Information Sciences 463–464 (2018) 86–91

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Notes on a provably-secure certificate-based encryption against malicious CA attacks

Wenjie Yang, Jian Weng*, Anjia Yang, Congge Xie, Yaxi Yang

College of Cyber Security/College of Information Science and Technology, Jinan University, Guangzhou 510632, China

ARTICLE INFO

Article history:
Received 11 December 2017
Revised 16 June 2018
Accepted 20 June 2018
Available online 20 June 2018

Keywords:
Certificate-based encryption
Certification authority attack
Chosen ciphertext attack
Malicious CA attack
Bilinear pairings
Standard model

ABSTRACT

Certificate-based encryption (CBE) is a very useful cryptographic primitive which not only simplifies the certificate management in traditional public-key encryption, but also solves the key escrow problem inherent in identity-based encryption. How to construct a provably-secure CBE scheme without using random oracles has been attracting the attentions of the research community. Recently, Lu et al. introduced a CBE scheme and claimed that their scheme is secure against adaptive chosen ciphertext attacks even considering a malicious certification authority (CA). In this paper, we demonstrate that a chosen ciphertext attacker can easily distinguish the challenge ciphertext generated by the challenger according to their security model. Further, the CA can trivially decrypt any entity's ciphertext without knowing the entity's secret key. In addition, we also point out that their security proof has some flaws and give a new CBE scheme secure against malicious CA attacks in the standard model.

1. Introduction

In Eurocrypt 2003, Gentry proposed the concept of certificate-based encryption (CBE) and formalized its security model [9]. In CBE, the decryption requires both a secret key that is set by the corresponding public key owner and a current certificate that is produced by the certification authority (CA). Meanwhile, a sender neither verifies the certificate binding a receiver with its public key like in traditional public key settings [3–5,10], nor worries that the CA decrypts any ciphertext like private key generator in the identity-based settings [6,13,14].

Specifically, CBE can be used to build efficient public key infrastructures requiring fewer infrastructures than the conventional ones. Furthermore, there are no certificate revocation problem because the receiver cannot decrypt the received ciphertexts without the corresponding up-to-date certificate or key distribution problem beacuse the certificates need not be kept secret and can be sent to the users via public communication channels. Based on these advantages, CBE has received wide attention from both academia and industry [8,11,12,17].

Unfortunately, Gentry's security model implies a precondition that the CA is always honest before initializing system parameters. The condition restricts the CA's ability of launching attacks, which is very unreasonable. To match real case, Wu et al. [20] improved the original security model and depicted the malicious CA on the basis of [1,19]. The malicious CA can adaptively set some trapdoors at the very beginning of the setup phase to mount an attack more easily in practical

* Corresponding author.
E-mail addresses: njnuywj@163.com (W. Yang), cryptjweng@gmail.com (J. Weng), anjiayang@gmail.com (A. Yang), xiecongge14an@sina.com (C. Xie), jnuyaxiyang@foxmail.com (Y. Yang).

https://doi.org/10.1016/j.ins.2018.06.049
0020-0255/© 2018 Published by Elsevier Inc.

applications. Most of the early CBE schemes relying on Gentry's original security model [9] are insecure against the attacks from the malicious CA. In addition, the security analysis of most of these schemes are based on the random oracles [2], which may not be truly preserved when the random oracle is substituted for any real hash function. Therefore, the design of CBE schemes provably-secure against the malicious CA in the standard model becomes a major concern.

In 2016, Lu et al. [16] presented a concrete CBE scheme and claimed that their CBE scheme is not only secure against adaptive chosen ciphertext attacks even considering a malicious CA but also efficient and practical compared with the previous similar schemes [7,15,18,21]. However, in this paper, our analysis reveals that their scheme cannot withstand chosen ciphertext attacks or certification authority attacks. We give an attack to show that a chosen ciphertext attacker can easily distinguish the challenge ciphertext produced by the challenger in the challenge phase according to their security model. We also demonstrate that the CA can lightly decrypt any entity's ciphertext without knowing the entity's secret key. Finally, we illustrate the flaws in their security proof and design a new provably secure CBE scheme without using random oracles.

The paper is organized as follows: we firstly give some preliminaries in Section 2. Then, we review Lu et al.'s certificate-based encryption and do some cryptanalysis on it in Section 3. Next, we give a new provably-secure construction in Section 4. Finally, the conclusion is presented in Section 5.

2. Preliminaries

2.1. Bilinear groups

Here, we adopt the standard concept about bilinear pairings described in [16]. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of same prime order q and g be a generator of $\mathbb{G}$. The bilinear map $\hat{e}:\mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$ has the following properties:

- Bilinearity: $\forall g, h\in\mathbb{G}$ and $\forall a, b\in\mathbb{Z}_p^*$, we have $\hat{e}(g^a, h^b)=\hat{e}(g, h)^{ab}$;
- Non-degenracy: $\hat{e}(g, g)\neq 1_{\mathbb{G}_T}$ where $1_{\mathbb{G}_T}$ is the identity element of $\mathbb{G}_T$;
- Computability: There exists an efficient algorithm to compute $\hat{e}(g, h)$.

2.2. Framework and security model

A certificate-based encryption scheme is composed of the following five algorithms which can expire in polynomial time.

- **Setup**(κ)**.** Given a security parameter κ, this algorithm outputs the master secret key *msk* and the public parameters *pp*. Notice that, all other algorithms defined below can use the public parameters *pp*.
- **UserKeySet**(*pp, E*)**.** Given an entity *E*, this algorithm outputs the entity *E*'s secret key sk_E and its public key pk_E, where sk_E is picked by the entity *E* for itself independently.
- **CertGen(***pp, msk, t,* pk_E*, E***).** Given the master secret key *msk*, an entity *E* and its public key pk_E, this algorithm outputs the certificate $cert_E^t$ for the public key pk_E within a validity period *t*.
- **Encrypt(***pp, t,* pk_E*, E, M***).** Given an entity *E*'s public key pk_E and a message *M* to be encrypted, this algorithm outputs the ciphertext *c* for the message *M* under the public key pk_E of the entity *E*.
- **Decrypt**$(pp, c, cert_E^t, pk_E, sk_E)$. Given the ciphertext *c*, a certificate $cert_E^t$ and an entity public/secret key pair (pk_E, sk_E), this algorithm outputs a plain message *M* or an error symbol $\perp$.

For the correctness of CBE schemes, we usually require that

$$Pr\left[Decrypt(pp, c, cert_E^t, pk_E, sk_E)=M \;\middle|\; \begin{array}{c} Setup(\kappa)=(msk, pp) \\ UserKeySet(pp, E)=(sk_E, pk_E) \\ CertGen(pp, msk, t, pk_E, E)=cert_E^t \\ Encrypt(pp, t, pk_E, E, M)=c \end{array} \right]=1.$$

For the security of CBE schemes, we often take into account of two types of attackers [16]: Type I attacker $\mathcal{A}_1$ and Type II attacker $\mathcal{A}_2$. The Type I attacker knows the target entity secret key but cannot request the certificate for the entity public key. Conversely, the Type II attacker is allowed to adaptively initialize the system parameters, but cannot learn the target entity secret value chosen by itself. Here, we define the following two games to capture them by interacting with the challenger $\mathcal{C}$, respectively.

Game 1(for Type I attacker $\mathcal{A}_1$**)**

- **Init:** Given a security parameter κ, the challenger $\mathcal{C}$ invokes the **Setup** algorithm to produce the master secret key *msk* and public parameters *pp*. Here, $\mathcal{C}$ sends *pp* to $\mathcal{A}_1$ and keeps *msk* by itself.
- **Query phase 1:** In this phase, to facilitate an attack, $\mathcal{A}_1$ inquiries the following oracles adaptively:
  $\mathcal{O}^{pk}(E)$: Given an entity *E*, $\mathcal{C}$ invokes the **UserKeySet** algorithm to obtain the entity public key pk_E and transmits it to the attacker $\mathcal{A}_1$. Meanwhile, the key pair (sk_E, pk_E) is added by the challenger $\mathcal{C}$ to a list *L* of created entities, which is a precondition that other oracles defined below can work properly.
  $\mathcal{O}^{rep}(E, pk_E')$: Given a new entity public key pk_E', the challenger $\mathcal{C}$ finds and replaces the corresponding item in the list *L*.
  $\mathcal{O}^{sk}(E, pk_E)$: Given an entity *E* and its public key pk_E, the challenger $\mathcal{C}$ searches the list *L* to find out the entity secret key sk_E and returns it to the attacker $\mathcal{A}_1$.