SCIENCE CHINA
Information Sciences

• **Supplementary File** •

Space Efficient Revocable IBE for Mobile Devices in Cloud Computing

Baodong QIN[1,2*], Ximeng LIU[3], Zhuo WEI[4] & Dong ZHENG[1,5*]

[1]*National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, Xi'an* 710121*, P.R. China;*
[2]*State Key Laboratory of Cryptology, P.O.Box5159, Beijing* 100878*, P.R. China;*
[3]*School of Information Systems, Singapore Management University, Singapore* 178902*, Singapore;*
[4]*Huawei Singapore Research Center, Singapore* 117674*, Singapore;*
[5]*Westone Cryptologic Research Center, Beijing* 100070*, P.R. China*

Appendix A Definition and Security Model for Server-Aided Revocable IBE

Let $\mathcal{I}$ and $\mathcal{T}$ be an identity space and a time space respectively. A server-aided revocable IBE scheme [1] consists of the following (probabilistic) polynomial-time algorithms:

• $\mathsf{Setup}(1^\lambda) \rightarrow (sp, msk, rl)$: The setup algorithm takes as input the security parameter 1^κ, and outputs the system parameters sp, a master secret key msk and a revocation list rl initialized as an empty set.

• $\mathsf{PubKG}(msk, id) \rightarrow pk_{id}$: The long-term transformation key generation algorithm takes as input the master secret key sk and an identity id. It outputs a long-term (identity-based) transformation key pk_{id}, and sends it to the server.

• $\mathsf{TruKU}(msk, rl, t) \rightarrow ku_t$: The transformation key update algorithm takes as input the master secret key msk, the current revocable list rl and the time period t. It outputs a key update information ku_t, and sends it to the server.

• $\mathsf{TranKG}(id, t, pk_{id}, ku_t) \rightarrow tk_t$: The transformation key generation algorithm takes as input an identity id, a time period t, the long-term transformation key pk_{id} corresponding to that user, and a key update information ku_t corresponding to that time period. The server computes a time-based short-term transformation key $tk_{id,t}$ for user id.

• $\mathsf{PrivKG}(msk, id) \rightarrow sk_{id}$: The long-term private key generation algorithm takes as input the master secret key msk and an identity $id \in \mathcal{I}$, and outputs a long-term private key sk_{id}. The PKG sends sk_{id} to the corresponding user.

• $\mathsf{DecKG}(sk_{id}, t) \rightarrow dk_{id,t}$: The decryption key generation algorithm takes as input the long-term private key sk_{id} and a time $t \in \mathcal{T}$, and outputs a short-term decryption key $dk_{id,t}$.

• $\mathsf{Encrypt}(sp, M, id, t) \rightarrow C_{id,t}$: The encryption algorithm takes as input the system parameters sp, a message M, an identity $id \in \mathcal{I}$ and a time $t \in \mathcal{T}$, and outputs a ciphertext $C_{id,t}$.

• $\mathsf{Transform}(tk_t, C_{id,t}) \rightarrow C'_{id,t}$: The ciphertext transformation algorithm takes as input the transformation key $tk_{id,t}$ (corresponding to user id and time period t) and a ciphertext $C_{id,t}$ (encrypted under (id, t)). It outputs a partially decrypted ciphertext $C'_{id,t}$ if id is not revoked at time t and the error symbol $\perp$ otherwise.

• $\mathsf{Decrypt}(dk_{id,t}, C'_{id,t}) \rightarrow M$: The decryption algorithm takes as input a decryption key $dk_{id,t}$ for (id, t) and a partially decrypted ciphertext $C'_{id,t}$ that was originally encrypted under (id, t). It outputs the message M if id is not revoked at time t and the error symbol $\perp$ otherwise.

• $\mathsf{Revoke}(rl, id, t) \rightarrow rl$: The revocation algorithm takes as input the revocation list rl, an identity to be revoked $id \in \mathcal{I}$ and revocation time $t \in \mathcal{T}$, and outputs an updated revocation list rl.

For correctness, we require that for all $\kappa \in \mathbb{N}$, all sp and msk output by Setup algorithm, all message $M \in \mathcal{M}$, $id \in \mathcal{I}$, $t \in \mathcal{T}$ and all revocation list rl, if identity id was not revoked before or at time t, then $\mathsf{Decrypt}(dk_{id,t}, \mathsf{Transform}(tk_{id,t}, \mathsf{Encrypt}(sp, M, id, t)) = M$, if $sk_{id} \leftarrow \mathsf{PrivKG}(msk, id)$, $dk_{id,t} \leftarrow \mathsf{DecKG}(sk_{id}, t)$, $pk_{id} \leftarrow \mathsf{PubKG}(msk, id)$, $ku_t \leftarrow \mathsf{TranKU}(msk, rl, t)$ and $tk_{id,t} \leftarrow \mathsf{TranKG}(id, t, pk_{id}, ku_t)$.

Selective-ID Security. We define the selective-ID security against chosen-plaintext attacks (sID-CPA security) for a server-aided revocable IBE scheme. Our security model captures the following scenarios: (1) A revoked user cannot decrypt any new ciphertexts, even if the user colludes with other users (including the server). (2) Any non-revoked user, e.g., Alice, can delegate a decryption key for some specified period of time t to other user, e.g., Bob, so that Bob can only decrypt

* Corresponding author (email: qinbaodong@xupt.edu.cn, zhengdong_xupt@sina.com)

Alice's cipertexts encrypted under time t, even if Bob colludes with other users (including the server). The formal definition of sID-CPA security is given below. The advantage of an adversary $\mathcal{A}$ in the following game is defined as $|\Pr[b' = b] - 1/2|$.

Game $\mathcal{G}$:

Initiation. The adversary $\mathcal{A}$ commits a challenge identity $id^* \in \mathcal{I}$ and a period of revocation time $t^* \in \mathcal{T}$ to the challenger.

Setup. The challenger runs $(sp, msk, rl) \leftarrow \mathsf{Setup}(1^\kappa)$ and sends sp to $\mathcal{A}$.

Phase 1. The adversary can repeatedly and adaptively make any of the following queries except that (1) if $\mathcal{A}$ queries the private key generation oracle for the challenge identity id^*, the challenge identity must be revoked at or before the period of time t^*, and (2) $\mathcal{A}$ can not make a query of (id^*, t^*) to the decryption key generation oracle.

- $\mathcal{O}^{\mathsf{PubKG}}(\cdot)$ (Long-term transformation key generation oracle): On input an identity id, it runs $\mathsf{PubKG}(msk, id)$ to return long-term transformation key pk_{id}.
- $\mathcal{O}^{\mathsf{TranKU}}(\cdot)$ (Key update generation oracle): On input a period of time t, it runs $\mathsf{TranKU}(msk, rl, t)$ to return key update information ku_t.
- $\mathcal{O}^{\mathsf{PrivKG}}(\cdot)$ (Private key generation query): On input an identity id, it runs $\mathsf{PrivKG}(msk, id)$ to return private key sk_{id}.
- $\mathcal{O}^{\mathsf{DecKG}}(\cdot, \cdot)$ (Decryption key generation query): On input an identity id and a time period t, it runs first runs $\mathsf{PrivKG}(msk, id)$ to obtain a private key sk_{id}, then runs $\mathsf{DecKG}(sk_{id}, t)$ to return decryption key $dk_{id,t}$.
- $\mathcal{O}^{\mathsf{Revoke}}(\cdot)$ (Revocation oracle): On input an identity id, it runs $\mathsf{Revoke}(rl, id, t)$ to update the revocation list.

Challenge. The adversary submits two equal length messages M_0 and M_1. The challenger flips a random coin $b \in \{0, 1\}$, and encrypts M_b under (id^*, t^*). The resulting ciphertext $C^*_{id^*,t^*}$ is given to the adversary.

Phase 2. The adversary proceeds as in Phase 1.

Guess. The adversary outputs a guess bit b' of b.

Definition 1 (sID-CPA security)**.** A server-aided revocable IBE scheme is sID-CPA secure if any PPT adversary has at most a negligible advantage in the sID-CPA security game.

Extension to Adaptive-ID Security. We say a server-aided revocable IBE scheme is adaptive-ID secure if the adversary commits to the challenge identity and period of time after Phase 1 rather than in Initiation stage.

Appendix B Proof of Theorem 1

This section provides the provable security of the above theorem. We reduce the security of our schemes to the security of the OSW scheme. In the security reduction, there are three parties: the OSW challenger $\mathcal{C}$, the SR-IBE adversary $\mathcal{A}$ and a simulator $\mathcal{B}$. The simulator has two identities. Faced with the OSW challenger, $\mathcal{B}$ serves as the OSW adversary to break the OWS scheme. While faced with the SR-IBE adversary, $\mathcal{B}$ serves as the SR-IBE challenger. The reduction process is depicted in Fig. B1.

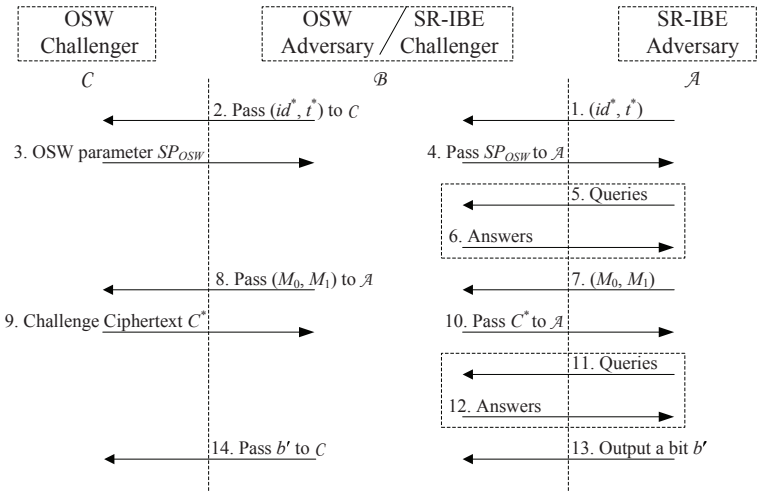

Figure B1 Proof reduction process

Let $\mathcal{A}$ be the adversary that breaks our server-aided revocable IBE scheme in the sID-CPA security game. Then we build an adversary $\mathcal{B}$ (also called simulator) that breaks the selective-set security of the underlying OSW KP-ABE scheme. The simulator plays the role of the challenger and interacts with $\mathcal{A}$ in the sID-CPA game.

We divide our proof into two separate parts for two different types of adversaries:

- Type I: $\mathcal{A}$ queries the private key generation oracle for the challenge identity id^*. In this case, the challenge identity must be revoked before the challenge time period t^*.

• Type II: $\mathcal{A}$ never queries the private key generation oracle for the challenge identity id^*. In this case, the adversary may query the decryption key generation oracle for (id, t) as long as $(id, t) \neq (id^*, t^*)$.

Note that the above two types of adversaries we consider are complements of each other and therefore partition the space of all possible adversaries. Hence, adversary $\mathcal{A}$ is a member of exactly one of these two sets. The simulator $\mathcal{B}$ has to randomly guess which type of adversary is going to be. The guess is correct with probability 1/2 which leads to a factor of two reduction in tightness of security.

Proof for Type I Adversary.

Initiation. Initially, the simulator $\mathcal{B}$ receives an identity id^* and a period of time t^* from $\mathcal{A}$. The simulator gives $S^* = (id^*, t^*)$ to the OSW challenger.

Setup. Next, $\mathcal{B}$ receives the system parameters sp_{OSW}=(g, g_1, $g_2 = g^{q(0)}$, $g^{q(1)}$, $g^{q(2)}$, $g^{h(0)}$, $g^{h(1)}$, $g^{h(2)}$, $T(x)$, $V(x)$) from the OSW challenger and passes them on to $\mathcal{A}$. Here, $g_1 = g^{\alpha}$ and $q(0) = \beta$ for some unknown $\alpha, \beta \in \mathbb{Z}_p$. In addition, the simulator chooses a random term $\alpha_2 \in \mathbb{Z}_p$ and implicitly sets $\alpha_1 = \alpha - \alpha_2 \pmod p$, where α is the master secret key of the OSW KP-ABE scheme.

Phase 1. The simulator answers $\mathcal{A}$'s queries as follows. As our scheme does not pre-distribute long-term transformation keys for each identity, we do not need to simulate adversary's long-term transformation key queries and time-based short-term transformation key queries. For queries on users' private keys and decryption keys, the simulator uses α_2 to compute them as in the actual scheme. For queries on the revocation list, the simulator directly adds the revoked identity ω into rl. For queries on the transformation key update for time period t, the simulator chooses the key policy $\mathbb{A} = t \wedge_{\omega \in rl} \neg\omega$ and sends it to its OSW challenger, and receives a secret key $D = (D_t, \{D_{\neg\omega}\}_{\omega \in rl})$. Here, $D_t = (g_2^{\lambda_t} T(t)^{r_t}, g^{r_t})$ and $D_{\neg\omega} = (g_2^{\lambda_\omega + r_\omega}, V(\omega)^{r_\omega}, g^{r_\omega})$, and $\lambda_t + \sum_{\omega \in rl} \lambda_\omega = \alpha \pmod p$. The simulator sets $E_{\neg\omega} := D_{\neg\omega}$ for all $\omega \in rl$, while $E_t := (g_2^{\lambda_t} T(t)^{r_t} \cdot g_2^{-\alpha_2}, g^{r_t})$. Then, the simulator returns $ku_t := (E_t, \{E_{\neg\omega}\}_{\omega \in rl})$ to $\mathcal{A}$. Note that, the attribute set (id^*, t^*) does not satisfy the key policy $\mathbb{A}$, as id^* must be revoked before time t^*. So, the OSW challenger never rejects to answer queries on the above key policy.

Recall that $\lambda_t = \alpha - \sum_{\omega \in rl} \lambda_\omega \pmod p$. So, E_t is distributed identically to that in the actual scheme for the same system parameter. Clearly, the other values are also well defined.

Challenge. For two equal-length messages M_0 and M_1, the simulator passes them on to the OSW challenger and receives a challenge ciphertext $C_{OSW} = (C^{(1)}, C^{(2)}, \{C_x^{(3)}, C_x^{(4)}\}_{x \in \{id^*, t^*\}})$. According to the OSW-KP-ABE scheme, the ciphertext C_{OSW} should have the following form: $C^{(1)} = M_b e(g_1, g_2)^s$, $C^{(2)} = g^s$, $C_x^{(3)} = T(x)^s$ and $C_x^{(4)} = V(x)^s$ for $x \in \{id^*, t^*\}$, for some random bit $b \in \{0, 1\}$ and $s \in \mathbb{Z}_p$. So, the OSW ciphertext has the same distribution as in our scheme. The simulator sends $C_{id^*, t^*} := C_{OSW}$ to $\mathcal{A}$.

Phase 2. The simulator acts exactly as it did in Phase 1.

Guess. Eventually, $\mathcal{A}$ will output a bit b' as the guess of b. The simulator sends the b' to the OSW challenger as its own guess.

By the above discussion, the simulated environment is identical to that in the actual sID-CPA game. So, if $\mathcal{A}$'s advantage in this game is ϵ, so does the simulator in breaking the selective-set security of the OSW KP-ABE scheme.

Proof for Type II Adversary. The proof in this case is identical to that in the previous case, except for the following modifications: In **Setup**, the simulator chooses a random term $\alpha_1 \in \mathbb{Z}_p$ rather than α_2, and implicitly sets $\alpha_2 = \alpha - \alpha_1 \pmod p$. In **Phase 1**, the simulator answers all queries issued by $\mathcal{A}$ using α_1, except for the queries on the private keys and the short-term decryption keys. For the private key generation query on identity id ($\neq id^*$), the simulator sends the key policy $\mathbb{A} = id$ to the OSW challenger and obtains a secret key $D = (g_2^{\alpha} T(id)^{s_{id}}, g^{s_{id}})$. The simulator sets $sk_{id} = (g_2^{\alpha} T(id)^{s_{id}} \cdot g_2^{-\alpha_1}, g^{s_{id}})$ and returns it to $\mathcal{A}$. For the decryption key generation query on (id, t) ($\neq (id^*, t^*)$), the simulator sends the key policy $\mathbb{A} = id \wedge t$ to the OSW challenger and receives a secret key $D = (D_{id}, D_t)$. Here, $D_{id} = (g_2^{\lambda_{id}} T(id)^{r_{id}}, g^{r_{id}})$, $D_t = (g_2^{\lambda_t} T(t)^{r_t}, g^{r_t})$, and $\lambda_{id} + \lambda_t = \alpha \pmod p$. The simulator sets

$$\begin{aligned} D_{id,t}^{(1)} &= g_2^{\lambda_{id}} T(id)^{r_{id}} \cdot g_2^{\lambda_t} T(t)^{r_t} \cdot g_2^{-\alpha_1} \\ D_{id,t}^{(2)} &= g^{r_{id}}, D_{id,t}^{(3)} = g^{r_t}. \end{aligned}$$

The simulator returns $dk_{id,t} = (D_{id,t}^{(1)}, D_{id,t}^{(2)}, D_{id,t}^{(3)})$ to $\mathcal{A}$.

Observe that, the first element of the secret key sk_{id} can be written as $g_2^{\alpha_2} T(id)^{s_{id}}$ and the first element of the decryption key $dk_{id,t}$ can be written as $g_2^{\alpha_2} T(id)^{r_{id}} T(t)^{r_t}$. So, these values are distributed identically to their distributions in the actual scheme. Therefore, in the case where $\mathcal{A}$ is a Type II adversary, the simulator has the same advantage, i.e., ϵ, to break the security of the underlying OSW scheme.

This completes the proof of Theorem 1.

References

1 Qin, B D, Deng, R H, Li, Y J, et al. Server-aided revocable identity-based encryption. In: Proceedings of 20th European Symposium on Research in Computer Security, Vienna, 2015. 286–304