CERIAS Tech Report 2005-110

OBLIVIOUS SIGNATURE-BASED ENVELOPE

by Ninghui Li, Wenliang Du, Dan Boneh

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

Oblivious Signature-Based Envelope*

Ninghui Li†
Department of Computer Sciences and CERIAS
Purdue University
656 Oval Dr, West Lafayette, IN 47907-2086
ninghui@cs.purdue.edu

Wenliang Du
Department of Electrical Engineering and Computer Science
Syracuse University
121 Link Hall, Syracuse, NY 13244
wedu@ecs.syr.edu

Dan Boneh
Department of Computer Science
Stanford University
Gates 4B, Stanford, CA 94305-9045
dabo@cs.stanford.edu

Abstract

We propose a new cryptographic primitive called oblivious signature-based envelope (OSBE). Informally, an OSBE scheme enables a sender to send an envelope (encrypted message) to a receiver, and has the following two properties: (1) The receiver can open the envelope if and only if it has a third party's (e.g., a certification authority's) signature on an agreed-upon message. (2) The sender does not learn whether the receiver has the signature or not. We show that OSBE can be used to break policy cycles in automated trust negotiation (ATN) and to achieve oblivious access control.

We develop a provably secure and efficient OSBE protocol for certificates signed using RSA signatures, as well as provably secure and efficient one-round OSBE protocols for Rabin and BLS signatures from recent constructions for identity-based encryption. We also present constructions for

*Invited submission to the journal *Distributed Computing*, special issue of selected papers of PODC 2003. Preliminary version appeared in Proceedings of PODC'2003 under the same title.

†Most of this work was performed while the first author was a Research Associate at the Department of Computer Science, Stanford University in Stanford, CA 94305.

Generalized OSBE, where signatures on multiple messages (and possibly by different authorities) are required to open the envelope.

1 Introduction

Consider the following scenario: 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 have top-secret clearance. Now imagine what happens when Alice and Bob wish to establish a secure session using automated trust negotiation techniques. Neither one is willing to present their certificate first. Consequently, they are stuck and cannot establish the session. We describe efficient cryptographic solutions to this problem. Our solutions work with standard certificate formats.

Exchanging digitally signed certificates is an increasingly popular approach for authentication and authorization in distributed systems. These certificates associate public keys with key holders' identity and/or attributes such as employer, membership of associations, credit card information, security clearance, and so on. Often, the attribute information contained in certificates is sensitive. The goal of a growing body of work on *automated trust negotiation (ATN)* [25, 21, 24, 26, 28, 29] is to protect this information. In ATN, each party establishes access control (AC) policies to regulate not only the granting of resources, but also the disclosure of certificates to opponents. (Engaging in a discussion about secret information can be viewed as an abstract resource protected by the AC policy that requires secret clearance certificates.) A negotiation begins when a requester requests to access a resource protected by an AC policy. The negotiation process consists of a sequence of exchanges of certificates and possibly AC policies. In the beginning, certificates that are not sensitive are disclosed. As certificates flow, higher levels of mutual trust are established, and AC policies for more sensitive certificates are satisfied, enabling these certificates also to flow. In successful negotiations, certificates eventually flow to satisfy the AC policy of the desired resource. A security requirement of ATN is that no certificate should flow to a party who does not satisfy the AC policy established for the certificate.

In the scenario we described in the beginning of this paper, current ATN protocols would conclude negotiation failure, because there is a cyclic interdependency between two negotiators' AC policies. Existing ATN protocols require one negotiator to reveal its certificate first; however, if the receiver does not have top-secret clearance, the AC policy is violated. Reporting negotiation failure in this scenario is not very satisfactory, as both parties have top-secret clearance and it would be

more productive for them to proceed. How to break this policy cycle? Observe that, in many cases, the secret information in a certificate is the signature created by the certificate authority (CA). For example, Alice's certificate may contain her public key and some string representing "top-secret clearance"; these are often public information, but the fact that a trusted authority signed the certificate is sensitive. Using this observation, the cycle can be broken as follows: First, Bob sends the content, including the CA's public key but not the signature, of his certificate to Alice.[1] Alice verifies that the content satisfies her requirement, then conducts a joint computation with Bob such that in the end Bob sees Alice's certificate if and only if Bob has the CA's signature on the content he sent earlier. Bob concludes negotiation success and proceeds with Alice if he has the signature and successfully verifies that Alice has the right certificate. Bob aborts the negotiation process when he does not have the signature or when Alice does not have the right certificate. Bob learns whether Alice has the certificate only when he has the required certificate, and vice versa.[2] This approach for breaking policy cycles requires solving the following 2-party Secure Function Evaluation (SFE) problem.

Problem 1 Let PK be a public key (the CA's public key). Let M and P be two messages. (M is the content of Bob's certificate without the CA's signature; P is Alice's complete certificate.) Let Verify be the verification algorithm of a signature scheme such that $\mathsf{Verify}_{PK}(M, \sigma) = \mathsf{true}$ when σ is PK's signature on M. Alice and Bob want to compute a family F of functions, parameterized by Verify, M and PK. Both parties have M and PK. Alice has private input P (Alice's certificate). Bob has private input σ (the CA's signature on M). The function F is defined as follows.

$$
\begin{array}{rcl}
F[\mathsf{Verify}, M, PK]_{Alice}(P, \sigma) & = & \perp \\
F[\mathsf{Verify}, M, PK]_{Bob}(P, \sigma) & = & \\
 & & \left\{ \begin{array}{ll} P & \text{if } \mathsf{Verify}_{PK}(M, \sigma) = \mathsf{true}; \\ \perp & \text{otherwise.} \end{array} \right.
\end{array}
$$

where $F[\mathsf{Verify}, M, PK]_{Alice}$ represents Alice's output, $F[\mathsf{Verify}, M, PK]_{Bob}$ represents Bob's output, and $\perp$ denotes a constant value. In other words, our goal is that Alice learns nothing (as Alice sees a constant value) and Bob learns P only when his private input σ is PK's signature on M.

The SFE problem can be solved using general solutions to 2-party SFE [27, 14]; however, the general solutions are not efficient, because signature verification is

[1] To prevent Alice from guessing whether Bob has top-secret clearance or not, Bob should follow the protocol to send the same thing even if he does not have the top-secret clearance certificate. This is possible because the content of a certificate is not secret.

[2] See Section 3 for another way of breaking the policy cycle.

done within the SFE. We propose the Oblivious Signature-Based Envelope (OSBE) scheme that solves the above 2-party SFE problem efficiently. Formal definition of OSBE will be given in Section 4. Informally, an OSBE scheme enables a sender to send an envelope (encrypted message) to a receiver, and has the following properties: the receiver can open the envelope if and only if it has a third party's (e.g., a CA's) signature on an agreed-upon message M. An OSBE scheme is *secure against the receiver* if a receiver who does not have the third party's signature on M cannot open the envelope. An OSBE scheme is *oblivious* if at the end of the protocol the sender cannot tell whether the receiver has the signature on M or not.

In this paper, our focus is to find efficient OSBE constructions for existing signature schemes, rather than to develop new signature schemes that make OSBE easy. In addition, we look for protocols that do not involve any interaction with (trusted or semi-trusted) third parties, except for the generation of signatures on certificates by the CA's. We present OSBE protocols for three existing signature schemes: RSA [19], Rabin [18], and BLS [8]. The RSA-OSBE protocol is two-round: one message from the receiver followed by one message from the sender. The receiver and the sender each computes two exponentiations. We prove in the Random Oracle Model [5] that our RSA-OSBE protocol is as secure as RSA signatures. We also show that any Identity Based public key Encryption (IBE) [22, 7, 10] scheme directly gives rise to an OSBE scheme for the signature scheme corresponding to the IBE scheme. We use IBE to build one-round OSBE protocols for Rabin and BLS. These two protocols involve only one message from the sender to the receiver. We also present constructions for Generalized OSBE, where multiple certificates (possibly issued by different CA's) are required to open the envelope.

The rest of this paper is organized as follows. Related work is discussed in Section 2. We discuss other applications of OSBE in Section 3, and give formal definition of OSBE and its security requirements in Section 4. In Section 5, we describe an OSBE protocol for RSA signatures and prove its security. In Section 6 we build a one round OSBE for Rabin and BLS signatures. In Section 7, we present constructions for Generalized OSBE. We conclude in Section 8.

2 Related Work

Holt et al. [16] introduced the notion of hidden credentials. The basic idea underlying hidden credentials is that the Boneh-Franklin IBE scheme [7] gives rise to an OSBE scheme for BLS signatures [8], the signature scheme corresponding to the IBE scheme. Hidden credentials are digitally signed using BLS signatures. Holt et al. [16] independently 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. Because IBE-derived OSBE is one-round, assuming the content of a certificate can be guessed, one can start communication by sending an encrypted message such that the other party can obtain the message only when using the correct certificate to decrypt. Holt et al. [16] also investigated how to use this property of hidden credentials to hide the request in an ATN as well as the policy.

Holt et al. [16] did not formalize the concept and the security requirements of OSBE. Also, they did not provide OSBE protocols for RSA signatures. Furthermore, hidden credentials can be used only when the content of certificates can be guessed. When a certificate contains a validity period and/or a serial number, as existing public-key certificate standards mandate, guessing the content becomes very difficult (if not impossible). Therefore, the hidden-credentials scheme does not seem to work with existing security standards.

Holt et al. [16] considered the situation where the policy protecting the encrypted message is complex in the sense that it requires the possession of multiple credentials to satisfy. Their construction is similar to yet different from the Generalized OSBE construction in Section 7. For example, consider the case that the policy requires the recipient to possess one of two credentials. In Generalized OSBE, there is only one ciphertext of the message, encrypted under a key that is in turn encrypted under two new random keys, which are then sent by OSBE, each can be retrieved with the possession of one credential. In [16], there are two ciphertexts for the message, each can be decrypted using one credential. When the message is long, the approach of Generalized OSBE results in smaller message sizes.

Balfanz et al. [3] proposed a construct called Secret Handshakes using pairings that are also the foundation of the Boneh-Franklin IBE scheme [7] and its corresponding BLS signature scheme [8]. Their scheme uses the pairing-based key agreement protocol by Sakai et al. [20]. In their scheme, each party receives a certificate from a central authority; the certificate consists of a pseudonym and a corresponding secret, which is computed from the pseudonym and an attribute string using the central authority's master secret. 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. The keys they computed agree only when they have the correct certificate.

Secret Handshakes require Alice and Bob to mutually authenticate using certificates from the same authority. In contrast, OSBE allows certificates issued by different authorities to be used.

Crescenzo et al. [11] 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. [11] developed an efficient protocol for a special case of Conditional Oblivious Transfer where the predicate is greater-than-or-equal-to ($\geq$). OSBE can be viewed as another special case of the Conditional Oblivious Transfer problem; however, the solutions in [11] do not apply. In OSBE only Bob has a private input and the predicate on the private input is that it has to be the digital signature of a shared message. Such predicates are quite different from the predicates considered in [11], and our constructions are quite different from those in [11].

Gertner et al. [12] introduced the notion of Conditional Disclosure in the context of private information retrieval. Aiello et al. [1] adapted the Conditional Disclosure notion to the single-server setting. Because the single-server setting is closer to our setting for OSBE, we will discuss the work by Aiello et al. [1]. In their setting, Bob holds a vector $y = (y_1, \ldots, y_m)$ over a large field $F = \mathbb{Z}_Q$ and the private key corresponding to a public key k, Alice holds the public key k, the encryptions of $E_k[y_1], \ldots, E_k[y_m]$, a predicate C, and a secret $s \in F$. The goal is for Alice to send to Bob a message such that Bob learns s if and only if $C(y) = 1$. The solutions described there was for predicates that test whether y satisfies some linear equation over F. This is quite different from OSBE, in which Bob holds the signature and Alice holds the message.

Another problem related to OSBE that has been studied in the literature is Fair Exchange of Signatures (FES) [2, 4], which enables two parties to exchange signatures such that either both parties obtain the other parties' signature or no party obtains the other party's signature. FES protocols are useful in contract signing and other e-commerce transactions. A common approach to FES is verifiable encryption of signatures, i.e., a signature encrypted in a way such that one can verify that the right signature is being encrypted, one can also go to a trusted third party (TTP) to obtain the signature when necessary, but one cannot retrieve the signature without the TTP. The TTP is involved only if one party tries to cheat. There are several differences between OSBE and FES. First, the signatures involved in OSBE are not generated by the two parties involved in the protocols, but rather generated by certification authorities before the OSBE protocol is used. Second, in FES protocols, at some stage, one party learns that the other party has a signature without obtaining that signature. This does not satisfy the security requirements of OSBE. Because of the above two reasons, FES protocols cannot be used directly to achieve OSBE. Third, OSBE does not require a fair exchange of signatures. The receiver is allowed to obtain the sender's signature without sending its own signature, as long as the receiver has the required signature. In this sense, OSBE is weaker than fair exchange of signatures. This weaker requirement enables efficient

OSBE protocols that do not involve third parties.

Another piece of related work is Brands' private certificates [9]. There, the main goal is that certificates can be used anonymously. In OSBE certificates are used in an oblivious fashion. That is, the service provider does not learn whether the other party has a certificate or not but the certificate holder retrieves information only when it has the correct certificate.

3 Other Applications of OSBE

We revisit the approach to use OSBE to break policy cycles as described in Section 1. In that approach Bob sends to Alice M, the content of a certificate he may have; then Alice and Bob run an OSBE with Alice's certificate in the envelope such that Bob receives Alice's certificate only if Bob possesses a correct signature on M. This is asymmetric in the sense that at the end of the protocol Bob receives Alice's certificate, provided that he has a signature on M, but Alice learns nothing. Alice has to wait until Bob sends her his certificate.

We can reduce the asymmetry to the extent that both Alice and Bob send signatures on their certificates only after being *convinced* that the other party has the required certificate. To do so we use two OSBE protocol runs to break a policy cycle. Alice sends to Bob her certificate content M_A and Bob sends to Alice his certificate content M_B (both without the signatures). Alice and Bob then run two OSBE's. In the first one, Alice sends to Bob a random challenge r_A in the envelope. When Bob has the CA's signature on M_B, he opens the envelope, recovers r_A, and sends to Alice (r_A, r_B) in the envelope, where r_B is a random challenge generated by Bob. If Bob doesn't have the signature, he sends $(0, r_B)$ in the envelope. If Alice opens the envelope and verifies that Bob has sent back r_A, then she knows that Bob has a valid signature on M_B and she sends back r_B, proving that she has a valid signature on M_A. If Alice cannot open the envelope or if she finds out that Bob didn't send back the correct challenge, she stops. When both Alice and Bob have the certificates, at the end of the protocol both are convinced that the other party has the right certificate, yet no one can prove to any third party. If they still would like to exchange signatures they can do so by transmitting signatures; this will not violate the security requirement of ATN.

Some level of asymmetry seems inherent in ATN protocols that exchange certificates. Either Alice or Bob has to send the certificate (or the signature) first. That is acceptable as our objective is safe disclosure of certificates and attribute information, rather than fair exchange.

Our original motivation for OSBE comes from automated trust negotiation; however, OSBE can be used for other purposes. An OSBE scheme enables the

sender to send a message with the assurance that it can be seen only by a receiver who has appropriate certificates while at the same time protecting the receiver's privacy such that the sender does not know whether the receiver has the required certificates or not. In other words, OSBE performs access control on a message in an oblivious (or privacy preserving) fashion. We envision that OSBE could be used in other contexts (possibly in conjunction with other protocols) to provide such oblivious access control.

Consider the following scenario, Alice wants to report a message to Bob, who claims to be a CIA agent. Alice wants to make sure that the message is only sent to a CIA agent who has the right certificate, but Bob doesn't want to prove to Alice that he is a CIA agent. Alice can use OSBE to send the message. Alice doesn't know whether the message reaches a CIA agent, but she is confident that the message doesn't leak to a party who is not a CIA agent.

A similar application of OSBE is Oblivious Subscription. Consider an online publishing service that gives access of various documents to members of several organizations. Users need membership certificates to gain access to specific documents. OSBE enables users to gain access without disclosing which organizations they are members of. To do so, the publishing service encrypts all documents with distinct keys. When a user requests to access a document, it sends contents of some membership certificates it may or may not possess, and runs multiple rounds of the OSBE protocol with the publishing service. The publishing service delivers decryption keys of the documents in corresponding envelopes. Only a user that has the required certificate can open the envelope and obtain keys to decrypt documents. The publishing service does not know what memberships the user has.

4 Oblivious Signature-Based Envelope (OSBE): Definition

In this section, we give formal definition of OSBE. We will use the following terminology. A function $f(t)$ 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.

In the following definition of OSBE, we use two receivers R_1 and R_2. Receiver R_1 has the CA's signature on some message M. Receiver R_2 does not have the signature. When using OSBE, a receiver follows the behavior of R_1 when it has the signature and follows that of R_2 when it does not.

Definition 1 *Oblivious Signature-Based Envelope (OSBE)*
An Oblivious Signature-Based Envelope (OSBE) scheme is parameterized by a signature scheme Sig. Given a public verification key PK and a message M, we

use $\mathsf{Sig}_{PK}(M)$ to denote the digital signature of M created using the private key corresponding to PK.

An OSBE scheme consists of four parties (that can be modelled as communicating Turing machines): CA, sender S, receiver R_1, and receiver R_2. We begin by describing the three phases of communications between the parties. We then define the security properties.

Setup. In the setup phase the CA takes a security parameter t and two messages M and P as input. The CA generates system parameters. As part of this, the key generation algorithm of Sig is executed to create a signing key whose public key is denoted by PK.

The CA keeps the secret signing key to itself. It gives the system parameters and the public key PK to S, R_1, and R_2. In addition, the CA gives the message M to S, R_1, and R_2, the message P to S, and the signature $\sigma = \mathsf{Sig}_{PK}(M)$ to R_1.

Interaction. The interaction phase has two kinds of interactions, one between S and R_1 and the other between S and R_2.

Open After an interaction between S and R_1, R_1 outputs the message P (onto a private tape) in the open phase. After an interaction between S and R_2, R_2 does nothing in the open phase.

An OSBE must satisfy three properties defined below. It must be sound, oblivious, and semantically secure against the receiver.

Sound. An OSBE scheme is *sound* if in the open phase, R_1 can output the message P with overwhelming probability, that is, the probability that R_1 cannot output P is negligible.

Oblivious. An OSBE scheme is *oblivious* if a malicious sender cannot tell the difference between an interaction with R_1 and an interaction with R_2. More precisely, an OSBE scheme is *oblivious* if no adversarial sender $\mathcal{A}$ has a nonnegligible advantage against a Challenger in the following game: The Challenger emulates the CA and generates the public key PK and other necessary public parameters. It then sends PK and these parameters to the adversary. The adversary responds with a message M. The Challenger picks random $b \in \{1, 2\}$ and interacts with the adversary by emulating R_b. Finally, the adversary outputs $b' \in \{1, 2\}$. The adversary wins the game if $b = b'$. In other words, an OSBE scheme is *oblivious* if for every probabilistic interactive Turing Machine $\mathcal{A}$,

$|\mathsf{Pr}[\mathcal{A}$ wins the above game$] - \frac{1}{2}| \leq f(t)$, where f is a negligible function in t. (The adversary cannot do substantially better than random guessing.)

Semantically secure against the receiver. An OSBE scheme is *semantically secure against the receiver* if a malicious receiver that does not have σ learns nothing about P. More precisely, no polynomially bounded adversarial receiver $\mathcal{A}$ has a nonnegligible advantage against the Challenger in the following game: The Challenger emulates the CA and generates the public key PK and other necessary public parameters. It then sends PK and these parameters to the adversary. The adversary responds with a message M and two equal-length messages P_0 and P_1. Then the Challenger picks a random $b \in \{0, 1\}$ and interacts with the adversary by emulating the sender S using message $P = P_b$. Finally, the adversary outputs $b' \in \{0, 1\}$. The adversary wins the game if $b = b'$. In other words, even if we give the adversary the power to pick two messages P_0 and P_1 of its choice, it still cannot distinguish an envelope containing P_0 from one containing P_1. This formalizes the intuition that the envelope leaks no information about its content, using the standard notion of semantic security [15].

We now argue that OSBE is an adequate solution to the 2-party SFE problem in Problem 1, by showing intuitively that the above security properties defined for OSBE suffice to prove that the scheme protects the privacy of the participants in the malicious model [13]. Observe that our definitions allow arbitrary adversaries, rather than just those following the protocol. The oblivious property guarantees that the sender's view of any protocol run can be simulated using just the sender's input, because one can simulate a protocol run between S and R_2, who has no private input. Soundness and semantic security against the receiver guarantee that the receiver's view can be simulated using just the receiver's input and output. If the receiver has the signature, then the message P is in the output, one can therefore simulates the sender S. If the receiver does not have the signature, one can simulate the sender S with a arbitrary message P' and no polynomially bounded receiver can tell the difference.

We assume that OSBE is executed on top of a secure communication channel that the sender and the receiver has already established. This assumption is common in secure multiparty computation literature. In the context of automated trust negotiation, this assumption is also valid, as secure communication is already required to protect against eavesdroppers. Technically, an SSL connection can be established between the sender Alice and receiver Bob using self-signed certificates. When Alice and Bob wants to use OSBE to break a policy cycle, Bob first sends M (the content of Bob's certificate) to Alice. At this time, Alice verifies that the public key in M is the same as the one Bob used to establish the communi-

cation channel and then runs the OSBE protocol to send P (Alice's certificate) to Bob. At the end of the OSBE, Bob verifies that the public key in P is the same as the one Alice used to establish the communication channel. A man-in-the-middle attack during the OSBE will not be a problem.

In our proofs, we use the random oracle model, which is an idealized security model introduced by Bellare and Rogaway [5] to analyze the security of certain natural cryptographic constructions. Roughly speaking, a random oracle is a function $H : X \to Y$ chosen uniformly at random from the set of all functions $\{h : X \to Y\}$ (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. In the random oracle model, each participant in a protocol is give access to the same random oracle. 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. Security proofs in this model prove security against attackers that are confined to the random oracle world.

5 An OSBE Scheme for RSA Signatures

In this section, we present an OSBE scheme for RSA signatures (i.e., when user certificates are signed using RSA). The RSA signature scheme [19] is as follows. The key space $\mathcal{K}$ is defined to be the following set:

$$\{(n, e, d) \mid n = pq,\ p, q \text{ equal size primes}, ed \equiv 1 \pmod{\phi(n)}\}$$

The values n and e are public, and the value d is secret.
For $K = (n, e, d)$, message M, and a message digest function $H : \{0,1\}^* \to \mathbb{Z}_n$, define

$$\mathsf{Sig}_K(M) = H(M)^d \bmod n$$
$$\text{and} \quad \mathsf{Verify}_K(M, \sigma) = \mathsf{true} \iff H(M) \equiv \sigma^e \pmod{n}$$

Our RSA-OSBE scheme runs a Diffie-Hellman style key agreement protocol. If it is run between S and R_1, then R_1 can derive the shared secret. If it is run between S and R_2, then R_2 cannot derive the shared secret. Let $h = H(M)$, then the signature on the message M is $\sigma = (h^d \bmod n)$. R_1 sends to S a blinded version of the signature $\eta = (\sigma h^x \bmod n)$ for some random x. S then computes $\eta^e h^{-1} \bmod n$, which should be $h^{ex} \bmod n$. S now holds $(h^e)^x$ such that only R_1 knows the value x. This achieves half of the Diffie-Hellman key agreement protocol, with h^e as the base. S then does the other half and creates the envelope using a symmetric key derived from the shared secret.

Definition 2 *RSA-OSBE* Let H be the message digest function used in the signature. Let $\mathcal{E}$ be a semantically secure symmetric encryption scheme [15]. We use $\mathcal{E}_k[P]$ to denote the ciphertext of using key k to encrypt plaintext P. Let H' be a function (e.g., a cryptographic hash function) that extracts a key for the symmetric encryption scheme from a shared secret.

Setup The CA takes a security parameter t and two messages M and P as input. It runs the RSA key generation algorithm to create an RSA key (n, e, d); in addition, it generates two security parameters t_1 and t_2, which are linear in t. In practice, $t_1 = t_2 = 128$ suffices.

The CA gives the following to S, R_1, and R_2: the RSA public key (n, e), the security parameters t_1 and t_2, and the message M. In addition, the CA gives to R_1 the signature $\sigma = (h^d \bmod n)$, where $h = H(M)$, and gives to S the message P.

Interaction We use $x \leftarrow [1..2^{t_1}n]$ to denote that x is randomly chosen from $[1..2^{t_1}n]$. In the following protocol, we describe actions for S, R_1, and R_2. Each protocol run is executed either between S and R_1 or between S and R_2.

- R_1 sends: $\eta = (\sigma h^x \bmod n)$, in which $x \leftarrow [1..2^{t_1}n]$.
  R_2 sends: $\eta = (h^{x'} \bmod n)$, in which $x' \leftarrow [1..2^{t_1}n]$.
- S receives η, checks that $\eta \notin \{0, 1, n-1\}$, picks $y \leftarrow [1..2^{t_2}n]$, computes $r = (\eta^{ey}\ h^{-y} \bmod n)$ and then sends the pair: $\langle \zeta = (h^{ye} \bmod n), C = \mathcal{E}_{H'(r)}[P] \rangle$.
- R_1 and R_2 receive $\langle \zeta, C \rangle$ from S.

Open R_1 computes $r' = (\zeta^x \bmod n)$, and decrypts C using $H'(r')$.

To see that this scheme is sound, observe that $\zeta = (h^{ye} \bmod n)$ and when η is sent by R_1, $\eta = (h^{d+x} \bmod n)$; therefore:

$$r = \eta^{ye} h^{-y} = h^{(d+x)ey}\ h^{-y} = h^{dey}\ h^{xey}\ h^{-y}$$
$$= h^{xye} = \zeta^x = r' \ (\mathrm{mod}\ n)$$

Thus S and R_1 share the same symmetric key.

The key idea of the RSA-OSBE scheme is that it converts R_1's knowledge of the e'th root of h to the knowledge of a discrete log with base h^e. The sender S then uses this fact to do a Diffie-Hellman style key agreement with R_1.

Before proving the oblivious property of RSA-OSBE, we introduce the following terminology. Two distribution families $\delta^0(t)$ and $\delta^1(t)$ are *statistically indistinguishable* if

$$\sum_y |\mathsf{Pr}_{x\in\delta^0(t)}[x=y] - \mathsf{Pr}_{x\in\delta^1(t)}[x=y]| \text{ is negligible in } t.$$

If two distribution families are statistically indistinguishable, then there exists no algorithm that can distinguish the two distribution families with nonnegligible advantage by sampling from them.

Theorem 1 *RSA-OSBE is oblivious.*

Proof. It suffices to show that what R_1 and R_2 send in the first step are drawn from two distribution families that are statistically indistinguishable, i.e., for all h, n, and d, the two distribution families $\delta^0(t_1) = \{h^{d+x} \bmod n \mid x \leftarrow [1..2^{t_1}n]\}$ and $\delta^1(t_1) = \{h^{x'} \bmod n \mid x' \leftarrow [1..2^{t_1}n]\}$ are statistically indistinguishable.

Let o be the order of h, i.e., the smallest number j such that $h^j \equiv 1 (\bmod\ n)$. For any fixed t_1, both distributions have o points. The probability of any point in either distribution is either $\lfloor 2^{t_1}n/o \rfloor/(2^{t_1}n)$ or $\lceil 2^{t_1}n/o \rceil/(2^{t_1}n)$. Therefore, the probability difference on any point is at most $1/(2^{t_1}n)$; the total difference is thus at most $o/(2^{t_1}n)$. As $o \leq \phi(n) < n$, the statistical difference between the two distributions is less than $1/2^{t_1}$, which is negligible in t_1. As t_1 is linear in t, the statistical difference is also negligible in t. ∎

Theorem 2 *Assuming that there exists no polynomial algorithm that successfully creates an existential forgery of RSA signatures under a key-only attack with nonnegligible probability, and H' is modelled as a random oracle, RSA-OSBE is secure against the receiver.*

Proof. RSA-OSBE uses a semantically secure symmetric encryption algorithm. When H' is modelled as a random oracle, RSA-OSBE is secure against the receiver when no receiver who does not have the signature can compute with nonnegligible probability the secret that the sender uses to derive the encryption key. More precisely, RSA-OSBE is secure against the receiver if no polynomially bounded adversary wins the following game against the Challenger with nonnegligible probability: The Challenger randomly picks a public key (n, e), and gives it to the adversary. The adversary responds with a message M and a η such that $\eta \notin \{0, 1, n-1\}$. The Challenger then picks a random y from $[1..2^{t_2}n]$ and sends to the adversary $H(M)^{ye} \bmod n$. The adversary then outputs r, and the adversary wins the game if $r = \eta^{ey}h^{-y} \bmod n$.

Given an attacker $\mathcal{A}$ that wins the above game with probability ϵ. We construct another attacker $\mathcal{B}$ that can successfully forge the RSA signature $H(M)^d \bmod n$ with probability ϵ', where $|\epsilon - \epsilon'|$ is negligible. $\mathcal{B}$ does the following (all arithmetic is $\bmod\ n$):

1. $\mathcal{B}$, when given (n, e), passes (n, e) to $\mathcal{A}$ and gets and M and η back.

2. $\mathcal{B}$ then computes $h = H(M)$, picks a random z from $[1..2^{t_2}n]$ and sends h^{1+ez} to $\mathcal{A}$. Note that $h^{1+ez} = h^{ed+ez} = h^{e(d+z)}$. Then $\mathcal{B}$ can get $r = \eta^{e(d+z)}h^{-(d+z)}$ from $\mathcal{A}$.

3. Note that $r = \eta^{1+ez}h^{-d}h^{-z}$. As $\mathcal{B}$ knows η, h, e, and z, then $\mathcal{B}$ can compute h^d.

$\mathcal{B}$ succeeds in forging an RSA signature if and only if $\mathcal{A}$ wins the above game, i.e., successfully compute $(\eta^{ey}h^{-y} \mod n)$. What $\mathcal{A}$ receives from the Challenger in the game is drawn from the distribution family $\{h^{e(d+z)} \mid z \leftarrow [1..n2^{t_2}]\}$. What $\mathcal{A}$ receives from $\mathcal{B}$ are drawn from $\{h^{ey} \mid y \leftarrow [1..n2^{t_2}]\}$. Using an argument similar to that in the proof of Theorem 1, it is easy to show that these two distribution families are statistically indistinguishable. Therefore, the difference between $\mathcal{A}$'s success probabilities in the two cases is negligible. ∎

RSA-OSBE does a Diffie-Hellman style key agreement that has the added twist that one party can recover the shared key only when knowing the signature. This construction may be useful for other purposes, in which case the following property of the RSA-OSBE scheme could be useful: no eavesdropping attacker against RSA-OSBE can recover the shared secret with nonnegligible probability, even if the eavesdropper knows the signature h^d. (This property is not required for OSBE because we assume secure communication channels.) We base the security on the CDH (Computational Diffie-Hellman) problem in $\mathbb{Z}_n^*$. 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. It is known that if the CDH problem in $\mathbb{Z}_n^*$ can be solved in polynomial time for a nonnegligible portion of all bases $g \in \mathbb{Z}_n^*$, then n can be factored in expected polynomial time [6].

Theorem 3 *Under the CDH assumption on $\mathbb{Z}_n^*$, where n is an RSA modulus, no eavesdropping attacker against RSA-OSBE can recover the shared secret with nonnegligible probability.*

Proof. We prove that there exists no polynomial bounded algorithm that can solve the following problem with non-negligible probability (all arithmetic is $\mod n$): given an RSA public key (n, e), which has corresponding private key d, and the following tuple $\langle h,\ h^d,\ h^{d+x},\ h^{ey}\rangle$, compute h^{exy}.

Given an algorithm $\mathcal{A}$ that solves the above problem, we construct another algorithm $\mathcal{B}$ that can solve the CDH problem in $\mathbb{Z}_n^*$. $\mathcal{B}$, when given (g, g^a, g^b), picks a small prime e and outputs $\mathcal{A}((n, e), \langle h = g^e,\ g,\ h_2 = gg^a,\ h_3 = (g^b)^e\rangle)$. Let x

denote $(ad \bmod \phi(n))$ and y denote $(bd \bmod \phi(n))$. Observe that $h_2 = (h)^{d+x}$, $h_3 = h^{ey}$; therefore, $h^{exy} = g^{e^2d^2ab} = g^{ab}$. ∎

6 One-round OSBE Using Identity Based Encryption

Next, we show how to implement a one-round OSBE using any Identity Based public key Encryption scheme (IBE). The one-round refers to the fact that during the interaction phase there is only one message — the sender sends a ciphertext to the recipient. As usual, the recipient is only able to decrypt the ciphertext if she has a third party's signature on some predefined message M. Using IBE we build a one-round OSBE where user certificates are signed using a Rabin [18] or BLS [8] signature.

Before we describe the one-round OSBE, we briefly review the concept of Identity Based Encryption. IBE was first proposed by Shamir [22], but the first usable IBE systems were found only very recently [7, 10]. An IBE public key encryption scheme is a public key system with the added twist that any string can function as a public key. In such a system there is a third party that has a secret master-key that enables it to generate the private key corresponding to any public key string. This third party plays a role that is similar to yet different from that played by a Certificate Authority (CA) in a standard PKI. Unlike the CA in a standard PKI, this third party in IBE knows every user's private key. A malicious CA in a standard PKI can cause almost as much damage as a malicious third party in IBE, as the CA can generate a new pair of public/private keys and certify this to be the user's public key, effectively knowing the user's private key. However, the compromise of the IBE master key leads to the immediate compromise of all existing encrypted messages, which is not the case in a standard PKI. Therefore, a higher level of trust on the third party is needed in IBE.

There are also global IBE system parameters given to all users, as is the CA's root certificate in a standard PKI. Shamir's idea was that user Alice uses her name (or email address) as a public key, thus avoiding the need for a public key certificate. Alice obtains her private key from the third party. More details on using IBE can be found in [7].

Any secure IBE system gives rise to a signature scheme [7]: to sign a message M we view M as an IBE public key; the signature on M is the private key corresponding to the public key M. Here the signer has the IBE master-key that enables it to generate the signature on any message M. The main point is that this signature on M can also function as an IBE decryption key. For the two recently proposed IBE systems the associated signature schemes are Rabin signatures and

BLS signatures.

We show how to build an OSBE from any IBE system. As usual, the sender wants to send an encrypted message P to the receiver so that the receiver is able to recover P only if the receiver has the third party's signature on M. The OSBE based on a generic IBE system works as follows:

Setup. The CA takes a security parameter t and two messages $M$and P as input. It runs the setup algorithm of the IBE system to generate the third party's master-key and the global IBE system parameters, which are viewed as PK.

Let $\mathsf{Sig}_{PK}(M)$ be the IBE private key corresponding to M when M is viewed as a public key. The CA gives PK, M, and P to the sender S, gives PK, M, and $\mathsf{Sig}_{PK}(M)$ to R_1, and gives PK and M to R_2 .

Interaction. The sender S encrypts P using M as an IBE public key and sends the resulting ciphertext C. The receivers R_1 and R_2 receive the ciphertext CT from S.

Open. The receiver R_1, using the private key $\mathsf{Sig}_{PK}(M)$, decrypts C to obtain P.

The OSBE described above is clearly oblivious because S receives no information during the interaction phase. The semantic security of this OSBE follows from the security of the IBE system. We summarize this in the following theorem. The theorem refers to the standard notion of security for IBE systems (IND-ID-CCA) defined in [7].

Theorem 4 *Let $\mathcal{E}_{IBE}$ be an IBE system that is semantically secure under a chosen ciphertext attack (*IND-ID-CCA*). Then the resulting OSBE is sound, oblivious, and secure against the receiver.*

Proof. The oblivious property is trivial, as the sender receives no information at all during the interaction phase, and thus cannot tell whether the receiver has the signature or not.

As $\mathsf{Sig}_{PK}(M)$ is the private key corresponding to M, the soundness property of the resulting OSBE scheme is immediate from the soundness property of the IBE scheme (given a private key and a message encrypted under the corresponding public key, one can decrypt the message).

In addition, if the resulting OSBE is not semantically secure against the receiver, then there exists an adversary $\mathcal{A}$ that wins the following game against the Challenger with nonnegligible probability: The Challenger gives PK and M to the adversary. The adversary responds with two messages P_0 and P_1. The Challenger

picks a random $b \in \{0,1\}$ and gives the adversary C, which is the IBE encryption of P_b with M as the public key. The adversary outputs $b' \in \{0,1\}$ and wins if $b' = b$. $\mathcal{A}$ is a direct attacker against the semantic security of the IBE scheme. Therefore, the OSBE is semantically secure when the IBE system is semantically secure. ∎

In Appendix A, we describe an OSBE for Rabin signatures, using Cocks' IBE system [10]. In this OSBE, communication during the interaction phase is quite large. This is because encryption in Cocks' IBE is done bit by bit, and the ciphertext for each bit is a number in $\mathbb{Z}_n$ (about 1024 bits in a typical setting). In the rest of this section, we describe an OSBE for BLS signatures [8], using an IBE system due to Boneh and Franklin [7]. With this OSBE, the amount of communication during the interaction phase is small.

The BLS short signature scheme [8] is based on bilinear maps. A number of recent cryptographic constructions make use of such maps [17, 7, 23, 20]. Let G_1, G_2 be two groups of prime order q. A bilinear map $e : G_1 \times G_1 \to G_2$ satisfies $e(g^x, g^y) = e(g,g)^{xy}$ for any $g \in G_1$ and $x, y \in \mathbb{Z}_q$. Using elliptic curves one can give examples of bilinear maps $e : G_1 \times G_1 \to G_2$ where the Computational Diffie-Hellman problem (CDH) in G_1 is believed to be hard. Throughout this section we let g be a generator of G_1.

The BLS signature scheme works as follows: the public key is $h = g^x \in G_1$ and the private key is $x \in \mathbb{Z}_q^*$. Let H be a hash function from $\{0,1\}^*$ to G_1. To sign a message M the signer computes $\sigma = H(M)^x \in G_1$. To verify a signature on M test that $e(g, \sigma) = e(h, H(M))$. When H is modelled as a random oracle the system is existentially unforgeable under a chosen message attack assuming CDH in G_1 is hard [8]. Note that a BLS signature is a single element of G_1. Using certain elliptic curves, elements in G_1 are represented as short strings, resulting in very short signatures.

To build an OSBE using BLS signatures we use the Boneh-Franklin IBE system [7]. We do not describe the system here, but note that in this IBE system, the private key corresponding to a public key $M \in \{0,1\}^*$ is exactly a BLS signature on M. Thus we can build a one-round OSBE out of this system. The advantage of this IBE system is that the encryption of a 128-bit message key results in a short ciphertext (two elements in a finite field). Encryption and decryption are also more efficient than in Cocks' system.
The OSBE works as follows:

Setup. The CA takes a security parameter t and two messages $M$and P as input. It generates a bilinear map $e : G_1 \times G_1 \to G_2$, picks a random $x \in \mathbb{Z}_q^*$, and compute $h = g^x \in G_1$. Let $PK = (G_1, G_2, h)$. Let $\mathsf{Sig}_{PK}(M)$ be the BLS signature on M, i.e., $\mathsf{Sig}_{PK}(M) = H(M)^x \in G_1$.

The CA gives to the sender S: PK, M, P, gives to R_1: $PK, M, \mathsf{Sig}_{PK}(M)$, and gives to R_2: PK, M.

Interaction. The sender S encrypts P using M as the public key and sends the resulting ciphertext CT. The public key M is only used to encrypt a message key k which is then used to encrypt P. The receivers R_1 and R_2 receive the ciphertext CT from S.

Open. The receiver R_1, using the private key $\mathsf{Sig}_{PK}(M)$, decrypts the ciphertext CT to obtain P.

The security of this OSBE follows from the security of BLS signatures [5] and the security of the Boneh-Franklin IBE [7], which is based on the assumption that the bilinear Diffie-Hellman problem is hard for the pairing $e : G_1 \times G_1 \rightarrow G_2$. See [7] for more details. We summarize this in the following corollary of Theorem 4.

Corollary 5 *In the Random Oracle Model, the OSBE above is sound, oblivious, and secure against the receiver, assuming that the bilinear Diffie-Hellman problem is hard for* $e : G_1 \times G_1 \rightarrow G_2$.

7 Generalized OSBE

OSBE guarantees that, for the receiver to receive a message, it must possess one specific certificate. This enforces a policy that requires the receiver to have one or more attributes documented by the certificate. However, in many scenarios, a policy can be more complicated, requiring simultaneous possession of attributes proved by multiple certificates or allowing multiple ways of satisfying the policy. For example, a policy may require that the receiver is either a student in a university or a member of a club, and, at the same time, is older than 21. This requirement involves three certificates issued by different certificate authorities: c_1 (student), c_2 (club membership), and c_3 (age ≥ 21). Bob must satisfy $(c_1 \vee c_2) \wedge c_3$.

We introduce a notion called Generalized Oblivious Signature-Based Envelope (GOSBE) to handle more sophisticated policy requirements. In GOSBE, the sender and the receiver share the description of a policy, which is specified using a boolean circuit. The circuit has ℓ inputs and one output, each input i is associated with a pair $\langle PK_i, M_i \rangle$, where PK_i is a public key and M_i is a message. The circuit consists of AND gates and OR gates; each gate has two or more inputs and one output. Intuitively, a receiver makes an input true if it possesses $\mathsf{Sig}_{PK_i}(M_i)$. A receiver satisfies the policy if it makes the output of the circuit true.

We now describe a GOSBE protocol that uses OSBE as a sub-protocol. In this protocol, the sender associates a symmetric encryption key with each circuit input

and each gate output. The key associated with the circuit output is used to encrypt the message P to be sent to the receiver. The receiver recovers the key associated with the circuit output if and only if it satisfies the policy specified by the circuit.

Definition 3 (A GOSBE Protocol) The sender does the following steps.

1. For each $i = 1..\ell$, the sender chooses a random key k_i and runs an OSBE protocol with the receiver, sending k_i in an envelope that can be opened only when the receiver has $\mathsf{Sig}_{PK_i}(M_i)$.

2. The sender computes the keys associated with (the output of) each gate as follows, starting from the bottom of the circuit. (We assume that circuit evaluation goes from bottom up; the input wires are located at the bottom and the output wires are located at the top).

   For an AND gate, let $k^{(1)}$, $k^{(2)}$, ..., $k^{(m)}$ be the keys associated with the m inputs, then the key corresponding to the output is $k = k^{(1)} \oplus k^{(2)} \oplus \ldots \oplus k^{(m)}$.

   For an OR gate, let $k^{(1)}$, $k^{(2)}$, ..., $k^{(m)}$ be the keys associated with the m inputs. The sender chooses a random key k as the output key. The sender then encrypts k under each of $k^{(1)}$, $k^{(2)}$, ..., $k^{(m)}$, and sends the m ciphertexts to the receiver.

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

The receiver runs ℓ instances of the OSBE protocol, one for each input. The receiver also receives, for each OR gate, m ciphertexts. Finally, the receiver also receives C, a ciphertext of P. The receiver tries to recover the output key as follows.

1. For each $i = 1..\ell$, if the receiver has $\mathsf{Sig}_{PK_i}(M_i)$, then the receiver recovers k_i, which was sent using OSBE.

2. The receiver tries to recover the keys associated with each gate, starting from the bottom.

   For an AND gate, if the keys associated with all the inputs are known, let them be $k^{(1)}$, $k^{(2)}$, ..., $k^{(m)}$, then the key corresponding to the output is $k = k^{(1)} \oplus k^{(2)} \oplus \ldots \oplus k^{(m)}$.

   For an OR gate, if the key associated with the j'th input is known, then let $k^{(j)}$ be the input and c_{ij} be the ciphertext. Use $k^{(j)}$ to decrypt c_{ij} to obtain the key corresponding to the output.

3. If the receiver successfully recovers the output key, it decrypts C to get P.

By the property of OSBE, the receiver recovers the key associated with an input if and only if it has the corresponding certificate. Furthermore, the receiver recovers the key associated with an AND gate if and only if it recovers the keys associated with all of the gate's inputs, and the receiver recovers the key associated with an OR gate if and only if it recovers the key associated with any one of the gate's inputs. Thus the receiver recovers the key associated with the circuit output if and only if it satisfies the policy.

Given a policy expressed using a circuit, the cost of GOSBE is linear in the size of the circuit. GOSBE requires ℓ OSBE's, where ℓ is the number of inputs to the circuit, i.e., the number of certificates mentioned in the policy. In addition, the sender sends the ciphertext of P and the ciphertexts of N intermediate keys, where N is the sum of the number of inputs of the OR gates. Clearly, N is bounded by the number of edges in the circuit.

We use an example to illustrate the entire procedure. We use $OSBE(k, c)$ to denote the OSBE protocol in which the sender sends k such that the receiver can recover k only if it possesses the certificate c. In the example, Alice wants to send P to Bob while ensuring that Bob can read P only if he satisfies $(c_1 \vee c_2) \wedge c_3 \wedge (c_4 \vee c_5 \vee c_6)$, where c_i, for $i = 1 \ldots 6$, represents certificates. Fig. 1 depicts the entire procedure.

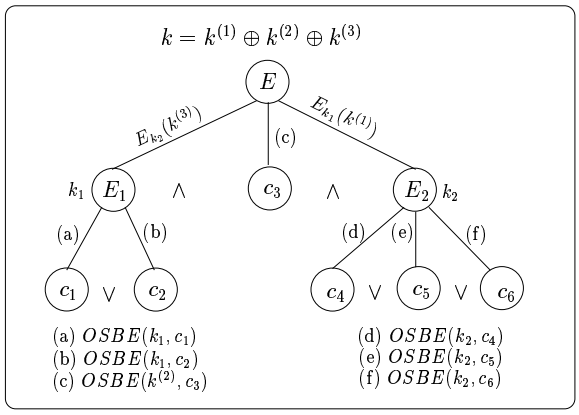

Figure 1: An Example

First, Alice generates three secret keys k, k_1, k_2; she also generates three other keys $k^{(1)}$, $k^{(2)}$, and $k^{(3)}$, such that $k = k^{(1)} \oplus k^{(2)} \oplus k^{(3)}$. Second, Alice sends $E_{k_1}(k^{(1)})$ and $E_{k_2}(k^{(3)})$ to Bob. Then Alice uses the following OSBE protocol to send k_1, k_2, and $k^{(2)}$ to Bob using the corresponding certificates, i.e., Alice and

Bob conduct $OSBE(k_1, c_1)$, $OSBE(k_1, c_2)$, $OSBE(k^{(2)}, c_3)$, $OSBE(k_2, c_4)$, $OSBE(k_2, c_5)$, and $OSBE(k_2, c_6)$.

From the procedure, we can see that when Bob has either c_1 or c_2, he can learn k_1, thus $k^{(1)}$; when Bob has c_3, he can learn $k^{(2)}$; when Bob has c_4, c_5, or c_6, he can learn k_2, thus $k^{(3)}$. Therefore, if he satisfy the entire requirement $(c_1 \vee c_2) \wedge c_3 \wedge (c_4 \vee c_5 \vee c_6)$, he can learn $k = k^{(1)} \oplus k^{(2)} \oplus k^{(3)}$.

8 Conclusion

Automated Trust Negotiation (ATN) is an approach to regulate the flow of sensitive information. Previous work on ATN, which only uses access control techniques, cannot deal with cyclic policy interdependency satisfactorily. We showed that cyclic policy interdependency in ATN can be handled by solving a particular 2-party Secure Function Evaluation (SFE) problem. We introduced oblivious signature-based envelope (OSBE) as a solution to the SFE problem and mentioned that OSBE can be used in other privacy sensitive applications as well. We developed an OSBE protocol for RSA signatures. The protocol does not involve a third party, is provably secure and quite efficient. We also showed that identity-based encryption can be used to build efficient one-round OSBE for Rabin and BLS signatures. We also presented constructions for Generalized OSBE, where signatures on multiple messages (an possibly by different authorities) are required to open the envelope.

An open problem is to find an efficient and provably secure OSBE scheme for DSA signatures. It would also be interesting to investigate other applications of the OSBE concept.

Acknowledgement

We would like to thank Will Winsborough for helpful discussions. We also thank anonymous reviewers for their suggestions that helped us improve the paper.

References

[1] Bill Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In *Advances in Cryptology: EUROCRYPT '01*, pages 119–135, May 2001.

[2] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18(4):591–610, April 2000.

[3] 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.

[4] Feng Bao, Robert H. Deng, and Wenbo Mao. Efficient and practical fair exchange protocols with off-line TTP. In *Proceedings of the 1998 IEEE Symposium on Security and Privacy*, pages 77–89. IEEE Computer Society Press, May 1998.

[5] 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.

[6] Eli Biham, Dan Boneh, and Omer Reingold. Breaking generalized Diffie-Hellman modulo a composite is no easier than factoring. *Information Processing Letters*, 70(2):83–87, 1999.

[7] 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.

[8] 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.

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

[10] 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.

[11] 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.

[12] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. *JCSS*, 60(3):592–629, 2000. Preliminary version in STOC'98.

[13] Oded Goldreich. *The Foundations of Cryptography — Volume 2*. Cambridge University Press, May 2004.

[14] 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.

[15] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[16] 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.

[17] Antoine Joux. A one round protocol for tripartite Diffie-Hellman. In *Proceddings of the 4th Algorithmic Number Theory Symposium*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer, 2000.

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

[19] 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.

[20] 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.

[21] Kent E. Seamons, Marianne Winslett, and Ting Yu. Limiting the disclosure of access control policies during automated trust negotiation. In *Proceedings of the Symposium on Network and Distributed System Security (NDSS'01)*, February 2001.

[22] 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.

[23] Eric R. Verheul. Self-blindable credential certificates from the weil pairing. In *Advances in Cryptology: AsiaCrypt 2001*, number 2248 in Lecture Notes in Computer Science, pages 533–551. Springer, 2001.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] Ting Yu and Marianne Winslett. Unified scheme for resource protection in automated trust negotiation. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 110–122. IEEE Computer Society Press, May 2003.

[29] 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*, 6(1):1–42, February 2003.

A One-round OSBE with Rabin Signatures

The Rabin signature scheme is similar to RSA, but one uses a public exponent $e = 2$, i.e., a signature on a message M is $H(M)^{1/2} \bmod N$. One just has to make sure that the square root exists.

To define Rabin signatures [18], let $n = pq$ be an RSA modulus with $p = q = 3 \bmod 4$. The public key is n and the signing key is p, q. Let $Q \subseteq \mathbb{Z}_n^*$ be the subset of $\mathbb{Z}_n^*$ containing all elements with Jacobi symbol 1. We know that the size of Q is approximately $n/2$. Let H be a hash function from $\{0,1\}^*$ to Q. Then for any $M \in \{0,1\}^*$ exactly one of $H(M)$ and $-H(M)$ is a quadratic residue in $\mathbb{Z}_n^*$.

To sign a message M the signer computes $\mathsf{Sig}(M) = (\pm H(M))^{1/2} \bmod n$ where the sign of $H(M)$ is chosen so that the square root exists. To verify the signature, test that $(\mathsf{Sig}(M))^2 = \pm H(M) \bmod n$. When H is modelled as a random oracle the system is existentially unforgeable under a chosen message attack assuming factoring RSA moduli is hard [5].

To build an OSBE using Rabin signatures we use Cocks' IBE system [10]. A private key in this system can be viewed as a Rabin signature of the public key. Cocks' IBE works as follows: the global parameters are simply "n" where $n = pq$ is an RSA modulus with $p = q = 3 \bmod 4$. The master-key is p, q. The private key corresponding to a public key $M \in \{0,1\}^*$ is $s = (\pm H(M))^{1/2} \bmod n$ (the sign of $H(M)$ is chosen so that the square root exists). To encrypt a plaintext bit $b \in \{0,1\}$ using the public key M one picks two random numbers $x_0, x_1 \in \mathbb{Z}_n^*$ such that the Jacobi symbols $\left(\frac{x_0}{n}\right) = \left(\frac{x_1}{n}\right) = (-1)^b$. The ciphertext is a pair (C_0, C_1) where $C_i = x_i + ((-1)^i H(M)/x_i) \bmod n$ for $i = 0, 1$. Suppose $H(M)$ is a quadratic residue in $\mathbb{Z}_n^*$. Then to decrypt a ciphertext (C_0, C_1), one computes the Jacobi symbol $\left(\frac{C_0+2s}{n}\right)$ which one can show is equal to $(-1)^b$ as required. If $-H(M)$ is a quadratic residue we use C_1 instead. The system can be shown to be semantically secure under a chosen ciphertext attack (IND-ID-CCA) in the random oracle model assuming that the problem of distinguishing quadratic residues from non-residues in Q is hard.

Note that in this system encryption of a plaintext P is done bit-by-bit. Thus, encrypting a 128-bit message key results in a long ciphertext – the ciphertext contains 256 elements in $\mathbb{Z}_n^*$. Nevertheless, this system gives a one-round OSBE using Rabin signatures.

The OSBE works as follows:

Setup. The CA takes a security parameter t and two messages $M$and P as input. It generates an RSA modulus $n = pq$ where $p = q = 3 \bmod 4$. Let $PK = n$. Let $\mathsf{Sig}_{PK}(M)$ be the Rabin signature on M, i.e., $\mathsf{Sig}_{PK}(M) = (\pm H(M)^{1/2}) \bmod n$.

The CA gives to the sender S: PK, M, P, gives to R_1: $PK, M, \mathsf{Sig}_{PK}(M)$, and gives to R_2: PK, M.

Interaction. The sender encrypts P bit-by-bit using M as the public key in Cocks' IBE and sends the resulting ciphertext CT to the receiver. For efficiency, one could pick a random block cipher message key k, encrypt P using k, and then encrypt k bit-by-bit using M as the public key.

The receivers R_1 and R_2 receive the ciphertext CT from S.

Open. The receiver R_1, using the private key $\mathsf{Sig}(M)$, decrypts the ciphertext CT to obtain P.

The security of this OSBE follows from the security of Rabin signatures [5] in the random oracle model. We summarize this in the following corollary of Theorem 4.

Corollary 6 *In the Random Oracle Model, the OSBE above is sound, oblivious, and secure against the receiver, assuming that the problem of distinguishing quadratic residue from non-residues in Q is hard.*