OACerts: Oblivious Attribute Certificates [*]

Jiangtao Li and Ninghui Li

CERIAS and Department of Computer Science, Purdue University

{jtli, ninghui}@cs.purdue.edu

Abstract

We propose Oblivious Attribute Certificates (OACerts), an attribute certificate scheme in which a certificate holder can select which attributes to use and how to use them. In particular, a user can use attribute values stored in an OACert obliviously, i.e., the user obtains a service if and only if the attribute values satisfy the policy of the service provider, yet the service provider learns nothing about these attribute values. This way, the service provider's access control policy is enforced in an oblivious fashion.

To enable the oblivious access control using OACerts, we propose a new cryptographic primitive called Oblivious Commitment-Based Envelope (OCBE). In an OCBE scheme, Bob has an attribute value committed to Alice and Alice runs a protocol with Bob to send an envelope (encrypted message) to Bob such that: (1) Bob can open the envelope if and only if his committed attribute value satisfies a predicate chosen by Alice, (2) Alice learns nothing about Bob's attribute value. We develop provably secure and efficient OCBE protocols for the Pedersen commitment scheme and comparison predicates as well as logical combinations of them.

Keywords

Privacy, Access Control, Automated Trust Negotiation, Digital Certificate, Cryptographic Commitment, Cryptographic Protocol, Secure Function Evaluation, Oblivious Transfer

[*]A preliminary version of this paper was presented at the 3rd International Conference on Applied Cryptography and Network Security, New York, June 2005.

1 Introduction

In trust management and attribute-based access control systems [3, 40, 19, 11, 34, 33], access control decisions are based on attributes of requesters, which are established by digitally signed certificates. Each certificate associates a public key with the key holder's identity and/or attributes such as employer, group membership, credit card information, birth-date, citizenship, and so on. Because these certificates are digitally signed, they can serve to introduce strangers to one another without online contact with the attribute authorities.

In a typical scenario for accessing a resource using traditional digital certificates such as X.509 certificates [29], a requester Bob first sends his request to Alice who responds with the policy that governs access to that resource. If Bob's certificates satisfy Alice's policy, he sends the appropriate certificates to Alice. After Alice receives the certificates and verifies them, she grants Bob access to the resource. Observe that, in this scenario, Alice learns all the attribute information in Bob's certificates. Privacy is an important concern in the use of Internet and web services. When the attribute information in a certificate is sensitive, the certificate holder may want to disclose only the information that is absolutely necessary to obtain services. Consider the following example.

Example 1 A senior citizen Bob requests from a service provider Alice a document that can be accessed freely by senior citizens. Bob wants to use his digital driver license to prove that he is entitled to free access. Bob's digital driver license certificate has fields for an identification number, expiration date, name, address, birth-date, and so on; and Bob would like to reveal as little information as possible.

In the above example, it might seem that Bob needs to reveal at least the fact that he is a senior citizen, i.e., his birth-date is before a certain date. However, even this seemingly minimal amount of information disclosure can be avoided. Suppose that the document is encrypted under a key and the encrypted document is freely available to everyone. Further suppose a protocol exists such that after the protocol is executed between Alice and Bob, Bob obtains the key if and only if the birth-date in his driver license is before a certain date and Alice learns nothing about Bob's birth-date. Under these conditions, Alice can perform access control based on Bob's attribute values while being oblivious about Bob's attribute information.

We call this *oblivious access control*, because Alice's access control policies for her resources are enforced without Alice learning any information about Bob's certified attribute values, not even whether Bob satisfies her policy or not. To enable such oblivious access control, we propose Oblivious Attribute Certificates (OACerts), a scheme for using certificates to document sensitive attributes. The basic idea of OACerts is quite simple. Instead of storing attribute values directly in the certificates, a certificate authority (CA) stores the cryptographic commitments [38, 23, 12, 16] of these values in the certificates. Using OACerts, a user can select *which* attributes to use as well as *how* to use them. An attribute value in an OACert can be used in several ways: (1) by opening a commitment and revealing the attribute value, (2) by using zero-knowledge proof protocols [13, 36, 18, 5] to prove that the attribute value satisfies a condition without revealing other information, and (3) by running a protocol so that the user obtains a message only when the attribute value satisfies a condition, without revealing any information about the attribute value. The idea of storing cryptographic commitments of attribute values in certificates was used in anonymous credentials [10, 7, 35, 9, 8]; however, we are not aware of prior work on the oblivious usage of such attribute values.

In Example 1, suppose that the driver-license certificate that Bob has is an OACert. With attribute values committed rather than stored in the clear in her certificates, Bob can send his certificate to Alice without revealing his birth-date or any other attribute information. Using zero-knowledge proof protocols [13, 36, 18, 5], Bob can prove to Alice that his committed birth-date is before a certain date without revealing any other information. However, our goal is that Alice should learn nothing about Bob's birth-date, not even whether Bob is a senior citizen or not. To enable oblivious access control, we need to solve the following two-party Secure Function Evaluation (SFE) problem:

Problem 1 Let commit be a commitment algorithm, let Params be public parameters for commit, and Pred be a public predicate. Let a be a private number (Bob's attribute value), $c = \mathsf{commit}_{\mathsf{Params}}(a, r)$ be a commitment of a under the parameters Params with a random number r, and M be a private message (Alice wants Bob to see M if and only if a satisfies Pred). Alice and Bob jointly compute a family F of functions, parameterized by commit and Pred. Both parties have commit, Pred, Params, and c. Alice has private input M. Bob has private input a and r. The function F is defined as follows.

$$
\begin{aligned}
F[\mathsf{commit}, \mathsf{Pred}]_{Alice}(\mathsf{Params}, c, M, a, r) &= 0 \\
F[\mathsf{commit}, \mathsf{Pred}]_{Bob}(\mathsf{Params}, c, M, a, r) &= \begin{cases} M & \text{if } c = \mathsf{commit}_{\mathsf{Params}}(a, r) \wedge \mathsf{Pred}(a) = \mathsf{true}; \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}
$$

where $F[\mathsf{commit}, \mathsf{Pred}]_{Alice}$ represents Alice's output, $F[\mathsf{commit}, \mathsf{Pred}]_{Bob}$ represents Bob's output. In other words, our goal is that Alice learns nothing and Bob learns M only when his committed attribute value satisfies the predicate Pred.

The preceding problem can be solved using general solutions to two-party SFE [45, 26, 25]; however, the general solutions are inefficient, as commitment verification is done within the SFE. We propose an Oblivious Commitment Based Envelope (OCBE) scheme that solves the above two-party SFE problem efficiently. Formal definition of OCBE will be given in Section 4. Informally, an OCBE scheme enables a sender Alice to send an envelope (encrypted message) to a receiver Bob, such that Bob can open the envelope if and only if his committed value satisfies the predicate. An OCBE scheme is *oblivious* if at the end of the protocol the sender cannot learn any information about the receiver's committed value. An OCBE scheme is *secure against the receiver* if a receiver whose committed value does not satisfy the predicate cannot open the envelope.

We develop efficient OCBE protocols for the Pedersen commitment scheme [38] and six kinds of comparison predicates: $=, \neq, <, >, \leq, \geq$, as well as conjunctions and disjunctions of multiple predicates. These predicates seem to be the most useful ones for testing attribute values in access control policies. We present a protocol (called EQ-OCBE) for equality predicates and a protocol (called GE-OCBE) for greater-than-or-equal-to predicates and prove that these protocols are provably secure in the Random Oracle Model [2]. These protocols use cryptography hash functions to efficiently derive symmetric encryption keys from a shared secret, and random oracles are used to model such usage of hash functions. We also show that it is easy to construct OCBE protocols for other comparison predicates using variants of EQ-OCBE, GE-OCBE.

The contributions of this paper are as follows.

- We introduce the notion of OACerts and OCBE, which together enable oblivious access control. OACerts and OCBE may be of interests in other applications as well.
- We present efficient and provably secure OCBE protocols for the Pedersen commitment scheme [38] and several kinds of comparison predicates.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 presents the architecture and application of OACerts. Section 4 gives a formal definition of OCBE. Section 5 reviews the Pedersen commitment scheme. Section 6 presents several efficient and provably secure OCBE protocols. Section 7 describes our implementation and performance measurements. Section 8 concludes our paper.

2 Related Work

Recent works on using cryptographic protocols for certificate-based access control include Hidden Credentials [28, 6, 22], Secret Handshakes [1], and Oblivious Signature Based Envelope [31]. While these schemes are useful for the kinds of "ultra-sensitive" scenarios described in [28, 1], where policies are based on attributes such as secret clearance or memberships in some secret underground movements, they are not suitable for the kind of e-commerce scenarios such as Example 1, for the following reasons. Using any of these schemes, the service provider Alice could send an encrypted message to a client Bob such that Bob can decrypt if and only if he has certificates whose contents are the same as those identified by Alice's policy; at the same time, Alice does not know whether Bob has those certificates or not. (In Secret handshakes [1], Alice computes a key such that Bob can compute if and only if Bob has the required certificate.) These schemes can implement oblivious access control when Alice's policies have very specific forms. In Example 1, if Alice's policy is that Bob's birth-date is April 1st, 1974, then oblivious access control can be achieved using these existing schemes, as Alice could identify the contents of the certificates that would enable Bob to satisfy her policy. However, for the policy in Example 1 (birth-date in a certain range) where many possible attribute values would satisfy a policy, these schemes do not work well.

Our work is also closely related to anonymous credentials [10, 7, 35, 9, 8]. Indeed, the ideas of storing commitments of attribute values in certificates and using zero-knowledge proofs to prove properties of these values appeared in the literature on anonymous credentials, e.g. [7]. These schemes differ from OACerts in that they provide orthogonal privacy protections. None of the existing anonymous credential schemes enables oblivious access control as the verifier learns whether the prover satisfies her policy or not. On the other hand, anonymous credentials enable Bob to use a credential anonymously, i.e., Alice and other service providers cannot link together transactions in which Bob's credential is used. For such protection to make sense, anonymous communication channels are required. The OACerts scheme does not provide anonymity protection and therefore does not require anonymous communication channels. Furthermore, anonymous credential schemes tend to involve protocols dramatically different from existing public-key infrastructure standards. It is not clear how credential distribution and revocation are to be handled in these systems. On the other hand, the OACerts scheme is compatible with existing standards, such as X.509 [29].

Crescenzo et al. [15] introduced a variant of oblivious transfer called Conditional Oblivious Transfer (Conditional OT), in which Alice and Bob each has a private input and shares with each other a public predicate that is evaluated over the private inputs. In the conditional OT of a bit b from Alice to Bob, Bob receives the bit only when the predicate holds; furthermore, Alice learns nothing about Bob's private input or the output of the predicate. Crescenzo et al. [15] developed an efficient protocol for a special case of Conditional OT where the predicate is greater-than-or-equal-to. OCBE can be viewed as another special case of the Conditional OT problem; in which Alice has no private inputs, the commitment c of Bob's private input a is made public, and the public predicate for this Conditional OT is a conjunction of two conditions: (1) Bob's private input a must be the value he committed in c, and (2) Bob's private input a must also satisfy a predicate (e.g., greater-than-or-equal-to some value). The additional requirement of (1) makes OCBE quite different from Conditional OT for greater-than-or-equal-to predicate; therefore, the solution in [15] cannot apply to OCBE.

Crépeau [14] introduced the notion of Committed Oblivious Transfer (COT). In COT, Alice commits two bits: a_0 and a_1, and Bob commits a bit b. All three committed values are public knowledge.

The goal of COT is enable Bob to learn a_b without learning anything else, while Alice learns nothing. Garay et al. [24] gave an efficient construction of COT in the universal composability framework. OCBE differs from COT in that Bob's input in OCBE is an integer whereas Bob's input in COT is a single bit. Furthermore, because the predicate in OCBE could be arbitrary, results in COT cannot apply directly to our OCBE protocol.

Our work is related to zero-knowledge proof protocols [13, 36, 18, 5] that prove a committed value satisfies some property. Our GE-OCBE protocol has similarities with the range proof protocol in [36, 18], which proves that a committed value lies within a range. Also, the details of our GE-OCBE protocol are reminiscent of the techniques used in the oblivious transfer protocols [37, 41] and the comparison method for millionaires [21].

3 Architecture and Applications of OACerts and OCBE

In this section, we present the architecture of OACerts and OCBE and outline their applications.

3.1 Architecture

There are three kinds of parties in the OACerts scheme: certificate authorities (CA's), certificate holders, and service providers. A CA issues OACerts for certificate holders. Each CA and each certificate holder has a unique public-private key pair. A service provider, when providing services to a certificate holder, performs access control based on the attributes of the certificate holder, as certified in OACerts.

An OACert is a digitally signed assertion about the certificate holder by a CA. Each OACert contains one or more attributes. We use $attr_1, \ldots, attr_m$ to denote the m attribute names in an OACert, and $v_1, \ldots, v_m$ to denote the corresponding m attribute values. Let $c_i = \mathsf{commit}_{\mathsf{Params}}(v_i, r_i)$ be the commitment of attribute value v_i for $1 \leq i \leq m$ with r_i being the secret random number. The attribute part of the certificate consists of a list of m entries, each entry is a tuple $\langle attr_i, c_i \rangle$. When the commitment scheme used is secure, the certificate itself does not leak any information about the sensitive attributes. Thus, an OACert's content can be made public. A certificate holder can show his OACerts to others without worrying about the secrecy of his attributes.

In many commitment schemes [38, 23, 12], the input domain is the set of integers; hence it is necessary to map an arbitrary attribute value to an integer in OACerts. For example in a digital driver license, gender can be expressed by a single bit, state can be expressed by a number from $[1, 50]$, birthdate can be expressed by the number of days between January 1st of 1900 and the date of birth. In an another example, suppose a digital student certificate contains an attribute for major. As the number of different majors is finite (and quite small in practice), we can easily encode each major with a number. There are certain attributes of which the values could be arbitrary, such as name or home address. We cannot represent those attribute values directly with integers, in this case, the CA hashes the attribute values using a collision-free hash function and commit the hash values in OACerts.

OACerts can be implemented on existing public-key infrastructure standards, such as X.509 Public Key Infrastructure Certificate [4, 29] and X.509 Attribute Certificate [20]. The commitments can be stored in X.509v3 extension fields, in which case a certificate includes also the following fields: serial number, validity period, issuer name, user name, certificate holder's public key, and so on. The distribution and revocation of OACerts can be handled using existing infrastructure and techniques. See Section 7 for our implementation and performance measurements of OACerts.

There are four basic protocols in the OACerts scheme:

- **CA-Setup:** A CA picks a signature scheme Sig with a public-private key pair $(K_{\mathrm{CA}}, K_{\mathrm{CA}}^{-1})$, and a commitment scheme commit with public parameters Params. The public parameters of the CA are $\{\mathsf{Sig}, K_{\mathrm{CA}}, \mathsf{commit}, \mathsf{Params}\}$.

- **Issue Certificate:** A CA uses this protocol to issue an OACert to a user. A user Bob generates a public-private key pair (K_B, K_B^{-1}) and sends to the CA a certificate request that includes his public key K_B and attributes information $(attr_1, v_1), \ldots, (attr_m, v_m)$, and is signed by K_B^{-1}. After the CA verifies the correctness of $v_1, \ldots, v_m$ (most likely using off-line methods), it issues an OACert for Bob. In this process, the CA computes $c_i = \mathsf{commit}_{\mathsf{Params}}(v_i, r_i)$ and sends the certificate along with the secrets $r_1, \ldots, r_m$ to Bob. Bob stores the certificate and stores the values $(v_1, r_1), \ldots, (v_m, r_m)$ together with his private key K_B^{-1}. The role of the CA here is similar to the role of a CA in the traditional Public Key Infrastructure.

- **Alice-Bob initialization:** Bob, a certificate holder, establishes a secure communication channel with Alice, a service provider, and at the same time proves to Alice the ownership of an OACert. In this protocol, Alice checks the signature and the validity period of the certificate, then verifies that the certificate has not been revoked (using, e.g., standard techniques in [29]). Alice also verifies that Bob possesses the private key corresponding to K_B in the OACert. All these can be done using standard protocols such as TLS/SSL [39].

  Bob then requests the decryption key for an encrypted document, and Alice sends Bob her policy.

- **Alice-Bob Interaction:** Bob can show any subset of his attributes using the show attribute protocols. These protocols are executed after the show certificate protocol, through a secure communication channel between Alice and Bob. To show t attributes, Bob runs show attribute protocols t times. There are three kinds of show attribute protocols; each gives different computational and communication complexity and privacy level.

  1. *direct show:* Bob gives v_i and r_i directly to Alice, and Alice verifies $c_i = \mathsf{commit}(v_i, r_i)$. This protocol is used when Bob trusts Alice with the attribute values, or when Bob is very weak in computational power. This protocol is the most efficient one but offers the least privacy protection. Alice not only knows v_i but also can convince others that Bob has attribute v_i.

  2. *zero-knowledge show:* Bob uses zero-knowledge proofs to prove v_i satisfies some properties Alice requires, e.g., is equal to some value or belongs to some range. This kind of protocols is more expensive than the direct show, but offers better privacy protection. Alice learns whether v_i satisfies her policies, but she cannot convince others about this. Alice also doesn't learn the exact value of v_i provided that multiple values satisfy her policies.

  3. *oblivious show:* Bob interacts with Alice using OCBE protocols. Alice learns nothing about v_i. This kind of oblivious show protocols offers the best privacy protection among the three

types of show protocols. Often times, it has similar or less amount of computation as the zero-knowledge show protocols.

In practice, Alice and Bob may not share the same CA. That is, Alice may not know the CA that issues the OACerts to Bob and Alice may not trust that CA. We can handle this problem using a hierarchy of CAs with only the root of the hierarchy being trusted by Alice. For example, Bob is a student at StateU, and has a student certificate issued by College of Science (CoS) of StateU using the OACerts scheme. CoS has a valid certificate issued by StateU; and StateU is certified by Accreditation Board for Engineering and Technology (ABET). The certificate chain to prove that Bob is a valid student takes the form $\mathrm{ABET} \rightarrow \mathrm{StateU} \rightarrow \mathrm{CoS} \rightarrow \mathrm{Bob}$. There are three certificates associated with this chain, where the first two certificates are regular certificates (as there are no sensitive information in these certificates) and the last one is an OACert. Suppose Alice's policy is that only students in computer science can access the resource, and suppose Alice trusts ABET. Bob can first show the certificate chain to Alice without leaking any attribute information in his student certificate, and then run a zero-knowledge proof protocol to prove that his major is computer science.

Another practical consideration is that different CAs may use different attribute names for the same attribute. For example, Bureau of Motor Vehicles (BMV) use DoB as the attribute name for birth-date in the driver license, whereas Bureau of Consular Affairs use $\mathrm{date\ of\ birth}$ as the attribute name for birth-date in the passport. Alice and Bob can use application domain specification documents [33, 34] to make name agreement between different attribute names. It is also possible that different CAs use different encoding methods to convert an attribute value to an integer. To address this problem, each CA publishes its encoding methods online and signs them using its private key. When Bob shows his OACert to Alice, he also sends to Alice the encoding methods for his attributes signed by his CA. Alice can then adjust her policy based on the encoding methods. For example, in the digital driver license issued by BMV, birth-date field is encoded using the number of days between January 1st of 1900 and the actual date of birth. Suppose Alice's policy is that Bob's age must be between 30 and 40, Alice can convert her policy to be that the value of birth-date in Bob's OACert is between a and b, where a and b are birth-date values corresponding to age 30 and age 40, respectively.

3.2 Applications of OACerts

In additional to enabling oblivious access control, OACerts and OCBE are useful in the following settings.

Break policy cycles OACerts and OCBE can be used to break policy cycles (see [31] for definition) in automated trust negotiation [43, 42, 44, 46]. Consider the following scenario where Alice and Bob want to exchange their salary certificates. Alice's policy says that she can show her salary certificate only to those whose salary is great than $100k. Similarly, Bob will reveal his certificate only to other who earns more than $80k a year. Using current trust negotiation techniques, neither Alice nor Bob is willing to present her/his certificate first. The technique developed in [31] does not work well here neither, because the salary requirement in the policies is a range, not a specific value. Such problem can be solved using OACerts and OCBE. Suppose both Alice and Bob use OACerts as their salary certificates, Alice and Bob can first exchange their OACerts without revealing their salary values, then Bob uses an OCBE scheme to send Alice his salary value together with a non-interactive proof that the value sent is indeed the value committed in the OACerts, on the condition that Alice can open them (i.e., the value and the proof) only if her salary is more than $80k. Bob is certain that his salary figure is revealed to Alice only if Alice's income is more than $80k, thus Bob's policy is enforced without him knowing Alice's salary value.

Improve the efficiency of trust negotiation The goal of automated trust negotiation [43, 42, 44, 46] is to establish trust between strangers through interactive disclosure of certificates. OACerts and OCBE can simplify the trust negotiation process by reducing the rounds of interactions and the number of certificates exchanged. Consider the following scenario where Alice is web publisher and Bob is a senior citizen who wants to get access to Alice's resource. Alice's policy requires Bob to be older than 60. On the other hand, Bob only shows his birth-date to those who are a member of better business bureau (BBB). Using traditional trust negotiation techniques, Alice first shows her BBB certificate, then Bob reveals his driver license, finally Alice sends Bob her resource. The negotiation could be more complicated (and take more rounds) if there is an access control policy for Alice's BBB certificate.

Using OACerts, the trust between Alice and Bob can be established in one round – Alice sends her resource using an OCBE protocol such that Bob can receive the resource if and only if he is a senior citizen.

4 Definition of Oblivious Commitment-Based Envelope (OCBE)

We now give a formal definition of OCBE. While the definition follows the usage scenario described in Section 3 in general, it abstracts away some of the details in the scenario that have been solved using OACerts and focuses on the parts that still need to solved by the OCBE protocol.

Definition 1 (OCBE) An Oblivious Commitment-Based Envelope (OCBE) scheme is parameterized by a commitment scheme commit. It involves a sender S, a receiver R, and a trusted CA, and has the following phases:

CA-Setup CA takes a security parameter t and outputs the following: the public parameters Params for commit, a set $\mathcal{V}$ of possible values, and a set $\mathcal{P}$ of predicates. Each predicate in $\mathcal{P}$ maps an element in $\mathcal{V}$ to either true or false. The domain of $\mathsf{commit}[\mathsf{Params}]$ contains $\mathcal{V}$ as a subset.

CA-Commit R chooses a value $a \in \mathcal{V}$ (R's attribute value) and sends to CA. CA picks a random number r and computes the commitment $c = \mathsf{commit}_{\mathsf{Params}}(a, r)$. CA gives c and r to R, and c to S.

Recall that in the actual usage scenario, CA does not directly communicate with R. Instead, CA stores the commitment c in R's OACert certificate. The certificate is then sent by R to S, enabling S to have c as if it is sent from CA. Here we abstract these steps away to have CA sending c to S. We stress that CA does *not* participate in the interactions between S and R.

Initialization S chooses a message $M \in \{0, 1\}^*$. S and R agree[1] on a predicate $\mathsf{Pred} \in \mathcal{P}$.

Now S has Pred, c, and M. R has Pred, c, a, and r.

[1]The main effect of having both the sender and the receiver to affect the predicate is that in the security definitions both an adversarial sender and an adversarial receiver can choose the predicate they want to attack on.

Interaction S and R run an interactive protocol, during which an envelope containing an encryption of M is delivered from S to R.

Open After the interaction phase, if $\mathsf{Pred}(a)$ is $\mathtt{true}$, R outputs the message M; otherwise, R does nothing.

Observe that the receiver R's attributed value a is committed by a trusted CA. This is natural (and necessary) in our intended usage scenarios for OCBE.

4.1 Basic Cryptographic Assumptions

We say that a function f is *negligible* in the security parameter t if, for every polynomial p, $f(t)$ is smaller than $1/|p(t)|$ for large enough t; otherwise, it is *non-negligible*. The security of our OCBE protocols is based on two standard assumptions in cryptography and the random oracle model.

- *Discrete Logarithm (DL) Assumption.* The DL problem is the following: Given a finite cyclic group G, a generator $g \in G$, and a group element y, compute $\log_g y$. The DL assumption is that there exists no polynomial-time algorithm that can solve the DL problem with non-negligible probability.
- *Computational Diffie-Hellman (CDH) Assumption.* The CDH problem is the following: Given a finite cyclic group G, a generator $g \in G$, and group elements g^a, g^b, compute g^{ab}. The CDH assumption is that there exists no polynomial-time algorithm that can solve the CDH problem with non-negligible probability.
- *Random Oracle Model.* The random oracle model is an idealized security model introduced by Bellare and Rogaway [2] to analyze the security of certain natural cryptographic constructions. Roughly speaking, a random oracle is a function H: $X \rightarrow Y$ chosen uniformly at random from the set of all functions $\{h: X \rightarrow Y\}$ (we assume Y is a finite set). An algorithm can query the random oracle at any point $x \in X$ and receive the value $H(x)$ in response. Random oracles are used to model cryptographic hash functions such as SHA-1.

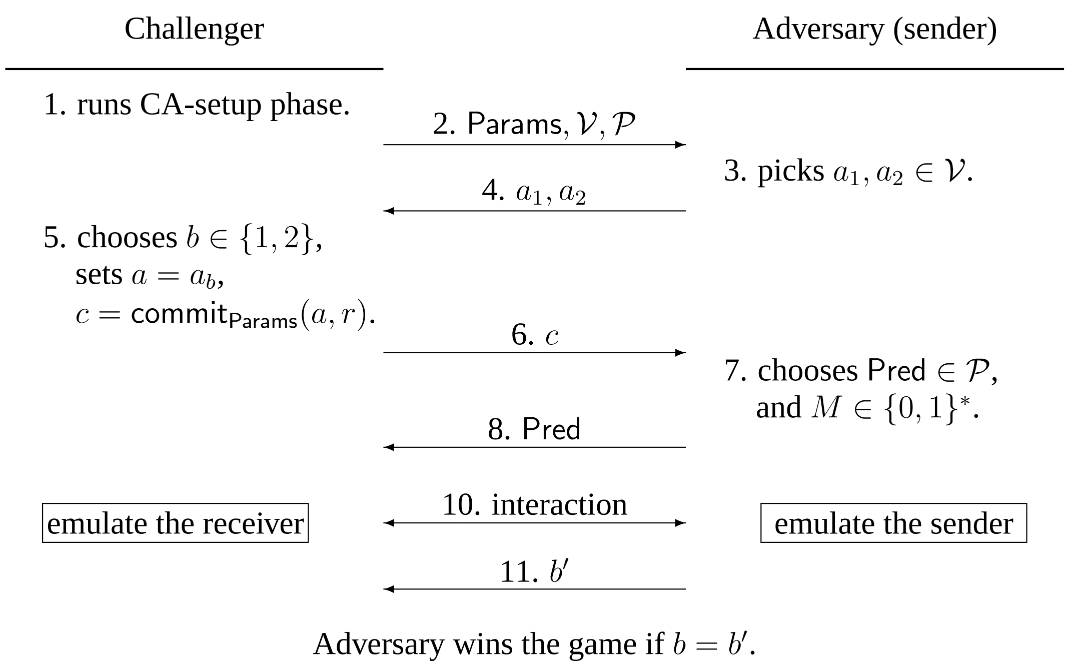
Challenger
Adversary (sender)
1. runs CA-setup phase.
2. Params, $\mathcal{V}, \mathcal{P}$
3. picks $a_1, a_2 \in \mathcal{V}$.
4. a_1, a_2
5. chooses $b \in \{1, 2\}$, sets $a = a_b$, $c = \mathsf{commit}_{\mathsf{Params}}(a, r)$.
6. c
7. chooses $\mathsf{Pred} \in \mathcal{P}$, and $M \in \{0, 1\}^*$.
8. Pred
emulate the receiver
10. interaction
emulate the sender
11. b'
Adversary wins the game if $b = b'$.

Figure 1: The attacker game for OCBE's oblivious property. We allow the adversary to pick a predicate Pred and two attribute values a_1, a_2 of her choice; yet the adversary still should not be able to distinguish a receiver with attribute a_1 from one with attribute a_2.

4.2 Security Definitions

Let an *adversary* be a probabilistic interactive Turing Machine [27]. An OCBE scheme must satisfy the following three properties. It must be sound, oblivious, and semantically secure against the receiver.

Sound An OCBE scheme is *sound* if in the case that $\mathsf{Pred}(a)$ is true, the receiver can output the message M with overwhelming probability, i.e., the probability that the receiver cannot output M is negligible.

Oblivious An OCBE scheme is *oblivious* if the sender learns nothing about a, i.e., no adversary $\mathcal{A}$ has a non-negligible advantage against the challenger in the game described in Figure 1 where the challenger emulates CA and the receiver, and the adversary emulates the sender. In other words, an OCBE scheme is *oblivious* if for every probabilistic interactive Turing Machine $\mathcal{A}$, $|\,\mathsf{Pr}[\mathcal{A}$ wins the game in Figure 1$] - \frac{1}{2}\,| \leq f(t)$, where f is a negligible function in t.

Secure against the receiver An OCBE scheme is *secure against the receiver* if the receiver learns nothing about M when $\mathsf{Pred}(a)$ is false, i.e., no adversary $\mathcal{A}$ has a non-negligible advantage against the

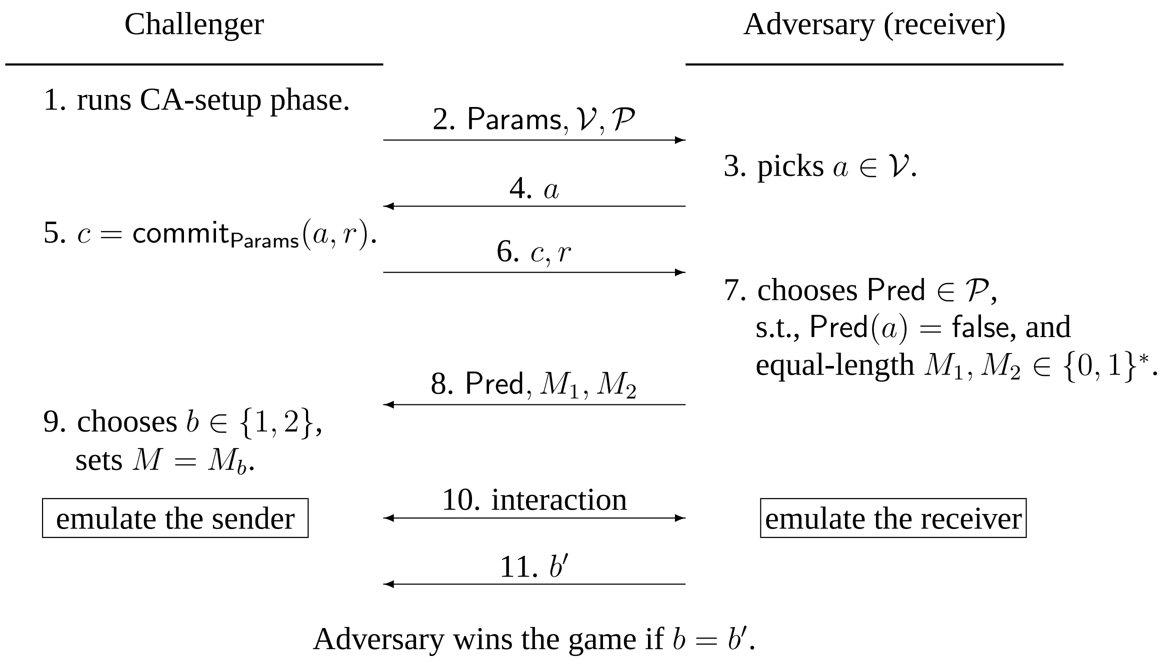
Challenger
Adversary (receiver)
1. runs CA-setup phase.
2. $\mathsf{Params}, \mathcal{V}, \mathcal{P}$
3. picks $a \in \mathcal{V}$.
4. a
5. $c = \mathsf{commit}_{\mathsf{Params}}(a, r)$.
6. c, r
7. chooses $\mathsf{Pred} \in \mathcal{P}$, s.t., $\mathsf{Pred}(a) = \mathsf{false}$, and equal-length $M_1, M_2 \in \{0,1\}^*$.
8. Pred, M_1, M_2
9. chooses $b \in \{1,2\}$, sets $M = M_b$.
emulate the sender
10. interaction
emulate the receiver
11. b'
Adversary wins the game if $b = b'$.

Figure 2: The attacker game for OCBE's security property against the receiver. Even if we give the adversary the power to pick two equal-length messages M_1 and M_2 of her choice, she still cannot distinguish an envelope containing M_1 from one containing M_2. This formalizes the intuitive notion that the envelope leaks no information about its content.

challenger in the game described in Figure 2 where the challenger emulates CA and the sender, and the adversary emulates the receiver.

We now argue that OCBE is an adequate solution to the two-party SFE problem in Problem 1, by showing intuitively that the security properties defined for OCBE suffice to prove that the scheme protects the privacy of the participants in the malicious model [25]. Observe that our definitions allow arbitrary adversaries, rather than just those following the protocol (semi-honest adversaries). The oblivious property guarantees that the sender's view of any protocol run can be simulated using just the sender's input, because one can simulate a protocol run between the sender and receiver, and no polynomially bounded sender can figure out the receiver's input. Soundness and security against the receiver guarantee that the receiver's view can be simulated using just the receiver's input and output. If the receiver's committed value a satisfies Pred, then the message M is in the output, one can therefore simulates the sender S. If the receiver's committed value a does not satisfy Pred, one can simulate the sender with a arbitrary message M' and no polynomially bounded receiver can tell the difference.

The security properties defined for OCBE guarantee also the correctness [25] of the OCBE protocol against malicious receivers. Our security definitions do not cover the correctness of the protocol against malicious senders, i.e., if the receiver's value does not satisfy the predicate, a malicious sender may trick the receiver to output the message M which violates the correctness of the protocol[2]. However, this malicious behavior does not make sense in the applications. If a malicious sender does not want to send the message M, she can choose not to participate in the protocol; on the other hand, if a malicious sender wants the receiver to see M without satisfying her policy; she can choose to send M directly rather than participating in the protocol.

We assume that the interaction phase of the OCBE scheme is executed on top of a previously established private communication channel between the sender and the receiver. Recall that the certificate holder establishes an SSL channel with the service provider using OACerts described in Section 3.

Note that the OCBE scheme itself does not have the non-transferability property. That is, a legitimate receiver, whose attribute value satisfies a sender's predicate, can share the values a, r, and c to others so that a non-legitimate receiver who knows a, r, and c can successfully obtain the sender's message. However, we stress that the OCBE protocol is executed only after the receiver shows his OACert to the sender and proves to the sender that he owns the OAcert (see the previous section for the usage of OACerts). In other words, the receiver has to show that c is certified in his OACerts and he has the private key to his OACerts. Therefore, our overall scheme is non-transferable. In order for a non-legitimate receiver to access the sender's message, the non-legitimate receiver has to know not only a, r, c from a legitimate receiver but also the private key to the legitimate receiver's OACert.

5 The Pedersen Commitment Scheme

We now review the Pedersen commitment scheme [38], which will be used in the OCBE protocols.

Definition 2 (The Pedersen Commitment Scheme)

Setup A trusted third party T chooses two large prime numbers p and q such that q divides $p-1$. It is

[2]In such case, the views of the sender and receiver cannot be simulated in the ideal model.

typical to have p be 1024 bits and q be 160 bits. Let g be a generator of G_q, the unique order-q subgroup of $\mathbb{Z}_p^*$. We use $x \leftarrow \mathbb{Z}_q$ to denote that x is uniformly randomly chosen from $\mathbb{Z}_q$. T picks $x \leftarrow \mathbb{Z}_q$ and computes $h = (g^x \bmod p)$. T keeps the value x secret and makes the values p, q, g, h public.

Commit The domain of the committed values is $\mathbb{Z}_q$. For a party A to commit an value $a \in \mathbb{Z}_q$, A chooses $r \leftarrow \mathbb{Z}_q$ and computes the commitment $c = (g^a h^r \bmod p)$.

Open To open a commitment c, A reveals a and r, and a verifier verifies whether $c = (g^a h^r \bmod p)$.

The above setting is slightly different from the standard setting of commitment schemes, in which the verifier runs the setup program and does a zero-knowledge proof to convince A that the parameters are constructed properly. We use a trusted third party to generate the parameters, because this is done by a trusted CA in the OACerts scheme.

The Pedersen commitment scheme is *unconditionally hiding*: Even with unlimited computational power it is impossible for an adversary to learn any information about the value a from c, because the commitments of any two numbers in $\mathbb{Z}_q$ have exactly the same distribution. This commitment scheme is *computationally binding*: Under the DL assumption, it is computationally infeasible for an adversarial committer to open a value a' other than a in the open phase of the commitment scheme. Suppose an adversary finds a' (other than a) and r' such that $g^{a'} h^{r'} \equiv g^a h^r (\bmod\ p)$, then she can compute $\frac{a'-a}{r-r'} \bmod q$, which is $\log_g(h)$, the discrete logarithm of h with respect to the base g.

6 OCBE Protocols

In this section, we present two OCBE protocols using the Pedersen commitment scheme, one for equality predicates, the other for greater-than-or-equal-to predicates. We then sketch how to construct OCBE protocols for other comparison predicates. All arithmetic in this section is assumed to be $\bmod\ p$ unless otherwise specified.

6.1 EQ-OCBE: an OCBE protocol for equality predicates

Our EQ-OCBE protocol runs a Diffie-Hellman style key-agreement protocol [17] with the twist that the receiver can compute the shared secret if and only if the receiver's committed value a is equal to a_0.

Protocol 1 (EQ-OCBE) Let $\mathcal{E}$ be a semantically secure symmetric encryption scheme with keyspace $\{0,1\}^s$. Let $H : G_q \to \{0,1\}^s$ be a cryptographic hash function that extracts a key for $\mathcal{E}$ from an element in the group G_q, the order-q subgroup of $\mathbb{Z}_p^*$. EQ-OCBE involves a sender S, a receiver R, and a trust CA.

CA-Setup CA takes a security parameter t and runs the setup algorithm of the Pedersen commitment scheme to create $\mathsf{Params} = \langle p, q, g, h \rangle$. CA also outputs $\mathcal{V} = \mathbb{Z}_q$ and $\mathcal{P} = \{\mathsf{EQ}_{a_0} \mid a_0 \in \mathcal{V}\}$, where $\mathsf{EQ}_{a_0} \colon \mathcal{V} \to \{\mathsf{true}, \mathsf{false}\}$ is a predicate such that $\mathsf{EQ}_{a_0}(a)$ is true if $a = a_0$ and false if $a \neq a_0$.

CA-Commit R chooses an integer $a \in \mathcal{V}$ and sends to CA. CA picks $r \leftarrow \mathbb{Z}_q$ and computes the commitment $c = g^a h^r$. CA gives c and r to R, and c to S.

Initialization S chooses a message $M \in \{0,1\}^*$. S and R agree on a predicate $\mathsf{EQ}_{a_0} \in \mathcal{P}$.

Now S has EQ_{a_0}, c, and M. R has EQ_{a_0}, c, a, and r.

Interaction S picks $y \leftarrow \mathbb{Z}_q^*$, computes $\sigma = (cg^{-a_0})^y$, and then sends to R the pair $\langle \eta = h^y, C = \mathcal{E}_{H(\sigma)}[M] \rangle$.

Open R receives $\langle \eta, C \rangle$ from the interaction phase. If $\mathsf{EQ}_{a_0}(a)$ is true, R computes $\sigma' = \eta^r$, and decrypts C using $H(\sigma')$.

To see that EQ-OCBE is sound, observe that when $\mathsf{EQ}_{a_0}(a)$ is true,

$$\sigma = (cg^{-a_0})^y = (g^a h^r g^{-a_0})^y = (g^{a-a_0} h^r)^y = (h^r)^y = (h^y)^r = \eta^r = \sigma'.$$

Therefore the sender and receiver share the same symmetric key.

Also observe that the interaction phase of the EQ-OCBE protocol is one-round; it involves only one message from the sender to the receiver. In the interaction and open phases, the sender does two exponentiations and the receiver does one exponentiation.

The key idea of EQ-OCBE is that if the receiver's committed value a is equal to a_0, the sender can compute $cg^{-a_0} = g^{a-a_0}h^r = h^r$. The sender now holds h^r such that the receiver knows the value r. This achieves half of the Diffie-Hellman key-agreement protocol [17], with h as the base. The sender then does the other half by sending h^y to the receiver. Thus both the sender and receiver can compute $\sigma = (cg^{-a_0})^y = h^{ry}$. If the receiver's committed value a is not equal to a_0, then it is presumably hard for him to compute $\sigma = (cg^{-a_0})^y$ from h^y and cg^{-a_0}. The receiver cannot effectively compute $\log_h(cg^{-a_0})$, because if the receiver is able to find a number $r' = \log_h(cg^{-a_0})$, he can break the binding property of the commitment scheme, i.e., he finds a (a_0, r') pair such that $g^{a_0}h^{r'} = g^a h^r$.

Theorem 1 *EQ-OCBE is oblivious.*

Proof. The interaction phase involves only one message from the sender to the receiver. Among what the sender sees, the only piece of information that is related to the receiver's attribute value a is the commitment c. As the Pedersen commitment scheme is unconditionally hiding; c does not leak *any* information about a. Thus EQ-OCBE is oblivious even against an infinitely powerful adversary. ∎

Theorem 2 *Under the CDH assumption on G_q, the order-q subgroup of $\mathbb{Z}_p^*$, and when H is modeled as a random oracle, EQ-OCBE is secure against the receiver.*

Proof. EQ-OCBE uses a semantically secure symmetric encryption algorithm. When H is modeled as a random oracle, EQ-OCBE is secure against the receiver when no receiver whose committed value is not equal to a_0 can compute with non-negligible probability $\sigma = (cg^{-a_0})^y$, the secret that the sender uses to derive the encryption key. More precisely, EQ-OCBE is secure against the receiver if no polynomial-time adversary wins the following game against the challenger with non-negligible probability (this game is instantiated from the game in Figure 2 with details from the EQ-OCBE protocol): The challenger runs the setup phase and sends $\mathsf{Params} = \langle p, q, g, h \rangle$ and the descriptions of $\mathcal{V}$ and $\mathcal{P}$ to the adversary. The adversary picks an integer $a \in \mathcal{V}$. The challenger chooses $r \leftarrow \mathbb{Z}_q$ and computes the commitment of a as $c = g^a h^r$, and gives r and c to the adversary. The adversary responds with an equality predicate EQ_{a_0} such that $\mathsf{EQ}_{a_0}(a)$ is false. The challenger then picks $y \leftarrow \mathbb{Z}_q^*$ and sends to the adversary h^y. The adversary then outputs σ, and the adversary wins the game if $\sigma = (cg^{-a_0})^y$.

Given an attacker $\mathcal{A}$ that wins the above game with probability ϵ, we construct another attacker $\mathcal{B}$ that solves the CDH problem in G_q with the same probability. $\mathcal{B}$ does the following:

1. $\mathcal{B}$, when given $p, q, h \in G_q, h^x, h^y$, gives $\mathsf{Params} = \langle p, q, h^x, h \rangle$ and the descriptions of $\mathcal{V} = \mathbb{Z}_q$ and $\mathcal{P} = \{\mathsf{EQ}_{a_0} \mid a_0 \in \mathcal{V}\}$ to $\mathcal{A}$. Let g denote h^x.
2. $\mathcal{B}$ receives an integer $a \in \mathbb{Z}_q$ from $\mathcal{A}$, picks $r \leftarrow \mathbb{Z}_q$, computes $c = (h^x)^a h^r$, and sends r and c to $\mathcal{A}$.
3. $\mathcal{B}$ receives an equality predicate EQ_{a_0} from $\mathcal{A}$ where $a \neq a_0$, and sends h^y to $\mathcal{A}$.
4. $\mathcal{B}$ receives σ from $\mathcal{A}$, computes $\delta = \sigma h^{-ry}$, and outputs $\delta^{(a-a_0)^{-1} \bmod q}$.

   When $\mathcal{A}$ wins the game, $\sigma = (cg^{-a_0})^y = (g^{a-a_0}h^r)^y = (g^y)^{a-a_0}h^{ry}$, then $\delta = \sigma h^{-ry} = (g^y)^{a-a_0} = (h^{xy})^{a-a_0}$. $\mathcal{B}$ outputs $\delta^{(a-a_0)^{-1} \bmod q} = h^{xy}$.

$\mathcal{B}$ succeeds in solving the CDH problem if $\mathcal{A}$ wins the above game, i.e., successfully computes $(cg^{-a_0})^y$. ∎

6.2 GE-OCBE: an OCBE protocol for greater-than-or-equal-to predicates

In this section, we present an OCBE protocol (GE-OCBE) for the Pedersen commitment scheme with greater-than-or-equal-to predicates. The basic idea of the GE-OCBE protocol is as follows. Let ℓ be an integer such that $2^\ell < q/2$. Let a and a_0 be two numbers in $[0..2^\ell - 1]$, and let $d = ((a - a_0) \bmod q)$. Let $c = g^a h^r$ be a commitment of a where r is known to the receiver, then $cg^{-a_0} = g^{a-a_0}h^r = g^d h^r$ is a commitment of d that the receiver knows how to open. Notice that if $a \geq a_0$ then $d \in [0..2^\ell - 1]$, otherwise $d \notin [0..2^\ell - 1]$.

If $a \geq a_0$, the receiver generates ℓ new commitments $c_0, \ldots, c_{\ell-1}$, one for each of the ℓ bits of d. The sender picks a random encryption key k and split it into ℓ secrets $k_0, \ldots, k_{\ell-1}$. Then the sender and receiver run a "bit-OCBE" protocol for each commitment, i.e., if c_i is a bit-commitment, the receiver obtains k_i, otherwise he gets nothing, while the sender learns nothing about the value committed under c_i.

Protocol 2 (GE-OCBE) Let $\mathcal{E}$ be a semantically secure symmetric encryption scheme with keyspace $\{0,1\}^s$. Let $H : G_q \rightarrow \{0,1\}^s$ and $H' : \{0,1\}^{s\ell} \rightarrow \{0,1\}^s$ be two cryptographic hash functions. Our GE-OCBE protocol involves a sender S, a receiver R, and a trust CA.

CA-Setup CA takes two parameters, a security parameter t and a parameter ℓ (which specifies the desired range of the attribute values). CA runs the setup algorithm of the Pedersen commitment scheme to create $\mathsf{Params} = \langle p, q, g, h \rangle$ such that $2^\ell < q/2$. CA also outputs $\mathcal{V} = [0..2^\ell - 1]$ and $\mathcal{P} = \{\mathsf{GE}_{a_0} \mid a_0 \in \mathcal{V}\}$, where GE_{a_0}: $\mathcal{V} \rightarrow \{\mathsf{true}, \mathsf{false}\}$ is a predicate such that $\mathsf{GE}_{a_0}(a)$ is true if $a \geq a_0$ and false otherwise.

CA-Commit R chooses an integer $a \in \mathcal{V}$ and sends to CA. CA picks $r \leftarrow \mathbb{Z}_q$ and computes the commitment $c = g^a h^r$. CA gives c and r to R, and c to S.

Initialization S chooses a message $M \in \{0,1\}^*$. S and R agree on a predicate $\mathsf{GE}_{a_0} \in \mathcal{P}$.

Now S has GE_{a_0}, c, and M. R has GE_{a_0}, c, a, and r.

Interaction Let $d = ((a - a_0) \mod q)$, $\mathsf{GE}_{a_0}(a) = \mathsf{true}$ if and only if $d \in [0..2^\ell - 1]$. Note that $cg^{-a_0} = g^d h^r$ is a commitment of d that R can open.

1. R picks $r_1, \ldots, r_{\ell-1} \leftarrow \mathbb{Z}_q$ and sets $r_0 = r - \sum_{i=1}^{\ell-1} 2^i r_i \mod q$. When $\mathsf{GE}_{a_0}(a) = \mathsf{true}$, let $d_{\ell-1} \ldots d_1 d_0$ be the binary representation of d, i.e., $d = d_0 2^0 + d_1 2^1 + \cdots + d_{\ell-1} 2^{\ell-1}$. When $\mathsf{GE}_{a_0}(a) = \mathsf{false}$, R randomly picks $d_1, d_2, \ldots, d_{\ell-1} \leftarrow \{0,1\}$, and sets $d_0 = d - \sum_{i=1}^{\ell-1} 2^i d_i \mod q$. R computes, for $0 \leq i \leq \ell - 1$, the commitment $c_i = \mathsf{commit}(d_i, r_i) = g^{d_i} h^{r_i}$. R sends $c_0, \ldots, c_{\ell-1}$ to S.
2. S verifies that $cg^{-a_0} = \prod_{i=0}^{\ell-1} (c_i)^{2^i}$. S randomly chooses ℓ symmetric keys $k_0, \ldots, k_{\ell-1} \in \{0,1\}^t$ and sets $k = H'(k_0 || \cdots || k_{\ell-1})$. S picks $y \leftarrow \mathbb{Z}_q^*$, computes $\eta = h^y$ and $C = \mathcal{E}_k[M]$. For each $0 \leq i \leq \ell - 1$, S computes $\sigma_i^0 = (c_i)^y$, $\sigma_i^1 = (c_i g^{-1})^y$, $C_i^0 = H(\sigma_i^0) \oplus k_i$, and $C_i^1 = H(\sigma_i^0) \oplus k_i$. S sends to R the tuple $\langle \eta, C_0^0, C_0^1, \ldots, C_{\ell-1}^0, C_{\ell-1}^1, C \rangle$.

Open R receives $\langle \eta, C_0^0, C_0^1, \ldots, C_{\ell-1}^0, C_{\ell-1}^1, C \rangle$ from the interaction phase. If $\mathsf{GE}_{a_0}(a)$ is true, $d = \sum_{i=0}^{\ell-1} 2^i d_i$ where $d_i \in \{0,1\}$. For each $0 \leq i \leq \ell - 1$, R computes $\sigma_i' = \eta^{r_i}$, and obtains $k_i' = H(\sigma_i') \oplus C_i^{d_i}$. R then computes $k' = H'(k_0' || \cdots || k_{\ell-1}')$, and decrypts C using k'.

To see that the GE-OCBE protocol is sound, observe that when $\mathsf{GE}_{a_0}(a)$ is true, $d_0, \ldots, d_{\ell-1}$ are either 0 or 1. If the receiver follows the protocol, the sender will succeed in verifying $\prod_{i=0}^{\ell-1}(c_i)^{2^i} = \prod_{i=0}^{\ell-1}(g^{d_i}h^{r_i})^{2^i} = g^d h^r = cg^{-a_0}$. For each $0 \le i \le \ell - 1$, if $d_i = 0$, $\sigma_i^0 = (c_i)^y = (g^{d_i}h^{r_i})^y = (h^y)^{r_i} = \eta^{r_i} = \sigma_i'$, the receiver can compute $k_i = C_i^0 \oplus H(\sigma_i')$; if $d_i = 1$, $\sigma_i^1 = (c_i g^{-1})^y = (g^{d_i-1}h^{r_i})^y = (h^y)^{r_i} = \eta^{r_i} = \sigma_i'$, the receiver can compute $k_i = C_i^1 \oplus H(\sigma_i')$. As $k = H'(k_0||\cdots||k_{\ell-1})$, the receiver can successfully obtain k. Thus the sender and receiver share the same symmetric key k if $\mathsf{GE}_{a_0}(a)$ is true.

The interaction phase of the GE-OCBE protocol is two rounds. The receiver does about 2ℓ exponentiations. The sender does about ℓ exponentiations (observe that σ_i^1 can be computed as $\sigma_i^0 g^{-y}$, where g^{-y} needs to be computed only once).

We briefly sketch the idea why the receiver cannot obtain M if $\mathsf{GE}_{a_0}(a)$ is false. If the receiver follows the protocol, then $d_1, \ldots, d_{\ell-1} \in \{0,1\}$ and $d_0 \not\in \{0,1\}$. The receiver can successfully compute $k_1, \ldots, k_{\ell-1}$, but fails to compute k_0 because he can compute neither $\sigma_0^0 = (c_0)^y = (g^{d_0}h^r)^y$ nor $\sigma_0^1 = (c_0 g^{-1})^y = (g^{d_0-1}h^r)^y$. Even if the receiver does not follow the protocol, it is impossible for him to find $d_0, \ldots, d_{\ell-1} \in \{0,1\}$ and $r_0, \ldots, r_{\ell-1}$ such that $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i}$ and $c_i = g^{d_i}h^{r_i}$. Suppose the receiver finds such $d_0, \ldots, d_{\ell-1} \in \{0,1\}$ and $r_0, \ldots, r_{\ell-1}$; let $d' = \sum_{i=0}^{\ell-1} d_i 2^i \in [0..2^\ell - 1]$ and $r' = \sum_{i=0}^{\ell-1} r_i 2^i \pmod q$, then

$$g^{a-a_0}h^r = cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i} = \prod_{i=0}^{\ell-1}(g^{d_i}h^{r_i})^{2^i} = g^{\sum_{i=0}^{\ell-1} d_i 2^i} h^{\sum_{i=0}^{\ell-1} r_i 2^i} = g^{d'}h^{r'}.$$

As $a - a_0 \not\in [0..2^\ell - 1]$ and $d' \in [0..2^\ell - 1]$, $d' \neq a - a_0$, the receiver is able to find $a - a_0$, r, d', and r' such that $g^{a-a_0}h^r = g^{d'}h^{r'}$, which breaks the binding property of the Pedersen commitment scheme.

Theorem 3 *GE-OCBE is oblivious.*

Proof. Consider the game for the oblivious property of OCBE (in Figure 1), let us examine what an adversary would see in the case of GE-OCBE. The adversary sees a commitment c and ℓ commitments $c_0, \ldots, c_{\ell-1}$ such that $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i}$. The joint distribution of $c, c_0, \ldots, c_{\ell-1}$ is independent of whether the challenger picked a_0 or a_1, as $c, c_1, \ldots, c_{\ell-1}$ are totally random (because of the random choices of $r, r_1, \ldots, r_{\ell-1}$), and c_0 is always equal to $cg^{-a_0}\prod_{i=1}^{\ell-1}(c_i)^{-2^i}$. GE-OCBE is oblivious even against an infinitely powerful adversary. ∎

Theorem 4 *Under the CDH assumption on G_q, the order-q subgroup of $\mathbb{Z}_p^*$, and when H and H' are modeled as random oracles, GE-OCBE is secure against the receiver.*

Proof. GE-OCBE uses a semantically secure symmetric encryption algorithm. When H' is modeled as a random oracle, EQ-OCBE is secure against the receiver when no receiver whose committed value a does not satisfy GE_{a_0} can compute with non-negligible probability $k_0||\ldots||k_{\ell-1}$, the secret that the sender uses to derive the encryption key k. In other words, if $\mathsf{GE}_{a_0}(a)$ is false, we need to show that no receiver can compute $k_0, \ldots, k_{\ell-1}$ with non-negligible probability. Recall that the receiver is given $C_i^0 = H(\sigma_i^0) \oplus k_i$ and $C_i^1 = H(\sigma_i^1) \oplus k_i$, when H is also modeled as a random oracle, the receiver has to know either σ_i^0 or σ_i^1 to recover k_i.

GE-OCBE is secure against the receiver if no polynomial-time adversary wins the following game against the challenger with non-negligible probability (this game is instantiated from the game in Figure 2 with details from the GE-OCBE protocol): The challenger runs the setup phase and sends $\mathsf{Params} = \langle p, q, g, h\rangle$ and the descriptions of $\mathcal{V}$ and $\mathcal{P}$ to the adversary. The adversary picks an integer $a \in \mathcal{V}$. The challenger chooses $r \leftarrow \mathbb{Z}_q$ and computes the commitment of a as $c = g^a h^r$, and gives r and c to the adversary. The adversary responds with a greater-than-or-equal-to predicate GE_{a_0} such that $\mathsf{GE}_{a_0}(a)$ is false. The adversary outputs ℓ commitments $c_0, \ldots, c_{\ell-1}$ such that $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i}$. The challenger then picks $y \leftarrow \mathbb{Z}_q^*$ and sends to the adversary h^y. The adversary then outputs $\sigma_0, \ldots, \sigma_{\ell-1}$ and $d_0, \ldots, d_{\ell-1} \in \{0, 1\}$, and the adversary wins the game if each $0 \leq i \leq \ell - 1$, $\sigma_i = (c_i g^{-d_i})^y$ holds.

Given an attacker $\mathcal{A}$ that wins the above game with probability ϵ, we construct another attacker $\mathcal{B}$ that solves the CDH problem in G_q with the same probability. $\mathcal{B}$ does the following:

1. $\mathcal{B}$, when given $p, q, h \in G_q, h^x, h^y$, gives $\mathsf{Params} = \langle p, q, h^x, h\rangle$ and the descriptions of $\mathcal{V} = \mathbb{Z}_q$ and $\mathcal{P} = \{\mathsf{GE}_{a_0} \mid a_0 \in \mathcal{V}\}$ to $\mathcal{A}$. Let g denote h^x.

2. $\mathcal{B}$ receives an integer $a \in \mathbb{Z}_q$ from $\mathcal{A}$, picks $r \leftarrow \mathbb{Z}_q$, computes $c = (h^x)^a h^r$, and sends r and c to $\mathcal{A}$.

3. $\mathcal{B}$ receives a great-than-or-equal-to predicate GE_{a_0} from $\mathcal{A}$ where $a < a_0$. $\mathcal{B}$ computes $d = ((a - a_0) \bmod q)$.

4. $\mathcal{B}$ receives ℓ commitments $c_0, \ldots, c_{\ell-1}$ where $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i}$, and sends h^y to $\mathcal{A}$.

5. $\mathcal{B}$ receives $\sigma_0, \ldots, \sigma_{\ell-1}$, and $d_0, \ldots, d_{\ell-1}$ from $\mathcal{A}$. $\mathcal{B}$ computes $\delta = \prod_{i=0}^{\ell-1}(\sigma_i)^{2^i}$ and $d' = \sum_{i=0}^{\ell-1} d_i 2^i$, and outputs $(\delta h^{-ry})^{(d-d')^{-1} \bmod q}$.

   When $\mathcal{A}$ wins the game, $\sigma_i = (c_i g^{-d_i})^y$, then

$$\delta = \prod_{i=0}^{\ell-1}(\sigma_i)^{2^i} = \prod_{i=0}^{\ell-1}((c_i g^{-d_i})^y)^{2^i} = (g^{-d'} \prod_{i=0}^{\ell-1}(c_i)^{2^i})^y = (g^{-d'} cg^{-a_0})^y = (g^{d-d'} h^r)^y = g^{(d-d')y} h^{ry}.$$

   $\mathcal{B}$ outputs $(\delta h^{-ry})^{(d-d')^{-1} \bmod q} = (g^{(d-d')y})^{(d-d')^{-1} \bmod q} = g^y = h^{xy}$.

$\mathcal{B}$ succeeds in solving the CDH problem if $\mathcal{A}$ wins the above game, i.e., successfully computes $(c_0 g^{-d_0})^y, \ldots, (c_{\ell-1} g^{-d_{\ell-1}})^y$, where $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i}$, and $d_0, \ldots, d_{\ell-1} \in \{0, 1\}$. ∎

6.3 OCBE protocols for other predicates

In this section, we first present two logical combination OCBE protocols, one for $\wedge$ (AND-OCBE), the other for $\vee$ (OR-OCBE). Then we describe OCBE protocols for comparison predicates: $>$ (GT-OCBE), $\leq$ (LE-OCBE), $<$ (LT-OCBE), $\neq$ (NE-OCBE). Finally, we present an OCBE protocol for range predicates (RANGE-OCBE). Instead of formally presenting these protocols, we briefly sketch the ideas. We use $OCBE(\mathsf{Pred}, a, M)$ to denote an OCBE protocol with predicate Pred and committed value a, the receiver outputs M if $\mathsf{Pred}(a)$ is true. Similar techniques have been used before in [7, 32].

1. **AND-OCBE**: Suppose there exists OCBE protocols for Pred_1 and Pred_2, the goal is to build an OCBE protocol for the new predicate $\mathsf{Pred} = \mathsf{Pred}_1 \wedge \mathsf{Pred}_2$. An $OCBE(\mathsf{Pred}_1 \wedge \mathsf{Pred}_2, a, M)$ can be constructed as follows: In the interaction phase, the sender picks two random keys k_1 and k_2 and sets $k = H(k_1||k_2)$, where H is a cryptographic hash function. The sender then runs the interaction phases of $OCBE(\mathsf{Pred}_1, a, k_1)$ and $OCBE(\mathsf{Pred}_2, a, k_2)$ with the receiver. Finally, the sender sends $\mathcal{E}_k[M]$ to the receiver. The receiver can recover M in the open phase only if both $\mathsf{Pred}_1(a)$ and $\mathsf{Pred}_2(a)$ are true.

2. **OR-OCBE**: An $OCBE(\mathsf{Pred}_1 \vee \mathsf{Pred}_2, M)$ can be constructed as follows: In the interaction phase, the sender picks a random key k. The sender then runs the interaction phases of $OCBE(\mathsf{Pred}_1, a, k)$ and $OCBE(\mathsf{Pred}_2, a, k)$ with the receiver. Finally, the sender sends $\mathcal{E}_k[M]$ to the receiver. The receiver can recover M in the open phase if either $\mathsf{Pred}_1(a)$ or $\mathsf{Pred}_2(a)$ is true.

3. **GT-OCBE**: For integer space, $a > a_0$ is equivalent to $a \geq a_0 + 1$. An $OCBE(>_{a_0}, a, M)$ protocol is equivalent to an $OCBE(\geq_{a_0+1}, a, M)$ protocol.

4. **LE-OCBE**: The idea of LE-OCBE protocol is similar to the GE-OCBE protocol. Observe that $a \leq a_0$ if and only if $d = ((a_0 - a) \bmod q) \in [0..2^\ell - 1]$. Let $c = g^a h^r$ be a commitment of a, then $g^{a_0} c^{-1} = g^{(a_0-a) \bmod q} h^{-r \bmod q}$ is a commitment of d such that the receiver knows how to open. The LE-OCBE protocol uses the same method as in GE-OCBE.

5. **LT-OCBE**: For integer space, $a < a_0$ is equivalent to $a \leq a_0 - 1$. An $OCBE(<_{a_0}, a, M)$ protocol is equivalent to an $OCBE(\leq_{a_0-1}, a, M)$ protocol.

6. **NE-OCBE**: $a \neq a_0$ is equivalent to $(a > a_0) \vee (a < a_0)$. Therefore, an $OCBE(\neq_{a_0}, a, M)$ can be built as $OCBE(>_{a_0} \vee <_{a_0}, a, M)$.

7. **RANGE-OCBE**: $a_0 \leq a \leq a_1$ is equivalent to $(a \geq a_0) \wedge (a \leq a_1)$. Therefore, a RANGE-OCBE can be built as $OCBE(\geq_{a_0} \wedge \leq_{a_1}, a, M)$.

6.4 MOCBE: Multi-attribute OCBE

OCBE guarantees that, for the receiver to receive a message, her attribute committed in her OACert must satisfy the sender's policy. In many scenarios, access control policies are based on multiple attributes rather than one. For example, a policy may require that the receiver either has GPA more than 3.0 or is older than 21. This requirement involves two attribute a_1 (GPA) and a_2 (age), and the predicate for the sender is $(a_1 > 3.0) \vee (a_2 > 21)$. It is natural to extend OCBE to support multiple attributes, called MOCBE. In this subsection, we present constructions of MOCBE for two types of multi-attribute comparison predicates which we believe are useful in practice. Let $\diamond$ denote a comparison operation where $\diamond \in \{=, \neq, <, >, \leq, \geq\}$. Our constructions use the Pedersen commitment scheme and use the OCBE protocols as sub-protocols.

Linear Relation Predicates The linear relation predicates $\mathsf{Pred}(a_1, \ldots, a_n)$ take the form of $a_1b_1 + \cdots + a_nb_n \ \diamond \ e$, where $b_1, \ldots, b_n$, and e are public integers from $\mathcal{V}$. In other words, $\mathsf{Pred}(a_1, \ldots, a_n)$ is true if $a_1b_1 + \cdots + a_nb_n \ \diamond \ e$ is true, and is false otherwise. The MOCBE protocol of this type of predicates can be built as follows: Since the Pedersen commitment scheme is a homomorphic commitment scheme, the sender and receiver each can compute the commitment of $a_1b_1 + \cdots + a_nb_n$ (denoted as x) by computing $c_1^{b_1}c_2^{b_2} \cdots c_n^{b_n}$ (denote as c). Now both the sender and the receiver have c, the receiver knows how to open the commitment c, and we want the receiver to obtain the sender's message if and only if x (the value committed in c) satisfies $x \diamond e$. We reduce the MOCBE protocol to the OCBE protocols for comparison predicates.

General Comparison Predicates The idea of this construction comes from [32]. The predicate $\mathsf{Pred}(a_1, \ldots, a_n)$ is specified as a boolean circuit with n input and one output, each input i is associate with a predicate $a_i \diamond e_i$ where e_i is an integer in $\mathcal{V}$. The circuit consists of AND gates and OR gates; each gate has two or more inputs and one output. Intuitively, a receiver makes an input true if $a_i \diamond e_i$ is true. A receiver satisfies the predicate if it makes the output of the circuit true. The MOCBE protocol is as follows:

1. For each $i = 1..n$, the sender chooses a random key k_i and runs an OCBE protocol with the receiver, sending k_i in an envelope that can be opened only when $a_i \diamond e_i$ is true.

2. The sender computes the keys associated with (the output of) each gate as follows, starting from the input of the circuit. For an AND gate, let $k^{(1)}, k^{(2)}, \ldots, k^{(m)}$ be the keys associated with the m inputs, then the key corresponding to the output is $k = k^{(1)} \oplus \ldots \oplus k^{(m)}$. For an OR gate, let $k^{(1)}, k^{(2)}, \ldots, k^{(m)}$ be the keys associated with the m inputs. The sender chooses a random key k as the output key. The sender then encrypts k under each of $k^{(1)}, k^{(2)}, \ldots, k^{(m)}$, and sends the m ciphertexts to the receiver.

3. The sender encrypts the message M using the key associated with the circuit output and sends the ciphertext C to the receiver.

It is not hard to see that if the receiver's attributes $a_1, \ldots, a_n$ satisfy the predicate $\mathsf{Pred}(a_1, \ldots, a_n)$, then the receiver can obtain the key associated with the circuit output. Thus the receiver is able to decrypt C and obtain M.

7 Implementation and Performance

We have implemented a toolkit that generates X.509 certificates [29] that are also OACerts using Java v1.4.2 SDK and JCSI PKI Server Library [30]. In our implementation, both the parameters of the Pedersen commitment scheme and commitments of certificate holder's attributes are encoded in the X.509v3 extension fields. Recall that the parameters of the Pedersen commitment scheme are $\langle p, q, g, h \rangle$; they are large integer numbers. The commitments can also be viewed as large integers. We convert each of these integers into an octet string and bind it with an unique object identifier (OID) [29], and place them (octet string and OID) in the extension fields as a non-critical extension. Note that attribute name is not encoded in the certificate. The CA can publish a list of attribute names and their corresponding OID, so that service providers know which commitment corresponds to which attribute. Our OACerts can be recognized by OpenSSL.

We implemented also the OCBE protocols and zero-knowledge proof protocols [12, 16, 13, 36] in Java with Java 2 Platform v1.4.2 SDK. We use the Pedersen commitment scheme with security parameters $p = 1024$ bits and $q = 160$ bits. Thus the size of a commitment is 1024 bits, or 128 bytes. We set the attribute values in OACerts to be unsigned long, i.e., $\ell = 32$. In the implementation of the OCBE protocols, we use MD5 as the cryptographic hash function, AES as the symmetric key encryption scheme. Given an arbitrary size message, MD5 outputs a 128-bit message digest. In our setting, M is typically a 16-byte symmetric key, the size of $\mathcal{E}[M]$ is also 16 bytes using AES in ECB mode. In EQ-OCBE, η is 128 bytes (1024 bits) and C is 16 bytes, the total size of communication is 144 bytes. We ran our implementation on a 2.53GHz Intel Pentium 4 machine with 384MB RAM running RedHat

Linux 9.0. We simulate the certificate holder and service provider on the same machine. With p of size 1024 bits and q of size 160 bits in the Pedersen commitment scheme, and $\ell = 32$, the performance of two zero-knowledge proof protocols and two OCBE protocols is summarized in Table 1.

	execution time	communication size
Zero-knowledge proof that $a = a_0$	28 ms	168 bytes
Zero-knowledge proof that $a \geq a_0$	2.2 s	15 KB
EQ-OCBE	75 ms	144 bytes
GE-OCBE	0.9 s	5.1 KB

Table 1: Running time and size of communication on a 2.53GHz Intel Pentium 4 running RedHat Linux. Security parameters are $\ell = 32$, $p = 1024$ bits, and $q = 160$ bits.

8 Conclusion and Future Work

In this paper, we proposed OACerts, an attribute certificate scheme that enables oblivious access control. We introduced the notion of OCBE, and developed provably secure and efficient OCBE protocols for the Pedersen commitment scheme and predicates such as $=, \geq, \leq, >, <, \neq$ as well as logical combinations of them. Future work includes developing efficient OCBE protocols for predicates other than comparison predicates, such as set predicate. Take the set predicate as an example, although a set predicate $x \in \{x_1, \ldots, x_n\}$ can be represented with $x = x_1 \vee \ldots \vee x = x_n$, and we can construct an OCBE protocol for any set predicate using a combination of EQ-OCBE protocol and OR-OCBE protocol, unfortunately such construction is not efficient if the set is large.

Acknowledgement This work is supported by NSF ITR grant CCR-0325951 and by sponsors of CERIAS. We would like to thank Dan Boneh for his helpful discussions and thank anonymous reviewers for their helpful comments. We thank also Ziad Bizri, Ji-Won Byun, Klorida Miraj, and Mahesh V. Tipunitara for reading drafts of the paper and making suggestions that have improved the paper's presentation.

References

[1] Dirk Balfanz, Glenn Durfee, Narendar Shankar, Diana Smetters, Jessica Staddon, and Hao-Chi Wong. Secret handshakes from pairing-based key agreements. In *Proceedings of the IEEE Symposium and Security and Privacy*, pages 180–196, May 2003.

[2] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73. ACM Press, 1993.

[3] Matt Blaze, Joan Feigenbaum, and Jack Lacy. Decentralized trust management. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 164–173. IEEE Computer Society Press, May 1996.

[4] Sharon Boeyen, Tim Howes, and Patrick Richard. Internet X.509 Public Key Infrastructure LDAPc2 Schema. IETF RFC 2587, June 1999.

[5] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *Advances in Cryptology: EUROCRYPT ’00*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444. Springer, May 2000.

[6] Robert Bradshaw, Jason Holt, and Kent Seamons. Concealing complex policies with hidden credentials. In *Proceedings of 11th ACM Conference on Computer and Communications Security*, October 2004.

[7] Stefan A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, August 2000.

[8] Jan Camenisch and Els Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS ’02*, pages 21–30. ACM, nov 2002.

[9] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Advances in Cryptology: EUROCRYPT ’01*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, 2001.

[10] David Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[11] Dwaine Clarke, Jean-Emile Elien, Carl Ellison, Matt Fredette, Alexander Morcos, and Ronald L. Rivest. Certificate chain discovery in SPKI/SDSI. *Journal of Computer Security*, 9(4):285–322, 2001.

[12] Ronald Cramer and Ivan Damgård. Zero-knowledge proof for finite field arithmetic, or: Can zero-knowledge be for free? In *Advances in Cryptology: CRYPTO ’98*, volume 1462 of *Lecture Notes in Computer Science*, pages 424–441. Springer, 1998.

[13] Ronald Cramer, Matthew K. Franklin, Berry Schoenmakers, and Moti Yung. Multi-authority secret-ballot elections with linear work. In *Advances in Cryptology: EUROCRYPT ’96*, volume 1070 of *Lecture Notes in Computer Science*, pages 72–83. Springer, 1996.

[14] Claude Crépeau. Verifiable disclosure of secrets and applications (abstract). In *Advances in Cryptology: EUROCRYPT ’89*, volume 434 of *Lecture Notes in Computer Science*, pages 150–154. Springer, 1990.

[15] Giovanni Di Crescenzo, Rafail Ostrovsky, and S. Rajagopalan. Conditional oblivious transfer and timed-release encryption. In *Advances in Cryptology: EUROCRYPT ’99*, volume 1592 of *Lecture Notes in Computer Science*, pages 74–89, March 1999.

[16] Ivan Damgård and Eiichiro Fujisaki. An integer commitment scheme based on groups with hidden order. In *Advances in Cryptology: ASIACRYPT ’02*, volume 2501 of *Lecture Notes in Computer Science*, pages 125–142. Springer, December 2002.

[17] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.

[18] Glenn Durfee and Matt Franklin. Distribution chain security. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*, pages 63–70. ACM Press, 2000.

[19] Carl Ellison, Bill Frantz, Butler Lampson, Ron Rivest, Brian Thomas, and Tatu Ylonen. SPKI certificate theory. IETF RFC 2693, September 1999.

[20] Stephen Farrell and Russell Housley. An internet attribute certificate profile for authorization. IETF RFC 3281, April 2002.

[21] Marc Fischlin. A cost-effective pay-per-multiplication comparison method for millionaires. In *CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology*, volume 2020 of *Lecture Notes in Computer Science*, pages 457–472. Springer, 2001.

[22] Keith B. Frikken, Mikhail J. Atallah, and Jiangtao Li. Hidden access control policies with hidden credentials. In *Proceedings of the 3rd ACM Workshop on Privacy in the Electronic Society*, October 2004.

[23] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology: CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 1997.

[24] Juan Garay, Philip MacKenzie, and Ke Yang. Efficient and universally composable committed oblivious transfer and applications. In *Theory of Cryptography, TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 297–316. Springer, 2004.

[25] Oded Goldreich. Secure multi-party computation, October 2002.

[26] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game. In *Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 218–229, May 1987.

[27] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18:186–208, feb 1989.

[28] Jason E. Holt, Robert W. Bradshaw, Kent E. Seamons, and Hilarie Orman. Hidden credentials. In *Proceedings of the 2nd ACM Workshop on Privacy in the Electronic Society*, October 2003.

[29] Russell Housley, Warwick Ford, Tim Polk, and David Solo. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. IETF RFC 2459, January 1999.

[30] JCSI. Java cryptographic secure implementation. Wedgetail Communications, 2004.

[31] Ninghui Li, Wenliang Du, and Dan Boneh. Oblivious signature-based envelope. In *Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing (PODC)*. ACM Press, July 2003.

[32] Ninghui Li, Wenliang Du, and Dan Boneh. Oblivious signature-based envelope. *Distributed Computing*, 2005. To appear.

[33] Ninghui Li, John C. Mitchell, and William H. Winsborough. Design of a role-based trust management framework. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 114–130. IEEE Computer Society Press, May 2002.

[34] Ninghui Li, William H. Winsborough, and John C. Mitchell. Distributed credential chain discovery in trust management. *Journal of Computer Security*, 11(1):35–86, February 2003.

[35] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In *Selected Areas in Cryptography, 6th Annual International Workshop, SAC '99*, volume 1758 of *Lecture Notes in Computer Science*, pages 184–199. Springer, 1999.

[36] Wenbo Mao. Guaranteed correct sharing of integer factorization with off-line shareholders. In *Public Key Cryptography: PKC'98*, volume 1431 of *Lecture Notes in Computer Science*, pages 60–71. Springer, February 1998.

[37] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *Proceedings of SODA 2001 (SIAM Symposium on Discrete Algorithms)*, pages 448–457, January 2001.

[38] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology: CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

[39] Eric Rescorla. *SSL, TLS: Designing, and Building Secure Systems*. Addison-Wesley, 2001.

[40] Ronald L. Rivest and Bulter Lampson. SDSI — a simple distributed security infrastructure, October 1996. Available at http://theory.lcs.mit.edu/∼rivest/sdsi11.html.

[41] Wen-Guey Tzeng. Efficient 1-out-n oblivious transfer schemes. In *PKC '02: Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems*, number 2274 in Lecture Notes in Computer Science, pages 159–171. Springer, 2002.

[42] William H. Winsborough and Ninghui Li. Towards practical automated trust negotiation. In *Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks (Policy 2002)*, pages 92–103. IEEE Computer Society Press, June 2002.

[43] William H. Winsborough, Kent E. Seamons, and Vicki E. Jones. Automated trust negotiation. In *DARPA Information Survivability Conference and Exposition*, volume I, pages 88–102. IEEE Press, January 2000.

[44] Marianne Winslett, Ting Yu, Kent E. Seamons, Adam Hess, Jared Jacobson, Ryan Jarvis, Bryan Smith, and Lina Yu. Negotiating trust on the web. *IEEE Internet Computing*, 6(6):30–37, November/December 2002.

[45] Andrew C. Yao. How to generate and exchange secrets. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science*, pages 162–167. IEEE Computer Society Press, 1986.

[46] Ting Yu, Marianne Winslett, and Kent E. Seamons. Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. *ACM Transactions on Information and System Security (TISSEC)*, 6(1):1–42, February 2003.