Privad: Practical Privacy in Online Advertising

Paper #82, 14 pages

ABSTRACT

Online advertising is a major economic force in the Internet today. The revenue from well-targeted ad placement underlies the lucrative business models of many online services including search, email, social networks and, indirectly, ISPs and data centers. Today's deployments, however, erode privacy and degrade performance as browsers wait for ad networks to deliver ads. We present Privad, a practical private online advertising system. Privad serves ads from the endhost; this is attractive from three standpoints — privacy, profit, and performance. Tracking the user's profile on their computer and not at a third-party improves privacy. Better targeting and potentially lower operating costs can improve profits. And relying more on the local endhost rather than a distant central third-party improves performance. An anonymizing proxy hides the network address of the client, while encryption prevents the proxy from viewing client messages. The system design and security analysis presented in this paper cover all aspects of a practical and deployable system, including ad dissemination, ad auctions, view and click reporting, and click-fraud detection. As proof of concept, we have implemented Privad and deployed it on a small scale. We argue in this paper that, while Privad's security is not bulletproof, it substantially improves on the status quo, and represents a legitimate alternative to today's centralized ad networks.

1. INTRODUCTION

Online advertising is a key economic driver in the Internet economy. It funds services provided by such industry giants as Google and Facebook, and helps pay for data centers and, indirectly, ISPs. Internet advertisers increasingly work to provide more personalized advertising. Unfortunately, personalized online advertising, at least so far, has come at the price of individual privacy [16] and poor user experience. And while privacy advocates would like to put an end to advertising models that violate privacy, aside from a few highly publicized battles, they have had little success with the more entrenched ad brokers like Google and Yahoo! [11]. Arguably the reason why privacy advocates have failed is that they offer no viable alternatives. The deal they offer, privacy *or* personalization, is not acceptable to the entrenched players. This paper takes a first stab at providing that alternative.

In this paper, we present a practical private online advertising system, which we call Privad. Our high-level goals for Privad are that it:

1. is private enough, and certainly substantially more private than current systems,
2. is at least as scalable as current systems,
3. targets ads as well as or better than current systems, and
4. fits within the current business framework for online advertising.

As these goals suggest, we are looking for a design that finds a sweet spot between privacy and other practical aspects of the system (scalability, targeting, business model). Privad preserves privacy by maintaining user profiles on the user's computer instead of in the cloud. A small amount of information necessarily leaves the user's computer: coarse-grained classes of ads a user is interested in (for scalability), the ads the user has viewed or clicked on, the websites that carried the ads, and the ranking of ads for auction purposes (for accounting). This information, however, is handled in such a way that no party can link it back to the individual user, or link together multiple pieces of information about the same user. An anonymizing proxy hides the user's network address, while encryption prevents the proxy from learning any user information.

A key question is, how private is private enough? Current advertising systems, such as Google and Yahoo!, are in a deep architectural sense not private: they gather information about users and store it within their data centers. Users typically have no control over how and when this data is used. Nor do these systems lend themselves to being audited by privacy advocates or regulators. Users are essentially required to completely trust these systems to not do anything bad with the information.

Privad is considerably more private than current systems. Privad does not, for instance, require trust in any single organization. Additionally, Privad is designed to be auditable by third-parties. Most of this auditing is automatic, through the use of a simple reference monitor in the client. While Privad raises the cost of misbehavior substantially, Privad's security protocols are not bullet-proof, and so Privad allows the use of human-assisted or learning-based monitoring to detect misbehavior at the semantic level. But is Privad private enough? There is obviously no single universal answer to this question. We believe that ultimately

it is up to society to decide what is private enough, and society here tends to be represented by consumer and privacy advocacy groups like the Electronic Frontier Foundation (EFF), the American Civil Liberties Union (ACLU), and others [6]. Our strategy, then, is to design and build a system that is as private as possible while still achieving the practical goals, and to then see whether privacy advocates and regulatory agencies want to support or oppose Privad.

Another key challenge is incentivising deployment. Privad is not aimed for users that today disable ads altogether. For users that do view, and occasionally click ads today, deploying requires first that Privad not degrade user experience in any way. We can ensure this by only showing ads in the same ad boxes that are common today (unlike previous adware, which employed disruptive advertising). Second, especially early on there must be some positive incentive for users to install it. This could be done through bundling other useful software, shopping discounts, or other incentives. Finally, it requires that privacy advocates (e.g. EFF, ACLU, and government agencies) endorse Privad. This at least prevents anti-virus software from actively removing Privad from clients. Ideally, it even leads to privacy-conscious browser vendors (e.g. Firefox) or operating systems installing it by default, or by governments mandating that existing advertising companies deploy Privad technology.

The contributions of this paper are as follows: it presents what is to our knowledge the first complete *practical* private advertising system. It describes the design of Privad, and contributes a security analysis including both privacy and click-fraud aspects. While this paper does provide a brief summary of our implementation, evaluation, and deployment efforts, full details can be found in [4].

That said, we readily acknowledge that advertising is not the only privacy issue that plagues the Internet, or even the most important (identity theft comes to mind). Advertising is, however, an important problem, and one that is not isolated from other privacy issues. For instance, arguably the primary motivation for social networks to gather private and Personally Identifying Information (PII) is ultimately in support of advertising. Overall, Privad, along with its proof-of-concept implementation and pilot deployment, represents an argument that highly-targeted practical online advertising and good user-privacy are not mutually exclusive. We hope that this first stab at a feasible design leads to additional research on privacy in advertising as well as on privacy in other aspects of online life.

2. PRIVAD OVERVIEW

There are six components in Privad: client software, client reference monitor, publisher, advertiser, broker, and dealer (see Figure 1). Publisher, advertiser, and broker all have analogs in today's advertising model, and play the same basic business roles. *Users* visit *publisher* webpages. *Advertisers* wish their ads to be shown to users on those webpages. The *broker* (e.g. Google) brings together advertisers, publishers, and users. For each ad viewed or clicked, the advertiser pays the broker, and the broker pays the publisher.

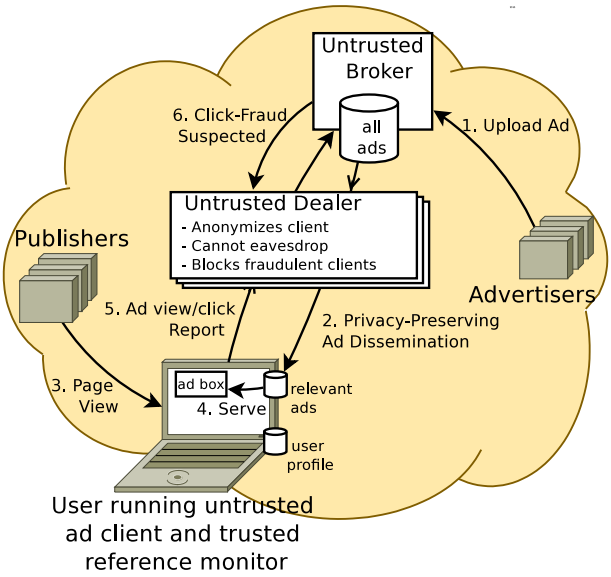

Figure 1: The Privad architecture

There are three new key components for privacy in Privad. First, the task of profiling the user is done at the user's computer rather than at the broker. This is done by *client* software running on the user's computer. Second, all communication between the client and the broker is proxied anonymously by a kind of proxy called the *dealer*. The dealer also coordinates with the broker to identify and block clients participating in click-fraud. Finally, a thin trusted reference monitor between the client and the network ensures that the client conforms to the Privad protocol and provides a hook for auditing the client software. Encryption is used to prevent the dealer from seeing the contents of messages that pass between the client and the broker. The dealer prevents the broker from learning the client's identity or from linking separate messages from the same client. The dealer is run by a consortium that is itself untrusted with user profile information, but is nevertheless unlikely to collude with the broker. This could for instance be arranged by having prominent privacy advocacy groups (e.g. EFF or ACLU) or government regulatory agencies like the Federal Trade Commission (FTC) participate in the consortium.

At a high level, the operation of Privad goes as follows. The client software monitors user activity (for instance webpages seen by the user, personal information the user inputs into social networking sites, the

contents of emails or chats sessions, and so on) and creates a user *profile* which contains a set of user *attributes*. These attributes consist of *interests* and *demographics*. Interests include products or services like `sports.tennis.racket` or `outdoor.lawn-care`. Demographics include things like gender, age, salary, and location.

Advertisers upload ads to the broker, including the bid and the set of interests and demographics targeted by each ad. The client requests ads from the broker by anonymously subscribing to a broad interest category combined with a few broad non-sensitive demographics (gender, language, region). The broker transmits ads matching that interest and demographics. These ads cover all other demographics, and so are a superset of the ads that will be shown to the user. If the user has multiple interests, there is a separate subscription for each interest, and the broker cannot link the separate subscriptions to the same user.

Ad auctions determine both which ads are shown to the user and in what order. In addition to bid information, ranking is based on both user and global metrics. User metrics include things like how well the targeting information matches the user, and the user's past interest in similar ads. Global metrics include the aggregate click-through-rate (CTR) observed for the ad, the quality of the advertiser webpage, etc.

When the user browses a website that provides ad space, or runs an application like a game that includes ad space, the client selects an ad from the local database and displays it in the ad space. A report of this *view* is anonymously transmitted to the broker via the dealer. If the user clicks on the ad, a report of this *click* is likewise anonymously transmitted to the broker. These reports identify the ad and the publisher on who's webpage or application the ad was shown. Multiple reports from the same user cannot be linked together by the broker. The broker uses these reports to bill advertisers and pay publishers. The broker also forwards the reports (or summaries) to the advertisers so that they may better manage their ad campaigns.

Unscrupulous users or compromised clients may launch click-fraud attacks on publishers, advertisers, or brokers. Both the broker and dealer are involved in detecting and mitigating these attacks (Section 3.4). When the broker detects an attack, it indicates to the dealer which reports relate to the attack. The dealer then traces these back to the clients responsible. The mitigation strategy is for the dealer to suppress reports from attacking clients.

Users, or privacy advocates operating on behalf of users, must be able to convince themselves that the client cannot *undetectably* leak private information. While having a trusted third-party writing the client software appears at first glance to be an option, it doesn't solve the problem — a trusted client simply moves the trust

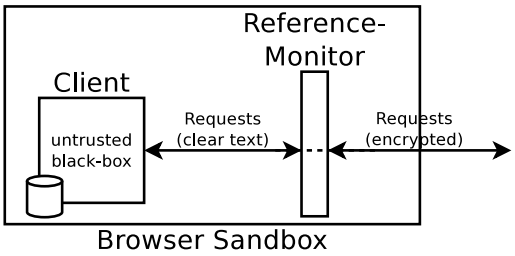

Figure 2: The Client framework

users place on brokers today to the third-party. At the same time, it requires brokers to make their trade-secret profiling algorithms known to the third party, and potentially to parties auditing the client. The reference monitor placed between the client and the network gives users and privacy advocates a hook to detect privacy violations (Section 3.5). It treats the client in a black-box manner (Figure 2), allowing the broker to use existing technological and legal frameworks for protecting trade-secret code. The reference monitor itself is simple, open source, and open to validation so its correctness can be verified, and can therefore be trusted by the user.

Finally, there may of course be multiple competing brokers each with a client on a given user's computer. These clients could operate independently of each other, for instance with each client fully implementing the Privad protocol, scraping webpages, and even arranging for separate dealers. Alternatively, there could be some common support in the user's browser to handle multiple clients more efficiently, for instance, by sharing a common Privad protocol implementation and common webpage scraping modules. Multiple brokers could also share dealers.

3. PRIVAD DETAILS

This section provides details on ad dissemination, ad auctions, view/click reporting, click-fraud defense and the reference monitor. It also puts forth some of the rationale for our design decisions. These details represent a snapshot of our current thinking. While ad dissemination, reporting, and reference monitor are quite stable, the click-fraud defense, and auctions may easily evolve as we do more analysis and testing. We additionally mention briefly optimizations for crypto operations, and our current designs for user profiling, and post-click behavior; interested readers may refer to a longer technical report for details [3, 4]. We provide them here so as to present a complete argument for Privad's viability.

We discuss privacy and security concerns of each component in the next section.

3.1 Ad Dissemination

The most privacy-preserving way to disseminate ads would be for the broker to transmit all ads to all clients. In this way, the broker would learn nothing about the

Figure 3: Message exchange for pub-sub ad dissemination. $E_x(M)$ represents the encryption of message M under key x. B is the public key of the broker. C is a symmetric key generated by the client for only this subscription.

clients. In [12], the authors measured Google search ads and concluded that there are too many ads and too much *ad churn* for this kind of broadcast to be practical. They observed that the number of impressions for ads is highly skewed: a small fraction of ads (10%) garner a disproportionate fraction of impressions (80%). Furthermore, this 10% of ads tend to be more broadly targeted and therefore of interest to many users. It may therefore be cost effective to disseminate only this small fraction of ads to all users, for instance using a P2P mechanism like BitTorrent. For the remaining 90% of ads, however, a different approach is needed. Therefore, we design a privacy-preserving pub-sub mechanism between the broker and client to disseminate ads.

The pub-sub protocol (Figure 3) consists of a client's request to join a *channel* (defined below), followed by the broker serving a stream of ads to the client.

Each channel is defined by a single interest attribute and limited non-sensitive broad demographic attributes, for instance geographic region (city granularity), gender, and language. The purpose of the additional demographics is to help scale the pub-sub system: limiting an interest by region or language greatly reduces the number of ads that need to be sent over a given channel while still maintaining a large number of users in that channel (in the k-anonymity sense). Channels are defined by the broker. The complete set of channels is known to all clients, for instance by having dealers host a copy (signed by the broker). A client joins a channel when its profile attributes match those of the channel.

The join request is encrypted with the broker's public key (B) and transmitted to the dealer. The request contains the pub-sub channel (*chan*), and a per-subscription symmetric key (C) generated by the client and used by the broker to encrypt the stream of ads sent to the client. The dealer generates for each subscription a unique request ID (*Rid*) that is unrelated to the client. It stores a mapping between *Rid* and the client, and appends the *Rid* to the message forwarded to the broker. The broker attaches the *Rid* with ads published, which the dealer uses to lookup the intended client to forward the ads to.

The broker determines which ads should be sent and for how long they should be cached at the client. For instance, the broker stops sending ads for an advertiser when the advertiser nears his budget limit. Note that not all ads transmitted are appropriate for the user, and so may not be displayed to the user. For instance, an ad may be targeted towards a married person, while the user is single. Because the subscription does not specify marital status, the broker sends all ads independent of marital status or other targeting, and the client filters out those that do not match. Over time, the broker can estimate the number of ads that must be sent out for a particular advertiser to generate a target number of views and clicks.

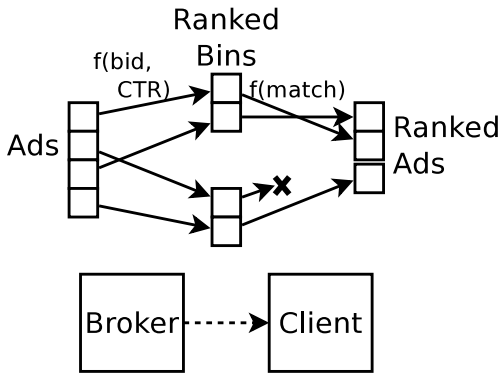

Figure 4: Design-I: Simple Auctions. For each pub-sub channel, broker bins ads by bid and global click-through rate. For each bin, client ranks ads by quality of match, filtering ads that don't match the user.

3.2 Ad Auctions

Auctions determine which ads are shown to the user and in what order. The goal of the auction is to provide a fair marketplace where advertisers can influence the frequency and position of their ads through their bids. The ordering may incorporate both global metrics and user metrics. The challenge, of course, is in doing so while preserving the privacy of the user as well as the advertiser's bid. Specifically, the user should not be able to learn anything more than they can today, i.e., they may learn the final order of ads, but not what the advertiser bid.

As a proof of viability, we present two auction designs that meet our requirements. The first design implements a basic auction. The second design, which is more complex, implements the GSP auction used by Google today [8] within the confines of the Privad model. Other privacy-preserving auction designs may be possible, and are for further study.

3.2.1 Design-I: Simple Auctions

One simple approach is to perform auctions during the ad dissemination phase (Figure 4). For each pub-sub channel, the broker bins ads by some function of the global metrics (e.g. product of bid and click-through-rate) and sorts the bins in decreasing order. Thus an ad that bids half as much as another ad but is three times more likely to be clicked is put in a higher ranked bin than the other ad. The ranked bins are sent to clients subscribed to that channel. The client sorts ads

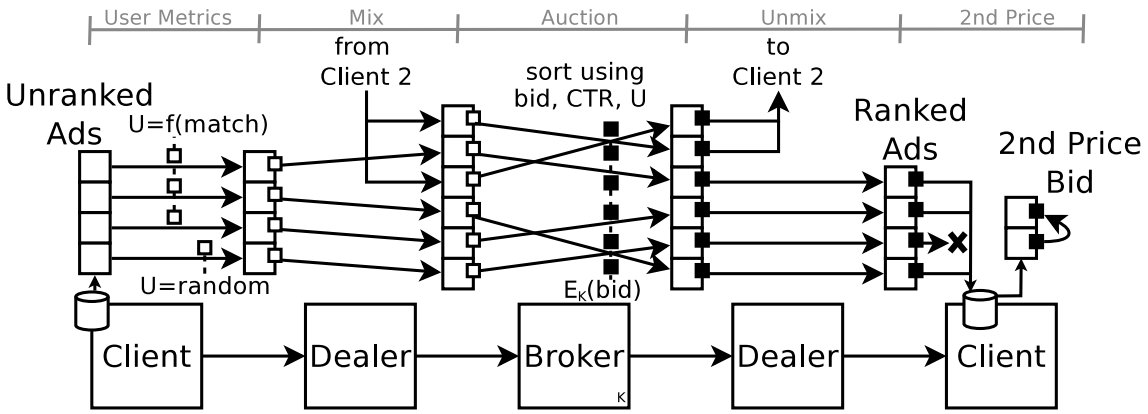

Figure 5: Design-II: Combined Auctions. Client annotates ads (across all channels) with quality of match, or random number if the ad doesn't match the user. Dealer mixes annotations from multiple clients. Broker ranks ads by bid, global click-through rate, and match quality, and annotates the result with opaque bid information. Dealer slices auction result by client. Client filters out non-matching ads. Client reports second-price bid on click.

within each bin based on local user metrics (e.g. quality of match). When an ad box is encountered, the client picks a channel to show ads from; the ads are shown in ranked order. The advertiser is charged the amount he bid for each click report (and a fraction of the bid for each view report). The broker periodically repeats this process, excluding advertisers that reach their budget limit. Note, just like today, the user only learns some function of the bid and CTR, but since the CTR is known only to the broker (and the advertiser), the user cannot learn the bid.

While this approach is simple, functional, and preserves user and advertiser bid privacy without any changes to the ad dissemination protocol, simple auctions are coarse-grained. First, ads in different channels are not compared even if the client subscribes to multiple channels. Second, per-user information is used only to rank ads within one bin and not across bins. And third, the auction is volatile; this is inherent with first price auctions (where a bidder pays exactly what he bid) in a setting where bids can be updated and the outcome tested quickly. To illustrate: consider advertiser A bids \$2 and is ranked first, while advertiser B bids \$1 and is ranked second. From A's perspective, if he lowers his bid to \$1.01, he pays 99¢ less without changing the auction outcome. A can determine his most optimal bid by trial and error. At which point, B can determine by trial and error that by bidding only 2¢ higher, B gains a significant ranking advantage. This constant trial and error driven by real financial incentives results in volatile prices and a constantly changing ranking of ads, which interacts poorly with our goal of caching ads and auction results at the client.

3.2.2 *Design-II: Combined Auctions*

In the combined approach (Figure 5), the broker conducts the auction in a separate exchange. First, ads are sent to clients using pub-sub as originally described. The broker attaches a unique instance ID (Iid) to each copy of the ad published (not shown in figure). For each ad, the client computes a coarse score (U), typically between 1 and 5, as follows: for ads that match the user, the score reflects the quality of match with 5 signifying the best possible match. For ads that don't match the user, the score is a random number. To rank ads, the client sends (Iid, U) tuples for all ads in the client's database to the dealer. The dealer aggregates and mixes tuples for different clients before forwarding them to the broker. The broker ranks all the ads in the message. The ranking is based on both global and user metrics (e.g. bids, CTR, and client score). Note the ranked result contains all ads from the same client in the correct order, interspersed with ads for other clients (also in their correct order). The broker returns this ranked list to the dealer. The dealer uses the Iid to slice the list by client and forwards them to the clients. The client discards the ads that do not match the user, and stores the rest in ranked order.

The issue of ranking volatility is solved using second-price auctions. In second-price auctions, each bidder is charged the next highest bid. Thus the highest bidder pays the second-highest bid, second-highest bidder pays third-highest bid, and so on until the lowest bidder that pays some minimum bid (typically 1¢). The second-price outcome is identical to the steady state behavior of the first-price auction without the associated volatility. However, a straightforward application of second-price auctions at the broker does not work because the broker does not know which ads are from the same client, much less which ads will be discarded as they do not match the user.

Second-Price Auctions. To perform second-price auctions, the broker encrypts the bid information with a symmetric key (K) known only to the broker and sends it along with the ad. When a set of ads are chosen to be shown to the user, the client copies the encrypted bid

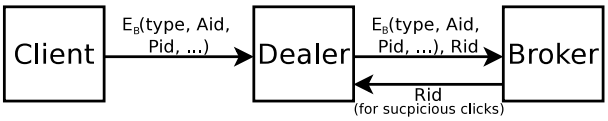

Figure 6: Message exchange for view/click reporting and blocking click-fraud. B is the public key of the broker. *Aid* identifies the ad. *Pid* identifies publisher website or application where the ad was shown. For second-price auctions, the opaque auction result is included. *Rid* uniquely identifies the report at the dealer.

information from ad $n+1$ to ad n. This encrypted bid information is sent as part of the click report, which the broker decrypts to determine what the advertiser should be charged. Second-price bid information is not sent for view reports for privacy reasons (Section 4.5).

3.3 View/Click Reporting

Ad views and clicks, as well as other ad-initiated user activity (purchase, registration, etc.) needs to be reported to the broker. The protocol for reporting ad events (Figure 6) is straightforward. The report containing the ad ID (*Aid*), publisher ID (*Pid*), and type of event (view, click, etc.) is encrypted with the broker's public-key and sent through the dealer to the broker. The dealer attaches a unique request ID (*Rid*) and stores a mapping between the request ID and the client, which it uses later to trace suspected click-fraud reports.

3.4 Click-Fraud Detection

Click-fraud consists primarily of users or bots clicking on ads for the purpose of attacking one or more parts of the system. It may be used to drive up a given advertiser's costs, or to drive up the revenue to a publisher. It can also be used to drive up the click-through-ratio of an advertiser so that that advertiser is more likely to win auctions.

Generally speaking, privacy makes click-fraud more challenging because clients are hidden from the broker. Privad overcomes this challenge through explicit privacy-preserving coordination between broker and dealer. Both the broker and dealer participate in detecting and blocking click-fraud; the dealer by measuring view and click volumes from clients, the broker by looking at overall click behaviors for advertisers and publishers. We discuss next the blocking mechanism and several detection mechanisms.

Blocking a fraudulent client once an attack is detected is straightforward. When a publisher or advertiser is under attack, the broker tells the dealer which report IDs are suspected as being involved in click-fraud. The dealer traces the report ID back to the client, and if the client is suspected of engaging in click-fraud more than some set threshold, subsequent reports from that client are blocked.

As with today's ad networks, there is no silver bullet for detecting click-fraud. And like ad networks today, the approach we take is *defense in depth* — a number of overlapping detection mechanisms (described below) operate in parallel; each detection mechanism can be fooled with some effort; but together, they raise the bar.

Per-User Thresholds. The dealer tracks the number of subscriptions, and the rates of view/click reports for each client (identified by their IP address). Clients that exceed thresholds set by the broker are flagged as suspicious. The broker may provide a list of NATed networks or public proxies used by multiple users so higher thresholds may apply to them.

Blacklist. Dealers flag clients on public blacklists, such as lists maintained by anti-virus vendors or network telescope operators that track IP addresses participating in a botnet. Dealers additionally share a blacklist of clients blocked at other dealers.

Honeyfarms. The broker operates honeyfarms that are vulnerable to botnet infection. Once infected, the broker can directly track which publishers or advertisers are under attack. When a report matching the attack signature is received, the broker asks the dealer to flag the originating client as suspicious.

Historical Statistics. The broker maintains a number of per-publisher and per-advertiser statistics including volume of view reports, and click-through rates. Any sudden increase in these statistics cause clients generating the reports to be flagged as suspicious.

Premium Clicks. Based on the insight behind [15], a user's purchase activity is used as an indication of honest behavior. Clicks from honest users command higher revenues. The broker informs the dealer which reports are purchases. The dealer flags the origin client as "premium" for some period of time, and attaches a single "premium bit" to subsequent reports from these clients.

Bait Ads. An approach we are actively investigating is something we term "bait ads", which can loosely be described as a cross between CAPTCHAs and the invisible-link approach to robot detection [19]. Basically, bait ads contain the targeting information of one ad, but the content (graphics, flash animation) of a completely different ad. For instance, a bait ad may advertise "dog collars" to "cat lovers". The broker expects a very small (but non-zero) number of such ads to be clicked by humans. A bot clicking on ads, however, would unwittingly trigger the bait. It is hard for a bot to detect bait, which, for image ads, amounts to solving semantic CAPTCHAs (e.g. [9]). Bait ads are published by the broker just like a normal ad. When a click report for a bait ad is received, the broker informs the dealer, which flags the client as potentially suspicious.

These mechanisms operate in concert as follows: per-user thresholds force the attacker to use a botnet. Honeyfarms help discover botnets, and blacklists limit the

amount of time individual bots are of use to the attacker. Historical statistics block high-intensity attacks, instead forcing the attacker to gradually mount the attack, which buys additional time for honeyfarms and blacklists to kick in before significant financial damage is caused. At the same time, bait ads disseminated proactively can detect low volume attacks due to the strong signal generated by a relatively small number of clicks, while disseminated reactively, bait ads can reduce false positives. And finally, premium ads, by forcing the attacker to spend money to acquire and maintain "premium" status for each bot, apply significant economic pressure, which is magnified by bots being blacklisted.

Overall these mechanisms have the effect of more-or-less putting Privad back on an even footing with current ad networks as far as click-fraud is concerned.

3.5 Reference Monitor

The reference monitor has six functions geared towards making it *difficult* (but not impossible) for the black-box client to leak private information. First, the reference monitor validates that all messages in and out of the client follow Privad protocols. To this end, the client is operated in a sandbox such that all network communication must go through the reference monitor in the clear (Figure 2). Second, the monitor is responsible for encrypting outbound messages from the client (and decrypting inbound messages). Third, the monitor is the source of almost all randomness in messages (e.g. session keys, randomized padding for encryption etc.). Section 4.4 discusses the single exception in the context of covert channels. Fourth, the monitor may additionally provide cover traffic or introduce noise to protect user privacy in certain Privad operations. Fifth, the monitor arbitrarily delays messages or adds jitter to disrupt certain timing attacks.

Technological means for disrupting covert channels is, of course, not enough since the client may attempt to leak information through semantic means. For instance, the client might send `lima-beams` when it really means `no-health-insurance`. The sixth and final function of the reference monitor is therefore to provide a hook to involve a human-in-the-loop. Privacy advocates can set up honeyfarm clients, train them with specific profiles, and monitor them for inconsistent message contents. Or, interested users can install reference monitors that occasionally check with the user that the client message makes semantic sense, and aggregates responses across many users to detect inconsistencies.

3.6 Optimization: Crypto Offload

Initially we were concerned that, even though view reports do not need to be processed in real-time, requiring public-key operations at the broker for each report would limit scalability. We solved the problem with an optimization that leverages idle clients to reduce broker overhead without compromising the privacy properties, and without any noticeable performance degradation for users. We briefly sketch our approach below. [4] contains a detailed description and performance analysis of this offload mechanism.

The key insights behind the optimization are first, that the broker in any event cannot violate privacy, and therefore privacy properties are not affected by having the broker offload message handling to some third-party. Second, clients already perform a public-key operation for each report, and so performing two public-key operations adds relatively little overhead.

The offload mechanism combines these two insights as follows: when sending a message the client encrypts its message as usual, but with the public key belonging to some random other client (which it was given by the broker). At the same time the client decrypts with its own private key some random other client's message (which it was sent by the broker). The broker can validate correct decryption by comparing checksums. This allows the broker to operate without having to perform any public-key crypto operations whatsoever. Keys are exchanged without a PKI, and the system is engineered to not degrade client performance, and to handle ungraceful client departures [4]. We have implemented and deployed this optimization as part of our live running system.

3.7 User Profiling

Even though the client is ultimately in charge of profiling the user, it can nevertheless leverage existing cloud-based crawlers and profilers through a privacy-preserving query mechanism. At a high level the query protocol is similar to the pub-sub protocol (Figure 3) operating as a single request-response pair; the request contains the website URL and the response contains profile attributes. Beyond this, the client can, of course, locally scrape and classify pages, incorporate social feedback, or even allow publisher websites to explicitly influence the profile. [3] details these profiling options. Overall, the user profiling options in Privad *adds to* existing cloud-based algorithms while preserving privacy, and therefore, we believe, has the potential to target ads better than existing systems.

3.8 Post-Click Behavior

Privad so far focuses primarily on privacy from the broker. However, privacy from the advertiser after the click is equally, if not more, important. For instance, if the advertiser targets its ads to, say, people with AIDS and no health insurance, the advertiser is well-positioned to take advantage of the user. Web proxies are an option, but then the proxy can glean user infor-

mation. We find the core Privad dealer/broker infrastructure is quite well positioned to anonymize clicks. At a high level, by chaining both dealer and broker as proxies, the dealer is unable to learn what advertisers a specific client goes to, and the broker is unable to learn which clients went to specific advertisers; this is, in essence, a 2-hop TOR [7] circuit without nearly as much performance degradation since the dealer/broker are managed professionally. At what point proxying is terminated and the client handed-off to the advertiser, is both a policy and engineering question and very much work-in-progress [3].

4. PRIVACY ANALYSIS

In this section we first define what we mean by user privacy. We then present our adversary model, and the deployment model and organizational incentives that justify our chosen adversary model. We address the issue of covert channels in this context. We then consider a series of attacks on the system, the defense to the attack, and a discussion of the extent to which the defense truly solves the attack. Finally, we look beyond user privacy to issues of concern to the advertiser and the broker.

4.1 Defining Privacy

Our privacy goals are based on Pfitzmann and Köhntopp's definition of anonymity [20] which is unlinkability of an *item of interest* (IOI) and some logical user identifier. Privad has three types of IOI; IP address, and interest attributes and demographic attributes. Pfitzmann and Köhntopp consider anonymity in terms of an *anonymity set*, which is the set of users that share the given item of interest — the larger this set, the "better" the anonymity. Personally Identifiable Information (PII) is information for which the anonymity set comprises a single (or a small number of) elements; e.g., the IP address is PII. Examples of non-PII anonymity sets in Privad include: the set of users that join a pub-sub channel, the set of users that visit a given publisher, and the set of users that view or click a given ad (i.e. probably share some or all of the ad's attributes).

In our definition of privacy we draw a distinction between IOI that contain PII and IOI that do not, as follows:

P1) Profile Anonymity: No single player is able to link any PII for a user with any attribute in the user's profile.

P2) Profile Unlinkability: No single player is able to link together more than a threshold number of (non-PII) profile attributes for the same user, which would otherwise allow them to, over time, construct a unique profile that could be deanonymized using external databases.

Existing ad networks, of course, satisfy neither Profile Anonymity, nor Profile Unlinkability.

Note that for Profile Unlinkability we use "number of profile attributes" rather than the size of the anonymity set even though the former doesn't per se map directly onto the latter. Different attributes imply different sizes of anonymity sets (e.g., `music` vs. `sports.skiing.cross-country`). Ideally, Privad would dynamically guarantee a minimum anonymity set size at runtime, but this is not possible because any measurement approach is easily attacked with a botnet of clients masquerading as members of that set. It is possible, however, to estimate the rough expected anonymity set size for a given attribute using outside semantic knowledge, and then use it to analyze the static privacy properties.

The approach towards privacy in Privad is then as follows: 1) static analysis ensures that no single message can violate Profile Unlinkability; this is enforced by the monitor as we discuss later in Attack 9. 2) Mechanisms in Privad ensure multiple messages from the same client cannot be linked together, and therefore the system as a whole cannot violate Profile Unlinkability. And 3) since the dealer is the only party that learns PII (IP address) and nothing else about the user, Profile Anonymity is trivially satisfied. In the remainder of this section we focus on attacks that attempt to compromise these provisions.

4.2 Trust Assumptions and Adversary Model

Broadly speaking, Privad defends against non-colluding *honest-but-curious organizations* with *malicious insiders* (defined below). The user trusts the reference monitor, and additionally trusts the dealer and broker to not collude. The advertiser and publisher, like today, trust the broker to perform accurate accounting.

Honest-but-curious organization (HBCO): An organization that acts according to its prescribed roles in the protocol when interacting with other players (i.e. when open to audit), but can attempt to *passively* break privacy based on information it gathers in the process.

Malicious insider: An individual in an HBCO that acts alone to attempt to *actively* break privacy; he may influence components operating under the exclusive control of the HBCO, but cannot, however, influence components operated, supervised, or audited by other Privad participants. Specifically, a malicious insider may inject, drop, or modify arbitrary protocol messages from within the HBCO, as well as enlist the help of third-parties not associated with Privad (e.g. a botnet).

These definitions are not arbitrary. They stem from beliefs about the nature of the organizations that operate the various components. Since our trust assumptions are at the very core of our design, it is critical that we state what these beliefs are and why. The following subsection does this.

4.3 Deployment Model and Organizational Incentives

4.3.1 Privacy Advocates

We define a privacy advocate broadly as an organization whose charter is to protect the privacy of users. Privacy advocates may be private or government. Today privacy advocates can have a strong impact on the advertising industry. They were able for instance to effectively shut down the new trial advertising services launched by NebuAd, Phorm and Facebook [5, 13].

Privacy advocates play several key roles in Privad. First, given that privacy advocates can kill technology deployments, especially early on, any organization trying to grow a Privad-based broker business would need at least the implicit support of privacy advocates. Second, the Privad client looks like adware: it is installed on user computers, and it delivers ads. Anti-virus companies routinely try to identify and disable adware on their customers computers. A Privad broker company would need to convince anti-virus companies that its client does no harm. Explicit support from privacy advocates would be key to obtaining this.

Third, there are a number of cases in Privad where a diligent external observer can detect an attack. While in practice this is often done by watchdog organizations or academic researchers, for the purposes of this paper we refer to these as privacy advocates as well. We expect privacy advocates to write the reference monitor, or at a minimum, validate its correctness. To this end, the reference monitor is designed to be extremely small and simple (see [4] for details) so that correctness can be verified manually. Another viable candidate for writing the reference monitor is an anti-virus company, as part of their product offering. In the context of this paper, we don't care who writes the monitor, as long as it is open source and open to validation.

Finally, we expect privacy advocates to oversee the operation of dealers. In our original design we expected privacy advocates to operate dealers. However, after discussions with a prominent privacy advocacy group and multiple brokers, we now believe dealers are better operated as a consortium of members that include privacy advocates.

4.3.2 Dealer Consortium

Deploying dealers on a scale necessary for global advertising is an expensive undertaking. Privacy advocates today have neither the funding nor the expertise to run dealers. We envision that oversight from privacy advocates would be funded through a levy placed on brokers. The actual technical operations would be sub-contracted to IT organizations and data centers. Since broker business depends on the effective operation of dealers, brokers would naturally demand some influence on how dealers are operated. It is therefore inevitable that members of broker, privacy advocate, and subcontractor IT organizations would find themselves working together. This necessary proximity unfortunately presents an opportunity for collusion.

Fortunately there are significant factors working against this opportunity being exploited. By far the most valuable asset to a privacy advocate is the trust placed in it by the public. If this trust is broken, i.e. by being caught in a collusionary relationship with a broker, then the privacy advocate is dead. Therefore there is a strong disincentive for privacy advocates to collude.

The risk to the privacy advocate of failing to detect collusion between the broker and the IT organization is unfortunately far less than the risk of being caught in a collusion itself. It is the difference between incompetence and malice. This could be mitigated by having multiple privacy advocates oversee the operation of the dealer, with its concomitant costs.

Finally, there is a possibility that collusion could be forced by legal authorities, for instance through subpoenas or wiretap warrants. Privacy advocates can verify the legality of such requests and take necessary action. That being said, the dealer is designed such that no information needs to be stored for an extended period of time (more than a few days). The chances of being compelled through legal means can therefore be reduced by aggressively pruning logs.

4.3.3 Broker Organization

Broker organizations provide both the broker and the client. While perhaps to a lesser extent than privacy advocates, brokers put high value in maintaining a reputation of trust. As examples, today Microsoft and Google, to name two, go to great lengths to not only portray themselves as trustworthy but also to live up to that reputation. As such, we believe that brokers would avoid collusion.

Nevertheless, brokers are in business to make money, and so may exploit opportunities to game the system that are handed to them. What's more, broker organizations may contain adversarial insiders who try to exploit information made available to them for personal gain. While individual adversarial insiders within an honest broker organization may act arbitrarily, if they were to affect externally visible elements they would be discovered quickly. Specifically, it would be hard for individual adversarial insiders to undetectably compromise the client software, but they may be in a position to skip internal procedures and access messages logs or inject malicious ads.

4.3.4 Advertisers and Publishers

Advertisers are a mixed bag, ranging from perfectly legitimate to highly adversarial. Indeed today phishing

attacks are carried out through dishonest advertising (see [24] for one example). We therefore characterize advertisers as being adversarial. The primary goal of the advertiser is to discover as much about the user as it can. This allows the advertiser to exploit this knowledge in any subsequent interaction with the user.

This illustrates a basic tension in the advertising system. On one hand, it is in everybody's interest that well-targeted advertising exists. Many useful services are supported through advertising which benefit users. On the other hand, taken too far, targeting erodes user privacy in fundamental ways even when the user's identity is protected. There needs to be a social or regulatory framework in place that puts limits on how detailed targeting can be, and what categories of targeting are off-limits. Within this framework, advertisers and brokers will always push for more targeting, and users and privacy advocates will push back. For the sake of this paper, we assume that this framework is in place, and any amount of targeting detail allowed to an advertiser is agreed upon within this framework.

Like advertisers, we assume that publishers may be unscrupulous. In general, Privad does not change the nature of user interactions with publishers. Users browse websites exactly as they do today. Publishers can, however, collude with other players to help them learn the IP address of the users.

Both advertisers and publishers are affected by click-fraud. They must today trust the broker to minimize click fraud and bill (or pay) them only for legitimate clicks. Brokers are incentivised to do so to reduce advertiser costs in a competitive market. Privad does not change any of this.

4.4 Covert Channels

Since the broker organization both writes the client and runs the broker, it can in principle create a covert channel between client and broker. An honest-but-curious broker, by definition, would not do this at an institutional level. An adversarial individual within a broker organization would have a hard time doing this. To see why, we first describe the constraints placed on any covert channel.

Note first of all that the covert channel must come from Privad application message fields, not encapsulating protocol fields such as those in the crypto messages. This is because it is the reference monitor that takes care of crypto and other message delivery functions. In addition, it is also the monitor that generates the one-time shared keys (for subscriptions) which otherwise represent the best covert channel opportunity.

Note next that the values of most message fields are driven by user behavior (outside client-control) and are subject to audit by privacy advocates or users. This includes the channel in subscriptions, and the type, publisher ID, and ad ID in reports, which together compose all remaining bits in subscribe and report messages.

The next best opportunity for a covert channel comes from the user score in the auction message (Figure 5). That is because this is the only client-controlled message field. Furthermore, this field has a random component, albeit within a small range since the user score need only be 2 or 3 bits in size.

Lastly, the covert channel must either be contained in a single message, or employ complex coding tricks. This is because the monitor adds arbitrary delay or jitter to messages to disrupt time-based correlation. Added complexity (i.e., more bits) raises the risk of detection.

It would be hard for an individual in the broker organization to generate the covert channel without getting caught. He would have to write code in the client that overrides some operation. He would have to write code in the broker that detects the sequence of messages that provides a covert signal, and transmits this signal back to himself. Even if the attacker is lucky enough to have access to the necessary code files, there are many opportunities in the software development process for the code or its behavior to be detected. In short, while the covert channel is possible, it is hard to imagine that it could be pulled off by any single individual.

Note this is very different from today where an insider does not need to modify client or server code, or modify or add messages on the wire, and need only passively capture traffic or access log files (done easily [23]) to violate user privacy.

4.5 Attacks and Defenses

We next consider some key attacks and their defenses. Additional attacks are considered in [3].

4.5.1 Attacker at Client

Attack A1: The attacker installs malware on a user's computer which provides the profile information to the attacker or otherwise exploits it.

Defense D1: Privad does not protect against malware reading the profile it generates. Our general stance is that even without Privad, malware today can learn anything the client is able to learn, and so not protecting against this threat does not qualitatively change anything. Having said that, obviously the existence of the profile does make the job of malware easier. It saves the malware from having to write its own profiling mechanisms. It may also allow the malware to learn the profile more quickly since it doesn't have to monitor the user over time to build up the profile.

Ultimately what goes into the profile is a policy question that privacy advocates and society need to answer. Clearly information like credit card number, passwords, and the like have no place in the profile (though malware can of course get at this information anyway).

Whether a user has AIDS probably also does not belong there. Whether a user is interested in AIDS medication, however, arguably may belong in the profile.

Indeed, there are pros and cons to keeping profile contents open. On the pro side, this makes it easier for privacy advocates to monitor the client and to an extent broker operation. On the con side, it makes life easier for malware. One option, if the operating system supports it, is to make the profile available only to the client process (e.g. through SELinux [18]). This would protect against userspace malware, but not rootkits that compromise the OS. Another option is to leverage trusted hardware (e.g. [22]) when available. How best to handle the profile from this perspective is both an ongoing research question and a policy question.

4.5.2 Attacker at Dealer

A2: The attacker attempts to learn user profile information by reading messages at the dealer.

D2: The dealer proxies five kinds of messages: subscribe, publish, auction request and response, and reports. Of these, the dealer cannot inspect the contents of subscribe, report, and publish messages since the first two are encrypted with the broker's public key, and the last is encrypted with a symmetric key that is exchanged via the encrypted subscribe message. Auction messages, which are unencrypted, contain a random single-use Iid that identifies the ad at the broker and the client (exchanged over the encrypted publish message), but is meaningless to the dealer; the broker must be careful to choose Iids randomly so different Iids cannot be linked to the same ad.

A3: The attacker injects messages at the dealer in order to learn a user's profile information.

D3: The dealer cannot inject a fake publish message since it would not validate at the client after decryption. If the dealer injects a fake subscribe message, all resulting publish messages would be discarded by the client since the client would not have a record of the subscribe or the associated key. The dealer cannot inject fake auction messages since the client would not have a record of the Iid. The dealer could reorder the auction result, but would not learn which ad the client viewed or clicked since reports are encrypted. The dealer injecting fake reports has no impact on the client; it is, however, identical to dealer-assisted click-fraud, which we consider next.

A4: The dealer itself engages in click-fraud, or otherwise does not comply with the broker's request to block fraudulent clients.

D4: The broker can independently audit that the dealer is operating as expected both actively and passively. The broker can passively track view/click volumes, and historical statistics on a per-dealer basis to identify anomalous dealers. Additionally the broker can passively monitor the rate of fraudulent clicks (e.g. using bait ads) on a per-dealer basis. The broker can detect suspicious dealer behavior if after directing dealers to stem a particular attack the rate of fraudulent clicks through one dealer does not drop (or drops proportionally less) than for other dealers. Finally, the broker can actively test a dealer by launching a fake click-fraud attack from fake clients, and ensuring the dealer blocks them as directed.

A5: A particularly sneaky attack aimed at learning which users send view or click reports for a given publisher (or advertiser) is as follows. The dealer first launches a click-fraud attack on the given publisher (or advertiser). The broker identifies the attack. When a user sends a legitimate report for that publisher (or advertiser), the broker mistakenly suspects the report as fraudulent and asks the dealer to block the client. The dealer can now infer that the encrypted report it proxied must have matched the attack signature it helped create.

D5: First node that this attack applies only in the scenario where there are no other click-fraud attack taking place other than the one controlled by the dealer (and the dealer somehow knows this). As part of the Privad protocol (Figure 6), however, the dealer does not learn how many attacks are taking place (even if there is only one ongoing attack), or which publishers or advertisers are under attack, or which attack the client was implicated in. Thus there is too much noise for the dealer to reach any conclusions about implicated clients.

4.5.3 Attacker at Broker

A6: The broker, or an insider, attempts to link multiple messages from the same user using passive or active approaches.

D6: As long as the dealer and broker are not colluding, multiple messages must have some correlation for the broker to link them. We are only concerned with subscribe and reports messages since the dealer mixes auction requests.

Privad messages do not contain any PII, unique identifier, or sequence number. The monitor ensures the per-subscription symmetric keys are unique and random. Additionally, the monitor disrupts timing based correlation, for instance by staggering bursts of messages (e.g. when the client starts up, or views a website with many adboxes). Towards this end Privad protocol messages are designed to be asynchronous and not require end-to-end acknowledgments. Altogether these defenses prevent the broker from linking two subscriptions, or two reports from the same user.

The broker may attempt to link a report with a subscription. The only way to do this is by publishing an ad with a unique ad ID, and waiting for a report

with that ID. Privacy advocates can detect if the broker engages in such practice by running honeyfarms of identical clients and ensuring ad IDs are repeated. [3] discusses another solution to that uses a second dealer between the broker and the first.

A7: During the combined auction mechanism the broker attempts to link two ads published to the same client through different pub-sub subscriptions, thereby effectively linking two subscriptions.

D7: The property of the mix constructed at the dealer is such that tuples from the same client but for ads on different pub-sub channels are indistinguishable from tuples from two different clients each subscribed to one or the other pub-sub channels. The pub-sub protocol provides the same property. Thus the broker doesn't learn anything new from the auction protocol.

Note the broker can obviously link which ads it sent for the same subscription, but cannot determine which of them actually matched the user. This is because the client submits all ads received on a channel for auction whether or not it matched the user (enforced by the monitor); bogus user scores for non-matching ads prevents the broker from distinguishing between the two.

A8: The broker masquerades as a dealer and hijacks the client's messages thus learning the client's IP address. Possible methods of hijacking the traffic may include subverting DNS or BGP.

D8: The solution is to require Transport Layer Security (TLS) between client and dealer, and to use a trusted certificate authority. The reference monitor can insure that this is done correctly.

A9: The broker creates a channel with a large enough number of attributes that an individual user is uniquely defined. When that user joins the channel, the broker knows that a user with those attributes exists. This could be done for instance to discover the whereabouts of a known person. It could also be used to discover additional attributes of a known person. For instance, if n attributes are known to uniquely define the person, then any additional attributes associated with a joined channel can be discovered.

D9: It is precisely for this reason that pub-sub channels definitions are static, well-known, and public (Section 3.1). Privacy advocates can look at channel definitions and ensure they meet a minimum expected anonymity set size. Additionally, the monitor can filter out channel definitions when the attributes for that channel exceed some set threshold.

Similar restrictions apply to the set of profile attributes an ad can target, with one difference. In the context of second-price auctions, the broker needs to necessarily link adjacent ads. Thus the monitor needs to enforce that the sum of attributes of the two ads involved in a click-report is below the threshold.

Note the ability to link two ads applies only to clicks. View reports do not contain second price information since otherwise a page with many ads would allow the broker to link each consecutive pair of ads, and therefore a whole chain of ads. While the same problem exists if the user were to click on the whole chain of ads, since clicks are rare this is not a big concern.

4.6 Advertiser Privacy

Up to now this document has concerned itself with user privacy. The advertiser, however, also has privacy concerns, which we discuss here. Advertisers would like to keep details about their advertising campaigns private. These include ad targeting information (interest categories, keywords, or demographics), the amount it bids for ads, as well as its overall advertising budget.

With current advertising systems it is possible to learn at least some of an advertiser's targeting information. To do so, the recipient can make a hypothesis as to what keywords the advertiser is targeting, then try some searches to see if the hypothesis is correct. To the extent that online advertisers today target demographics, it is somewhat possible to determine what those demographics are by "training" a browser to match a certain demographic, and then attracting ads as above.

With Privad, the process of learning an advertiser's targeting information is similar, though significantly easier. The recipient would make a hypothesis as to what interest categories are being targeted. In many though not all cases, this would be quite obvious because interest categories are aligned with products and services. The recipient then joins the appropriate interest channels. The ads received for the advertiser will have the targeted demographics attached.

With current advertising systems, it is hard if not impossible to learn how much an advertiser bids for certain keywords. Even though the order of ads is public, and one can compete with the advertiser on these keywords and see what price beats the advertiser, one doesn't know what the advertiser's CTR is (which modifies their auction rank) and therefore doesn't know what the advertiser bid to get that rank. Privad does not change this for either auction strategy.

It is hard to determine the overall budget an advertiser has with current systems, and Privad also does not change this.

4.7 Broker Privacy

Finally, the broker also has some privacy concerns, mainly in the form of intellectual property protection of its profiling mechanisms. Desktop software companies like Microsoft, Apple, and Adobe, to name three, have figured out that intellectual property is best protected through legal mechanisms (patents, copyrights, and trade secrets), and to a lesser degree with technological hurdles (hardware support, obfuscated binaries).

Privad doesn't change any of this.

5. IMPLEMENTATION AND PILOT DEPLOYMENT

We have implemented the full Privad system and deployed it on a small scale. The system comprises a client implemented as a 154KB addon for the Firefox web browser, a dealer, and a broker. We have deployed Privad with a small group of users comprised primarily of friends and family, and around 250 volunteers we recruited using Amazon's Mechanical Turk service [1]. The primary purpose of the deployment is to convince ourselves that Privad represents a complete system. To this end the system performs all aspects of Privad including user profiling, ad dissemination, auctions, view/click reporting, crypto optimizations, and basic click-fraud defense; we re-publish Google ads through the system for testing purposes. [4] describes in detail our implementation, deployment, and microbenchmark-based experimental evaluation.

6. RELATED WORK

There is surprising little past work on the design of private advertising systems, and what work there is tends to focus on isolated problems rather than a complete system like Privad. This related work section focuses only on systems that target private advertising per se, and mainly concentrates on the privacy aspects of those systems; [4] contains a broader survey of related work. In particular, we look at Juels [14], Adnostic [21], Nurikabe [17] and Freudiger et. al [10].

Juels by far predates the other work cited here, and indeed is contemporary with the first examples of the modern advertising model (i.e. keyword-based bidding). As such, Juels focuses on the private distribution of ads and does not consider other aspects such as view-and-click reporting or auctions. Privad's dissemination model is similar to Juels' in that a client requests relevant ads which are then delivered. Indeed, Juels' trust model is stronger than Privad's. Juels proposes a full mixnet between client and broker, thus effectively overcoming collusion. We believe that Juels' trust model is overkill, and that his system pays for this both in terms of efficiency and in the mixnet's inability to aid the broker in click fraud.

Like Juels and Privad, Adnostic also proposes client-side software that profiles and protects user privacy. When a user visits a webpage containing an adbox, the URL of the webpage is sent to the broker as is done today. The broker selects a group of ads that fit well with the ad page (they recommend 30), and sends all of them to the client. The client then selects the most appropriate ad to show the user. The novel aspect of Adnostic is how to report which ad was viewed without revealing this to the broker. Adnostic uses additively homomorphic encryption and efficient zero-knowledge proofs to allow the broker to reliably add up the number of views for each ad without knowing the results (which remain encrypted). Instead, they send the results to a trusted third-party which decrypts them and returns the totals. By contrast to views, Adnostic treats clicks the same as current ad networks: the client reports clicks directly to the broker.

The privacy model proposed by Adnostic is much weaker than that of Privad. Privad considers users' web browsing behavior and click behavior to be private, Adnostic does not. Indeed, we would argue that the knowledge that Adnostic provides to the broker allows it to very effectively profile the user. A user's web browsing behavior says a lot about the user interests and many demographics. Knowledge of which ads a user has clicked on, and the demographics to which that ad was targeted, allow the broker to even more effectively profile the user. Finally, the user's IP address provides location demographics and effectively allows the broker to identify the user. Adnostic's trust model for the broker is basically honest-and-*not*-curious. If that is the case, then today's advertising model should be just fine.

Nurikabe also proposes client-side software that profiles the user and keeps the profile secret. With Nurikabe, the full set of ads are downloaded into the client. The client shows ads to the user as appropriate. Before clicking any ads, the client requests a small number of click tokens from the broker. These tokens contain a blind signature, thus allowing the tokens to be later validated at the broker without the broker knowing who it previously gave the token to. The user clicks on an ad, the click report is sent to the advertiser along with the token. The advertiser sends the token to the broker, who validates it, and this validation is returned to the client via the advertiser.

Nurikabe has an interesting privacy model. They argue that, since the advertiser anyway is going to see the click, there is no loss of privacy by having the advertiser proxy the click token. By taking this position, Nurikabe avoids the need for a separate dealer. Our problem with this approach is that Nurikabe basically gives up on the problem of privacy from the advertiser altogether. It cannot report views without exposing this to the advertiser, thus reducing user privacy from the advertiser even more than today. View reporting is important, in part because it allows the advertiser to compute the CTR and know how well its ad campaign is going. Nurikabe also gives up any visibility into click fraud. Nurikabe mitigates click fraud only by rate limiting the tokens it gives to every user. As a result, the attacker need only Sybil itself behind a botnet and solve CAPTCHAs to launch a massive click-fraud attack which cannot be defended. Finally, in [12] the authors find through ad measurements that there are

simply far too many ads (with too much churn) to be able to distribute them all to all clients.

The overall goal of Freudiger is quite different from that of the other systems. Freudiger proposes to give the user control over which web browsing activity is reported and which is not by allowing the user to determine when 3rd-party cookies are and are not reported. The idea here is to strike a balance between the user's privacy needs and the advertiser's targeting needs (and the user's desire to have targeted ads served to him or her). By contrast, we believe that both good targeting and complete privacy can be achieved.

7. SUMMARY AND FUTURE DIRECTIONS

This paper describes a practical private advertising system, Privad, which attempts to provide substantially better privacy while still fitting into today's advertising business model. We have designs and detailed privacy analysis for all major components to such a system: ad delivery and reporting, click fraud defense, advertiser auctions, user profiling, post-click behavior, and optimizations for scalability.

It would be easy at this point to conclude that Privad significantly improves the privacy landscape and be done. In the end, however, it is not up to us to decide if Privad is private enough. This can only be done by society at large. Towards this end, we have started dialogs with a number of privacy advocates and policy makers, and have submitted the first of what are expected to be many public opinions (this one to the FTC privacy roundtable, jointly authored with the authors of Adnostic [2]).

Besides this, we need a better understanding of a number of Privad components. Foremost among these are how best to do profiling, the bait approach to click-fraud, and privacy from the advertiser. We are actively working on all of these problems. We are also working with application developers and ad agencies to scale our deployment to Internet scale to give researchers a platform for conducting experiments with real users and advertisements at scale.

Finally, we hope that Privad and other recently proposed private advertising systems spurs a rich debate among researchers as to the best ways to do private advertising, the pros and cons of the various systems, and how best to move private advertising forward in society.

8. REFERENCES

[1] Amazon Mechanical Turk. `http://www.mturk.com`.

[2] Blinded. Privacy Roundtables – Comment, Project No. P095416, Comment No. 544506-00017. `URL blinded`.

[3] Blinded. Privacy Analysis of the Privad Privacy-preserving Advertising System. Technical Report TR-2010-2, 2010. `http://tinyurl.com/yhqlw3t` (3rd-party file host).

[4] Blinded. Privad: Practical Privacy in Online Advertising. Technical Report TR-2010-1, 2010. `http://tinyurl.com/ycbjllc` (3rd-party file host).

[5] J. Cheng. Facebook Beacon shines for last time as part of settlement. `http://tinyurl.com/kowkfg`, Sept. 2009.

[6] J. Chester, S. Grant, J. Kelsey, J. Simpson, L. Tien, M. Ngo, B. Givens, E. Hendricks, A. Fazlullah, and P. Dixon. Letter to the House Committee on Energy and Commerce. `http://tinyurl.com/y85h98g`, Sept. 2009.

[7] R. Dingledine, N. Mathewson, and P. Syverson. TOR: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium (Security '04)*, San Deigo, CA, Aug. 2004.

[8] B. Edelman, M. Benjamin, and M. Schwarz. Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. *American Economic Review*, 97(1):242–259, Mar. 2007.

[9] J. Elson, J. R. Douceur, J. Howell, and J. Saul. Asirra: A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*, Alexandria, VA, Oct. 2007.

[10] J. Freudiger, N. Vratonjic, and J.-P. Hubaux. Towards Privacy-Friendly Online Advertising. In *Proceedings of the W2SP 2009 Workshop*, Oakland, CA, May 2009.

[11] G. Gross. FTC Sticks With Online Advertising Self-regulation. *IDG News Service*, Feb. 2009.

[12] S. Guha, A. Reznichenko, K. Tang, H. Haddadi, and P. Francis. Serving Ads from `localhost` for Performance, Privacy, and Profit. In *Proceedings of the 8th Workshop on Hot Topics in Networks (HotNets '09)*, New York, NY, Oct. 2009.

[13] A. Jesdanun. Ad Targeting Based on ISP Tracking Now in Doubt. *Associated Press*, Sept. 2008.

[14] A. Juels. Targeted Advertising ... And Privacy Too. In *Proceedings of the 2001 Conference on Topics in Cryptology*, pages 408–424, London, UK, 2001. Springer-Verlag.

[15] A. Juels, S. Stamm, and M. Jakobsson. Combating Click Fraud via Premium Clicks. In *Proceedings of 16th USENIX Security Symposium (Security '07)*, pages 1–10, Boston, MA, 2007.

[16] B. Krishnamurthy and C. E. Wills. Cat and Mouse: Content Delivery Tradeoffs in Web Access. In *Proceedings of the 15th international conference on World Wide Web (WWW '06)*, Edinburgh, Scotland, 2006.

[17] D. Levin, B. Bhattacharjee, J. R. Douceur, J. R. Lorch, J. Mickens, and T. Moscibroda. Nurikabe: Private yet Accountable Targeted Advertising. Under submission. Contact `johndo@microsoft.com` for copy, 2009.

[18] P. Loscocco and S. Smalley. Integrating Flexible Support for Security Policies into the Linux Operating System. In *Proceedings of the 2001 USENIX Annual Technical Conference*, Boston, MA, June 2001.

[19] K. Park, V. S. Pai, K.-W. Lee, and S. Calo. Securing Web Service by Automatic Robot Detection. In *Proceedings of the 2006 USENIX Annual Technical Conference*, Boston, MA, 2006.

[20] A. Pfitzmann and M. Köhntopp. Anonymity, Unobservability, and Pseudonymity — A Proposal for Terminology. *Designing Privacy Enhancing Technologies*, 2001.

[21] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas. Adnostic: Privacy Preserving Targeted Advertising. In *Proceedings of NDSS 2010*, San Diego, CA, Feb. 2010. To appear.

[22] Trusted Computing Group. TPM Specification Version 1.2. `http://www.trustedcomputinggroup.org/`.

[23] P. Wong. Conversations About the Internet #5: Anonymous Facebook Employee. `http://tinyurl.com/yaxu5j5`, Jan. 2010.

[24] D. Yu. How To Spam Facebook Like A Pro: An Insider's Confession. `http://tinyurl.com/yg52kyn`, Nov. 2009.