A Closer Look at PKI: Security and Efficiency

Alexandra Boldyreva[1] and Marc Fischlin[2] and Adriana Palacio[3] and Bogdan Warinschi[4]

[1]Georgia Institute of Technology, USA. sasha@gatech.edu

[2]Darmstadt University of Technology, Germany. marc.fischlin@gmail.com

[3]Bowdoin College, USA. apalacio@bowdoin.edu

[4]University of Bristol, UK. bogdan@cs.bris.ac.uk

Abstract. In this paper we take a closer look at the security and efficiency of public-key encryption and signature schemes in public-key infrastructures (PKI). Unlike traditional analyses which assume an "ideal" implementation of the PKI, we focus on the security of joint constructions that consider the certification authority (CA) and the users, and include a key-registration protocol and the algorithms of an encryption or a signature scheme. We therefore consider significantly broader adversarial capabilities. Our analysis clarifies and validates several crucial aspects such as the amount of trust put in the CA, the necessity and specifics of proofs of possession of secret keys, and the security of the basic primitives in this more complex setting. We also provide constructions for encryption and signature schemes that provably satisfy our strong security definitions and are more efficient than the corresponding traditional constructions that assume a digital certificate issued by the CA must be verified whenever a public key is used. Our results address some important aspects for the design and standardization of PKIs, as targeted for example in the standards project ANSI X9.109.

Keywords: PKI, public-key encryption, digital signature, provable security.

1 Introduction

Public key cryptography implicitly relies on the existence of a *public-key infrastructure* (PKI), where each user has a pair of public and secret keys for the cryptosystem, and that this association is publicly available. The designers of public-key cryptosystems always define how the public and the secret keys are generated and used, but almost never carefully specify how the binding between keys and user identities takes place. The tacit assumption is that this binding is established *a priori* through PKI management operations.

1.1 Motivation

The policies and the procedures regarding PKIs are continuously changing and detailed descriptions are invariably long and tedious.[1] Unfortunately, existing

[1] See for example the document that describe the current state-of-the-art: "Internet X.509 Public Key Infrastructure – Certificate Management Protocol (CMP)" [1].

literature still does not answer several important questions. What exactly is the certification authority (CA), the entity that links public keys to identities, trusted not to do? Can and should some degree of security be ensured even when the CA is malicious or becomes compromised? Proofs of possession (POP) —in which a user proves possession of the secret key when registering a public key with the CA— are a defense mechanism for protecting against rogue-key and key-substitution attacks, but what exactly should they be and, more importantly, are they really necessary?

A question that is perhaps even more important is whether provably-secure encryption and signature schemes are indeed secure when used in a particular PKI. Although it is largely believed to be the case, the question is far from moot since most existing schemes are analyzed in settings where compositional aspects are neglected. In particular, the security of the combination of a key-registration protocol with existing encryption or signature schemes does not immediately follow from the security of the individual components. In principle, by cleverly combining its ability to attack the key-registration protocol and its ability to attack the primitive (encryption or signatures), an adversary could mount a successful attack against the joint construction.

Limitations of security analyses that do not explicitly include the behavior of the CA or the key-registration protocol have been previously pointed out in other contexts. In the case of key exchange, Shoup [41] suggests that registration of public keys should be considered explicitly as part of the key agreement protocol to be analyzed. Kaliski [27] exemplifies the importance of such measures by presenting unknown key-share attacks on the MQV key exchange protocol [34]. These attacks could have been discovered with a thorough analysis that considers the CA as an active party participating in the protocol. We review further related work at the end in Section 5.

1.2 Contributions

In this paper we initiate a study of PKIs with respect to security of the two most important public-key primitives: encryption and digital signature schemes. Our main motivation is to answer the questions raised above and other related issues.

Models. Security arguments in the absence of rigorous models do not provide strong security guarantees, and such models are conspicuously absent in the case of PKIs. Our first contribution are rigorous definitions for primitives when used in this setting together with appropriate security notions. The inherent complexity of the PKI settings, the non-typical adversarial powers, and the difficulty of precisely identifying the situations that constitute a security breach make the design of such models an entirely non-trivial task.

Since security goals depend on the primitive used, we treat the cases in which keys are used for encryption and for signing separately. Specifically, we define two primitives, called *certified encryption* and *certified signature* schemes, and for each primitive we define a notion of security. Besides the standard algorithms for

encryption and signing, we model explicitly interactive protocols for registering the public keys with a CA. Consequently, our security notions are against an adversary with broad capabilities that take into account threats arising from the key-registration protocol, possibly run concurrently, the presence of several parties, including the users and the (possibly corrupt) CA. The details are in Sections 2, 3 and 4.

Our security definitions are general and powerful. The models we propose directly capture settings where users have multiple public keys, and where keys have additional attributes, such as an expiration date. They easily extend to handle hierarchical certification and certificate revocation. Moreover, while we capture the original goal for which PKI was invented we make flexible assumptions on how certification is achieved. In particular, schemes that aim at achieving certification but avoid the original mechanism of explicit certificates specific to the traditional PKIs (e.g. schemes similar to those in [21, 2]) can still be analyzed in our models. We provide a detailed discussion in Sections 3 and 4.

The design of our models in general and that of the security goals in particular are motivated by the "core" properties of the primitives, namely, confidentiality for encryption and integrity and authenticity for signatures. For protocols in which encryption schemes or signatures are used beyond these basic properties, e.g., encryption schemes used as commitments, additional analysis in light of the new goals is required. Yet, our attack model should be easily transferable to those scenarios, and only the security definitions would need to be adapted.

Analysis of Traditional Schemes. Next we focus on constructions that satisfy the proposed notions of security. We start with an analysis of "traditional" certified encryption and certified signature schemes. In these constructions, the CA uses a signature scheme to issue digital certificates, and then parties produce ciphertexts (resp., signatures) using a standard encryption (resp., signature) scheme. These schemes are defined in detail in Sections 3 and 4, respectively.

Although it seems folklore that the traditional approach is "secure", to the best of our knowledge no formal validation in a sound model with respect to clearly expressed security goals has been devised prior to our work. We offer a rigorous analysis that shows that these schemes are indeed secure in the appropriate security model we design. Our proof gives concrete security bounds that support recommendations for practical parameter choices. While expected, these results are important to increase confidence in the use of the schemes and allow to make security statements based on solid foundations. Our concrete security results are in Sections 3 and 4.

The results that we obtain regarding the design of proofs of possession are less expected, if not surprising. Our investigation shows that formal proofs of knowledge are not necessary for basic security of the certified encryption and signature schemes, and that simpler challenge-response protocols suffice. For signatures, the user simply signs a distinct message[2] provided by the CA. Perhaps

[2] It is necessary to ensure that this message will not be signed by this user later. One way to achieve this, which is also our approach, is to prepend the "challenge" messages chosen by the CA with `0`, and the messages the user signs with `1`.

surprisingly, we show that for basic encryption no proof of possession is required. Intuitively, in the case of encryption, this means that data privacy is not compromised if a user does not have the secret key associated to the public key it registers. We note that these results do not eliminate the proof-of-knowledge requirements imposed on these primitives in other settings (e.g., [4, 11, 9, 24, 32, 36]) and only concern the security of certified encryption and signatures.

More efficient constructions. Since our models do not require that solutions use explicit certificates as in the traditional constructions, it is natural to ask if it is possible to obtain improvements over the traditional solutions, e.g., in terms of efficiency. We answer this question affirmatively. We present more efficient constructions for certified encryption and certified signature schemes that use implicit certificates therefore avoiding the explicit verification of the binding between public keys and identities.

Our certified encryption scheme uses a variant of ElGamal encryption [19] combined with implicit certificates realized through Schnorr signatures, and is proven secure according to our definition in the random oracle model [6] under the Computational Diffie-Hellman assumption. This scheme is more efficient than the traditional certified encryption scheme where the CA uses Schnorr signatures to issue explicit certificates and users employ ElGamal encryption[3]. For security parameter k the latter requires $4.75k$ modular multiplications to encrypt (using the square-and-multiply exponentiation method combined with well-known speed-up techniques for multi-exponentiations) while our scheme only requires $3.25k$ multiplications, coming thus quite close to the performance of regular ElGamal encryption without certification.

For signatures, we propose a construction based on Schnorr signatures [38], provably secure according to our definition in the random oracle model under the Discrete Logarithm assumption. Compared to the traditional approach of using such signatures as explicit certificates, our solution reduces the average number of modular multiplications for verification from $3.5k$ to $1.875k$, and thus achieves almost the same efficiency as regular Schnorr signatures without certification. Notice that the increase in efficiency comes at the expense of a loss in provable security due to looser reductions. It is an open problem to find tighter reductions.

We define the schemes and provide concrete security results in [10]. We note that in the stateful settings where valid certificates of the other parties are stored permanently, traditional schemes are the expedient choice. For the stateless case, however, our constructions offer computational savings over the traditional approach.

2 Modeling Public-Key Infrastructures

To model public-key infrastructures we assume that there is a designated party, the certification authority (CA), and a set of users. Each user has a unique identity $\mathsf{ID} \in \{0,1\}^*$ in form of an X.509 entry, an e-mail address or a similar

[3] Or a version of ElGamal that is IND-CCA secure in the random oracle model.

distinguished name. The identity may also contain auxiliary information like an expiration date which refers for example to the contract period of an employee or to the validity period of the certificate.

Certification Authority. The CA holds a public key pk_{CA} and a corresponding secret key sk_{CA}. We presume that the public key is authenticated and known to all parties, i.e., once it is published it cannot be changed by the adversary. This is usually accomplished by a hierarchical arrangement of CAs, each intermediate CA certifying the validity of the public key of its successor. Only the key of the root CA has to be authenticated by other means. Here we focus on the the simpler one-tier approach of having only one CA, i.e., our model can be viewed as a condensed hierarchy with our single CA as the root CA. We discuss the more general case of hierarchical CAs in [10].

Registration of Keys. Each user can register keys with the CA by running the registration protocol. The required validation of the user's identity ID is usually done before by the so-called registration authority (RA), which sometimes coincides with the CA. Checking the identity of the user wishing to register its public key is typically performed by the RA through personal identification and physical validation (e.g., with help of a passport or a driver license). Hence, this part is beyond our computational model and we simply assume that bindings between user identities and their public keys are authentic.

We do not assume the existence of private channels. We do, however, presume authenticated channels between the CA and the users, even though the user most likely does not have a certified signature key when the registration starts. Without this minimal assumption about authenticated communication achieving any reasonable security guarantee seems to be impossible. The assumption can be enforced by a variety of means that include for example having the certification authority confirm the registration of a key through regular mail, signed electronic mail (with the signature verification key included in pk_{CA}), legally binding documents, or simply meeting in person.

The registration protocol itself is defined very generically. In this process the user derives a public key pk which may be used for encryption or signature verification and a secret key sk for decrypting or signing. We do not specify how the keys are generated (i.e., picked by the user alone or generated jointly between the user and the CA), yet we postulate that the CA should not be able to learn the corresponding secret key of the user. This inevitably requires interaction between both parties. The user also obtains a certificate cert which, classically, is an *explicit certificate* of type X.509, including the CA's signature. But since we also use other approaches like *implicit certificates* cert should be rather thought of as an arbitrary, possibly empty string. We assume, however, that each pair (ID, pk), where pk is registered, is unique; this can be achieved as is done for X.509 certificates by issuing serial numbers or other auxiliary information.

Revocation. For simplicity, we do not introduce revocation techniques in our basic model. Due to the lack of space the discussion on how to augment our

definitions and schemes to address revocations is delegated to the full version of the paper [10].

3 Secure Encryption in Public-Key Infrastructures

Syntax of Certified Encryption Schemes. A *certified encryption scheme* is a tuple $\mathsf{CS} = (\mathcal{EG}, \mathcal{K}, (\mathcal{C}, \mathcal{U}), \mathcal{E}, \mathcal{D})$ of probabilistic polynomial-time algorithms:

- $\mathcal{EG}$ is a randomized *parameter-generation* algorithm. It takes input 1^k, where k is the security parameter, and outputs some global parameters I, available to all parties. For sake of readability we omit I from the input of the parties.
- $\mathcal{K}$ is a randomized *key-generation* algorithm. It takes input I, and outputs a pair $(pk_{\mathsf{CA}}, sk_{\mathsf{CA}})$ consisting of a public key and a matching secret key.
- $(\mathcal{C}, \mathcal{U})$ is a pair of interactive randomized algorithms forming the (two-party) *public-key registration protocol*. $\mathcal{C}$ takes input a secret key sk_{CA}. $\mathcal{U}$ takes input the identity ID of a user and the public key pk_{CA} corresponding to sk_{CA}. As result of the interaction, the output of $\mathcal{C}$ is $(\mathsf{ID}, pk, \mathsf{cert})$, where pk is a public key and cert is an issued certificate. The local output of $\mathcal{U}$ is $(\mathsf{ID}, pk, sk, \mathsf{cert})$, where sk is a secret key that user ID uses to decrypt ciphertexts. We write $((\mathsf{ID}, pk, \mathsf{cert}), (\mathsf{ID}, pk, sk, \mathsf{cert})) \xleftarrow{\$} (\mathcal{C}(sk_{\mathsf{CA}}), \mathcal{U}(\mathsf{ID}, pk_{\mathsf{CA}}))$ for the result of this interaction. Either party can quit the execution prematurely, in which case the output of the party is set to $\perp$.
- $\mathcal{E}$ is a randomized *encryption* algorithm that takes input a user's identity ID, a public encryption key pk, a certificate cert, the authority's public key pk_{CA}, and a message $M \in \mathrm{MsgSp}(I)$, and outputs a ciphertext $C \in \{0,1\}^* \cup \{\perp\}$.
- $\mathcal{D}$ is a deterministic *decryption* algorithm which takes input a user's identity ID, a secret decryption key sk, a certificate cert, the authority's public key pk_{CA}, and a ciphertext C, and outputs $M \in \mathrm{MsgSp}(I) \cup \{\perp\}$. If $M = \perp$ we say that the ciphertext C is invalid (relative to $\mathsf{ID}, sk, \mathsf{cert}, pk_{\mathsf{CA}}$).

The scheme is *correct* iff for any parameters I, any pk_{CA}, any message $M \in \mathrm{MsgSp}(I)$, any user ID, and any $((\mathsf{ID}, pk, \mathsf{cert}), (\mathsf{ID}, pk, sk, \mathsf{cert})) \xleftarrow{\$} (\mathsf{CA}(sk_{\mathsf{CA}}), \mathcal{U}(\mathsf{ID}, pk_{\mathsf{CA}}))$, and any $C \xleftarrow{\$} \mathcal{E}(\mathsf{ID}, pk, \mathsf{cert}, pk_{\mathsf{CA}}, M)]$, it holds that $\mathcal{D}(\mathsf{ID}, sk, pk_{\mathsf{CA}}, \mathsf{cert}, C) = M$.

Remark 1. Our syntax does not explicitly deal with verifying the certificates, even though this may be necessary for security of the scheme. We assume that the constructions include such checks as part of their encryption algorithms.

Remark 2. The certificateless encryption schemes of [21, 2] are special cases of certified encryption schemes where the certificate cert is empty.

Security of Certified Encryption Schemes. We start with an informal discussion of the more interesting aspects of our model for secure certified encryption, and motivate some of the design choices that we made.

We envision a powerful adversary that is allowed to even corrupt the CA (i.e. learn its secret key and act on its behalf). At a superficial glance it may seem

that no security requirements would make sense in this case since under these circumstances the adversary could create new keys with valid certificates on behalf of honest users, and then decrypt any ciphertext created with these keys. We wish however to ensure that even if the CA is corrupt, the communication encrypted with keys truly registered by honest users is still protected. At least that requires the CA not to have users' secret keys. This requirement is somewhat akin to forward security. Without loss of generality, we treat the case when the corruption of the CA is static, i.e., the adversary decides at the beginning of its execution whether to control the CA or not. Indeed, we are able to show that our definition is equivalent (up to a constant factor in the security statement) to the analogous definition where the adversary can corrupt the CA at any point (see [10]).

Naturally the fundamental security requirement for certified encryption is privacy of encrypted data. However, as discussed in the introduction, we take into account potential threats arising from the use of the registration protocol. In particular, we require that an adversary cannot pass as genuine (registered) an unregistered key upon an honest user, in a way that allows the adversary to recover messages encrypted with this key. In other words, encryptions with unregistered keys can not be decrypted by the adversary.

Our model uses the standard definitional idea of indistinguishabiliy [22] captured via left-right encryption oracles [5]. The left-right encryption oracle is initialized with a secret bit b and encrypts either the left message M_0 or the right message M_1 of the two messages submitted by the adversary. The oracle is universal in the sense that the adversary can query it about any party ID and for any (not necessarily valid) key/certificate pair pk, cert. We restrict the kind of queries that are allowed in order to exclude trivial attacks. We demand that either (1) user ID is honest and $(\mathsf{ID}, pk, \mathsf{cert})$ has been registered before with the CA, or (2) $(\mathsf{ID}, pk, \mathsf{cert})$ is not registered but the CA is still honest.

The first condition covers the case of "standard" queries for proper keys of honest users, and encompasses the case when the CA might be corrupt. The second restriction prevents the adversary to register a key for some honest user (after corrupting the CA) and to determine the bit b easily. Also, if the CA is corrupt then the adversary can generate a certificate for any user locally, without invoking the registration protocol. This would also allow the adversary to create unregistered keys for which the oracle produces a valid ciphertext and which the adversary can still decrypt. Hence, we only permit queries where the key of the user has not been registered with the honest CA.

Finally, we emphasize that in our model we do not assume that the communication between the users and the CA is encrypted, i.e., we assume public channels. We therefore avoid the "chicken-and-egg"-like problem: how to assume secret transmissions if one is still trying to establish a public encryption key through this communication?

Definition 1. ***[Security of Certified Encryption Schemes]*** *Let* $\mathsf{CE} = (\mathcal{G}, \mathcal{K}, (\mathcal{C}, \mathcal{U}), \mathcal{E}, \mathcal{D})$ *be a certified encryption scheme. We associate to scheme* CE*, an adversary* $\mathcal{A}$*, and a bit* b *the experiments* $\mathbf{Exp}^{\text{cenc-ind-atk}}_{\mathsf{CE},\mathcal{A},b}(k)$ *for* $\text{atk} \in \{\text{cpa}, \text{cca}\}$.

In both experiments $\mathcal{A}$ is given as input $I \xleftarrow{\$} \mathcal{G}(1^k)$. The experiment maintains two virtual arrays RegListPub, RegListSec used to store public and secret information pertaining to users (respectively). We note that $\mathcal{A}$ knows the elements of RegListPub but not those of RegListSec. Also the adversary has access to all transcripts of the protocols executed during the experiment.

- Corruption of certification authority: *First, $\mathcal{A}$ decides if to corrupt the* CA*. If so, $\mathcal{A}$ chooses the key $\mathrm{pk}_{\mathsf{CA}}$ of the CA, else $\mathrm{pk}_{\mathsf{CA}}$ is generated via $(pk_{\mathsf{CA}}, sk_{\mathsf{CA}}) \xleftarrow{\$} \mathcal{K}(I)$ and given to $\mathcal{A}$.*
- Registering keys of users: *During the experiment, $\mathcal{A}$ can specify a user* ID *from the set of identities, to initiate a run of the public-key registration protocol with the honest or corrupt certification authority. If this is the first time the user* ID *is activated then $\mathcal{A}$ first decides whether to corrupt this user or not. In the execution with the CA we assume wlog. that at least one party is honest. At the end of the execution, when $\mathcal{C}$ outputs values $(\mathsf{ID}, pk, \mathsf{cert})$ and $\mathcal{U}$ outputs (possibly different) values $(\mathsf{ID}', pk', sk', \mathsf{cert}')$, we store $(\mathsf{ID}', pk', \mathsf{cert}')$ in RegListPub and $(\mathsf{ID}', pk', sk', \mathsf{cert}')$ in RegListSec if $\mathcal{U}$ is honest, or merely $(\mathsf{ID}, pk, \mathsf{cert})$ in RegListPub if only $\mathcal{C}$ is honest. If one of the parties is dishonest or stops prematurely then $\perp$ is stored in the corresponding array. Notice that all steps in the experiment, including steps of this interactive protocol may be arbitrarily interleaved.*
- Encryption queries: *$\mathcal{A}$ can query $\mathcal{UE}_{\mathsf{CE}}(b, \mathrm{pk}_{\mathsf{CA}}, \cdot, \cdot, \cdot)$, a universal left-right encryption oracle. It takes as input a tuple $(\mathsf{ID}, pk, \mathsf{cert})$ and two messages $M_0, M_1 \in \mathrm{MsgSp}(I)$ of equal length and returns a ciphertext $C \xleftarrow{\$} \mathcal{E}(\mathsf{ID}, pk, \mathsf{cert}, pk_{\mathsf{CA}}, M_b)$. We impose the restriction that user* ID *is honest and at this point $(\mathsf{ID}, pk, \mathsf{cert})$ is listed in RegListPub, or that the certification authority is still honest but $(\mathsf{ID}, pk, \mathsf{cert})$ does not appear in RegListPub at this point.*
- Decryption queries: *In experiment $\mathbf{Exp}^{\text{cenc-ind-cca}}_{\mathsf{CE},\mathcal{A},b}(k)$ the adversary is also given access to a universal decryption oracle $\mathcal{UD}_{\mathsf{CE}}(pk_{\mathsf{CA}}, \cdots)$ which has access to the array RegListSec. The queries to the oracle are tuples $(\mathsf{ID}, pk, \mathsf{cert}, C)$ where we require that C has not been previously returned by oracle $\mathcal{UE}_{\mathsf{CE}}(b, pk_{\mathsf{CA}}, \cdots)$ as answer to some query $((\mathsf{ID}, pk, \mathsf{cert}), M_0, M_1)$. If $(\mathsf{ID}, pk, sk, \mathsf{cert})$ occurs in RegListSec the oracle returns $\mathcal{D}(\mathsf{ID}, sk, \mathsf{cert}, pk_{\mathsf{CA}}, C)$; otherwise, it returns $\perp$.*

The adversary eventually stops and outputs a guess bit d which is also considered to be the output of the experiment. For $\mathrm{atk} \in \{\mathrm{cpa}, \mathrm{cca}\}$ the adversary's advantages in attacking the scheme are defined as follows.

$$\mathbf{Adv}^{\text{cenc-ind-atk}}_{\mathsf{CE},\mathcal{A}}(k) = \Pr[\mathbf{Exp}^{\text{cenc-ind-atk}}_{\mathsf{CE},\mathcal{A},1}(k) = 1] - \Pr[\mathbf{Exp}^{\text{cenc-ind-atk}}_{\mathsf{CE},\mathcal{A},0}(k) = 1] \ .$$

CE *is said to be IND-CPA (resp. IND-CCA) secure if the corresponding advantage of any $\mathrm{poly}(k)$-time adversary $\mathcal{A}$ is negligible.* ▮

"Traditional" Certified Encryption Schemes. We confirm that the classical approach of using signature-based certificates for the encryption scheme yields a secure certified encryption scheme. We show this to be the case even

when during a public key registration a user does not prove that it knows the corresponding secret key. The schemes that we use in our construction satisfy standard security notions (see the full version [10] for precise definitions of syntax and security). We now give the scheme (Construction 31) and state our security result (Theorem 1). The proof is in the full version [10].

Construction 31 [**Traditional Certified Encryption Scheme**] Let $\mathsf{DS} = (\mathcal{SG}_s, \mathcal{SK}_s, \mathcal{S}_s, \mathcal{V}_s)$ be a digital signature scheme, and $\mathsf{AE} = (\mathcal{EG}_e, \mathcal{EK}_e, \mathcal{E}_e, \mathcal{D}_e)$ be an asymmetric encryption scheme. Define $\mathsf{TCE} = (\mathcal{EG}, \mathcal{K}, (\mathcal{C}, \mathcal{U}), \mathcal{E}, \mathcal{D})$:

- *Parameter generation:* Algorithm $\mathcal{EG}(1^k)$ executes $I_s \xleftarrow{\$} \mathcal{SG}_s(1^k)$, $I_e \xleftarrow{\$} \mathcal{EG}_e(1^k)$ and outputs $I = (I_s, I_e)$.
- *Key generation:* Algorithm $\mathcal{K}$ generates a key pair $(pk_{\mathsf{CA}}, sk_{\mathsf{CA}}) \xleftarrow{\$} \mathcal{SK}_s(I_s)$.
- *Registration:* In order to register a key user, ID first generates a key pair $(pk, sk) \xleftarrow{\$} \mathcal{EK}_e(I_e)$ and sends (ID, pk) to $\mathcal{C}$ who computes $s \xleftarrow{\$} \mathcal{S}_s(sk_{\mathsf{CA}}, \mathsf{ID}||pk)$ and outputs (ID, pk, s). The user sets $\mathsf{cert} = s$ and outputs $(\mathsf{ID}, pk, sk, \mathsf{cert})$.
- *Encryption:* To encrypt a message M under identity ID, public key pk, certificate cert and key pk_{CA} the encryption algorithm $\mathcal{E}$ first verifies with $\mathcal{V}_s$ that cert is a valid signature for $\mathsf{ID}||pk$ under key pk_{CA}. If not then return $\perp$. Else compute $C \xleftarrow{\$} \mathcal{E}_e(pk, M)$ and return C.
- *Decryption:* To decrypt a ciphertext C with (ID, sk, cert) and pk_{CA} run algorithm $\mathcal{D}_e(sk, C)$ and return the answer. ▮

Theorem 1. *Let* DS *be a secure signature scheme and let* AE *be an IND-CPA secure (resp. IND-CCA secure) encryption scheme. Then the certified encryption scheme in Construction 31 is IND-CPA secure (resp. IND-CCA secure).* ▮

The proof idea is as follows. We turn a successful adversary $\mathcal{A}$ on the certified encryption scheme into an adversary $\mathcal{B}_{\mathsf{AE}}$ on the underlying encryption scheme. This algorithm $\mathcal{B}_{\mathsf{AE}}$ tries to guess in advance which of the registered keys adversary $\mathcal{A}$ will use to break the security of the certified scheme. This simulation works as long as adversary $\mathcal{A}$ does not use an unregistered but valid key, in which case we derive a successful attack on the signature scheme used in the certification procedure.

Efficient Certified Encryption Scheme. In the sequel we present our ElGamal-based encryption scheme with implicit certificates. We show that if the computational Diffie-Hellman problem is hard (see [10] for a precise statement of this assumption), our scheme guarantees IND-CCA security. At the same time, the efficiency of our scheme is close to that of the basic ElGamal encryption *without* certificate verifications. The security of the following construction is captured by Theorem 2. Its security is also provided in [10].

The idea of our scheme is to let the CA issue certificates in forms of Schnorr signatures for identity ID and to use these values for a CCA2-version of the ElGamal encryption. That is, for the CA's public key $pk_{\mathsf{CA}} = Z = g^z$ the CA hands the user the values $R = g^r$ and $\log_g RZ^c$ for $c = H(R, \mathsf{ID})$. To send the user encrypted messages one uses the value RZ^c as the public ElGamal key, and

the user can decrypt with his decryption key $sk = \log_g RZ^c$. Below we use a slightly different variant in which the user contributes to the Schnorr signature via a random value $S = g^s$, in order to deny the CA knowledge of the decryption key.

Construction 32 [Certified ElGamal Encryption] We construct the certified ElGamal encryption scheme $\mathsf{CE} = (\mathcal{EG}, \mathcal{EK}, (\mathcal{C}, \mathcal{U}), \mathcal{E}, \mathcal{D})$ as follows:

- *Parameter generation:* Algorithm $\mathcal{EG}$ on input 1^k generates a (description of a) group $\mathcal{G}$ of prime order $q = q(k)$, as well as a generator g of this group. Let $2^k \leq q < 2^{k+1}$. Algorithm $\mathcal{EG}$ also picks (descriptions of) hash functions $F = \{0,1\}^* \rightarrow \mathbf{Z}_q$, $G : \{0,1\}^* \rightarrow \{0,1\}^{t+k}$, $H : \{0,1\}^* \rightarrow \mathbf{Z}_q$. It returns $I = (\mathcal{G}, q, g, F, G, H)$. The associated message space is $\{0,1\}^t$. These parameters are given to all parties and algorithms as additional input.
- *Key generation:* Algorithm $\mathcal{EK}$ on input I selects $z \xleftarrow{\$} \mathbf{Z}_q$ and computes $Z = g^z$. It returns $(pk_{\mathsf{CA}}, sk_{\mathsf{CA}}) = (Z, (Z, z))$.
- *Key registration:* The pair $(\mathcal{C}, \mathcal{U})$ of interactive algorithms is defined by the following steps. $\mathcal{C}$ gets as input $sk_{\mathsf{CA}} = (Z, z)$, while $\mathcal{U}$ gets some identity ID and $pk_{\mathsf{CA}} = Z$. The authority $\mathcal{C}$ first picks $r \xleftarrow{\$} \mathbf{Z}_q$, computes $R = g^r$ and and sends R to $\mathcal{U}$. User $\mathcal{U}$ chooses $s \xleftarrow{\$} \mathbf{Z}_q$, computes $S = g^s$ and sends (S, ID) back to $\mathcal{C}$. Upon receiving (S, ID) algorithm $\mathcal{C}$ sets $c = H(R, S, \mathsf{ID})$ and $y = r + cz \bmod q$. Let $pk = (R, S)$ and $\mathsf{cert} = \varepsilon$ be empty. $\mathcal{C}$ returns (R, y) to $\mathcal{U}$ and outputs $(\mathsf{ID}, pk, \mathsf{cert})$. $\mathcal{U}$ verifies that $g^y = RZ^c$ for $c = H(R, S, \mathsf{ID})$, computes $sk = s + y \bmod q$ and outputs $(\mathsf{ID}, pk, sk, \mathsf{cert})$. Note that $sk = \log_g RSZ^c$.
- *Encryption:* For input $\mathsf{ID}, pk = (R, S), \mathsf{cert} = \varepsilon, pk_{\mathsf{CA}} = Z$ and message $M \in \{0,1\}^t$ the encryption algorithm picks $\alpha \xleftarrow{\$} \{0,1\}^k$, computes $a = F(\mathsf{ID}, pk, \mathsf{cert}, \alpha||M)$, $A = g^a$ and $B = G(\mathsf{ID}, pk, \mathsf{cert}, (RSZ^c)^a) \oplus \alpha||M$ where $c = H(R, S, \mathsf{ID})$. It outputs $C = (A, B)$.
- *Decryption:* For input $\mathsf{ID}, sk, \mathsf{cert} = \varepsilon, pk_{\mathsf{CA}} = Z$ and $C = (A, B)$ the decryption algorithm computes $\alpha||M = B \oplus G(\mathsf{ID}, pk, \mathsf{cert}, A^{sk})$ and verifies that $A = g^{F(\mathsf{ID}, pk, \mathsf{cert}, \alpha||M)}$. In this case it returns M, else it returns $\perp$. ▌

Theorem 2. *Suppose that the parameter generator $\mathcal{EG}$ in the encryption scheme in Construction 32 generates CDH-secure groups, and that F, G, H are modeled as random oracles. Then the scheme CE in Construction 32 is IND-CCA secure in the random oracle model.* ▌

The efficiency of our scheme is comparable to the one of regular ElGamal encryption *without* certificate verification. With the square-and-multiply exponentiation method, basic ElGamal encryption without certification needs $3k$ expected multiplications, our scheme based on implicit certificates requires $3.25k$ multiplications on the average, whereas regular ElGamal encryption with explicit Schnorr signature certificates would require $4.75k$ expected modular multiplications.

4 Secure Signatures in Public-Key Infrastructures

Syntax of Certified-Signature Schemes. A *certified-signature scheme* is a tuple $\mathsf{CS} = (\mathcal{SG}, \mathcal{K}, (\mathcal{C}, \mathcal{U}), \mathcal{S}, \mathcal{V})$, where the constituent algorithms run in polynomial time and are defined as follows.

- Algorithms $\mathcal{SG}, \mathcal{K}$ and registration protocol $(\mathcal{C}, \mathcal{U})$ are as in the definition of certified encryption schemes (here, $\mathcal{SG}$ replaces $\mathcal{EG}$).
- $\mathcal{S}$ is a (possibly) randomized *signing* algorithm. It takes input an identity ID, a secret key sk, a certificate cert, the authority's public key pk_{CA} and a message $M \in \{0,1\}^*$, and outputs a signature σ.
- $\mathcal{V}$ is a deterministic *verification* algorithm. It takes input an identity ID, a public key pk, a certificate cert, a public key pk_{CA}, a message M and a signature σ, and outputs 0 or 1. In the latter case, we say that σ is a *valid* signature for M relative to $(\mathsf{ID}, pk, \mathsf{cert}, pk_{\mathsf{CA}})$.

We require that for all $M \in \{0,1\}^*$ and all users ID, if (pk, sk) is a key pair for user ID with cert, i.e., $((\mathsf{ID}, pk, \mathsf{cert}), (\mathsf{ID}, pk, sk, \mathsf{cert})) \xleftarrow{\$} (\mathcal{C}(sk_{\mathsf{CA}}), \mathcal{U}(\mathsf{ID}, pk_{\mathsf{CA}}))$ for $(pk_{\mathsf{CA}}, sk_{\mathsf{CA}})$ generated by $\mathcal{K}(I)$ and I output by $\mathcal{G}(1^k)$, then, for verification, $\mathcal{V}(\mathsf{ID}, pk, \mathsf{cert}, pk_{\mathsf{CA}}, M, \mathcal{S}(\mathsf{ID}, sk, \mathsf{cert}, pk_{\mathsf{CA}}, M)) = 1$.

Security of Certified-Signature Schemes. Our model is for the basic setting outlined in Section 2. Users register public-keys by interacting with a certification authority on public, authenticated channels. After registration, parties can sign messages using the secret keys associated to the public key the have registered. Signatures can then be verified, and we emphasize that the verification process involves both the public key of the CA and that of the user.

We consider again a powerful adversary whose capabilities combine the more standard chosen-message attacks with additional capabilities specific to our setting. The adversary attempts a forgery by outputting a user identity, a public key, a message and a signature. Roughly, the adversary wins if the signature is valid with respect to the chosen public key, and either (1) the honest user has registered the public key and has not priorly signed the message, (2) the public key has not been registered, or (3) the same public key has been registered by a different (honest) user. Condition (1) corresponds to the notion of existential unforgeability [23] for standard digital signature schemes, and we require that it holds even if the CA is corrupt. Condition (2) guarantees that signatures for keys that are not bound to identities of the users (i.e., "outside of the PKI") are not valid. Condition (3) prevents attacks where for example a malicious user claims authorship of a message signed by another user.

Definition 2. [Security of Certified-Signature Schemes] *Let* $\mathsf{CS} = (\mathcal{SG}, \mathcal{K}, (\mathcal{C}, \mathcal{U}), \mathcal{S}, \mathcal{V})$ *be a certified-signature scheme. We associate to scheme* CS*, an adversary* $\mathcal{A}$*, and security parameter* k *an experiment* $\mathbf{Exp}^{\text{cs-uf}}_{\mathsf{CS},\mathcal{A}}(k)$*. The experiment maintains arrays RegListPub, RegListSec which are as in the experiments defining security for certified encryption schemes. In the beginning of the experiment, public parameters are generated via* $I \xleftarrow{\$} \mathcal{G}(1^k)$ *and are given as input to the adversary, and then,* $\mathcal{A}$ *can make the following requests or queries:*

- Corruption of certification authority: *This stage is as in the experiment defining the security of certified encryption.*
- Registering keys of users: *This is handled as in the model for defining security of certified encryption.*
- Signature queries: $\mathcal{A}$ *can make signature requests to a universal signing oracle* $\mathcal{US}_{\mathsf{CS}}^{pk_{\mathsf{CA}}}$*: on a query* $(\mathsf{ID}, pk, \mathsf{cert}, M)$ *the oracle verifies that user* ID *is honest, and if so it looks up the corresponding entry* $(\mathsf{ID}, pk, sk, \mathsf{cert})$ *in RegListSec and returns to* $\mathcal{A}$ *a signature* $\mathcal{S}(\mathsf{ID}, sk, \mathsf{cert}, pk_{\mathsf{CA}}, M)$*. Otherwise, the answer of the oracle is* $\perp$*.*

Eventually, $\mathcal{A}$ *stops and outputs an attempted forgery* $(\mathsf{ID}, pk, \mathsf{cert}, M, \sigma)$*. The experiment returns* 1 *if* $\mathcal{V}(pk_{\mathsf{CA}}, \mathsf{ID}, pk, \mathsf{cert}, M, \sigma) = 1$ *and the following conditions are satisfied (otherwise it returns* 0*):*

1. ID *is honest, and no valid signing query* $(\mathsf{ID}, pk, \mathsf{cert}', M)$ *was made for any* cert'*, or*
2. CA *is honest and* $(\mathsf{ID}, pk, \mathsf{cert}') \notin$ *RegListPub, for any* cert' *(i.e. the user* ID *never registered the key* pk*), or*
3. CA *is honest and* $(\mathsf{ID}', pk, \mathsf{cert}') \in$ *RegListPub for some honest user* $\mathsf{ID}' \neq \mathsf{ID}$ *(i.e. some honest user registered* pk*),*

We define the advantage of adversary $\mathcal{A}$ *as*

$$\mathbf{Adv}^{\text{cs-uf}}_{\mathsf{CS},\mathcal{A}}(k) \;=\; \Pr\left[\,\mathbf{Exp}^{\text{cs-uf}}_{\mathsf{CS},\mathcal{A}}(k) = 1\,\right].$$

We say that CS *is a secure certified-signature scheme if the function* $\mathbf{Adv}^{\text{cs-uf}}_{\mathsf{CS},\mathcal{A}}(\cdot)$ *is negligible for all poly(*kl*-time adversaries* $\mathcal{A}$*.* ▌

"Traditional" certified signature schemes. Here we analyze the traditional approach to certified signatures, where the public-keys of users are certified by the certification authority using a a digital signature scheme. In turn, users produce signatures by using the secret keys associated with their certificated public-keys. Signature verification consist in verifying the signature of the user and the validity of the certificates for the users' public-keys. An interesting aspect that we clarify is that proofs of knowledge of the secret key associated to the public key of the user are not necessary to ensure security of the scheme. We show that simply signing a designated message in a proof of possession is sufficient for security. We now give the scheme (Construction 41) and state our security result (Theorem 3). The proof is in [10], along with the concrete security result.

Construction 41 [**Traditional Certified Signature Scheme**] Let $\mathsf{DS} = (\mathcal{SG}, \mathcal{SK}, \mathcal{S}_1, \mathcal{V}_1)$ be a digital signature scheme[4]. The first two algorithms of a certified-signature scheme $\mathsf{TCS} = (\mathcal{G}, \mathcal{K}, (\mathcal{C}, \mathcal{U}), \mathcal{S}, \mathcal{V})$ are those of DS, and the rest of polynomial time algorithms are defined as follows.

[4] For simplicity we consider a case when the certification authority and a user use a single signature scheme. The definition and other results can be easily modified to accommodate a case when different signatures are used by the parties.

- *Parameter and key generation:* $\mathcal{G} \equiv \mathcal{SG}, \mathcal{K} \equiv \mathcal{SK}$.
- *Registration:* To register pk, a user ID sends pk to the CA. CA sends to the user a random "challenge" message[5] $M' \xleftarrow{\$} \{0,1\}^k$. The user computes $\sigma' \xleftarrow{\$} \mathcal{S}(sk,\, 0||M')$ and sends it to CA. If $\mathcal{V}(pk, 0||M', \sigma') = 1$ then CA computes $\mathsf{cert} \xleftarrow{\$} \mathcal{S}(sk_{\mathsf{CA}}, (\mathsf{ID}, pk))$, sends cert to the user and outputs $(\mathsf{ID}, pk, \mathsf{cert})$. The user outputs $(\mathsf{ID}, pk, sk, \mathsf{cert})$.
- *Signing:* $\mathcal{S}$ on input $(\mathsf{ID}, sk, \mathsf{cert}, pk_{\mathsf{CA}}, M)$ outputs $\sigma \xleftarrow{\$} \mathcal{S}_1(sk, 1||M)$.
- *Verification:* $\mathcal{V}$ takes $(\mathsf{ID}, pk, \mathsf{cert}, pk_{\mathsf{CA}}, M, \sigma)$. It outputs 1 iff $\mathcal{V}_1(pk_{\mathsf{CA}}, (\mathsf{ID}, pk), \mathsf{cert}) = 1$ and $\mathcal{V}_1(pk, 1||M, \sigma) = 1$. ▮

Theorem 3. *Let* $\mathsf{DS} = (\mathcal{SG}, \mathcal{SK}, \mathcal{S}, \mathcal{V})$ *be a digital signature scheme. Then if* DS *is secure (existentially unforgeable under chosen-message attack), then* TCS *is a secure certified signature scheme.* ▮

The proof idea is to transform an attacker against the certified signature scheme into one against the underlying signature scheme (by guessing the right target key in advance). It is not hard to see that each successful attack on the certified scheme (new signatures under keys of honest users, generating an unregistered but valid key, and registering keys of honest users under different names) immediately yields a forgery for the signature scheme.

Efficient certified signature schemes. Here we give a construction of an efficient, provably secure certified signature scheme based on Schnorr signatures. Its security, captured by Theorem 4, is based on the discrete logarithm assumption (a precise definition is given in [10]). The idea is similar to the encryption case, where the CA issued Schnorr signatures to be used as the secret and public ElGamal keys by users, only this time we let the users deploy the key pairs for Schnorr signatures themselves.

Construction 42 [**Schnorr-based Certified Signature Scheme**] We define scheme $\mathsf{CS} = (\mathcal{SG}, \mathcal{K}, (\mathcal{C}, \mathcal{U}), \mathcal{S}, \mathcal{V})$ by the algorithms:

- *Parameter generation:* Algorithm $\mathcal{SG}$ on input 1^k generates a (description of a) group $\mathcal{G}$ of prime order $q = q(k)$, as well as a generator g of this group. Let $2^k \leq q < 2^{k+1}$. Algorithm $\mathcal{SG}$ also picks (descriptions of) hash functions $G : \{0,1\}^* \to \mathbf{Z}_q$, $H : \{0,1\}^* \to \mathbf{Z}_q$. It returns $I = (\mathcal{G}, q, g, G, H)$. These parameters are given to all parties and algorithms as additional input.
- *Key generation:* Algorithm $\mathcal{EK}$ on input I selects $z \xleftarrow{\$} \mathbf{Z}_q$ and computes $Z = g^z$. It returns $(pk_{\mathsf{CA}}, sk_{\mathsf{CA}}) = (Z, (Z, z))$.
- *Key registration:* The pair $(\mathcal{C}, \mathcal{U})$ of interactive algorithms is defined by the following steps. $\mathcal{C}$ gets as input $sk_{\mathsf{CA}} = (Z, z)$, while $\mathcal{U}$ gets some identity ID and $pk_{\mathsf{CA}} = Z$. The authority $\mathcal{C}$ first picks $r \xleftarrow{\$} \mathbf{Z}_q$, computes $R = g^r$ and sends R to $\mathcal{U}$. User $\mathcal{U}$ chooses $s \xleftarrow{\$} \mathbf{Z}_q$, computes $S = g^s$ and sends

[5] We need that all challenge messages be different with overwhelming probability. An alternative approach would be to include a current date and time in the challenge message.

(S, ID) back to $\mathcal{C}$. Upon receiving (S, ID) algorithm $\mathcal{C}$ sets $c = H(R, S, \mathsf{ID})$ and $y = r + cz \bmod q$. Let $pk = (R, S)$ and $\mathsf{cert} = \varepsilon$. $\mathcal{C}$ returns (R, y) to $\mathcal{U}$ and outputs $(\mathsf{ID}, pk, \mathsf{cert})$. $\mathcal{U}$ verifies that $g^y = RZ^c$ for $c = H(R, S, \mathsf{ID})$, computes $sk = s + y \bmod q$ and outputs $(\mathsf{ID}, pk, sk, \mathsf{cert})$. Note that $sk = \log_g RSZ^c$.

- *Signing:* For input ID, sk, $\mathsf{cert} = \varepsilon$, (R, S), certificate ε and Z the signing algorithm picks $a \xleftarrow{\$} \mathbf{Z}_q$ and computes $A = g^a$ and $B = a + sk \cdot G(\mathsf{ID}, A, M)$. The signature is $\sigma = (A, B)$.
- *Verification:* For input ID, $pk = (R, S)$, ε, $pk_{\mathsf{CA}} = Z$, a message M and a signature $\sigma = (A, B)$ the verification algorithm outputs 1 if the equation $g^B = A(RSZ^c)^d$ holds, where $c = H(R, S, \mathsf{ID})$ and $d = G(\mathsf{ID}, A, M)$. Otherwise it outputs 0. ▮

Theorem 4. *The certified-signature scheme of Construction 42 is secure in the random oracle model if the parameter generation algorithm generates DL-secure groups.* ▮

For the above scheme, signing is exactly as in standard Schnorr signature schemes and thus as efficient. Verification of a signature, however, now requires on the average only $1.875k$ modular multiplications with the square-and-multiply method, as opposed to $3.5k$ modular multiplications as required to verify two separate Schnorr signatures.

5 Related Work

Here we review several PKI-related works in the literature and put our results in the context [21, 2, 15, 16, 28, 29, 41, 27, 34, 30, 40, 17, 7, 18, 42, 43, 25, 31].

Gentry [21], Al-Riyami and Paterson [2] and subsequent works [30, 40, 17, 7, 18, 3, 42, 43, 25, 31] recently proposed public-key encryption schemes that do not assume a standard PKI. Similarly to our efficient scheme, certificates are implicit, that is, a sender does not have to verify the certificate before sending an encrypted message, yet only the user who properly registered its public key is able to decrypt. The goals of their schemes and ours, however, differ. The motivation for the works of [21, 2, 30, 40, 17, 7, 18, 42, 43, 25, 31] is to overcome the main weakness of identity-based encryption (IBE) [39, 12], namely, the requirement that the trusted party called a private key generator (PKG) knows the secret keys of the users, while preserving the advantage of IBE of simplified management of expired and revoked public keys. Gentry also eliminates the requirement of a secure channel between a user and the PKG. On the other hand, the goal of our scheme is to achieve an efficiency improvement over "traditional" discrete-logarithm-based certified encryption schemes and, similar, for certified signature schemes. While our schemes require the key management support of a traditional PKI, they come with computational savings.

Security of implicit certificates in the context of digital signatures had been priorly investigated by Brown, Gallant, and Vanstone [13]. Their security model is only concerned with the certification process and does not consider the usage

of the resulting keys. However, for the particular application analyzed in the paper, the resulting security model (which is strictly weaker and less general than the one we introduce here) appears to suffice.

Canetti [15] recently presented a universally composable certification protocol, that uses traditional signature-based certificates. And while universal composition provides very strong security guarantees, his approach falls again short of investigating the combined certification and encryption or signature process, neither does his model take CA corruptions and the broader adversarial capabilities into account. In contrast to our more efficient solutions based on implicit certificates, alternatives to the traditional approach are not discussed in [15].

Stronger security requirements on signatures are imposed by the model suggested by Menezes and Smart [35] for the use of signatures in the "multi-user setting". Condition (3) of our definition of security for signature schemes is reminiscent of their security requirement. However, the framework proposed in [35] does not explicitly consider the registration protocol.

Since our basic model is concerned with security under one-level of certification, some of the issues specific to hierarchical PKIs do not show up explicitly. When tackling such settings (which naturally arise in practice, e.g. when using PGP [44]), special attention needs to be payed to the trust that parties put in certificate chains, given that one or several of the CAs could have been corrupted. Prior research that could prove useful in extending our framework to these settings include various models for trust in key authenticity [33, 14, 26], quantitative trust evaluation [8], as well as various defenses against multiple CA corruption (e.g. multi-certificate chains [37]).

Our efficient certified encryption scheme resembles the PKI-enabled CA-Oblivious encryption scheme independently proposed by Castelluccia et al. in their recent work [16] as a building block for a secret handshake protocol. The authors analyze their schemes with respect to a significantly weaker security notion, namely one-wayness. Moreover, in their scheme the CA knows the secret keys of the users and is trusted to behave honestly. In their model, it is also assumed that the CA is trusted not to use the knowledge of the secret keys. Moreover, no standard outsider attacks are considered, that is a scheme where an adversary can decrypt messages addressed to the registered users can be proven secure! The authors prove that their scheme satisfies their security notion in the RO model. The authors suggest how to modify the scheme to allow the CA to be less trusted, but the security of the resulting scheme is unclear. It is also suggested in [16] that the scheme can be made IND-CCA secure using the Fujisaki-Okamoto transform [20]. We note, however, that it is not immediate without a new proof that this would work, since the Fujisaki-Okamoto transform can be provably applied to basic encryption schemes that assume a standard PKI with explicit certificates. The primitive of [16] is different; it employs implicit certificates. Therefore, a new security definition and a proof will be necessary to validate this suggestion. On the other hand, our scheme provably satisfies a very strong security definition discussed above, where the CA is only trusted not to register new keys for users without their permission. Our results also show that

the Fujisaki-Okamoto transform is not needed as our scheme is very simple and yet IND-CCA secure.

Our efficient certified signature scheme resembles the proxy signature scheme of Kim, Park and Won [28] and the self-certified signature scheme of Lee and Kim [29]. Unlike our scheme, the proxy signature scheme assumes a PKI where each user already holds a public key and a digital certificate. Neither of these papers provides formal security definitions and analyses. A modification of the proxy signature scheme of [28] has been proven secure in [11], but their proof is for a primitive that differs from ours in that it assumes a PKI, explicit certificates, and involves a different security notion.

6 Acknowledgments

Part of the work was done while the authors were at UCSD. Alexandra Boldyreva is supported in part by NSF CAREER award 0545659. Marc Fischlin is supported by the Emmy Noether Program Fi 940/2-1 of the German Research Foundation (DFG). Bogdan Warinschi is supported in part by the ACI Jeunes Chercheurs JC 9005.

References

1. C. Adams and S. Farrell. Internet x.509 public key infrastructure: Certificate management protocols. Work in progress, 2004.
2. S. Al-Riyami and K. G. Paterson. Certificateless public key cryptography. In *ASIACRYPT*, volume 2894 of *LNCS*, pages 452–473. Springer-Verlag, 2003.
3. J. Baek, R. Safavi-Naini, and W. Susilo. Certificateless public key encryption without pairing. In *Information Security (ISC)*, volume 3650 of *LNCS*, pages 134–148. Springer-Verlag, 2005.
4. M. Bellare, A. Boldyreva, and J. Staddon. Multi-recipient encryption schemes: Security notions and randomness re-use. In *Public-Key Cryptography (PKC)*, volume 2567. Springer-Verlag, 2003.
5. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *FOCS '97: Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97)*, page 394, Washington, DC, USA, 1997. IEEE Computer Society.
6. M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Conference on Computer and Communications Security (CCS)*. ACM, 1993.
7. K. Bentahar, P. Farshim, J. Malone-Lee, and N.P. Smart. Generic constructions of identity-based and certificateless kems. *Cryptology ePrint Archive, Report 2005/058.*, 2005.
8. T. Beth, M. Borcherding, and B. Klein. Valuation of trust in open networks. In *Computer Security—ESORICS 94 (Lecture Notes in Computer Science 875)*, pages 3–18. Springer-Verlag, 1994.
9. A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the Gap-Diffie-Hellman-group signature scheme. In *Public Key Cryptography (PKC) '03*, pages 31–46, 2003.

10. A. Boldyreva, M. Fischlin, A. Palacio, and B. Warinschi. A closer look at PKI: Security and efficiency. *A full version of this paper. Available at* `http://www-static.cc.gatech.edu/~aboldyre/publications.html`, 2007.
11. A. Boldyreva, A. Palacio, and B. Warinschi. Secure proxy signaure schemes for delegation of signing rights. *Cryptology ePrint Archive, Report 2003/096.*, 2003.
12. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In *CRYPTO*, volume 2139 of *LNCS*. Springer-Verlag, 2001.
13. D.R. L. Brown, R. P. Gallant, and S. A. Vanstone. Provably secure implicit certificate schemes. In *Conference on Financial Cryptography '01*, pages 156–165. Springer-Verlag, 2002.
14. M. Burmester, Y. Desmedt, and G. Kabatianskii. Trust and security: A new look at the Byzantine generals problem. In R. N. Wright and P. G. Neumann, editors, *Network Threats, DIMACS, Series in Discrete Mathematics and Theoretical Computer Science, December 2–4, 1996, vol. 38*. AMS, 1998.
15. R. Canetti. Universally composable signature, certification, and authentication. In *CSFW*. IEEE Computer Society, 2004.
16. C. Castelluccia, S. Jarecki, and G. Tsudik. Secret handshakes from CA-oblivious encryption. In *ASIACRYPT*, volume 3329 of *LNCS*, pages 293–307. Springer-Verlag, 2004.
17. Z. Cheng and R. Comley. Efficient certificateless public key encryption. *Cryptology ePrint Archive, Report 2005/012.*, 2005.
18. A. W. Dent and C. Kudla. On proofs of security for certificateless cryptosystems. *Cryptology ePrint Archive, Report 2005/348.*, 2005.
19. T. ElGamal. A public key cryptosystem and signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, Vol. 31, 1985.
20. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO*, volume 1666 of *LNCS*, pages 537–554, 1999.
21. C. Gentry. Certificate-based encryption and the certificate revocation problem. In *EUROCRYPT*, volume 2656 of *LNCS*, pages 272–293. Springer-Verlag, 2003.
22. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Science*, 1984.
23. S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, 17(2):281–308, April 1988.
24. J. Herzog, M. Liskov, and S. Micali. Plaintext awareness via key registration. In *CRYPTO*, volume 2729. Springer-Verlag, 2003.
25. B. Hu, D. Wong, Z. Zhang, and X. Deng. Key replacement attack against a generic construction of certificateless signature. In *ACISP*, volume 4058 of *Lecture Notes in Computer Science*, pages 235–246. Springer-Verlag, 2006.
26. A. Josang. Trust-based decision making for electronic transactions. In *Fourth Nordic Workshop on Secure IT Systems (NORDSEC'99)*, pages 99–105, 1999.
27. B. Kaliski. An unknown key-share attack on the mqv key agreement protocol. *ACM Transactions on Information and System Security (TISSEC)*, 4(3):275–288, 2001.
28. S. Kim, S. Park, and D. Won. Proxy signatures, revisited. In *ICICS*. Springer-Verlag, 1997.
29. B. Lee and K. Kim. Self-certified signatures. In *Indocrypt*, volume 2551 of *LNCS*. Springer-Verlag, 2002.
30. Y.-R. Lee and H.-S. Lee. An authenticated certificateless public key encryption scheme. *Cryptology ePrint Archive, Report 2004/150.*, 2004.

31. . Libert and J.-J. Quisquater. On Constructing Certificateless Cryptosystems from Identity Based Encryption. In *Public Key Cryptography (PKC)*, LNCS. Springer-Verlag, 2006.
32. A. Lysyanskaya, R. L. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. In *Selected Areas in Cryptography (SAC)*, pages 184–199. Springer-Verlag, 1999.
33. U. Maurer. Modeling public-key infrastructure. In *Computer Security—ESORICS 96 (Lecture Notes in Computer Science 1146)*, pages 325–350. Springer-Verlag, 1996.
34. A. Menezes, M. Qu, and S. A. Vanstone. Some new key agreement protocols providing mutual implicit authentication. In *Selected Areas in Cryptography (SAC)*, 1995.
35. A. Menezes and N. Smart. Security of signature schemes in a multi-user setting. *Designs, Codes and Cryptography*, 33:261–274, 2004.
36. S. Micali, K. Ohta, and L. Reyzin. Accountable-subgroup multisignatures. In *Conference on Computer and Communications Security (CCS)*, pages 245–254. ACM, 2001.
37. M. K. Reiter and S. G. Stubblebine. Path independence for authentication in large scale systems. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 57–66, April 1997. Zurich.
38. C. P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.
39. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*. Springer-Verlag, 1984.
40. Y. Shi and J. Li. Provable efficient certificateless public key encryption. *Cryptology ePrint Archive, Report 2005/287.*, 2005.
41. V. Shoup. On formal models for secure key exchange. *IBM Research Report RZ 3120*, 1999.
42. D. H. Yum and P. J. Lee. Generic construction of certificateless encryption. In *EuroPKI*, volume 3043 of *Lecture Notes in Computer Science*, pages 802–811. Springer-Verlag, 2004.
43. D. H. Yum and P. J. Lee. Generic construction of certificateless signature. In *ACISP*, volume 3108 of *Lecture Notes in Computer Science*, pages 200–211. Springer-Verlag, 2004.
44. P. R. Zimmermann. *The Official PGP User's Guide*. MIT Press, Cambridge, Massachussets, 1995.