Privacy Analysis of the Privad* Privacy-preserving Advertising System

***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, and social networks. The centralized nature of these online services, however, results in substantial privacy leakage for users. This paper analyzes the privacy and click-fraud defense properties of Privad, a practical privacy-preserving online advertising system. Privad preserves privacy by maintaining user profiles on the user's computer instead of in the cloud, minimizing information released to the ad network, and tightly controlling what various participants may learn. An anonymizing proxy hides the network address of the client, while encryption prevents the proxy from viewing client messages. The security analysis presented in this paper covers all aspects of a practical and deployable system, including profiling, ad dissemination, auctions, click fraud, view and click reporting, and click anonymization. We argue 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.**

I. Introduction

Online advertising is a key economic driver in the Internet economy. It funds services provided by such industry giants as Google and Yahoo!, and helps pay for data centers and, indirectly, ISPs. Internet advertisers increasingly work to provide more personalized and therefore better targeted advertising. Unfortunately, personalized online advertising, at least so far, has come at the price of individual privacy. In order to deliver ads that the individual is interested in, the online service must first learn the individual's interests and demographics.

Privacy advocates would like to put an end to advertising models that violate privacy. In some cases, for instance in the highly publicized battles with the ad broker companies Phorm and NebuAd [9], the privacy advocates have had some success. They have however had much less success with the more entrenched ad brokers like Google and Yahoo!; in 2004, lawmakers stopped short of preventing Google from scanning emails for advertising purposes [8]. Arguably the reason why privacy advocates have failed is that they offer no viable alternatives. The deal they offer, privacy *or* advertising, is not acceptable to industry, governments, or probably most individuals. As a result, current government guidelines on privacy protection related to advertising [4] focus not on preventing ad networks from collecting user data, but rather on how they treat that data once they have it.

Privad is a practical privacy-preserving online advertising system that aims to supply the alternative that privacy advocates are looking for. It promises that privacy and targeting do not have to be mutually exclusive. Privad preserves privacy by maintaining user profiles on the user's computer instead of in the cloud. A small amount of information is necessarily leaked to the ad network: the classes of ads a user is interested in, the ads the user has viewed or clicked on and the websites that carried the ads, and the ranking of ads for auction purposes. This information, however, is handled in a way that no party can link it back to the user. An anonymizing proxy hides the user's network address, while encryption prevents the proxy from learning any user information.

Privad is first and foremost meant to be a *practical* alternative to existing systems. This means that, among other things, it must scale to global proportions, and must fit reasonably well into the advertising business model that has emerged over the past few years. It must also support all aspects of the advertising business, including profiling, ad dissemination, auctions, click fraud, view and click reporting, and advertiser anonymization. In earlier work [2] we addressed the practicality of Privad, especially its scalability but also its applicability to the advertising business model. That work answered the question *"Can Privad be deployed?"*

This paper contributes a security analysis of Privad, including both privacy and click-fraud defense. This analysis gives people and organizations the information they need to answer the question *"Do we want Privad to be deployed?"* This is important for privacy advocates (i.e. the EFF[1]) and policy makers (i.e. the FTC[2]) to determine whether they want to support or oppose Privad. This is also important for users, so that they are able to choose whether or not to opt-in to Privad. Finally, it provides a basis for academics and security experts to understand the security properties of Privad, and to compare it with existing [17] and future systems.

A. Outline

Section II presents a technical overview of the Privad architecture: the components and their roles in the system. Section III lays the groundwork for the security analysis. It states the privacy goals of the system. While precise, the goals are not formally specified and as such the arguments for how Privad supports the goals are informal. Section III also describes the kind of trust placed in each of the organizations that deploy Privad components, and justifies that trust. Sections IV through VIII give the security analysis

*Not to be confused with Adnostic (formerly known as Privads) [17]

[1]Electronic Frontier Foundation

[2]Federal Trade Commission (USA)

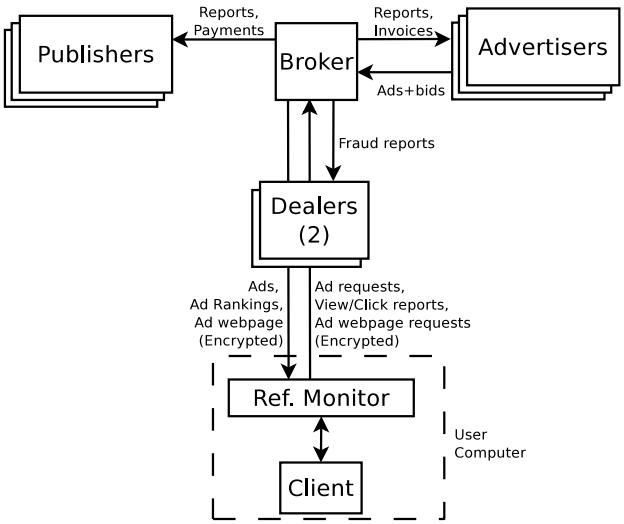

Fig. 1: Privad System Overview

of each of the major system functions: user profiling, ad dissemination, ad auctions, ad event (view, click) reporting, and click anonymization. In each of these sections, a precise description of the protocol is given, followed by a privacy analysis of that protocol. Each analysis consists of a series of attacks on the system, the defense of the attack, and a discussion of the extent to which the defense truly solves the attack. Some privacy attacks involve multiple functions. The privacy analysis of these attacks is deferred until all related functions have been described. In this way, the overall privacy analysis of the system builds from relatively more basic to more comprehensive. Section IX describes the click-fraud defense mechanisms of Privad, and well as an analysis of the privacy attacks these mechanisms open up. Section XI looks at related work.

II. 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.

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. (The need for two dealers is explained in later sections.) Finally, a thin trusted reference monitor between the client and the network ensures that the client conforms to the Privad protocol. 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.

At a high level, the operation of Privad goes as follows. The client 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 subscribing to an interest category combined with a few broad non-sensitive demographics (region, gender, language). 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 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 page the ad was shown. Individual reports cannot be linked 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.

When the user clicks on an ad, the advertiser's webpage is initially proxied to protect user anonymity. Eventually, however, the user may leak his identity in some manner, for instance by revealing his credit-card number or shipping address. Future work will determine the right scalability-privacy tradeoff for when proxying is terminated and the user connected directly to the advertiser.

Unscrupulous users or 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 IX). 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.

The reference monitor has five functions that we summarize here (see [2] for details). 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. 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 X-A 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. Finally, the monitor arbitrarily delays messages or adds jitter to disrupt certain timing attacks.

III. Preliminaries

In this section we first present the design goals behind Privad. We then present the incentives and deployment model of the various players, on which we base our adversary model and trust assumptions.

A. Privacy Goals

Our privacy goals are based on Pfitzmann and Köhntopp's definition of anonymity [16] which is unlinkability of an *item of interest* (IOI) and a user identifier, where the user identifier is not necessarily some Personally Identifiable Information (PII), but rather simply a locally-defined identifier to mean "some user". Privad has three types of IOI; IP address, and interest and demographic attributes. Of these, the IP address is PII. Demographics can in theory be PII (i.e. a demographic defined as "CEO of Acme Inc."), and so Privad must be operated in such a way that PII-based demographics are never defined.

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. Examples of anonymity sets in Privad include: the set of users that may browse a given URL, the set of users that join an ad pub/sub channel, and the set of users that may view a given ad (i.e. share some or all of the ad's attributes).

In our privacy goals, we draw a distinction between IOI that contain personally identifiable information (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.

Note that the number of profile attributes per se doesn't directly map into the size of the anonymity set. Different attributes imply a different number of users (`gender.female` versus `sports.tennis`). In Privad, we are not able to accurately measure the size of any given anonymity set because any measurement approach is easily attacked with a botnet of clients masquerading as members of that set. Nevertheless, it is very feasible to, for instance, annotate attributes with a rough indication of their expected anonymity set size, and use that annotation to finetune the threshold number. While such approaches are for further study, our definition of profile unlinkability (P2) is meant to include such approaches.

It should be noted that these privacy goals must be executed within a system that is practically deployable. For us, this means every system function (profiling, auctions, etc.) should perform at least as well as it performs in today's ad networks, and that the system should scale at least as well as today's ad networks. This is shown to be the case in [2].

B. Trust Assumptions and Adversary Model

We make the following trust assumptions:

T1) *User:* The user trusts the reference monitor. Additionally, the user trusts the dealer and broker to not collude. Finally, the user trusts that malicious individuals in the broker organization cannot undetectably create a covert channel between client and broker (see Section X-A).

T2) *Advertiser and Publisher:* The advertiser and publisher trust the broker to perform accurate accounting.

Broadly speaking, Privad defends against non-colluding *honest-but-curious organizations* with *malicious individuals*. We informally define our adversary model as follows.

V1) *Honest-but-curious organization (HBCO):* The organization acts according to its prescribed roles in the protocol when interacting with other players, but can attempt to *passively* break privacy based on local information it gathers in the process.

V2) *Malicious individual:* 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 or supervised by other Privad participants. Specifically, a malicious individual 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.

C. Deployment Model and Organizational Incentives

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 [9].

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 [2] 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.

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 subcontracted 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.

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) Advertisers: Advertisers are a mixed bag, ranging from perfectly legitimate to highly adversarial. Indeed today phishing attacks are carried out through dishonest advertising (see [19] 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. The fact that the dealer and broker proxy the post-click user/advertiser interaction does not mean that knowledge of user characteristics (interests and demographics) cannot be exploited by the advertiser. 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.

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.

In any event, while Privad can protect the user's IP address from the advertiser post-click, Privad does nothing to prevent a user from voluntarily giving up personally identifying information to the advertiser.

5) Publishers: 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.

6) Click Fraudsters: Finally, Privad may be attacked by click fraudsters. 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.

Click fraud is typically defined as a click with a strictly zero probability of it resulting in a sale. Since advertisers (or publishers) cannot conclusively prove the intent behind a click, they must 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.

D. Notation

We use the following notation in the remainder of the paper.

- *Participants:* Client x ($\mathbb{C}_x$), Dealer i ($\mathbb{D}_i$), Broker ($\mathbb{B}$)
- *Operations:*
  - $p \Rightarrow q$ **send** x
    Sending a message containing x from p to q.
  - $@p$ ***action*** x
    Perform *action* at p (e.g. store) with data x.
- *Cryptographic Primitives:*
  - $\mathcal{E}^{\mathcal{P}}_K(x), \mathcal{E}^{\mathcal{S}}_K(x)$
    An encryption of message x under the key K. $\mathcal{E}^{\mathcal{P}}$ is a public-key encryption; K is this case is the public key. $\mathcal{E}^{\mathcal{S}}$ is a symmetric-key encryption; K in this case is the shared symmetric key.
- *Common Inputs:*
  - B, b: Broker's public/private keypair where B is public, and b is private to the broker.
  - xid: Unique transaction ID chosen by the dealer for a given exchange

IV. User Profiling

In this section we present and analyze a set of mechanisms that illustrate it is possible to profile the user better than brokers can today while preserving user privacy.

A. Mechanisms

A user's profile is a set of attributes (e.g. `sports.tennis`, `gender.male`) that describe the user's demographics and interests. As mentioned, the profile is constructed by the client software by monitoring user activity. There are four basic approaches to profiling: crawling, scraping, metadata, and social feedback. We discuss each below.

Input at $\mathbb{C}$*:* Url — URL of website visited by user
Input at $\mathbb{C}$*:* τ — Number of attributes requested
Input at $\mathbb{C}$*:* K — Single-use symmetric key
Input at $\mathbb{B}$*:* $\{\mathsf{P}_i\}$ — Profile attributes associated with Url
Input at $\mathbb{C}$*:* UProf — User profile

protocol ProfileRequest
  $\mathbb{C} \Rightarrow \mathbb{D}$ **send** $\mathcal{U} = \mathcal{E}^{\mathcal{P}}_B(\langle \mathsf{Url}, \tau, K \rangle)$
  $@\mathbb{D}$ **store** $\langle xid, \mathbb{C} \rangle$
  $\mathbb{D} \Rightarrow \mathbb{B}$ **send** $\langle xid, \mathcal{U} \rangle$
  $@\mathbb{B}$ **recover** $\langle \mathsf{Url}, \tau, K \rangle$ **from** $\mathcal{U}$
protocol ProfileResponse
  $\mathbb{B} \Rightarrow \mathbb{D}$ **send** $\langle xid, \mathcal{P} = \mathcal{E}^{\mathcal{S}}_K(\mathsf{P}_{1\ldots\tau}) \rangle$
  $@\mathbb{D}$ **recover** $\mathbb{C}$ **using** xid
  $\mathbb{D} \Rightarrow \mathbb{C}$ **send** $\mathcal{P}$
  $@\mathbb{C}$ **recover** $\mathsf{P}_{1\ldots\tau}$ **from** $\mathcal{P}$
function UpdateProfile($\mathsf{P}_{1\ldots\tau}$)
  for all P_i **in** $\mathsf{P}_{1\ldots\tau}$ **do**
    if *encountered* P_i *enough times* **then**
      $@\mathbb{C}$ **update** $\mathsf{UProf} \leftarrow \mathsf{UProf} \cup \mathsf{P}_i$

Protocol 1: Privacy-preserving User Profiling: Crawling Mechanism

1) Crawling: The simplest approach to profiling users is for the broker to crawl the web and pre-classify websites. This is closest to the approach taken today. The client anonymously queries the attributes associated with a webpage visited by the user. A straightforward approach is listed in Protocol 1. The client encrypts the webpage URL and a single-use symmetric key chosen by the client with the broker's public key and sends it to the broker through the dealer. The broker encrypts the associated attributes with the symmetric key, and sends the response along the reverse path. The client updates the user's profile with attributes that are encountered on several websites visited by the user or on websites the user visits often.

One advantage of crawling is being able to use complex algorithms and large datasets to classify webpages at the broker. The disadvantage, however, is that crawling doesn't work for webpages that require the user to log in, or for desktop applications. This limitation applies to existing ad networks as well.

2) Scraping: The client software scrapes information from webpages visited by the user and from desktop applications. This is easily done for websites (and applications) that present structured information that maps directly to predefined attributes. Examples include online social networking sites, shopping, and travel sites, the user's local audio and video library, etc. We envision the client will have a modular architecture with website and application specific plugins that will be written and kept up-to-date by the broker.

In contrast to crawling, scraping works with websites that require the user to log in. However, mapping less structured content (e.g. blogs, search terms, word documents) to predefined attributes on the client is hard because of practical limits on the complexity of the client. One can, however, imagine a dictionary or a small natural-language model mapping text

fragments to attributes in the client being feasible. A more comprehensive model would require external classifiers in the cloud. Protocol 1 with URL replaced by the scraped text fragment can be used for this purpose. This produces a risk of revealing PII in the text fragment, and so we do not further consider cloud-based classification of scraped data.

3) Metadata: Third, websites can directly embed profile attributes as metadata in the webpage, which the Privad client can use directly. Local applications can directly communicate profile attributes to the client. The broker would incentivise this by offering a portion of the ad revenue to the website or application providing profile information (separate from publisher that provided ad space). To this end, the client would keep track of which sources contributed profile information that ultimately led to a click, and report it as part of the anonymous reporting mechanism. By rewarding websites that help profile, Privad would, somewhat paradoxically, lead to *fewer* ads and a better user experience by giving websites with highly targeted content the option of generating ad revenue even without showing ads.

4) Social feedback: The client can make use of social information when available. One can imagine a user's profile affecting the profiles of the user's friends in a privacy-preserving manner; something industry has been attempting but, so far, has been unsuccessful in accomplishing [15]. Future work will determine the precise protocol for performing this privacy-preserving peer-to-peer profiling.

B. Security Analysis

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

Solution S1: 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 needs to be answered by privacy advocates and users. 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 [13]). This would protect against userspace malware, but not rootkits that compromise the OS. Another option is to leverage trusted hardware (e.g. [18]) when available. How best to handle the profile from this perspective is both an ongoing research question and a policy question.

A2: A related attack is leaking the user's profile in case the user's computer is stolen or confiscated.

S2: Again our general stance is that while Privad makes it easier for the attacker by collecting the information in one place, Privad does not fundamentally change anything. The attacker today can already access browser cache and history, and private documents. And existing solutions to protecting documents, such as full-disk encryption, apply equally to the user profile.

In the remainder of this section, we look at attacks on each of the four profiling mechanisms.

2) Crawling: **A3:** The dealer reads the URL in the messages in line 2 of Protocol 1, or the profile attributes in the messages in line 7, thus learning the browsing behavior of the users and eventually building up profiles of users identified by the IP address thereby violating Profile Anonymity.

S3: The message in line 2 is encrypted with the broker's public key, the private key for which is known only to the broker. And the message in line 7 is encrypted with a shared-key that was distributed only to the broker through the message in line 2. This trivially prevents the dealer from viewing the contents of messages. Note that this same attack and solution apply for all message exchanged between client and broker in later sections, and is not mentioned again.

A4: The broker associates the contents of a single client message with the identity of the client, thus learning some of that client's browsing activity and profile information thereby violating Profile Anonymity.

S4: No PII is made available to the broker. The dealer hides the IP address from the broker. There is no PII information in the message itself. Only collusion between dealer and broker defeats this solution. Note that this same attack and solution apply for all message exchanges between client and broker, and is not mentioned again.

A5: The broker links together multiple messages associated with the same client. Over time this allows the broker to build a unique profile of the client thereby violating Profile Unlinkability, and to then use external means to identify the client.

S5: There is nothing in the message that allows it to be linked to other messages from the same client. The shared key K is unique for each message. τ is well-known and fixed across all users. Finally, the URL is stripped of all URL parameters, for instance website session IDs, that might otherwise link it to another URL visited by the same user. The reference monitor enforces this. We consider timing attacks next.

A6: The broker could try to link messages from the same client, but with different URLs, through the timing of those messages. For instance, based on typical browsing time for a URL.

S6: Since messages from many different clients are interspersed, reliably linking messages based on timing alone is highly unlikely to be successful. The broker could improve the chances by additionally performing outgoing link analysis on websites, but, uncertainty increases as the time between the two URLs increases. In fact, the reference monitor can arbitrarily increase the time between URLs by randomly delaying messages. This does not effect performance because no real-time activity at the client waits on the result of this message. Indeed, there is no real-time component of the entire Privad system that depends on synchronous message delivery.

A7: 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.

S7: 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.

3) Scraping: **A8:** Information scraped by the client includes both PII and profile information, which the client transmits to the broker.

S8: The monitor insures that the client follows the Privad protocol. Since no message exists to send scraped information, the client must encode it in some Privad message. There client has no degree of freedom in the contents of the message in line 2: τ is fixed, K is picked by the reference monitor, and Url is a URL visited recently by the user (which can be validated by the reference monitor). Indeed, no Privad message allows the client to send arbitrary data. Thus the client cannot send any scraped information, at least overtly, in this or any other Privad message. Section X deals with covert channels.

4) Metadata: **A9:** A publisher may present a profile attribute that is uniquely bound to the user. Later, acting as an advertiser the publisher may target ads to that unique attribute plus additional attributes. But analyzing which ads are shown, the publisher can link the additional attributes that match the user to the unique identifier. Since the publisher also know the user's IP address, this would violate Profile Anonymity.

S9: First, uniqueness of profiling metadata can be detected by accessing the website from different computers. Privacy advocates may operate crawlers to detect such behavior. Second, the client requires multiple publishers to present the same attribute before adding it to the user's profile (line 1.13). This raises the bar by requiring multiple colluding publishers to convince the user to visit their sites.

V. Ad Dissemination

The goal of the ad dissemination protocol is to scalably disseminate ads to users in a privacy-preserving manner. As we discuss in [2], sending all ads to all users, while private, doesn't scale to present-day requirements. Private information retrieval (PIR) provides stronger guarantees than necessary at significant cost. We therefore design a scalable privacy-preserving publish-subscribe (pub-sub) protocol.

Input at $\mathbb{C}$: UProf — User profile from line 1.14
Input at $\mathbb{C}$: $\mathsf{Chan} = \{\mathsf{Int}, \mathsf{Dem}_{1\ldots\tau}\}$ — Channel
  where: Int — Some interest category in UProf
  where: $\mathsf{Dem}_{1\ldots\tau}$ — Some broad demographics in UProf
Input at $\mathbb{C}$: K — Per-subscription symmetric key
Input at $\mathbb{B}$: $\{A_i\}$ — Some set of ads matching Chan
  where: $A_i = \langle aid, \mathsf{Tgt}_{1\ldots\eta}, \mathsf{data}\rangle$ — An Ad
  where: aid — Ad ID
  where: $\mathsf{Tgt}_{1\ldots\eta}$ — Targeting information ($\supseteq$ of Chan)
  where: data — Text, image or flash content
Input at $\mathbb{B}$: iid_i — Single-use instance ID for A_i

1: **protocol** SUBSCRIBE
2: $\quad\mathbb{C} \Rightarrow \mathbb{D}$ **send** $\mathcal{S} = \mathcal{E}^{\mathcal{P}}_{B}(\langle \mathsf{Chan}, K\rangle)$
3: $\quad@\mathbb{D}$ **store** $\langle xid, \mathbb{C}\rangle$
4: $\quad\mathbb{D} \Rightarrow \mathbb{B}$ **send** $\langle xid, \mathcal{S}\rangle$
5: $\quad@\mathbb{B}$ **recover** $\langle \mathsf{Chan}, K\rangle$ **from** $\mathcal{S}$
6: **protocol** PUBLISH
7: $\quad$**for all** A_i **do**
8: $\quad\quad@\mathbb{B}$ **store** $\langle iid_i, A_i\rangle$
9: $\quad\quad\mathbb{B} \Rightarrow \mathbb{D}$ **send** $\langle xid, \mathcal{P} = \mathcal{E}^{\mathcal{S}}_{K}(\langle A_i, iid_i\rangle)\rangle$
10: $\quad\quad@\mathbb{D}$ **recover** $\mathbb{C}$ **using** xid
11: $\quad\quad\mathbb{D} \Rightarrow \mathbb{C}$ **send** $\mathcal{P}$
12: $\quad\quad@\mathbb{C}$ **recover** $\langle A_i, iid_i\rangle$ **from** $\mathcal{P}$

Protocol 2: Privacy-preserving Ad Dissemination (initial version)

A. Basic Protocol

The pub-sub protocol (Protocol 2) consists of a client's request to join a *channel* (defined below), followed by the broker serving a stream of *ads* (defined below) to the client.

Each channel is defined by an 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. 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 and transmitted through the dealer (lines 2–5). The request contains the channel, and a per-subscription symmetric key chosen by the client. The broker encrypts ads using this key and sends them to the client along the reverse path (lines 7–12). The dealer stores a mapping between the request and the client (line 3), which it later uses to route subscribed ads to the correct client (in line 10). The broker also attaches a unique instance identifier (iid) to each ad published for use in the auction protocol described later in Section VI. Note, in Section VII we modify Protocol 2 slightly by adding a second dealer to defend against an attack arising from combining it with the reporting protocol.

Ads and targeting: In addition to an ad ID (aid) and the

ad content, ads contain targeting attributes ($\mathsf{Tgt}_{1\ldots\eta}$). An ad matches a channel if the targeting attributes for the ad is a superset of the attributes that define the channel. However, not all ads matching a channel may be sent to the client (e.g. ads nearing their daily budget). The client filters out ads if the set of targeting attributes is not a subset of the user's profile; filtering happens after the auction protocol (Section VI).

B. Security Analysis

Attacks analogous to A3, A4, A5, A6, and A7 also apply to ad dissemination and have the same respective solutions. The timing attack (A6) corresponds to a flurry of subscribe messages (e.g. when the client starts up), which the reference monitor diffuses as before.

A10: 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.

S10: The demographic attributes that may define a channel are limited by convention. We would expect the limitation to be quite tight and to consist of non-sensitive demographics, for instance gender, region, and language only. Since channel definitions are public, a watchdog group can detect when the limitation is exceeded. Additionally, the monitor can filter out any channel definitions that exceed the defined limit.

A11: The dealer registers an ad with the broker, and then attempts to inject it into the stream of ads published to a given client, such that if the ad is shown, the dealer can link the targeting attributes with the client's IP address in violation of Profile Anonymity.

S11: All ads published are encrypted with the symmetric key K known only to the client and the broker (line 9). The dealer cannot generate a message that validates after decryption. The reference monitor insures message validity (e.g. using checksums). Note that this same attack and solution apply for all message exchanged between client and broker in later sections, and is not mentioned again.

VI. Ad Auctions

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 challenge, of course, is in doing so while preserving user privacy and, like today, not revealing the advertiser's bid to anyone other than the broker. Since the broker does not trust the client, the naïve approach of sending bid information with the ad in the dissemination phase and performing the auction at the client does not apply.

One simple approach is to perform auctions during the ad dissemination phase (see [2] for details). Once the broker determines the set of ads to be published for a given subscription, the ads are ranked and sent to the client in sorted order. The client shows higher ranked ads first. This approach does not require any new protocol, and doesn't introduce any new privacy risks, and as such, serves as a proof-of-existence that extremely simple privacy-preserving (both user and bid privacy) auctions are possible. The limitation, however, is that because ads are auctioned at the broker on a per-subscription basis, the ranking cannot compare ads on different channels or take per-user metrics into account. It is therefore a considerably weaker auction than those which can be performed in today's centralized ad networks.

We therefore design and analyze a second auction protocol that, while considerably more complex, implements precisely the GSP auction used by Google today [5] within the confines of the Privad model.

Input at $\mathbb{C}$: $\{\langle A_i, iid_i\rangle\}$ — published ads, from line 2.12
where: A_i — An Ad (defined in Protocol 2)
where: iid_i — Unique instance ID for $\mathbb{C}$'s copy of A_i
Input at $\mathbb{C}$: $Q_i^{\mathbb{C}}$ — User metrics for A_i if ad matches user (e.g. quality of match), otherwise random number
Input at $\mathbb{B}$: Bid_i — Advertiser bid for A_i
Input at $\mathbb{B}$: $Q_i^{\mathbb{B}}$ — Broker metrics for A_i (e.g. CTR)
Input at $\mathbb{B}$: K — Symmetric key known only to $\mathbb{B}$
Define: $\mathcal{S}_x$ — Ranking score for item x

1: **protocol** MixAuctionRequests
2: $\quad @\mathbb{D}$ **initialize** $\mathcal{A} \leftarrow \emptyset$
3: $\quad$ **for** *multiple clients* $\mathbb{C}_x$ **do**
4: $\quad\quad \mathbb{C}_x \Rightarrow \mathbb{D}$ **send** $\mathcal{T} = \{\langle iid_i, Q_i^{\mathbb{C}}\rangle\}$
5: $\quad\quad$ **for** $\mathcal{T}_i = \langle iid_i, Q_i^{\mathbb{C}}\rangle$ **in** $\mathcal{T}$ **do**
6: $\quad\quad\quad @\mathbb{D}$ **store** $\langle iid_i, \mathbb{C}_x\rangle$
7: $\quad\quad\quad @\mathbb{D}$ **update** $\mathcal{A} \leftarrow \mathcal{A} \cup \{\mathcal{T}_i\}$ ▷ Mix
8: **protocol** Auction
9: $\quad \mathbb{D} \Rightarrow \mathbb{B}$ **send** $\mathcal{A}$
10: $\quad$ **for all** $\mathcal{T}_i = \langle iid_i, Q_i^{\mathbb{C}}\rangle$ **in** $\mathcal{A}$ **do**
11: $\quad\quad @\mathbb{B}$ **recover** A_i **using** iid_i ▷ see line 2.8
12: $\quad\quad @\mathbb{B}$ **set** $S_{\mathcal{T}_i}$ *based on* $\mathsf{Bid}_i, Q_i^{\mathbb{B}}, Q_i^{\mathbb{C}}$
13: $\quad @\mathbb{B}$ **sort** $\mathcal{T}_{1\ldots\kappa}$ *based on* $S_{\mathcal{T}_i}$ ▷ Auction
14: $\quad$ **for all** $\mathcal{T}_i = \langle iid_i, Q_i^{\mathbb{C}}\rangle$ **in** $\mathcal{T}_{1\ldots\kappa}$ **do** ▷ sorted order
15: $\quad\quad \mathbb{B} \Rightarrow \mathbb{D}$ **send** $\mathcal{R}_i = \langle iid_i, \mathcal{E}_K^{\mathcal{S}}(\mathsf{Bid}_i)\rangle$
16: $\quad\quad @\mathbb{D}$ **recover** $\mathbb{C}_x$ **using** iid_i
17: $\quad\quad \mathbb{D} \Rightarrow \mathbb{C}_x$ **send** $\mathcal{R}_i$ ▷ Un-mix
18: $\quad\quad @\mathbb{C}_x$ **recover** A_i **using** iid_i
19: $\quad\quad$ **if** A_i *matches user at* $\mathbb{C}_x$ **then**
20: $\quad\quad\quad @\mathbb{C}$ **store** *(in order)* $\mathcal{D}_i = \langle A_i, \mathcal{E}_K^{\mathcal{S}}(\mathsf{Bid}_i)\rangle$

Input at $\mathbb{C}$: $\mathcal{D}_{1\ldots\eta}$ — an ordered subset of ads (from 3.20)
Define: SPBid_i — $\mathcal{E}_K^{\mathcal{S}}(\mathsf{Bid}_{i+1})$ from $\mathcal{D}_{i+1}$ if $i < \eta$, else ϕ
21: **function** SecondPrice($\mathcal{D}_{1\ldots\eta}$)
22: $\quad$ **for all** $\mathcal{D}_i$ **in** $\mathcal{D}_{1\ldots\eta}$ **do**
23: $\quad\quad @\mathbb{C}$ **set** $\mathcal{D}'_i = \langle A_i, \mathsf{SPBid}_i\rangle$
24: $\quad$ **return** $\mathcal{D}'_{1\ldots\eta}$

Protocol 3: Privacy-preserving GSP Auctions

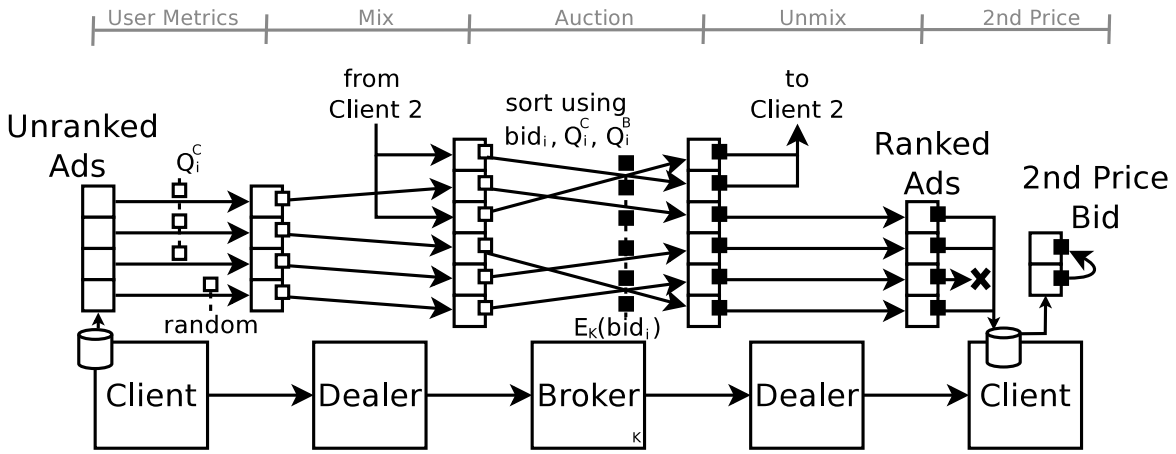

Fig. 2: Privacy-preserving GSP Auctions

A. Basic Protocol

The broker conducts the auction in a separate phase after ad dissemination. Figure 2 illustrates the auction protocol listed in Protocol 3. For each ad received, the client computes a score (integer between 1–5) as follows: for ads that match the user profile, the score reflects the quality of the match. For ads that don't match the user, the score is a random number. The client sends the instance ID and score 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 (lines 3.3–3.7). The broker ranks all the ads in the request. The ranking is based on bids, click-through-rates, and the client score (lines 3.10–3.13). Note the 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 slices the list and forwards each client its result (lines 3.15–3.17). The client discards the ads that did not originally match the user, and stores the rest in ranked order (lines 3.18–3.20). Note that the entire exchange is unencrypted.

Second-Price Auctions: GSP uses second price auctions[3] to reduce price volatility [5]. To perform second-price auctions, the broker encrypts the bid information with a key known only to the broker and sends it with the auction result (line 15). When a set of ads are chosen to be shown to the user, the client copies the encrypted bid information from ad $i+1$ to ad i (line 3.23). As we mention later, this encrypted bid information is sent as part of the click report, which the broker decrypts to determine what the advertiser should be charged.

B. Security Analysis

A12: The dealer uses the iid_i to guess which ads (and therefore which pub-sub channels) the user is interested in, thus violating Profile Anonymity.

S12: iid_i are single use. As long as the broker chooses them in a randomized manner, the dealer cannot track which ad it is associated with since the ad is never sent in the clear. Even if the dealer (using a fake client) is subscribed to the same channel as a client, and they both receive a copy of the same ad, the iids will be unrelated.

A13: The broker uses the $Q_i^{\mathbb{C}}$ to determine which ads matched the user. Since the broker can link ads sent on the same pub-sub subscription, the broker could conceivably violate Profile Unlinkability.

S13: The client submits *all* ads received for auction, whether or not they match the user (enforced by the reference monitor). Thus the broker learns no more than he already knows unless he is able to determine which $Q_i^{\mathbb{C}}$ are real and which fake. This can be made extremely unlikely by using approaches that generate noise that is statistically indistinguishable from the real data.

A14: The broker attempts to link two ads published to the same client through different pub-sub subscriptions, thereby violating Profile Unlinkability.

S14: The property of the mix constructed at the dealer is such that tuples ($\mathcal{T}_i$) from the same client but for ads on different pub-sub channels are indistinguishable from tuples from two different clients each subscribed to one of the pub-sub channels. The pub-sub protocol provides the same property. Thus the broker doesn't learn anything new from the auction protocol.

We discuss the privacy implications of second-price auctions in the next section. Lastly, as before, straightforward attacks and solutions analogous to A4, A6, A7, and A11 apply to auctions. In the analog for the dealer injection attack (A11), while the dealer can inject arbitrary $\mathcal{R}_i$ tuples since messages are not encrypted, the reference monitor can still filter out invalid tuples (that were not part of the original request).

VII. Ad Event Reporting

Ad views and clicks, as well as other ad-initiated user activity (purchase, registration, etc.) needs to be reported to the broker in a privacy-preserving manner.

[3]Where each advertiser is charged what the advertiser ranked immediately below them bid.

Input at $\mathbb{C}$*:* type — view, click, purchase, etc.
Input at $\mathbb{C}$*:* pid — Publisher ID
Input at $\mathbb{C}$*:* $\mathcal{D}'_i = \langle A_i, \mathsf{SPBid}_i \rangle$ — The ad (see 3.23)
where: aid_i — Ad ID for A_i
where: SPBid_i — Second price bid (ϕ for views)
Input at $\mathbb{D}$*:* $\mathcal{B}$ — Set of blocked clients

1: $\mathbb{C} \Rightarrow \mathbb{D}$ **send** $\mathcal{R} = \mathcal{E}^{\mathcal{P}}_{B}(\langle \mathsf{type}, aid_i, pid, \mathsf{SPBid}_i \rangle)$
2: **if** $\mathbb{C} \notin \mathcal{B}$ **then**
3: $\quad \mathbb{D} \Rightarrow \mathbb{B}$ **send** $\langle xid, \mathcal{R} \rangle$
4: $\quad @\mathbb{D}$ **store** $\langle xid, \mathbb{C} \rangle$
5: $\quad$ **if** *click-fraud is suspected by broker* **then**
6: $\quad\quad \mathbb{B} \Rightarrow \mathbb{D}$ **send** xid
7: $\quad\quad @\mathbb{D}$ **recover** $\mathbb{C}$ **using** xid
8: $\quad\quad$ **if** $\mathbb{C}$ *encountered enough times* **then**
9: $\quad\quad\quad \mathcal{B} \leftarrow \mathcal{B} \cup \{\mathbb{C}\}$

Protocol 4: Privacy-preserving Ad Event Reporting and Blocking Click-Fraudsters (after detection)

A. Basic Protocol

The protocol for reporting ad events (Protocol 4) is straightforward. The report containing the ad ID and publisher ID is encrypted with the broker's public-key and sent through the dealer to the broker. The dealer stores a mapping between the request and the client (line 4.4), which it uses later to trace suspected fraudulent reports identified by the broker back to their originating client (lines 4.5–4.9). If a client is suspected of engaging in click-fraud more than some set threshold, the dealer blocks subsequent reports from that client (line 4.2).

B. Security Analysis

A15: The broker may use aid_i to link together reports from the same user. This is because Protocol 2 assumed (but did not enforce) that aid_i was reused for multiple users so that if ad A_1 was send both to $\mathbb{C}_1$ and $\mathbb{C}_2$, they both would receive aid_1, and the broker wouldn't be able to know which user clicked the ad when it saw aid_1 in a report. But if the broker uses a unique aid_i for each ad published, it can later link together reports for ads sent to the same pub-sub subscription thus learning which ads matched the user, thereby violating Profile Unlinkability.

S15: A simple defense against this is for the client to generate two subscriptions for every pub-sub channel that it joins. It then only considers ads (identified by the aid) that it receives on both channels. This defeats the attack because it forces the broker to send the same ad to many subscriptions in order to cause any one client from receiving copies on both of its duplicate channels. In doing so, this ensures other users are extremely likely to receive the same ad.

The problem with this defense is that it increases the number of ads distributed significantly. This is due in large part to the large number of ads from advertisers with small budgets. Such ads should only be viewed by a small number of clients, and therefore should ideally be distributed to a small number of clients. If, however, any client has to see a copy of the ad on both of its duplicate channels, the broker has to increase the number of times the ad is sent tremendously in order to insure that the right fraction of clients see duplicates.

An alternate approach is to add a second dealer ($\mathbb{D}_2$) to Protocol 2, placed between the original dealer and the broker. $\mathbb{D}_2$ terminates the encryption to/from the client — all messages encrypted with the broker's public key are instead encrypted with $\mathbb{D}_2$'s public key, and $\mathbb{D}_2$ is responsible for encrypting published ads with the shared key. This allows $\mathbb{D}_2$ to see anything the broker in the original protocol can see (which doesn't violate user privacy as discussed earlier). More importantly, $\mathbb{D}_2$ can now audit the broker by insuring aids are reused.

The broker could attempt to fool $\mathbb{D}_2$ with a Sybil attack: the broker creates n fake clients (botnet) that subscribe to a target channel; the broker satisfies $\mathbb{D}_2$ by reusing the aid $n+1$ times; but in reality, n of those are to his Sybils, thus effectively canceling the aid reuse. This attack doesn't work since the broker cannot steer specific messages to his Sybils because of the anonymity provided by the original dealer $\mathbb{D}$.

A16: The broker may use the second-price bid (SPBid_i) to link two ads that matched the user, thus potentially violating Profile Unlinkability if the combination of targeting attributes behind the ads exceeds the threshold. For honest-but-curious brokers, this requires ads to have unique bids so the SPBid_i can be linked to the ad. But since bids sent to the client are opaque, malicious individuals in the broker may encrypt a unique id instead of the bid. We protect against both.

S16: Second-price, by definition, requires the broker to link two ads. The simple solution is for the reference monitor to unset the SPBid_i when the combination of targeting information from the linked ad exceeds the threshold. While that works for the rare click report, it doesn't work for view reports. This is because even though a single report doesn't allow more than two ads to be linked, multiple reports may allow a chain to be constructed where each individual link is below the threshold, but the sum total of information linked exceeds the threshold. This is typically encountered for views when there are multiple ad boxes in a webpage. For precisely this reason, view reports have SPBid_i set to ϕ to prevent the broker from linking together a chain of ads.

A17: An advertiser (or publisher) may attempt to benefit by causing the client to report an incorrect SPBid_i. While the attacker cannot inject an arbitrary bid (since bids are encrypted by the broker), the attacker may substitute a higher or lower bid from another ad. Conducting this attack requires the attacker to compromise the client (e.g. using malware).

S17: We are not concerned with under-reporting — an advertiser attempting to lower his costs, considering he has malware presence on the user's computer, does not need an ad network to drive user traffic in the first place. Over-reporting the second-price bid is capped (at the broker) by the advertiser's actual bid. In order to appreciably influence ad revenue, the attacker must compromise a large number of users (i.e. a botnet); Section IX discusses how Privad defends against botnets.

Lastly, attacks A3, A4, A5, A6, and A7 and their respective

solutions have straightforward analogs in the reporting protocol.

VIII. Click Anonymization

If when a user clicks on an ad the user goes directly to the advertiser's website, then the advertiser would know a significant amount of information about the user: some demographic information (whatever matched the ad) and the user's IP address. Worse, if the advertiser links multiple clicks, for instance because they came from the same IP address or supplied the same cookie, then the advertiser can build up a profile of the user derived from the Privad profile.

One approach to mitigating this would be for the client to go through a standard web proxy. The problem here is that the web proxy is in a position to know a great deal about the user. Of course, this is true in any event for web proxies today, and that doesn't stop many people from using them, but Privad makes the situation slightly worse by making it possible for the web proxy to glean the user's profile.

A stronger approach is to use the dealer/broker infrastructure itself to anonymize clicks. By using 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. Unfortunately this approach cannot be used to keep the client anonymous from the advertiser forever. Advertisers may legitimately eventually ask the user for PII, for instance credit information to make a purchase. Once this happens, there is little value in continuing to proxy the client/advertiser session.

A. Basic Protocol

The reference monitor requires that the click URL must be a secure HTTP (https) URL. It establishes a secure session encrypted end-to-end between the client and the advertiser, proxied through the dealer and then the broker. The monitor watches the pre-encrypted session, and either prevents any "child" sessions from being launched (i.e. new sessions in new browser windows), or allows them but also insures that they are https-based and watches them as well. As long as no data is posted by the browser, the monitor continues to steer the sessions to the dealer. Once data is posted (either through the HTTP POST or through URL parameters), the monitor posts a warning to the user indicating that he or she will be directly connected to the advertiser, and listing the profiling information that the advertiser can deduce from the user as well as what information is going to be posted. Once the user approves this handoff, the HTTP request is sent directly to the advertiser. The monitor also sends a report indicating that a handoff has taken place. The report type is "handoff" and contains only the ad ID. This can be used by the broker to monitor for malicious advertisers (for instance, advertisers that very frequently force a handoff).

Note that the advertiser needs to be prepared to receive this request from a new IP address, but there are a number of ways that this can be handled.

B. Security Analysis

A18: An attacker wishes to associate identifiable users with profile information. This might be done for instance to build up a marketing database. The attacker masquerades as an advertiser, and posts ads for various profile attributes. When users click, the attacker somehow collects PII and associates it with the attribute. This might also be done to attract users with certain profiles in order to gain some advantage over the user in the ensuing transaction. By way of example, suppose that two of the user attributes collected by the client are 1) whether the user has any given disease (say AIDS), and 2) whether the user has health insurance. By targeting ads to these attributes, companies could exploit this information in a number of ways, for instance to avoid selling insurance to people that appear to have preexisting conditions (even when the customers try to hide the existence of those conditions, or in fact don't have the conditions), or to prey on desperate people with certain diseases and no health insurance.

S18: Ultimately Privad does not guard against this attack. Broadly speaking, after a user clicks a targeted ad, the advertiser has some knowledge of the user and therefore an implicit advantage. This is true of any targeted advertising system, not just Privad. Generally it is up to the broker to limit the target categories, or combinations of categories and advertisers. This can for the most part be monitored. For instance, brokers, advertisers, and privacy advocates could work together to define a set of guidelines for categories and combinations of categories and advertisers. Privacy advocates could also provide users with reference monitors that allow or disallow ads according to these guidelines, tunable by the user. Of course, taken too far, this could simply degenerate into ad blocking. As a result, there needs to be some give-and-take between the interests of advertisers and those of users so that on one hand advertising continues to be a source of revenue for web sites, but on the other exploitation through targeted advertising is minimized. Note that if the purpose of the attack is to simply gather large amounts of data about many users, there is a potentially substantial money cost since the attacker must pay for clicks.

A19: An adversarial broker and advertiser collude to fool users into thinking that a category means something that it doesn't. For instance, say an advertiser wants to target people with some disease and no health insurance, but privacy advocates wish to prevent it. The broker and advertiser could collude to agree that the category "tennis strings" really means "AIDS", and the category "lima beans" really means "no health insurance". The advertiser then targets its ads to "tennis strings" and "lima beans", and the ad gets shown to people with AIDS and no health insurance.

S19: This can be detected by humans monitoring categories and see if the ads shown make sense. The cost of doing so, however, is quite high. This cost can be mitigated somewhat by displaying the matching attributes to users, and providing a simply way for (savvy) users to indicate that the attributes don't match the ad. Once again, note that the cost of being

caught in this attack is quite high for the broker.

IX. Click Fraud Detection

Generally speaking, privacy makes click-fraud more challenging because clients are hidden from the broker. As briefly described in Section VII, Privad overcomes that challenge through explicit privacy-preserving coordination between broker and dealer. Both the broker and dealer participate in detecting click-fraud; the dealer by measuring view and click volumes from clients, the broker by looking at overall click behaviors for advertisers and publishers. The broker can tell a dealer when a client is suspected of click-fraud by telling the dealer which reports are suspected as being involved in click-fraud. This is possible because each individual report has a unique identifier associated with it that the dealer can map back to the client. This allows the dealer to identify suspicious clients, and to drop future reports coming from a suspicious client.

This mechanism has the effect of more-or-less putting Privad back on an even footing with current ad networks as far as click-fraud is concerned. As with today's ad networks, there is no silver bullet for defending against click-fraud. And like ad networks today, the approach we take is *defense in depth* — a number of overlapping detection strategies operate in parallel; each detection strategy can be fooled with some effort; but together, they raise the bar.

A. Mechanisms

1) 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 NAT'ed networks or public proxies used by multiple users so higher thresholds may apply to them.

2) 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.

3) 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, using the mechanism described in Section VII, the originating client is flagged as suspicious.

4) 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.

5) Premium Clicks: Based on the insight behind [11], 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.

6) 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 [14]. Basically, bait ads contain the targeting information of one ad, but the content (text, graphics) 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. [6]). 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 blocks the client after the client is implicated enough times.

B. Security Analysis

1) Click-Fraud: **A20:** An individual clicks on a large number of ads (or clicks on an ad a large number of times). The individual may be a publisher clicking on ads on his webpage to increase ad revenue, or an advertiser clicking on his competitor's ads to inflict financial damage.

S20: The per-user thresholds mechanism defends against individual attackers. The individual may attempt to spread his clicks over multiple dealers such that he stays below the detection threshold at each dealer. One option is for dealers to share per-client statistics. Another option is to restrict the dealers a client can access (e.g. by country). Of course, the attacker can enlist a botnet, which is discussed next.

A21: An attacker (publisher, advertiser) uses a botnet to click ads.

S21: Blacklists and honeyfarms help detect botnets. A bot may attempt to detect and avoid infecting a honeyfarm, but that can be made arbitrarily hard by the honeyfarm operator (cost permitting).

In addition, historical statistics can help detect high-intensity attacks. The attacker can avoid this by building up the attack gradually to avoid any sudden changes to statistics, but in so doing he gives the other detection mechanisms more time to kick in.

Finally, bait ads help detect attackers that manage to avoid threshold and statistic based detection. The intent is for the broker to proactively disseminate bait ads on all pub-sub channels. Since clicking on bait is a strong signal, even relatively stealthy attacks can be detected quickly.

At the same time, premium clicks mitigate the impact of fraudulent non-premium clicks by effectively devaluing them. Since the attacker needs to spend money to acquire and maintain "premium" status for each bot, this limits the size of the botnet.

A22: The dealer may itself engage in click-fraud, or otherwise not comply with the broker's request to stem an attack.

S22: The broker can audit that the dealer is operating as expected. One approach is for the broker to launch a fake click-fraud attack from fake clients, and ensure the dealer

blocks them as expected. Another approach is for the broker to monitor the rate of bait clicks on a per-dealer basis before and after identifying an attack. If the bait rate drops for one dealer, but doesn't drop for another dealer, the broker can conclude that the latter may be misbehaving.

2) User Privacy: **A23:** The dealer launches a click-fraud attack. The broker learns some attack signature, e.g. publisher $\mathbb{P}$ (or advertiser $\mathbb{A}$) is under attack, and begins flagging click reports for $\mathbb{P}$ (or $\mathbb{A}$). In addition to flagging clients under the dealer's control, this creates collateral damage by flagging other (innocent) clients ($\mathbb{C}_i$). The dealer can now infer that $\mathbb{C}_i$ visited and clicked an ad on $\mathbb{P}$ (or clicked an ad for $\mathbb{A}$), which would allow it to violate Profile Anonymity.

S23: This attack applies only in the unlikely scenario that there are no other click-fraud attack taking place other than the one controlled by the dealer. When multiple click-fraud attacks are taking place, the dealer doesn't learn which publishers and advertisers are under attacks in the first place since the message in line 4.6 identifies only the (encrypted) message the broker suspects but doesn't reveal its contents. In any even, the dealer cannot learn which click-fraud attack a given client is implicated in. Together this ensures the dealer does not learn anything new about innocent clients.

X. Other Privacy Aspects

A. 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. An adversarial individual within a broker organization would have a hard time doing this. To see why, we first describes the characteristics of the 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 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 (profiling, and subscription messages) which otherwise represent the best covert channel opportunity.

The next best opportunity for a covert channel comes from the user score $Q_i^{\mathbb{C}}$ in the auction message (Protocol 3). That is because this is the only client-controlled message field that originates from the client (versus being received from the broker). Furthermore, this field has a random component, albiet within a small range since the user score need only be 2 or 3 bits in size. Even here, it would be hard for an individual in the broker organization to generate the channel without getting caught. He would have to write code in the client that over-rides the normal selection of user scores. He would have to write code in the broker that detects the sequence of $Q_i^{\mathbb{C}}$ values that provides a signal and transmits this signal to the attacker. 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.

B. 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 would have to make a hypothesis as to what keywords the advertiser is targeting, then try some tests to see if the hypothesis is correct. For instance, the recipient could search for those keywords and see if the advertiser's ads appear. The recipient could also see if the ads show up on webpages that are found by searching on those keywords, though in this case the recipient can't be sure that it is those keywords that caused the ad to appear. 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 the 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 also possible, though costly, to learn how much an advertiser bids for certain keywords. This is done by competing with the advertiser for those keywords, and seeing what price beats the advertiser. Privad does not change this. It is hard to determine the overall budget an advertiser has with current systems, and Privad also does not change this.

C. Broker Privacy

Finally, the broker also has some privacy concerns, mainly in the form of intellectual property protection of its profiling mechanisms. Intellectual property is best protected through patents, copyrights and DMCA[4] provisions. Companies often additionally obscure binaries to frustrate attackers. Privad doesn't change any of this. In fact, it is in part for this reason that we place trust in the monitor rather than the client. The other reason, of course, is that client code is so complex that it would be difficult to be sure that it really follows the protocol in any event.

XI. Related Work

There is surprising little past work on the design of private advertising systems, and what work there is tends to focus

[4]Digital Millenium Copyright Act (USA)

on isolated problems rather 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 ([2] contains a broader survey of related work). In particular, we look at Juels [10], Adnostic [17], Nurikabe [12], and Freudiger et. al [7].

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 result (which remain encrypted). Instead, they send the results to a 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 centralized advertising 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. With Privad as described herein, unless the user has a reason to expose his or her PII (i.e. make a purchase), he or she remains anonymous to the advertiser. Thus for the majority of clicks, privacy is maintained. Further, we believe that the Privad architecture is well-suited to take privacy from the advertiser further still.

The Nurikabe approach also has a number of practical shortcomings. It cannot report views without exposing this to the advertiser, thus reducing user privacy from the advertiser even more. View reporting is important, in part because it allows the advertiser to know how well its ad campaign is going. Nurikabe also gives up any visibility into click fraud, because they expect either advertisers to fend for themselves, or require trusted advertisers, neither of which is practical. 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, we have shown through ad measurements [3] 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 good targeting and complete privacy can be achieved.

XII. Summary and Future Work

This paper presents a privacy and click-fraud analysis of a privacy-preserving online advertising system called Privad. Privad was designed to improve significantly on current advertising privacy. If this were the only goal of Privad, the design would be quite easy. Equally important, however, is that Privad successfully compete with existing online advertising companies. This means that Privad needs to fit into the contemporary advertising business model, and needs to be deployable.

It would be easy at this point to conclude that Privad satisfies these goals 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. The main voices for society are privacy advocates and government policy makers. An important component of our future work is to engage with

privacy advocates and policy makers both to educate them and to obtain their feedback. Towards this end, we have started dialogs with a number of these groups, 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 [1]). Besides this, Privad has a number of operational privacy parameters for which policies must be set. These parameters determine for instance how fine-grained user profiles can be, and what information is allowed and disallowed. Because Privad dramatically changes the privacy equation, we believe that the debate on what these policies should be must be started anew. As future work, we hope to push this debate along.

Privad so far focuses primarily on privacy from the broker. However, a key element is privacy from the advertiser. While this paper takes a first small step in this direction, we believe that much more can be done within the context of the Privad architecture. We expect this to be a major direction of our work looking forward.

Besides this, we need a better understanding of a number of Privad components. Foremost among these are the bait approach to click-fraud, Privad's auction, and how best to do profiling. We are actively working on all of these problems.

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.

REFERENCES

[1] Authors omitted. privacy roundtables - comment, project no. p095416, comment number: 544506-00017. In *URL blinded*.

[2] Authors omitted. privad: Practical privacy in online advertising. In *Currently under submission to NSDI10, paper provided to PC chairs*.

[3] Authors omitted. Serving Ads from `localhost` for Performance, Privacy, and Profit. In *Blinded*.

[4] Council of Europe, Data Protection Working Party. Opinion 1/2008 on data protection issues related to search engines, Apr. 2008.

[5] 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.

[6] 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.

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

[8] E. Hansen. California senate approves anti-Gmail bill. In *CNET News*, May 2004.

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

[10] 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.

[11] 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.

[12] D. Levin, B. Bhattacharjee, J. R. Douceur, J. R. Lorch, J. Mickens, and T. Moscibroda. Nurikabe: Private yet accountable targeted advertising. Submitted to NSDI10, contact johndo@microsoft.com for copy, 2009.

[13] 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.

[14] 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.

[15] PCWorld. Facebook's Beacon More Intrusive Than Previously Thought. http://www.pcworld.com/article/id,140182-c,onlineprivacy/article.html, Nov. 2007.

[16] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymity - a proposal for terminology, 2001.

[17] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas. Adnostic: Privacy Preserving Targeted Advertising. http://crypto.stanford.edu/adnostic/adnostic.pdf, 2009.

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

[19] D. Yu. How to spam facebook like a pro: An insider's confession. http://www.techcrunch.com/2009/11/01/how-to-spam-facebook-like-a-pro-an-insiders-confession/, 2009.