Reverse Public Key Encryption

David Naccache[1], Rainer Steinwandt[2], and Moti Yung[3,4]

[1] École normale supérieure, Département d'informatique, Équipe de cryptographie, 45 rue d'Ulm, F-75230 Paris CEDEX 05, France
[2] Department of Mathematical Sciences, Florida Atlantic University, 777 Glades Road, Boca Raton, FL 33431, USA
[3] Columbia University, Department of Computer Science, 1214 Amsterdam Avenue, New York, NY 10027, USA
[4] Google Inc, USA

david.naccache@ens.fr; rsteinwa@fau.edu; moti@cs.columbia.edu

Abstract. This exposition paper suggests a new low-bandwidth public-key encryption paradigm. The construction turns a weak form of key privacy into message privacy as follows: let $\mathcal{E}$ be a public-key encryption algorithm. We observe that if the distributions $\mathcal{E}(pk_0, \bullet)$ and $\mathcal{E}(pk_1, \bullet)$ are indistinguishable for two public keys pk_0, pk_1, then a message bit $b \in \{0, 1\}$ can be embedded in the choice of pk_b.
As the roles of the public-key and the plaintext are reversed, we refer to the new mode of operation as *Reverse Public-Key Encryption* (RPKE). We present examples of and variations on the idea and explore RPKE's relationship with key privacy, and we also discuss how to employ it to enable a new implementation of deniable encryption.

1 Introduction

System designers traditionally distinguish between *predictive*, *protective* and *reactive* security means.

Predictive security mechanisms are meant to sense that an attack is in preparation. Port scanning detection and critical function monitoring are typical predictive security functions.

Protective mechanisms must block or slow-down attacks. Encryption, signature, passwords and tamper-resistance all fall into the protective category.

Finally, should an attack succeed, *reactive* security functions must contain damage and allow system recovery.

Public-key encryption is a mature protective discipline. Indeed, current theory provides the practitioner with efficient and well-understood public-key encryption primitives with provable security guarantees.

In other words, considerable attention is currently being focused on the validity of the underlying *security guarantees*. In practice, security guarantees may fail for a variety of reasons.[5] Should this happen, replacing the underlying cryptographic functions is an option. As large-scale replacement can be impractical or costly, it is interesting to explore the existence of *reactive* fall-back public-key encryption modes whose security guarantees are only indirectly linked to the traditional system's security guarantees.

This work introduces such a mode of operation called *Reverse Public-Key Encryption* (RPKE). Denoting by $\mathcal{E} = \mathcal{E}(pk, m)$ a public-key encryption algorithm[6], the underlying idea is rather simple and intuitive: we start by assigning to each user *two* public keys pk_0 and pk_1 and a couple of message sampling algorithms $(\mathcal{M}_0, \mathcal{M}_1)$. The $\mathcal{M}_i$ depend on the specific properties of $\mathcal{E}$ but might be very simple and even have constant output (*i.e.*, $\mathcal{M}_0 = \mathcal{M}_1 =$ fixed constant).

We now assume that without the trapdoor information (sk_0, sk_1), encryptions[7] under pk_0 and pk_1 are computationally indistinguishable. The idea consists in public-key encrypting a message bit b by sending to the owner of (sk_0, sk_1) the quantity $c = \mathcal{E}(pk_b, \text{something})$.

Despite its simplicity, elaborating on this basic construction turns out to be worthwhile: even if the traditional public key encryption scheme built upon $(\mathcal{E}, \mathcal{D})$ does not offer strong provable security guarantees, using the "insecure" scheme in reverse encryption mode may still provide (low bandwidth) encryption with strong security guarantees. Thus RPKE can serve as a temporary fall-back mode, should (the traditional use of) $\mathcal{E}$ and $\mathcal{D}$ fail. In addition, as delineated in Section 4, when RPKE is concurrently used with traditional public-key encryption, a new form of deniable encryption may be obtained.

Throughout this paper we will use the acronym TPKE to refer to traditional, (*i.e.*, regular) public-key encryption.

This paper: after introducing the basic RPKE notion, we show how to create RPKE schemes from TPKEs featuring (a weak form of) key privacy. We then explore the connections between RPKE, TPKE and key privacy.

[5] *e.g.*, sudden algorithmic progress, the advent of new adversarial models, improper implementations, *etc.*

[6] $\mathcal{D} = \mathcal{D}(sk, c)$ being the corresponding decryption algorithm.

[7] Throughout this paper "encryption under pk_i", is to be understood as an abbreviation of "encryption under pk_i of messages sampled using $\mathcal{M}_i$".

Finally, Section 4 discusses how the concurrent use of TPKE and RPKE may serve as enabling mechanism for deniable encryption (a notion formalized in [CDNO97]).

This paper remains at the informal level, and the presented examples are not to be seen as fully worked out cryptographic schemes. We hope, however, that the presented ideas will stimulate further, more formal, follow-up work on reverse public key encryption.

Related work: to the best of our knowledge, RPKE has not been discussed in past literature so far. Nonetheless, given that RPKE might be interpreted as imposing a form of key privacy in a TPKE, we attract the reader's attention to the following references: following the formalization of key privacy in [BBDP01a], several key-privacy instantiations were proposed for RSA [HOT04], ElGamal [ZHI07] and McEliece [YCK+07]. Sufficient conditions for key privacy are given in [Hal05].

Group encryption [KTY07] can be interpreted in RPKE terms, too: the receiver's anonymity provided by group encryption hides the plaintext, while the authority's capability to remove anonymity corresponds to a reverse decryption capability. The idea of receiver anonymity also appears in the context of broadcast encryption [BBW06], but *the interpretation of such a privacy guarantee in encryption terms* is, to the best of the authors' knowledge, new.

In the light of the above, considering RPKE as a fall-back mode, raises the question of labeling the *basing of key and message privacy on different hardness assumptions* as a desirable cryptographic design goal.

2 Preliminaries and basic construction

As usual, we regard public key encryption as a collection of four (potentially randomized) polynomial time algorithms. We adopt the notation used in [BBDP01b].

Definition 1 (Traditional Public Key Encryption (TPKE)). *A (traditional) public key encryption scheme $\mathcal{P} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ consists of four polynomial time algorithms:*

- *A randomized common-key generation algorithm $\mathcal{G}$, taking as input a security parameter k and outputting a common key ck.*

– *A randomized key generation algorithm $\mathcal{K}$ transforming ck into a matching public/secret key-pair* (pk, sk)*:*

$$(pk, sk) \xleftarrow{R} \mathcal{K}(ck)$$

– *A randomized encryption algorithm $\mathcal{E}$ transforming pk and a plaintext $m \in \mathfrak{M}(pk)$ into a ciphertext c:*

$$c \xleftarrow{R} \mathcal{E}(pk, m)$$

where $\mathfrak{M}(pk)$ is the message space associated with pk.

– *A deterministic decryption algorithm $\mathcal{D}$ transforming sk and c into the corresponding plaintext m or into an invalidity symbol $\perp$ not contained in any message space.*

For all $m \in \mathfrak{M}(pk)$ the following relation must hold:

$$\mathcal{D}(sk, \mathcal{E}(pk, m)) = m$$

In addition to the above definition, we silently assume the existence of a fifth algorithm $\mathcal{M}$ allowing to generate (draw) plaintexts from $\mathfrak{M}(pk)$:

$$m \xleftarrow{R} \mathcal{M}(pk) \text{ such that } m \in \mathfrak{M}(pk)$$

Typically, our instantiations of $\mathcal{M}$ will be trivial and under some circumstances even constant (*i.e.*, output a single plaintext message). It can be useful, however, to have the flexibility to choose a plaintext uniformly at random. We hence use the notation $m \xleftarrow{R} \mathcal{M}(pk)$ to denote (potentially randomized) plaintext message sampling. Moreover, we denote by

$$\mathrm{Im}(\mathcal{M}(pk)) \subseteq \mathfrak{M}(pk)$$

the set of possible outputs of $\mathcal{M}(pk)$ and assume that membership in $\mathrm{Im}(\mathcal{M}(pk))$ can be tested in polynomial time.

2.1 Reverse public-key encryption and key privacy

Given a public-key encryption scheme $\mathcal{P} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, we construct an RPKE $\mathcal{P}^{\mathrm{Rev}} = (\mathcal{G}^{\mathrm{Rev}}, \mathcal{K}^{\mathrm{Rev}}, \mathcal{E}^{\mathrm{Rev}}, \mathcal{D}^{\mathrm{Rev}})$ encrypting one-bit messages as follows:

Common-Key Generation: This algorithm is not altered, *i.e.*:

$$\mathcal{G}^{\mathrm{Rev}} = \mathcal{G}$$

Key Generation: $\mathcal{K}^{\text{Rev}}$ runs $\mathcal{K}$ twice and obtains two independent key-pairs (pk_0, sk_0) and (pk_1, sk_1).

The public-key of $\mathcal{P}^{\text{Rev}}$ is (pk_0, pk_1) and the secret-key is (sk_0, sk_1).

Encryption: for $b \in \{0, 1\}$ we re-define the encryption process as:

$$\mathcal{E}^{\text{Rev}}(pk, b) := \mathcal{E}(pk_b, m_b) \quad \text{where} \quad m_b \xleftarrow{R} \mathcal{M}(pk_b)$$

Put differently, the plaintext bit b determines whether we apply pk_0 or pk_1 to a plaintext m_b sampled from $\mathfrak{M}(pk_b)$.

Decryption: for a given ciphertext c, $\mathcal{D}^{\text{Rev}}$ computes:

$$\begin{aligned} \tilde{m}_0 &\leftarrow \mathcal{D}(sk_0, c) \\ \tilde{m}_1 &\leftarrow \mathcal{D}(sk_1, c) \end{aligned}$$

$$\mathcal{D}^{\text{Rev}}(sk, c) := \begin{cases} 0 & \text{, if } \tilde{m}_0 \in \text{Im}(\mathcal{M}(pk_0)) \text{ and } \tilde{m}_1 \notin \text{Im}(\mathcal{M}(pk_1)) \\ 1 & \text{, if } \tilde{m}_1 \in \text{Im}(\mathcal{M}(pk_1)) \text{ and } \tilde{m}_0 \notin \text{Im}(\mathcal{M}(pk_0)) \\ \perp & \text{, otherwise.} \end{cases}$$

Note that $\tilde{m}_i = \perp$ satisfies the condition $\tilde{m}_i \notin \text{Im}(\mathcal{M}(pk_i))$.

Remark 1. While the above construction can be easily generalized to any polynomial number of public keys $pk_1, \ldots, pk_n$, in this paper, we restrict our discussion to the case $n = 2$. Similarly, one may consider a variant of reverse encryption with a single public key—cf. Example 3 below.

Before we proceed, let us provide a first informal RPKE example:

Example 1. For a k-bit RSA modulus n, let $H : \mathbb{Z}/n\mathbb{Z} \longrightarrow \{0, 1\}^k$ be a random oracle. To encrypt a plaintext $m \in \mathbb{Z}/n\mathbb{Z}$ with public RSA key (n, e), the encryption algorithm computes the ciphertext as

$$c := (m^e \bmod n, H(m)) \quad .$$

Obviously, this scheme does not offer semantic security since the encrypted message can be checked. However, the reverse mode of operation yields a scheme (with small bandwidth) where guessing the (one-bit) plaintext appears harder, as we discuss next.

Embedding b in e: generate $n = pq$ and compute two private keys d_i such that $d_i \times \text{FDH}(i) = 1 \bmod \phi(n)$ for $i \in \{0, 1\}$, where

$$\text{FDH}: \ \{0,1\} \longrightarrow \left(\frac{\mathbb{Z}}{\phi(n)\mathbb{Z}}\right)^*$$

assigns a random number to each of the two possible plaintexts 0, 1. Encrypt $b \in \{0, 1\}$ as

$$c = (r^{\text{FDH}(b)} \bmod n, H(r))$$

where the redundancy $H(r)$ is used to spot the uniformly at random chosen $r \in_R \mathbb{Z}/n\mathbb{Z}$ during decryption.

Remark 2. Note that under the strong RSA assumption, decrypting the first element in c is hard, and the added redundancy (either given as a result of a random oracle over r or a few hard core bits of r) does not help in the decryption process.

Another option for hiding the key consists in using different moduli to implement reverse encryption:

Embedding b in n: generate two moduli n_i, fix e and let

$$d_i \times e = 1 \bmod \phi(n_i).$$

Let $B \gg \max\{n_i\}$ be a large bound (*e.g.*, $B = \lfloor \max\{n_i\}^{\frac{3}{2}} \rfloor$). Encrypt $b \in \{0, 1\}$ as

$$c = (r^e \bmod n_b) + r' \times n_b$$

for $r \in_R \mathbb{Z}/n_b\mathbb{Z}$ and $r' \in_R \{0, \ldots, \lfloor B/n_b \rfloor\}$ and provide $H(r, r')$ to allow spotting the correct decryption.

Intuitively, for an RPKE to be secure, encryptions under pk_0 and pk_1 must "look the same" without the trapdoors. It is easy to check that a TPKE offering key privacy in the sense of key indistinguishability is a sufficient condition for $\mathcal{P}^{\text{Rev}}$ to offer security in the sense of indistinguishable encryptions, here $\mathcal{M}(pk_b)$ can simply output a constant plaintext. For making this more precise, motivated by [BBDP01b], we define an adversary $\mathcal{A}_{\mathsf{ik}}$ running in two phases:

- At the find phase, $\mathcal{A}_{\mathsf{ik}}$ is given two public-keys (pk_0, pk_1) and outputs a plaintext m.

– At the guess phase, $\mathcal{A}_{\mathsf{ik}}$ is given the encryption of m under one of the pk_i and attempts to guess i.

Definition 2 (Indistinguishable Keys). *Let $\mathcal{P} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public key encryption scheme. We say that $\mathcal{P}$ is* IK-CPA secure*, if for all polynomial time $\mathcal{A}_{\mathsf{ik}}$ the function*

$$\mathbf{Adv}^{\mathsf{ik\text{-}cpa}}(k) := \left|\Pr[\mathbf{Exp}^{\mathsf{ik\text{-}cpa\text{-}1}}(k) = 1] - \Pr[\mathbf{Exp}^{\mathsf{ik\text{-}cpa\text{-}0}}(k) = 1]\right|$$

is negligible.

Experiment 1 : $\mathbf{Exp}^{\mathsf{ik\text{-}cpa}}(k)$

for $b \in \{0, 1\}$ **do**
  let $ck \xleftarrow{R} \mathcal{G}(k)$
  let $(pk_0, sk_0) \xleftarrow{R} \mathcal{K}(ck)$
  let $(pk_1, sk_1) \xleftarrow{R} \mathcal{K}(ck)$
  let $(m, state\_info) \xleftarrow{R} \mathcal{A}_{\mathsf{ik}}(\mathsf{find}, pk_0, pk_1)$
  let $c \xleftarrow{R} \mathcal{E}(pk_b, m)$
  let $\mathbf{Exp}^{\mathsf{ik\text{-}cpa\text{-}b}}(k) \xleftarrow{R} \mathcal{A}_{\mathsf{ik}}(\mathsf{guess}, c, state\_info)$
end for
return $(\mathbf{Exp}^{\mathsf{ik\text{-}cpa\text{-}0}}(k), \mathbf{Exp}^{\mathsf{ik\text{-}cpa\text{-}1}}(k))$

We now consider a second two-phase adversary $\mathcal{A}_{\mathsf{ind}}$ and define:

Definition 3 (Indistinguishable Encryptions). *(cf. [GM84,RS92]) Let $\mathcal{P} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public key encryption scheme. We say that $\mathcal{P}$*

Experiment 2 : $\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}(k)$

1: **for** $b \in \{0, 1\}$ **do**
2: **let** $ck \xleftarrow{R} \mathcal{G}(k)$
3: **let** $(pk, sk) \xleftarrow{R} \mathcal{K}(ck)$
4: **let** $(m_0, m_1, state\_info) \xleftarrow{R} \mathcal{A}_{\mathsf{ind}}(\mathsf{find}, pk)$
5: **let** $c \xleftarrow{R} \mathcal{E}_{pk}(m_b)$
6: **let** $\mathbf{Exp}^{\mathsf{ind\text{-}cpa\text{-}b}}(k) \xleftarrow{R} \mathcal{A}_{\mathsf{ind}}(\mathsf{guess}, c, state\_info)$
7: **end for**
8: **return** $(\mathbf{Exp}^{\mathsf{ind\text{-}cpa\text{-}0}}(k), \mathbf{Exp}^{\mathsf{ind\text{-}cpa\text{-}1}}(k))$

is IND-CPA secure*, if for all polynomial time $\mathcal{A}_{\mathrm{ind}}$ the function*

$$\mathbf{Adv}^{\mathsf{ind\text{-}cpa}}(k) := \left|\Pr[\mathbf{Exp}^{\mathsf{ind\text{-}cpa\text{-}1}}(k) = 1] - \Pr[\mathbf{Exp}^{\mathsf{ind\text{-}cpa\text{-}0}}(k) = 1]\right|$$

is negligible.

Note that at Step 4 of $\mathbf{Exp}^{\mathsf{ind\text{-}cpa}}(k)$ *we require that* $m_0, m_1 \in \mathfrak{M}(pk)$ *with* $m_0 \neq m_1$ *and such that* m_0 *and* m_1 *are of equal length.*

If we assume the existence of a plaintext m_0 that can be encrypted under all public keys, then the following proposition establishes the IND-CPA security of RPKE.[8]

Proposition 1. *Denote by* $\mathcal{P}$ *an* IK-CPA *secure public key encryption scheme. If the sampling algorithm* $\mathcal{M}(pk)$ *has a constant output* m_0 *such that* $\forall pk: \ m_0 \in \mathfrak{M}(pk)$*, then* $\mathcal{P}^{\mathrm{Rev}}$ *as defined above is* IND-CPA *secure.*

Proof. (sketch) Suppose that there exists a polynomial time adversary $\mathcal{A}_{\mathsf{ind}}$ contradicting the IND-CPA security of $\mathcal{P}^{\mathrm{Rev}}$. As shown in Algorithm 1 and 2, we turn $\mathcal{A}_{\mathsf{ind}}$ into a polynomial time IK-CPA adversary $\mathcal{A}_{\mathsf{ik}}$ against $\mathcal{P}$. If $\mathcal{A}_{\mathsf{ind}}$'s guess d is correct, the so-constructed $\mathcal{A}_{\mathsf{ik}}$ has successfully

Algorithm 1 : find phase of IK-CPA adversary $\mathcal{A}_{\mathsf{ik}}$ against $\mathcal{P}$

1: **receive** $pk = (pk_0, pk_1)$
2: **launch** $\mathcal{A}_{\mathsf{ind}}(\mathsf{find}, (pk_0, pk_1))$
3: As $\mathcal{P}^{\mathrm{Rev}}$ encrypts only one bit, $\mathcal{A}_{\mathsf{ind}}(\mathsf{find}, \cdot)$ returns a triple $(0, 1, \mathit{state\_info})$.
4: **return** $(m_0, \mathit{state\_info})$ as the result of $\mathcal{A}_{\mathsf{ik}}(\mathsf{find}, \cdot, \cdot)$.

Algorithm 2 : guess phase of IK-CPA adversary $\mathcal{A}_{\mathsf{ik}}$ against $\mathcal{P}$

1: **receive** the IK-attack challenge $c \xleftarrow{R} \mathcal{E}(pk_b, m_0)$
2: **launch** $\mathcal{A}_{\mathsf{ind}}(\mathsf{guess}, c, \mathit{state\_info})$
3: **let** d be the value returned by $\mathcal{A}_{\mathsf{ind}}$.
4: **return** d

identified the public key used to encrypt m_0. □

Note that in our initial examples, not granting the adversary access to the message was crucial to achieving the necessary form of key privacy: the message basically played the role of internal random coins of the encryption process. However, as will be illustrated in Examples 3 and 4 below, reverse encryption can also be implemented in settings where the (traditional) plaintext is known, or even chosen, by the adversary.

[8] Assuming the existence of such a "universal plaintext" does not seem a major restriction—in all practical cryptosystem embodiments, one actually wants the *entire message space* to be independent of *pk*.

3 Examples of RPKE

Viewing the requirements of message privacy and key privacy as orthogonal (see [BBDP01a,BBDP01b,ZHI07]), reverse encryption seems attractive. Actually, indistinguishable keys are more than we need for secure RPKE: for secure RPKE it is enough to have *some* choice of plaintext pairs (m_0, m_1) for which ciphertexts of the forms $\mathcal{E}_{pk_0}(m_0)$ and $\mathcal{E}_{pk_1}(m_1)$ are computationally indistinguishable. Differing from the IK-CPA attack setting, an adversary against the IND-CPA security of $\mathcal{P}^{\mathrm{Rev}}$ is not allowed to choose a particular plaintext to distinguish between pk_0 and pk_1.

Example 2 (Reverse encryption without key privacy). Denote by $\mathcal{P}_2$ the following ElGamal public key encryption variant:

$\mathcal{G}$: fix a generator g of a cyclic group G of order q and a random oracle $H: G \times G \longrightarrow G$

$\mathcal{K}$: choose $a \leftarrow \{0, \ldots, q-1\}$ uniformly at random and return

$$(sk, pk) := (a, g^a),$$

the associated message space $\mathfrak{M}(pk)$ is the underlying cyclic group G.

$\mathcal{E}$: on input a public key $pk \in G$ and a message $m \in G$, choose a value $s \leftarrow \{0, \ldots, q-1\}$ uniformly at random and return the ciphertext

$$(g^s, pk^s \cdot m, H(pk, m))$$

$\mathcal{D}$: given a secret key sk and a ciphertext $(c_1, c_2, c_3) \in G \times G \times G$, the plaintext is:

$$m' := \begin{cases} c_1^{-sk} \cdot c_2 & \text{, if } c_3 = H(pk, c_1^{-sk} \cdot c_2) \\ \perp & \text{, otherwise.} \end{cases}$$

Because of the poor use of the random oracle in the last ciphertext component, $\mathcal{P}_2$ obviosuly fails to offer IK-CPA or IND-CPA security.

However, letting $\mathcal{M}(pk)$ choose a plaintext $m \in_R G$ uniformly at random and assuming that the Decisional Diffie Hellman assumption holds in G, we obtain another encryption scheme $\mathcal{P}_2^{\mathrm{Rev}}$, operating in reverse mode (and has one-way security). The latter scheme is far less obvious to attack.

Our next example builds on an intentionally weakened Goldwasser-Micali scheme [GM84] and illustrates an RPKE variant with a single public key.

Example 3 (Reverse encryption without encryption). Consider the following public key encryption scheme which is obviously neither IND-CPA nor IK-CPA secure:

$\mathcal{G}$**:** (simply outputs the size of the public modulus n).

$\mathcal{K}$**:** choose a Blum integer $n = p \cdot q$ and

$$g \in \left(\frac{\mathbb{Z}}{n\mathbb{Z}}\right)^* \quad \text{with} \quad \left(\frac{g}{p}\right) = \left(\frac{g}{q}\right) = -1$$

and return $(sk, pk) := ((p, q), (g, n))$; the associated message space $\mathfrak{M}(pk)$ is the set $\{0, 1\}$ and can be naturally extended to strings of bits (by concatenation of ciphertexts of the string's bits).

$\mathcal{E}$**:** on input $pk = (g, n)$ and a plaintext $m \in \{0, 1\}$, choose at random

$$r \in \left(\frac{\mathbb{Z}}{n\mathbb{Z}}\right)^*$$

and return the triple

$$(pk, m, r^2 \cdot g^{1-m})$$

$\mathcal{D}$**:** decrypt a ciphertext (pk, m, h) as:

$$m' := \begin{cases} 1 & \text{, if } \left(\frac{h}{p}\right) = \left(\frac{h}{q}\right) = 1 \\ 0 & \text{, if } \left(\frac{h}{p}\right) = \left(\frac{h}{q}\right) = -1 \\ \perp & \text{, otherwise.} \end{cases}$$

The above scheme, which sends send the plaintext as well, is naturally not a good encryption scheme anymore.

Now, consider a reverse public key encryption mode of the scheme, using only a single public key $pk := (g, n)$ which can be defined as follows:

- Fix a public key $pk := (g, n)$ with a message sampling algorithm $\mathcal{M}(pk)$ selecting $m \in_R \{0, 1\}^k$ uniformly at random (excluding the all ones string string 1^k).
- To encrypt an individual bit b in the reverse mode, compute first the local "generator" for this encryption operation as $g' := g^{(b+1) \bmod 2}$ (namely either use g or 1 as the encrypting "generator"). Then, bit by bit, encrypt the random (but now fixed) message m to produce the ciphertext as in the traditional scheme (*i.e.*, when $g' = g$ is used, the

string m will produce a quadratic non-residue for a 0 and a quadratic residue for a 1 in the last ciphertext component; when $g' = 1$ is used, the string m is just producing k quadratic residues, rather than really encrypting the messsage: thus it is easy to recognize, given the factorization of n, which is the b encrypted this way).

– In this reverse mode, a ciphertext decrypts to 0 if the ciphertext is a (traditional) encryption of the all ones string (since $g' = 1$ was used as the generator) and to 1 if the ciphertext encrypts m, otherwise the decryption returns an invalidity symbol $\perp$.

Without the factorization it is hard to tell what b is unless the quadratic residuosity assumption fails. Thus the reverse mode gives us security while our ciphertext includes the plaintext m itself (no message privacy in the traditional mode).

The fact that we employ g or $g^0 = 1$ as the generator can, in fact, be viewed as two cyptosystems.

Finally, we outline a possible reverse mode of operation in the context of identity-based cryptography.

Example 4 (Reverse encryption in an identity-based setting). Consider Boneh and Franklin's BasicIdent scheme [BF01,BF03]. The public system parameters can be specified as a tuple $(q, G_1, G_2, \hat{e}, n, P, P_{\text{pub}}, H_1, H_2)$:

– G_1 and G_2 are groups of prime order q; we write G_1 additively (with neutral element 0) and G_2 multiplicatively
– $\hat{e} : G_1 \times G_1 \longrightarrow G_2$ is an admissible bilinear map
– n fixes the length of plaintexts
– P is a generator of G_1
– $P_{\text{pub}} = s \cdot P$ is a random multiple of P, where $s \in_R \{1, \ldots, q-1\}$
– $H_1 : \{0,1\}^* \longrightarrow G_1 \setminus \{0\}$ and $H_2 : G_2 \longrightarrow \{0,1\}^n$ are random oracles

Implement two instances ($i \in \{0,1\}$) of this scheme:

$$P_{\text{pub}}^{(i)} := s_i \cdot P \quad \text{and} \quad d_i := s_i \cdot H_1(\text{Alice})$$

where the d_i represent the secret keys corresponding to the identity Alice.

Motivated by a result of Holt [Hol06], we can define an RPKE mode of BasicIdent as follows: Bob sends a plaintext $b \in \{0,1\}$ to Alice by

"encrypting the fixed message 0^n" under $P_{\text{pub}}^{(b)}$. In other words, Bob picks a random $r \in_R \{1, \ldots, q-1\}$ and sends to Alice the ciphertext

$$c := \left(r \cdot P, H_2\left(\hat{e}(H_1(\text{Alice}), P_{\text{pub}}^{(b)})^r\right)\right)$$

To decrypt a ciphertext $c = (U, v)$ in reverse mode, Alice checks for which $b \in \{0, 1\}$ the condition $v = H_2(\hat{e}(d_b, U))$ holds.

4 RPKE and deniable encryption

Interestingly, RPKE may also find applications in settings where Alice and Bob share secret key material. In particular, we can use the method in the setting of deniable encryption (which we present here without a complete formalization).

Let (n, e) be Alice's RSA public key and denote by $d = e^{-1} \bmod \phi(n)$ Alice's secret key. We further assume that before interaction starts, Alice generates a secret κ, a random message $r \in_R \mathbb{Z}/n\mathbb{Z}$ and a random mask $s \in_R \{0, 1\}^k$.

κ is used as key for an information-theoretically secure message authentication code (MAC) denoted by f.

The values r and $t' := s \oplus f_\kappa(r)$ are given to Bob.

We consider two low bandwidth modes of operation for encrypting a plaintext bit $b \in \{0, 1\}$:

TPKE: In this mode Bob encrypts b by sending

$$c := (r^e \bmod n, t)$$

where r is a random integer such that $r \bmod 2 = b$, and $t \in_R \{0, 1\}^k$ is a randomly chosen tag .

RPKE: In this mode Bob encrypts b as in Example 1. Namely by sending $c := (r^{\text{FDH}(b)} \bmod n, t')$ where t' is the secret pre-agreed with Alice. After creating c, Bob erases r. Alice uses (r, s, κ) to determine which exponent was initially used by Bob.

Suppose that we allow concurrent usage of TPKE and RPKE. Namely, Bob sends b in one of the above modes without informing Alice in advance which of the two modes was used.

Alice can try both possible plaintexts in reverse mode and, should the redundancy checks succeed or fail, determine if TPKE was used. However,

Alice can still claim to a third party that RPKE was used: Alice selects a random bit $\tilde{b}$ and decrypts by exponentiation with $\text{FDH}(\tilde{b})^{-1} \bmod \phi(n)$ to obtain a random RSA plaintext $\tilde{r}$. To construct a matching redundancy check, Alice chooses a random MAC key $\tilde{\kappa}$, computes a matching tag $f_{\tilde{\kappa}}(\tilde{r})$ and opens the value $\tilde{s} := f_{\tilde{\kappa}}(\tilde{r}) \oplus t'$, which is consistent with t'.

Conversely, claiming that an RPKE-decrypted ciphertext is actually valid in TPKE mode is straightforward.

The above assumed a one-time operation based on a shared one time key. We can extend this to many time operation by producing the values from a pseudorandom generator that is one-way and erasing the values used in past operations. The parties share initial seeds and at a point they are asked to deny a message the generator's state does not remember the shared values. Deniability now relies on the fact that the values claimed are impossible to verify computationally.

5 Conclusion and further research

This paper presented a new public-key encryption paradigm, allowing to turn a weak form of key privacy into message privacy. While our discussion makes a first attempt to formalize the concept, it clearly falls short of presenting a thorough treatment of the subject. In fact, a number of questions arise which deserve further exploration. For instance, bandwidth could be improved if (in our definition of $\mathcal{P}^{\text{Rev}}$ in Section 2.1) a construction allowing to distinguish between the two $\perp$ sub-cases

$$\text{if } \tilde{m}_0 \in \text{Im}(\mathcal{M}(pk_0)) \text{ and } \tilde{m}_1 \in \text{Im}(\mathcal{M}(pk_1))$$

$$\text{if } \tilde{m}_0 \notin \text{Im}(\mathcal{M}(pk_0)) \text{ and } \tilde{m}_1 \notin \text{Im}(\mathcal{M}(pk_1))$$

could be exhibited. Devising such an extension is an interesting problem left unanswered by this paper. Combining such a construction with an n-key-pair scheme (as in Remark 1) would allow encoding in a linear size ciphertext an exponentially large plaintext space.

Reverse encryption and digital signatures. Another open question, related to reverse encryption, is the transformation of *certain* digital signature schemes into public-key encryption schemes: at the minimalist extreme we can regard a one-bit public-key encryption algorithm $\mathcal{E}$ as an algorithm that does not encrypt anything, but generates "valid strings" $c = \mathcal{E}(pk, r)$. A "valid string" is a string c that satisfies a confidential validity predicate

$\mathcal{D}(sk, c) = \texttt{true}$. Here "validity" stands for the transmission of the bit one whereas invalidity will stand for the transmission of the bit zero.[9]

Consider now a digital signature scheme with probabilistic signing algorithm $\mathcal{S}$ and deterministic verification algorithm $\mathcal{V}$ producing digital signatures on messages m such that:

$$\sigma \leftarrow \mathcal{S}(sk, m) \implies \mathcal{V}(pk, m, \sigma) = \texttt{true}$$

If—amongst other conditions—given sk one cannot infer pk (this is not a misprint) and if σ cannot be verified using sk alone, then $\mathcal{S}$ and $\mathcal{V}$ could potentially be turned into encryption and decryption algorithms of a public-key encryption scheme as follows: publish sk and give pk to the receiver (only). To encrypt a one, pick a random message and sign it. To encrypt a zero, send a random string. A different starting point could be a construction using two different types of redundancy in the message that is signed—say along the following line: use a message recovery signature scheme and either sign a message of the form $r \parallel H(r)$ or of the form $H(r) \parallel r$ with a randomly chosen r. The signature scheme has to be such that knowledge of pk is essential for being able to distinguish these cases (and given the secret key, it is hard to determine it).

Acknowledgment

We thank Eike Kiltz for several interesting comments regarding this work.

References

[BBDP01a] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-Privacy in Public-Key Encryption. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer-Verlag, 2001.

[BBDP01b] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-Privacy in Public-Key Encryption. Available at `http://cseweb.ucsd.edu/~mihir/papers/anonenc.html`, September 2001. Full version of [BBDP01a].

[BBW06] Adam Barth, Dan Boneh, and Brent Waters. Privacy in Encrypted Content Distribution Using Private Broadcast Encryption. In Giovanni Di Crescenzo and Aviel D. Rubin, editors, *Financial Cryptography and Data Security, 10th International Conference, FC 2006*, volume 4107 of *Lecture Notes in Computer Science*, pages 52–64. Springer-Verlag, 2006.

[9] Invalidity may either be achieved by just sending a random c which will be very probably invalid or by an "intelligent" construction of a deliberately invalid c.

[BF01] Dan Boneh and Matt Franklin. Identity-Based Encryption from the Weil Pairing. In J. Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.

[BF03] Dan Boneh and Matt Franklin. Identity-Based Encryption from the Weil Pairing. *SIAM J. of Computing*, 32(3):586–615, 2003. Extended abstract appeared in [BF01].

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable Encryption. In *Advances in Cryptology — CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104. Springer-Verlag, 1997.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, April 1984.

[Hal05] Shai Halevi. A sufficient condition for key-privacy. Cryptology ePrint Archive: Report 2005/005, January 2005. Available at `http://eprint.iacr.org/2005/005`.

[Hol06] Jason E. Holt. Key Privacy for Identity Based Encryption. Internet Security Research Lab Technical Report 2006-2, Internet Security Research Lab, Brigham Young University, March 2006. Available at `http://isrl.cs.byu.edu/pubs/isrl-techreport-2006-2.pdf`.

[HOT04] Ryotaro Hayashi, Tatsuaki Okamoto, and Keisuke Tanaka. An RSA Family of Trap-Door Permutations with a Common Domain and Its Applications. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *Public Key Cryptography – PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 291–304. Springer-Verlag, 2004.

[KTY07] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Group Encryption. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 181–199. Springer-Verlag, 2007.

[RS92] Charles Rackoff and Daniel R. Simon. Non-interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer-Verlag, 1992.

[YCK+07] Shigenori Yamakawa, Yang Cui, Kazukuni Kobara, Manabu Hagiwara, and Hideki Imai. On the Key-Privacy Issue of McEliece Public-Key Encryption. In Serdar Boztaş and Hsiao-Feng (Francis) Lu, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, AAECC-17*, volume 4851 of *Lecture Notes in Computer Science*, pages 168–177. Springer-Verlag, 2007.

[ZHI07] Rui Zhang, Goichiro Hanaoka, and Hideki Imai. Orthogonality between Key Privacy and Data Privacy, Revisited. In Dingyi Pei, Moti Yung, Dongdai Lin, and Chuankun Wu, editors, *Information Security and Cryptology – Inscrypt 2007*, volume 4990 of *Lecture Notes in Computer Science*, pages 313–327. Springer-Verlag, 2007. Preliminary full version available at `http://staff.aist.go.jp/r-zhang/research/KeyPrivacy.pdf`.