Provably Secure Timed-Release Public Key Encryption

JUNG HEE CHEON
Seoul National University, Korea
and
NICHOLAS HOPPER, YONGDAE KIM and IVAN OSIPKOV
University of Minnesota - Twin Cities

A timed-release cryptosystem allows a sender to encrypt a message so that only the intended recipient can read it only after a specified time. We formalize the concept of a secure timed-release public-key cryptosystem and show that, if a third party is relied upon to guarantee decryption after the specified date, this concept is equivalent to identity-based encryption; this explains the observation that all known constructions use identity-based encryption to achieve timed-release security. We then give several provably-secure constructions of timed-release encryption: a generic scheme based on any identity-based encryption scheme, and two more efficient schemes based on the existence of cryptographically admissible bilinear mappings. The first of these is essentially as efficient as the Boneh-Franklin Identity-Based encryption scheme, and is provably secure and authenticated in the random oracle model; the final scheme is not authenticated but is provably secure in the standard model (i.e., without random oracles). Categories and Subject Descriptors: E.3 [Data]: Data Encryption—Public Key Cryptosystems General Terms: Security.

Theory Additional Key Words and Phrases: timed-release, authenticated encryption, key-insulated encryption

1. INTRODUCTION

The goal of timed-release cryptography is to “send a message into the future.” One way to do this is to encrypt a message such that the receiver cannot decrypt the ciphertext until a specific time in the future. Such a primitive would have many practical applications; a few examples include preventing a dishonest auctioneer from prior opening of bids in a sealed-bid auction [Rivest et al. 1996], preventing early opening of votes in e-voting schemes, fair exchange, release of classified information, and delayed verification of a signed document,

such as electronic lotteries [Syverson 1998] and check cashing. The problem of timed-release cryptography was first mentioned by [May 1993] and then discussed in detail by [Rivest et al. 1996]. Let us assume that Alice wants to send a message to Bob such that Bob will not be able to open it until a certain time. Previous solutions fall into two categories:

—Time-lock puzzles: Alice encrypts her message so that Bob needs to perform non-parallelizable computation without stopping for the required time to decrypt it. If Alice accurately predicts Bob's computing resources between now and the desired time, then Bob recovers the message.

—Trusted decryption agents: Alice encrypts a message such that Bob needs some secret value, published by a trusted agent on the required date, in order to decrypt the message. Once the agent releases the information, Bob can decrypt the message.

The first approach puts considerable computational overhead on the message receiver, which makes it undesirable for real-life scenarios. In addition, knowing the computational complexity of decryption, while giving us a lower bound on the computing time Bob may need to decrypt the message, does not guarantee that the plaintext will be available at a certain date. Still, this approach is widely used for specific applications [Boneh and Naor 2000; Belare and Goldwasser 1996; Syverson 1998; Garay and Pomerance 2002, 2003]. The agent-based approach, on the other hand, relieves Bob from performing nonstop computation, sets the date of decryption precisely and does not require Alice to have information on Bob's capabilities. This comes at a price, though: the agents have to be trusted and they have to be available at the designated time.

In this article we concentrate on schemes that use such "decryption agents." We formalize this notion of a secure timed-release encryption scheme and show that it is equivalent to the notion of strongly key-insulated encryption [Dodis et al. 2002]; when there is no a priori bound on the number of time periods, this notion is, in turn, known to be equivalent to identity-based encryption, or IBE [Bellare and Palacio 2002]. We also give several provably-secure constructions of timed-release public-key encryption, including the first provablysecure generic construction in the literature, and the first efficient scheme that is provably secure in the standard model, that is, without random oracles.

Our results also cast new light on several previous schemes that appear in the literature: each can be seen as an adaptation of a known key-insulated encryption scheme. For example, Rivest et al. [1996] propose that the agent could encrypt messages on request with a secret key which will be published on a designated date by the agent, or the agent can precompute pairs of public/private keys, publish all public keys and release the private keys on the required days; these exactly fit known key-insulated schemes appearing in the literature. The scheme of Crescenzo et al. [1999] essentially replaces publication of the key with publication of the message, requiring the receiver to engage in a conditional oblivious transfer protocol with the agent to decrypt the message. In Chen et al. [2002], the authors proposed to use Boneh and Franklin's IBE scheme [Boneh and Franklin 2003] for timed-release encryption:

essentially, the scheme replaces the identity in an IBE scheme with the time of decryption. Similar proposals appear in Marco Casassa Mont and Sadler [2003] and Blake and Chan [2005].

While some of the above proposals contain informal proofs of security, none of them consider and/or give a formal treatment of the security properties of timed-release public key encryption (or TR-PKE). The first formal treatments of TR-PKE security were displayed in Cheon et al. [2004] and then strengthened in Cheon et al. [2006]. Independently, Cathalo et al. [2005] introduce another notion of timed-release security and argue that it is not implied by key-insulated encryption; however, this seems to be a side effect of an overly-restrictive model in which a user must commit to a specific decryption agent before choosing his public key.

Authentication for Timed-Release Encryption. Many of the applications of timed-release cryptographymentioned above require some form of authentication as well. For example, if there is no authentication of bids in a sealed-bid auction, any bidder may be able to forge bids for others, or force the auction to fail by submitting an unreasonably high bid. In this article, we consider the security properties required by these applications and develop formal security conditions for a Timed-Release Public Key Authenticated Encryption (TR-PKAE) scheme.

One avenue for developing a TR-PKAE scheme would be composing an unauthenticated TR-PKE scheme with either a signature scheme or a (non-timed-release) PKAE scheme. Although such constructions are possible, we note that the details of this

composition are not trivial; examples from An [2001] and Dodis and Katz [2005] illustrate that naive constructions can fail to provide the expected security properties. Additionally, we note that such schemes are likely to suffer a performance penalty relative to a scheme based on a single primitive. Thus, besides introducing a generic construction, we also introduce a provably secure construction of a TR-PKAE scheme that is essentially as efficient as previous constructions of non-authenticated TR-PKE schemes [Chen et al. 2002; Marco Casassa Mont and Sadler 2003; Blake and Chan 2005].

ACM Transactions on Information and Systems Security, Vol. 11, No. 2, Article 8, Pub. date: May 2008.

2. DEFINITIONS

In this section we review security definitions that will be used in the article. In addition, we introduce new definitions, namely, those of timed-release public key encryption (TR-PKE) and authenticated TR-PKE (TR-PKAE).

Identity Based Encryption. Formally, we define an IBE scheme IBES to be a tuple of four randomized algorithms:

—SetupIBE(1k), which given input 1k (the security parameter), produces public parameters π_{IBE}, which include hash functions, message and ciphertext spaces among others. In addition, master secret δ_{IBE} is generated which is kept confidential by the central authority.

—ExtractIBE(π_{IBE}, δ_{IBE}, I), given public parameters π_{IBE}, master secret δ_{IBE} and identity $I \in \{0, 1\}^*$, outputs a secret key skI . The I (together with π_{IBE}) serves as the public key corresponding to identity I.

—EncryptIBE(π_{IBE}, I,m) computes the ciphertext c denoting the encryption for identity I of message mwith public parameters π_{IBE}.

—DecryptIBE(π_{IBE}, skI ,bc) outputs the plaintext corresponding tobc if decryption is successful or the special symbol “fail” otherwise.

For consistency, we require that DecryptIBE(π_{IBE}, skI ,EncryptIBE(π_{IBE}, I,m)) = m, for all valid (I, skI), (π_{IBE}, δ_{IBE}), and m. We use the IND-ID-CCA notion of security for and IBE scheme [Boneh and Franklin 2003]. Briefly, in this case, an adversary may

adaptively ask for secret keys corresponding to arbitrary identities, and may also ask for decryption of any ciphertext using any identity. Eventually the adversary presents a “challenge identity” and a pair of “challenge plaintexts” and is given the encryption of one of these plaintexts under the challenge identity. The adversary may then continue to ask for secret keys and decryptions, except that it cannot query for the secret key of the challenge identity or for decryption of the challenge ciphertext under the challenge identity. The adversary wins if it can correctly guess which of the challenge ciphertexts was encrypted by the challenger, and the scheme is secure if no polynomial time adversary wins with an advantage non-negligibly greater than one half. Public Key Encryption. A public key encryption system PKE consists of three algorithms:

—KeyGenPKE, which on input 1k, outputs public/private key pair (pk, sk). The public key also includes public parameters needed for encryption/decryption.

—EncryptPKE, which on input of pk and message m, outputs ciphertext c.

—DecryptPKE, which on input of ciphertextbc and private key sk, outputs either some message bmor failure symbol.

For consistency, it is required that DecryptPKE(sk,EncryptPKE(pk,m)) = m, for all valid (pk, sk) and m.

ACM Transactions on Information and Systems Security, Vol. 11, No. 2, Article 8, Pub. date: May 2008.

Provably Secure Timed-Release Public Key Encryption· 8: 5

We make use of a PKE that is IND-CCA2 secure against adaptive adversary as described in Bellare et al. [1998]. Briefly, the challenger generates a public private key pair and gives the public key to the adversary. The adversary is allowed to query for the decryption of any ciphertext using the private key. In the challenge step, the adversary produces a pair of challenge plaintexts and is given the encryption of one of the pair. The adversary wins if, given the ability to query the decryption of any message but the challenge ciphertext, it can correctly guess which of the two plaintexts was encrypted in the challenge step.

We note that given a secure IBES, we can easily obtain a secure PKE. For that purpose, each user simply runs IBES’s SetupIBE and ExtractIBE, using an arbitrary identity I, to obtain its public key and private key (i.e., the master secret key in IBES). The identity I along

with IBES's public parameters serves as user's public key, while the master secret key serves as the private key. A straightforward argument shows that if IBES is IND-ID-CCA secure then the corresponding PKE is IND-CCA2 secure. However, since in practical applications we expect one to use more efficient PKE constructions, we make use of separate IBE and PKE schemes in this article.

Digital Signatures and Labels. In addition to the above primitives, we will also use signature schemes. We start with review of standard signatures first. A signature scheme DS consists of three algorithms:

—SigGen, which on input 1k, outputs signing/verification key pair (SK, VK). The VK also includes also public parameters such as the message space among others.

—Sig, which on input SK and message m, outputs signature σ.

—Ver, which on input message m, signature σ and VK, outputs either true or false.

For consistency, it is required that for every valid pair (SK, VK) and message m, VerVK(m, SigSK(m)) = true. We will use the notion of strong unforgeability under adaptive chosen plaintext attacks (SUF-CMA). Briefly, the challenger generates a (SK, VK) pair and gives the VK to the adversary. The adversary is given signatures σ1, σ2, . . . , σq on adaptively chosen messages m1,m2, . . . ,mq and outputs a pair (m, σ). The adversary wins if (m, σ) is a valid message signature pair and is not equal to any pair (mi, σi).

Beside standard signatures, we will also use one-time signatures which are defined analogously, except that in SUF-CMA the adversary is allowed to make only one query. Any public-key signature that is SUF-CMA secure is also a secure one-time signature. However, the opposite is obviously not true and one-time signatures are generally much more efficient.

We can also add public labels to IBE and PKE encryption/decryptionmechanisms, which are bound in a nonmalleable way to the ciphertext [Shoup 2004] while preserving security. In effect, ciphertext generation additionally takes as input a label, which becomes part of the ciphertext. When decrypting, one applies not only the decryption key but also the public label.

and IND-CCA2 games can be modified in a natural way to take labels into account.

1 Timed-Release Public Key Encryption (TR-PKE)

In this section we formalize the functionality and security requirements for a timed-release public key encryption system. These requirements are meant to capture the implicit security requirements not addressed in previous work [May 1993; Rivest et al. 1996; Chen et al. 2002; Marco Casassa Mont and Sadler 2003; Blake and Chan 2005]; in particular they do not address the authentication requirements, which we add in Section 2.3. Informally, we can think of any principal in a TR-PKE system as filling one or more of three roles. The timed-release agent—or TiPuS (TImed-release PUblic Server)—publishes a timed-release public key and releases "tokens" that allow decryption of messages encrypted for the current time at regular intervals. The receiver publishes a public key that allows others to encrypt messages so that only he can decrypt them, using a secret key that he keeps private, and the appropriate timed-release token. The sender uses the receiver's public key and the TiPuS public key to encrypt messages that can later be decrypted at the time of his choice.

1.1 Functional Requirements. Formally, we define a timed-release

public-key encryption system $\hat{W}$ to be a tuple of five randomized algorithms:

—Setup, which given input 1k (the security parameter), produces public parameters πg, which include hash functions, message, and ciphertext spaces among others.

—TRSetup, which on input πg, produces a pair (δ, πtr) where δ is a master secret and πtr the corresponding timed-release public parameters. This setup is carried out by TiPuS which keeps the master secret key confidential, while all other parameters are public. We denote the combined public parameters of πg and πtr by π.

—KeyGen, given public parameters πg, outputs a pair of secret key and public key (sk, pk).

—TG(π, δ, T) computes the token tknT corresponding to time T using (δ, π). This functionality is performed by TiPuS which publishes tknT at time T.

—Encrypt(π, pk,m, T) computes the timed-release ciphertext c denoting the encryption of message m using public key pk, public parameters π and time encoding T.

—Decrypt(π , sk,bc, tknT) outputs the plaintext corresponding tobc if decryption is successful or the special symbol “fail” otherwise.

For consistency, we require that Decrypt(π , sk,Encrypt(π , pk,m, T),TG (π , δ , T)= m, for all valid (pk, sk), (π , δ), T, and m. Unlike the functional requirements specified in Cathalo et al. [2005], we explicitly separate the functions TRSetup and KeyGen, allowing a user to generate keys independent of any timed-release server. This allows the sender to choose which servers to trust during encryption.