CERIAS Tech Report 2004-11

OACERTS: OBLIVIOUS ATTRIBUTE CERTIFICATES

by Jiangtao Li, Ninghui Li

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

OACerts: Oblivious Attribute Certificates

Jiangtao Li and Ninghui Li
CERIAS and Department of Computer Sciences, Purdue University
250 N. University Street, West Lafayette, IN 47907
{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.

To build 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 predicates such as $=, \geq, \leq, >, <, \neq$ as well as logical combinations of them.

1 Introduction

Access control presents difficult problems in open distributed environments such as the Internet, particularly when resources and requesters belong to different security domains controlled by different authorities. Many commonly used access control mechanisms make access decisions based on the identity of the requester. However, when the resource owner and the requester are unknown to each other, access control based on the requester's identity may be ineffective. An alternative is to grant resources based on the characteristics of the requester that may be more relevant than his identity, such as age, employer, credit status, or security clearance. We call this approach attribute-based access control (ABAC) [20, 10, 29, 28, 41]. In ABAC systems, access decisions are based on attributes of the requester, which are established by digitally signed certificates through which certificate issuers assert their judgements about the attributes of entities. 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.

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. For example, a digital driver license may have fields for an identification number, expiration date, name, address, date of birth, etc. Often times, only partial information of some attributes needs to be revealed to obtain a service. Consider the following scenario: a senior citizen Bob requests from a service provider 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. What is the minimal amount of information Bob has to reveal to get free access? It might seem that Bob needs to reveal at least the fact that he is a senior citizen. However, even this seemingly minimal amount of information disclosure can be avoided. Suppose 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 the service provider and Bob, Bob obtains the key if and only if the

birth-date in his driver license is before a certain date (thus proving that Bob is a senior citizen) and the service provider learns nothing about Bob's birth-date. Under these conditions, the service provider can perform access control based on Bob's attribute values while protecting Bob's attribute information.

In 2003, three groups of researchers independently proposed schemes that can use a certificate in an oblivious fashion similar to the way described above. These schemes are Oblivious Signature Based Envelope [27], Hidden Credentials [25], and Secret Handshakes [1]. In these schemes, a service provider does not learn whether a user has a certificate or not. However, all these schemes require the service provider to know the content of the requester's certificate (which includes the attribute values), either by guessing or by having the requester send the content while withholding the signature. The service provider does not know whether the user has a signature on the content or not. This is acceptable for attributes that have binary values, i.e., one either has the attribute (e.g., a secret clearance) or does not have it. These schemes do not work well when an attribute (e.g., birth-date) can take many values and the access policy is a predicate on the attribute value. Some of these schemes also have other limitations. We discuss these schemes and their limitations in more details in Section 2.

In this paper we propose Oblivious Attribute Certificates (OACerts), a scheme for using certificates to document sensitive attributes. Using OACerts, a user can select *which* attributes to use as well as *how* to use them. An attribute can be used through one of the following three methods.

1. A user sends the attribute value to the service provider, revealing the attribute information completely.
2. A user proves in a zero-knowledge fashion that the attribute value satisfies the policy set by the service provider without revealing the actual attribute value. The service provider learns only that the attribute value satisfies the policy and nothing else.
3. A user can use the attribute value *obliviously*, i.e., the user runs a protocol with the service provider such that the user obtains the service if and only if the attribute value satisfies the policy set by the service provider, yet the service provider learns nothing about the attribute value, not even whether the value satisfies the policy.

The cryptographic tools that we use in building OACerts are commitment schemes, zero-knowledge proof of assertions about committed values, and Oblivious Commitment Based Envelope (OCBE). The first two exist in literature [32, 14, 12, 21, 17, 31, 19]; OCBE is a novel cryptographic primitive introduced in this paper.

Informally, a commitment scheme enables a prover to commit a value to a verifier such that the verifier does not know which value has been committed, and the prover cannot change its mind after having committed. We use commit to denote the commitment algorithm of a commitment scheme. To be secure, a commitment scheme cannot be deterministic; thus a commitment of a value a also depends on a secret random value r. We use $c = \mathsf{commit}(a, r)$ to denote a commitment of a. For some commitment schemes, zero-knowledge proof protocols exist to prove that the committed value satisfies some properties.

The basic idea of OACerts is quite simple. Instead of storing attribute values directly in the certificates, we store the commitments of these values in the certificates. OACerts can be easily integrated into current standards for public-key certificates such as X.509 [3, 26]. For example, the commitments can be stored in X.509v3 extension fields. The distribution and revocation of OACerts can be handled using existing infrastructure and techniques.

The first two methods of disclosing attribute values can be achieved using existing techniques. However, to allow the third (oblivious) method of using attributes, we need to solve the following 2-party Secure Function Evaluation (SFE) problem:

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

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

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 (as Alice sees a constant 0) and Bob learns M only when his committed attribute value satisfies the predicate Pred.

The SFE problem can be solved using general solutions to 2-party SFE [44, 22]; however, the general solutions are inefficient. We propose an OCBE scheme that solves the above 2-party SFE problem efficiently. Formal definition of OCBE is given in Section 5.1. Informally, an OCBE scheme enables a sender Alice to send an envelope (encrypted message) to a receiver Bob, and has the following properties: Bob can open the envelope if and only if his committed value satisfies the predicate. An OCBE scheme is *secure against the receiver* if a receiver whose committed value does not satisfy the predicate cannot open the envelope. An OCBE scheme is *oblivious* if at the end of the protocol the sender cannot tell whether the receiver's committed value satisfies the predicate or not.

We develop OCBE protocols for the Pedersen commitment scheme [32] and six kinds of comparison predicates: $=, \neq, <, >, \leq, \geq$. Each comparison predicate Pred has a parameter a_0 and an input a; for example, when Pred is $\geq_{a_0}$, $\mathsf{Pred}(a)$ is true if $a \geq a_0$. These predicates seem to be the most useful ones for testing attribute values in access control policies. We present a protocol EQ-OCBE for equality predicates and two protocols GE-OCBE and GE-OCBE2 for greater-than-or-equal-to predicates and show that these protocols are provably secure in the random oracle model [2]. We also show that it is easy to construct OCBE protocols for other comparison predicates using variants of EQ-OCBE, GE-OCBE, GE-OCBE2 as well as for conjunctions and disjunctions of multiple predicates. Designing OCBE protocols for more sophisticated predicates such as testing whether a committed number is a prime, or whether several committed values satisfy some linear relation, is beyond the scope of this paper. We also implemented EQ-OCBE, GE-OCBE, and GE-OCBE2 in Java and tested their performance.

The contributions of this paper are as follows.

- We introduce the OACerts scheme, which overcomes some limitations of previous schemes to use certified attributes in an oblivious fashion.
- We introduce OCBE, a cryptographic primitive that enables OACerts. OCBE may be of independent interest in other applications as well.
- We present efficient and provably secure OCBE protocols for the Pedersen commitment scheme [32] and several kinds of comparison predicates.

The rest of this paper is organized as follows. We begin with discussion of related work in Section 2. In Section 3, we present the architecture of OACerts and discuss the applications of OACerts. We then review the Pedersen commitment scheme and two associated zero-knowledge proof protocols in Section 4. In Section 5, we present a formal definition of OCBE and several OCBE protocols. We describe our implementation and performance measurements in Section 6. We conclude our paper in Section 7. An appendix includes proofs for all theorems in the paper.

2 Related Work

The notion of OCBE is closely related to the notion of Oblivious Signature-Based Envelope (OSBE) introduced by Li et al. [27]. In OSBE, the content of a certificate is assumed to be non-sensitive (as anyone can come up with it) and only the signature is considered to be sensitive (as only the CA can generate the signature). Bob sends to Alice the content of his certificate or a certificate he would have if he has the attribute, and Alice runs

an OSBE protocol with Bob, sending an encrypted envelope to Bob such that Bob can open the envelope if and only if he has the signature on the content he sent earlier. Li et al. [27] developed an OSBE protocol for RSA signatures [36] and gave a general construction for any signature scheme corresponding to an Identity-Based public-key Encryption (IBE) scheme [39]. Based on the general construction, they gave OSBE protocols for BLS signatures [5], which correspond to the Boneh-Franklin IBE scheme [4], and for Rabin signatures [33], which correspond to the Cocks IBE scheme [11].

To use OSBE, Bob has to send to Alice the content of a certificate he has or would have. This works fine when the certificate proves that Bob has an attribute that is binary, e.g., Bob has top secret clearance or is a member of an organization. However, when Bob wants to use his birth-date information in an oblivious fashion, OSBE does not work well, as the certificate content contains Bob's birth-date in clear. To use OSBE for this purpose, Bob has to run many OSBE protocols with Alice using different dates as his birth-date, hiding his actual birth-date "in the crowd". This is very inefficient.

Holt et al. [25] proposed a Hidden Credentials system to protect sensitive attributes. The basic idea underlying Hidden Credentials is that the Boneh-Franklin IBE scheme [4] gives rise to an OSBE scheme (although it was not called OSBE in [25]) for the BLS signatures [5]. Holt et al. [25] also observed that when a signature scheme derived from an IBE scheme is used to sign a certificate, then the certificate content can be used as a public encryption key such that the signature is the corresponding decryption key. Assuming the content of a Hidden Credential can be guessed, one can start communication by sending an encrypted message such that the other party can derive the message only when using the correct credential to decrypt. (The receiver may have to try all credentials it has in order to see which one can decrypt the encrypted message.)

Hidden Credentials can be used only when the content of credentials can be guessed. When a credential contains a validity period and/or a serial number, as all existing public-key certificate standards mandate, guessing the content becomes very difficult (if not impossible). Therefore, the Hidden Credentials scheme cannot work with existing security standards. Furthermore, similar to OSBE, Hidden Credentials cannot be used when attributes may take many values, such as the birth-date attribute.

Balfanz et al. [1] proposed a construct called Secret Handshakes using pairings that are also the foundation of the Boneh-Franklin IBE scheme [4] and the corresponding BLS signatures [5]. The Secret Handshakes scheme uses the pairing-based key-agreement protocol by Sakai et al. [37]. In Secret Handshakes, each party receives a credential from a central authority; the credential consists of a pseudonym and a corresponding secret, which can be viewed as a signature of the pseudonym together with an attribute string, signed using the central authority's master secret. The possession of the credential proves that one has the attribute documented in the attribute string. When Alice and Bob meet, they exchange the pseudonyms and each computes a key based on their own secret, and the other party's pseudonym and attribute string. The keys they compute agree only when they both have the correct credentials.

Similar to OSBE and Hidden Credentials, the Secret Handshakes scheme also requires one party to know the other party's attribute string in order to use the secret value in an oblivious way. Furthermore, unlike OSBE and Hidden Credentials, the Secret Handshakes scheme requires Alice and Bob to use credentials issued by the same authority; this further limits its applicability.

Our work is also closely related to anonymous credentials [9, 6, 30, 8, 7]. 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. [6]. These schemes differ from OACerts in that they provide orthogonal privacy protections. None of the existing anonymous credential schemes has the oblivious usage feature the OACerts scheme has. Using anonymous credentials, Alice still learns whether Bob's attribute satisfies her policies. 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. Otherwise, one can link transactions together using information such as the IP address of the user. While several protocols for anonymous communications have been proposed [34, 40], none is widely adopted. On the other hand, the OACerts scheme does not

provide anonymity protection and does not require anonymous communication channels. Finally, anonymous credential schemes tend to involve protocols dramatically different from existing public-key infrastructure standards, e.g., the anonymous credential system in Camenisch et al. [8, 7]. 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.

Both the anonymity property of anonymous credentials and the oblivious property of OACerts are limited to transactions where services can be delivered in digital form. When services have to delivered physically, e.g., a product needs to be shipped to a physical address, such high levels of privacy protection are not possible. In anonymous credentials, the shipping address can be used to link transactions together. In OACerts, the service provider Alice knows whether Bob satisfies her policies or not as she needs to know whether to ship the product or not. Observe that in this case, both schemes still offer a higher level of privacy protection than standard certificates, as one can prove that an attribute satisfies a property without revealing any other information.

Crescenzo et al. [15] introduced a variant of Oblivious Transfer called Conditional Oblivious Transfer, 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 oblivious transfer 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 Oblivious Transfer where the predicate is greater-than-or-equal-to ($\geq$). OCBE can be viewed as another special case of the Conditional Oblivious Transfer problem; however, the solutions in [15] do not apply. In OCBE, Alice's input is a commitment, the predicate is that Bob's input must be the value he committed in Alice's input and furthermore Bob's input must satisfy some property (e.g., greater than a certain value). The additional requirement about the commitments makes our protocols quite different from the ones in [15].

3 Architecture and Applications of OACerts

In this section, we present the architecture of OACerts and discuss the applications of OACerts.

3.1 Architecture of OACerts

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. One entity may serve as a CA, a certificate holder, or a service provider in different settings.

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 $c_i = \mathsf{commit}(v_i, r_i)$ be the commitment of attribute value v_i for $1 \leq i \leq m$ with r_i being the secret random. The attribute part of the certificate consists of a list of m entries, each entry is an pair $(attr_i, c_i)$. 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.

OACerts can be implemented on existing public-key infrastructure standards, such as X.509 [3, 26]. The commitments can be stored in X.509v3 extension fields, in which case a certificate also includes the following fields: serial number, validity period, issuer name, user name, certificate holder's public key, and so on.

There are four basic protocols in the OACerts scheme:

- **CA Initialization:** A CA picks up a signature scheme Sig with a public-private key pair $(K_{\mathrm{CA}}, K_{\mathrm{CA}}^{-1})$. The CA also picks a commitment scheme commit with public parameters CP. The public parameters of the

CA are $\{\mathsf{Sig}, K_{\mathrm{CA}}, \mathsf{commit}, \mathsf{CP}\}$. This is different from tradition PKI systems where the public parameters have only $\{\mathsf{Sig}, K_{\mathrm{CA}}\}$.

- **Issue Certificate:** A CA uses this protocol to issue an OACert to a user. A user Bob sends his public key K_B and attributes information $(attr_1, v_1), \ldots, (attr_m, v_m)$ to the CA. After the CA verifies the correctness of $v_1, \ldots, v_m$ (using physical methods), it issues an OACert for Bob. In this process, it computes $c_i = \mathsf{commit}(v_i, r_i)$ and sends the certificate along with the secrets $r_1, \ldots, r_m$ to Bob. Bob keeps $(v_1, r_1), \ldots, (v_m, r_m)$ with his private key K_B^{-1} secret.

- **Show Certificate:** 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 [26]). 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 [35].

- **Show Attribute:** 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 protocols. Often times, it has similar or less amount of computation as the zero-knowledge show protocols.

3.2 Applications of OACerts

The OACerts scheme enables oblivious access control, an service provider can perform access control on resources without learning any information about the attributes of requesters.

OACerts can also be used in other settings. One of the original motivations for introducing OSBE [27] was to break policy cycles in Automated Trust Negotiation [42, 41, 43, 45]. The following scenario was described in [27]: user Alice has a certificate showing that she has top-secret clearance. To protect herself, Alice will only present the certificate to other parties who also have a top-secret clearance certificate. Similarly, user Bob has a top-secret certificate and he will only reveal his certificate to others who are top-secret clearance. When Alice and Bob wish to establish a secure session using automated trust negotiation techniques, neither one is willing to present his/her certificate first. Consequently, they are stuck and cannot establish the session.

Such policy cycles and those cycles involving predicates on attribute values may be broken using OACerts and OCBE. Suppose both Alice and Bob have OACerts and security clearance is an attribute, Alice and Bob can first exchange their certificates, then Bob uses OCBE scheme to send Alice the zero-knowledge proof of his

top-secret clearance on the condition that Alice can open it only if her attribute in OACert is top-secret. Bob is certain that his security clearance attribute is revealed to Alice only if Alice has top security clearance.

4 A Commitment Scheme and Zero-Knowledge Proofs of Certain Problems

In this section, we first review the Pedersen commitment scheme [32], which we use in the OCBE protocols and OACerts. We then review two zero-knowledge proof protocols [12, 14, 31, 21, 19] that prove certain properties of values committed under the Pedersen commitment scheme.

Definition 1 (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 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 $x \leftarrow \mathbb{Z}_q$, let $h = (g^x \bmod p)$. 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 integer $a \in \mathbb{Z}_q$, 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 have a trusted third party to generate the parameters, because this is done by a CA in the OACerts scheme.

The above 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 given any commitment c every value a is equally likely to be the value committed in c. This commitment scheme is *computationally binding*: Under the discrete logarithm 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 one finds a' and r' such that $g^{a'} h^{r'} \equiv g^a h^r (\bmod\, p)$, then he 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.

Since the domain of the Pedersen commitment scheme is integers in $\mathbb{Z}_q$, it is necessary to map an arbitrary attribute value to an integer in the OACerts scheme. For example in a digital driver license, gender can be expressed by a single bit, state can be expressed by a number from $[1, 50]$, birth-date can be expressed by the number of days between January 1st of 1900 and the date of birth. For an attribute value that cannot be represented by a number such as home address, the CA can hash the attribute using a collision-free hash function.

Recall that in the OACerts scheme, a certificate holder needs to be able to prove that an attribute in the certificate satisfies some property without revealing the actual value. We here review two classic zero-knowledge proofs for values committed using the Pedersen commitment scheme. One protocol proves that a committed value is a bit; the other protocol proves that a committed value belongs to an interval. We present these protocols in detail because our OCBE schemes are built on these protocols by adding the oblivious feature.

Protocol 1 (Bit Proof Protocol) Let $\langle p, q, g, h \rangle$ be the public parameters and c is a commitment of a. The prover proves to the verifier that a is from set $\{0, 1\}$ without revealing a.

The bit proof protocol has appeared in several places [14, 31]. The basic idea is to show either $c = gh^r$ or $c = h^r$ without revealing which one is the case. In other words, the prover proves that he knows either $\log_h(c)$ or $\log_h(c/g)$. This is done using Schnorr's proof of knowledge of a discrete logarithm [38] and proofs of partial knowledge [13]. Recall that to prove one knows r such that $h^r = x$ using the Schnorr protocol, one chooses a

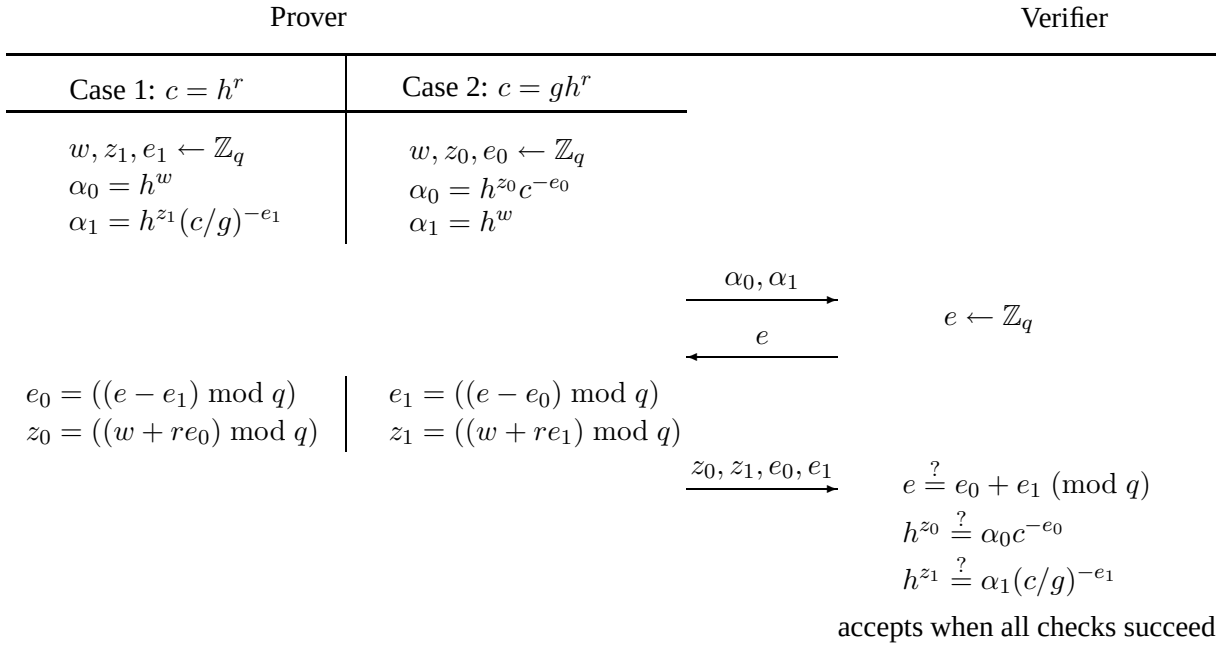


Figure 1: Bit proof protocol with public parameters $\langle p, q, g, h\rangle$. All computations are modulo p unless explicitly specified.

random w and sends $\alpha = h^w$ to the verifier, who then challenges with a random e. The prover sends $z = w + re$ to the verifier, which checks whether $\alpha = h^z x^{-e}$. If the prover can predict the challenge, the prover can cheat by picking a random z first and sending $\alpha = h^z x^{-e}$ in the first message. In the bit-proof protocol, the prover knows either $\log_h(c)$ or $\log_h(c/g)$. He uses the Schnorr protocol to prove the one he knows and cheats on the other one. The protocol is presented in Figure 1.

Correctness of this protocol is discussed in Appendix A.1. We observe that this protocol requires three rounds, the prover does three exponentiations, and the verifier does four exponentiations.

Protocol 2 (Range Proof Protocol) Let $\langle p, q, g, h\rangle$ be the public parameters and c is a commitment of a. The prover proves to the verifier that a belongs to the interval $[0..2^\ell - 1]$ without revealing the actual a.

This range proof appeared in [31, 19]. Let $c = \mathsf{commit}(a, r) = g^a h^r$ be the commitment of $a \in [0..2^\ell - 1]$ with secret random $r \in \mathbb{Z}_q$. Let $a_{\ell-1} \ldots a_1 a_0$ be the binary representation of a, i.e., $a = a_0 2^0 + a_1 2^1 + \cdots + a_{\ell-1} 2^{\ell-1} = \sum_{i=0}^{\ell-1} a_i 2^i$. The prover picks $r_1, \ldots, r_{\ell-1} \leftarrow \mathbb{Z}_q$ and computes $r_0 = r - \sum_{i=1}^{\ell-1} 2^i r_i$. The prover computes $c_i = \mathsf{commit}(a_i, r_i) = g^{a_i} h^{r_i}$ for $i = 0, 1, \ldots, \ell - 1$ and sends them to the verifier. Then the prover runs the bit proof protocol with the verifier to prove that a_i is either 0 or 1 for $0 \le i \le \ell - 1$. Finally, the verifier checks that $\prod_{i=0}^{\ell-1} (c_i)^{2^i} = c$.

If c is a commitment of $a \in [0..2^\ell - 1]$ and both parties follow the protocol, then the verifier is convinced, because

$$\prod_{i=0}^{\ell-1} (c_i)^{2^i} = \prod_{i=0}^{\ell-1} (g^{a_i} h^{r_i})^{2^i} = g^{\sum_{i=0}^{\ell-1} a_i 2^i} h^{\sum_{i=0}^{\ell-1} r_i 2^i} = g^a h^r = c \pmod p$$

If the prover is able to convince the verifier, then for each c_i, $0 \le i \le \ell - 1$, the prover knows $a_i \in \{0, 1\}$ and r_i such that $c_i = g^{a_i} h^{r_i}$. The prover thus knows $a' = \sum_{i=0}^{\ell-1} a_i 2^i$ and $r' = \sum_{i=0}^{\ell-1} r_i 2^i$ such that $c = g^{a'} h^{r'}$. Assuming that Discrete Logarithm is hard, $a' = a$ and $r' = r$; thus a is in the range $[0..2^\ell - 1]$.

The range proof protocol runs ℓ bit proof protocol instances. It takes three rounds when these instances are run in parallel. Overall, the prover does about 4ℓ exponentiations (computing each c_i amounts to about 1

exponentiation), and the verifier does about 4ℓ exponentiations plus $\frac{\ell(\ell+1)}{2}$ multiplications. When ℓ is about 20 (which is sufficient to represent birth-dates) and q is 160-bit, then computing $\frac{\ell(\ell+1)}{2}$ multiplications takes about the same time as several exponentiations. The range proof protocol can be used to prove that a committed value is greater-than-or-equal-to another value. The basic idea for doing this is used in our GE-OCBE protocol in Section 5.3.

5 Oblivious Commitment-Based Envelope (OCBE)

The OACerts scheme requires OCBE schemes, which enable oblivious show of attributes in OACerts.

5.1 Definition of OCBE

In this section, we give a formal definition of OCBE. We use the following terminology. 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*. An *adversary* is a probabilistic interactive Turing Machine [23].

Definition 2 (OCBE) An Oblivious Commitment-Based Envelope (OCBE) scheme is parameterized by a commitment scheme commit. An OCBE scheme involves a sender S, a receiver R, and a trusted third party T, and has the following four phases:

Setup The Setup algorithm takes a security parameter t and outputs public parameters CP 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 $\mathsf{commit}_{\mathsf{CP}}$ contains $\mathcal{V}$ as a subset.

The sender, the receiver, and the trusted third party share CP and the descriptions of $\mathcal{V}$ and $\mathcal{P}$.

Pre-interaction The sender chooses a message $M \in \{0,1\}^*$. The receiver chooses a value $a \in \mathcal{V}$. The sender and the receiver agree[1] on a predicate $\mathsf{Pred} \in \mathcal{P}$. The trusted third party T computes the commitment $c = \mathsf{commit}_{\mathsf{CP}}(a, r)$ where a is the committed value and r is a random number. T gives r and c to the receiver, and c to the sender.[2]

The sender S has Pred, c, and M. The receiver R has Pred, c, a, and r.

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 true, R outputs the message M. Otherwise, R does nothing.

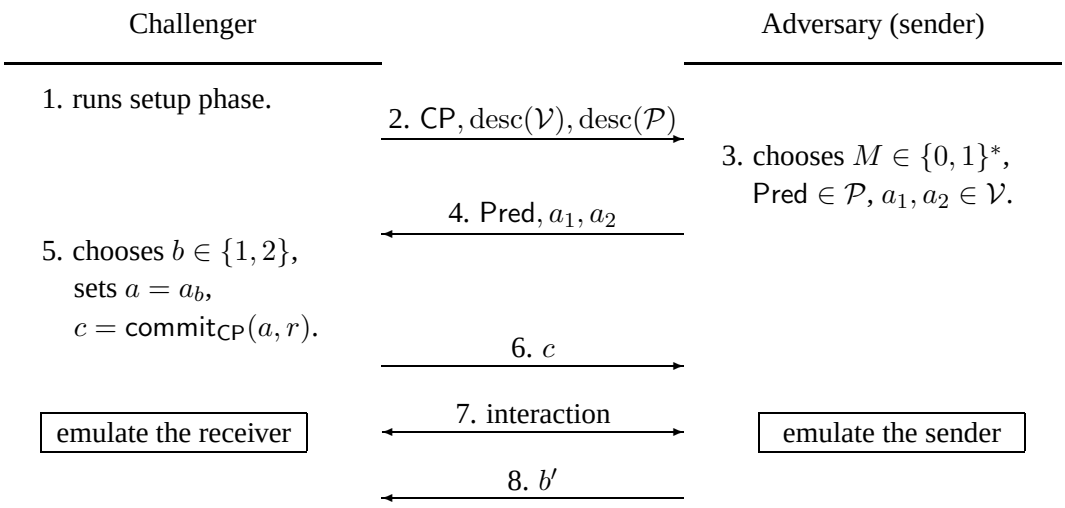
An OCBE scheme must satisfy the following three properties. It must be sound, oblivious, and semantically secure against the receiver.

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

Oblivious An OCBE scheme is *oblivious* if the sender S learns nothing about a. More precisely, no adversary $\mathcal{A}$ has a non-negligible advantage against the Challenger in the game described in Figure 2. In other words, an OCBE scheme is *oblivious* if for every probabilistic interactive Turing Machine $\mathcal{A}$, $|\Pr[\mathcal{A} \text{ wins the game in Figure 2}] - \frac{1}{2}| \leq f(t)$, where f is a negligible function in t. (The adversary cannot do substantially better than random guessing.)

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

[2] Here the commitment c is computed by T because this is done by a CA in the OACerts scheme.



Adversary wins the game if $b = b'$.

Figure 2: The attacker game for the oblivious property of OCBE. We use $\mathrm{desc}(\mathcal{V})$ to denote the description of $\mathcal{V}$ and $\mathrm{desc}(\mathcal{P})$ for the description of $\mathcal{P}$. We allow the adversary to pick a predicate Pred and two attribute values a_1, a_2 of its choice; yet the adversary sill should not be able to distinguish a receiver with attribute a_1 from one with attribute a_2.

Semantically secure against the receiver An OCBE scheme is *semantically secure against the receiver* if R learns nothing about M when $\mathsf{Pred}(a)$ is false. More precisely, no adversary $\mathcal{A}$ has a non-negligible advantage against the Challenger in the game described in Figure 3.

We assume that the interaction phase of OCBE 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 during the show certificate protocol described in Section 3.

In our proofs, we use the random oracle model, which 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 $H\colon X \to Y$ chosen uniformly at random from the set of all functions $\{h\colon X \to Y\}$ (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. Note that security in the random oracle model does not imply security in the real world. Nevertheless, the random oracle model is a useful tool for validating natural cryptographic constructions.

5.2 EQ-OCBE: an OCBE protocol for equality predicates

In this section, we present an OCBE protocol (EQ-OCBE) for the Pedersen commitment scheme with equality predicates. Our EQ-OCBE protocol runs a Diffie-Hellman style key-agreement protocol [18]. If the committed value a is equal to a_0, then R can derive the shared secret. If the committed value a is not equal to a_0, then R cannot derive the shared secret.

Definition 3 (EQ-OCBE) Let $\mathcal{E}$ be a semantically secure symmetric encryption scheme. Let H be a function (e.g., a cryptographic hash function) that extracts a key for $\mathcal{E}$ from a shared secret.

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

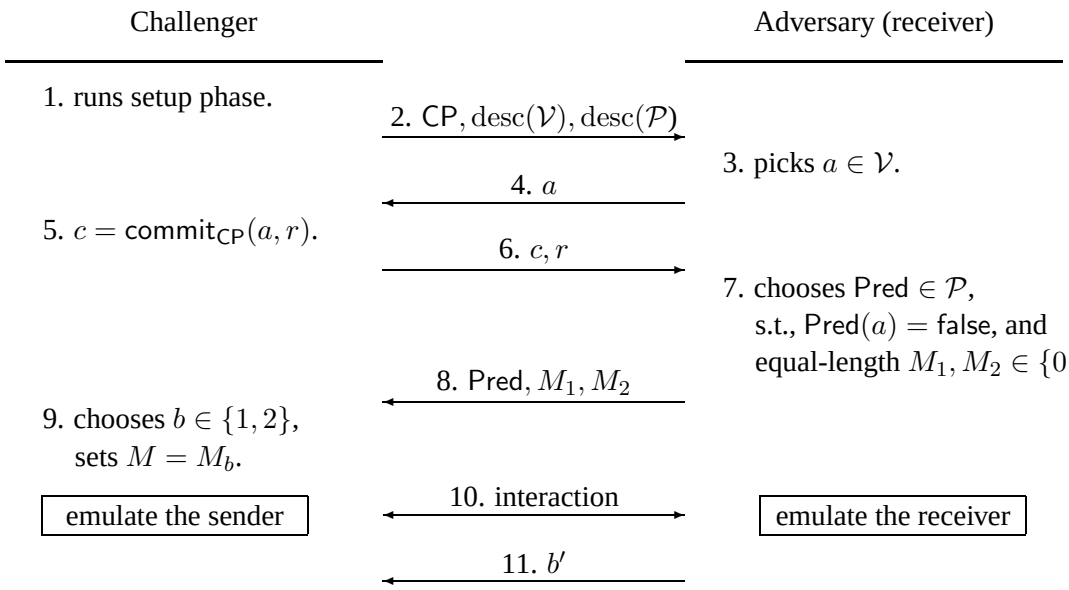


Figure 3: The attacker game for OCBE's semantic security property against the receiver. Even if we give the adversary the power to pick two equal-length messages M_1 and M_2 of its choice, it still cannot distinguish an envelope containing M_1 from one containing M_2. This formalizes the intuitive notion that the envelope leaks no information about its content.

Pre-interaction The sender chooses a message $M \in \{0,1\}^*$. The receiver chooses an integer $a \in \mathcal{V}$. The sender and the receiver agree on a predicate $\mathsf{EQ}_{a_0} \in \mathcal{P}$. The trusted third party T picks $r \leftarrow \mathbb{Z}_q$ and computes the commitment $c = (g^a h^r \mod p)$. T gives r and c to the receiver, and c to the sender.

The sender S has EQ_{a_0}, c, and M. The receiver R has EQ_{a_0}, c, a, and r.

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

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

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

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

Therefore S and R share the same symmetric key.

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

The key idea of EQ-OCBE is that if R's committed value a is equal to a_0, S can compute $cg^{-a_0} = g^{a-a_0}h^r = h^r \pmod p$. S now holds h^r such that R knows the value r. This achieves half of the Diffie-Hellman key-agreement protocol [18], with h as the base. S then does the other half by sending h^y to R. Now both R and S can compute h^{ry}. If R's committed value a is not equal to a_0, then it is presumably hard for R to compute $\log_h(cg^{-a_0})$. The reason is if R is able to find a number $r' = \log_h(cg^{-a_0})$, R can effectively 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.*

The proofs for this theorem as well as all other theorems are in Appendix A.

EQ-OCBE does a Diffie-Hellman style key agreement that has the added twist that one party can recover the shared key only when the committed value a is equal to a_0. We base the security of EQ-OCBE on the hardness of the CDH (Computational Diffie-Hellman) problem in $\mathbb{Z}_p^*$. The CDH problem is the following: given a finite cyclic group G, a generator $g \in G$, and group elements g^a, g^b, find g^{ab}. The difficulty of this problem is the security foundation of Diffie-Hellman key-agreement protocol and many other protocols. The *CDH assumption* is that there exists no polynomial probabilistic algorithm that can solve the CDH problem.

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

5.3 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 $\langle p, q, g, h\rangle$ be the parameters for the Pedersen commitment scheme. 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 $d = ((a - a_0) \bmod q)$. We have $a \geq a_0$ if and only if $d \in [0..2^\ell - 1]$. Let $c = g^a h^r$ be a commitment of a such that a party R knows r, then $cg^{-a_0} = g^d h^r$ is a commitment of d that R knows how to open. Recall that the range proof protocol (Protocol 2 in Section 4) enables R to prove that d belongs to $[0..2^\ell - 1]$ without leaking any information about d, by generating ℓ new commitments $c_0, \ldots, c_{\ell-1}$, one for each of the ℓ bits of d, and conducting a bit proof (Protocol 1) for each c_i. Our GE-OCBE protocol adds a twist to this protocol. R only proves that the committed values in $c_1, \ldots, c_{\ell-1}$ are bits. For bit 0, R runs a protocol with the other party S such that R obtains a secret shared with S only when he can open c_0 as a bit, yet at the same time S learns nothing about what R committed in c_0.

Definition 4 (GE-OCBE) Let $\mathcal{E}$ be a semantically secure symmetric encryption scheme. Let H be a function (e.g., a cryptographic hash function) that extracts a key for $\mathcal{E}$ from a shared secret.

Setup The setup algorithm takes two parameters, a security parameter t and a parameter ℓ, which specifies the desired range of the attribute values. The setup algorithm runs the commitment setup algorithm to create $\mathsf{CP} = \langle p, q, g, h\rangle$ such that $2^\ell < q/2$. It 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}: $\mathcal{V} \rightarrow \{\mathsf{true}, \mathsf{false}\}$, such that $\mathsf{GE}_{a_0}(a)$ is true if $a \geq a_0$ and false otherwise.

Pre-interaction The sender chooses a message $M \in \{0,1\}^*$. The receiver chooses an integer $a \in \mathcal{V}$. The sender and the receiver agree on a predicate $\mathsf{GE}_{a_0} \in \mathcal{P}$. The trusted third party T picks $r \leftarrow \mathbb{Z}_q$ and computes the commitment $c = (g^a h^r \bmod p)$. T gives r and c to the receiver, and c to the sender.

The sender S has GE_{a_0}, c, and M. The receiver R has GE_{a_0}, c, a, and r.

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

1. R picks $r_1, \ldots, r_{\ell-1} \leftarrow \mathbb{Z}_q$ and sets $r_0 = r - \sum_{i=1}^{\ell-1} 2^i r_i \bmod q$. 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 $d_0 = d - \sum_{i=1}^{\ell-1} 2^i d_i \bmod q$. Observe that d_0 is neither 0 nor 1 in this case.
2. R computes ℓ commitments $c_i = \mathsf{commit}_{\mathsf{CP}}(d_i, r_i) = g^{d_i} h^{r_i} \bmod p$, for $0 \leq i \leq \ell - 1$. R sends $c_0, \ldots, c_{\ell-1}$ to S.

3. For each i such that $1 \leq i \leq \ell - 1$, R proves to S that each d_i committed in c_i is either 0 or 1, using the bit proof protocol in Figure 1.
4. S verifies that $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i} \mod p$. S picks $y \leftarrow \mathbb{Z}_q^*$, computes $\sigma_0 = ((c_0)^y \mod p)$ and $\sigma_1 = ((c_0 g^{-1})^y \mod p)$. S sends to R the tuple: $\langle \eta = (h^y \mod p), C_0 = \mathcal{E}_{H(\sigma_0)}[M], C_1 = \mathcal{E}_{H(\sigma_1)}[M] \rangle$.

Open R receives $\langle \eta, C_0, C_1 \rangle$ from the interaction phase. If $\mathsf{GE}_{a_0}(a)$ is true, it computes $\sigma' = (\eta^{r_0} \mod p)$ and decrypts C_{d_0} using $H(\sigma')$ to recover M.

To see that this scheme is sound, observe that when $\mathsf{GE}_{a_0}(a)$ is true, d_0 is either 0 or 1. If $d_0 = 0$, $\sigma_0 = (c_0)^y = (g^{d_0} h^{r_0})^y = (h^y)^{r_0} = \eta^{r_0} = \sigma' (\mathrm{mod}\ p)$, R can decrypt C_0. If $d_0 = 1$, $\sigma_1 = (c_0 g^{-1})^y = (g^{d_0 - 1} h^{r_0})^y = (h^y)^{r_0} = \eta^{r_0} = \sigma' (\mathrm{mod}\ p)$, R can decrypt C_1.

The interaction phase of GE-OCBE can be done in four rounds. In the first round, R sends $c_0, \cdots, c_{\ell-1}$ to S as well as the α_0, α_1's needed for the bit proofs. In the second round, S replies with challenges for the bit proofs. In the third round, R sends the responses for the bit proofs. In the fourth round, S sends the envelope. In the interaction and open phases, both R and S conduct about 4ℓ exponentiations. The amount of computation is comparable to that in a zero-knowledge proof using the range proof protocol.

Theorem 3 *GE-OCBE is oblivious.*

Theorem 4 *Under the CDH assumption on G_q, the order-q subgroup of $\mathbb{Z}_p^*$, and assuming that H is modelled as a random oracle, GE-OCBE is secure against the receiver.*

5.4 GE-OCBE2: an alternative OCBE protocol for greater-than-or-equal-to predicates

In this section, we present another OCBE protocol (GE-OCBE2) for the Pedersen commitment scheme with greater-than-or-equal-to predicates GE. The GE-OCBE2 protocol is more efficient than the GE-OCBE protocol; however, its security property is proved based on a weaker assumption.

In GE-OCBE, we run $\ell - 1$ bit proof protocols for $c_1, \cdots, c_{\ell-1}$ and a "bit-OCBE" protocol for c_0 — if a bit is committed in c_0, then the receiver can decrypt. In GE-OCBE2, we run ℓ instances of the "bit-OCBE" protocol, one for each c_i, and the receiver can recover a key only when a bit is committed in each c_i.

Definition 5 (GE-OCBE2) Let $\mathcal{E}$ be a semantically secure symmetric encryption scheme. Let H be a function (e.g., a cryptographic hash function) that extracts a key for $\mathcal{E}$ from a shared secret. The Setup and Pre-interaction phases are same as those in the GE-OCBE protocol.

Interaction Let d denote $(a - a_0) \mod q$, $\mathsf{GE}_{a_0}(a) = \mathsf{true}$ if and only if $d \in [0..2^\ell - 1]$. Note that $cg^{-a_0} = g^d h^r (\mathrm{mod}\ p)$ 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 $d_1, d_2, \ldots, d_{\ell-1} \leftarrow \mathbb{Z}_q$, and sets $d_0 = d - \sum_{i=1}^{\ell-1} 2^i d_i \mod q$.
2. R computes ℓ commitments $c_i = \mathsf{commit}_{\mathsf{CP}}(d_i, r_i) = g^{d_i} h^{r_i} \mod p$, for $0 \leq i \leq \ell - 1$. R sends $c_0, \ldots, c_{\ell-1}$ to S.
3. S verifies that $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i} \mod p$. S randomly chooses ℓ symmetric keys $k_0, \ldots, k_{\ell-1}$ and sets $k = k_0 \oplus \cdots \oplus k_{\ell-1}$. S picks $y \leftarrow \mathbb{Z}_q^*$, computes $\eta = (h^y \mod p)$ and $C = \mathcal{E}_k[M]$. For each $0 \leq i \leq \ell - 1$, S computes $\sigma_i^0 = ((c_i)^y \mod p)$, $\sigma_i^1 = ((c_i g^{-1})^y \mod p)$, $C_i^0 = \mathcal{E}_{H(\sigma_i^0)}[k_i]$, and $C_i^1 = \mathcal{E}_{H(\sigma_i^1)}[k_i]$. 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, $a - a_0 = \sum_{i=0}^{\ell-1} 2^i d_i$ where d_i is from $\{0, 1\}$. For each $0 \leq i \leq \ell - 1$, R computes $\sigma_i' = (\eta^{r_i} \bmod p)$, and uses $H(\sigma_i')$ to decrypt $C_i^{d_i}$ and derive k_i'. Then R computes k' as $k_0' \oplus \cdots \oplus k_{\ell-1}'$. Finally, R decrypts C using k'.

To see that this scheme is sound, observe that when $\mathsf{GE}_{a_0}(a)$ is true, $d_0, \ldots, d_{\ell-1}$ are either 0 or 1. For each $0 \leq i \leq \ell - 1$, if $d_i = 0$, $\sigma_i^0 = (c_i)^y = (g^{d_i} h^{r_i})^y = (h^y)^{r_i} = \eta^{r_i} = \sigma' (\bmod p)$, R can decrypt C_i^0 and get k_i; if $d_i = 1$, $\sigma_i^1 = (c_i g^{-1})^y = (g^{d_i - 1} h^{r_i})^y = (h^y)^{r_i} = \eta^{r_i} = \sigma' (\bmod p)$, R can decrypt C_i^1 and get k_i. Because $k = k_0 \oplus \cdots \oplus k_{\ell-1}$, R can successfully reconstruct k. Therefore S and R share the same symmetric key k if $\mathsf{GE}_{a_0}(a)$ is true.

GE-OCBE2 requires two rounds whereas GE-OCBE requires four 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).

Theorem 5 *GE-OCBE2 is oblivious.*

The security of GE-OCBE2 is based on a non-standard computational assumption related to the CDH problem. The assumption was originally proposed by Damgård [16] and was also used in Hada and Tanaka [24]. The assumption, which was called DA-1 in [24], says that given randomly chosen instances of the discrete logarithm problem (p, q, g, g^q), it is infeasible to compute (B, X) such that $X = B^a \bmod p$ *without knowing* the value b satisfying $B = g^b \bmod p$.

Theorem 6 *Under the DA-1 assumption and the DL assumption (discrete logarithm is hard) on G_q, the order-q subgroup of $\mathbb{Z}_p^*$, and assuming that H is modelled as a random oracle, GE-OCBE2 is secure against the receiver.*

5.5 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). In stead of formally presenting these protocols, we briefly sketch the ideas. We use $OCBE(\mathsf{Pred}, M)$ to denote an OCBE protocol with predicate Pred, it outputs M if the predicate is true.

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 $OCBE(\mathsf{Pred}_1 \wedge \mathsf{Pred}_2, M)$ can be constructed as follows: The sender picks two random keys k_1 and k_2 and sets $k = k_1 \oplus k_2$. The sender runs $OCBE(\mathsf{Pred}_1, k_1)$ and $OCBE(\mathsf{Pred}_2, k_2)$ with the receiver. Finally, the sender sends $\mathcal{E}_k[M]$ to the receiver. The receiver can recover M only if both Pred_1 and Pred_2 are true.

2. **OR-OCBE**: An $OCBE(\mathsf{Pred}_1 \vee \mathsf{Pred}_2, M)$ can be constructed as follows: The sender picks a random key k. The sender runs $OCBE(\mathsf{Pred}_1, k)$ and $OCBE(\mathsf{Pred}_2, k)$ with the receiver. Finally, the sender sends $\mathcal{E}_k[M]$ to the receiver. The receiver can recover M if either Pred_1 or Pred_2 is true.

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

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

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 $OCBE(\geq_{a_0} \wedge \leq_{a_1}, M)$.

6 Implementation and Performance

We implemented the three kinds of show attribute protocols 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 either unsigned short or unsigned long, i.e., $\ell = 16$ or $\ell = 32$. For instance, the direct show protocol requires a certificate holder sending a and r, if the attribute value a is 32 bits, the total size of communication in that protocol is 20 bytes ($160 + 32$ bits).

In the implementation of 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 bytes 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 2.53GMz 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 different show attribute protocols is summarized in Table 1. As can be seen, the direct show protocol is most efficient and takes only 25 ms. The zero-knowledge show protocols and oblivious show protocols have similar execution times and amounts of communication. Also observe that GE-OCBE2 is more efficient than GE-OCBE and the zero-knowledge protocol for proving the $\geq$ relation; it improves the performance approximately by a factor of 2. We compare the performance of show attribute protocols on different attribute sizes in Table 2, $\ell = 32$ is roughly two-times expensive as $\ell = 16$.

	execution time	communication size
Direct Show	25 ms	24 bytes
Zero-knowledge Show (prove $a = a_0$)	28 ms	168 bytes
Zero-knowledge Show (prove $a \geq a_0$)	2.2 s	15 KB
Oblivious show (EQ-OCBE)	75 ms	144 bytes
Oblivious Show (GE-OCBE)	2.2 s	15 KB
Oblivious Show (GE-OCBE2)	0.9 s	5.1 KB

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

7 Conclusion

In this paper, we proposed 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, one can use attributes in OACerts in an oblivious fashion. We introduced Oblivious Commitment Based Envelope (OCBE) to enable

	GE-ZKShow		GE-OCBE		GE-OCBE2	
	time	size	time	size	time	size
$\ell = 16$	1.1 s	7.5 KB	1.1 s	7.4 KB	0.5 s	2.6 KB
$\ell = 32$	2.2 s	15 KB	2.2 s	15 KB	0.9 s	5.1 KB

Table 2: Compare running time and size of communication on two different ℓ, on a 2.53GMz Intel Pentium 4 running RedHat Linux. Security parameters are $p = 1024$ bits, and $q = 160$ bits.

the oblivious usage of OACerts. We 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. Our implementation showed that the OACerts scheme is practical and efficient. Future work includes investigation of applying OACerts and OCBE to Automated Trusted Negotiation and other areas.

Acknowledgement

This work is supported by NSF ITR and by sponsors of CERIAS. We would like to thank Dan Boneh for referring us to [24] for the DA-1 assumption used in the proof of security of the GE-OCBE2 protocol. We also thank Ziad Bizri, Ji-Won Byun, Klorida Miraj, and Mahesh V. Tipunitara for providing suggestions on presentation.

A Proofs

A.1 Correctness of Protocol 1

The Bit Proof protocol is zero-knowledge when e is small. Even with large e, the verifier still cannot distinguish whether the Prover committed a 0 or a 1 in c, since what the Prover sends in the two cases are drawn from the same distribution.

We now show that the protocol is indeed a proof of partial knowledge by constructing a knowledge extractor. Suppose that a prover is challenged twice on the same α_0, α_1, first with e and then with $e' \neq e$, and the prover succeeds both times. Then we have z_0, z_1, e_0, e_1 and z'_0, z'_1, e'_0, e'_1 such that

$$\begin{array}{rclrclrcl} e_0 + e_1 & = & e \pmod q & \alpha_0\, c^{-e_0} & = & h^{z_0} \pmod p & \alpha_1\, (c/g)^{-e_1} & = & h^{z_1} \pmod p \\ e'_0 + e'_1 & = & e' \pmod q & \alpha_0\, c^{-e'_0} & = & h^{z'_0} \pmod p & \alpha_1\, (c/g)^{-e'_1} & = & h^{z'_1} \pmod p \end{array}$$

Because $e \neq e'$, it has to be either $e_0 \neq e'_0$ or $e_1 \neq e'_1$. When $e_0 \neq e'_0$, one can compute $r = ((z_0 - z'_0)(e'_0 - e_0)^{-1} \bmod q)$ such that $c = h^r \pmod p$. When $e_1 \neq e'_1$, one can compute $r = ((z_1 - z'_1)(e'_1 - e_0)^{-1} \bmod q)$ such that $c = gh^r \pmod p$.

A.2 Proof of Theorem 1

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. ∎

A.3 Proof of Theorem 2

Proof. EQ-OCBE uses a semantically secure symmetric encryption algorithm. When H is modelled 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 the secret that the sender uses to derive the encryption key. More precisely, EQ-OCBE is secure against the receiver if no polynomially bounded adversary wins the following game against the Challenger with non-negligible probability (this game is instantiated from the game in Figure 3 with details from the EQ-OCBE protocol): The Challenger runs the setup phase and sends $\mathsf{CP} = \langle p, q, g, h\rangle$ and the descriptions of $\mathcal{V}$ and EQ 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 $c = (g^a h^r \mod p)$, 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 $h^y \mod p$. The adversary then outputs σ, and the adversary wins the game if $\sigma = (cg^{-a_0})^y \mod p$.

Given an attacker $\mathcal{A}$ that wins the above game with probability ϵ, we construct another attacker $\mathcal{B}$ that can solve the CDH problem in G_q, the order-q subgroup of $\mathbb{Z}_p^*$, with the same probability. $\mathcal{B}$ does the following (all arithmetic is $\mod p$):

1. $\mathcal{B}$, when given $p, q, h \in G_q, h^x, h^y$, gives $\mathsf{CP} = \langle p, q, h^x, h\rangle$ and the descriptions of $\mathcal{V} = \mathbb{Z}_q$ and $\mathcal{P} = \{\mathsf{EQ}_{a_0} \mid a_0 \in \mathcal{V}\}$ to $\mathcal{A}$.
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 $\mathcal{A}$ such that $a \neq a_0$, and sends h^y to $\mathcal{A}$.
4. $\mathcal{B}$ receives σ from $\mathcal{A}$, computes $\delta = \sigma h^{-ry}$, and outputs $\delta^{(a-a_0)^{-1} \mod q}$.

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

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

A.4 Proof of Theorem 3

Proof. Consider the game for the oblivious property of OCBE (in Figure 2), 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 $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i} \mod p$. The adversary also participates as the verifier in the bit proof protocols conducted for each c_i where $1 \leq i \leq \ell-1$. 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 $cg^{-a_0} \prod_{i=1}^{\ell-1}(c_i)^{-2^i} \mod p$. From the bit proof protocols, the adversary learns only that the Challenger is able to open c_i with either 0 or 1 for $1 \leq i \leq \ell - 1$. This does not leak any information about whether a_0 or a_1 was chosen because the Challenger can do this in either case. Thus GE-OCBE is oblivious even against an infinitely powerful adversary. ∎

A.5 Proof of Theorem 4

Proof. GE-OCBE uses a semantically secure symmetric encryption algorithm. When H is modelled as a random oracle, GE-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 the secret that the sender uses to derive the encryption key. More precisely, GE-OCBE is secure against the receiver if no polynomially bounded adversary wins the following game against the Challenger with non-negligible probability (this game is instantiated from the game in Figure 3 with details from the GE-OCBE protocol): The Challenger runs the setup phase and sends $\mathsf{CP} = \langle p, q, g, h\rangle$ and the descriptions of $\mathcal{V}$ and GE to the adversary. The adversary picks an integer $a \in \mathcal{V}$. The Challenger chooses a random $r \leftarrow \mathbb{Z}_q$ and computes the commitment of a as $c = (g^a h^r \mod p)$. 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 responds with the commitments $c_0, c_1, \ldots, c_{\ell-1}$ such that $cg^{-a_0} = \prod_{i=0}^{\ell-1}(c_i)^{2^i} \mod p$. The adversary proves that he can open $c_1, \ldots, c_{\ell-1}$ with

bits using the bit proof protocol. The Challenger then picks $y \leftarrow \mathbb{Z}_q^*$ and sends to the adversary $h^y \bmod p$. The adversary then outputs σ, and the adversary wins the game if $\sigma = ((c_0)^y \bmod p)$ or $\sigma = ((c_0 g^{-1})^y \bmod p)$.

The CDH assumption in G_q implies the assumption that discrete logarithm in G_q is hard. Under this assumption, the bit proof protocol proves that the committed value is indeed from $\{0, 1\}$ [14]. Because the bit proof protocol is a proof of partial knowledge (see Appendix A.1), the adversary has the knowledge of d_i and r_i for $1 \leq i \leq \ell - 1$. Thereby, given an adversary $\mathcal{A}$ that wins the above game with probability ϵ, we can construct a new adversary $\mathcal{A}'$ that not only wins the above game with same probability ϵ but also outputs $d_1, \ldots, d_{\ell-1}, r_1, \ldots, r_{\ell-1}$.

Given such attacker $\mathcal{A}'$ that wins the above game with probability ϵ, we construct another attacker $\mathcal{B}$ that can solve the CDH problem in G_q, the order-q subgroup of $\mathbb{Z}_p^*$ with probability $\epsilon/2$. $\mathcal{B}$ does the following (all arithmetic is $\bmod p$):

1. $\mathcal{B}$, when given $p, q, h \in G_q, h^x, h^y$, chooses a positive integer ℓ such that $2^\ell < q/2$ and gives $\mathsf{CP} = \langle p, q, h^x, h\rangle$ and the descriptions of $\mathcal{V} = [0, 2^\ell - 1]$ and $\mathcal{P} = \{\mathsf{GE}_{a_0} \mid a_0 \in \mathcal{V}\}$ to $\mathcal{A}'$.
2. $\mathcal{B}$ receives an integer $a \in [0, 2^\ell - 1]$ 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 greater-than-or-equal-to predicate GE_{a_0} from $\mathcal{A}'$ such that $\mathsf{GE}_{a_0}(a)$ is false. $\mathcal{B}$ also receives $c_0, c_1, \ldots, c_{\ell-1}, d_1, \ldots, d_{\ell-1}, r_1, \ldots, r_{\ell-1}$ from $\mathcal{A}'$ such that $d_i \in \{0, 1\}$ and $c_i = (h^x)^{d_i} h^{r_i}$ for $1 \leq i \leq \ell - 1$ and $c(h^x)^{-a_0} = \prod_{i=0}^{\ell-1} (c_i)^{2^i}$.
4. $\mathcal{B}$ sets $d_0 = a - a_0 - \sum_{i=1}^{\ell-1} d_i 2^i \bmod q$ and $r_0 = r - \sum_{i=1}^{\ell-1} r_i 2^i \bmod q$. Observe that $c_0 = c(h^x)^{-a_0} \prod_{i=1}^{\ell-1} (c_i)^{-2^i} = (h^x)^{a-a_0-\sum_{i=1}^{\ell-1} d_i 2^i} h^{r-\sum_{i=1}^{\ell-1} r_i 2^i} = (h^x)^{d_0} h^{r_0}$. Also observe that $a - a_0 \bmod q \notin [0..2^{\ell-1}]$ and $d_i \in \{0, 1\}$ for $1 \leq i \leq \ell - 1$, thus d_0 is neither 0 nor 1.
5. $\mathcal{B}$ sends h^y to $\mathcal{A}'$. $\mathcal{B}$ receives σ from $\mathcal{A}'$, computes $\delta = \sigma h^{-r_0 y}$, and randomly outputs one of $\delta^{d_0^{-1} \bmod q}$ and $\delta^{(d_0-1)^{-1} \bmod q}$.

   When $\mathcal{A}'$ wins the game, $\sigma = (c_0)^y$ or $\sigma = (c_0 (h^x)^{-1})^y$. For the first case, $\sigma = (c_0)^y = ((h^x)^{d_0} h^{r_0})^y = (h^{xy})^{d_0} h^{r_0 y}$, then $\delta = \sigma h^{-r_0 y} = (h^{xy})^{d_0}$, $\mathcal{B}$ outputs $\delta^{d_0^{-1} \bmod q} = h^{xy}$. For the second case, $\sigma = (c_0 (h^x)^{-1})^y = ((h^x)^{d_0 - 1} h^{r_0})^y = (h^{xy})^{d_0 - 1} h^{r_0 y}$, then $\delta = \sigma h^{-r_0 y} = (h^{xy})^{d_0 - 1}$, $\mathcal{B}$ outputs $\delta^{(d_0-1)^{-1} \bmod q} = h^{xy}$.

$\mathcal{B}$ succeeds in solving the CDH problem if and only if $\mathcal{A}'$ wins the above game, i.e., successfully computes $((c_0)^y \bmod p)$ or $((c_0 g^{-1})^y \bmod p)$, and $\mathcal{B}$ picks correctly among $\delta^{d_0^{-1} \bmod q}$ and $\delta^{(d_0-1)^{-1} \bmod q}$. ∎

A.6 Proof of Theorem 5

Proof. Consider the game for the oblivious property of OCBE, let us examine what an adversary would see in the case of GE-OCBE2. The adversary sees a commitment c and ℓ commitments $c_0, \ldots, c_{\ell-1}$ such that $cg^{-a_0} = \prod_{i=0}^{\ell-1} (c_i)^{2^i} \bmod p$. Using the same reasoning in the proof for Theorem 3, $c, c_0, \ldots, c_{\ell-1}$ together do not reveal *any* information about a. Thus GE-OCBE is oblivious even against an infinitely powerful adversary. ∎

A.7 Proof of Theorem 6

Proof. Sketch. Because the encryption key k is the XOR of ℓ random symmetric keys $k_0, \ldots, k_{\ell-1}$, the adversary is able to derive the encryption key only if he can compute all these random keys. In order to get k_i, the adversary has to decrypt either C_i^0 or C_i^1. Assume H is a random oracle, the adversary has to compute either $\sigma_i^0 = c_i^y \bmod p$ or $\sigma_i^1 = (c_i g^{-1})^y \bmod p$. Under the DA-1 assumption, the adversary knows the discrete log of either c_i or $c_i g^{-1}$ for each i. Then the adversary can open cg^{-a_0} with a value $d \in [0..2^\ell - 1]$. Since the adversary can also open cg^{-a_0} with $a - a_0$, assuming discrete log is hard, $a - a_0 = d \in [0..2^\ell - 1]$. ∎

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

[4] Dan Boneh and Matt Franklin. Identity-Based Encryption from the Weil Pairing. In *Proceedings of Crypto 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.

[5] Dan Boneh, Ben Lynn, and Hovav Shacham. Short Signatures from the Weil Pairing. In *Proceedings of Asiacrypt 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–32. Springer, 2001.

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

[7] 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.

[8] 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.

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

[10] 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.

[11] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *8th IMA International Conference on Cryptography and Coding*, volume 2260, pages 360–363. Springer, December 2001.

[12] Ronald Cramer and Ivan Damgard. 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, Ivan Damgard, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Advances in Cryptology: EUROCRYPT '94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer, 1994.

[14] 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.

[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 Damgard. Towards practical public key systems secure against chosen ciphertext attacks. In *Advances in Cryptology: CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456, 1992.

[17] Ivan Damgard 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.

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

[19] 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.

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

[21] 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.

[22] 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.

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

[24] Satoshi Hada and Toshiaki Tanaka. On the existence of 3-round zero-knowledge protocols. In *Advances in Cryptology: CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 408–423, 1998.

[25] 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.

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

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

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] Michael O. Rabin. Digitalized signatures as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, January 1979.

[34] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security (TISSEC)*, 1(1):66–92, November 1998.

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

[36] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

[37] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *Proceedings of the Symposium on Cryptography and Information Security (SCIS 2000)*, January 2000.

[38] Claus P. Schnorr. Efficient identification and signatures for smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[39] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology: CRYPTO '84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.

[40] Paul F. Syverson, David M. Goldschlag, and Michael G. Reed. Anonymous connections and onion routing. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 44–54, May 1997.

[41] 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.

[42] 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.

[43] 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.

[44] 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.

[45] 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.