Negotiated Privacy
(extended abstract)

Stanisław Jarecki[1], Patrick Lincoln[2], and Vitaly Shmatikov[2] *

[1] Computer Science Department, Stanford University, Stanford, CA 94305 U.S.A.
stasio@theory.stanford.edu

[2] SRI International, 333 Ravenswood Avenue, Menlo Park, CA 94025 U.S.A.
{lincoln,shmat}@csl.sri.com

Abstract. Exponential growth in digital information gathering, storage, and processing capabilities inexorably leads to conflict between well-intentioned government or commercial datamining, and fundamental privacy interests of individuals and organizations. This paper proposes a mechanism that provides cryptographic fetters on the mining of personal data, enabling efficient mining of previously-negotiated properties, but preventing any other uses of the protected personal data. Our approach does not rely on complete trust in the analysts to use the data appropriately, nor does it rely on incorruptible escrow agents. Instead, we propose conditional data escrow where the data generators, not the analysts, hold the keys to the data, but analysts can verify that the pre-negotiated queries are enabled. Our solution relies on verifiable, anonymous, and deterministic commitments which play the role of tags that mark encrypted entries in the analyst's database. The database owner cannot learn anything from the encrypted entries, or even verify his guess of the plaintext on which these entries are based. On the other hand, the verifiable and deterministic property ensures that the entries are marked with consistent tags, so that the database manager learns when the number of entries required to enable some query reaches the pre-negotiated threshold.

1 Introduction

Striking the right balance between security and privacy is a challenging task. Massive collection of personal data and ubiquitous monitoring of individuals and organizations is increasingly seen as a necessary step to thwart crime, fraud, and terrorism, ensure early detection of epidemics, monitor compliance with financial regulations, etc. Once the data are gathered, privacy laws and codes of conduct restrict how they can be mined and/or what queries may be performed on them. Searching patient records to find the source of an infectious disease is acceptable. Rifling through a neighbor's medical history is not. Development of

* Partially supported by ONR Grants N00014-02-1-0109 and N00014-01-1-0837, DARPA contract 9N66001-00-C-8015 and NSF grant INT98-15731.

a practical mechanism to ensure that the usage of collected data complies with the privacy policy is thus of paramount importance.

Simply declaring any form of personal data collection illegal is not a realistic option. In many situations—national security, law enforcement, financial supervision—massive monitoring is inevitable. Moreover, consumer incentive schemes such as grocery discount cards and frequent flyer programs have motivated many people to complain if detailed records are *not* kept of all their activities. It is, however, imperative to ensure that the monitoring process cannot be abused. The conventional approach places trust in the individuals and organizations who access the data. IRS workers are trusted not to take voyeuristic forays into famous actors' or their acquaintances' private tax information. Department of Motor Vehicles (DMV) workers are trusted not to provide unauthorized access to databases containing private address and other information. Such a system of trust is only as strong as its weakest link, which is often a low-level clerk.

Negotiated Privacy is a technical solution to this problem which is based on the concept of *personal data escrow*. The proposed approach relies on *self-reporting* of *self-encrypted* data by the subjects of monitoring. The data are reported in an encrypted escrowed form, which is verifiable, yet preserves privacy. This ensures both i) *accuracy*, *i.e.*, the subjects are prevented from cheating or lying about their activities, and ii) *privacy*, *i.e.*, the collectors of data are able to perform only authorized queries on the collected data. The properties of personal data escrow make all unauthorized queries on the collected data computationally infeasible. The most important feature of the proposed approach is that it does not require any trust to be placed either in the subjects of monitoring, or the collectors of data.

We assume that, before the monitoring system is initialized, the set of permitted queries is pre-negotiated between all stakeholders: legislators, monitoring agencies, privacy advocates, etc. This approach balances the need of the individuals to protect the privacy of their personal data, and that of the data collectors, such as law enforcement agencies, to obtain accurate information. In the simplest version of the scheme, the set of permitted queries is known to everybody involved. We believe that transparency is extremely important when collecting massive amounts of data about individuals and organizations. We expect that public and legislative support for any data collection scheme will be preceded by a consensus about the queries that the data collectors are permitted to evaluate.

2 Applications of Negotiated Privacy

Negotiated Privacy based on personal data escrow can be used in any context where the goal of data collection is to detect and reveal an "exceptional" combination of events, while keeping routine events completely private.

For example, consider a hypothetical Biological Warning and Communication System (BWACS) monitoring infrastructure for infectious diseases maintained by Centers for Disease Control and Prevention (CDC) Such a system is particularly important for rare infectious diseases, whose initial symptoms may be almost consistent with a common illness and thus often be misdiagnosed by the primary care physician, resulting in continuing spread of the epidemic unless detected early. To ensure early detection, the CDC may automatically collect all medical records from primary care facilities around the nation. Each record is escrowed so that no CDC staffer, even someone with unrestricted access to the BWACS database, can look up the details of a particular record at her whim. At the same time, certain queries, *e.g.*, "If there are 5 or more patients with a particular combination of symptoms treated in the same metropolitan area within 24 hours of each other, reveal some more information from records related to those patients," must be efficiently computable, without leaking *any* information about the patients who had different symptoms, or any information whatsoever if there were fewer than 5 patients that fit the specified pattern.

In another example, Food and Drug Administration (FDA) may analyze the progress of a new drug study by requiring that a record describing every patient's reaction be submitted in an escrowed form. The records cannot be viewed, unless there is a patient with more than 3 adverse reactions, in which case all records *pertaining to that patient* should be disclosed. Moreover, agents responsible for populating the monitoring database (*e.g.*, pharmaceutical companies conducting the drug trial) should not be able to conceal some of the records, or prevent the FDA from learning that the database contains an over-the-threshold number of records pertaining to the same patient and documenting adverse reactions.

In the national security arena, intelligence officials may be interested in learning "Has anyone flown to an airport within 100 miles of Kandahar 3 times in the last year?," but any information about individuals who flew fewer than 3 times, or flew elsewhere, should not be revealed. Another example is banking and finance, where fraud investigators may wish to look for specific repeated patterns of transactions, but individuals and organizations want to preserve the proprietary transaction details. Certain steps of auditing and even shareholder review of corporate finances could be enabled by personal data escrow, while preserving the privacy of transaction details.

As a last motivating example, consider the partially adversarial interests of a digital media user and a digital media platform provider. A user may wish to employ various digital artifacts such as computer software, music, video, etc. A provider of such valuable content may wish to restrict the use of that content. Peripherals or network devices such as video display boards, web services, and audio output devices play an essential role in the transactions of interest. A digital rights management system has the difficult task of providing some controls over the potential abuses of digital content (ie copying), while not providing the content provider intrusive access into the users entire system context and history. Negotiated privacy may allow a user to purchase the rights to use a piece of content or web service a limited number of times, and enable a analyst to determine if the limit has been exceeded.

3 What Negotiated Privacy Is NOT

Private information retrieval. The setting and the goals of negotiated privacy are different from those of private information retrieval (PIR) schemes [CGKS98] or symmetric PIR schemes [GIKM98]. In the PIR setting, the database manager is trusted to populate the database with correct entries, but he may be curious to learn which entries the client chooses to retrieve. Negotiated Privacy offers a means to privately *populate* databases while PIR offers a means to privately *retrieve* data from a database. However, techniques for oblivious polynomial evaluation [NP98,KY01] can be useful in the Negotiated Privacy setting for constructing an efficient threshold escrow protocol or for keeping the privacy disclosure conditions secret from the user.

Searching on encrypted data. There has been significant recent work on the problem of searching on encrypted data [SWP00,BO02]. Unlike general computation on encrypted data [AF88], the search problem has received efficient solutions for the case of symmetric [SWP00] and asymmetric [BO02] encryption schemes. In this scenario, the user gives to the server a database of ciphertexts, and then the user is able to efficiently search over this database for records containing specific plaintexts, or the user can enable the server itself to perform some restricted searches on this database. The encrypted data search problem does not have *verifiability* and *conditional secret release* properties needed in the Negotiated Privacy system. The encrypted search problem does not protect against a dishonest user, while in the Negotiated Privacy system the analyst needs to *verify* that the user's entries are formed from a correctly computed commitment and a ciphertext. Moreover, whereas in the work of [SWP00,BO02] the client could give a trapdoor to the database server which allows the server to identify some fields in the database, in the Negotiated Privacy scenario we need to support a conditional release of the plaintext of database fields to the server.

Digital cash. Digital cash schemes [CFN88] hold the promise of enabling transfer of value over digital media, and have been the subject of intense academic and commercial interest. Some digital cash schemes enable disclosure of information once a certain threshold is exceeded. For example, the user can remain anonymous as long as he spends each digital coin no more than once, but an attempt to double-spend results in disclosure of his identity. However, such schemes are insufficient for Negotiated Privacy because we also need to enforce that the user always uses the same coin when he escrows activities that are subject to the same privacy disclosure condition.

Privacy preserving datamining. Privacy preserving datamining focuses on deducing certain associations in a large database without revealing private information contained in the database. Two approaches to this problem include sanitizing the data prior to mining it [ERAG02] and splitting the data among multiple organizations while ensuring that no organization obtains private information about the other's data [LP00]. These problems are very different from the focus of this research. Negotiated Privacy is about controlling the conditions under which private information is exposed (*e.g.*, a passenger's ID is revealed after three trips to Kandahar). There is no datamining taking place since the type of information revealed is negotiated ahead of time.

Other related work. Extensions to the basic Negotiated Privacy scheme, such as threshold escrows with automatic disclosure, can employ (verifiable) secret sharing techniques [Sha79,Fel87]. General multi-party computation [Yao82,GMW87] and research on practical fault-tolerant distributed computation of various cryptographic functions [DF89] can also be instrumental in building efficient scalable solutions. Key escrow schemes [Mic92,KL95,YY98], blind signatures, anonymous credentials, and anonymous (or "key-private") public-key encryption are also closely related to Negotiated Privacy, but do not provide all the required properties [Cha82,KP98,Bra00,CL01,AR02,BBDP01].

4 Security Model

We describe the basic research problem underlying Negotiated Privacy, explain our model of privacy disclosure policies, the roles of the entities involved in the system, and the threat model. The *only* entity that must be trusted in order for Negotiated Privacy to work is the public-key infrastructure. The proposed approach is secure against threats presented by any other participant.

4.1 Basic problem of personal data escrow

Conditional data escrow. Consider for a moment just two parties, a *user* who performs some activities, and an *analyst* who wants to record all of the user's activities in his database. To balance the analyst's need to monitor all activities and the user's right to privacy, the records should be encoded in such a way that the analyst learns something useful from them if and only if the user's activities match some pre-specified suspicious pattern or *disclosure condition.* We call this conditional release of information *personal data escrow* because, by default, the owner of the database cannot decrypt the records,

A data escrow protocol is an example of a secure two-party computation. While, in principle, there are polynomial-time protocols for secure computation of any functionality that can be encoded as a polynomial-time algorithm [Yao82], including personal data escrow, their complexity depends at least linearly on the size of the logic circuit that implements this functionality, rendering them utterly impractical. Therefore, we propose techniques that *efficiently* solve the particular case of two-party computation needed for Negotiated Privacy: the personal data escrow problem.

Why "Negotiated Privacy"? We foresee a political or commercial process of *negotiation* aimed to balance the objective of protecting people's privacy against the objective to gather information, *e.g.*, for national security purposes. Negotiation would lead to the establishment of an acceptable trade-off, specifying exceptional patterns of activities that should trigger disclosure of personal records. For example, an acceptable disclosure policy, arrived at as a result of negotiation between all stakeholders, might specify that the only time individual medical records are disclosed to CDC is when more than 100 people in the same zip code exhibit certain disease symptoms. Similarly, commercial negotiation might lead to a policy specifying that the only time personal data of a software user are revealed to the software manufacturer is when the user installs the same copy of software on three or more different computers. Our research project, however, does not deal with the political, legal, organizational, or commercial aspects of this negotiation process. Instead, assuming that such negotiation will lead to the establishment of binding privacy disclosure conditions for some class of activities, we propose to investigate the technical means of implementing a monitoring system which ensures, based on strong cryptographic assumptions, that the established privacy disclosure conditions are adhered to.

4.2 Disclosure policies: predicates and privacy thresholds

Negotiated Privacy can support any disclosure policy that consists of setting a numerical *privacy threshold* for a certain class of events. The database of escrowed entries should satisfy the following property: the database owner only learns the plaintext of the entries that correspond to events which (1) belong to the same class, and (2) their number is no less than the pre-negotiated privacy threshold for this event class. Each permitted query must be of the following form: "if the database contains at least k entries, each of which satisfies a particular predicate P, reveal all entries satisfying P." A privacy disclosure condition is thus fully specified by the $\langle P, k \rangle$ pair. All of the examples outlined above can be implemented with threshold disclosure conditions. For example, an intelligence agency might establish the privacy threshold of 3 for events in the "itineraries with destination within 100 miles of Kandahar" class.

The disclosure predicate P may evaluate the user's activity at different levels of granularity. A patient's medical record might contain information relevant to a given query, such as blood pressure readings, but also irrelevant information, such as the social security number, marital status, etc. A passenger itinerary may include relevant fields, such as destination, and irrelevant fields, such as the name of the airline. Whether a particular field of the user's entry is relevant or not depends on the disclosure predicate. We will refer to the smallest subset of the user's entry that determines the value of the disclosure predicate as the *core* of that entry. Formally, given two entries t and t' and a predicate P, if $\mathsf{core}(t) = \mathsf{core}(t')$, then $P(t) = P(t')$.

4.3 Principals

Analyst. We use the term *analyst* for the owner of the database in which information about individuals, organizations, and their activities is collected. The role of the analyst can be played, for example, by a law enforcement or medical agency, government regulator, etc. The analyst issues a digitally signed credential or *receipt* for each escrowed datum he receives. The data escrow protocol must ensure that the analyst outputs a valid credential on a datum if and only if he receives a valid escrow of that datum. Secondly, the analyst should be able to efficiently evaluate on his database only the queries allowed by the pre-negotiated privacy disclosure policies.

User. A *user* is an individual or organization whose information is being collected in the analyst's database. We use the term *activity* to refer to the information that is of potential interest to the analyst. We assume that each activity is performed on the user's behalf by some service provider.

Service provider. A *service provider* is an agent who performs activities on users' behalf. For example, a commercial airline (service provider) transports a passenger (user) who has to report an escrow of his itinerary (activity) to an Office of Homeland Security database (analyst). Each service provider performs the requested service only if it is accompanied by a valid receipt issued by the analyst. An honest service provider should consider the receipt valid only if the user has previously submitted a correctly formed escrow to the analyst. We assume that the service provider knows the user's identity and the requested activity, but the user can choose which service provider he wants to receive services from. If the user wishes to protect his privacy, he is free to pick a provider he trusts not to reveal his activities to other parties. We consider the threats presented by malicious service providers below.

PKI/Magistrate. The magistrate is an independent official, trusted by all principals involved in Negotiated Privacy. The magistrate plays the role of a Public-Key Infrastructure (PKI), and can be replaced by a conventional PKI if one is available. The magistrate is employed only when the system is initialized and, in some variants of Negotiated Privacy, for dispute resolution.

4.4 Threat model

Since no assumptions are made about the trustworthiness of any of the principals except PKI, the Negotiated Privacy system must be secure against malicious misbehavior of any of the principals.

Malicious analyst. A malicious analyst should not be able to extract any information from the escrowed entries unless they satisfy one of the pre-negotiated disclosure conditions. Even then, the analyst should not learn the plaintext of the escrows that do not satisfy any such condition. If the analyst guesses the value of the plaintext on which the escrow is based, he should not be able to verify whether his guess is correct or not. Given two escrows, the analyst should not be able to determine whether they originate from the same user, or are based on the same plaintext.

Malicious user. A malicious user should not be able to obtain a valid receipt without submitting to the analyst a properly formed escrow. He also should not be able to prevent the analyst from efficiently recognizing that a subset of escrows accumulated in the analyst's database satisfies some disclosure condition, and can thus be feasibly opened.

Malicious service provider. If the service provider cooperates with the analyst against the user, no privacy can be guaranteed for the data that pass through that provider since the latter learns their contents in their entirety (*e.g.*, a passenger's itinerary cannot be hidden from the airline that transports this passenger). The analyst should be prevented, however, from learning the contents of *other* escrows, including those created by the same user and based on the same core plaintext (see lemma 4).

No data collection or monitoring scheme is feasible if the service provider cooperates with the *user* against the analyst and simply performs the requested activity without verifying whether the user has received a valid receipt from the analyst. If an airline is willing to transport passengers without any checks, then it is impossible to collect any reliable information about passenger itineraries. Therefore, our proposed scheme does not address this threat.

5 Cryptographic Toolkit

The Negotiated Privacy solution we propose is based on the assumption that computing so-called Decisional Diffie-Hellman (DDH) problem is intractable. Even though this assumption is not equivalent to the assumption of hardness of discrete logarithm computation, the two assumptions are related, and so far the only known way to compute DDH is to compute discrete logarithms. (See [Bon98] for a review of the literature on the hardness of computing DDH and on the use of this assumption in cryptographic literature.) The DDH problem is as follows. Let p and q be large primes and g be a generator of a subgroup $G_q = \{g^0, \ldots, g^{q-1}\}$ of order q of the multiplicative group $Z_p^* = \{1, \ldots, p-1\}$. In the DDH problem one is asked to distinguish between tuples of form $\{g, g^a, g^b, g^{ab}\}$ where a, b are distributed uniformly in Z_q, from tuples $\{g, g^a, g^b, g^c\}$ where a, b, c are distributed uniformly in Z_q. The DDH assumption is that the task of distinguishing between such tuples is intractable.

In our solution we assume the discrete-log setting given by the tuple (p, q, g) as above. We furthermore assume a hash function $\mathsf{hash} : \{0,1\}^* \rightarrow G_q$, which can be implemented with SHA-1 or MD5. In our analysis we will treat this hash function as an ideal hash function (see, *e.g.*, [BR93]). We also assume an a semantically secure symmetric encryption and a chosen-message-attack secure signature scheme.

5.1 First tool: anonymous commitment and encryption

Each user U has a private/public key pair (x, y). Suppose that plaintext t describes U's planned reportable activity (*e.g.*, airline travel). If $P(t)$ evaluates to

true, then the user must submit a personal data escrow $\lceil t \rceil_x$ to the analyst's database.

The escrow must satisfy several requirements. First, it must contain a tag which is a *commitment* to i) the user's identity, and ii) the core plaintext $\mathsf{core}(t)$, *i.e.*, the part that determines the value of the disclosure predicate P. The user should not be able to open this commitment with t' which has a different disclosure predicate, *i.e.*, such that $\mathsf{core}(t') \neq \mathsf{core}(t)$, or with x' that is different from the private key x that corresponds to the user's public key certificate issued by the magistrate. Moreover, this commitment must be *deterministic*, *i.e.*, the value of the commitment must be the same each time the user computes it for the given values of $\mathsf{core}(t)$ and x. The reason for this is that the analyst must be able to determine when the threshold number of escrows based on the same $\mathsf{core}(t)$ and x has been reached, in order to effect disclosure. Third, this commitment must be verifiable without the user revealing his private key x. Instead, the service provider must be able to verify that the commitment is computed correctly only knowing the user's public key y. Fourth, this commitment must protect anonymity of the user. Namely, given $\lceil t \rceil_x$, the analyst should not be able to determine if this escrow corresponds to some user U with public key y and/or some reportable activity t. The analyst should also not be able to determine if some escrows $\lceil t_i \rceil_{x_i}$ correspond to the same or different users or to the same reportable activities.

In addition to the tag which is a commitment to x and $\mathsf{core}(t)$, the escrow $\lceil t \rceil_x$ must also contain an encryption of the value t itself. When the required privacy threshold for events that meet some disclosure condition is reached, the analyst will ask the user to open the escrows whose tag matches this disclosure condition. Because this encryption must also be verifiable by the service provider, and it must protect the anonymity of the user, we implement both functionalities with the same tool of an anonymous one-way function which is unpredictable on random inputs.

Let hash be an ideal hash function and *Enc* a semantically secure symmetric encryption. Assuming the hardness of the DDH problem, the function we will use is $f_x(m) = m^x (\mathrm{mod} p)$. The encryption part of an escrow $\lceil t \rceil_x$ is a pair $s^x (\mathrm{mod} p)$, $Enc_s\{t\}$, where s is a randomly chosen element in G_q, and *Enc* is a semantically secure symmetric encryption scheme. Under DDH this is a semantically secure encryption scheme. The tag part of the $\lceil t \rceil_x$ escrow is a commitment to x and $\mathsf{core}(t)$ computed using the same function as $tag = h^x (\mathrm{mod} p)$, where $h = \mathsf{hash}(\mathsf{core}(t))$. Under DDH, both functionalities are *anonymous* in the sense that they cannot be matched to a public key $y = g^x (\mathrm{mod} p)$.

5.2 Second tool: verifiability via zero-knowledge

The encryption and commitment functions $(\mathsf{hash}(\mathsf{core}(t)))^x$ and s^x, $Enc_s\{t\}$ protect user's anonymity, but we need to provide, *e.g.*, to the service provider, the ability to verify that these functions were correctly computed without the user having to reveal his secret key x. Note that to prove that the value $z = h^x$ has been correctly computed, the user has to prove that the discrete logarithm

$DL_h(z) = DL_h(h^x) = x$ is equal to the discrete logarithm $DL_g(y) = DL_g(g^x) = x$. To do that, we use the non-interactive zero-knowledge proof of equality of discrete logarithms based on standard honest-verifier zero-knowledge proof due to Schnorr [Sch91]. (See Appendix A.)

Using this proof, the user who reveals t, s, and his public key $y = g^x$ to the service provider can prove that the tag part of $\lceil t \rceil_x$ is indeed equal to h^x, where $h = \mathsf{hash}(\mathsf{core}(t))$ can be computed by the service provider himself, and that the ciphertext part of $\lceil t \rceil_x$ is indeed a pair $(s^x, Enc_s\{t\})$.

6 Negotiated Privacy Database System

We describe the procedures for managing the Negotiated Privacy database for a single disclosure condition $\langle P, k \rangle$. The procedure can be easily generalized to multiple disclosure conditions by executing the steps described in section 6.2 with a separate key for each condition whenever a new escrow is added to the database.

Let U be the user, A the analyst and S the service provider. We assume that the disclosure conditions are publicly known, and therefore both U, A, S can derive the disclosure predicate P, the projection core, and the disclosure threshold k, for any activity t. We assume that the values of projection $\mathsf{core}(\cdot)$ are unique for every predicate P, *i.e.*, that for $P \neq P'$, for every t, t' such that $P(t)$ and $P'(t')$ are both true, $\mathsf{core}_P(t) \neq \mathsf{core}_{P'}(t')$. This can be easily achieved by appending a description $\langle P \rangle$ to $\mathsf{core}(t)$.

With Negotiated Privacy, the user encrypts the plaintext record t describing his activities, but does not reveal the record to the analyst unless the disclosure condition is satisfied. We assume that each t includes some verifiably fresh information that can be traced to the user, for example, the user's signature on the current time.

Although the analyst cannot verify that the record accurately describes the user's activity, he issues a digitally signed receipt for the user-submitted escrow. The service provider will require the user to open the analyst-signed encryption and verify that the record is indeed accurate and that the analyst's signature is fresh and valid.

In our solution, the analyst simply accumulates the encryptions and, once the disclosure threshold is reached, refuses to issue further receipts unless the user decrypts all of his previous encryptions together with the current one (and proves that he decrypted them correctly). From then on, this user will get the analyst's signature only if he reports his activity to the analyst in the clear.

6.1 Initialization

During initialization, the user creates a key pair $\langle x, y = g^x \rangle$, where g is some suitable base, x is the private key and the pair g^x, g is the public key. The user then approaches the magistrate, who, given g and $y = g^x$, verifies that the user indeed knows x (using, *e.g.*, Schnorr's authentication protocol [Sch91]) and issues

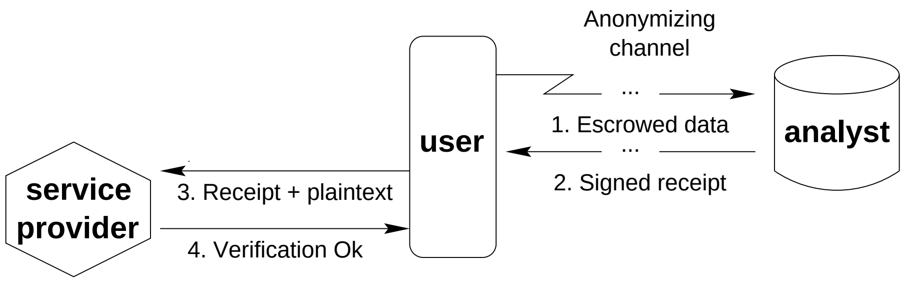

Fig. 1. Submitting and verifying a new escrow

a digitally signed certificate for the user's public key, binding the user's identity to the user's public key: $\mathsf{sig}_{K_M}(U, y)$. It is important that no more than one certificate is issued to each user. The user will need to present this certificate to service providers so that they can verify compliance as described in section 6.3 before performing a reportable activity.

The private key x and the corresponding public key certificate are the only values that the user has to remember to participate in Negotiated Privacy. The user-side software can, therefore, be implemented in constant memory and stored, for example, on a "trusted traveller" smartcard. Our scheme imposes fairly modest computational requirements on the user. In particular, the user does not need to obtain a special token from the magistrate for each activity.

6.2 Submitting a new escrow

The user generates the escrow $\lceil t \rceil_x = (tag, c, Enc_s\{t\}, k)$ consisting of the following parts:

1. $tag = h^x$ where $h = \mathsf{hash}(\mathsf{core}(t))$
2. $c = s^x$ where s is a fresh random element in G_q
3. $Enc_s\{t\}$, which is t encrypted using a symmetric cipher under a key derived by hashing s into the key space of the cipher
4. k, the anonymity threshold value

The user then sends this escrow to the analyst A. We assume that the user and the analyst communicate via an anonymizing channel (*e.g.*, an onion routing network [GRS99] or simply a public online bulletin board) that prevents the analyst from deducing the submitter's identity.

A keeps a database indexed by tag fields. After receiving the above escrow $\lceil t \rceil_x = (tag, c, Enc_s\{t\}, k)$, A checks that this escrow has not been submitted before. If it is fresh, the analyst checks whether the number of escrows in its database indexed with the same tag index is under $k - 1$. If it is larger, A requests that U open his escrow (see section 6.4). If it is equal to $k - 1$, A furthermore requests that U decrypts all of his previous ciphertexts of the other

$k-1$ escrows indexed by the same *tag* field. If the user complies or if there are fewer than $k-1$ escrows with this *tag* field in A's database, A issues the receipt by signing the entire received escrow. The procedure for submitting and verifying a new escrow is summarized in fig. 1.

6.3 Compliance checking

We rely on the service provider S to perform all the checks necessary to ensure that the escrow submitted by the user was is correctly formed and based on accurate information. We furthermore assume that the activity description t contains enough specific and timely details to guarantee that the same description cannot be used twice. While this approach requires that service providers be trusted to perform the checks, this is inevitable as discussed in section 4.4.

Before the service provider S provides a service or performs an activity on behalf of user U, S requests the record t describing the details of the activity from U and evaluates the disclosure predicate P. If $P(t)$ is true, then U should have submitted an escrow $\lceil t \rceil_x$ based on t to the analyst. S requests the following:

- $\mathsf{sig}_{K_A}(tag, c, Enc_s\{t\}, k)$, the digitally signed receipt from the analyst containing the entire escrow submitted by U;
- $\mathsf{sig}_{K_M}(U, y)$, the digitally signed certificate from the magistrate binding U's identity to U's public key y;
- the one-time key material s;
- a non-interactive zero-knowledge proof that $tag = h^x$, $c = s^x$, and $y = g^x$.

After receiving these items, S performs the following checks:

- Verify the magistrate's digital signature on the certificate $\mathsf{sig}_{K_M}(U, y)$ and the analyst's signature on $\mathsf{sig}_{K_A}(tag, c, Enc_s\{t\}, k)$.
- Compute the activity description t by decrypting the $Enc_s\{t\}$ ciphertext with a key derived by hashing the one-time key material s.
- Verify that the user's identity is indeed U, that the activity performed is adequately described by t (including the user's signature on a plausible time value), and that the anonymity threshold for this activity is indeed k.
- Given t, compute $h = \mathsf{hash}(\mathsf{core}(t))$, and verify the non-interactive zero-knowledge proof that $tag = h^x$, $c = s^x$, and $y = g^x$ (see section 5.2).

6.4 Disclosure

If the database contains $k-1$ escrows with the same index $tag = \mathsf{hash}(\mathsf{core}(t))^x$, the analyst refuses to issue a receipt for any user-submitted escrow with that index unless the user decrypts all his escrows associated with the index. Since an analyst-signed receipt is necessary to perform a reportable activity, this ensures proper disclosure. The user does not need to remember his old escrows. The analyst simply presents the list of escrows stored under the *tag* index back to the user and asks him to open them.

To open any escrow $\lceil t \rceil_x = (tag, c, Enc_s\{t\}, k)$, the user uses his private key x to compute $s = (c)^{\frac{1}{x}}$, computes t by decrypting $Enc_s\{t\}$ with the key derived from s, computes $h = \mathsf{hash}(\mathsf{core}(t))$, and sends to the analyst values h, s, his certificate $\mathsf{sig}_{K_M}(U, y)$, and a non-interactive zero-knowledge proof that $tag = h^x$, $c = s^x$, and $y = g^x$.

The analyst checks these values similarly to the verification procedure of the service provider described in section 6.3. In this process, the analyst learns the activity description t and the user's identity U.

6.5 Security properties of Negotiated Privacy

Assuming the hardness of the DDH problem, the following properties follow immediately in the random oracle model:

Lemma 1. *Given t and $h = \mathsf{hash}(\mathsf{core}(t))$ and the user's public key y, the adversary cannot distinguish $tag = h^x$ from a random element in G_q.*

Therefore, the analyst who guesses the value of t and the user's identity U, y, cannot verify this guess.

Lemma 2. *Given t, t' and $h = \mathsf{hash}(\mathsf{core}(t)), h' = \mathsf{hash}(\mathsf{core}'(t'))$ s.t. $h \neq h'$, the adversary cannot distinguish between pairs $h^x, (h')^x$ and pairs $h^x, (h')^{x'}$ for any pair of public keys $y = g^x, y' = g^{x'}$.*

Therefore, given two encrypted values, the analyst cannot determine whether they were generated by the same user or by two different users.

Lemma 3. *Given t, t' and $h = \mathsf{hash}(\mathsf{core}(t)), h' = \mathsf{hash}(\mathsf{core}'(t'))$ s.t. $h \neq h'$, the adversary cannot distinguish between pairs $h^x, h^{x'}$ and pairs $h^x, (h')^{x'}$ for any pair of public keys $y = g^x, y' = g^{x'}$ (including pairs s.t. $y = y'$).*

Therefore, given two encrypted values, the analyst cannot determine whether they encrypt the same or different plaintext, *i.e.*, the analyst cannot tell whether they were generated because of the same predicate or two different predicates.

Moreover, we can show the following:

Lemma 4. *If the service provider is cooperating with the analyst, the analyst learns nothing except*

- *Values t and U for all escrows verified by that provider.*
- *Associations between values tag and values U and $t^* = \mathsf{core}(t)$. This allows the analyst to tell how many reportable activities of type t^* are performed by user U. It does not leak, however, any more specific information about these activities.*

7 Conclusions

We presented a simple technique that provides an efficient and, assuming the Decisional Diffie-Hellman problem is intractable, provably secure way for a subject of monitoring to escrow his or her personal data in a Negotiated Privacy database and be assured that the database owner cannot extract any information from it. At the same time, the database owner is assured that the collected escrows accurately describe the monitored activities and, if one of the pre-negotiated disclosure conditions is satisfied, he will be able to open the relevant escrows and learn the details of the disclosed activities. In the future, we envision extending this approach to enable automatic opening of escrows once the privacy threshold is reached and to ensure privacy for subjects of monitoring even when corrupt service providers are cooperating with the analyst.

References

[AF88] M. Abadi and J. Feigenbaum. A simple protocol for secure circuit evaluation. In *Proc. STACS '88*, pages 264–272, 1988.

[AR02] M. Abadi and P. Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *J. Cryptology*, 15(2):103–127, 2002.

[BBDP01] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Proc. ASIACRYPT '01*, pages 566–582, 2001.

[BO02] D. Boneh and R. Ostrovsky. Search on encrypted data, 2002.

[Bon98] D. Boneh. The decisional Diffie-Hellman problem. In *Proc. 3rd Algorithmic Number Theory Symposium*, volume 1423 of *LNCS*, pages 48–63. Springer-Verlag, 1998.

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proc. ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[Bra00] S. Brands. *Rethinking Public Key Infrastructure and Digital Certificates – Building in Privacy.* MIT Press, Cambridge, MA, 2000.

[CFN88] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Proc. CRYPTO '88*, volume 403 of *LNCS*, pages 319–327. Springer-Verlag, 1988.

[CGKS98] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.

[Cha82] D. Chaum. Blind signatures for untracable payments. In *Proc. CRYPTO '82*, pages 199–203, 1982.

[CL01] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Proc. EUROCRYPT '01*, volume 2045 of *LNCS*, pages 93–118. Springer-Verlag, 2001.

[CP92] D. Chaum and T. Pedersen. Wallet databases with observers. In *Proc. CRYPTO '92*, volume 740 of *LNCS*, pages 89–105. Springer-Verlag, 1992.

[DF89] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Proc. CRYPTO '89*, volume 435 of *LNCS*, pages 307–315. Springer-Verlag, 1989.

[ERAG02] A. Evfimievski, R.Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proc. 8th ACM SIGKDD Int'l Conference on Knowledge Discovery in Databases and Data Mining*, 2002.

[Fel87] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proc. 28th IEEE Symposium on Foundations of Comp. Science*, pages 427–438, 1987.

[GIKM98] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private inforomation retrieval schemes. In *Proc. 31th Annual ACM Symposium on Theory of Computing*, pages 151–160, 1998.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with and honest majority. In *Proc. 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, 1987.

[GRS99] D. Goldschlag, M. Reed, and P. Syverson. Onion routing for anonymous and private Internet connections. *Communications of the ACM*, 42(2):39–41, 1999.

[KL95] J. Kilian and F.T. Leighton. Fair cryptosystems, revisited. In *Proc. EUROCRYPT '95*, volume 963 of *LNCS*, pages 208–220. Springer-Verlag, 1995.

[KP98] J. Kilian and E. Petrank. Identity escrow. In *Proc. CRYPTO '98*, volume 1462 of *LNCS*, pages 169–185. Springer-Verlag, 1998.

[KY01] A. Kiayias and M. Yung. Secure games with polynomial expressions. In *ICALP '01*, pages 939–950, 2001.

[LP00] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Proc. CRYPTO '00*, volume 1880 of *LNCS*, pages 36–47. Springer-Verlag, 2000.

[Mic92] S. Micali. Fair public-key cryptosystems. In *Proc. CRYPTO '92*, volume 740 of *LNCS*, pages 113–138. Springer-Verlag, 1992.

[NP98] M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Proc. 31th Annual ACM Symposium on Theory of Computing*, pages 245–254, 1998.

[Sch91] C.P. Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991.

[Sha79] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[SWP00] D.X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *Proc. IEEE Symposium on Security and Privacy*, pages 44–55, 2000.

[Yao82] A.C. Yao. Protocols for secure computations. In *Proc. 23rd IEEE Symposium on Foundations of Comp. Science*, pages 160–164, 1982.

[YY98] A. Young and M. Yung. Auto-recoverable and auto-certifiable cryptosystems. In *Proc. EUROCRYPT '98*, volume 1043 of *LNCS*, pages 17–31. Springer-Verlag, 1998.

A Zero-Knowledge Proof of Equality of Discrete Logs

To prove the equality of discrete logarithms, namely to prove that $DL_h(z) = DL_g(y)$, we use the following zero-knowledge proof due to Chaum and Pedersen [CP92]:

The prover's (user) secret input is x, a number in Z_q, and the verifier's (service provider) inputs are g, h, y, z, two numbers in G_q. To prove that $y = g^x$ and $z = h^x$, the prover picks a random number k in Z_q and sends $u = g^k$ and $v = h^k$ to the verifier. The verifier picks a random challenge e in Z_q and sends it to the prover. The prover sends back a response $f = k + ex$. The verifier accepts the interactive proof if $u = g^f/y^e$ and $v = h^f/z^e$. This protocol is zero-knowledge only for honest-verifiers, but in the random oracle model it can be converted to a non-interactive zero-knowledge proof by setting the challenge $e = H(g, h, y, z, u, v)$. In other words, the user generates the values k, u, v as above, computes $e = H(g, h, y, z, u, v)$ and $f = k + ex$, and sends to the service provider the pair (e, f) as the proof of discrete logarithms equality. To check this proof, the service provider computes $u = g^f/y^e$ and $v = h^f/z^e$ as above, and accepts if $e = H(g, h, y, z, u, v)$.

We note that this protocol can easily be extended to prove equality of three (or more) discrete logarithms. Moreover, it can be used as a way to authenticate the user by requiring that the challenge e is computed by hashing also a random authentication challenge sent to the user by the verifier, or by including in this hash the current time value.