executed in a secure enclave provided by a trusted cloud computing service. This approach can provide ad tech companies with stronger assurances about the confidentiality of their bidding logic.

This approach involves the use of a trusted server. Browsers encrypt markings, buyers encrypt bidding logic, and sellers encrypt both information and screening logic. Only the server can decrypt each.

The main reason to use a server for the auction is that buyers and sellers can avoid having to send proprietary bidding logic to user agents that they cannot trust. Auctions are a competitive context in which knowledge of a competitor's strategy can provide an advantage. For this reason, ad tech companies have a strong interest in keeping their bidding logic secret. On-device auctions would mean that a competitor could easily obtain logic by running some user agents.

With a trusted server, people (or their user agent) need to trust that the server will not reveal browsing information. Additionally, buyers and sellers who participate in auctions need to trust the server to protect their information.

Running in a TEE provides an opportunity to limit the information that might leak to the site that runs an auction. Though malicious bidding logic might attempt to generate signals through side channels, some safeguards in the TEE environment are enough to ensure that signals cannot be accessed, especially by the site that initiates the auction.

The primary means of leakage is then due to failures in the TEE and in the running time. Setting a fixed running time is necessary to ensure that the execution time of individual programs cannot be used to leak information. This is more necessary in this case because the TEE host runs software provided by the auction runner, who might have an incentive to access both the browsing data provided by the user agent and the bidding logic provided by their competitors.

Though execution of auctions in a TEE might provide ad tech companies with more assurances about the secrecy of their commercial data, it might only change how browsing data leaks. That is, the execution might be made safe, but the process of getting the data to the server might leak.

Sending private information to the TEE creates new potential side channels that can leak information. The size of the encrypted bundle that the browser sends to the TEE will reveal some information about the markings that are included in that bundle. The proposal currently suggests that padding might be used to limit information leakage. However, given the adversarial nature of the API, and the number of ways in which the size and number of markings can be controlled by adversaries, this is not sufficient to protect privacy. Any scheme that allows cross-site information to leak via this side channel effectively creates a means of querying that information.

A TEE might seem to allow for more compute resources to be deployed for executing bidding logic. Even where bidding logic is relatively simple, the very large performance advantage that a TEE might have over the client does not guarantee that overall performance improves. User agents will still incur costs related to encrypting markings and submitting the necessary information to the TEE. These costs are likely to be significantly higher than executing bidding logic locally, both in terms of computation and latency. Only in cases where bidding programs are extremely complex would there be a net gain in efficiency.

Potential Isolation Improvements

It might not be necessary to build an isolated processing system for auctions. Modern machine learning-based bidding logic can — at least in many cases — be reduced to a simpler inner or dot product. Features from both the context and the interest group could be expressed as a set of vectors that could be merged into a single vector. Ads would similarly be expressed as a vector of the same dimension, updated in real time as necessary. Bids would be the inner product of these two vectors.

The same process could be used by the auction runner to compute a scaling factor for the bidding value when screening bids.

Computing a dot product in constant time is far less likely to create side channels than other options. This does not directly address any leak that comes from encoding varying numbers of markings for a trusted server, which would need additional safeguards in either case.

This is not a novel approach in the API. The scheme that is used to prioritize and select markings for inclusion in auctions uses the same inner product approach.

The advantage of this sort of approach is that an inner product is very easily computed in constant time, eliminating the risk of information leakage.

The disadvantage is that it imposes a usability constraint on buyers and sellers. Writing JavaScript code for bidding and screening logic is complex, but expressing that logic in the form of a vector might not be compatible with existing practices.

The reasons for using a bidding and auction server are not eliminated by this approach. The vector assigned to each advertisement still represents the same sort of proprietary value as a program. Protecting that information from competitors is still likely required.

No Winner Leakage

An auction that does not produce a winner results in the site that initiated an auction learning that the auction failed. This can be used to leak one bit of information per invocation. At a minimum, this can be used to probe for the presence of markings. Once the presence of a marking has been established, additional auctions can produce one bit of information from those markings per invocation. Revealing auction outcomes in this way can also be used to reveal when people disable the Protected Audience feature.

The ultimate goal of Protected Audience appears to be that this leakage is not present. Instead, a seller would provide a set of ads that will be used in case there are no markings or the auction fails to produce a winner for any reason. The result would be that the auction would always select a URL, removing the information leak.

The reason that sites are allowed to learn when there is no result is so that those sites can avoid having to use Protected Audience for any contextually-targeted ads. If a site wants to use interest groups (that is, markings) for some ad targeting, having an auction

always succeed would mean that any ads that use purely contextual information would need to be included in the same auction under the same privacy rules. That means that contextual ad placements would be subject to the same constraints as those based on cross-site information.

Revealing that an auction failed to produce a winning URL allows sellers to conduct two separate auctions: one that uses interest group information for targeting and one that relies only on contextual information. The motivating idea is that revealing this information will boost uptake of the proposal. Sites can run an auction using interest groups, but fall back to using purely contextual ads if there is no winner. Any purely contextual parts do not have to deal with all of the more complex aspects of Protected Audience for those ads, such as the real-time updates and reporting system.

The cost of allowing this leakage is that revealing that an auction has failed creates a cross-site information flow. Though an auction failing to produce a winner does not obviously reveal anything, it can be used to reveal one fresh bit of information upon each invocation of the API.

The seller that initiates an auction can use systems designed to refine the set of buyers as a form of query language to select which markings are considered in an auction. The seller can then pass information to the buyer scripts that can alter how the buyer releases information through their bids. In effect, the meaning of success or failure of a given auction can be changed dynamically, so that — with each auction — the seller learns new information about what occurred on other sites.

Though this pattern of queries might seem to be suspicious, it only requires a small number of such “queries” to link cross-site identities. For a site with a million visitors, as few as 20 queries would be sufficient. Such queries could be intermixed with real uses of the API so that it would be difficult for a browser to distinguish those abusive uses from genuine ones. Indeed, a careful attacker could make such “queries” double as entirely valid ad placements, either based on interest groups when the auctions succeed or contextual information when auctions fail to produce a winner.

Negative Targeting

The privacy consequences of leaking the status of failed auctions is known and acknowledged. A negative targeting feature has been developed that is intended to make it easier to integrate contextual bids into the auction. In effect, negative targeting enters

contextual ads into the same auction, but provides ways for bids on those ads to be negated in the presence of certain markings.

Unlike other explicitly temporary measures in the design, there is no planned timeline for addressing this shortcoming.

A requirement to use negative targeting is expected to have a serious detrimental effect on uptake of the API as sites would not be able to conditionally deploy Protected Audience, where ordinarily contextual targeting is used if it fails to produce a winner. Instead, the contextual targeting would need to be integrated into the auction, which means dealing with all of the complexity of selection and (critically) reporting without the visibility that might otherwise be available.

Observability of Opt-Out

Having the option to refuse participation is necessary for this sort of functionality. Having that opt-out be undetectable to sites is a valuable tool for protecting people who choose not to participate. Detecting opted-out visitors might allow sites to discriminate on that basis.

People who choose not to participate in Protected Audience can do so by refusing to accept markings. This avoids the privacy loss by preventing any flow information between sites. These people could still pretend to support all the functions. However, any auction would be guaranteed to fail, which could be used to detect when someone opts out.

Preventing the outcome of an auction from leaking would make it harder to detect who has opted out. Only leakage from the auction would reveal this fact. Effective protection against discrimination means ensuring that someone who opts out (who will always have no markings) cannot be effectively distinguished from someone who has markings. This is the same leakage protection that is already needed to protect those people who do participate.

Fetching Bidding Logic

Each buyer in the auction will have zero or more markings registered prior to the auction. For any buyers that have one or more markings, each marking will be processed using different bidding logic: a small program. That bidding program needs to be

retrieved. This occurs while the user is interacting with another site, not directly associated with activity on the site that applies the marking.

Some information leaks from the browser when a site initiates an auction. If the timing of auctions is known or controlled, the seller reveals cross-site information to each buyer. The set of people for whom auctions are initiated, which might be as small as a single person, can be identified at the site where the ad would be placed. This person could be identified as being one of a finite number of people at the site where the marking was applied. As multiple auctions are conducted, people could be reidentified through the fetching of programs.

Protected Audience aims to prevent this sort of linkability for the advertisement that is output from the auction by applying joint *k*-anonymity limits on the combination of: the interest group owner, the bidding URL, and the URL of each advertisement. No such limit applies for fetching programs.

Making this fetch *k*-anonymous through the use of something like Oblivious HTTP is necessary but not sufficient to address this concern. Limitations on the general effectiveness of *k*-anonymity protections are discussed below. Furthermore, even if *k*-anonymity is completely effective, the following attack works.

Linkability Attack on Bidding Logic Fetch

Fetching bidding logic leaks information that can be exploited to bypass the privacy protections in the API. The following attack assumes that bidding logic is only fetched if the URL for that logic is jointly *k*-anonymous with the bidder identity, though this is not a requirement in the current design.

To understand the attack, start with the case where the site that initiates the auction wishes to learn the identity of a single person on another site. To prepare, the site requests that another site apply markings (i.e., assignments to interest groups) across their entire user population in a specific pattern.

Each visitor is assigned a unique pattern of markings, though each is given as many as would be needed to identify them from the defined anonymity set. If a *k*-anonymity threshold is applied, then one marking needs to apply to multiple people. For instance, if the threshold is 50, then 6 markings are enough to both reach the threshold and still uniquely identify each person.

Markings need to be unique to the auction site for this attack to be effective. For a marking-side identity to be linked to multiple auction-side identities, fresh markings are needed for each auction site.

For the single person of interest on the site of the auction, an auction is then executed. The auction causes bidding logic to be fetched for their set of markings in a pattern that uniquely identifies the person. The timing of the auction is used to provide an indication about which person is being identified on the site that executes the auction.

An isolated auction can link multiple unique marking-side identities to the identity on the auction side. That is, a single auction can be used to perform this attack on one person across multiple sites.

It is marginally more challenging to scale the attack to re-identify multiple people at the site of the auction. Each auction can be bounded in time, with a distinct start and end, which means that fetches can be linked to a specific auction. However, the need to do a single auction at a time — combined with the potential for failures — means that the rate at which identities can be linked is limited.

If concurrent auctions are conducted on the auction site, this creates uncertainty about which of the auction-side identities correspond to each of the marking-side identities. However, this can greatly increase the rate of linking. For instance, running two auctions at the same time puts a site visitor into a set of two potential marking-side identities. Follow-up auctions that pair each of the two visitors differently would allow the ambiguity to be resolved. Pairwise iteration saves nothing, but larger groupings

In this way, a million people could be re-identified with a million sequential, single-person auctions. For a million people, twenty thousand auctions over a thousand people each is considerably more efficient. Larger individual auctions reduce the overall number of auctions, but the availability of visitors for auction participation becomes a limitation in that case.

Attacking privacy in this way would not prevent a site from also using Protected Audience in the intended fashion, making it easier to hide this pattern of abuse.

This attack can be avoided by having fetches for bidding logic be driven only by the sites that apply markings. This means only fetching and caching bidding logic when markings are applied or reapplied. It also means avoiding fetches during auctions, relying on cached logic exclusively.

That protection would effectively remove the most direct means of updating markings. Markings could not be updated after an auction completes as documentation presently suggests. This is because the timing of that fetch is under the control of the auction site, which would allow for a similar cross-site signal. That attack might have an uncertain bound on the end time for the fetch, but the effect is only an increase in uncertainty, so it would not be an effective defense.

Real-time Updates

Interest group registrations (markings) can last for up to 30 days in the browser. During this time, the advertisements that are included in each interest group might run through their allocation of money, be stopped by the advertiser, or otherwise cease to be relevant.

Interest groups are updated after each auction that they participate in. However, updates that occur after an auction are not sufficient to prevent unwanted advertisements from running. Similarly, setting an expiration date on an interest group does not ensure that an advertiser is able to properly respond to changes in circumstances.

Protected Audience provides buyers with the ability to query a server for updated information about campaigns at the time of an auction. A base URL is indicated as part of each marking. Requests are coalesced in the case that multiple markings have the same base URL.

The URL is expanded to include the origin of the site where the auction is running, the interest groups that are active, a set of arbitrary keys provided with each interest group, and a single key specified by the seller.

The server returns arbitrary information associated with each of the keys, plus new information that can be used to alter the priority of each interest group. A subset of this response is passed to the bidding script for the associated interest groups.

For the auction runner, a similar service exists. A base URL is expanded to include the URLs of the ads for which bids were placed by buyers. The per-bid information is returned to the screening script.

Obviously, the servers that produce these requests are given access to information that is sensitive and private. The Protected Audience design depends on being able to trust servers with this information.

Like the resource used to retrieve bidding logic, making a request to and response from the key-value server leaks information through the size of messages. Padding is applied, but with 16 different sizes, each request could leak four bits and each response leak another four bits. This leak is observable from the network and from the server that hosts the TEE, which is operated by an adversary.

Requests to the same server are coalesced. This would reduce the rate at which data can be obtained, except that there are no limits on how many different servers can be contacted as part of an auction. The only advantage apparent from coalescing is a performance gain, not a privacy one: that is, it only allows the overheads incurred from each request to be amortized over multiple requests from the same client.

There are temporary measures that loosen the protections in place for real-time updates.

Updating Interest Groups

Each marking comes with a URL for a resource that can provide updated information in the event that the site that applied the marking is not visited again. This resource is polled any time that an auction is run, with a rate limit so that it is polled at most once per day.

Like the resource used to retrieve bidding logic, updates leak information. There is no *k*-anonymity check on the update URL, so each URL can be unique to a specific visitor. If each update is the result of an auction, then that creates cross-site information flow.

Having some uncertainty about timing of update checks — an unspecified time after the completion of an auction — along with the rate limits, makes updates a somewhat less reliable means of performing cross-site re-identification attacks.

For instance, the logic for selecting markings that are available to a seller might not apply to updating interest groups. That is, it is possible that the browser would update all interest groups for all of the buyers that are mentioned in an auction, even for interest groups that were not part of the auction.

These limitations could be circumvented by specifying unique buyer domains for interest groups. Each visitor to a site can be given a unique interest group with an update URL that uses a similarly unique registrable domain. Browsers would then have no cause to initiate an update unless that interest group was mentioned in an auction.

The main drawback — to an attacker — that results from using unique origins would be that the number of origins listed in the auction configuration would likely need to be very large. In order to make use of updates for a cross-site re-identification attack, the list would need to contain all possible origins from the candidate set, which in early iterations would include every site visitor. This might still be used to supplement attacks that rely on fetching of bidding logic, particularly in later iterations, once the set of candidate identities involved in an auction reduces after multiple iterations.

Fetching Ad Content

Advertising content for a winning bid needs to be rendered by the browser. If fetched by the browser, this will leak the ad selection — and its timing — to the server that hosts the content.

Interest group registrations that are made by websites include references to advertising content. Bids can only select from the limited set of ads in the interest group, with some limited templating to allow each piece of content to be adapted to the page dimensions or similar things. This presents the possibility that ad content might be fetched at the time an interest group is either registered or updated, so that the fetching of ad content does not leak information at the time an ad is displayed.

The original design had content provided in the form of web bundles, which are effectively self-contained web pages that can be loaded offline. However, this technology failed to achieve enough interest in its benefits to outweigh some substantial complexity and usability shortcomings.

Whether in the form of web bundles or simply cached data, storing ad content without a live fetch introduces other issues. Stored content means that advertisers are unable to change ad content dynamically. Content might be refreshed when the browser decides to update an interest group, but this is not guaranteed to occur before the ad is rendered. Errors in advertising content that need to be corrected, particularly those that could have an effect on the reputation of the advertiser, would not be reliably corrected if content were saved in browsers.

Additionally, the potential for there to be a large number of registered interest groups, each with multiple linked ads means that the size requirements for storing ad content is likely infeasible. The Chrome implementation allows up to 1000 interest groups for each of up to 1000 buyers. If each interest group has 20 ads — allowing for both

different ads and multiple sizes of the same ad — and each ad is a modest 5kB in size on average, that would require 100GB of space. Even with fewer interest groups or buyers, it seems unlikely that heavier ad content could be stored and kept updated without costing significantly more than this.

The present specification deals with these limitations by allowing ad content to be fetched at the time that an ad is rendered. Though the URL that is rendered is subject to *k*-anonymity constraints, fetches reveal information to the server that provides the ad content.

Even with use of some form of anonymous fetching mode (such as using Oblivious HTTP or proxying options) — something that Protected Audience does not require — timing information might be used to weaken or eliminate *k*-anonymity protections.

The present design does not appear to include any sort of anonymous fetching, making it trivial to use IP addresses to correlate fetches with site visits. However, a browser could conceal IP addresses through the use of an anonymizing proxy at any time with no effect on other parts of the system, except perhaps some degradation in responsiveness metrics.

Interacting with Ads

Advertisements that are shown might be interacted with. After all, many advertising campaigns seek to have people interact with the advertisement in order to achieve the campaign goal.

Prior to interaction, the browser needs to prevent isolated ads from exfiltrating the confidential information they hold. Once someone clicks on or otherwise interacts with an advertisement, if the intent is to follow a link, the browser needs to allow the link to be followed. Anything else would be surprising to all parties involved and it would remove much of the value that the advertising provides to advertisers. Even if there are many uses for advertising that does not depend on interaction, there is a strong expectation that interaction does something.

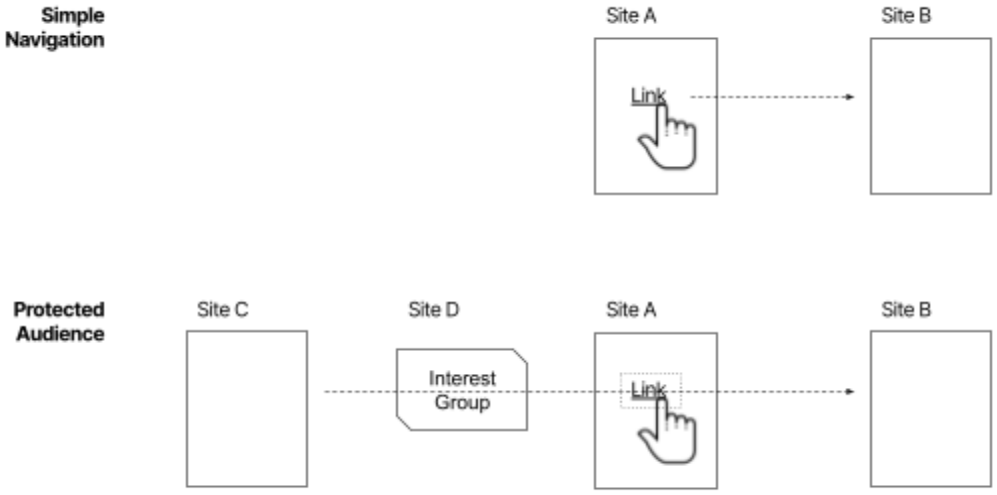
For Protected Audience, this creates a problem. Following a link causes all of the information that is available to the isolated advertisement to be released. Typical actions, like navigating the top-level context (that is, a simple page navigation) or opening a new window leaks some of the information that was used in selecting the ad.

For example, an interest group could contain two ads: https://example.com/ad1 and https://example.com/ad2. The content of each could be identical, except in the choice of URL. The bidding logic could use the choice of ad to carry one bit of information to the destination site. Having more ads in the interest group would allow more information to be revealed. Each additional ad placement could reveal more information.

This information release undermines the goals of the system. The Protected Audience design implies that this release is intentional and therefore acceptable. That is, when someone interacts with an advertisement, they implicitly consent to the information being shared. The browser has to release the information in order to comply with the expressed intention of the person who chooses to interact with the ad. Furthermore, information that is released is *k*-anonymous.

The shortcomings of this justification are several. *k*-Anonymity protections provide an imperfect or ineffective defense. The major flaw in the design is that the information released through any interaction is much bigger than is immediately obvious. When someone clicks on a link on a web page, the information carried in that link is usually limited to things that that site knows. Here, the information content of the link is limited, but it includes cross-site information that a site would not ordinarily be able to access.

That is, the information that the site at the destination of the link (Site B in the diagram below) ordinarily comes just from the site that provided the link (Site A). With Protected Audience, information also comes from the site that applied the marking (Site C), via the owner of the marking (Site D) and the site that shows the link (Site A). This new source of information is not involved in any obvious way.



Considering abuse scenarios further undermine the argument, because the intent behind a click cannot be assumed to signal an intent to interact with an advertiser.

Protected Audience Does Not Necessarily Produce Ads

Advertising content does not need to be presented as an advertisement in any obvious way. There is no obligatory distinction between advertising content and other less privileged content, visual or otherwise. A malicious "advertiser" might then present content in a way that maximizes the chance that someone might interact with it without recognizing that this interaction will trigger the release of their browsing history to unknown entities.

Such content could be presented as interface elements, like buttons, that form part of what might be perceived as the interface of a website. For instance, isolated "advertisements" could be used in place of buttons that navigate to the next page of a multi-page article.

To understand the privacy risk associated with this, the person interacting with the "ad" needs to understand that the "ad" was produced by Protected Audience and where the information was drawn from. That is, that Protected Audience integrates information from other sites in selecting what content is shown and that the information that each ad holds is released to the site that is navigated to.

A *k*-anonymous ad might appear to be limited. The amount of information that can be passed from a single marking is limited by the number of advertisements it holds. There is no documented limit on the number of advertisements, but this is expected to be relatively small. However, the choice of marking carries information that is effectively unbounded, because there is no limit on the number of markings that can exist. Consequently, it is the choice of interest group that carries information, not the choice of ad within that interest group.

Attack on Ad Interaction

To give a concrete example, a malicious site might wish to learn about events that occur on other sites. This is information that Protected Audience specifically tries not to make available to sites.

With the cooperation of those sites, it creates markings for each of those events. For simplicity, we consider a single advertisement for marking. Each advertisement has a

unique URL on the malicious site. Each advertisement is identified differently and has a different destination page when clicked.

> **Note**: It is possible that advertisements will be (or are) prevented from navigating the current page in Protected Audience, which would make this attack less appealing, but this does not significantly alter the attack. A number of variations on the same basic idea are possible.

When a person visits the malicious site, the site runs an auction. The site can provide information to any bidding logic that instructs it to bid according to a predetermined priority. This ensures that the highest priority interest group that is present will win the auction. The winning "advertisement" is then shown in a prominent location on the page that increases the likelihood of someone interacting with it.

If someone interacts with the ad, they will navigate to a unique page on the site that runs the auction. This page can then tell the site that this visitor is part of the chosen interest group. The page can use cookies or any other persistent state mechanism to link the new information with a visitor profile.

Including additional advertisements in the interest group can increase the amount of information that can be conveyed, up to the limit on the number of advertisements per marking. This has only marginal additional value over the single advertisement as the choice of interest groups is what carries the most information.

This is a very effective transfer of information about activity on one site with another. The concrete amount of information that is transferred is nearly unbounded, because there is no limit on the number of interest groups that can exist. The only constraints that apply are the limits on the number of markings that can be stored by the browser, how easily those markings can be selected when executing an auction, and *k*-anonymity protections.

The primary drawback of this approach is that it depends on having someone interact with specific parts of a website. Recall however that our threat model assumes that a site is able to induce at least some people to click on links and buttons. A site can saturate their interface with buttons that are backed with these "advertisements" rather than real links, each passing information from other sites to the link destination. A site could put these false advertisements in place of important controls, like page navigation controls or those that dismiss cookie banners.

k-Anonymity Provides No Meaningful Protection

Limits on use of advertisements that do not reach *k*-anonymity thresholds might reduce the ability to have very granular markings, but this should not be an impediment for information that is shared between multiple people. Indeed, if the goal of the adversary is cross-site recognition – that is, connecting pseudonymous identities on different sites with the same person – a relatively small set of markings could be used and shared between multiple people.

For instance, if the goal is to link the identity of 1 million people across two sites, then each person could receive two different markings from a set of 10,000. As long as no two people receive the same pair of markings, two ad interactions is sufficient to perform cross site recognition.

In the prior example, each marking is applied 200 times on average. With two advertisements per person and a *k*-anonymity threshold of 50 (and no attempt to circumvent that limit), if a million people interact with the false advertisements. around half of the population could be recognized successfully.

This demonstrates that it is relatively easy to change parameters to maximize the odds of successful reidentification. Reducing the set of different advertisements to be 1,000 increases the number of people with any given marking by a factor of 10. The recognition rate then increases to be closer to 95%, without increasing the number of interactions required.

Additional interactions can eliminate the risk that people evade recognition due to *k*-anonymity thresholds. People who are successfully recognized could become part of a pool that can contribute to meeting thresholds for a given advertisement. This can be controlled so that it is easier to target any unidentified people.

This approach works even if the markings are applied across different websites. This means that a larger site can learn about how their visitors interact with multiple other websites. The only restriction is that each visitor will only release information about one marking at a time, so learning about activity on multiple sites requires additional rounds of interaction.

Reporting

The eventual goal for Protected Audience is to provide aggregated and differentially private reporting for each winning bid. The Private Aggregation API proposal provides programmatic access to a histogram aggregation function that is aggregated by a service that is trusted in several ways.

User agents trust the service to protect their contributions by not revealing individual inputs and by adding enough noise to aggregates that they can assume differential privacy protection. This includes maintaining a privacy budget for each site.

Websites trust the service to faithfully add up contributions and to not add more noise than is necessary to meet differential privacy requirements.

The basic approach of using an aggregation service is sound, but there are significant tradeoffs that need to be made between privacy and utility.

There are many considerations when implementing secure aggregated reporting. In particular, the choice of privacy parameters, such as the ε and δ in (ε, δ)-differential privacy, has a significant effect on a privacy-utility balance. Our current experience with aggregation for attribution suggests that this sort of system is broadly feasible, though we do not have deployment experience with this particular form.

Though the system is broadly workable, the parameters that are used will ultimately determine the degree to which it maintains privacy. We were unable to determine the values that Google uses in its current deployment. If these parameters are similar to those used in their attribution reporting API, which uses the same core design and infrastructure, then those parameters provide very little meaningful privacy.

Quantifying the privacy loss in reporting is particularly important, because an ideal and leak-free version of Protected Audience might only reveal information through reporting. That makes the privacy parameters that are used at this stage critical to understanding the entire proposal.

A concern particular to this setting is the potential requirement to report on losing bids in auctions. This is not fundamentally difficult, but reporting on losses means either degrading privacy or reducing the utility of reporting on wins, especially considering that there are likely to be many more losses than wins that might need reporting.

k-Anonymity Design

Protected Audience proposes the use of what is labeled "*k*-anonymity". In this design, each use of the same advertisement URL is counted in a central server. Until enough attempts (*k*) are made to use the URL, the URL will not be available for use. This design does not follow the usual definition of *k*-anonymity as used in academic literature, but the underlying idea is similar: any information that might be revealed about someone through the system might have been revealed around *k* times before.

This *k*-anonymity is applied to all advertisements that win auctions. The same approach is also used for event-level reporting, though the inclusion of event-level reports is a temporary measure. A user agent is responsible for running the central service.

The user agent sends the advertisement URL to the service along with some information about the auction and interest group configuration. The service then indicates whether that URL has won a sufficient number of auctions under that configuration previously. The user agent will use a fallback advertisement if the advertisement has not reached the threshold.

The originally-stated reason for using *k*-anonymity is to ensure that ads are not microtargeted. That is, an ad is not shown to a single person or a very small group of people. A minimum threshold might limit the ability of advertisers to use targeting techniques that are sometimes regarded as being more manipulative. Microtargeting might also be used to make ad monitoring and accountability more difficult.

When querying the *k*-anonymity service, it is desirable to avoid revealing any client-specific information at the same time as a URL. This avoids having the service exposed to identifying information and URLs at the same time. For this reason, Oblivious HTTP is used with the Google Chrome *k*-anonymity service to ensure that the service cannot observe client IP addresses at the same time as a URL. This does not prevent the URL itself revealing private information the *k*-anonymity service, but the mixing of requests from multiple browsers makes it harder to correlate a single URL with a specific client.

Technical privacy solutions in this space, such as Private Information Retrieval or PIR are unlikely to be sufficiently efficient at this point in time. For that reason, the proposal regards the *k*-anonymity service as being trusted with private information. Google uses a trusted execution environment to implement their service.

The proposal suggests a threshold of 50 wins in the preceding 30 days before an advertisement is eligible for rendering. Auction wins prior to reaching this threshold count toward the threshold, but the winning advertisement can be replaced by a fallback.

Weak Differential Privacy Protections

Advertisements are therefore not available at precisely 50 auction wins. Differential privacy mechanisms are used to add noise to the threshold.

Adding noise provides additional privacy protection when other parts of the Protected Audience system leak information. Without a randomized threshold, a person could be targeted in an auction when the number of wins was exactly one below the threshold. If other parts of the system leak information, those leaks could be attributed to that person exclusively.

With a noisy threshold, the idea is that leaks need to be attributed to multiple auction wins, across multiple people. However, this anonymity set is not 50 people, but the smaller group who won auctions around the time that the threshold was crossed. The current proposed parameters mean that a ~98% confidence interval can be drawn from 35 to 58 (or maybe slightly more).

Any of the information leaks previously identified (auction failure, fetching ad content, ad interaction, or reporting) would be a more effective privacy violation if they could be matched to a single person. However, the *k*-anonymity design is not entirely effective in preventing this correlation.

Linking Leaks to People

An attacker can control the rate at which the *k*-anonymity service is queried. If queried at the same rate at which status changes can be observed, this reduces the anonymity set to a size of one. Changes in status therefore can be observed and attributed to a single person, at the cost of some amount of delay and some risk that the linking fails.

Linking an event to a single person means executing a probe when the interest group is precisely at the threshold. This threshold has a fixed random component and a separate noise component that is added at each update interval. The idea behind the attack is to supply a number of fake auction wins so that the count at the server is far enough below the target threshold that the threshold is not reached, but with a non-negligible chance of reaching the threshold with subsequent queries. From that set point, multiple probes

are then made, exploiting the fact that one component of the randomness changes at each update.

The person of interest is served an auction win with a minimum between each, as determined by the update rate. Though the odds of each successfully hitting the threshold are low, multiple attempts increase the odds greatly. Adding fake auction wins carries some risk of reaching the threshold, but this risk can be managed to balance the number of repetitions needed.

For instance, at around 46 fake wins there is about a 5% risk of reaching the threshold while adding fake wins, which would make that ad unavailable for the attack. Having the 47th input linked to a specific site visitor carries a greater chance of reaching the threshold. This increased chance could be close to guaranteed if the fixed offset added to the threshold is negative, or very small if the fixed offset is maximally large. However, in most cases the odds will be improved and repeated queries will eventually result in the threshold being reached.

Each update period randomizes part of the threshold, which adds uncertainty about whether a change in status was due to an auction win or a shifting threshold. However, there is no cap on the number of concurrent items that can be set up and queried in this way. Each success or failure releases a small amount of information, which can be averaged out over multiple trials. The only limitation is the rate at which queries can be made, which is one per period per URL.

The update rate will eventually be hourly, though it is temporarily limited to an update every 12 hours. A 12 hour update rate would make this style of attack far less practical. This is a case where the temporary form of the proposal is more private than the intended permanent form.

That rate limit is less effective when there are no similar constraints on the number of URLs. There is no effective limit on the number of attacks that can be mounted concurrently, except for the need to seed a number of auction wins for each URL.

Privacy Limitations of k-Anonymity Protections

Our threat model assumes that browsers are readily obtained by any entity. An adversary can therefore be assumed to have the ability to run instances of browsers in controlled conditions.

Circumventing the minimum threshold for auction wins is trivial under this assumption. Relatively few controlled browsers are needed to execute auctions with predetermined outcomes, so that advertisements clear the minimum win threshold and can be thereafter used for microtargeting.

Google Chrome attempts to limit this attack by requiring the use of unlinkable tokens that are tied to a Google account. Only logged in users will be able to contribute to *k*-anonymity thresholds for each URL. One negative effect of this is to increase the number of people that each ad is shown to, as Chrome users without a Google account likely cannot count toward the threshold.

It is potentially possible to harvest a finite number of tokens from genuine users for this purpose. Tokens are not ever made available to websites, so this would require the use of modified browsers with real accounts. However, the anonymity provided by the combination of anonymous tokens and Oblivious HTTP could protect an attacker from detection and any consequences for the use of tokens; excessive token issuance could be detected.

Alternatively, an adversary looking to seed auction "wins" for selected URLs to defeat *k*-anonymity can use chumboxes or other low-value placement areas on pages viewed by genuine regular users. It is also possible to take genuine, but decommissioned, ad campaigns and update those for use in targeting individuals.

No browsers is needed to probe the state of the database maintained by the *k*-anonymity service. A query option is provided that does not alter the server state and does not require a token, which makes queries cheap and efficient for an attacker. Differential privacy protections are used to avoid queries being used to watch for changes in status. As a result, this does not prevent a query from revealing that the associated ad URL won an auction, it just limits the rate at which such ads can be placed into auctions and subsequently queried.

Necessity of *k*-Anonymity

In an idealized Protected Audience design, the function of the *k*-anonymity service might be eliminated.

Though the original impetus for having *k*-anonymity might have been to suppress microtargeting, the primary privacy function appears to be limiting the impact of information leaks. That is, *k*-anonymity provides a second layer of defense for any leaks

in auction isolation, anonymous fetching, ad interaction, and event-level reporting, plus maybe failures in those trusted components that produce real-time information for auctions. These protections are not always effective, but they can make leaks — particularly one-off leaks — harder to use to target people.

As long as information leaks in the design are significant, the design relies heavily on *k*-anonymity to provide privacy. However, if those leaks were to be addressed somehow, the *k*-anonymity service would be redundant and could be removed.

Trusted Execution Environment Design

The proposal uses a Trusted Execution Environment (TEE) as a means of safeguarding private information in several different capacities.

A TEE is used in the implementation of the off-device execution of auctions, in the provision of the key-value servers that provide bidding logic with updated information, and in some implementations of the *k*-anonymity service. The current proposal does not currently require the use of a TEE for the key-value service, but Google indicates an intent to require that in future. This section will look more closely at the bidding and auction service as its primary example.

In comparison to other services, the *k*-anonymity service is wholly operated by browser vendors. In practice, alternative protections are possible, though Google opted to use a TEE. Use of a TEE for *k*-anonymity has fewer constraints on its operation than the other services and so it might not have the same weaknesses.

TEE Overview

In effect, a TEE is a computer that is attached to a more general purpose host computer, with two key capabilities. The first is the ability to provide assurances that it is running specific software. The second is to prevent the host computer from accessing its running state.

A host computer initializes a TEE with software that it selects. Thereafter, the host can only communicate with the software in the TEE through pre-arranged interfaces: those that are provided by the software inside the TEE. Though the TEE uses the memory of the host computer for its state, that state is encrypted so that the host cannot read it. The

host computer is only able to observe some effects, like the timing of messages and the approximate quantity of processing and memory resources that the TEE consumes.

The software that runs in a TEE can use facilities provided by the TEE to produce an assertion that is backed by the manufacturer of the TEE about the software itself. That is, a TEE can offer proof that it is running specific code. This allows external entities to initiate communication with the TEE and be confident that it is running a particular piece of software.

With appropriate use of encryption and authentication, these assurances can enable the processing of information that might otherwise not be available for processing. This has many applications, particularly for applications that have an interest in protecting privacy or commercial interests.

The protections that a TEE provides come with significant conditions:

- A TEE cannot protect against side channel leaks, especially timing leaks. For instance, if secret information affects the running time of an operation that can be externally observed, then that secret might leak.
- TEE manufacturers generally do not attempt to prevent someone with physical access to the host computer from being able to recover information. That means that someone with the ability to use imaging devices, physical probes, or power and fault analysis techniques might be able to recover information from a TEE.
- Enclaves are relatively new technology and a number of vulnerabilities have been discovered in TEE implementations.

Any system that uses a TEE needs to consider these risks and put mitigations in place for each.

Bidding and Auction Overview

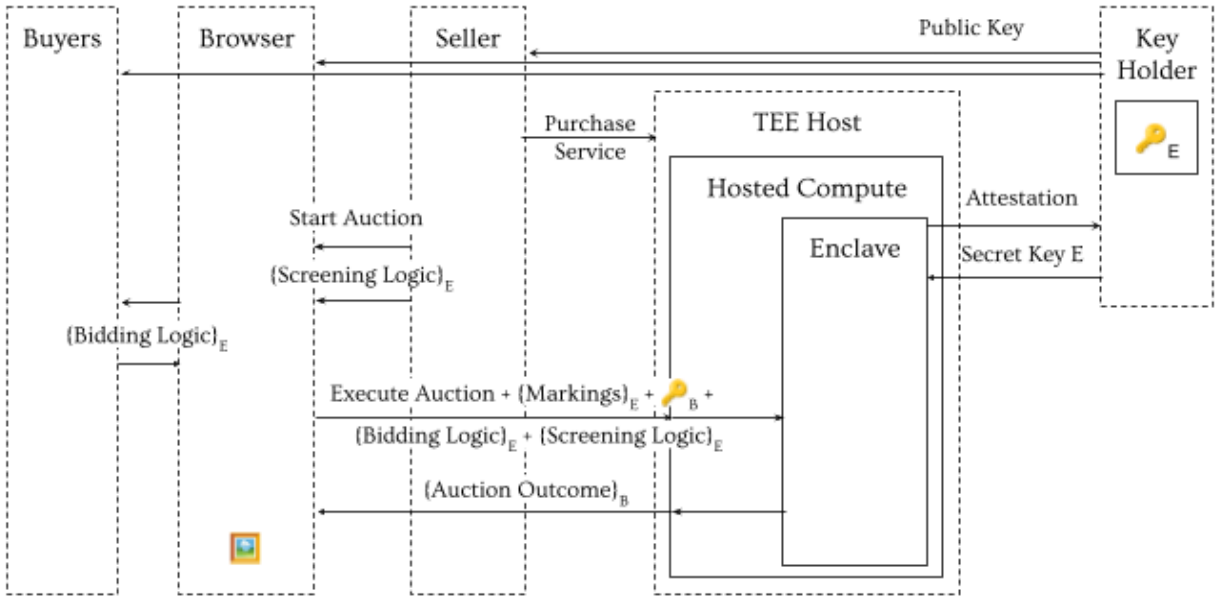
In all uses of a TEE in Protected Audience, confidential information is encrypted toward a key that is held by a trusted key holder. The key holder is trusted to only release that key to a TEE running approved software. For a bidding and auction service, trust is needed from the browser, buyers, and sellers. For a key-value server, when a TEE is eventually required, similar trust conditions apply.

The key holder makes the private key available to an instance of approved software that runs in the TEE. The trusted entity also verifies that the TEE is running in an

environment that has safeguards in place that counter some of the aforementioned weaknesses in the TEE threat model.

Once the approved software receives the encrypted information and private key, it can decrypt the information and execute the same basic process as a user agent would in the local version.

For the bidding and auction service, the process is visualized in the following diagram:



A similar process is used for other services, though some details are different. In general, the enclave is queried using a simple request-response exchange.

Setup and Trust

For the the bidding and auction service, there are several setup steps:

1. The browser, buyers, and sellers all nominate an independent key holder to hold a secret key (🔑E in the diagram). This entity is trusted to release the secret key only to certain software and in certain conditions, as agreed jointly by all parties.
2. The seller contracts with an independent TEE host. There are constraints on the operation of a TEE that mean that a seller will need to contract with a host that can be trusted to follow these constraints and cannot run the TEE themselves; see below.

3. The seller will create services within the hosted service provided by the TEE host and start the TEE, using the agreed software.
4. The TEE makes an assertion, with keys that are backed by the TEE manufacturer, toward the key holder and requests a secret key. This assertion might need to be paired with an assertion from the TEE host that this TEE instance is running in their infrastructure.
5. The key holder validates that the attestation is produced by a TEE manufacturer that it trusts, that the specific hardware is trusted, that the software that is running in the TEE is trusted, and that the TEE is running in infrastructure operated by a trusted TEE host. After these checks, the key holder releases the secret key.

Using a server depends greatly on trust to protect privacy. People (or their user agents) — and buyers and sellers — need to trust that the key holder will not reveal the key unless all of the following protections are adequate:

- The TEE host has safeguards that prevent access to the TEE that might allow information leaks. For instance, an adversary with physical access might be able to circumvent protections in the TEE and obtain the private key or the private information that is processed by the TEE. Attacks that rely on physical access are not protected against in a TEE threat model.
- The manufacturer of the TEE has adequate protections against leaking information to other processes on the same host or any remote entity. TEE technology is still relatively new and many vulnerabilities have been found in TEE implementations along these lines.
- The software that runs in the TEE will operate correctly and without side channels. It might be necessary to have supply chain integrity systems in place, so that the specific binary program can be reliably traced to source code that has been audited.

Stronger protections for the private key might also be helpful. A threshold secret sharing of the private key might be used to ensure that no single entity can release the private key. Distributing the private key across multiple, mutually distrustful entities might help protect against corruption of a single private key holder. Threshold sharing can also ensure that the private key is available even when a subset of the key holders are unreachable for any reason.

Once these conditions are met and setup is complete, the TEE is ready to accept requests to execute bidding logic.

Operation

The primary difference between a service running in a TEE and a regular HTTP web service is that the host system, which might be responsible for managing requests, is an adversary that should not receive private information. For instance, in the bidding and auction service case, the host computer runs software that the seller controls.

The payload of requests for the TEE therefore needs an additional layer of encryption to protect it from the entity that operates the server. Any information that is intended for the TEE is encrypted toward the secret key that is provided to the TEE by the key holder, using the corresponding public key. The response from the TEE is also encrypted toward a private key held by the browser; the corresponding public key is included in each request. Architecturally, this bears some resemblance to Oblivious HTTP, so it is unsurprising that Google has chosen to adopt the same message protection design in their specifications.

In some cases, information will be relayed via additional intermediaries — such as the information supplied by the buyers and sellers in an auction, which is provided to the browser for inclusion in a request to execute the auction. This information requires separate protection, if the TEE host is not permitted to access that information. For the bidding and auction service, the seller might not encrypt their information, but buyers will want to ensure that their proprietary information is protected.

Limitations of TEE-based Approach

Using a TEE in the described manner has a number of advantages and limitations.

Relative to alternative technology in the same space — such as multi-party computation (MPC) or fully-homomorphic encryption (FHE) — a TEE has relatively low performance overheads. Developing for a TEE involves some constraints, but these are not particularly onerous and do not represent a major challenge for the sort of purpose-built software that Protected Audience requires. Similarly, some performance overheads exist when executing code in a TEE, but — while older TEE implementations impose significant costs — newer TEEs have very modest performance overheads.

The primary downside of TEE is the number of potential points of failure where the corruption or breach of a single entity would result in compromise of all of the information that the system is built to protect:

- The technology backing TEEs is relatively new and the literature is filled with attacks on TEE implementations. Hardware or firmware that is broken needs to be identified and removed from the trusted set promptly.
- TEE manufacturers hold the root keys that are used in creating attestations. Access to these keys would allow for the creation of false attestations that would allow the hardware protections to be circumvented.
- The TEE host needs to apply constraints on the operation of the servers that include TEEs. This includes physical access limitations and strict controls on monitoring systems that might reveal information through power consumption, heat generation, radio frequency, and any other side channels that are identified as needing safeguards.

Software that is approved to run might have vulnerabilities or side channels that could be exploited. This software is run by an entity who has an interest in the information it processes, meaning that attacks on software that would be considered online attacks in other deployment configurations have to be treated as though they are offline attacks. This means that security assumptions that might be made for typical software deployments might not apply.

- Side channel protections need to be robust and granular. Greater allowances need to be made for the number of times that software can be invoked by an adversary, to avoid the amplification of small side channels into useful signals.
- The design cannot rely on protections provided outside of the software, like rate-limiting or firewalls.
- Stateful protections cannot be used, as state might be eliminated by a host.

This does not include other factors that are less likely to be problematic, even though there are some risks involved:

- The key holder. In the description thus far, the key holder is described as a singular entity. If the key holder were a singular entity, then they also represent a potential point of compromise. There are relatively simple cryptographic techniques that allow for a secret key to be jointly generated and held by a group to mitigate this risk. Jointly holding keys creates an operational risk in that it

increases the odds of an availability-affecting outage, but robustness can be improved through the use of threshold secret sharing and generation protocols. These would allow multiple key holders to jointly hold a key and only require a subset of those key holders to be available.

- Cryptographic algorithms. Many of the cryptographic techniques involved are well understood and use well-studied algorithms. The composition of these techniques will be subject to careful review that will mitigate the risks inherent in the use of cryptography. Similarly, though the design relies on classical cryptography and the development of a cryptographically-relevant quantum computer is an exigent threat, most of the cryptographic components used have established strategies for managing this risk.
- Software. Developing software for use in this system will require careful procedures, but those procedures are relatively well understood and the technology mature. The requirements for software do not depend on an extensive supply chain and so it should be possible to build software that can be traced to source code. Source code can then be published and scrutinized.

TEE reliance also introduces a modest centralization risk. Though browsers might allow advertisers to choose a TEE operator, the approval process for new TEE operators could make it harder for a new TEE operator to enter the market. A site that wants to use a new TEE operator might have to wait for all browsers to approve that operator before they can start using it.

Temporary Measures

This analysis has considered only those parts of Protected Audience that are explicitly permanent aspects of its design. The implementation of Protected Audience in Chrome includes a number of temporary measures that greatly weaken privacy protections. These temporary measures exist to either make the adoption of Protected Audience by the advertising industry or to make the implementation more tractable.

Adoption of Protected Audience depends on advertising companies, particularly Supply- and Demand-Side platforms, being willing to invest significant engineering resources into development and deployment. The complexity of the design also manifests in the complexity of its use in a number of ways.

The reduced information that Protected Audience makes available to advertising companies is probably the most significant impediment to adoption. Modern Web advertising systems are very complicated systems that can fail in surprising ways. Visibility into their operation is a tool that is used to ensure that they are functioning correctly. Removing most of that visibility is a serious barrier to implementing Protected Audience.

Therefore, it is understandable that Google chose to progressively enable those aspects of the design that reduce visibility into its operation. However, this means that the privacy protections that the system affords to people is significantly worse than the preceding analysis indicates, at least until these temporary measures are removed.

This section reviews these temporary measures and their consequences for privacy.

Revealing the Winning Ad URL

A very effective tool for debugging ad auctions is transparency about the outcome. Chrome offers a mechanism that simply reveals the URL of the ad that won an auction. Though this does not allow for detailed interrogation of the internals of the auction, it ensures that sites are able to observe the effects of auctions directly.

Providing this capability means that Protected Audience becomes a direct means of cross-site communication, in direct contravention of the privacy goals. However, where cross-site cookies exist and provide a more efficient means of achieving the same end, the net privacy effect is negligible.

This is the first temporary measure that will be removed along with Chrome's forced blocking of cross-site cookies. Sites cannot learn the winning ad URL unless they also have access to cross-site cookies.

Event-Level Reporting for Buyers

Similar to the interface that reveals the winner of an auction, a per-impression report can be requested by a seller and seller. The buyer report will be sent to a destination chosen by buyers; the seller report will go to a destination they choose.

Reports will include the URL of the winning ad, the value of the bid, the interest group involved, the identity of the script, and some additional information about the auction including the second place bid value. Supplementary data can be supplied by the buyer or

seller, which each can generate that signal based on information that is supplied by the other.

Reports will be suppressed unless they reach a *k*-anonymity threshold, with many of the aforementioned limitations on that protection. The *k*-anonymity protection applies to more fields than for displaying a winning ad, which might result in reduced report volume.

Reports will also include some amount of differential privacy protections for some of the fields that are included. However, this noise is unlikely to provide any meaningful privacy, as this only adds a 1% chance of randomizing just a few fields. Consequently, though it might make it marginally more difficult for advertising businesses to use the information, it provides no meaningful privacy protection as the primary means of passing information between sites (the URL) remains.

This feature has no timeline for removal and a commitment to be supported until 2026.

Network Access for Ad Creatives

The use of fenced frames is designed to ensure that advertisements cannot send information to sites. However, like other privacy measures, building an advertisement that cannot communicate severely limits what it can do.

Earlier versions of Protected Audience required that advertisers supply ad creatives in their entirety (using web bundles). However, this was found to be overly restrictive. Two primary concerns exist:

- The ability of advertisers to make timely changes to ad creatives is limited.
- Including multiple ad creatives, each with different sizes, added significantly to the size requirements for each interest group. Video ads in particular were considered too large for this approach to be feasible.

Removing communication restrictions on ads avoids these issues. However, it means that it is trivial for an ad to communicate the outcome of an auction to a server. Note however, that the ad does not have access to full interest group information, only the limited amount of information inherent in its own selection. This is significant information, because the number of possible ads is effectively infinite.

Ultimately, only *k*-anonymity protections constrain the information that can be obtained from each ad placement when communication is available. Though if a single ad is not enough, there is no limit on the number of ads that can be placed.

Effective isolation for advertisements starts with how they are fetched, but also includes their entire lifecycle, up to and beyond any interaction that might result in navigation of the page. Not preventing communication for the advertisement makes those other protections meaningless.

This feature will be supported until 2026, but no candidate replacement has been identified, unless the shortcomings of web bundles can be addressed.

Sellers Provide Real-Time Updates Directly

The use of real-time updates for interest groups relies on trusting the server that provides the updates. This server is exposed to private information about the state of an auction, including information from the seller that might be used to correlate cross-site activity.

The use of a TEE is the proposed method of ensuring that this service is trustworthy. That is, the approved TEE software will not pass what it learns about auctions to any other entity.

The information that is leaked when fetching real-time updates is significant. However, relative to other temporary measures, the effect on privacy is not as significant. This temporary loosening of protections is also one of the earliest to be removed, with removal planned for Q3 2025.

Microsoft's Ad Selection API

Microsoft has implemented a version of Protected Audience that they are naming the Ad Selection API. These proposals are more similar than they are different, with a few significant differences:

- Different markings can be merged if they have the same owner.
- Ad URLs do not need to be predetermined.
- Bidding and screening programs can only be run in a TEE, not the browser.

These changes do not significantly alter the privacy analysis, with some minor notes.

The added flexibility in bidding logic has no direct impact on overall privacy only to the extent that the choice of ad does not leak. Additional flexibility in ad choice makes the API far easier for advertisers to use, including being able to adapt the meaning of markings. That has some adverse effect on transparency in the sense that an interest group can more easily mean different things depending on context. Thus, it becomes harder to pin down what a given marking means.

To the extent that running auction logic in a TEE might be less desirable than running in the browser, the TEE requirement might be undesirable. That said, given that the privacy characteristics of both are similar, having an adversary be able to choose the weaker of two options could be worse than only permitting a single option.

In addition to these changes, Microsoft indicated a willingness to consider several features that could significantly worsen the privacy characteristics. In particular, the proposal to allow advertisers to run the TEE in their own data centers would create an uncontrollable privacy exposure. TEEs cannot protect against an attacker with physical access. The same consideration might apply to their consideration of the same *k*-anonymity servers, but, consistent with reservations about their necessity, this potential is not likely to affect privacy significantly.

Technical Privacy Situation

There are a lot of interacting components in Protected Audience. The preceding sections identified the following privacy issues across those components:

- Potential isolation weaknesses in both in-browser and TEE-based auction processing
- Leaking of whether an auction produced a result
- Leaking interest groups through fetching of
    - Bidding logic
    - Real-time updates
    - Ad creatives
- Ad interaction revealing selected ads
- Flaws in the application of *k*-anonymity protections

Some of these are acknowledged shortcomings, but there are no known mitigations for some.

These problems are in addition to the privacy problems caused by temporary measures. Any one of these temporary measures effectively negates the privacy that the full design might offer.

Even with temporary measures removed, there is no obvious way to address some of these issues. Some of these issues are inherent and fundamental. Some information leaks might be reduced by introducing additional constraints on use, but not all.

The goal is still to show ads. Showing ads causes information to leak.

That strongly suggests a conclusion that the advertising goals of Protected Audience are not compatible with its own technical privacy goals.

Looking more broadly at the proposal, there are some non-technical aspects to how Protected Audience seeks to integrate into other systems. The remainder of this document examines some of these aspects.

Transparency and Accountability

Protected Audience provides browsers the ability to present some information about the sorts of information that is being stored. This provides a measure of accountability for people for whom markings are being applied.

Browsers might allow people to inspect the markings that sites have applied, the markings that sites own, or both. Though browsers only require information about the site that owns markings, they can also store additional information that might aid accountability. Though markings might be owned by one site, they might be applied or updated by one or more other sites. Each marking might therefore include information about all of the sites that applied or updated it, so that markings can be presented for inspection when any of the involved sites are examined.

Each marking (or interest group) has one or more advertisements. These can be presented in each of the sizes that the marking lists. This might allow someone to determine what each marking means to an advertiser at the current time.

Other information that is attached to a marking, including its name, is not required to be comprehensible, so it is of limited value in terms of understanding what a marking means. Sites might use naming conventions that embed some information in names, but they are not obligated to do so.

Many Ads

The primary thing standing in the way of someone understanding what Protected Audience is doing with their browser data is one of scale. The large number of sites that might own or apply markings is large and the number of markings that a site can own is also large. As each marking can choose from a set of multiple advertisements, the number of advertisements that might be used at any given time could be very large. For that reason, it seems unlikely that any transparency system will allow for an easy understanding of what interest groups mean.

Updates to markings or the advertisements that they refer to (for unbundled advertisements) means that the set of advertisements can change. An advertiser can make it harder to understand what a marking means inadvertently simply by adapting their practices. This is unlikely to even be malicious — though that could make it possible to avoid being accountable — as changes to advertising campaigns are frequent.

The same basic marking can be applied to multiple people, each with a different meaning that is encoded in supplementary data that is unique to each person. Though the set of advertisements that can be used is limited, the supplementary data can be used to activate or deactivate the marking for some people. Any person that has a deactivated marking could be led to believe that the interest group has a completely different purpose. An update to the interest group could then change the advertisements and activate and deactivate different people.

This sort of activation/deactivation pattern should be unnecessary. Though large advertisers might find that people who are very active create more markings than the browser is willing to store, this can be managed in ways other than reusing markings.

To address the concern around changes to the ads in each marking, a browser might acquire advertisements that are bid upon, which can be presented alongside the current set of advertisements for each marking. This would ensure better transparency, but comes with two costs: the amount of space required to store ads and the network

bandwidth required to fetch them both increase considerably. This cost would have another benefit in that it removes leaks involved with ad retrieval.

Beyond Privacy

While the rest of this document is largely concrete and technical and focused on privacy, the remainder of the document is both speculative and opinionated. The idea is to look beyond the direct effects of the proposal and instead consider what effect a successful Protected Audience might have in the longer term.

Success Conditions

For Protected Audience to be successful, it has to provide advertisers with meaningful improvements in advertising efficiency that exceed the cost of implementation.

An improvement means that we need to compare the use of Protected Audience with something. Setting that baseline is not simple. The use of Protected Audience cannot be compared with a world in which tracking and profiling are trivially possible through the use of cross-site cookies. Nor can be assumed that tracking and profiling are totally non-existent.

Trials of Protected Audience provide less than perfect information. In a competitive market where advertising money can be directed at alternatives that can use tracking and profiling, any alternative is immediately at a disadvantage.

The introduction of technical limitations on tracking and profiling is unlikely to have immediate effect.

Limited forms of tracking are likely to remain viable options for advertisers seeking actionable information about their visitors. Though legislation in some jurisdictions could render some techniques unlawful, it is quite likely that some amount of tracking will be possible within set constraints. For instance, companies that are able to obtain linkable primary identifiers for visitors, such as email addresses, phone numbers, or similar. Sites that gain permission to use those identifiers will be able to use those for cross-site tracking and profiling.

Depending on the scale of these practices, tracking and profiling might be limited to establishing reference points for privacy-preserving systems like Protected Audience. Or it could remain as a viable option for advertising in the long term.

In the short term, sites that were visited by people whose browsers permitted the use of cross-site cookies have the ability to link the identity of visitors across sites. That ability remains for all visitors that retain cookies. A separate event, like changing to a new device, is needed to break any existing cross-site correlation. This could mean that the immediate effect of disabling cross-site cookies on cross-site tracking and profiling would be spread out over time.

If we accept that tracking-based advertising systems will eventually be infeasible, Protected Audience does not necessarily need to be more effective than those systems that rely on tracking and profiling. Success only depends on being better than the alternatives that are presently available and only in the aggregate.

Advertising Market Concentration

Protected Audience is one of the Privacy Sandbox features that are specifically targeted at assuaging competition concerns held by regulators. This includes the UK Competition and Markets Authority, with whom Google has signed voluntary commitments.

The Effect of Operational Complexity

Use of Protected Audience comes with significant costs, both in the fixed aspects of adaptation and in the ongoing operational expenditure that the system requires.

If modes of advertising that use Protected Audience are more effective than other options — a necessary precondition for success of the proposal, as noted — then the benefits for small sites need to be greater than the increase in variable costs of operation.

For smaller sites that have less native access to information about their visitors, the potential gap between uninformed targeting and targeting using Protected Audience might be enough to justify the greater expenditure needed to realize any benefits. Sites with more access to information will see less relative benefit. The theory is that these sites will also be larger and so will also be better positioned to amortize any fixed development costs, which the complexity of the proposal virtually guarantees will be high.

An analysis on that basis suggests that maybe market conditions are likely such that the inherent skew introduced by a higher fixed component in operating costs could be absorbed by the market.

Small sites might need to effectively pool their resources by outsourcing the responsibility for development and operations to an ad tech company. Such an intermediary can realize benefits of scale by serving multiple sites.

Outsourcing capabilities is already a reality of the advertising market. This is largely due to the highly competitive nature of the business and the high levels of expertise needed to maintain comparable performance to other participants. Protected Audience clearly aims to rely on existing outsourcing arrangements, rather than being something that small website operators can independently deploy.

Greater operational costs narrows the window where advertising modes like retargeting are efficient. That is, while the use of retargeting, or other capabilities that Protected Audience enables, might be able to produce more revenue for market participants, a greater share of that revenue will need to go to the intermediaries that provide the capability.

A major goal that Protected Audience might achieve is to provide some redress for market inequity between sites. However, in doing so, the complexities involved with its use might also negatively affect the balance between ad tech companies and their customers: those who operate websites and those who buy advertising.

Addressing Inequities Between Sites

Websites and advertisers represent a much larger stake in the balance created by changes in the way that online advertising operates.

The benefits provided by Protected Audience might be higher for sites that lose the most from loss of cross-site tracking. That is, the sites that know the least about their visitors obtain the greatest marginal benefit from being able to use cross-site information for ad targeting.

On that basis and in the context of deploying technical measures that limit tracking and profiling, Protected Audience might then be viewed as a proportionate response to the reduction in the availability of cross-site information. Sites that start from a position of

knowing more about visitors gain less because the incremental gain in value is smaller relative to the fixed costs of implementation.

Alternatively, support for Protected Audience might be framed in terms of compensation for the loss of information that comes from deploying anti-tracking and anti-profiling measures. That line of reasoning is dangerous as it presupposes that the action of protecting privacy harms genuine equities in a way that requires restitution. That implicitly grants the practice of tracking and profiling an unjustified credibility. The opposite is true: deploying privacy-protective measures only seeks to end a dependence on abusive practices. The information gathering practices employed by advertisers are extractive far out of proportion to the value that they return to the people they exploit. Protected Audience should not be justified on such a basis, when more positive rationale exist.

Browser Competition

Protected Audience is massively complex. Increases to the complexity of the Web exert an innate pressure toward consolidation of implementations. More complexity means that it is harder for a new entrant to enter the browser market.

Complexity alone is not sufficient cause to regard Protected Audience as creating unfair advantage for larger actors. However, there are other structural characteristics that potentially disadvantage browsers that are used by fewer people. These problems are inherent to any system that relies on centralized systems that place privacy and utility in balance.

However, though not necessarily central to the design, the *k*-anonymity service ensures that browsers with fewer users will not be able to reach thresholds as quickly as other browsers. This makes it more difficult for small advertising campaigns to reach people who use less popular browsers. Protected Audience already introduces delays before new ads are available for display, but these delays are inversely proportional to the size of the user base of a browser.

This disparity is exacerbated when it comes to the (temporary) event-level reporting that is deployed. Adding more conditions to *k*-anonymity thresholds means that more reports are needed to reach those thresholds.

A smaller browser might be able to reduce *k*-anonymity thresholds to compensate for this. Given that size disparities in the browser market start in the order of a multiple of

ten, any correction along those lines greatly diminishes any privacy protection that the feature might provide.

The Future of Personal Information Use

The positive vision of the effect that Protected Audience might have on competition in advertising is one that focuses on market imbalance. However, it takes an advertising-centric view of the system that is narrow. In a sense, Protected Audience is the product of the conjunction of competitive pressure on privacy and antitrust fears, so this is a natural conclusion.

When viewed from the broader perspective of how information about people might be used, by whom, and for what purposes, the discussion is far more interesting. Google is making a far bigger statement.

Protected Audience is one of a few examples of a new paradigm for thinking about personal data. In this view, the collection of data is prevented, but the data is made available for selected uses. In this way, there are controls on those uses.

This is a viewpoint that Mozilla is supportive of, but there are some important questions that need to be discussed. Google's proposal presupposes the answers to these questions, but it is important to highlight them. The key question being:

> If there is a societal benefit to be realized by making personal information available for use, who decides which uses are acceptable and which are not? More importantly, how is that sort of decision made?

Protected Audience proposes one answer to that question. It lays out a specific way in which private information about what people do on the Web might be made available for use in targeting advertising.

The question of how to decide what is acceptable obviously contains a more difficult question of who might benefit from the use of information. Protected Audience answers that question by allowing the benefits that come from the use of information to be privatized. That answer is an unavoidable conclusion from the goals it sets, but it is not similarly inevitable that the design carries a slight bias toward ad tech intermediaries over those that supply money (advertisers) and audience (publishers).

We observe parallels in this presumption of being able to privatize benefits to the attitude of Google and their peers when it comes to the use of copyrighted information for use in teaching machine learning models. There are obvious benefits to be gained through the use of that resource, but we do not concede that this is the only possible way to allocate the consequent benefits.

One of the primary advantages of separating use of information from its gathering is that it introduces the possibility that any usage can be regulated. A discussion can then be had about the merits of different uses and collective decisions made that balance the concerns of multiple stakeholders. There is an amazing opportunity to move from a system that is hostile to privacy – with its opaque system that trades in personal information – to a more open and accountable system of governance.

In part, Protected Audience embodies the potential for that vision. Protected Audience includes some major components that would allow data to be made available for uses that are subject to both personal and collective control. That the same design also entrenches a particular answer about how data usage is to be regulated is perhaps the most disappointing aspect of the design. The Protected Audience design forecloses on the potential for usage to be subject to open governance, instead providing a comprehensive plan for how that personal information shall be used.