Efficient and Privacy-Preserving Enforcement of Attribute-Based Access Control

Ning Shang[*]
Microsoft Corporation
One Microsoft Way
Redmond, Washington
nishang@microsoft.com

Federica Paci[†]
University of Trento
Via Sommarive 14
Povo, Trento, 38123
paci@disi.unitn.it

Elisa Bertino
Purdue University
305 N. University Street
West Lafayette, Indiana
bertino@cs.purdue.edu

ABSTRACT

Modern access control models, developed for protecting data from accesses across the Internet, require to verify the identity of users in order to make sure that users have the required permissions for accessing the data. User's identity consists of data, referred to as *identity attributes*, that encode relevant-security properties of the users. Because identity attributes often convey sensitive information about users, they have to be protected. The Oblivious Commitment-Based Envelope (OCBE) protocols address the protection requirements of both users and service providers. The OCBE protocols makes it possible for a party, referred as sender, to send an encrypted message to a receiver such that the receiver can open the message if and only if its committed value satisfies a predicate and that the sender does not learn anything about the receiver's committed value. The possible predicates are comparison predicates $=, \neq, >, <, \leq, \geq$. In this paper, we present an extension that improves the efficiency of EQ-OCBE protocol, that is, the OCBE protocol for equality predicates. Our extension allows a party to decrypt data sent by a service provider if and only if the party satisfies all the equality conditions in the access control policy.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: [Security and protection]

General Terms

Security

Keywords

Identity, Privacy, Agg-EQ-OCBE

1. INTRODUCTION

Modern data access control models, developed for interactions across different domains and Internet, allow one to specify and enforce access control policies, that is, policies regulating accesses to the protected data, in terms of conditions expressed against *user identity attributes.* Because such attributes often encode relevant-security properties of the users, they have to be protected as well. The implementation of such attribute-based access control models thus requires mechanisms whereby a user obtains access to data if and only if its identity attributes satisfy the service provider[1] policy, whereas the service provider learns nothing about user's identity attributes.

Several approaches based on anonymous credentials [6, 2, 10, 4, 3] have been proposed to allow users to prove that their identity attributes satisfy conditions in the policies by the service provider without revealing the identity attributes in clear. These approaches are based on storing cryptographic commitments of attribute values in certificates and using zero-knowledge proofs protocols [5] to prove properties of these values. A major drawback of those approaches is that, even though the service provider does not learn the attribute values, it learns whether users' identity attributes satisfy its policy conditions and may thus infer information about the values of these attributes.

The Oblivious Commitment-Based Envelope (OCBE) protocols [9] is an approach that addresses such shortcoming and can thus satisfy the protection requirements of both the service providers and the users. The OCBE protocols allow a service provider to send an encrypted message, containing the protected data, to a user such that the user can open the message if and only if the committed value of a specified identity attribute satisfies a predicate. Under such protocol service provider does not learn anything about the user's committed value and does not learn whether the value satisfies the conditions in the access control policy. The possible predicates supported by OCBE are the comparison predicates, that is, $=, \neq, >, <, \leq, \geq$. A major drawback of the OCBE protocol is that it is only able to enforce a condition (consisting of a single predicate) against a single identity attribute. Therefore, if the access control policy requires verifying conditions against several identity attributes, sev-

[*] This work was done while the author was at Purdue University.

[†] This work was done while the author was at Purdue University.

[1] We use the term 'service provider' to refer to the party managing and securing the protected data.

IDtrust'10 April 13–15, 2010, Gaithersburg, MD

eral rounds of the protocol have to be carried out which results in inefficient access control. Efficient access control systems are crucial for mobile identity systems and mobile devices.

In this paper, we present the Agg-EQ-OCBE[2] protocol that addresses the efficiency issue of the EQ-OCBE protocol, that is, the OCBE protocol for equality predicates. Our approach provides an efficient approach under which the user can quickly decrypt the data, even when multiple conditions are imposed against its identity attributes. Like the original EQ-OCBE, Agg-EQ-OCBE assures user privacy in that the service provider does not learn the values of the user identity attributes nor whether these attributes verify the access control policies.

The paper is organized as follows. Section 2 reviews the EQ-OCBE protocol. Section 3 presents the Agg-EQ-OCBE protocol. In Section 4 we prove that Agg-EQ-OCBE is secure against a malicious user. Section 5 describes our implementation and performance measurements. Section 6 concludes the paper.

2. OVERVIEW OF THE EQ-OCBE PROTOCOL

We give an overview of the EQ-OCBE protocol in this section. We shall describe the protocol in a more general setting of finite abelian groups. This can be viewed as a natural extension of the originally proposed EQ-OCBE protocol [9].

The EQ-OCBE protocol is built on the Pedersen commitment scheme [12], which is described in [12] in a particular implementation using a subgroup of the multiplicative group of a finite field. Note that this is not intrinsic for the scheme. It also can be implemented using other abelian groups, e.g., elliptic curves over finite fields.

We rewrite the Pedersen commitment scheme as follows.

Definition 1. (The Pedersen Commitment Scheme)
Setup A trusted third party T chooses a finite cyclic group G of large prime order p so that the *computational Diffie-Hellman problem*[3] is hard in G. Write the group operation in G as multiplication. T chooses an element $g \in G$ as a generator, and another element $h \in G$ such that it is hard to find the discrete logarithm of h with respect to g, i.e., an integer α such that $h = g^{\alpha}$. T may or may not know the number α. T publishes G, p, g and h as the system's parameters.
Commit The domain of committed values is the set of integers $D = \{0, 1, \ldots, p-1\}$. For a party U to commit a value $x \in D$, it randomly chooses $r \in D$, and computes the commitment $c = g^x h^r \in G$.
Open U shows the values x and r to open a commitment c. The verifier checks whether $c = g^x h^r$.

The EQ-OCBE is a Diffie-Hellman-like protocol that allows the user to correctly retrieve the protected data only if the user's committed value equals the one specified by the policy of the service provider. It involves three communication parties: a user U, a service provider SP, and a trusted party T which generates initialization parameters for the protocol to use.

There are several cryptographic components in EQ-OCBE:

- A semantically secure symmetric-key encryption algorithm $\mathcal{E}$ (e.g., AES) with keyspace $\{0,1\}^k$. We use $\mathcal{E}_{\mathsf{Key}}[\mathsf{Message}]$ to denote the encrypted plaintext Message with encryption key Key under the encryption algorithm $\mathcal{E}$.
- A finite cyclic group G of large prime order p, over which the computational Diffie-Hellman problem is intractable. The group operation is written multiplicatively.
- A cryptographic hash function $H(\cdot) : G \to \{0,1\}^k$.

We shall describe how the EQ-OCBE protocol works in our case of policy enforcement for an equality condition.

Protocol 1. (EQ-OCBE)
Parameter generation
T runs a Pedersen commitment setup program to generate system parameters $\mathsf{Param} = \langle G, g, h\rangle$. T also outputs the order of G, p.
Commitment This step is a modified version of the one described in [9]. Instead of requiring T to generate the Pedersen commitment, we let U perform this procedure and ask T to verify the validity of the commitment[4] .

To commit to an element $x \in \mathbb{Z}/(p)$, U randomly chooses $r \in \mathbb{Z}/(p)$, computes the Pedersen commitment $c = g^x h^r$, and sends c to T. T asks U to open the commitment c, and checks that U can indeed commit to the value x. T digitally signs c and send its signature to U. This is an alternative to the **CA-Commit** step in the original EQ-OCBE protocol, in which T sends c to SP. By adopting a public-key infrastructure, T can go off-line after this step. Later in communications, U sends c as well as its signature from T to SP; SP verifies the signature is valid, thus believes that the commitment c is valid. In this way, no further communications are needed between T and U.
Interaction

- U makes a data service request to SP.
- Based on this request, SP sends its policy definition, which requires the value $x_o \in \mathbb{Z}/(p)$ be committed by U.
- Upon receiving this policy, U sends a Pedersen commitment, $c = g^x h^r$, signed by T, to SP.
- After verification of T's signature, SP randomly picks $y \in \mathbb{Z}/(p)^*$, computes $\sigma = (cg^{-x_0})^y$, and sends to U a pair $\langle \eta = h^y, C = \mathcal{E}_{H(\sigma)}[M]\rangle$, where M is the message containing the requested data.

Open Upon receiving $\langle \eta, C\rangle$ from SP, U computes $\sigma' = \eta^r$, and decrypts C using $H(\sigma')$.

The adapted EQ-OCBE protocol above guarantees that U can successfully decrypt the ciphertext if its committed

[2] 'Agg' stands for 'aggregated'.

[3] For a cyclic group G of order q, written multiplicatively, the computational Diffie-Hellman problem is the following problem: Given a randomly-chosen generator g of G, and g^a, g^b for random $a, b \in \{0, \ldots, q-1\}$, it is computationally intractable to compute the value g^{ab}.

[4] We say a Pedersen commitment c is valid if its holder, U, is allowed to commit to the value x.

value is equal to the one specified in SP's policy, and that it is computationally infeasible for U to do so if otherwise. SP will not know if the message M has been successfully decrypted, without further communications with U.

3. AGGREGATION OF EQ-OCBE

The modification of the original EQ-OCBE protocol works for one equality condition. In many cases, we want the user to be able to decrypt a message, containing the protected data, if and only if several equality conditions are all satisfied. We can do this by dividing the encryption key into many shares, then performing the EQ-OCBE protocol multiple times, once for each share. More specifically, this can be done as follows.

- Suppose the user U requests data from the service provider SP.
- SP responds with its policy which specifies that n values $x_1, \ldots, x_n \in \mathbb{Z}/(p)$ need to be committed by U in order that U can be served.
- U then sends to SP its n corresponding commitments $c_1, \ldots, c_n$.
- SP chooses $n-1$ random messages $M_1, \ldots, M_{n-1}$, which have the same bit length as the to-be-sent message M (containing the data) and sets

$$M_n = M \bigoplus_{i=1}^{n-1} M_i,$$

  where $\oplus$ denotes the bitwise exclusive-or operation.
- SP and U performs the interaction and open procedures as above for n times, for n encrypted M_i.
- U computes

$$M = \bigoplus_{i=1}^{n} M_i.$$

However, such an approach is not very efficient in terms of bandwidth and computation. For n such equality conditions, the number of packets sent in communications and the computational cost increase by approximately n times.

We shall present an aggregated version of the EQ-OCBE protocol, Agg-EQ-OCBE, which handles multiple equality conditions at the same time, without significantly increasing computational cost. Agg-EQ-OCBE also requires less bandwidth compared to the above n-round EQ-OCBE.

Protocol 2. (Agg-EQ-OCBE)
In addition to $\mathcal{E}, H(\cdot)$, and G as in EQ-OCBE, another cryptographic component, a cryptographic hash function $H_1(\cdot) : \{0,1\}^* \to \mathbb{Z}/(p)$, is used.
Parameter generation The system parameters $\mathsf{Param} = \langle G, g, h \rangle$ are generated in the same way as in Protocol 1.
Commitment To commit to an element $x \in \mathbb{Z}/(p)$, U randomly chooses $r \in \mathbb{Z}/(p)$, computes the Pedersen commitment of the hash value $H_1(x)$, $c = g^{H_1(x)}h^r$, and sends c to T. T asks U to open the commitment c by revealing x and r. After verifying that x can be committed by U and indeed $c = g^{H_1(x)}h^r$, T digitally signs c and sends the signature to U. U can hold multiple such commitments corresponding to different committed values.
Interaction (with aggregation)

- U makes a data request to SP.
- Based on this request, SP sends its policy, specifying that n values $x_0^{(i)}, i = 1, \ldots, n$, must be committed by U, i.e., U must hold n commitments $c_i = g^{H_1(x_0^{(i)})}h^{r_i}, i = 1, \ldots, n$, all signed by T, in order to be served.
- Upon receiving this policy, U picks its n corresponding commitments c_i, all signed by T, and sends these commitments together with the signatures to SP. Note that these signatures can be sent in an aggregated way, up to the requirements and design of the system, as described in [8, 1]. We shall use aggregate signature in this protocol.
- SP verifies T's signatures, in an aggregated way, for all commitments c_i. SP computes the aggregate commitment

$$c = \prod_{i=1}^{n} c_i,$$

  and the value

$$x_0 = \sum_{i=1}^{n} H_1(x_0^{(i)}) \in \mathbb{Z}/(p).$$

  SP randomly picks $y \in \mathbb{Z}/(p)^*$, computes $\sigma = (cg^{-x_0})^y$, and sends to U a pair $\langle \eta = h^y, C = \mathcal{E}_{H(\sigma)}[M] \rangle$, where M is the message related to the requested service.

Open
Upon receiving $\langle \eta, C \rangle$ from SP, U computes

$$r = \sum_{i=1}^{n} r_i,$$

and

$$\sigma' = \eta^r.$$

U then decrypts C using $H(\sigma')$.

Definition 2. (Soundness of Agg-EQ-OCBE)
An Agg-EQ-OCBE protocol is *sound*, if the user U, whose committed values $x_0^{(i)}, i = 1, \ldots, n$ are those specified by SP's policy, can output the plain-text message M with non-negligible probability.

It can be easily seen that Agg-EQ-OCBE is sound. When $c_i = g^{H_1(x_0^{(i)})}h^{r_i}$, we have that

$$\begin{aligned}
\sigma = (cg^{-x_0})^y &= (\prod_{i=1}^{n} c_i g^{-x_0})^y \\
&= \left(\left(\prod_{i=1}^{n} g^{H_1(x_0^{(i)})} h^{r_i} \right) g^{-\sum\limits_{i=1}^{n} H_1(x_0^{(i)})} \right)^y \\
&= \left(h^{\sum\limits_{i=1}^{n} r_i} \right)^y = (h^r)^y = (h^y)^r = \eta^r.
\end{aligned}$$

4. SECURITY ANALYSIS

Due to the unconditional hiding property of the Pedersen commitment scheme, the service provider SP is not able to learn whether any of the user U's attributes satisfy the required conditions in the policy.

The security analysis of EQ-OCBE [9] implies that when a single commitment is considered, it is hard for a user U to obtain useful information if U's committed value is not equal to that specified by SP, i.e., EQ-OCBE is *oblivious*. It can be easily seen that a similar argument holds true for Agg-EQ-OCBE. For the Agg-EQ-OCBE protocol, we have the additional concern that a user U who does not possess all commitments corresponding to the values specified by the SP may still be able to correctly decrypt the communications. For Example, if the SP's policy requires two commitments $c_1 = g^{21}h^{r_1}, c_2 = g^{35}h^{r_2}$ to be presented, a user U who holds two commitments $c_3 = g^{18}h^{r_3}, c_4 = g^{38}h^{r_4}$ can open the envelope, because the two aggregate commitment $c_1 \cdot c_2$ and $c_3 \cdot c_4$ have $56 = 21 + 35 = 18 + 38$ as their exponents for g, although U does not conform to the policy. The Agg-EQ-OCBE is designed to prevent such an attack from happening.

For the security analysis of Agg-EQ-OCBE, we shall introduce a new and reasonable property for the cryptographic hash function $H_1(\cdot) : \{0,1\}^* \to \mathbb{Z}/(p)$ that we use in Agg-EQ-OCBE. This new definition of property relies on the fact that the range of the hash function is a subset of a group, in which group operations can be considered.

Definition 3. (Group 2nd-preimage resistance)
Let $(\widetilde{G}, +)$ be a finite abelian group of large cardinality[5]. Let $\widetilde{H} : \{0,1\}^* \to \widetilde{G}$ be an unkeyed hash function. We say that $\widetilde{H}(\cdot)$ has the property of *group 2nd-preimage resistance* if for any positive integer m and n sufficiently smaller than $|G|$, and for any given m inputs $x_1, \ldots, x_m$, it is computationally infeasible to find n inputs $y_1, \ldots, y_n$, with

$$\{x_1, \ldots, x_m\} \neq \{y_1, \ldots, y_n\},$$

such that

$$\sum_{i=1}^{m} \widetilde{H}(x_i) = \sum_{i=1}^{n} \widetilde{H}(y_i).$$

Note that the group 2nd-preimage resistance property is stronger than the well-known 2nd-preimage resistance property (cf. e.g. [11]) of cryptographic hash functions, where the latter property is an instance of the former with $m = n = 1$. It is not known yet whether the property of group 2nd-preimage resistance is a consequence of the three basic properties of a general cryptographic hash function: preimage resistance, 2nd-preimage resistance, and collision resistance.

Given this definition, we now can give a security proof of Agg-EQ-OCBE.

Since we assume that $\mathcal{E}$ is a semantically secure symmetric-key encryption algorithm, the ability to decrypt a message is equivalent to the knowledge of the secret encryption key. When the hash function H is modeled as a random oracle, the user U can compute this secret key $H(\sigma)$ only if U can compute the value $\sigma = (cg^{-x_0})^y$. We therefore say the Agg-EQ-OCBE protocol is *secure against the user U* when no polynomial time adversary can win the following game with non-negligible probability.

Game:
Players: challenger $\mathcal{C}$, adversary $\mathcal{A}$
Rules:

- $\mathcal{C}$ generates and sends Param $= \langle G, g, h \rangle$ to $\mathcal{A}$. $\mathcal{C}$ chooses and sends $x_1, \ldots, x_n \in \mathbb{Z}/(p)$ to $\mathcal{A}$. $\mathcal{C}$ chooses $b \in \mathbb{Z}/(p)^*$, and sends h^b to $\mathcal{A}$.
- $\mathcal{A}$ chooses $y_1, \ldots, y_n, r_1, \ldots, r_n \in \mathbb{Z}/(p)$, with $\{x_1, \ldots, x_n\} \neq \{y_1, \ldots, y_n\}$, and sends $y_i, r_i, 1 \leq i \leq n$ to $\mathcal{C}$. $\mathcal{A}$ outputs a value σ'.
- $\mathcal{C}$ computes $c = \prod_{i=1}^{n} g^{H_1(y_i)}h^{r_i}$, $x = \sum_{i=1}^{n} H_1(x_i)$, and
$$\sigma = (cg^{-x})^b.$$
- $\mathcal{A}$ wins the game if $\sigma' = \sigma$.

Theorem 1. *Assume that the computational Diffie-Hellman problem is intractable in G. Model H as a random oracle, and assume that H_1 has the property of group 2nd-preimage resistance. Then Agg-EQ-OCBE is secure against the user U.*

The proof of Theorem 1 is reported in Appendix A

5. EXPERIMENTAL RESULTS

We have performed an experimental evaluation to compare the performance of the multiple-round EQ-OCBE and Agg-EQ-OCBE protocols. For multiple-round EQ-OCBE, we generate the Pedersen commitments by committing to the actual values $x_0^{(i)}$ and do not introduce the cryptographic hash function $H_1(\cdot)$. For Agg-EQ-OCBE, we use the hash function $H_1(\cdot)$ and commit to the hash values $H_1(x_0^{(i)})$. The experiment compares the creation time of σ and η at the service provider's side, which consists of the most computationally costly part for both protocols, and the derivation time of σ' from η at the user's side. We also compare the creation time of aggregate commitment and the creation time for σ and η ("envelope"), both at the service provider's side. We do not include communication time and symmetric encryption time in the comparisons, which vary with different network settings and plaintext lengths, in order to focus on the core components of the protocols. We also do not include the signature verification time in the comparison, for the same reason. We expect Agg-EQ-OCBE to outperform multiple-round EQ-OCBE, when the number of involved commitments increases.

In our experiment, we choose the group G to be the rational points of the Jacobian variety (aka. Jacobian group) of a genus 2 curve $C : y^2 = x^5 + 2682810822839355644900736x^3 + 226591355295993102902116x^2 + 2547674715952929717899918x + 4797309959708489673059350$ over the prime field $\mathbb{F}_q$, with $q = 5 \cdot 10^{24} + 8503491$ (83 bits). The Jacobian group of this curve has a prime order (164 bits)[6]:

$$p = 24999999999994130438600999402209463966197516075699.$$

The parameter generation program chooses non-unit points g and h in the Jacobian group as the base points for constructing the Pedersen commitments.

In the experiment, we run both multiple-round EQ-OCBE and Agg-EQ-OCBE at the service provider's side for $n (1 \leq n \leq 50)$ Pedersen commitments of randomly generated values $x_0^{(i)}, 1 \leq i \leq n$. We use $x_0^{(i)}$ as the exponents of g for

[5] Let $|G|$ denote the cardinality of a set G, for all G.

[6] The data is taken from [7].

multiple-round EQ-OCBE, and the hash values of $x_0^{(i)}$ as the exponents for Agg-EQ-OCBE, where the hash function $H_1(\cdot) : \{0,1\}^* \rightarrow \mathbb{Z}/(p)$ is built on SHA-1. We also simulate the aggregation of n commitments at the user's side for Agg-EQ-OCBE. For each $n, 1 \leq n \leq 50$, we run 50 rounds of both protocols on n Pedersen commitments. In each round, the n Pedersen commitments under test are different (randomly chosen) and we take the average running times of the 50 rounds. The experimental results are presented in Figures 1: From top to bottom:

- Computation time comparison at service provider's side of multiple-round EQ-OCBE and Agg-EQ-OCBE;
- Computation time comparison at user's side of multiple-round EQ-OCBE and Agg-EQ-OCBE;
- Computation time comparison at service provider's side of commitment aggregation and envelope creation, for Agg-EQ-OCBE.

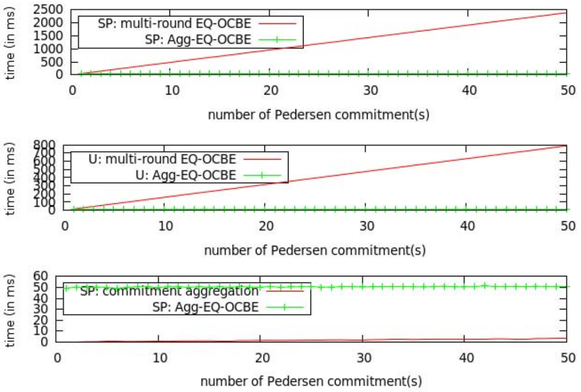

Figure 1: Running time comparison

The experiment was performed on a machine running GNU/Linux kernel version 2.6.9-67.0.1.ELsmp with 4 AMD Opteron (tm) Processor 850 2390MHz and 7.36 Gbytes memory. Only one processor was used for computation. The code is written in C++, and built with gcc version 3.6.4, optimization flag -O2. The code is built over the G2HEC C++ library [13], which implements the arithmetic operations in the Jacobian groups of genus 2 curves.

The experimental results show that while in multi-round EQ-OCBE the running time for composing the EQ-OCBE envelopes linearly increases with the number of involved Pedersen commitments, in Agg-EQ-OCBE it is nearly constant. The experimental results also imply that the overhead of the hash computation introduced in Agg-EQ-OCBE takes negligible time. We have obtained similar results for the envelope opening operations executed at the user's side. We can see that the operation of aggregation of commitments at the service provider's side takes very little time compared to the envelope creation operations. Therefore, Agg-EQ-OCBE is more efficient than the solution based on running EQ-OCBE for multiple rounds.

6. CONCLUSIONS

In this paper, we have proposed, Agg-EQ-OCBE, an extension that improves the efficiency of the EQ-OCBE protocol by allowing a user to decrypt data sent by a service provider if and only if the user satisfies several equality conditions. We have proved the security of our Agg-EQ-OCBE protocol. The experimental results show that the Agg-EQ-OCBE is more efficient than running the EQ-OCBE protocol iteratively for each equality predicate. Future work includes developing efficient OCBE protocols for inequality predicates.

Acknowledgements

The work reported in this paper has been partially supported by the MURI award FA9550-08-1-0265 from the Air Force Office of Scientific Research.

7. REFERENCES

[1] D. Boneh and C. Gentry. Aggregate and verifiably encrypted signatures from bilinear maps. In *Proceedings of Eurocrypt 2003, volume 2656 of LNCS*, pages 416–432. Springer-Verlag, 2003.

[2] S. Brands. Rethinking public key infrastructures and digital certificates: Building in privacy. *MIT Press*, 2000.

[3] J. Camenisch and E. Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proc. Ninth ACM Conf. Computer and Comm. Security*, pages 21–30, 2002.

[4] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Advances in Cryptology, Proc. EUROCRYPT 01*, pages 93–118, 2001.

[5] M. Camenisch, J.and Stadler. Efficient group signature schemes for large groups. *Advances in Cryptology, CRYPTO '97*, pages 410–424, 1997.

[6] D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Comm. ACM*, 28(10):1030–1044, 1985.

[7] P. Gaudry and É. Schost. Construction of secure random curves of genus 2 over prime fields. In *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 239–256. Springer-Verlag, 2004.

[8] L. Harn. Batch verifying multiple RSA digital signatures. *Electronics Letters*, 34(12):1219–1220, Jun 1998.

[9] J. Li and N. Li. OACerts: Oblivious attribute certificates. *IEEE Transactions on Dependable and Secure Computing*, 3(4):340–352, 2006.

[10] A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. In *Proc. Sixth Workshop Selected Areas in Cryptography*, pages 184–199, 1999.

[11] A. J. Menezes, P. C. V. Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, FL, USA, 1996.

[12] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO '91: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, pages 129–140, London, UK, 1992. Springer-Verlag.

[13] N. Shang. G2HEC: A Genus 2 Crypto C++ Library. http://www.math.purdue.edu/˜nshang/libg2hec.html.

APPENDIX

A. PROOF OF THEOREM 1

PROOF. We shall show that if there is an adversary $\mathcal{A}$ who wins the game with probability ϵ, we can construct another adversary $\mathcal{B}$ who can either break the group 2nd-preimage resistance property of H_1, or solve the computational Diffie-Hellman problem in G, with the same probability ϵ. Indeed, $\mathcal{B}$ executes the following procedures:

- When given a group G, $h, h^a, h^b \in G$, and $x_1, \ldots, x_n \in \mathbb{Z}/(p)$, $\mathcal{B}$ gives $\mathsf{Param} = \langle G, h^a, h \rangle$ to $\mathcal{A}$. $\mathcal{B}$ also sends $x_1, \ldots, x_n$, and h^b to $\mathcal{A}$. Let $g = h^a$.
- $\mathcal{B}$ receives $y_1, \ldots, y_n, r_1, \ldots, r_n$, and σ' from $\mathcal{A}$, where $\{x_1, \ldots, x_n\} \neq \{y_1, \ldots, y_n\}$.
- $\mathcal{B}$ computes $x = \sum_{i=1}^{n} H_1(x_i)$, $y = \sum_{i=1}^{n} H_1(y_i)$, and checks whether $x = y$. If $x \neq y$, $\mathcal{B}$ computes $r = \sum_{i=1}^{n} r_i$, and outputs
$$\delta = (\sigma'(h^b)^{-r})^{(y-x)^{-1}},$$
where $(y-x)^{-1}$ is the multiplicative inverse of $y - x$ in $\mathbb{Z}/(p)$.

When $\mathcal{A}$ wins the game, we have

$$\begin{aligned}\sigma' &= \left(\left(\prod_{i=1}^{n} g^{H_1(y_i)} h^{r_i}\right) g^{-x}\right)^b \\ &= (g^{y-x} h^r)^b.\end{aligned}$$

If $x = y$, then the group 2nd-preimage resistance property of H_1 fails to hold. Otherwise,

$$\delta = (\sigma'(h^b)^{-r})^{(y-x)^{-1}} = g^b = (h^a)^b = h^{ab},$$

i.e., the computational Diffie-Hellman problem is solved. □