Targeted Advertising... And Privacy Too

Ari Juels

RSA Laboratories
Bedford, MA 01730, USA
E-mail: ajuels@rsasecurity.com

Abstract. The Web presents a rich and powerful tool for aggregation of consumer information. A flurry of recent articles in the popular press has documented aggressive manipulation of such information by some companies for the purposes of targeted advertising. While advertisers tout the economic and social benefits of such advertising, consumer privacy groups have expressed grave concerns about its potential abuses, and called for legislative policies to protect sensitive consumer data. In this paper, we explore the notion that targeted advertising and privacy protection need not necessarily be conflicting goals. We describe some conceptually simple technical schemes that facilitate targeted advertising, but also offer protection for sensitive consumer data. Some simple proposals do not even require the use of cryptography. (As an example, we mention an existing scheme in commercial deployment.) We also consider some more sophisticated protocols offering greater assurance of privacy. These involve cryptographic constructions that may be thought of as partial, practical PIR (private information retrieval) schemes.

1 Introduction

In February 2000, a major Web advertising firm known as DoubleClick touched off a furor in the press with the announcement of a more aggressive policy of consumer data aggregation. DoubleClick declared that it would begin to integrate offline information about consumers into its existing database of online information, this latter derived from surveillance of consumer Web surfing [40]. This announcement came in the midst of a number of articles in the popular press regarding surreptitious sharing of consumer information. For example, a week earlier, a report released by the California HealthCare Foundation alleged that a number of health-related Web sites were violating their own stated privacy policies and divulging sensitive information about customers to third parties [21]. Bowing to public pressure, DoubleClick retracted its policy announcement in early March [8]. A number of companies have recently attempted to allay consumer concerns by making more explicit claims about their privacy policies.

While consumer and privacy advocacy groups vigorously decry abuses by firms like DoubleClick, advertisers defend their policy of harvesting and exploiting demographic information by highlighting the benefits of targeted advertising. Consumers, they maintain, are more likely to find interest in advertising tailored

to their own preferences, and such advertising consequently leads to greater consumer market efficiency. The United States government has addressed the issue by promoting a policy of industry self-regulation, leading to friction with the European Union, which has sought more stringent consumer privacy guarantees.

In this paper, we explore the notion that targeted advertising and consumer privacy need not in fact be conflicting aims. We describe several simple, practical technical solutions that enable use of detailed consumer profiles for the purposes of targeting advertisements, but protect these profiles from disclosure to advertisers or hostile third parties. The most basic schemes described here do not even require use of cryptography. We mention one naïve variant that even serves as the basis of a current product offering [2].

The underlying idea is quite simple. Rather than gathering information about a consumer in order to decide which advertisements to send her, an advertiser makes use of a client-side agent called a *negotiant*. The *negotiant* serves a dual purpose: It acts as a client-side proxy to protect user information, and it also directs the targeting of advertisements. The negotiant requests advertisements from the advertiser that are tailored to the profile provided by the user. The advertiser can control the palette of advertisements available to the negotiant, as well as the process by which it decides which ads to request. At the same time, the advertiser learns no information about the consumer profile beyond which advertisements the negotiant requested. In more sophisticated variants, the negotiant is able to participate in a protocol whereby the advertiser does not even learn what ads a given user has requested, but only sees ad requests in the aggregate. The end result is that the advertiser is able to target ads with a high degree of sophistication, and also to gather information on ad display rates, all without learning significant information about individual consumer profiles.

Some restriction must be placed on advertiser control of negotiants. Otherwise, the advertiser can manipulate them so as to extract profile information from individual consumers. The fact that negotiants may be viewed and controlled by users helps offset this vulnerability, as we discuss below.

1.1 Previous Work

A negotiant may be viewed as a client-side software proxy. The related approach of using server proxies as a means of protecting consumer privacy is a well established one. For example, for a subscription fee, companies such as Zero-Knowledge Systems [5] offer customers an encrypted channel to one or more proxy servers that anonymously reroute requests to destination servers. The proxy servers thus act as intermediaries, shielding the client from positive identification. Proxy services may be cryptographically strengthened through the use of *mix networks*. A mix network is essentially a distributed cryptographic algorithm for interleaving multiple channels so as to anonymize them. We describe the idea in more detail in Section 2.1. For on-the-fly communications, however, the most powerful mix networks are often not practical.

A variant on the idea of proxy servers is the Crowds project at AT&T Labs [1, 37, 38]. A "crowd" is a group of users, preferably with disparate geographical

and other characteristics, that serve to shield one another's identities. The service requests of a user in a crowd are randomly rerouted through other crowd members, rendering the identity of the user indistinguishable from those of other crowd members. In this system, trust is embodied partly in an administrative server responsible for forming crowds, and partly in other crowd members. The user trusts other crowd members not to eavesdrop on or tamper with communications, and, to a lesser extent, not to perform traffic analysis.

The proxy server and crowd approaches seek to provide a maximum of consumer privacy. While they can be combined with cookies, or other user tracking devices, they do not aim to accommodate more fine-grained control of Web server access to user data. The Platform for Privacy Preferences Project, known as P3P [4], focuses precisely on this latter problem of refined user control of personal demographic information. The goal of P3P is to enable Web sites to publish precise specifications of their privacy policies, and to enable users to exercise control over how and when their private data are divulged in response to these policies. Under the aegis of the World Wide Web (W3) Consortium, P3P aims to set forth a standard syntax and body of protocols for general use on the Web.

Another system, described in [9], combines properties of the P3P scheme as well as a variant of the proxy server approach. This scheme enables users to perform Web serving using a variety of different "personae". It offers controls for the user in the release of information, and also permits merchants to pool information in a controlled manner. The system aims to accommodate existing infrastructural elements, and assumes the use of periodic merchant auditing, in conjunction with consumer control, to achieve privacy assurances.

Underlying the P3P and related approaches is the presumption that mediation between consumers and advertisers is a matter of deciding what information consumers choose to reveal explicitly. Of course, though, once a user reveals a given piece of information, its dissemination is no longer within his or her control. As we explain above, we set forth a different approach in which consumers and advertisers to decide jointly in a privacy-protecting manner what advertisements consumers should be sent, without explicit revelation of consumer information. For the more strongly privacy protecting variants of our negotiant scheme, we consider variants on the idea of *private information retrieval* (PIR).

A PIR scheme enables a client to request a piece of data from a server – such as an advertisement – in such a way that the server learns no information about the client request. Let Bob represent a user, and let Alice represent a server that maintains a database containing bits $B = \{b_1, b_2, \ldots, b_n\}$. Alice might be an advertiser, and B might represent the collection of advertisements held by Alice. The aim of a PIR scheme is to enable Bob to retrive a bit $b_r \in B$ (or, by extension, multiple bits) of his choice from Alice in such a way that Alice learns no information about r. Of course, this may be accomplished trivially by having Alice send all of B to Bob. Following early work in [13, 33], it was shown in [28] that a single-server PIR scheme may in fact be designed with $o(n)$ communication, in particular, $\mathcal{O}(n^\epsilon)$ communication for any $\epsilon > 0$ under the quadratic residuosity assumption. This was recently improved to $\mathcal{O}(\text{polylog}(n))$

communication overhead under the so-called Φ-hiding assumption [10]. A number of variant PIR schemes have been proposed in the literature, such as symmetric PIR (SPIR) schemes, which include the additional property that the client sees only the data it has requested [20], and a variant with auxiliary servers [19]. None of these proposed PIR schemes, however, is practical for wide scale deployment. Even the scheme in [10] requires on average roughly $n/2$ exponentiations by the server per transmitted bit. For example, to service 100 users requesting ads from a (small) database consisting of, say, 10 ads of size 1k bytes, the server needs to perform roughly 4,000,000 exponentiations.

In this paper, we consider a practical alternative to these proposed PIR schemes. To obtain improved communications and computational efficiency, we consider two relaxations of the common security model. First, in lieu of a single server (Alice), or auxilliary servers, we assume a collection of communicating servers among which a majority behave in an honest fashion. We refer to this as a *threshold* PIR scheme. In principle, it is possible to achieve a threshold PIR (or even SPIR) scheme with optimal client-to-server communication using general secure multiparty computation, as introduced in [22]. In this paper, we demonstrate a threshold PIR scheme that does not require this very costly general apparatus, and instead achieves greater efficiency through reliance a mix network. Our threshold PIR scheme is capable of achieving server-to-client communication overhead of $\mathcal{O}(1)$ per consumer request under appropriate cryptographic assumptions. (This is optimal, of course.) As a second, additional relaxation, we consider a scenario in which requests from a large number of users may be batched. In this case, it is acceptable for servers to learn what has been requested, but not by whom. In other words, in consonance with the Crowds principle, we permit full disclosure of aggregate information, but hide information regarding the requests of individual users. We refer to a threshold PIR scheme with this latter property as a *semi-private* PIR scheme. A semi-private PIR scheme, in addition to achieving communication overhead of $\mathcal{O}(1)$, is computationally quite efficient, involving $\mathcal{O}(1)$ basic cryptographic operations per item per server.[1]

The negotiant approach we propose in this paper is not necessarily meant as a substitute for proxy servers, Crowds, or P3P. It may instead be viewed as a complementary technology, deployable in conjunction with any of these other ideas. Moreover, any of a range of tradeoffs between efficiency and security may be used in the construction of a negotiant function. We show this by presenting in this paper not one, but four different negotiant schemes.

1.2 Organization

In Section 2, we describe the cryptographic primitives used in our more advanced negotiant protocols. We also formalize the model in which we propose our schemes, and set forth basic definitions regarding privacy. In Section 3, we

[1] It is worth noting that both the threshold PIR scheme and the semi-private PIR scheme proposed here are in fact SPIR schemes. We do not make use of the special SPIR property in our schemes, however.

propose some negotiant function constructions. We consider some practical implementation issues in Section 4, and conclude in Section 5 with a brief discussion of some future avenues of investigation.

2 Preliminaries

2.1 Building blocks

Let us begin by introducing some of the cryptographic primitives used in the more advanced variants of our protocol. Readers familiar with the basic cryptographic literature may wish to skip to Section 2.2. Most of the protocols we describe are (t, m)-*threshold* protocols. These are protocols executed by a collection of servers $S_1, S_2, \ldots, S_m$, where $m \geq 1$, such that protocol privacy and the correctness of the output are ensured given an honest coalition of any t servers. In such protocols, servers hold a private key x in an appropriate distributed fashion, with a corresponding published public key $y = g^x$. It is common to use the Pedersen protocol [35, 34] as a basis for distributed key generation, although see [18] for a caveat. We do not discuss key generation or related details here.

El Gamal cryptosystem: Where we require public-key cryptography, we employ the El Gamal cryptosystem [15, 17]. Encryption in this scheme takes place over a group G_q of prime order q. Typically, G_q is taken to be a subgroup of Z_p^*, where $q \mid (p-1)$. Alternatives are possible; for example, G_q may be the group of points of an elliptic curve over a finite field.[2]

Let g be a generator of G_q. This generator is typically regarded as a system parameter, since it may be used in multiple key pairs. The private encryption key consists of an integer $x \in_U Z_q$, where $\in_U$ denotes uniform random selection. The corresponding public key is defined to be $y = g^x$. To encrypt a message $M \in G_q$, the sender selects $z \in_U Z_q$, and computes the ciphertext $(\alpha, \beta) = (My^z, g^z)$, where it may be seen that $\alpha, \beta \in G_q$. To decrypt this ciphertext using the private key x, the receiver computes $\alpha/\beta^x = My^z/(g^z)^x = M$. The El Gamal cryptosystem is *semantically secure* under the Decision Diffie-Hellman (DDH) assumption over G_q [41].

Let $(\alpha_0, \beta_0) \otimes (\alpha_1, \beta_1) \equiv (\alpha_0\alpha_1, \beta_0\beta_1)$. A useful feature of the El Gamal cryptosystem is its *homomorphism* under the operator $\otimes$. If (α_0, β_0) and (α_1, β_1) represent ciphertexts for plaintexts M_0 and M_1 respectively, then $(\alpha_0, \beta_0) \otimes (\alpha_1, \beta_1)$ represents an encryption of the plaintext M_0M_1. It is also possible, using knowledge of the public key alone, to derive a random *re-encryption* (α', β') of a given ciphertext (α, β). This is accomplished by computing $(\alpha', \beta') = (\alpha, \beta) \otimes (\gamma, \delta)$, where (γ, δ) represents an encryption of the plaintext value 1. It is possible to prove quite efficiently in zero-knowledge that (α', β') represents a valid re-encryption of (α, β) using, e.g., a variant of the Schnorr proof-of-knowledge protocol [39]. This proof may also be made non-interactive using the Fiat-Shamir

[2] Most commonly, we let $p = 2q + 1$, and we let G_q be the set of quadratic residues in Z_p^*. Plaintexts not in G_q can be mapped onto G_q by appropriate forcing of the Legendre symbol, e.g., by multiplication with a predetermined non-residue.

heuristic [16]. In this latter case, soundness depends on the random oracle model, while communication costs are $\mathcal{O}(1)$ group elements and computational costs are $\mathcal{O}(1)$ modular exponentiations. See, e.g., [11] for an overview.

Quorum-controlled asymmetric proxy re-encryption: This is a threshold algorithm enabling an El Gamal ciphertext encrypted under public key y to be re-encrypted under a new public key y'. Input to the protocol is an El Gamal public key y', as well as a ciphertext $(\alpha, \beta) = E_y[M]$. The output of the protocol is $(\alpha', \beta') = E_{y'}[M]$. While is assumed that the servers share the private key corresponding to y, they do not necessarily have any knowledge of the private key for y'. Jakobsson [25] proposes a protocol that is computationally secure in the sense that it is robust against any adversary controlling any minority coalition of cheating servers, and also preserves the privacy against such an adversary. Additionally, the protocol is efficient in a practical sense. Assuming use of non-interactive proofs, robustness depends on the random oracle model, while privacy depends only on the DDH assumption for the underlying cipher. Computational costs are $\mathcal{O}(m)$ modular exponentiations per server, while the broadcast communication complexity is $\mathcal{O}(m)$ group elements.

Distributed plaintext equality test: This is a threshold protocol described in [27]. Given El Gamal ciphertexts (α, β) and (α', β'), a collection of servers determines whether the underlying plaintexts are identical. Each server in turn commits to a blinding of the publicly computable ciphertext $(\gamma, \delta) = (\alpha/\alpha', \beta/\beta')$ by raising both integers in the pair to a common random exponent. All servers then decommit and prove their blindings correct non-interactively. The resulting combined, blinded ciphertext is jointly decrypted by the servers, yielding the value 1 if the underlying plaintexts are equivalent, and a random value otherwise. We write $(\alpha, \beta) \approx (\alpha', \beta')$ to denote equality of underlying plaintexts in the two ciphertexts (α, β) and (α', β'). The scheme is robust against any minority coalition in the random oracle model. Computational costs are $\mathcal{O}(m)$ modular exponentiations per server; the broadcast complexity is $\mathcal{O}(m)$ group elements.

Bulletin Board: Our proposed schemes with multiple players or servers assume the availability of a *bulletin board*. This may be viewed as a piece of memory which any player may view or add a new entry to, but which no player may edit or erase any portion of. A bulletin board may be realized as a public broadcast channel, or is achievable through Byzantine agreement (under the assumption that an attacker controls at most $\lfloor m/3 \rfloor$ servers) [29], or some appropriate physical assumption. Postings to a bulletin board may be made authenticable, i.e., their source may be securely validated, through use of such mechanisms as digital signatures. In many cases, our proposed algorithms only require bulletin board access by servers, not by other players.

Mix networks: A critical building block in our protocols is a threshold algorithm known as a *mix network*. Let $E_y[M]$ represent the encryption under public key y of message M in a probabilistic public-key cryptosystem, typically El Gamal.

This notation is informal, in the sense that it does not take into account the random encryption exponent that causes two encryptions of the same plaintext to appear different from one another. While we retain this notation for simplicity, the reader must bear it in mind, particularly with regard to the fact that mix networks involve re-encryption of ciphertexts.

A mix network takes as input a vector of ciphertexts denoted by $V = \{E_y[M_1], E_y[M_2], \ldots, E_y[M_n]\}$. Output from the mix network is the vector $V' = \{E_y[M_{\sigma(1)}], E_y[M_{\sigma(2)}], \ldots, E_y[M_{\sigma(n)}]\}$, where σ is a random permutation on n elements. A mix scheme is said to be *robust* if, given a static adversary with active control of a minority coalition of servers, V' represents a valid permutation and re-encryption of ciphertexts in V with overwhelming probability. A mix scheme is said to be *private* if, given valid output V', for any $i \in \{1, 2, \ldots, n\}$, an adversary with active control of a minority coalition and passive control of at most $m - 1$ servers cannot determine $\sigma^{-1}(i)$ with probability non-negligibly larger than $1/n$ (assuming unique plaintexts). It should be noted that to prevent attacks in which some players post re-encryptions of other players' inputs, it is often a requirement that input be encrypted in a manner that is *plaintext aware*. For this, it suffices that a player presenting El Gamal ciphertext (α, β) also provide a zero-knowledge proof of knowledge of $\log_g \beta$, and that servers check the correctness of this proof. See, e.g., [24] for further details.

Mix servers were introduced by Chaum [12] as a basic primitive for privacy. In his simple formulation, each server S_i takes the output V_i of the previous server and simply permutes and re-encrypts the ciphertexts therein. A security caveat for this scheme was noted in [36]. While the Chaum scheme and related proposals are private, they are not robust. A number of robust, threshold mix networks have appeared in the literature [6, 7, 14, 23, 24, 26, 30, 31]. The most efficient to date is the flash mixing proposal of Jakobsson [24]. Mitomo and Kurosawa [30] recently discovered a security flaw, for which they propose a very efficient remedy.

Robustness is in general not of critical importance in the schemes proposed here, as a server corrupting the computation can at best insert a false or incorrect advertisement, something likely to be detected if widespread. On the other hand, our scheme has two additional requirements. First, we must make use of a mix network that converts plaintexts into ciphertexts, not the reverse, as is usual in the literature. Second, input elements in our scheme, namely ads, are likely to be long. Robust mix networks are typically inefficient in such cases. (A recent scheme of Ohkubo and Abe [32] may be viewed as an exception, although that scheme requires a number of servers quadratic in the number of tolerable malicious servers.) For these two reasons, we propose a special plaintext-to-ciphertext variant on the basic Chaumian mix network in Section 4.2.

There are many variations on mix networks. For example, there are efficient mix networks in which V is a vector of tuples of ciphertexts. Additionally, a mix network may take ciphertexts and/or plaintexts as inputs and likewise output a combination of plaintexts and ciphertexts as desired. We employ a variety of such operations in our protocols, and omit implementation details.

2.2 Model and definitions for our scheme

Let $C_1, C_2, \ldots, C_k$ be a collection of consumers toward whom advertisements are to be directed. Let $P_1, P_2, \ldots, P_k$ be the respective profiles of these consumers. These profiles may contain a variety of information on the consumer, including standard demographic information such as age, sex, annual income, etc., as well as other information such as recently visited URLs and search engine queries. Let us designate the set of possible consumer profiles by $\mathcal{P}$. We denote the advertiser by A, and let $AD = \{ad_1, ad_2, \ldots, ad_n\}$ be the set of advertisements that A seeks to distribute. The advertiser chooses a *negotiant function* $f_{AD} : \mathcal{P} \to \{1, 2, \ldots, n\}$, which may be either deterministic or probabilistic. This function takes the profile of a consumer as input and outputs a choice of advertisement from AD to direct at the consumer. It is important to note that f_{AD} need not take AD explicitly as input, even if its output is indirectly dependent on AD. As an example, f_{AD} might derive a list of the most common words in URLs visited by the user and seek to match these to text descriptors associated with the ads in AD. We assume that the set AD is consistent from user to user (an assumption we revisit later in the paper). Thus, in most cases, we write f for clarity of notation, leaving the subscript implicit. Of course, it is possible to extend our definition of f to include inputs other than user profiles, such as the current date, or the list of advertisements already sent to the consumer; we do not consider such extensions in this paper, however. We assume that A is represented by a set of servers $S_1, S_2, \ldots, S_m$, for $m \geq 1$. These servers share a bulletin board, to which all consumers post their ad requests. When enough ads have accumulated or some other triggering criterion occurs (as discussed in Section 4.2), servers perform any necessary computation and then initiate communication with consumers and dispense ads to them.

Let l be an appropriately defined security parameter. We say that a function $q(l)$ is *negligible* if for any polynomial p, there exists a value d such that for $l \geq d$, we have $q(l) < 1/|p(l)|$. Otherwise, we say that q is *non-negligible*. We say that probability $q(l)$ is *overwhelming* if $1 - q(l)$ is negligible.

Let A_1 be a static polynomial-time adversary that actively controls a set of t servers and has knowledge of f and AD. Consider the following experiment. Assume that A_1 does not control consumer C_i. A_1 chooses a pair of profiles $(\tilde{P}_0, \tilde{P}_1) \in \mathcal{P}^2$. A bit $b \in_U \{0, 1\}$ is selected at random and P_i is set to $\tilde{P}_b$. Now the protocol is executed, and A_1 outputs a guess for b. We say that the protocol has (t, m)-*privacy* if for any such adversary A_1, it is the case that $\mathsf{pr}[A_1 \text{ outputs } b] - 1/2$ is negligible, where the probability is taken over the coin flips of all participants. This definition states informally that the protocol transcript reveals no significant information about P_i, even if all other consumers are in the control of A_1.

Now let us modify the experiment slightly and consider a polynomial-time adversary A_1 that controls t servers, but no consumers. This adversary selects a pair of distinct profile assignments $(\boldsymbol{P}_0, \boldsymbol{P}_1) \in (\mathcal{P}^k)^2$ for the k players such that both profile assignments yield the same set of ad requests. A bit $b \in_U \{0, 1\}$ is selected at random, and the profile set $\boldsymbol{P}_b$ is assigned to the players. We say

that a negotiant protocol has (t, m)-*group privacy* if for any such adversary A_1, it is the case that $\mathsf{pr}[A_1 \text{ outputs } b] - 1/2$ is negligible. The property of group privacy means, roughly stated, that an advertiser can learn only the aggregate ad requests of a group of consumers, but no further information about individual profiles. We refer to the special case of $(1, 1)$-group privacy as *profile privacy*. This limited but still valuable form of privacy means that an advertiser learns the ad request of a given consumer C_i, but no additional information about P_i.

We say that a negotiant protocol is *aggregate transparent* if any server can determine the set $\{f(P_1), f(P_2), \ldots, f(P_k)\}$ – in an unknown, random order – with overwhelming probability. In real-world advertising scenarios, it is important that a protocol be aggregate transparent, as the clients of advertisers typically wish to know how many times their ads have been displayed.

The final property we consider is that of *robustness*. Roughly stated, we say that a targeted advertising protocol is *robust* if, given a static, polynomial-time adversary that controls a minority coalition of servers, every consumer C_i receives $f(P_i)$ with overwhelming probability. In other words, the adversary is incapable of altering or making substitutions for the ads requested by consumers.

3 Some Negotiant Schemes

We now present several schemes representing a small spectrum of tradeoffs between security properties and resource costs. In measuring asymptotic communication costs, we regard a single ad as the basic unit of communication, and assume that ciphertext lengths and security parameter l are $O(|q|)$.

3.1 Scheme 1: Naïve PIR scheme

We present this simple scheme as a conceptual introduction. Here, requests are directed from a single consumer C with profile P to a single server S. (Thus the scheme may be modeled by $m = k = 1$.) The scheme is this: The server sends all of AD to C, who then views $ad_{f(P)}$.

Clearly, this scheme enjoys full privacy, that is, (m, m)-privacy, and is robust. The chief drawback is the $\Theta(n)$ communication cost. Another drawback is the fact that the scheme is not aggregate transparent. Nonetheless, given a limited base of advertisements and good bandwidth, and if advertisers are satisfied with recording click-through rates, this scheme may be useable in certain practical scenarios. In fact, more or less exactly this scheme serves as the basis of product known as an Illuminated Statement™ offered by a company called Encirq [2].

3.2 Scheme 2: Direct request scheme

This is another conceptually simple scheme involving a one-on-one consumer and server interaction. In this scheme, C simply sends $f(P)$ to S, who returns $ad_{f(P)}$. This scheme enjoys profile privacy and has communication and computation overhead $\mathcal{O}(1)$. It is also robust. Despite (or because of) its simplicity, it may in

many cases be appealing from a practical standpoint. Recall that profile privacy may be regarded as a form of $(1, 1)$-group privacy. In the next scheme, we show how to achieve stronger group privacy.

3.3 Scheme 3: Semi-private PIR scheme

We now show how to invoke some of the cryptographic apparatus described above in order to achieve a semi-private PIR scheme useable as the basis for a negotiant scheme. Given database $AD = \{ad_1, ad_2, \ldots, ad_n\}$, the goal is for a collection of consumers $C_1, C_2, \ldots, C_k$ to retrieve respective elements $ad_{r_1}, ad_{r_2}, \ldots, ad_{r_k}$ in such a way that the database servers learn requests only in the aggregate. Of course, our aim here is to apply this scheme to the retrieval of advertisements, and we shall present it in this context. In other words, we assume that $r_i = f(P_i)$. As above, we assume a public/private El Gamal key pair (y, x) held in an appropriate distributed manner by servers $S_1, S_2, \ldots, S_m$. We also assume that each consumer C_i has a public/private El Gamal key pair (y_{C_i}, x_{C_i}). Finally, for simplicity of presentation, we assume that an ad may be encrypted as a single El Gamal ciphertext. (In a real-world setting, an ad would have to be encrypted across multiple ciphertexts. We treat this issue further and propose a more practical alternative in Section 4.2.) The scheme is as follows.

1. Each consumer C_i computes $r_i = f(P_i)$ and posts the pair $(E_y[r_i], i)$ to the bulletin board. Let $V_1 = \{E_y[r_i], i\}_{i=1}^k$ be a vector of ciphertext/plaintext pairs accumulated when all consumers have posted their requests.
2. Servers apply a mix network to V_1 to obtain V_2. This mix network encrypts first column elements and simultaneously decrypts second column elements. Thus V_2 is a vector of pairs $\{(r_{\sigma_1(i)}, E_y[\sigma_1(i)])\}_{i=1}^k$ for random, secret permutation σ_1.
3. Servers replace each integer r_j in V_2 with ad_{r_j}. Call the resulting vector V_2'.
4. Servers apply a mix network to V_2' to obtain a vector V_3, where V_3 is a vector of pairs $\{(E_y[ad_{r_{\sigma_2(i)}}], \sigma_2(i))\}_{i=1}^k$, and σ_2 is an random, secret permutation.
5. Let $(E_y[ad_{r_i}], i)$ be an element in V_3. For each pair, the servers apply quorum-controlled asymmetric proxy re-encryption to obtain $(E_{y_{C_i}}[ad_{r_i}], i)$. Let the resulting vector be V_4.
6. For each element $(E_{y_{C_i}}[ad_{r_i}], i)$ in V_4, the servers send $E_{y_{C_i}}[ad_{r_i}]$ to C_i.
7. Consumers decrypt their respective ciphertexts to obtain their ads.

The security of the scheme is predicated on that of the underlying mix network. If we use a threshold mix network such that proposed in [24] (with the caveat from [30]), it may be shown that this is a semi-private PIR scheme, with $(\lfloor m/2 \rfloor, m)$-group privacy, relative to the DDH assumption. In other words, the scheme retains group privacy against an adversary controlling a minority coalition of servers. Scheme 3 may also be shown to be robust in this case relative to the discrete log problem in the random oracle model. As exponentiation in G_q incurs cost $\mathcal{O}(l^3)$, the computational costs of the scheme are $\mathcal{O}(ml^3)$ per element per server. The communication overhead of the scheme is $\mathcal{O}(1)$. With appropriate

implementation enhancements, some of which we discuss in Section 4, we believe that this scheme may be deployed in a highly practical manner. For instance, to draw again on our example above, for 100 users requesting one of 10 ads, each of size 1k bytes, and with three servers, the total per-server computational cost would be very roughly 50,000 exponentiations in our scheme, as opposed to 4,000,000 exponentiations for the single server in [10]. (This estimate assumes use of the mix network proposed in [7, 26], as is best for small groups of users. With use of the mix network described in [24], the per-server computational cost is substantially lower for large groups of users. By using our proposal in Section 4.2, we can do much better still.)

3.4 Scheme 4: Threshold PIR

The semi-private PIR scheme described above can be converted into a threshold PIR scheme with a few extra steps, and at the expense of additional computational overhead. The idea is to perform a blind lookup of consumer ad requests. This is accomplished by mixing ads and then invoking the distributed plaintext equality test described in Section 2.1. The construction is such that processing consumer requests one at a time is no less efficient as processing many simultaneously. We therefore present the protocol as applied to a single consumer C with profile P and private/public key pair (y_C, x_C). Consumer C computes $r = f(P)$ and posts $E_y[r]$ to the bulletin board. The protocol is then as follows.

1. Servers construct a vector U_1 of ads, in particular, of pairs $\{(j, E_y[ad_j])\}_{j=1}^{n}$.
2. Servers mix U_1 to obtain a vector U_2 of the form $(E_y[\sigma(j)], E_y[ad_{\sigma(j)}])$ for a random, secret permutation σ.
3. For each j, the servers perform a distributed plaintext equality test to see whether $E_y[j] \approx E_y[r]$. Assuming correct protocol execution, when a match is found, this indicates the ciphertext pair $(E_y[r], E_y[ad_r])$.
4. The servers apply quorum-controlled asymmetric proxy re-encryption to obtain $E_{y_C}[ad_r]$. They send this to C.
5. C decrypts $E_{y_C}[ad_r]$ to obtain ad_r.

Assuming use of a threshold mix network, this scheme enjoys $(\lfloor m/2 \rfloor, m)$-privacy under the DDH assumption. It is also in this case robust given the discrete log assumption and the random oracle model. The communication overhead is $\mathcal{O}(1)$. The computational complexity for each server is is $\mathcal{O}(mnl^3)$ with use of the [24] construction, while the computational complexity for the client is $\mathcal{O}(l^3)$. Note that the bulk of the computational effort in this scheme occurs in step 2, in which a vector of ads must be mixed for every user. This step is not consumer-specific, and may be performed offline, or even by a different set of servers than that responsible for executing steps 3 and 4.

Perhaps the most suitable basis in the literature for comparison is the single-server, computationally secure scheme proposed in [10]. That scheme has communication complexity $\mathcal{O}(\text{polylog } n)$, server complexity proportional to $n \log q$ times a polynomial in l. (The multiplicative factor of $\log q$ as opposed to m imposes high overhead in practice, and the polynomial in l is $\Omega(l^3)$.) The per-client computational complexity is polynomial in l (again $\Omega(l^3)$), $\log n$, and $\log q$.

4 Security and Implementation Issues

4.1 Attacks outside the model

We have offered cryptographically based characterizations of the security of our schemes according to the definitions in Section 2.2. We see that for Schemes 2 and 3, an attacker in control of a minority coalition of servers can learn little beyond individual or aggregate ad requests. As mentioned above, however, even with these security guarantees an advertiser with full control of the negotiant function f can manipulate it so as to extract detailed profile information from individual users. Let us suppose, for example, that an advertiser wishes, through Scheme 2, to learn the approximate annual household income in dollars of a given consumer C with profile P. The advertiser can construct a function f such that $f(P) = \lfloor I/10,000 \rfloor$, where I is the annual household income of the consumer. In fact, given enough latitutude in the distribution of the negotiant function to consumers, an advertiser can even defeat the aggregate security of Scheme 3. She may do this by distributing a function f that encodes the ID of a consumer in the output $f(P)$, or by distributing a different function f to each consumer. We propose several potentially complementary safeguards against such abuses.

- **Open source negotiant function:** The idea here is to allow easy reverse engineering of f by consumers or watchdog organizations. This may be done by requiring that f be encoded in a high level language such as Java, or even by providing user-friendly software tools for viewing the behavior of f. Consumers or organizations that deem f unduly invasive may refuse to receive ads or may lodge complaints. P3P mechanisms for mediation between consumers and Web sites might also come into play here.
- **Seal of approval:** The problem of verifying that f does not threaten consumer privacy is somewhat similar to the problem of verifying that executable code is not malicious. Thus, we may adopt an approach similar to the ActiveX system, which is used for verification of the safety of executable code [3]. An organization that believes a given piece of code to be safe applies a digital signature to it prior to distribution. If a user trusts the holder of the certificate supporting the signature, then she has some assurance about the safety of the code. We may adopt a similar approach to negotiant functions, allowing watchdog organizations to provide authenticable seals of approval.
- **Restricted negotiant language:** Another approach to protecting clients against malicious code is the so-called *sandbox* approach, adopted to some extent in Java virtual machines. The sandbox idea dictates that code be executable only in a protected environment, i.e., that the permissible set of instructions be restricted so as to guarantee safety to the client. In a loosely analogous fashion, we can create a "privacy safe" language for f. That is, we constrain f to execute on a virtual machine that restricts the forms of access it may gain to consumer profiles, so as to ensure against unfair data extraction by advertisers.

– **Consumer profile control:** The idea here is to permit the consumer to choose what portion of his or her profile to divulge to or conceal from f. P3P seems a natural platform to support this form of consumer control.
– **Controlled distribution of negotiant function:** To ensure against the advertiser extracting user data by customizing f, we wish to ensure that f is employed in a homogeneous fashion during a given time period or *distribution epoch*. One possible means of enforcement is to have a signed and time-stamped hash of f publicly posted by the advertiser, with some legal assurance of homogeneous distribution. Alternatively, f might be distributed by a semi-trusted site not directly associated with the advertiser. Of course, even if distribution of f is uniform, some users may not update their copies. Given distribution epochs of reasonably large duration, say, a week or a month, this should not be problematic.

Another possible attack by the advertiser involves collusion with consumers in Scheme 3 or creation of fictitious users. If, for example, the advertiser has access to $\{f(P_2), f(P_3), \ldots, f(P_k)\}$, then she can deduce $f(P_1)$ from the aggregated set of requests. The Crowds system suffers from a similar problem. The inventors of Crowds propose several countermeasures, for details of which we refer the reader to [1, 37, 38]. Since user anonymity is not required for privacy in our system, we can attach a substantial cost to the creation of fictitious users by, e.g., requiring that a consumer register by presenting a valid credit card or Social Security number. We should note, however, that the cost and trouble of mounting attacks involving widespread collusion or fraud, coupled with the small amount of information that such attacks are likely to reveal to the advertiser, should in most cases act as sufficient deterrents in and of themselves.

4.2 Practical implementation issues for Schemes 3 and 4

Aggregation and offline mixing: As mentioned above, mix networks involve computationally intensive cryptographic operations, and as such are not typically practical for applications in which mixing results must be produced on the fly. With the right type of setup, however, we can schedule the mixing operations in Schemes 3 and 4 so that execution may take place offline. The idea is that the first time a consumer C_i visits a Web site controlled by the advertiser, she submits $f(P_i)$. On this first visit, she does not receive the targeted ad $ad_{f(P_i)}$; she may instead receive a generic ad. In the interval of time between her first and second visits, however, her request $f(P_i)$ is aggregated with those of other consumers, and the ad servers perform the necessary mixing operations. On the second visit of C_i, then, her requested ad $ad_{f(P_i)}$ will be ready for her. She may at this point request another ad, to be ready on her third visit, and so on. In short, consumer ad requests may be pipelined in such a way that aggregation and processing takes place between visits, rather than during visits. Of course, it is possible to define an ad distribution epoch in any way that is convenient. For example, it may be that a consumer does receive a requested ad until the next day, with server mixing of ad requests taking place overnight.

This scheme for offline mixing may not work in the absence of multiple visits by a single user to the same site or to associated sites, or with the inability to recognize repeat visitors. In practice, however, most users frequently visit the same groups of sites repeatedly and do not shield their identities. This is reflected by, e.g., the still pervasive use of cookies on the Web, not to mention the extensive presence of DoubleClick.

Bulk encryption: We assume in our descriptions of Schemes 3 and 4 above that an advertisement may be represented as a single ciphertext. Of course, in reality, it is impractical to use ads small enough or a group G_q large enough to support this assumption. We may represent an advertisement as a sequence of associated ciphertexts, but this becomes computationally intensive for long ads. An alternative is to encrypt ads using an enveloping scheme involving both asymmetric and symmetric encryption. We describe a simple mix network of this type here, essentially a plaintext-to-ciphertext variant on the initial proposal of Chaum [12]. An important distinction, however, is that what we propose here involves use of the El Gamal cryptosystem and its re-encryption properties.

Let $\epsilon_\kappa[M]$ represent a symmetric-key encryption of plaintext M, where $\kappa \in_U K$ is a key drawn from keyspace K. We represent a full encryption of M for the mix network as $\tilde{E}_y[M] = (\gamma, \delta)$, where $\gamma = \{E_y[\kappa_1], E_y[\kappa_2], \ldots, E_y[\kappa_z]\}$ and $\delta = \epsilon_{\kappa_z}\epsilon_{\kappa_{z-1}} \ldots \epsilon_1[M]$ for some integer z. To re-encrypt $\tilde{E}_y[M]$ as (γ', δ'), a server does the following:

1. Re-encrypt all ciphertexts in γ.
2. Select $\kappa_{z+1} \in_U K$.
3. Append $E_y[\kappa_{z+1}]$ to γ to obtain γ'.
4. Compute δ' as $\epsilon_{\kappa_{z+1}}[\delta]$.

We leave further details of the mix network to the reader. There are two potential drawbacks to this scheme. First, the size of a ciphertext, as well as the computational cost of re-encryption, grows linearly in z, the number of re-encryptions. In practice, however, the performance is likely to be quite good, particularly when the number of mix servers m is small and ad sizes are large. A second drawback is the lack of robustness. As discussed above, however, robustness is a much less important consideration than privacy in our negotiant schemes. The incentive for a server to corrupt ads or substitute new ads is small, as such misbehavior would almost certainly become quickly apparent. Nonetheless, detection of tampering may be achieved by having servers include encrypted signatures of the symmetric keys they have generated, and formatting plaintexts such that it is easy to identify a correct decryption.

5 Conclusion

This paper seeks to convey two ideas, the first cryptographic and the second sociological. On the cryptographic side, we observe that by relaxing the conventional PIR model to allow for threshold and aggregate security properties, we

are able to achieve considerable practical improvements in terms of both communication and computational complexity. On the sociological side, we consider a new perspective on the contention between online advertisers and consumer privacy advocates. We explore some conceptually simple technical approaches to advertising that bring the objectives of both camps into closer alignment.

One of the main issues left unaddressed in this paper is how the negotiant function f should be constructed. Determining what features will be most effective in targeting advertisements is, of course, largely an advertising issue, and as such outside the scope of our investigations. The Encirq [2] system would seem to demonstrate that negotiant functions can be constructed that are effective and practical. The problem of formulating effective, adequately privacy-preserving negotiant functions f presents an open problem with interesting sociological and technical facets.

Acknowledgments

The author wishes to thank Markus Jakobsson and Burt Kaliski for their detailed comments and suggestions, as well as the anonymous referees of the paper.

References

1. Crowds homepage. AT&T Labs. http://www.research.att.com/projects/crowds.
2. Encirq, Inc. http://www.encirq.com.
3. Microsoft ActiveX resource page. Microsoft Corporation. http://www.microsoft.com/com/tech/ActiveX.asp.
4. Platform for privacy preferences (P3P) project. World Wide Web Consortium (W3C). http://www.w3.org/p3p.
5. Zero-Knowledge Systems, Inc. http://www.zeroknowledge.com.
6. M. Abe. Universally verifiable mix-net with verification work independent of the number of mix-servers. In *EUROCRYPT '98*, pages 437–447, 1998.
7. M. Abe. A mix-network on permutation networks. In *ASIACRYPT '99*, pages 258–273, 1999.
8. Reuters News Agency. DoubleClick awaits FTC OK: CEO says Web ad firm will wait for privacy policy before it uses ad tracking. 2 March 2000.
9. R.M. Arlien, B. Jai, M. Jakobsson, F. Monrose, and M. K. Reiter. Privacy-preserving global customization. In *ACM E-Commerce '00*, 2000. To appear.
10. C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *EUROCRYPT '99*, pages 402–414, 1999.
11. J. Camenisch and M. Michels. Proving that a number is the product of two safe primes. In *EUROCRYPT '99*, pages 107–122, 1999.
12. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
13. B. Chor, E. Kushilevitz, O. Goldreich, and M .Sudan. Private information retrieval. *JACM*, 45(6):965–981, 1998.
14. Y. Desmedt and K. Kurosawa. How to break a practical mix and design a new one. In *EUROCRYPT '00*, pages 557–572, 2000.

15. W. Diffie and M.E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, (22):644–654, 1976.
16. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *EUROCRYPT '86*, pages 186–194, 1986.
17. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.
18. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for d-log based cryptosystems. In *EUROCRYPT '99*, pages 295–310, 1999.
19. Y. Gertner, S. Goldwasser, and T. Malkin. A random server model for PIR. In *RANDOM '98*, pages 200–217, 1998.
20. Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. In *STOC '98*, pages 151–160, 1998.
21. J. Goldman, Z. Hudson, and R.M. Smith. Report on the privacy policies and practices of health Web sites, 2000.
22. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC '87*, pages 218–229, 1987.
23. M. Jakobsson. A practical mix. In *EUROCRYPT '98*, pages 448–461, 1998.
24. M. Jakobsson. Flash mixing. In *PODC '99*, pages 83–89, 1999.
25. M. Jakobsson. On quorum controlled asymmetric proxy re-encryption. In *PKC '99*, pages 112–121, 1999.
26. M. Jakobsson and A. Juels. Millimix: Mixing in small batches, 1999. DIMACS Technical Report 99-33.
27. M. Jakobsson and A. Juels. Mix and match: Secure function evaluation via ciphertexts. In *ASIACRYPT '00*, 2000. To appear.
28. E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *FOCS '97*, pages 364–373, 1997.
29. N. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1995.
30. M. Mitomo and K. Kurosawa. Attack for flash mix. In *ASIACRYPT '00*, 2000. To appear.
31. W. Ogata, K. Kurosawa, K. Sako, and K. Takatani. Fault tolerant anonymous channel. In *ICICS '97*, pages 440–444, 1997.
32. M. Ohkubo and M. Abe. A length-invariant hybrid mix. In *ASIACRYPT '00*, 2000. To appear.
33. R. Ostrovsky and V. Shoup. Private information storage. In *STOC '97*, pages 294–303, 1997.
34. T. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO '91*, pages 129–140, 1991.
35. T. Pedersen. A threshold cryptosystem without a trusted third party. In *EUROCRYPT '91*, pages 522–526, 1991.
36. A. Pfitzmann and B. Pfitzmann. How to break the direct RSA-implementation of MIXes. In *EUROCRYPT '89*, pages 373–381, 1989.
37. M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.
38. M. K. Reiter and A. D. Rubin. Anonymous Web transactions with Crowds. *Communications of the ACM*, 42(2):32–38, 1999.
39. C.P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4:161–174, 1991.
40. B. Tedeschi. E-commerce report; Critics press legal assault on tracking of Web users. *New York Times*. 7 February 2000.
41. Y. Tsiounis and M. Yung. On the security of ElGamal-based encryption. In *PKC '98*, pages 117–134, 1998.