Protected Audience talk for TPAC 2023

- The Problem We Are Trying To Solve
    - Context: Privacy Sandbox, 3pcd, publisher revenue loss, esp. news publishers
    - Our goal: a way for ad tech to pick which ads to put on pages, without needing or offering a way to recognize the same person across different websites
    - There are basically two plausible approaches: "The Easy Way" and "The Hard Way"

- The Easy Way: Give the people who pick ads a little ad targeting info
    - Q: Is it possible for the browser to offer some amount of information which will not allow tracking and will not make users unhappy, but will let websites recover a reasonable amount of that revenue?
    - A: We think so. Topics API is our proposal in this space.
    - Structurally familiar: call a function, get some information, use that information
    - I lied, it's not actually easy.
    - Problem (with the entire approach, not just with Topics API): The information returned can be added to a profile associated with the user's first-party cookie and accumulate over time. We can slow that down but with this model, we can't eliminate it.

- The Hard Way: Let the people who pick ads use some information without revealing it
    - Is that even possible?
        - We think so.
    - Essential ingredients:
        - Some repository of "secret information"
        - Some way to make decisions about what ad to show using that secret info, without revealing secret info in the process
        - Some way to render the chosen ad WORSI
        - Some way for money to move around (dictated by ads business) WORSI
        - Some way for parties involved to learn what happened WORSI
        - Some feedback mechanism for the choice (ML model training) WORSI

- Chrome Privacy Sandbox has a family of answers to all of these.
    - Actually, we have two families of answers to all of these
    - One we are shipping now. This has leaks — does not satisfy "WORSI" with an adversary trying to exfiltrate the SI.
    - One that will have the desired privacy properties. There is still a lot of work to get there.

- Central question of picking ads based on secret information: Protected Audience API (previously FLEDGE, previously TURTLEDOVE). Incubation since 2020, and what I will

talk about this hour. Other parts of Privacy Sandbox include solutions for some of those needs:
    - Rendering: Fenced Frames (1hr WICG slot tomorrow)
    - People learning what happened: ARA (for attribution, 1hr WICG slot tomorrow, and also lots of PATCG related discussion), PAA (noisy histograms)

- What Is the Secret Information?
    - **Protected Audience concept of an "Interest Group"**
        - **1. A collection of (URLs of) ads and ad-pieces the IG might want to show**
        - **2. (URL of) JS function that can produce a "bid" i.e. a way for the IG to participate in an auction among multiple IGs**
        - **3. Some locally-stored data, inputs to that JS bidding function**
        - **4. Some keys to look up remotely-stored real-time data, also inputs to that bidding function**
        - **5. A URL to enable periodic updates of the above from a server**
    - Rest of talk: Discuss each one of these — five interesting design questions
    - This is not the whole Protected Audience API
        - For example: Lots of complexity in the actual choice of what ad appears, including control by the party whose site the ad is going to appear on (and the ad techs that they delegate to)
        - Can't fit everything into one hour. TBH can't fit it all into one hour *per week*, which is the rate at which we hold WICG calls, because with an hour every other week we weren't finishing our agenda.

- **1. "URLs of ads the IG might want to show"**
    - Original design: IG contained Web Bundles, rendered entirely offline
        - Why did we want "rendered entirely offline"? If you load the contents of the ad over the network, then any server that serves a resource used in rendering the ad gets to learn that the ad just won an auction, and may be able to conduct a timing attack.
        - However, Navigable Web Bundles seem to have less support now than when we proposed this in 2020
    - So we're using good old URLs
    - Some mitigation against tracking a person using the timing attack:
        - Rendering in Fenced Frames, which cut communication between the subframe and the surrounding page. Currently optional in Chrome, but will eventually be required. Now a server can tell that the ad URL got rendered and when, but doesn't know what page it was on.
        - k-anonymity of the ad render URL. Privacy Sandbox project is running a k-anonymity server: count how often a particular URL wins the Protected Audience auction, only allow an ad to render if it has won recently in enough different browsers. Using k=50 (plus DP noise!) in the past week.

Now a server can tell that the ad URL got rendered and when, but doesn't know who the ad was shown to. (Yes, this could surely be circumvented.)
    - Would love a better answer here.
        - Seems like "Web Tiles" and "Content-addressable" and "IPFS" include attempts to solve a similar problem, someone please tell me if you think they fit the bill here.

- **2. "JS function that produce a bid"**
    - These JS functions process both the IG data and the surrounding page data — so we must assume the information they process implies some cross-site identity match.
    - That means we need to let these functions process data but must prevent them from exfiltrating that data
    - Original plan: On-device computation in an isolated worklet. Two separate problems:
        - On-device? Some ad tech will want to use lots of compute
        - Isolated worklet? Side channels, e.g. to exfiltrate data to an ad tech script running in the main page at the same time the worklet is running the auction. (We've done a lot to mitigate, but not bullet-proof.)
    - Early testing of "Bidding & Auction Service": push this JS execution to a server, in a sandboxed v8 JS environment running in a TEE.
        - Google-written, open-source code for the execution environment, ad tech's bidding JS running inside
        - Hosted on a trustworthy cloud provider, key management by two independent coordinators who perform remote attestation of the TEE
        - Only helps if you believe in TEEs running on a trustworthy cloud provider
    - If you don't believe in the non-exfiltration properties of on-device worklets or TEEs, then what?

- **3. "Bids using some locally-stored data"**
    - Protected Audience is a purpose-built API. This puts the browser in a technical position to act as the user agent and be opinionated about the use of data.
    - *Opinion: IGs contain data from only one site*. The site the user was visiting when some ad tech called the joinAdInterestGroup API. In a world where we've removed 3p cookies and all other ways to recognize the same user across different sites, IGs can only bid based on data from a single site (beyond just the one where the auction is happening and the ad will appear).
    - *Opinion: IGs can last for 30 days*. So with Privacy Sandbox, 31d after the most recent time you've visited site X, nothing about what you've done on site X can influence the ads you see.
    - *UX Implication*: *Some kind of answer to "why did I see this ad?"* If an ad comes out of the Protected Audience auction, then the browser knows that it did, and that the information that led to that ad being in the browser came from your visit

to site X last Thursday. This is genuinely different from 3PC "Why did I see this ad? Because of everything you've ever done online."
    - *UX Implication: I don't like this ad.* If you don't like an ad you saw, and you saw it because of your visit to site X last Thursday, then the browser can delete all IGs that you were added to on site X. The root information is now gone! If you want to go back to site X and still not see ads based on it, the browser can just not retain IGs from site X visits. This is genuinely different from 3PC "Don't show me this ad any more" functionality which might perhaps disable one delivery channel for an ad, but you may well see the same ad through other delivery channels based on the same underlying events observed by many parties.

- **4. "Keys to look up remotely-stored real-time data"**
    - Ads need real-time data, e.g. budgets, "Do I have any money left to spend?
        - "Set of ad campaigns whose budget you need to check" is too identifying to leak
    - Sounds like a job for Private Information Retrieval ("get value from remote database without DB owner knowing what value you looked up")
    - State-of-the-art PIR not up to this task when DB is very large and changes very rapidly. (Yes these are both the case — budgets were one example, but in practice this is not limited to one bit per ad campaign budget.)
    - Fall back on the Privacy Sandbox standard answer: Key-Value server running in a TEE, so that we know for sure that it is not logging which sets of keys' values got looked up at the same time as one another.

- **5. "updates of all the above from a server"**
    - Key use case: Visit site X, advertiser learns you're interested in their product, a week later it goes on sale. They want to tell you (and at least 50 other people).
    - Original proposal: Interest Group contains a URL, once a day the browser contacts that URL to get updated IG data. Since an IG just contains data from a single site, there is no leakage, right? The server is just getting back the very same info that it put in the browser. (Wrong.)
    - Actually, two new pieces of info: Time and IP address.
        - IP: server saw you on IP#1, but 24hrs later you might be on IP#2, don't want to give out a way to join those up.
        - Time: Just because a user touched a server at some time doesn't imply OK to touch it later from a different context, e.g. home vs work wifi.
    - Mitigation 1: Instead of updating "in the background", update only after the IG was invited to be in an auction, i.e. a time when you might contact that server anyway (if its ad gets displayed)
    - Mitigation 2: IP-blinding proxy? https://github.com/GoogleChrome/ip-protection Since IP is such a fingerprinting surface, such a proxy seems inevitable.
    - Mitigation 3: Yet Another TEE?

- Discussion!