Timed-Release Encryption with Master Time Bound Key

Gwangbae Choi(✉) and Serge Vaudenay

LASEC - Security and Cryptography Laboratory,
Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
{gwangbae.choi,serge.vaudenay}@epfl.ch

Abstract. Timed-release encryption allows senders to send a message to a receiver which cannot decrypt until a server releases a time bound key at the release time. The release time usually supposed to be known to the receiver, the ciphertext therefore cannot be decrypted if the release time is lost. We solve this problem in this paper by having a master time bound key which can replace the time bound key of any release time. We first present security models of the timed-release encryption with master time bound key. We present a provably secure construction based on the Weil pairing.

Keywords: Timed-release encryption · Weil pairing · Bilinear Diffie-Hellman problem

1 Introduction

The concept of timed-release encryption was first proposed by May [16]. The idea is to introduce the concept of time into an encryption scheme, especially into the decryption algorithm. There are two distinct approaches. One is to focus on the amount of time it takes to decrypt and the other is to have a trusted server to unlock encryption in due time. As time is one of the important aspects in the real world, timed-release encryption can be used for several purposes [20] such as bidding in an auction, as a personal time capsule, key escrow, etc. It can also be used to store sensitive data which should not be accessible before some time.

The first category of timed-release encryptions uses time-lock puzzles [20], which involves heavy computation for the decryption. The second one involves a trusted server [3,6–8,10,13]. It requires a time bound key which is periodically released by the trusted server for the decryption.

The timed-release encryption with a time-lock puzzle was first introduced by Rivest et al. [20]. They showed that the approach which makes available only some part of the decryption key and makes a receiver to brute force the remaining part of the decryption key is not sufficient for the timed-release encryption because it is parallelizable, so it offers no guarantee of the amount of time required to decrypt. They proposed a construction based on a time-lock puzzle which requires some non-parallelizable sequential computations on a single

I. You (Ed.): WISA 2019, LNCS 11897, pp. 167–179, 2020.
https://doi.org/10.1007/978-3-030-39303-8_13

processor. Therefore, it has some guarantee that the receiver will spend at least some time doing sequential computations.

Timed-release encryption with a trusted server was first proposed by May [16] while introducing this concept. The first approach is to send a message and a release time to a trusted server who then transfers the message after the release time is passed. Then, Rivest et al. [20] proposed a construction in which the trusted server does not store any message but this scheme suffers from problems of anonymity and confidentiality. Di Crescenzo et al. [10] proposed a construction based on a conditional oblivious transfer which allows a sender to be anonymous. But the receiver cannot be anonymous and the trusted server is a subject to denial-of-service attack. Later, Blake and Chan [3] proposed a construction based on the identity-based encryption scheme by Boneh and Franklin [5] in which the trusted server interacts with neither the sender nor the receiver. As Blake and Chan did not provide any security notion, Cathalo et al. [6] proposed its security notions and improved its construction. Based on the construction of Blake and Chan, Hwang et al. [13] proposed a construction with pre-open capability which allows a receiver to decrypt before the release time by using the pre-open key. As security analysis of this construction was not sufficient, Dent and Tang [11] introduced additional security models for the construction of Hwang et al. On the other hand, Cheon et al. [8] proposed a construction of authenticated timed-release encryption. Later, Chalkias et al. proposed a more efficient timed-release encryption scheme [7]. In 2009, Nakai et al. [18] proposed a generic construction of the timed-release encryption with pre-open capability by using an identity-based encryption and a public key encryption. Their generic construction was improved by Matsuda et al. [15] in terms of efficiency. In 2010, Paterson et al. [19] proposed the time-specific encryption paradigm. In time-specific encryption, a ciphertext can only be decrypted during a chosen time interval rather than after a chosen time. Therefore, the time-specific encryption can be seen as the generalization of the timed-release encryption. Later, Kasamatsu et al. [14] showed how the time-specific encryption can be derived from forward-secure encryption.

The first approach does not require any trusted server, but the sender does not have the full control on the release time of the encrypted message since it depends on the computational power of the receiver and the time it started to decrypt. With the second approach, the release time can be fully controlled by the sender since it requires a time bound key which will be released by the trusted server at the release time. However, for the protocol to work, it is necessary to include a trusted server and thus it may lead to security vulnerabilities due to the addition of another participant in the protocol.

In this paper, we focus on the second approach and we will study another potential problem which did not consider in previous works. In the previous works, the release time was usually somehow known to the receiver and the receiver could execute the decryption algorithm with the time bound key of the corresponding release time. Then, what happens if the receiver loses the release time? The receiver obviously cannot deduce which time bound key should be used

for the decryption. The receiver therefore cannot correctly decrypt the ciphertext since the time bound key of the release time is required for the decryption.

There already exist some easy ways to solve this problem. The sender for example can store the release time after the encryption, and sends it again to the receiver when the receiver asks the release time. This approach however cannot be an actual solution of the problem since an intuitive goal of timed-release encryption is to send a message for the time period when the sender and the receiver do not communicate. Another approach which does not require any communication between the sender and the receiver is to make the receiver to decrypt with all time bound keys. This solution however requires too much computation, compare to the normal decryption, and the receiver requires a way to check the correctness of the decrypted message.

The constructions with pre-open capability [13] might be a solution for the problem of losing the release time by giving the pre-open key which allows the decryption without the time bound key. The sender however needs to know the release time of the ciphertext to generate the corresponding pre-open key, it is equivalent to store the release time on the sender side. If the sender is storing the release time of the ciphertext, the sender can simply resend the release time to the receiver. The problem therefore becomes trivial. We hence consider the case neither the sender nor the receiver knows the release time.

Our Contributions and Structure

In this paper, we propose a better solution on this problem. We introduce a master time bound key which can be used as a valid time bound key for any release time. The receiver therefore can ask to the trusted server to decrypt a ciphertext of an unknown release time. This however can raise another problem with confidentiality of the message if the receiver needs to send the entire ciphertext to the trusted server for the decryption with master time bound time bound key. Our solution also solves this problem. A ciphertext of our construction consists of three elements. The receiver needs to send a single element to the trusted server to do the computation with master time bound key. Since this element is independent from the message, the trusted server cannot learn anything about the message.

The master time bound key moreover can be used when the trusted server terminates its service. Since a time bound key of the release time is needed for the decryption, the ciphertext whose release time is after the termination of the trusted server can never be decrypted. If it is more important not to lose the message than being decrypted before its release time, the trusted server needs to reveal its secret key or all future time bound keys to make the users able to decrypt their ciphertexts. If the trusted server reveals its secret key, receivers must implement another decryption algorithm which decrypts with the trusted server secret key instead of a time bound key. If the trusted server generates all future time bound keys (and possibly encrypt them with a timed-release encryption of another server), there might have a problem with the storage complexity if the amount of remaining time periods is huge. All of these solutions

therefore require some additional works. However, if the trusted server has the master time bound key, it is enough if the trusted server releases the master time bound key at the end of its service. Moreover, the storage overhead is minimized since the size of master time bound key is equal to the size of time bound key.

Finally, our master time bound key can play the role of a backup solution to decrypt messages in emergency situations (e.g. sudden disappear of the trusted server).

In this paper, we propose a timed-release encryption scheme which has the master time bound key that can be used to decrypt a ciphertext of any time period. In Sect. 2, we show the notions that we will use in this paper. In Sect. 3, we define primitives of timed-release encryption. In Sect. 4, we propose a construction of timed-release encryption scheme with master time bound key.

2 Preliminaries

We denote a concatenation of two bit strings a and b as $a||b$ and an empty input or output by $\perp$. We write $x \xleftarrow{\$} G$ if x is uniformly chosen from a set G. We denote an empty string or algorithm by ε. For any probabilistic algorithm $f(x)$, we denote an instance of the algorithm $f(x)$ with a sequence of random coins γ as $f(x;\gamma)$. For any g in some group G, a subgroup generated by g is written as $\langle g \rangle$. Let $X : \Omega \to S$ and $Y : \Omega \to S$ be two random variables. Then, the statistical distance between two random variables X and Y is $d(X,Y) = \frac{1}{2}\sum_{s\in S} |\Pr[X = s] - \Pr[Y = s]|$. We denote the uniform distribution over a set G by $\mathcal{U}_G$.

Definition 1 (Weil pairing [21, III.8.1]). *Let K be a finite field and E be an elliptic curve over K. The Weil pairing $e : E[m] \times E[m] \longrightarrow \mu_m$, where $E[m]$ is m-torsion subgroup of E and μ_m is m-th roots of unity in the algebraic closure $\bar{K}$, satisfies the following properties.*

1. *Bilinear: $\forall P_1, P_2, Q_1, Q_2 \in E[m]$, $e(P_1 + P_2, Q_1) = e(P_1, Q_1)e(P_2, Q_1)$ and $e(P_1, Q_1 + Q_2) = e(P_1, Q_1)e(P_1, Q_2)$.*
2. *Non-degenerate: $\forall P \in E[m], \exists Q \in E[m]$ such that $e(P, Q) \neq 1$.*
3. *Alternating: $\forall P \in E[m], e(P, P) = 1$.*
4. *Galois invariant: $\forall \sigma \in G_{\bar{K}/K}, e(P^\sigma, Q^\sigma) = e(P, Q)^\sigma$.*

We note that the Weil pairing can be efficiently computed by the Miller's algorithm [17].

Definition 2 (Decisional bilinear Diffie-Hellman problem [5]). *Let $\mathsf{Gen}(1^\lambda) = \pi = (\lambda, K, E, m, e)$ be an algorithm which generates appropriate instance of the decisional bilinear Diffie-Hellman problem, given the security parameter λ, where K is a field, E is an elliptic curve over K, and $e : E[m] \times E[m] \longrightarrow \mu_m$ is a bilinear map.*

We say that the decisional bilinear Diffie-Hellman problem is hard for Gen *if*

$$Adv^{DBDH}_{\mathcal{A}}(\lambda) = \left|\Pr\left[DBDH\text{-}0^{\mathcal{A}}_{\mathsf{Gen}}(\lambda) = 1\right] - \Pr\left[DBDH\text{-}1^{\mathcal{A}}_{\mathsf{Gen}}(\lambda) = 1\right]\right|$$

is a negligible function in λ for all probabilistic and polynomial time algorithm $\mathcal{A}$ where DBDH-d is defined as follows for $d \in \{0,1\}$.

Game: $\mathsf{DBDH\text{-}d}^{\mathcal{A}}_{\mathsf{Gen}}(\lambda)$
1 $\pi \leftarrow \mathsf{Gen}(1^\lambda)$
2 $(a_0, b_0, c_0) \xleftarrow{\$} \mathbb{Z}_m^3$
3 $(a_1, b_1, c_1) \xleftarrow{\$} \mathbb{Z}_m^3$
4 $(P, Q) \xleftarrow{\$} E[m] \times E[m]$
5 $d' \leftarrow \mathcal{A}(\pi, P, Q, a_0P, b_0P, c_0P, a_0Q, b_0Q, c_0Q, e(P,Q)^{a_d b_d c_d})$
6 **return** d'

3 Primitives of Timed-Release Encryption with Master Time Bound Key

In this section, we formally define the primitives of timed-release encryption with master time bound key. Our primitives are similar to the primitives in literatures [3,6,8,11,13]. The difference however is the key generation algorithm of the trusted server outputs the master time bound key along with the secret key and the public key.

Let S be a sender, R be a receiver and TS be a trusted server. We define a timed-release encryption scheme with master time bound key as follows:

Definition 3 (Timed-release encryption scheme with master time bound key). *A timed-release encryption scheme consists of the following algorithms:*

- $\mathsf{Setup}(1^\lambda) = \pi$ *is a probabilistic polynomial time algorithm which generates a system parameter π given a security parameter λ.*
- $\mathsf{KeyGen}_{TS}(\pi) = (sk_{TS}, pk_{TS}, mk_{TS})$ *is a probabilistic polynomial time algorithm of the trusted server TS which takes a system parameter π, and generates a secret key sk_{TS}, a public key of the trusted server pk_{TS} and a master time bound key mk_{TS}.*
- $\mathsf{KeyGen}_R(\pi) = (sk_R, pk_R)$ *is a probabilistic polynomial time algorithm of the receiver R which takes a system parameter π, and generates a secret key sk_R and a public key of the receiver pk_R.*
- $\mathsf{Broadcast}(sk_{TS}, t, \pi) = \tau_t$ *is a probabilistic polynomial time algorithm of the trusted server TS which takes a secret key of the trusted server pk_{TS}, scheduled broadcast time t and a system parameter π, and broadcasts time bound key τ_t.*
- $\mathsf{Enc}(pk_{TS}, pk_R, m, t, \pi) = c$ *is a probabilistic polynomial time algorithm of the sender S which takes a trusted server public key pk_{TS}, a receiver public key pk_R, a message m, release time t, and a system parameter π, and outputs a ciphertext c.*
- $\mathsf{Dec}(sk_R, \tau_t, c, \pi) = m$ *is a deterministic polynomial time algorithm of the receiver R which takes a receiver secret key sk_R, a time bound key at the*

release time t τ_t, a ciphertext c, and a system parameter π, and outputs a message m or $\perp$.

Then, we expect a timed-release encryption scheme to satisfy the following condition:

- *For any security parameter λ, for any system parameter $\pi = \mathsf{Setup}(1^\lambda)$, for any trusted server key pair $(sk_{TS}, pk_{TS}, mk_{TS}) = \mathsf{KeyGen}_{TS}(\pi)$, for any receiver key pair $(sk_R, pk_R) = \mathsf{KeyGen}_R(\pi)$, for any message m and for any time period t,*

$$\Pr_{\gamma_1,\gamma_2}[\mathsf{Dec}(sk_R, \mathsf{Broadcast}(sk_{TS}, t, \pi; \gamma_1), \mathsf{Enc}(pk_{TS}, pk_R, m, t, \pi; \gamma_2), \pi) = m] = 1$$

and

$$\Pr_{\gamma}[\mathsf{Dec}(sk_R, mk_{TS}, \mathsf{Enc}(pk_{TS}, pk_R, m, t, \pi; \gamma), \pi) = m] = 1$$

The key generation algorithm of the receiver $\mathsf{KeyGen}_\mathrm{R}$ sometimes takes the trusted server public key pk_TS as input. We however define our $\mathsf{KeyGen}_\mathrm{R}$ to be independent from pk_TS as it was done in some constructions [15,18]. If $\mathsf{KeyGen}_\mathrm{R}$ is dependent to pk_TS, the receiver needs to get the trusted server public key before the generation of its key pair. If they are independent, the receiver does not need any communication with the trusted server before the release time, it will be therefore more efficient.

The timed-release encryption has two security objectives. One is the confidentiality of the message until its release time against the receiver. The other is the anonymity of the sender and the receiver against the trusted server.

4 Construction with Master Time Bound Key

In this section, we propose a timed-release encryption scheme TRE which has the master time bound key. In addition, our construction does not require $\mathsf{KeyGen}_\mathrm{R}$ to be dependent to pk_TS and a hash function which maps to a point on the elliptic curve. Let h_κ be a collision-resistant hash function from $K^* \times E[q]$ to a set F, $\mathcal{E}$ be an asymmetric encryption scheme which consists of $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with plaintext space $K \times F$, and f_π be a pseudorandom generator from μ_q to K, i.e. for $\omega \in \mu_q$ uniformly distributed, $f_\pi(\omega)$ is computationally indistinguishable from the uniform distribution over K. Then, our construction with plaintext space K^* is as follows. We note that our Broadcast is similar to KeyGen of the identity-based encryption scheme of Boneh and Boyen [4], which generates the secret key of a user which can be used to compute the inverse of the random value which is multiplied to the message, and TS-release of the timed-release encryption scheme of Cathalo et al. [6], which computes $g^{-(s+H(t))}$ where s is the secret key, $H(t)$ is the hash of a time period t and g is a generator of a group.

- $\mathsf{TRE.Setup}(1^\lambda)$: Pick two prime numbers p and q such that $q|(p \pm 1)$. Pick the finite field $K = \mathbb{F}_{p^2}$ and a supersingular elliptic curve $E(K)$ of cardinality $(p \pm 1)^2$. Then, compute q-torsion subgroup $E[q]$ and the Weil pairing $e : E[q] \times E[q] \longrightarrow \mu_q$ where μ_q is the group of q-th roots of unity in K. Pick κ from the key space of h and output $\pi = (\lambda, K, E, q, e, \kappa)$.

– $\mathsf{TRE.KeyGen}_{\mathrm{TS}}(\pi)$: Pick P and Q from $E[q]$ such that $|\langle P\rangle| = |\langle Q\rangle| = q$ and $P \notin \langle Q\rangle$, and pick a, b, c, d uniformly from $\mathbb{Z}_q^*$ until $\langle(1, a)\rangle$, $\langle(b, 1)\rangle$ and $\langle(c, d)\rangle$ are distinct subgroups of $\mathbb{Z}_q \times \mathbb{Z}_q$. Then, compute

$$mk_{\mathrm{TS}} = (1 - ab)(bd - c)^{-1}(bP + Q),$$
$$sk_{\mathrm{TS}} = (a, b, c, d, P, Q)$$

and

$$pk_{\mathrm{TS}} = (pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)}, pk_{\mathrm{TS}}^{(2)}) = (P + aQ, bP + Q, cP + dQ),$$

and output sk_{TS}, pk_{TS} and mk_{TS}.

Property 1. $e(P, P) = e(Q, Q) = 1$, $e(P, Q)e(Q, P) = 1$ and $e(P, Q) \neq 1$. (See the proof below.)

Property 2. $e(pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)}) = e(P, Q)^{1-ab} \neq 1$ because $\langle(1, a)\rangle$ and $\langle(b, 1)\rangle$ are distinct subgroups of $\mathbb{Z}_q \times \mathbb{Z}_q$.

– $\mathsf{TRE.KeyGen}_{\mathrm{R}}(1^\lambda)$: Generate a pair of secret and public keys (sk, pk) by calling $\mathcal{E}.\mathsf{KeyGen}(1^\lambda)$. Then, output $sk_{\mathrm{R}} = sk$ and $pk_{\mathrm{R}} = pk$.
– $\mathsf{TRE.Broadcast}(sk_{\mathrm{TS}}, t, \pi)$: Pick s uniformly from $\mathbb{Z}_q^*$. Compute

$$\tau_t = \begin{cases} sP + (ab - 1)(c + bt)^{-1}Q, & \text{if } t = -d \\ (1 - ab)(d + t)^{-1}P + sQ, & \text{if } t = -cb^{-1} \\ s(d + t)^{-1}P + (s + ab - 1)(c + bt)^{-1}Q, & \text{otherwise.} \end{cases}$$

Property 3. $e(\tau_t, t \cdot pk_{\mathrm{TS}}^{(1)} + pk_{\mathrm{TS}}^{(2)}) = e(mk_{\mathrm{TS}}, t \cdot pk_{\mathrm{TS}}^{(1)} + pk_{\mathrm{TS}}^{(2)}) = e(P, Q)^{1-ab}$ (See the proof below.)

– $\mathsf{TRE.Enc}(pk_{\mathrm{TS}}, pk_{\mathrm{R}}, m, t, \pi)$: Output $\perp$ if $m \notin K^*$. Pick r_1 uniformly from $\mathbb{Z}_q^*$ and pick r_2 uniformly from K^*. Then, compute

$$ct_0 = m \cdot r_2,$$
$$ct_1 = r_1 t \cdot pk_{\mathrm{TS}}^{(1)} + r_1 \cdot pk_{\mathrm{TS}}^{(2)},$$
$$ct_2 = \mathcal{E}.\mathsf{Enc}(pk_{\mathrm{R}}, (r_2 + f_\pi(e(pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)})^{r_1}), h_\kappa(ct_0, ct_1)))$$

and output $ct = (ct_0, ct_1, ct_2)$.

Property 4. $e(\tau_t, ct_1) = e(pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)})^{r_1}$

– $\mathsf{TRE.Dec}(sk_{\mathrm{R}}, \tau_t, ct, \pi)$: Compute

$$(r_2', \sigma) = \mathcal{E}.\mathsf{Dec}(sk_{\mathrm{R}}, ct_2).$$

Output

$$m = ct_0 \cdot (r_2' - f_\pi(e(\tau_t, ct_1)))^{-1}$$

if $\sigma = h_\kappa(ct_0, ct_1)$, and output $\perp$ otherwise.

Proof of Property 1. $e(P,P) = e(Q,Q) = e(P+Q,P+Q) = 1$ comes from the alternating property of the Weil pairing. Hence, $1 = e(P+Q,P+Q) = e(P,Q)e(Q,P)$ due to bilinearity. Now, assume that there exists $P,Q \in E[q]\backslash\{O\}$ such that $P \notin \langle Q\rangle$ and $e(P,Q) = 1$. Then, we have $e(P,\alpha P+\beta Q) = e(P,Q)^{\beta} = 1$ for any $\alpha,\beta \in \mathbb{Z}_q$. Since q is prime, $\{\alpha P + \beta Q : \alpha,\beta \in \mathbb{Z}_q\} = E[q]$. Hence, it contradicts non-degeneracy, and such P and Q do not exist. Consequently, $e(P,Q) \neq 1$ and $e(P,Q)^{-1} = e(Q,P)$. □

Proof of Property 3. When $t \neq -d$ and $t \neq -cb^{-1}$, we have

$$\begin{aligned}
&e(\tau_t, t\cdot pk_{\mathrm{TS}}^{(1)} + pk_{\mathrm{TS}}^{(2)})\\
&= e(s(d+t)^{-1}P + (s+ab-1)(c+bt)^{-1}Q, (c+bt)P + (d+t)Q)\\
&= e(s(d+t)^{-1}P, (d+t)Q)e((s+ab-1)(c+bt)^{-1}Q, (c+bt)P)\\
&= e(P,Q)^{s}e(Q,P)^{s+ab-1}\\
&= e(P,Q)^{1-ab}.
\end{aligned}$$

When $t = -d$, we have

$$\begin{aligned}
e(\tau_t, t\cdot pk_{\mathrm{TS}}^{(1)} + pk_{\mathrm{TS}}^{(2)}) &= e(sP + (ab-1)(c+bt)^{-1}Q, (c+bt)P)\\
&= e(P,Q)^{1-ab}.
\end{aligned}$$

Similarly, when $t = -cb^{-1}$, we have

$$\begin{aligned}
e(\tau_t, t\cdot pk_{\mathrm{TS}}^{(1)} + pk_{\mathrm{TS}}^{(2)}) &= e((1-ab)(d+t)^{-1}P + sQ, (d+t)Q)\\
&= e(P,Q)^{1-ab}.
\end{aligned}$$

With mk_{TS}, we can also obtain same result regardless of t.

$$\begin{aligned}
&e(mk_{\mathrm{TS}}, t\cdot pk_{\mathrm{TS}}^{(1)} + pk_{\mathrm{TS}}^{(2)})\\
&= e((1-ab)(bd-c)^{-1}(bP+Q), (c+bt)P + (d+t)Q)\\
&= e((1-ab)(bd-c)^{-1}bP, (d+t)Q)e((1-ab)(bd-c)^{-1}Q, (c+bt)P)\\
&= e(P,Q)^{(1-ab)(bd-c)^{-1}(b(d+t)-c-bt))}\\
&= e(P,Q)^{1-ab}.
\end{aligned}$$

□

By the choice of parameters, the q-th torsion subgroup $E[q]$ is a proper subset of E over K. Since $E[q] \cong \mathbb{Z}_q \times \mathbb{Z}_q$ [21], there exist $q+1$ distinct subgroups of order q in $E[q]$ and every element in $E[q] \setminus \{O\}$ generates a subgroup of order q. Therefore, we can deduce that $e(P,Q) = 1 \iff P \in \langle Q\rangle$ for all $P,Q \in E[q]$. Hence, in $\mathsf{TRE.KeyGen}_{\mathrm{TS}}$, $|\langle P\rangle| = |\langle Q\rangle| = q$ always holds and $P \notin \langle Q\rangle$ holds with probability of $\frac{q}{q+1}$ for any P and Q randomly chosen from $E[q]$, and $P \notin \langle Q\rangle$ can be easily verified by checking if $e(P,Q)$ is not equal to 1.

Assume that $\mathcal{E}.\mathsf{Dec}(sk, \mathcal{E}.\mathsf{Enc}(pk,m)) = m$ always holds for any message m and key pair (sk,pk) generated by using $\mathcal{E}.\mathsf{KeyGen}$ with some random coin. Then,

TRE.Dec is correct if $e(pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)})^{r_1} = e(\tau_t, ct_1)$. From the choice of keys, we have

$$\begin{aligned} e(pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)})^{r_1} &= e(P + aQ, bP + Q)^{r_1} \\ &= e(P, bP + Q)^{r_1} e(aQ, bP + Q)^{r_1} \\ &= e(P, bP)^{r_1} e(P, Q)^{r_1} e(aQ, bP)^{r_1} e(aQ, Q)^{r_1} \\ &= e(P, Q)^{r_1(1-ab)}. \end{aligned}$$

Since $ct_1 = r_1(t \cdot pk_{\mathrm{TS}}^{(1)} + pk_{\mathrm{TS}}^{(2)})$, the decryption is always correct.

4.1 Security Analysis

In this section, we will show the following results:

- Indistinguishability under chosen plaintext attacks (IND-CPA security) of $\mathcal{E}$ implies indsitinguishability under chosen plaintext attacks of trusted server[1] (IND-TS-CPA security) of TRE. This security does not depend on h_κ which could be set to a constant function;
- Indistinguishability under chosen ciphertext attacks (IND-CCA security) of $\mathcal{E}$ and the collision-resistance of h_κ imply indistinguishability under chosen ciphertext attacks of trusted server[1] (IND-TS-CCA security) of TRE;
- Hardness of the decisional bilinear Diffie-Hellman problem and the PRG property of f_π imply indistinguishability under chosen plaintext attacks of receiver for a selected release time[2] (IND-R-ST-CPA security) of TRE.

The detailed security definitions and the proofs of following theorems can be found from the full version of the paper [9].

Theorem 1 (IND-TS-CPA security). *Let $\mathcal{A}$ be an IND-TS-CPA adversary against* TRE *which runs in time η with advantage δ. Then, there exists an IND-CPA adversary $\mathcal{B}$ against $\mathcal{E}$. The advantage of $\mathcal{B}$ is at least δ and its time complexity is $\eta + \eta_e + \eta_{f_\pi}$ where η_e is the time to evaluate the pairing $e(\cdot, \cdot)$, η_e is the time to evaluate the pairing $e(\cdot, \cdot)$ and η_{f_π} is the evaluation time of f_π.*

Theorem 2 (IND-TS-CCA security). *Let $\mathcal{A}$ be an IND-TS-CCA adversary against* TRE *which runs in time η with advantage δ. Then, there exist an IND-CCA adversary $\mathcal{B}$ against $\mathcal{E}$ and a collision adversary $\mathcal{C}$ against h_κ. The advantage of adversary $\mathcal{B}$ is at least $\delta - \delta_{h_\kappa}$ and its time complexity is $\eta + \eta_e + \eta_{f_\pi} + \eta_{h_\kappa}$ where η_e is the time to evaluate the pairing $e(\cdot, \cdot)$, η_e is the time to evaluate the pairing $e(\cdot, \cdot)$, η_{f_π} is the evaluation time of f_π, η_{h_κ} is the evaluation time of h_κ and δ_{h_κ} is the advantage of $\mathcal{C}$.*

[1] An adversary can select pk_{TS}.

[2] An adversary needs to declare a release time that it wants to attack before getting any public key and can selects pk_{R}.

Theorem 3 (IND-R-ST-CPA security). *Let $\mathcal{A}$ be an IND-R-ST-CPA adversary against* TRE *which runs in time η with advantage δ. Then, there exist an algorithm $\mathcal{B}$ which solves the decisional bilinear Diffie-Hellman problem and a distinguisher $\mathcal{D}$ between $f_\pi(\mathcal{U}_{\mu_q})$ and $\mathcal{U}_K$. The advantage of $\mathcal{B}$ is at least $\delta - 3/q - \delta_{f_\pi}$ and its time complexity is $\eta + 3\eta_e + \eta_{\mathcal{E}.\mathsf{Enc}}$ where δ_{f_π} is the advantage of $\mathcal{D}$, η_e is the time to evaluate the pairing $e(\cdot,\cdot)$, and $\eta_{\mathcal{E}.\mathsf{Enc}}$ is the execution time of $\mathcal{E}.\mathsf{Enc}$.*

4.2 Decryption with Master Time Bound Key

The biggest difference between our construction and other constructions is the existence of the master time bound key. By using the master time bound key, a ciphertext of unknown release time can be decrypted. By our construction, a ciphertext consists of (ct_0, ct_1, ct_2). In order to decrypt a ciphertext, we need to compute $e(\tau_t, ct_1)$ should be computed. Due to Property 3, the master time bound key mk_{TS} can replace any time bound key. Indeed, the receiver only needs to ask the trusted server to compute $e(mk_{\mathrm{TS}}, ct_1)$ to decrypt the ciphertext. Since ct_1 is independent from the message, the trusted server cannot learn anything about the message while computing $e(mk_{\mathrm{TS}}, ct_1)$.

Similarly, the trusted server can terminate its service without any computational and storage overhead while preventing losing the encrypted data of users by revealing the master time bound key. Since the master time bound key can replace any time bound key, we do not need any extra algorithm for the decryption with mk_{TS}. This is an advantage for the trusted server as it does not need to provide any additional algorithm for the decryption with master time bound key.

On the other hand, the time bound key τ_t which is generated by $\mathsf{TRE.Broadcast}$ can be equal to the master time bound key mk_{TS} depending on the random value s. Therefore, the master time bound key can be broadcasted by the trusted server as a time bound key of a certain time period. However, it can happen with probability of at most $1/(q-1)$ where q is exponential in the security parameter λ, so it happens in negligible cases. The trusted server could also easily prevent this problem by comparing the time bound key with master time bound key before the broadcast.

4.3 Discussion

Since our construction uses an elliptic curve over an extension field $\mathbb{F}_{p^2}$, we first need to know what is the computational overhead compared to other constructions which work on $\mathbb{F}_p$. However, it is not easy to compare the exact overhead because some constructions [3,6–8,13] are based on the generic bilinear pairing, and some constructions [15,18] are based on the generic identity-based encryption. Therefore, their computational cost is dependent on the underlying bilinear pairing and the underlying identity-based encryption scheme. An identity-based

encryption scheme is usually based on the bilinear pairing[3], and it always requires at least one evaluation of the bilinear pairing. One of most common instantiation of the bilinear pairing is to use the Weil pairing or the Tate pairing after applying a distortion map to one of two input points. Since the distortion map maps a point defined on the elliptic curve over a field $\mathbb{F}_p$ to $\mathbb{F}_{p^2}$, the computation of the Weil pairing or the Tate pairing is actually the computations on $\mathbb{F}_{p^2}$. Therefore, the asymptotic complexities of our construction and other constructions are similar as long as the bilinear pairing is the most complex computation.

Our construction can also be built on the top of generic bilinear pairings. Let G be an additive cyclic group, G_T be a multiplicative cyclic group, and $\hat{e} : G \times G \longrightarrow G_T$ be a bilinear pairing. If we define $P = (g, 0)$, $Q = (0, g)$ and $e(aP + bQ, cP + dQ) = e((ag, bg), (cg, dg)) = \hat{e}(ag, dg)\hat{e}(cg, bg)^{-1}$, we can obtain the same construction on the top of generic pairing. The computation of e however requires two evaluations of a generic bilinear pairing $\hat{e}$. As we mentioned in the previous paragraph, a generic bilinear pairing is usually instantiated with the Weil pairing or the Tate pairing. We therefore use the Weil pairing over $\mathbb{F}_{p^2}$ for the efficiency. We note that the construction with a generic pairing can be more efficient than our construction with the Weil pairing if one can instantiate a more efficient bilinear pairing.

In our construction, the encryption requires a single evaluation of the Weil pairing e. Since the encryption always requires to compute $e(pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)})$, it can be precomputed by the trusted server and integrated into the trusted server public key. Therefore, we can make the encryption faster by replacing the trusted server public key pk_{TS} to $(e(pk_{\mathrm{TS}}^{(0)}, pk_{\mathrm{TS}}^{(1)}), pk_{\mathrm{TS}}^{(1)}, pk_{\mathrm{TS}}^{(2)})$.

5 Conclusion

In this paper, we proposed a timed-release encryption scheme which has the master time bound key. With master time bound key, a ciphertext can be decrypted even if the release time of the ciphertext is unknown. We also showed that our construction is IND-TS-CCA-secure and IND-R-ST-CPA-secure.

Acknowledgement. Gwangbae Choi is supported by the Swiss National Science Foundation (SNSF) Proejct funding no. 169110.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
2. Agrawal, S., Boyen, X.: Identity-based encryption from lattices in the standard model. Manuscript, July 2009

[3] There also exist several identity-based encryption schemes which do not require a bilinear pairing [1,2,12], but we do not compare with them.

3. Blake, I.F., Chan, A.C.: Scalable, server-passive, user-anonymous timed release public key encryption from bilinear pairing. IACR Cryptology ePrint Archive (2004)
4. Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_27
5. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. SIAM J. Comput. **32**, 586–615 (2003)
6. Cathalo, J., Libert, B., Quisquater, J.-J.: Efficient and non-interactive timed-release encryption. In: Qing, S., Mao, W., López, J., Wang, G. (eds.) ICICS 2005. LNCS, vol. 3783, pp. 291–303. Springer, Heidelberg (2005). https://doi.org/10.1007/11602897_25
7. Chalkias, K., Hristu-Varsakelis, D., Stephanides, G.: Improved anonymous timed-release encryption. In: Biskup, J., López, J. (eds.) ESORICS 2007. LNCS, vol. 4734, pp. 311–326. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74835-9_21
8. Cheon, J.H., Hopper, N., Kim, Y., Osipkov, I.: Timed-release and key-insulated public key encryption. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, pp. 191–205. Springer, Heidelberg (2006). https://doi.org/10.1007/11889663_17
9. Choi, G., Vaudenay, S.: Timed-release encryption with master time bound key (full version). Cryptology ePrint Archive, Report 2019/904 (2019). https://eprint.iacr.org/2019/904
10. Di Crescenzo, G., Ostrovsky, R., Rajagopalan, S.: Conditional oblivious transfer and timed-release encryption. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 74–89. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48910-X_6
11. Dent, A.W., Tang, Q.: Revisiting the security model for timed-release encryption with pre-open capability. In: Garay, J.A., Lenstra, A.K., Mambo, M., Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 158–174. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75496-1_11
12. Döttling, N., Garg, S.: Identity-based encryption from the Diffie-Hellman assumption. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 537–569. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_18
13. Hwang, Y.H., Yum, D.H., Lee, P.J.: Timed-release encryption with pre-open capability and its application to certified e-mail system. In: Zhou, J., Lopez, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 344–358. Springer, Heidelberg (2005). https://doi.org/10.1007/11556992_25
14. Kasamatsu, K., Matsuda, T., Emura, K., Attrapadung, N., Hanaoka, G., Imai, H.: Time-specific encryption from forward-secure encryption: generic and direct constructions. Int. J. Inf. Secur. **15**, 549–571 (2016)
15. Matsuda, T., Nakai, Y., Matsuura, K.: Efficient generic constructions of timed-release encryption with pre-open capability. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 225–245. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17455-1_15
16. May, T.C.: Timed-release crypto (1993). http://www.hks.net.cpunks/cpunks-0/1460.html
17. Miller, V., et al.: Short programs for functions on curves. Unpublished Manuscript 97 (1986)

18. Nakai, Y., Matsuda, T., Kitada, W., Matsuura, K.: A generic construction of timed-release encryption with pre-open capability. In: Takagi, T., Mambo, M. (eds.) IWSEC 2009. LNCS, vol. 5824, pp. 53–70. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04846-3_5
19. Paterson, K.G., Quaglia, E.A.: Time-specific encryption. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 1–16. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15317-4_1
20. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto (1996)
21. Silverman, J.H.: The Arithmetic of Elliptic Curves, vol. 106. Springer, New York (2009). https://doi.org/10.1007/978-0-387-09494-6