SCIENCE CHINA
Information Sciences

• HIGHLIGHT •

May 2018, Vol. 61 058101:1–058101:3
doi: 10.1007/s11432-017-9120-4

LR-RRA-CCA secure functional encryption for randomized functionalities from trapdoor HPS and LAF

Huige WANG[1,2], Kefei CHEN[3,6*], Baodong QIN[4,5] & Ziyuan HU[1]

[1]*Department of Network Engineering, Anhui Science and Technology University, Fengyang* 233100*, China;*
[2]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai* 200240*, China;*
[3]*Department of Mathematics, Hangzhou Normal University, Hangzhou* 311121*, China;*
[4]*National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, Xi'an* 710121*, China;*
[5]*State Key Laboratory of Cryptology, P.O.Box* 5159*, Beijing* 100878*, China;*
[6]*Westone Cryptologic Research Center, Beijing* 100070*, China*

Received 16 May 2017/Accepted 31 May 2017/Published online 30 August 2017

Citation Wang H G, Chen K F, Qin B D, et al. LR-RRA-CCA secure functional encryption for randomized functionalities from trapdoor HPS and LAF. Sci China Inf Sci, 2018, 61(5): 058101, doi: 10.1007/s11432-017-9120-4

Original public-key encryption is viewed as a tool to encrypt point-to-point communication. Encrypted data is aimed at a particular user, where a sender encrypts a message x under a specified public key PK. In this case, only the owner of the (unique) secret key corresponding to PK can decrypt the resulting ciphertext to recover the message x. However, on some occasions, encryption is necessary to implement more fine-grained control of the encrypted data. For example, the sender may hope that the receiver learns only partial information about the encrypted data rather than the full information. To achieve this goal, the notion of functional encryption is proposed and is further developed [1]. At a high level, in a functional encryption for some class $\mathcal{F}$, a secret key sk_f for any function $f \in \mathcal{F}$ can be derived from the master secret key msk. Given a ciphertext ct that encrypts message x, the user can decrypt ct to obtain $f(x)$ under the secret key sk_f but should learn nothing else about x beyond $f(x)$.

In the past few years, remarkable work in functional encryption was accomplished by exploring different security models [2] and constructions under different assumptions [3]. Recently, several studies began to focus on researching the classes of functions that can be supported by functional encryption. In function-oriented research, nearly all work is dedicated to deterministic functions. By contrast, for some special occasions, randomized functions [4] are urgently necessary to implement more functionalities.

In 2015, Goyal et al. [4] proposed a construction of functional encryption for randomized functionalities that were secure against CCA (chosen-ciphertext attacks). Subsequently, several different types of functional encryptions were proposed [5]. In this work, we provide a simulation-based LR-RRA-CCA (leakage-resilient, related-randomness attacks, chosen-ciphertext attacks) security definition of FE (functional encryption) for randomized functionalities and propose a construction consisting of a trapdoor hash-proof system and lossy algebraic filter. This is proven to achieve chosen-ciphertext security, leakage-resilient security, and related-randomness security simultaneously under the assumptions of different-input obfus-

* Corresponding author (email: kfchen@hznu.edu.cn)
The authors declare that they have no conflict of interest.

cation and a puncturable pseudorandom function. Our starting point is the LR-CCA secure public-key encryption scheme of Qin et al. [6]. We subsequently develop our proposed scheme by modifying this scheme step by step.

In the following, we explain some core assumptions used in the construction of our scheme.

Definition 1 (Differing-inputs circuits)**.** A sample algorithm $(C_0, C_1, z) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$ that samples circuits from a circuit ensemble $C = \{C_\lambda\}_{\lambda\in\mathbb{N}}$ is said to be a differing-inputs distribution, if for all PPT algorithms $\mathcal{A}$, there is a negligible function *negl* such that $\mathsf{Pr}[C_0(x) \neq C_1(x)] \leqslant negl(\lambda)$, where $(C_0, C_1, z) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$, $x \leftarrow_{\$} \mathcal{A}(1^\lambda, C_0, C_1, z)$.

Definition 2 (Differing-inputs obfuscation)**.** A PPT algorithm DIO is a differing-inputs obfuscator for a differing-inputs distribution Sam (for circuit ensemble $\{C_\lambda\}_{\lambda\in\mathbb{N}}$) if it satisfies: (1) Correctness. For all $\lambda \in \mathbb{N}$, $C \in C_\lambda$ and x, we have $\mathsf{Pr}[C(x) = C'(x) : C' \leftarrow_{\$} \mathsf{DIO}(1^\lambda, C)] = 1$. (2) Security. For any PPT distinguisher $\mathcal{D}$ and $(C_0, C_1, z) \leftarrow_{\$} \mathsf{Sam}(1^\lambda)$, there is a negligible function *negl* such that $\mathsf{Adv}^{\mathsf{dio}}_{C_0,C_1,\mathcal{A}}(\lambda) = \mathsf{Pr}[\mathcal{D}(1^\lambda, \mathsf{DIO}(1^\lambda, C_0), z)] - \mathsf{Pr}[\mathcal{D}(1^\lambda, \mathsf{DIO}(1^\lambda, C_1), z)] \leqslant negl(\lambda)$.

Definition 3. A puncturable pseudorandom function PF=($\mathsf{PF.K}$, $\mathsf{PF.Eval}$, $\mathsf{PF.Punc}$) over $\mathcal{D}_\lambda$ and $\mathcal{R}_\lambda$ is defined as follows.

• The key generation algorithm. $K \leftarrow_{\$} \mathsf{PF.K}(1^\lambda)$, where $K \leftarrow_{\$} \mathcal{R}_\lambda$.

• The punctured key generation algorithm. $K_S \leftarrow_{\$} \mathsf{PF.Punc}(K, S)$, where $K_S \in \mathcal{R}_\lambda$.

• The evaluation algorithm. $y \leftarrow_{\$} \mathsf{PF.Eval}(K, x)$, where $K \in \mathcal{R}_\lambda$ (punctured key or PRF key), $x \in \mathcal{D}_\lambda$, and $y \in \mathcal{R}_\lambda$.

Definition 4 (Security)**.** PF=($\mathsf{PF.K}$, $\mathsf{PF.Eval}$, $\mathsf{PF.Punc}$) is secure if it satisfies as follows.

(1) Functionality preserved under puncturing. For all $x \notin S$, we have $\mathsf{PF.Eval}(K, x) = \mathsf{PF.Eval}(K_S, x)$, where $K \leftarrow \mathsf{PF.K}(1^\lambda)$, $K_S \leftarrow \mathsf{PF.Punc}(K, S)$.

(2) Pseudorandomness preserved at punctured points. For any PPT adversary $\mathcal{A}' = (\mathcal{A}'_1, \mathcal{A}'_2)$ such that $\mathcal{A}'_1(1^\lambda)$ defines a set $S \subset \mathcal{D}_\lambda$, for all $x \in S$, $K \leftarrow \mathsf{PF.K}(1^\lambda)$, $K_S \leftarrow \mathsf{PF.Punc}(K, S)$ and $y \leftarrow_{\$} \mathcal{R}_\lambda$, then there exists a negligible function *negl* such that

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{PF},\mathcal{A}'}(\lambda) = \Pr[\mathcal{A}'_2(K_S, x, \mathsf{PF.Eval}(K, x)) = 1] - \Pr[\mathcal{A}'_2(K_S, x, y) = 1] \leqslant negl(\lambda).$$

(3) Pseudorandomness at non-punctured points. For any PPT adversary $\mathcal{A}$, all $K \leftarrow \mathsf{PF.K}(1^\lambda)$, $x \in \mathcal{D}_\lambda$, then

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{PF},\mathcal{A}} = \mathsf{Pr}[\mathcal{A}(1^\lambda, y) = 1 : y = \mathsf{PF}_K(x)] - \mathsf{Pr}[\mathcal{A}(1^\lambda, y) = 1 : y \leftarrow_{\$} \mathcal{R}_\lambda] \leqslant negl(\lambda).$$

Lemma 1. Let X, Y, and Z be random variables. If Y has at most 2^l possible values, then $\widetilde{\mathrm{H}}_\infty(X|(Y, Z)) \geqslant \widetilde{\mathrm{H}}_\infty(X|Z) - l$.

Definition 5 (Randomness extractor)**.** An efficient function $\mathsf{Ext} : \mathcal{X} \times \mathcal{S} \to \mathcal{Y}$ is an average-case (ν, ϵ)-strong extractor if for all pairs of random variables (X, Z) such that $X \in \mathcal{X}$ and $\widetilde{\mathrm{H}}_\infty(X|Z) \geqslant \nu$, we have

$$\mathsf{SD}((Z, s, \mathsf{Ext}(X, s)), (Z, s, U_\mathcal{Y})) \leqslant \epsilon_2.$$

$s \leftarrow_{\$} \mathcal{S}$ and $U_\mathcal{Y}$ is uniform over $\mathcal{Y}$.

Definition 6 (Correlated-input secure (CIS) Hash functions)**.** A CIS hash function h_κ is a PPT algorithm that is defined as $y \leftarrow_{\$} \mathsf{h}_\kappa(x)$. The security requires that even given the hash values of multiple correlated-input values $\mathsf{h}_k(\phi_1(x)), \ldots, \mathsf{h}_k(\phi_q(x))$, the hash value y is still pseudorandom, where the correlation functions $\phi_1, \ldots, \phi_q$ may be maliciously chosen and $q = q(\lambda)$ is a polynomial in λ. The advantage function is defined as $\mathsf{Adv}^{l\text{-mk-sci-pr}}_{\mathcal{H},\mathcal{A}}(\lambda)$.

Definition 7 (Trapdoor Hash proof system (THPS))**.** A trapdoor hash proof system consists of four PPT algorithms ($\mathsf{THPS.Gen}$, $\mathsf{THPS.Pub}$, $\mathsf{THPS.Priv}$, $\mathsf{THPS.Invert}$).

• The parameter generation algorithm. $(pp, td) \leftarrow \$\mathsf{THPS.Gen}(1^\lambda)$, where pp = ($\mathsf{Description}$, $\mathcal{PK}$, $\mathcal{SK}$, $\mathcal{K}$, $\Lambda_{(\cdot)} : \mathcal{C} \to \mathcal{K}$, $\mu : \mathcal{SK} \to \mathcal{PK}$) and $td \in \mathcal{TK}$ is trapdoor.

• The public evaluation algorithm. $K = \Lambda_{sk}(C) \leftarrow \$\mathsf{THPS.Pub}(pk, C, w)$, where $pk = \mu(sk)$, $C \in \mathcal{V}$ and w indicate $C \in \mathcal{V}$.

• The private evaluation algorithm. $K = \Lambda_{sk}(C) \leftarrow \$\mathsf{THPS.Priv}(sk, C)$.

• The inversion algorithm. $w \leftarrow \$\mathsf{THPS.Invert}(td, C)$, where $C \in \mathcal{V}$ and $\mathsf{THPS.Pub}(pk, C, w) = \mathsf{THPS.Priv}(sk, C)$ holds with probability 1, but for $C \in \mathcal{C}\backslash\mathcal{V}$, $\mathsf{THPS.Invert}(td, C)$ outputs $\perp$.

Furthermore, we require that the THPS must satisfy ϵ_1-**universal** and a subset membership problem [6].

Definition 8 (Lossy algebraic filter [7])**.** An (l_{LAF}, n') lossy algebraic filter (LAF) consists of three PPT algorithms LAF = ($\mathsf{LAF.KG}$, $\mathsf{LAF.Eval}$, $\mathsf{LAF.Ltag}$):

• Key Generation. $(lpk, ltk) \leftarrow_{\$} \mathsf{LAF.KG}(1^\lambda)$, where lpk is the public key, ltk is a trapdoor that will allow us to compute a lossy tag.

• Evaluation. The algorithm $\mathsf{LAF}_{lpk,t}(X) \leftarrow_{\$} \mathsf{LAF.Eval}(lpk, t, X)$, where $X \in \mathbb{Z}_p^{n'}$, $t = (t_a, t_c) \in \mathcal{T}$.

• Lossy Tag Generation. The algorithm $t_c \leftarrow_{\$} \mathsf{LAF.Ltag}(ltk, t_a)$, where $t = (t_a, t_c)$ is lossy.

• Lossiness. If t is injective, the function $\mathsf{LAF}_{lpk,t}(X)$ is injective. If t is lossy, then $\mathsf{LAF}_{lpk,t}(X)$ only depends on $\sum_{i=1}^{n'} \omega_i X_i \bmod p$ for $\omega_i \in \mathbb{Z}_p$.

Furthermore, an LAF also satisfies the indistinguishability and evasiveness securities. Their advantage functions are $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LAF},\mathcal{A}}(\lambda)$ and $\mathsf{Adv}^{\mathsf{evs}}_{\mathsf{LAF},\mathcal{A}}$ respectively.

Construction. We construct a simulation-based LR-RRA-CCA secure public-key functional encryption scheme rFE=($\mathsf{rFE.Setup}$, $\mathsf{rFE.KG}$, $\mathsf{rFE.E}$, $\mathsf{rFE.D}$) for randomized function family $\mathcal{F}$. Our construction needs the following building blocks.

(1) An *l-mk-sci-pr* secure keyed hash function description ($\mathsf{CIH.K}, \mathsf{CIH.D}, \mathsf{CIH.R}, \mathsf{h}$); (2) A ϵ_1-universal trapdoor hash proof system THPS = ($\mathsf{THPS.Gen}$, $\mathsf{THPS.Pub}$, $\mathsf{THPS.Priv}$, $\mathsf{THPS.Invert}$); (3) An (l_{LAF}, n') lossy algebraic filter LAF = ($\mathsf{LAF.KG}$, $\mathsf{LAF.Eval}$, $\mathsf{LAF.Ltag}$); (4) An average-case $((\nu-(q_{lr}-1).m-q_{lr}.l_{\mathsf{LAF}}-l_L), \epsilon_2)$-strong extractor $\mathsf{Ext}: \mathcal{K}\times\{0,1\}^d \to \{0,1\}^m$; (5) A secure differing-inputs obfuscator DIO.

The concrete construction is described as follows.

• Setup. The setup algorithm $\mathsf{rFE.Setup}(1^\lambda)$ on input 1^λ. It first runs $\mathsf{THPS.Gen}(1^\lambda)$ to generate a public parameter pp and the trapdoor td, then generates the key pair $(lpk, ltk) \leftarrow_{\$} \mathsf{LAF.KG}(1^\lambda)$. Next, it picks $k \leftarrow_{\$} \mathsf{CIH}.\mathcal{K}(1^\lambda)$, $sk \leftarrow_{\$} \mathcal{SK}$, and sets $pk = \mu(sk)$. Finally, the master public key is $mpk = (pp, lpk, k, pk)$, and the master secret key is $msk = sk$.

• Encryption. The encryption algorithm $\mathsf{rFE.E}(mpk, x)$ on input a master public key mpk and a message $x \in \{0,1\}^m$. It first chooses $r \leftarrow_{\$} \mathcal{R}_\lambda$, and then samples $(C, s, t_c) \leftarrow Sample_{\mathcal{V}\times\{0,1\}^d\times\mathcal{T}_c}(\mathsf{PF}_{\mathsf{h}_k(r)}(mpk||x))$, where C has a witness w, s is a random seed, and t_c is a random core tag. Next, it computes $K \leftarrow \mathsf{THPS.Pub}(pk, C, w)$, $U = \mathsf{Ext}(K, s)\oplus x$, $\pi = \mathsf{LAF}_{lpk,t}(K)$, where $t = (t_a, t_c)$ with $t_a = (C, s, U)$. Finally the ciphertext is set as $ct = (C, s, U, \pi, t_c)$.

• Key generation. The key generation algorithm $\mathsf{rFE.KG}(msk, f)$ takes as its input a master secret key msk and a function $f \in \mathcal{F}$. It chooses a random number $r \leftarrow_{\$} \mathcal{R}_\lambda$, then computes $r' = \mathsf{PF}_{\mathsf{h}_k(r)}(mpk||f)$. Finally, the secret key is set as $sk_f = \mathsf{DIO}(\mathcal{G}_{[mpk,msk,r',f]})$, where the function $\mathcal{G}_{[mpk,msk,r',f]}$ is described in Figure 1. Note that $\mathcal{G}_{[mpk,msk,r',f]}$ has the master public key mpk, the master secret key msk, the random number r', and the function f hardcoded in it.

• Decryption. With input ct, the decryption algorithm $\mathsf{rFE.D}(sk_f, ct)$ computes and outputs $y = sk_f(ct)$. Note that $sk_f = \mathsf{DIO}(\mathcal{G}_{[mpk,msk,r',f]})$ and the circuit $\mathcal{G}_{[mpk,msk,r',f]}$ are described in Figure 1).

Constants : mpk, msk, r', f
Input : ct
1. Parse ct into C, s, U, π, t_c and parse mpk into pp, lpk, k, pk.
2. Compute $K = \mathsf{THPS.Priv}(sk, C)$, where $msk = sk$.
3. Compute $\pi' = \mathsf{LAF}_{lpk,t}(K)$, where $t = (t_a, t_c)$, $t_a = (C, s, U)$.
4. If $\pi' \neq \pi$, output $\perp$, else proceed the following steps.
5. Compute $x = \mathsf{Ext}(K, s) \oplus U$.
6. Compute $r'' = \mathsf{PF.Eval}(r', ct)$.
7. Compute $y = f(x; r'')$ and output y.

Figure 1 Functionality $\mathcal{G}_{[mpk,msk,r',f]}$.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61472114, 61572318, 61702007, 61133014, 61502400), National Key Research and Development Program of China (Grant No. 2017YFB0802003), and other foundations (Grant Nos. 16ZB0140, LD14127X, ZRC2013380).

Supporting information The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Boneh D, Sahai A, Waters B, et al. Functional encryption: definitions and challenges. In: Proceedings of the 8th Conference on Theory of Cryptography (TCC'11). Berlin: Springer-Verlag, 2011. 253–273
2 Agrawal S, Gorbunov S, Vaikuntanathan V, et al. Cryptographic agents: towards a unified theory of computing on encrypted data. In: Advances in Cryptology — EUROCRYPT 2015. Berlin: Springer-Verlag, 2015. 501–531
3 Ananth P, Brakerski Z, Segev G, et al. From selective to adaptive security in functional encryption. In: Advances in Cryptology — CRYPTO 2015. Berlin: Springer-Verlag, 2015. 657–677
4 Goyal V, Jain A, Koppula V, et al. Functional encryption for randomized functionalities. In: Theory of Cryptography. Berlin: Springer-Verlag, 2015. 325–351
5 Agrawal S, Wu D J. Functional encryption: deterministic to randomized functions from simple assumptions. In: Advances in Cryptology — EUROCRYPT 2017. Berlin: Springer-Verlag, 2017. 30–61
6 Qin B D, Liu S L. Leakage-resilient chosen-ciphertext secure public-key encryption from Hash proof system and one-time lossy filter. In: Advances in Cryptology — ASIACRYPT 2013. Berlin: Springer-Verlag, 2013. 381–400
7 Hofheinz H. Circular chosen-ciphertext security with compact ciphertexts. In: Advances in Cryptology — EUROCRYPT 2013. Berlin: Springer-Verlag, 2013. 520–536