An RSA Family of Trap-door Permutations with a Common Domain and its Applications

Ryotaro Hayashi[1], Tatsuaki Okamoto[2], and Keisuke Tanaka[1] *

[1] Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, Japan
{hayashi9, keisuke}@is.titech.ac.jp
[2] NTT Labs, 1-1 Hikarino-oka, Yokosuka-shi, Kanagawa 239-0847, Japan
okamoto@isl.ntt.co.jp

Abstract. Bellare, Boldyreva, Desai, and Pointcheval [1] recently proposed a new security requirement of the encryption schemes called "key-privacy." It asks that the encryption provide (in addition to privacy of the data being encrypted) privacy of the key under which the encryption was performed. Incidentally, Rivest, Shamir, and Tauman [2] recently proposed the notion of ring signature, which allows a member of an ad hoc collection of users S to prove that a message is authenticated by a member of S without revealing which member actually produced the signature.
We are concerned with an underlying primitive element common to the key-privacy encryption and the ring signature schemes, that is, families of trap-door permutations with a common domain. For a standard RSA family of trap-door permutations, even if all of the functions in a family use RSA moduli of the same size (the same number of bits), it will have domains with different sizes. In this paper, we construct an RSA family of trap-door permutations with a common domain, and propose the applications of our construction to the key-privacy encryption and ring signature schemes, which have some advantage to the previous schemes.

Keywords: RSA, trap-door permutations, key-privacy, anonymity, encryption, ring signature

1 Introduction

Bellare, Boldyreva, Desai, and Pointcheval [1] recently proposed a new security requirement of the encryption schemes called "key-privacy." It asks that the encryption provide (in addition to privacy of the data being encrypted) privacy of the key under which the encryption was performed. The standard RSA encryption does not provide key-privacy. Since even if two public keys N_0 and N_1 ($N_0 < N_1$) are the same bits, $N_1 - N_0$ may be large. In [1], they provided the

* Supported in part by NTT Information Sharing Platform Laboratories and Grant-in-Aid for Scientific Research, Ministry of Education, Culture, Sports, Science, and Technology, 14780190.

key-privacy encryption scheme, RSA-RAEP, which is a variant of RSA-OAEP (Bellare and Rogaway [3], Fujisaki, Okamoto, Pointcheval, and Stern [4]), and solved this problem by repeating the evaluation of the RSA-OAEP permutation $f(x, r)$ with plaintext x and random r, each time using different r until the value is in the safe range (See section 3.2.). For deriving a value in the safe range, the number of the repetition would be very large (the value of the security parameter).

Incidentally, Rivest, Shamir, and Tauman [2] recently proposed the notion of ring signature, which allows a member of an ad hoc collection of users S to prove that a message is authenticated by a member of S without revealing which member actually produced the signature. Unlike group signature, ring signature has no group managers, no setup procedures, no revocation procedures, and no coordination. The signer does not need the knowledge, consent, or assistance of the other ring members to put them in the ring. All the signer needs is knowledge of their regular public keys. They also proposed the efficient schemes based on RSA and Rabin. In their RSA-based scheme, the trap-door RSA permutations of the various ring members will have domains of different sizes. This makes it awkward to combine the individual signatures, so one should construct some trap-door one-way permutation which has a common domain for each user. Intuitively, in the ring signature scheme, Rivest, Shamir, and Tauman solved this by encoding the message to an N_i-ary representation and applying a standard permutation f to the low-order digits (See section 4.2.). As mentioned in [2], for deriving a secure permutation g with a common domain, the domain of g would be 160 bits larger than that of f.

In this paper, we will take a different approach. We use neither the repetition of evaluation of a permutation nor an N_i-ary representation. We are concerned with an underlying primitive element common to the key-privacy encryption and the ring signature schemes, that is, families of trap-door permutations with a common domain. For a standard RSA family of trap-door permutations denoted by RSA, even if all of the functions in a family use RSA moduli of the same size (the same number of bits), it will have domains with different sizes. We construct an RSA family of trap-door permutations with a common domain denoted by RSACD, and prove that the θ-partial one-wayness of RSACD is equivalent to the one-wayness of RSACD for $\theta > 0.5$, and that the one-wayness of RSACD is equivalent to the one-wayness of RSA. Fujisaki, Okamoto, Pointcheval, and Stern [4] showed that the θ-partial one-wayness of RSA is equivalent to the one-wayness of RSA for $\theta > 0.5$. Thus, the following relations are satisfied for $\theta > 0.5$.

$$\begin{tikzcd}
\mathsf{RSA}\text{ is }\theta\text{-partial one-way} \arrow[d, shift right] & & \mathsf{RSACD}\text{ is }\theta\text{-partial one-way} \arrow[d, shift right] \\
\mathsf{RSA}\text{ is one-way} \arrow[u, shift right, "\text{[4]}"'] \arrow[rr, shift left, "\text{[this paper]}"] & & \mathsf{RSACD}\text{ is one-way} \arrow[ll, shift left] \arrow[u, shift right, "\text{[this paper]}"']
\end{tikzcd}$$

We then propose the application to the key-privacy encryption scheme. Our proposed scheme is more efficient than the previous scheme with respect to the number of exponentiations for encryption in the worst case. When we use the RSA moduli which is uniformly distributed in $(2^{k-1}, 2^k)$, the expected number of our scheme is the same as that of RSA-RAEP. In our scheme, the number of exponentiations for encryption is at most two, while in RSA-RAEP, the upper bound of this number is k_1 ($\gg 2$, security parameter).

We also propose the application to the ring signature scheme. We consider the case that the members of the same group use the RSA moduli of the same length. In our scheme, the domain of trap-door one-way permutation to sign and verify a ring signature is $\{0,1\}^k$, while that of the previous scheme is $\{0,1\}^{k+160}$, where k is the length of the RSA moduli. Thus, we can reduce the size of signature in this situation.

The organization of this paper is as follows. In Section 2, after reviewing the definitions of families of functions and the standard RSA family, we propose the RSA family of trap-door permutations with a common domain. We also prove that the θ-partial one-wayness of RSACD is equivalent to the one-wayness of RSACD for $\theta > 0.5$, and that the one-wayness of RSACD is equivalent to the one-wayness of RSA. In Section 3, we propose the application of our new family to the key-privacy encryption scheme. In Section 4, we propose the application of our new family to the ring signature scheme. We conclude in Section 5.

2 An RSA Family of Trap-door Permutations with a Common Domain

2.1 Preliminaries

In this section, we briefly review the definitions of families of functions, and the standard RSA family of trap-door permutations denoted by RSA.

Definition 1 (families of functions [1]). *A family of functions $F = (K, S, E)$ is specified by three algorithms.*

- *The randomized key-generation algorithm K takes as input a security parameter $k \in \mathbb{N}$ and returns a pair (pk, sk) where pk is a public key and sk is an associated secret key. (In cases where the family is not trap-door, the secret key is simply the empty string.)*
- *The randomized sampling algorithm S takes input pk and returns a random point in a set that we call the domain of pk and denote by $\mathrm{Dom}_F(pk)$.*
- *The deterministic evaluation algorithm E takes input pk and a point $x \in \mathrm{Dom}_F(pk)$ and returns an output we denote by $E_{pk}(x)$. We let $\mathrm{Rng}_F(pk) = \{E_{pk}(x) \mid x \in \mathrm{Dom}_F(pk)\}$ denote the range of the function $E_{pk}(\cdot)$.*

Definition 2 (families of trap-door permutations [1]). *We say that F is a family of trap-door functions if there exists a deterministic inversion algorithm I that takes input sk and a point $y \in \mathrm{Rng}_F(pk)$ and returns a point $x \in \mathrm{Dom}_F(pk)$ such that $E_{pk}(x) = y$. We say that F is a family of trap-door permutations if F is a family of trap-door functions, $\mathrm{Dom}_F(pk) = \mathrm{Rng}_F(pk)$, and E_{pk} is a permutation on this set.*

We describe the definition of θ-partial one-way.

Definition 3 (θ-partial one-way [1]). *Let $F = (K, S, E)$ be a family of functions. Let $b \in \{0,1\}$ and $k \in \mathbb{N}$ be a security parameter. Let $0 < \theta \leq 1$ be a constant. Let A be an adversary. Now, we consider the following experiments:*

```
Experiment $\mathbf{Exp}^{\theta-\mathrm{pow-fnc}}_{F,A}(k)$
    $(pk, sk) \overset{R}{\leftarrow} K(k)$
    $x_1||x_2 \overset{R}{\leftarrow} \mathrm{Dom}_F(pk)$ where $|x_1| = \lceil \theta \cdot |(x_1||x_2)| \rceil$
    $y \leftarrow E_{pk}(x_1||x_2)$
    $x'_1 \leftarrow A(pk, y)$ where $|x'_1| = |x_1|$
    for any $x'_2$ if $E_{pk}(x'_1||x'_2) = y$ then return $1$
    else return $0$
```

We define the advantages of the adversary via

$$\mathbf{Adv}^{\theta-\mathrm{pow-fnc}}_{F,A}(k) = \Pr[\mathbf{Exp}^{\theta-\mathrm{pow-fnc}}_{F,A}(k) = 1]$$

where the probability is taken over $(pk, sk) \overset{R}{\leftarrow} K(k)$, $x_1||x_2 \overset{R}{\leftarrow} \mathrm{Dom}_F(pk)$, and the coin tosses of A. We say that the family F is θ-partial one-way if the function $\mathbf{Adv}^{\theta-\mathrm{pow-fnc}}_{F,A}(\cdot)$ is negligible for any adversary A whose time complexity is polynomial in k. In particular, we say that the family F is one-way when F is 1-partial one-way.

We now describe the standard RSA family of trap-door permutations.

Definition 4 (the standard RSA family of trap-door permutations [1]). *The specifications of the standard RSA family of trap-door permutations $\mathsf{RSA} = (K, S, E)$ are as follows. The key generation algorithm takes as input a security parameter k and picks random, distinct primes p, q in the range $2^{k/2-1} < p, q < 2^{k/2}$. (If k is odd, increment it by 1 before picking the primes.) It sets $N = pq$. (i.e. $2^{k-2} < N < 2^k$.) It picks $e, d \in \mathbb{Z}^*_{\phi(N)}$ such that $ed = 1 \pmod{\phi(N)}$ where $\phi(N) = (p-1)(q-1)$. The public key is N, e, k and the secret key is N, d, k. The sets $\mathrm{Dom}_{\mathsf{RSA}}(N, e, k)$ and $\mathrm{Rng}_{\mathsf{RSA}}(N, e, k)$ are both equal to $\mathbb{Z}^*_N$. The evaluation algorithm $E_{N,e,k}(x) = x^e \bmod N$ and the inversion algorithm $I_{N,d,k}(y) = y^d \bmod N$. The sampling algorithm returns a random point in $\mathbb{Z}^*_N$.*

Fujisaki, Okamoto, Pointcheval, and Stern [4] showed that the θ-partial one-wayness of RSA is equivalent to the one-wayness of RSA for $\theta > 0.5$.

2.2 The Construction of RSACD

In this section, we propose the RSA family of trap-door permutations with a common domain denoted by RSACD.

Definition 5 (the RSA family of trap-door permutations with a common domain). *The specifications of the RSA family of trap-door permutations*

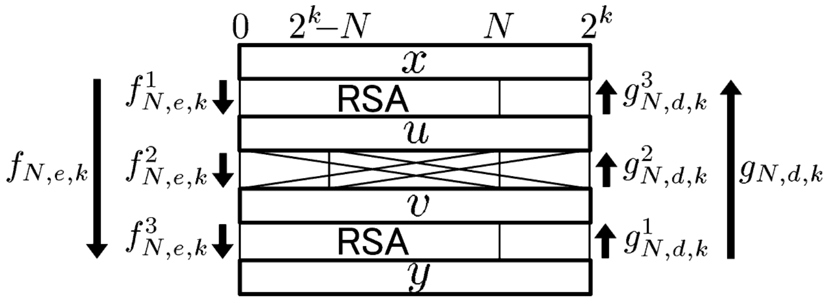

Fig. 1. Function $f_{N,e,k}$ and $g_{N,d,k}$

with a common domain RSACD= (K, S, E) *are as follows. The key generation algorithm is almost the same as that for* RSA *family. The difference is picking two distinct primes* p, q *such that* $2^{k/2-1} < p, q < 2^{k/2}$ *and* $2^{k-1} < pq < 2^k$*. The sets* $\mathrm{Dom}_{\mathsf{RSACD}}(N, e, k)$ *and* $\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$ *are both* $\{x \,|\, x \in [0, 2^k) \wedge x \bmod N \in \mathbb{Z}^*_N\}$*. The sampling algorithm returns a random point in* $\mathrm{Dom}_{\mathsf{RSACD}}(N, e, k)$*. The evaluation algorithm* $E_{N,e,k}(x) = f_{N,e,k}(x)$ *and the inversion algorithm* $I_{N,d,k}(y) = g_{N,d,k}(y)$ *are as follows (See Figure 1.).*

```
Function $f_{N,e,k}(x)$
  $u \leftarrow f^1_{N,e,k}(x)$; $v \leftarrow f^2_{N,e,k}(u)$
  $y \leftarrow f^3_{N,e,k}(v)$
  return $y$

Function $f^1_{N,e,k}(x)$
  if $(x < N)$ $u \leftarrow x^e \bmod N$
  else $u \leftarrow x$
  return $u$

Function $f^2_{N,e,k}(u)$
  if $(u < 2^k - N)$ $v \leftarrow u + N$
  elseif $(2^k - N \leq u < N)$ $v \leftarrow u$
  else $v \leftarrow u - N$
  return $v$

Function $f^3_{N,e,k}(v)$
  if $(v < N)$ $y \leftarrow v^e \bmod N$
  else $y \leftarrow v$
  return $y$
```

```
Function $g_{N,d,k}(y)$
  $v \leftarrow g^1_{N,d,k}(y)$; $u \leftarrow g^2_{N,d,k}(v)$
  $x \leftarrow g^3_{N,d,k}(u)$
  return $x$

Function $g^1_{N,d,k}(y)$
  if $(y < N)$ $v \leftarrow y^d \bmod N$
  else $v \leftarrow y$
  return $v$

Function $g^2_{N,d,k}(v)$
  if $(v < 2^k - N)$ $u \leftarrow v + N$
  elseif $(2^k - N \leq v < N)$ $u \leftarrow v$
  else $u \leftarrow v - N$
  return $u$

Function $g^3_{N,d,k}(u)$
  if $(u < N)$ $x \leftarrow u^d \bmod N$
  else $x \leftarrow u$
  return $x$
```

The choice of N from $(2^{k-1}, 2^k)$ ensures that all elements in $\mathrm{Dom}_{\mathsf{RSACD}}(N, e, k)$ are permuted by the RSA function at least once.

2.3 Properties of RSACD

In this section, we prove that the θ-partial one-wayness of RSACD is equivalent to the one-wayness of RSACD for $\theta > 0.5$, and that the one-wayness of RSACD is equivalent to the one-wayness of RSA.

Theorem 1. *Let A be an algorithm that outputs the $k - k_0$ most significant bits of the pre-image of its input $y \in \mathsf{Rng}_{\mathsf{RSACD}}(N, e, k)$ for $2^{k-1} < N < 2^k$ with $k > 2k_0$ (i.e. A is a $((k - k_0)/k)$-partial inverting algorithm for RSACD with $k > 2k_0$), with success probability $\epsilon = \mathbf{Adv}^{\theta-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSACD},A}(k)$ where $\theta = (k - k_0)/k > 0.5$, within time bound t. There exists an algorithm B that outputs a pre-image of y (i.e. B is an inverting algorithm for RSACD) with success probability $\epsilon' = \mathbf{Adv}^{1-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSACD},B}(k)$, within time bound t' where*

$$\epsilon' \geq \frac{\epsilon^2}{16} \cdot (1 - 2^{2k_0-k+7}), \quad t' \leq 2t + O(k^3).$$

To prove this theorem, we use the following lemma proved in [4].

Lemma 1 ([4]). *Consider an equation $\alpha t + u = c \pmod N$ which has solutions t and u smaller than 2^{k_0}. For all values of α, except a fraction $2^{2k_0+6}/N$ of them, (t, u) is unique and can be computed in time $O((\log N)^3)$. (We say "α is a good value" when we can solve the above equation.)*

Proof (Theorem 1). We construct the algorithm B to compute a pre-image of $y \in \mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$, then we analyze this algorithm and evaluate the success probability and the running time of B.

```
Algorithm $B((N, e, k), y)$
  $\alpha \xleftarrow{R} \mathbb{Z}_N$; pow $\xleftarrow{R} \{1, 2\}$; $c \xleftarrow{R} \{0, 1\}$          ⎫
  $y'_{temp} \leftarrow y \cdot \alpha^{e^{\mathrm{pow}}} \bmod N$                                    ⎪ [step 1]
  if $(c = 0)$ $y' \leftarrow y'_{temp}$                                                            ⎬ set $\alpha$, pow, $y'$
  elseif $(0 \leq y'_{temp} < 2^k - N)$ $y' \leftarrow y'_{temp} + N$                              ⎪
  else return fail                                                                                  ⎭
  $z \leftarrow A(y)$; $z' \leftarrow A(y')$                                                        } [step 2] run $A$
  find $(r, s)$ s.t. $\alpha r - s = (z' - z\alpha) \cdot 2^{k_0} \pmod N$                          ⎫ [step 3]
  $x \leftarrow z \cdot 2^{k_0} + r$                                                                ⎭ compute $g_{N,d,k}(y)$
  return $x$
```

Analysis

For $y \in \mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$ and $x = g_{N,d,k}(y)$, (x, y) satisfies one of the following equations.

$$(1) \quad y = x^e \pmod N \qquad\qquad (2) \quad y = x^{e^2} \pmod N$$

We say *type*$(y) = 1$ (respectively *type*$(y) = 2$) if (x, y) satisfies equation 1 (resp. equation 2).

After step 1, if B does not output `fail`, then y' is uniformly distributed over $\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$, and for y' and $x' = g_{N,d,k}(y')$, (x', y') satisfies one of the following equations.

$$(1') \quad y' = (x')^e \pmod N \qquad\qquad (2') \quad y' = (x')^{e^2} \pmod N$$

We say $\mathit{type}(y') = 1$ (respectively $\mathit{type}(y') = 2$) if (x', y') satisfies equation $1'$ (resp. equation $2'$).

After step 2, if A outputs correctly, namely, z is the $k - k_0$ most significant bits of x and z' is the $k - k_0$ most significant bits of x', then $x = z \cdot 2^{k_0} + r$ and $x' = z' \cdot 2^{k_0} + s$ for some (r, s) where $0 \le r, s < 2^{k_0}$. Furthermore, if $\mathit{type}(y) = \mathit{type}(y') = \text{pow}$, then $y = x^{e^{\text{pow}}} \pmod N$ and $y' = (x')^{e^{\text{pow}}} \pmod N$. Since $y' = y \cdot \alpha^{e^{\text{pow}}} \pmod N$ and $\gcd(e^{\text{pow}}, N) = 1$, we have $x' = \alpha x \pmod N$. Thus,

$$\begin{aligned} z' \cdot 2^{k_0} + s &= \alpha \cdot (z \cdot 2^{k_0} + r) \pmod N \\ \alpha r - s &= (z' - z\alpha) \cdot 2^{k_0} \pmod N \end{aligned}$$

where $0 \le r, s < 2^{k_0}$. If α is a good value, algorithm B can solve this equation in step 3 (Lemma 1), and outputs $x = z \cdot 2^{k_0} + r$.

Now, we analyze the success probability. We define the following events:

- Fail : B outputs `fail` in step 1,
- GV : α is a good value,
- $\mathsf{Type1}$: $\mathit{type}(y) = \mathit{type}(y') = 1$,
- $\mathsf{Type2}$: $\mathit{type}(y) = \mathit{type}(y') = 2$,
- SucA : $A(y)$ and $A(y')$ are correct.

We have $\epsilon = \Pr[A(y) \text{ is correct} \wedge \mathit{type}(y) = 1] + \Pr[A(y) \text{ is correct} \wedge \mathit{type}(y) = 2]$ where y is uniformly distributed over $\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$. Thus,

$$\Pr[A(y) \text{ is correct} \wedge \mathit{type}(y) = 1] > \frac{\epsilon}{2} \text{ or } \Pr[A(y) \text{ is correct} \wedge \mathit{type}(y) = 2] > \frac{\epsilon}{2}.$$

If B does not output `fail` in step 1, then y' is uniformly distributed over $\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$. Therefore,

$$\Pr[\mathsf{SucA} \wedge \mathsf{Type1} | \neg\mathsf{Fail}] > \frac{\epsilon^2}{4} \quad \text{or} \quad \Pr[\mathsf{SucA} \wedge \mathsf{Type2} | \neg\mathsf{Fail}] > \frac{\epsilon^2}{4}.$$

If $A(y)$ and $A(y')$ are correct, $\mathit{type}(y) = \mathit{type}(y') = \text{pow}$, and α is a good value, then B outputs correctly. Since $\Pr[\neg\mathsf{Fail}] > \Pr[c = 1] = 1/2$, $\Pr[\text{pow} = 1] = \Pr[\text{pow} = 2] = 1/2$, and $\Pr[\mathsf{GV}] > 1 - 2^{2k_0 - 6}/N > 1 - 2^{2k_0 - k + 7}$, we have

$$\begin{aligned} \epsilon' &\ge \Pr[\mathsf{SucA} \wedge \mathit{type}(y) = \mathit{type}(y') = \text{pow} \wedge \alpha \text{ is a good value}] \\ &\ge \Pr[\mathsf{GV}] \times \Pr[\neg\mathsf{Fail}] \times \Pr[\mathsf{SucA} \wedge \mathit{type}(y) = \mathit{type}(y') = \text{pow} | \neg\mathsf{Fail}] \\ &\ge \frac{1}{2} \cdot (1 - 2^{2k_0 - k + 7}) \times (\Pr[\mathsf{SucA} \wedge \mathsf{Type1} \wedge \text{pow} = 1 | \neg\mathsf{Fail}] \\ &\qquad\qquad + \Pr[\mathsf{SucA} \wedge \mathsf{Type2} \wedge \text{pow} = 2 | \neg\mathsf{Fail}]) \\ &= \frac{1}{2} \cdot (1 - 2^{2k_0 - k + 7}) \times (\Pr[\text{pow} = 1] \times \Pr[\mathsf{SucA} \wedge \mathsf{Type1} | \neg\mathsf{Fail}] \\ &\qquad\qquad + \Pr[\text{pow} = 2] \times \Pr[\mathsf{SucA} \wedge \mathsf{Type2} | \neg\mathsf{Fail}]) \\ &> \frac{\epsilon^2}{16} \cdot (1 - 2^{2k_0 - k + 7}). \end{aligned}$$

We estimate the running time of B. B runs A twice. B can solve $\alpha r - s = (z' - z\alpha) \cdot 2^{k_0} \pmod N$ in time $O(k^3)$. Therefore, $t' \le 2t + O(k^3)$. $\square$

Theorem 2. *If* RSA *is one-way, then* RSACD *is one-way.*

Proof. We prove that if there exists a polynomial-time inverting algorithm A for RSACD with non-negligible probability $\epsilon = \mathbf{Adv}^{1-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSACD},A}(k)$, then there exists a polynomial-time inverting algorithm D for RSA with non-negligible probability $\epsilon' = \mathbf{Adv}^{1-\mathrm{pow}-\mathrm{fnc}}_{\mathsf{RSA},D}(k)$. We specify the algorithm D to compute a pre-image of $Y \in \mathrm{Rng}_{\mathsf{RSA}}(N, e, k)$.

```
Algorithm $D((N, e, k), Y)$
  if $(2^{k-2} < N \leq 2^{k-1})$ return fail
  else
    $c \overset{R}{\leftarrow} \{0, 1\}$
    if $(c = 0)$ $y \leftarrow Y;\ x \leftarrow A((N, e, k), y);\ u \leftarrow f^1_{N,e,k}(x);\ v \leftarrow f^2_{N,e,k}(u);\ X \leftarrow v$
    else $u \leftarrow Y;\ v \leftarrow f^2_{N,e,k}(u);\ y \leftarrow f^3_{N,e,k}(v);\ x \leftarrow A((N, e, k), y);\ X \leftarrow x$
  return $X$
```

Now, we analyze the advantage of D. Let Fail be the event that D outputs `fail` and $\lambda = \Pr[\neg\mathsf{Fail}]$. It is clear that λ is non-negligible. In the following, $\Pr_1[\cdot]$ denotes $\Pr[\cdot|\neg\mathsf{Fail}]$. If D does not output `fail` and A outputs correctly, then D outputs correctly (See Figure 1). Therefore,

$$\begin{aligned}\epsilon' &> \Pr[\neg\mathsf{Fail}] \cdot (\Pr_1[c = 0 \ \wedge\ A((N, e, k), Y) \text{ is correct}] \\ &\qquad\qquad + \Pr_1[c = 1 \ \wedge\ A((N, e, k), Z) \text{ is correct}]) \\ &\geq \frac{\lambda}{2} \cdot (\Pr_1[A((N, e, k), Y) \text{ is correct}] \\ &\qquad\qquad + \Pr_1[A((N, e, k), Z) \text{ is correct } \wedge\ N \leq Z < 2^k]).\end{aligned}$$

where $Z = f^3_{N,e,k}(f^2_{N,e,k}(Y))$. We have

$$\begin{aligned}\Pr_1[A((N, e, k), Y) \text{ is correct}] &= \Pr_1[A((N, e, k), y) \text{ is correct} \,|\, 0 \leq y < N] \\ &> \Pr_1[A((N, e, k), y) \text{ is correct } \wedge\ 0 \leq y < N].\end{aligned}$$

Furthermore, we have $\Pr_1[N \leq Z < 2^k] > \Pr_1[N \leq y < 2^k]$ where Y is uniformly distributed over $\mathbb{Z}_N^*$ and y is uniformly distributed over $\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)$, since $\Pr_1[N \leq Z < 2^k] = \Pr_1[0 \leq Y < 2^k - N]$ and $|\mathbb{Z}_N^*| < |\mathrm{Rng}_{\mathsf{RSACD}}(N, e, k)|$. Since $\Pr_1[A((N, e, k), Z) \text{ is correct} \,|\, N \leq Z < 2^k] = \Pr_1[A((N, e, k), y) \text{ is correct} \,|\, N \leq y < 2^k]$, we have

$$\begin{aligned}&\Pr_1[A((N, e, k), Z) \text{ is correct} \wedge N \leq Z < 2^k] \\ &\qquad > \Pr_1[A((N, e, k), y) \text{ is correct } \wedge\ N \leq y < 2^k].\end{aligned}$$

Therefore,

$$\begin{aligned}\epsilon' &> \frac{\lambda}{2} \cdot (\Pr_1[A((N, e, k), y) \text{ is correct } \wedge\ 0 \leq y < N] \\ &\qquad\qquad + \Pr_1[A((N, e, k), y) \text{ is correct } \wedge\ N \leq y < 2^k]) \\ &= \frac{\lambda}{2} \cdot \Pr_1[A((N, e, k), y) \text{ is correct}] = \frac{\lambda}{2} \cdot \epsilon \qquad \square\end{aligned}$$

It is clear that if RSACD is one-way then RSA is one-way. Thus, the one-wayness of RSACD is equivalent to the one-wayness of RSA.

3 Application to Key-Privacy Encryption

3.1 Definitions of Key-Privacy

The classical security requirements of an encryption scheme, for example indistinguishability or non-malleability under the chosen-ciphertext attack, provide privacy of the encryption data. In [1], Bellare, Boldyreva, Desai, and Pointcheval proposed a new security requirement of encryption schemes called "key-privacy." It asks that the encryption provide (in addition to privacy of the data being encrypted) privacy of the key under which the encryption was performed.

In a heterogeneous public-key environment, encryption will probably fail to be anonymous for trivial reasons. For example, different users might be using different cryptosystems, or, if the same cryptosystem, have keys of different lengths. In [1], a public-key encryption scheme with common-key generation is described as follows.

Definition 6. *A public-key encryption scheme with common-key generation* $\mathcal{PE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *consists of four algorithms.*

- *The common-key generation algorithm* $\mathcal{G}$ *takes as input some security parameter* k *and returns some common key* I.
- *The key generation algorithm* $\mathcal{K}$ *is a randomized algorithm that takes as input the common key* I *and returns a pair* (pk, sk) *of keys, the public key and a matching secret key.*
- *The encryption algorithm* $\mathcal{E}$ *is a randomized algorithm that takes the public key* pk *and a plaintext* x *to return a ciphertext* y.
- *The decryption algorithm* $\mathcal{D}$ *is a deterministic algorithm that takes the secret key* sk *and a ciphertext* y *to return the corresponding plaintext* x *or a special symbol* $\perp$ *to indicate that the ciphertext was invalid.*

In [1], they formalized the property of "key-privacy." This can be considered under either the chosen-plaintext attack or the chosen-ciphertext attack, yielding two notions of security, IK-CPA and IK-CCA. (IK means "indistinguishability of keys".)

Definition 7 (IK-CPA, IK-CCA[1]). *Let* $\mathcal{PE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *be an encryption scheme. Let* $b \in \{0, 1\}$ *and* $k \in \mathbb{N}$. *Let* A_{cpa}, A_{cca} *be adversaries that run in two stages and where* A_{cca} *has access to the oracles* $\mathcal{D}_{sk_0}(\cdot)$ *and* $\mathcal{D}_{sk_1}(\cdot)$. *Note that* si *is the state information. It contains* pk_0, pk_1, *and so on. Now, we consider the following experiments:*

```
Experiment $\mathbf{Exp}^{\mathrm{ik-cpa}-b}_{\mathcal{PE},A_{\mathrm{cpa}}}(k)$
    $I \overset{R}{\leftarrow} \mathcal{G}(k)$
    $(pk_0, sk_0) \overset{R}{\leftarrow} \mathcal{K}(I);\ (pk_1, sk_1) \overset{R}{\leftarrow} \mathcal{K}(I)$
    $(x, \mathsf{si}) \leftarrow A_{\mathrm{cpa}}(\mathsf{find}, pk_0, pk_1)$
    $y \leftarrow \mathcal{E}_{pk_b}(x)$
    $d \leftarrow A_{\mathrm{cpa}}(\mathsf{guess}, y, \mathsf{si})$
    return $d$
```

```
Experiment $\mathbf{Exp}^{\mathrm{ik-cca}-b}_{\mathcal{PE},A_{\mathrm{cca}}}(k)$
    $I \overset{R}{\leftarrow} \mathcal{G}(k)$
    $(pk_0, sk_0) \overset{R}{\leftarrow} \mathcal{K}(I);\ (pk_1, sk_1) \overset{R}{\leftarrow} \mathcal{K}(I)$
    $(x, \mathsf{si}) \leftarrow A_{\mathrm{cca}}^{\mathcal{D}_{sk_0}(\cdot), \mathcal{D}_{sk_1}(\cdot)}(\mathsf{find}, pk_0, pk_1)$
    $y \leftarrow \mathcal{E}_{pk_b}(x)$
    $d \leftarrow A_{\mathrm{cca}}^{\mathcal{D}_{sk_0}(\cdot), \mathcal{D}_{sk_1}(\cdot)}(\mathsf{guess}, y, \mathsf{si})$
    return $d$
```

Above it is mandated that A_{cca} never queries $\mathcal{D}_{sk_0}(\cdot)$ and $\mathcal{D}_{sk_1}(\cdot)$ on the challenge ciphertext y. For atk $\in$ {cpa, cca} *we define the advantages via*

$$\mathbf{Adv}^{\mathrm{ik-atk}}_{\mathcal{PE},A_{\mathrm{atk}}}(k) = \left|\Pr[\mathbf{Exp}^{\mathrm{ik-atk-1}}_{\mathcal{PE},A_{\mathrm{atk}}}(k) = 1] - \Pr[\mathbf{Exp}^{\mathrm{ik-atk-0}}_{\mathcal{PE},A_{\mathrm{atk}}}(k) = 1]\right|.$$

The scheme $\mathcal{PE}$ is said to be IK-CPA secure (respectively IK-CCA secure) if the function $\mathbf{Adv}^{\mathrm{ik-cpa}}_{\mathcal{PE},A_{\mathrm{cpa}}}(\cdot)$ (resp. $\mathbf{Adv}^{\mathrm{ik-cca}}_{\mathcal{PE},A_{\mathrm{cca}}}(\cdot)$) is negligible for any adversary A whose time complexity is polynomial in k.

The "time-complexity" is the worst-case execution time of the experiment plus the size of the code of the adversary, in some fixed RAM model of computation.

3.2 RSA-RAEP by Bellare, Boldyreva, Desai, and Pointcheval

A simple observation that seems to be folklore is that standard RSA encryption does not provide key-privacy, even when all moduli in the system have the same length. Suppose an adversary knows that the ciphertext y is created under one of two keys (N_0, e_0) or (N_1, e_1), and suppose $N_0 \leq N_1$. If $y \geq N_0$ then the adversary bets it was created under (N_1, e_1), else it bets it was created under (N_0, e_0). It is not hard to see that this attack has non-negligible advantage.

In [1], they proposed an RSA-based encryption scheme which is secure in the sense of IK-CCA. It is RSA-RAEP which is a variant of RSA-OAEP. Since their variant chooses N from $(2^{k-2}, 2^k)$, it simply repeats the ciphertext computation, each time using new coins, until the ciphertext y satisfies $y < 2^{k-2}$.

Definition 8 (RSA-RAEP [1]). *RSA-RAEP = $(\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is as follows. The common-key generation algorithm $\mathcal{G}$ takes a security parameter k and returns parameters k, k_0 and k_1 such that $k_0(k)+k_1(k) < k$ for all $k > 1$. This defines an associated plaintext-length function $n(k) = k - k_0(k) - k_1(k)$. The key generation algorithm $\mathcal{K}$ takes k, k_0, k_1, runs the key-generation algorithm of* RSA*, and gets (N, e) and (N, d). The public key pk is $(N, e), k, k_0, k_1$ and the secret key sk is $(N, d), k, k_0, k_1$. The other algorithms are depicted below. Let $G : \{0,1\}^{k_0} \to \{0,1\}^{n+k_1}$ and $H : \{0,1\}^{n+k_1} \to \{0,1\}^{k_0}$. Note that $[x]^n$ denotes the n most significant bits of x and $[x]_m$ denotes the m least significant bits of x.*

```
Algorithm $\mathcal{E}^{G,H}_{pk}(x)$
  $ctr = -1$
  repeat
    $ctr \leftarrow ctr + 1$
    $r \xleftarrow{R} \{0,1\}^{k_0}$
    $s \leftarrow (x \,||\, 0^{k_1}) \oplus G(r);\ t \leftarrow r \oplus H(s)$
    $v \leftarrow (s||t)^e \bmod N$
  until $((v < 2^{k-2}) \vee (ctr = k_1))$
  if $(ctr = k_1)$ $y \leftarrow 1||0^{k_0+k_1}||x$
  else $y \leftarrow 0||v$
  return $y$
```

```
Algorithm $\mathcal{D}^{G,H}_{sk}(y)$
  $b \leftarrow [y]^1;\ v \leftarrow [y]_{k_0+k_1+n}$
  if $(b = 1)$
    $w \leftarrow [v]^{k_0+k_1};\ x \leftarrow [v]_n$
    if $(w = 0^{k_0+k_1})$ $z \leftarrow x$ else $z \leftarrow \perp$
  else
    $s \leftarrow [v^d]^{n+k_1};\ t \leftarrow [v^d]_{k_0}$
    $r \leftarrow t \oplus H(s)$
    $x \leftarrow [s \oplus G(r)]^n;\ p \leftarrow [s \oplus G(r)]_{k_1}$
    if $(p = 0^{k_1})$ $z \leftarrow x$ else $z \leftarrow \perp$
  return $z$
```

They proved RSA-RAEP is secure in the sense of IND-CCA2 and IK-CCA in the random oracle model assuming RSA is one-way. In RSA-RAEP, the expected number of exponentiations for encryption is

$$\sum_{i=1}^{k_1} i \left(1 - \frac{2^{k-2}}{N}\right)^{i-1} \frac{2^{k-2}}{N} = \frac{1-(1-p)^{k_1}}{p} - k_1(1-p)^{k_1}$$

where $p = 2^{k-2}/N$. Suppose that N is uniformly distributed in $(2^{k-2}, 2^k)$, the expected number of this scheme is two. However, the upper bound of the number of exponentiations for encryption is $k_1 (\gg 2$, security parameter).

3.3 Our Proposed Encryption Scheme

In this section, we propose our encryption scheme, which uses RSACD instead of RSA.

Definition 9. *The common-key generation algorithm $\mathcal{G}$, and the oracles G and H are the same as RSA-RAEP. The key generation algorithm $\mathcal{K}$ is almost the same as RSA-RAEP. The difference is running the key-generation algorithm of* RSACD *instead of* RSA*. The other algorithms are described as follows. Note that the valid ciphertext y satisfies $y \in [0, 2^k)$ and $y \bmod N \in \mathbb{Z}_N^*$.*

```
Algorithm $\mathcal{E}_{pk}^{G,H}(x)$
  $r \xleftarrow{R} \{0,1\}^{k_0}$
  $s \leftarrow (x \,||\, 0^{k_1}) \oplus G(r)$
  $t \leftarrow r \oplus H(s)$
  $v \leftarrow f_{N,e,k}(s||t)$
  return $y$
```

```
Algorithm $\mathcal{D}_{sk}^{G,H}(y)$
  $s \leftarrow [g_{N,d,k}(y)]^{n+k_1};\ t \leftarrow [g_{N,d,k}(y)]_{k_0}$
  $r \leftarrow t \oplus H(s)$
  $x \leftarrow [s \oplus G(r)]^n;\ p \leftarrow [s \oplus G(r)]_{k_1}$
  if $(p = 0^{k_1})$ $z \leftarrow x$ else $z \leftarrow \perp$
  return $z$
```

Using Theorem 1 and 2, we can prove the following theorem.

Theorem 3. *Our scheme is secure in the sense of IND-CCA2 and IK-CCA in the random oracle model assuming* RSA *is one-way.*

Proof (Idea). Fujisaki, Okamoto, Pointcheval, and Stern [4] proved OAEP with partial one-way permutation is secure in the sense of IND-CCA2. Thus, OAEP with $f_{N,e,k}$ is secure in the sense of IND-CCA2 assuming RSACD is partial one-way.

Bellare, Boldyreva, Desai, and Pointcheval [1] proved RSA-RAEP is secure in the sense of IK-CCA in the random oracle model assuming RSA is partial one-way. Noticing that the function $f_{N,e,k}$ and $g_{N,d,k}$, and the domain of valid signature change, we can prove in a similar way that our scheme is secure in the sense of IK-CCA in the random oracle model assuming RSACD is partial one-way.

Therefore, by Theorem 1 and 2, we can prove that our scheme is secure in the sense of IND-CCA2 and IK-CCA under the assumption that RSA is one-way. $\square$

In this scheme, the expected number of exponentiations in encryption is

$$1 \times \frac{2(2^k - N)}{2^k} + 2 \times \left(1 - \frac{2(2^k - N)}{2^k}\right) = \frac{N}{2^{k-1}}.$$

Suppose that N is uniformly distributed in $(2^{k-1}, 2^k)$, the expected number of our scheme is two, the same as RSA-RAEP. In our scheme, the number of exponentiations for encryption is at most two, while in RSA-RAEP, the upper bound of this number is k_1 ($\gg 2$, security parameter). Notice that we use the randomness only for an RSA-OAEP.

4 Application to Ring Signature

4.1 Definitions of Ring Signature

In [2], Rivest, Shamir, and Tauman proposed the notion of ring signature, which allows a member of an ad hoc collection of users S to prove that a message is authenticated by a member of S without revealing which member actually produced the signature. Unlike group signature, ring signature has no group managers, no setup procedures, no revocation procedures, and no coordination.

Definition 10 (Ring Signature [2]). *One assumes that each user (called a ring member) has received (via a PKI or a certificate) a public key P_k, for which the corresponding secret key is denoted by S_k. A ring signature scheme consists of the following algorithms.*

– **ring-sign**$(m, P_1, P_2, \cdots, P_r, s, S_s)$ *which produces a ring signature σ for the message m, given the public keys $P_1, P_2, \cdots, P_r$ of the r ring members, together with the secret key S_s of the s-th member (who is the actual signer).*
– **ring-verify**(m, σ) *which accepts a message m and a signature σ (which includes the public key of all the possible signers), and outputs either "true" or "false".*

The signer does not need the knowledge, consent, or assistance of the other ring members to put them in the ring. All he needs is knowledge of their regular public keys. Verification must satisfy the usual soundness and completeness conditions, but in addition the signature scheme must satisfy "signer-ambiguous", which is the property that the verifier should be unable to determine the identity of the actual signer with probability greater than $1/r + \epsilon$, where r is the size of the ring, and ϵ is negligible.

4.2 RSA-based Ring Signature Scheme by Rivest, Shamir, and Tauman

In [2], they constructed ring signature schemes in which all the ring member use RSA as their individual signature schemes. We review their scheme.

Let ℓ, k, and b be security parameters. Let E be a symmetric encryption scheme over $\{0,1\}^b$ using ℓ-bit keys and h be a hash function which maps arbitrary strings to ℓ-bit strings. They use h to make a key for E. They assume that each user has an RSA public key $P_i = (N_i, e_i)$ which specifies the trap-door one-way permutation f_i on $\mathbb{Z}_{N_i} : f_i(x) = x^e \bmod N_i$.

To sign and verify a ring signature, they proposed a combining function $C_{k,v}$ based on a symmetric encryption scheme E modeled by a (keyed) random permutation

$$C_{k,v}(y_1, \cdots, y_r) = E_k(y_r \oplus E_k(y_{r-1} \oplus \cdots E_k(y_2 \oplus E_k(y_1 \oplus v)) \cdots))$$

where v is an initialization value. In their scheme, the inputs y_i to the combining function are computed as $g_i(x_i)$ for some $x_i \in \{0,1\}^b$. They defined the extended trap-door permutation g_i over $\{0,1\}^b$ which has a common domain for each user as follows: for any b-bit input x_i define nonnegative integers q_i and r_i so that $x_i = q_i N_i + r_i$ and $0 \leq r_i < N_i$. Then

$$g_i(x_i) = \begin{cases} q_i N_i + f_i(r_i) & \text{if } (q_i + 1)N_i \leq 2^b \\ x_i & \text{otherwise.} \end{cases}$$

If b is sufficiently large (e.g. 160 bits larger than any of the N_i), g_i is a one-way trap-door permutation. (See also [5].)

A ring signature on a message m consists in a tuple $(v, x_1, \cdots, x_r)$ and the signature is valid iff $C_{h(m),v}(g_1(x_1), \cdots, g_r(x_r)) = v$. For any message m, any fixed values v and $\{x_i\}_{i \neq s}$, one can efficiently compute the value y_s such that the combining function outputs v by using the following equation:

$$y_s = E_k^{-1}\big(y_{s+1} \oplus \cdots E_k^{-1}(y_r \oplus E_k^{-1}(v)) \cdots\big) \oplus E_k\big(y_{s-1} \oplus \cdots E_k(y_1 \oplus v) \cdots\big).$$

Now using her knowledge of the trap-door for function f_s, s-th member (the actual signer) is able to compute x_s such that $g_s(x_s) = y_s$. Thus, the ring member can generate a valid signature. Rivest, Shamir, and Tauman proved this scheme is unconditionally signer-ambiguous and provably secure in the random oracle model assuming RSA is one-way.

4.3 Our Proposed Ring Signature Scheme

Unlike group signature, ring signature has no group managers, no setup procedures, no revocation procedures, and no coordination, and each user can use a public key whose length is different from other users.

In [2], Rivest, Shamir, and Tauman mentioned the case that a member of the cabinet of some country wished to leak her secret to a journalist. In this kind of situation, it was reasonable to consider that the members of the same group use the RSA moduli of the same length. In our scheme, we assume the situation that each user chooses her public key with the same size.

Our scheme is almost the same as the previous scheme. The difference is using $f_{N_i,e,k}(\cdot)$ in Section 2.2 instead of $g_i(\cdot)$ in Section 4.2. Then, the domain of $f_{N,e,k}(\cdot)$ is $\{0,1\}^k$, while that of the previous scheme is $\{0,1\}^{k+160}$, where k is the length of the RSA moduli. Thus, we can reduce the size of signature in this situation. In particular, the size of signature of our scheme is 160 bits smaller than that of the previous scheme in order to archive security parameter $k = 1024$. In our scheme, the number of exponentiations is one or two, while that of the original scheme in [2] is one. Since $f_{N_i,e,k}(\cdot)$ is a trap-door one-way permutation as well as $g_i(\cdot)$, we can easily prove the following theorem in a similar way as for the previous scheme.

Theorem 4. *Our scheme is unconditionally signer-ambiguous and provably secure in the random oracle model assuming* RSA *is one-way.*

We can also apply this scheme to the Rabin-based ring signature scheme in [2] in a similar way.

5 Conclusion

In this paper, we have constructed the RSA family of trap-door permutations with a common domain and proposed the applications of our construction to the key-privacy encryption and ring signature schemes, which have some advantage to the previous schemes. It might be interesting to consider other applications of our RSA family, and different constructions of a family of trap-door permutations with a common domain.

References

1. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-Privacy in Public-Key Encryption. [6] 566–582
2. Rivest, R.L., Shamir, A., Tauman, Y.: How to Leak a Secret. [6] 552–565
3. Bellare, M., Rogaway, P.: Optimal asymmetric encryption – How to encrypt with RSA. In Santis, A.D., ed.: Advances in Cryptology – EUROCRYPT '94. Volume 950 of Lecture Notes in Conputer Science., Perugia, Italy, Springer-Verlag (1994) 92–111
4. Fujisaki, E., Okamoto, T., Pointcheval, D., Stern, J.: RSA-OAEP is Secure under the RSA Assumption. In Kilian, J., ed.: Advances in Cryptology – CRYPTO 2001. Volume 2139 of Lecture Notes in Conputer Science., Santa Barbara, California, USA, Springer-Verlag (2001) 260–274
5. Desmedt, Y.: Securing traceability of ciphertexts: Towards a secure software escrow scheme. In Guillou, L.C., Quisquater, J.J., eds.: Advances in Cryptology – EUROCRYPT '95. Volume 921 of Lecture Notes in Conputer Science., Saint-Malo, France, Springer-Verlag (1995) 147–157
6. Boyd, C., ed.: Advances in Cryptology – ASIACRYPT 2001. In Boyd, C., ed.: Advances in Cryptology – ASIACRYPT 2001. Volume 2248 of Lecture Notes in Computer Science., Gold Coast, Australia, Springer-Verlag (2001)