Deniable encryption, authentication, and key exchange

Dimiter Ostrev

Abstract

We present some foundational ideas related to deniable encryption, message authentication, and key exchange in classical cryptography. We give detailed proofs of results that were previously only sketched in the literature. In some cases, we reach the same conclusions as in previous papers; in other cases, the focus on rigorous proofs leads us to different formulations of the results.

1 Introduction

Encryption, message authentication and key exchange are important primitives in cryptography. In this introduction, we give a high level idea of deniability for these primitives. We use the traditional names Alice and Bob for two honest users, and Eve for the adversary. We focus on the case of asymmetric keys.

The basic security notion for encryption gives to Eve the following powers: the ability to perform probabilistic polynomial time computation, and the ability to obtain cyphertexts for messages of her choice. Eve is then allowed to select two messages of her choice, one of which is encrypted and the cyphertext given to her. It is required that Eve be unable to guess which of the two messages was used with probability significantly greater than 1/2.

The notion of deniable encryption extends the basic notion by giving Eve an additional power: to approach the creator of the cyphertext, demand to see the message and random coins used, and check for consistency with the cyphertext she observed. The encryption scheme is called deniable if the creator of the cyphertext can lie at this point in a way that is undetectable by any efficient algorithm.

Do deniable encryption schemes exist? If they do, why would one be interested in them? We give three answers, going from more theoretical to more practical.

The first reason to be interested in deniable encryption is that it is a new mathematical puzzle that presents a challenge: if a given cyphertext can consistently match two different messages, then why does the intended receiver of the cyphertext not get confused between the two messages?

The second reason to be interested in deniable encryption is that it is a useful primitive for constructing some secure multi-party computation protocols.

The third reason is that in certain real-world scenarios, our intuition says that something like deniable encryption would be required. Such scenarios include, for example, Alice voting electronically, and then Eve, a coercer or vote buyer, approaching Alice and demanding to see how she voted. A second example would be Alice storing encrypted data on her computer, and Eve approaching Alice and demanding that Alice reveal the stored data.

We move now to message authentication. We begin our discussion with an example: a protocol that allows Alice to authenticate a message to Bob:

$$\begin{array}{ccc} \text{Alice} & & \text{Bob} \\ & \xrightarrow{m} & \\ & \xleftarrow{m, enc(pk_A, k)} & \\ & \xrightarrow{m, tag(k, m)} & \end{array} \tag{1}$$

Here, $enc(pk_A, \cdot)$ is encryption under Alice's public key, and k is a randomly chosen key for a symmetric key message authentication scheme drawn by Bob as a challenge for Alice. The intuition for the security of this protocol is that only Alice could see k, and so only she can create the correct message authentication tag under k.

While Bob is convinced that Alice sent him m, he cannot convince any third party that such is the case. This is because Bob can produce the entire transcript of a protocol execution for any message by himself. Therefore, we say that this protocol provides deniable public key message authentication.

Thus, we have two modes of using asymmetric cryptography for message authentication. A digital signature by Alice would assure Bob that Alice intended to say m to him, and would also be verifiable by any third party. A deniable message authentication protocol would again assure Bob that Alice intended to say m to him, but would not present evidence to any third party.

When would deniable message authentication be useful? [7] gives one example: a software producer wants to prove to paying customers that their copy of the software is legitimate, and at the same time not allow them to make pirate copies and sell them as original.

Other examples when deniable authentication would be useful are whistleblowing and off-the-record messaging. A recent press release from the Council of the European Union [1] contains the following definition

> Whistle-blowers are people speaking up when they encounter, in the context of their work, wrongdoing that can harm the public interest, for instance by damaging the environment, public health and consumer safety and public finances.

We can see that the ability to report wrongdoing without fear of retribution is important for a democratic society.

Finally, we discuss key exchange. The motivation for considering deniable key exchange stems from the motivation for deniable authentication, combined with the use of key exchange protocols to establish secure channels. Indeed, consider the following common cryptographic scenario: Alice and Bob, who each have a public key-secret key pair, wish to establish a secure communications session. They run a key exchange protocol, and then use the resulting key for symmetric key authenticated encryption of messages. Now, suppose that Alice and Bob wish to have not only secrecy and integrity for their secure communication session, but also deniability. Then, the key exchange protocol they used to establish the session must also be deniable.

The rest of this paper is structured as follows: in section 2, we discuss some background material from cryptography that will be useful in subsequent sections. In section 3 we discuss variations on the basic security notions; having these variations will simplify the arguments in subsequent sections. In section 4 we discuss the first proposed construction that achieves deniable encryption. In section 5, we show that our example of deniable authentication (1) is secure as a message authentication protocol. We also give a definition of deniable authentication and show that protocol (1) satisfies it under appropriate assumptions. Finally, in section 6 we discuss deniability for key exchange protocols. We show that the basic Diffie Hellman mechanism appropriately combined with (1) for authentication is secure as a key exchange protocol, and is also deniable under appropriate assumptions.

We conclude the introduction with brief remarks on the contributions of this work. We give detailed definitions, theorem statements and proofs. In some cases, we reach the same conclusions as the papers where the results were first presented. In other cases, our conclusions differ. The most notable of these concerns deniability of key exchange: we find that we require a different assumption to rigorously prove deniability of the SKEME key exchange protocol. This leads us to formulate a new security notion for asymmetric encryption, which we call multi-user PA1 plaintext awareness. The relation of this multi-user notion to the existing, single-user versions of plaintext awareness is left as an open question for future work. More details are given in section 6.

2 Background

In this section, we cover asymmetric encryption (subsection 2.1), symmetric key message authentication (subsection 2.2) and the Decisional Diffie Hellman assumption (subsection2.3); all of these will be used in subsequent sections.

2.1 Asymmetric encryption

A public key encryption scheme ENC consists of three algorithms (kg, enc, dec). kg takes as input a security parameter in unary, and returns a pair pk, sk. enc takes as input a public key and a message, and outputs a ciphertext. dec takes as input a secret key and a ciphertext, and outputs a message. kg, enc run in probabilistic polynomial time, and dec runs in deterministic polynomial time.

Most of the time, we require ENC to satisfy perfect correctness; that is, we require

$$\forall n \forall m,\ Pr((pk, sk) \leftarrow kg(1^n), c \leftarrow enc(pk, m) : dec(sk, c) = m) = 1$$

In the present paper, we will also have to consider schemes where decryption is correct with high probability, rather than perfect. More details are given in the corresponding section.

Given an encryption scheme ENC and adversary strategy A, the chosen plaintext attack security experiment proceeds as follows:

1. $(pk, sk) \leftarrow kg(1^n)$, and pk is given to A
2. A produces m_0, m_1.
3. $b \leftarrow \{0, 1\},\ c \leftarrow enc(pk, m_b)$, c is given to A.
4. A outputs a bit b'. We say here and afterwards leave implicit that A is allowed to carry an internal state of its choice from the end of step 2 to the beginning of step 4.

It is also possible to think of steps 1. and 3. as being performed by a challenger for the CPA experiment.

Definition 1. *Given an encryption scheme ENC and an adversary strategy A, the advantage of A in the CPA experiment against ENC is*

$$\begin{aligned} AdvCPA(A, ENC) = 2 * Pr((pk, sk) \leftarrow kg(1^n), (m_0, m_1) \leftarrow A(pk), \\ b \leftarrow \{0, 1\}, c \leftarrow enc(pk, m_b), b' \leftarrow A(c) : b = b') - 1 \end{aligned}$$

We say that ENC is IND-CPA secure if for all probabilistic polynomial time adversary strategies A, $AdvCPA(A, ENC)$ is a negligible function of the security parameter.

The chosen ciphertext attack proceeds as the chosen plaintext attack, but in addition, A is given access to a decryption oracle. Two variations are considered in the literature: CCA1, or CCA-pre, in which A is given access to the decryption oracle only during step 2, and CCA2, or CCA-post, in which A is given access to the decryption oracle both during step 2 and step 4 (during step 4, query of the challenge ciphertext to the decryption oracle is not allowed). In the present paper, we will use the CCA2 version.

Definition 2. *Given an encryption scheme ENC and an adversary strategy A, the advantage of A in the IND-CCA2 experiment against ENC is*

$$\begin{aligned} AdvCCA2(A, ENC) = 2 * Pr((pk, sk) \leftarrow kg(1^n), (m_0, m_1) \leftarrow A^{dec(sk)}(pk), \\ b \leftarrow \{0, 1\}, c \leftarrow enc(pk, m_b), b' \leftarrow A^{dec(sk)}(c) : b' = b) - 1 \end{aligned}$$

We say that ENC is IND-CCA2 secure if for all probabilistic polynomial time adversary strategies A, $AdvCCA2(A, ENC)$ is a negligible function of the security parameter.

For some proofs, it is more convenient to have an equivalent formulation of CCA2 security. Here, we think of A as interacting with a single computational entity, which we call $CH - CCA2$. $CH - CCA2$ takes as input the random bit b, performs the key generation, gives pk to A, answers A's decryption queries, produces the challenge ciphertext $c \leftarrow enc(pk, m_b)$. We also think of the bit b' produced by A as the output of the interaction $[A, CH - CCA2(b)]$. Thus, we have rewritten the experiment

$$(pk, sk) \leftarrow kg(1^n), (m_0, m_1) \leftarrow A^{dec(sk)}(pk),$$
$$b \leftarrow \{0,1\}, c \leftarrow enc(pk, m_b), b' \leftarrow A^{dec(sk)}(c)$$

as

$$b \leftarrow \{0,1\}, b' \leftarrow [A, CH - CCA2(b)]$$

Further, we can rewrite the advantage of A as follows:

$$\begin{aligned} AdvCCA2(A, ENC) &= 2{*}Pr(b \leftarrow \{0,1\}, b' \leftarrow [A, CH{-}CCA2(b)] : b' = b){-}1 \\ &= Pr([A, CH - CCA2(1)] = 1) + Pr([A, CH - CCA2(0)] = 0) - 1 \\ &= Pr([A, CH - CCA2(1)] = 1) - Pr([A, CH - CCA2(0)] = 1) \end{aligned}$$

The alternative formulation emphasizes the role of A as a distinguisher trying to tell whether it is interacting with $CH - CCA2(0)$ or with $CH - CCA2(1)$.

2.2 Symmetric key message authentication

A symmetric key message authentication scheme MAC consists of three algorithms: (kg, tag, vrf). kg takes as input a security parameter in unary and outputs a key k. tag takes as input a key and a message and outputs an authentication tag. vrf takes as input a key, message and authentication tag and outputs a decision: accept or reject. kg, tag run in probabilistic polynomial time, and vrf runs in deterministic polynomial time.

Most of the time, we require MAC to satisfy perfect correctness:

$$\forall n, \forall m,\ Pr(k \leftarrow kg(1^n), t \leftarrow tag(k, m) : vrf(k, m, t) = 1) = 1$$

Given a symmetric key message authentication scheme MAC and adversary strategy A, the chosen message attack experiment proceeds as follows:

1. $k \leftarrow kg(1^n)$
2. A is allowed to adaptively query $tag(k, \cdot)$ on messages of its choice and $vrf(k, \cdot, \cdot)$ on message-tag pairs of its choice.

The interaction of A and MAC produces a transcript of A's queries and the corresponding responses; we denote this by $T \leftarrow [A, MAC]$. T is a random variable.

We define $B(T)$ to be the event that there exist m, t such that the transcript T contains a vrf query with input m, t and output 1 and no prior tag query with input m and output t. Thus, the event $B(T)$ captures what we intuitively perceive as the adversary breaking message authentication.

Definition 3. *Given a message authentication scheme MAC and adversary strategy A, the advantage of A in a chosen message attack against MAC is*

$$AdvCMA(A, MAC) = Pr(T \leftarrow [A, MAC] : B(T))$$

We say that MAC is EUF-CMA secure if for all probabilistic polynomial time strategies A, $AdvCMA(A, MAC)$ is a negligible function of the security parameter.

2.3 The Decisional Diffie Hellman assumption

In the present paper, we use the Decisional Diffie Hellman assumption in constructing translucent sets (subsection 4.2) and in proving the security of SKEME (subsection 6.3).

Definition 4. *A group generator is a probabilistic polynomial time algorithm GroupGen that takes as input a security parameter 1^n in unary, and outputs a triple (G, q, g) where G is a finite cyclic group of prime order q and g is a generator of G.*

Conjecture 1 (Decisional Diffie Hellman assumption)**.** *There exists a group generator GroupGen such that for all efficient D*

$$\begin{aligned} &AdvDDH(D, GroupGen) \\ &= |Pr((G,q,g) \leftarrow GroupGen(1^n), (x,y) \leftarrow Z_q^2, h_1 \leftarrow g^x, h_2 \leftarrow g^y, h_3 \leftarrow g^{xy} \\ &\qquad : D(G,q,g,h_1,h_2,h_3) = 1) \\ &\qquad - Pr((G,q,g) \leftarrow GroupGen(1^n), (h_1,h_2,h_3) \leftarrow G^3 \\ &\qquad\qquad : D(G,q,g,h_1,h_2,h_3) = 1)| \end{aligned} \tag{2}$$

is a negligible function of n.

3 Variations of the basic notions: multiple challenges and multiple keys

In the previous section, we presented the basic notions of security for public key encryption and symmetric key message authentication. In order to better prepare for the arguments in subsequent sections, we present the natural extension of these notions to the case of multiple challenges and multiple keys. We begin in subsection 3.1 by discussing an IND-CCA2 security experiment that allows an adversary to adaptively ask for multiple challenge cyphertexts. Then, in subsection 3.2, we consider an experiment in which the adversary attacks multiple key pairs, and can adaptively ask for multiple challenge cyphertexts under each pair. Finally, in subsection 3.3, we discuss a variation of the EUF-CMA security experiment in which the adversary is trying to break at least one out of multiple keys.

3.1 The IND-CCA2 security experiment with one key pair and multiple challenges

Let $ENC = (kg, enc, dec)$ be an asymmetric encryption scheme and let $N(n)$ be a polynomially bounded efficiently computable function. Let A be an adversary strategy that asks for at most $N(n)$ challenge cyphertexts. The variation of the IND-CCA2 security experiment with up to $N(n)$ challenge cyphertexts proceeds as follows:

1. CH receives as input the security parameter 1^n and a vector $\vec{b} \in \{0,1\}^{N(n)}$.
2. CH draws $(pk, sk) \leftarrow kg(1^n)$ and gives pk to A.
3. CH answers A's decryption queries.
4. If A submits the i-th request for a challenge cyphertext, with messages $m_{i,0}, m_{i,1}$, CH draws $c_i \leftarrow enc(pk, m_{i,b_i})$ and replies with c_i. None of the challenge cyphertexts can be queried to the decryption oracle.
5. At the end, A outputs a bit b'. We consider b' as the output of the interaction $[A, CH(\vec{b})]$.

We define the N-challenge advantage of A as

$$AdvCCA2_N(A, ENC) = Pr([A, CH(1^N)] = 1) - Pr([A, CH(0^N)] = 1) \quad (3)$$

The hybrid argument allows us to relate the advantage for multiple challenges to the advantage for a single challenge.

Theorem 1. *Let $ENC = (kg, enc, dec)$ be an asymmetric encryption scheme. Let $N(n)$ be a polynomially bounded efficiently computable function. Let A be an efficient adversary strategy that asks for at most N challenges. Then, there exists efficient adversary strategy A' that asks for at most one challenge such that*

$$AdvCCA2_N(A, ENC) = N * AdvCCA2(A', ENC)$$

Proof. The main idea of the hybrid argument is contained in the equation

$$\begin{aligned} &Pr([A, CH(1^N)] = 1) - Pr([A, CH(0^N)] = 1) \\ &\qquad = \sum_{i=1}^{N} Pr([A, CH(1^i 0^{N-i})] = 1) - Pr([A, CH(1^{i-1} 0^{N-i+1})] = 1) \end{aligned}$$

This suggests to take algorithm A' to be the following:

1. Receive pk from CH.
2. Pass pk to subroutine A.
3. Answer A's decryption queries using CH.

4. Draw $l \leftarrow \{1, \ldots, N\}$.

5. For $i = 1, \ldots, l-1$, for the i-th of A's challenge requests $(m_{i,0}, m_{i,1})$, answer with $c_i \leftarrow enc(pk, m_{i,1})$.

6. For the l-th challenge request of A, answer using CH.

7. For $i = l+1, \ldots, N$, for the i-th of A's challenge requests $(m_{i,0}, m_{i,1})$, answer with $c_i \leftarrow enc(pk, m_{i,0})$.

8. When A outputs b', output b'.

Then, A' asks for at most one challenge and, for $i = 1, \ldots, N$,

$$Pr([A', CH(1)] = 1 | l = i) = Pr([A, CH(1^i 0^{N-i})] = 1)$$
$$Pr([A', CH(0)] = 1 | l = i) = Pr([A, CH(1^{i-1} 0^{N-i+1})] = 1)$$

Then,

$$AdvCCA2(A', ENC)$$
$$= \sum_{i=1}^{N} Pr(l = i)(Pr([A', CH(1)] = 1 | l = i) - Pr([A', CH(0)] = 1 | l = i))$$
$$= \frac{1}{N} \sum_{i=1}^{N} (Pr([A, CH(1^i 0^{N-i})] = 1) - Pr([A, CH(1^{i-1} 0^{N-i+1})] = 1))$$
$$= \frac{1}{N}(Pr([A, CH(1^N)] = 1) - Pr([A, CH(0^N)] = 1)) = \frac{1}{N} AdvCCA2_N(A, ENC)$$

which proves the theorem. □

3.2 The IND-CCA2 security experiment with multiple key pairs and multiple challenges

Let $ENC = (kg, enc, dec)$ be an asymmetric encryption scheme and let $N(n)$, $N'(n)$ be polynomially bounded efficiently computable functions. Let A be an adversary strategy that takes as input the security parameter 1^n and a vector of $N'(n)$ public keys and asks for at most $N(n)$ challenge cyphertexts under each public key. The variation of the IND-CCA2 security experiment with $N'(n)$ key pairs and up to $N(n)$ challenge cyphertexts per pair proceeds as follows:

1. CH receives as input the security parameter 1^n and a matrix $B \in \{0,1\}^{N' \times N}$.

2. For $i = 1, \ldots N'(n)$, CH draws $(pk_i, sk_i) \leftarrow kg(1^n)$. CH gives $\vec{pk} = (pk_1, \ldots pk_{N'})$ to A.

3. CH answers A's decryption queries.

4. If A submits the i-th request for a challenge cyphertext to the j-th public key, with messages $m_{j,i,0}, m_{j,i,1}$, CH draws $c_{j,i} \leftarrow enc(pk, m_{j,i,B_{j,i}})$ and replies with $c_{j,i}$. None of the challenge cyphertexts can be queried to the decryption oracle.

5. At the end, A outputs a bit b'. We consider b' as the output of the interaction $[A, CH(B)]$.

We define the N'-user N-challenge advantage of A as

$$AdvCCA2_{N',N}(A, ENC) = Pr([A, CH(1^{N' \times N})] = 1) - Pr([A, CH(0^{N' \times N})] = 1) \quad (4)$$

Again, a hybrid argument allows us to relate the advantage for multiple users and multiple challenges to the advantage for a single user and multiple challenges.

Theorem 2. *Let $ENC = (kg, enc, dec)$ be an asymmetric encryption scheme. Let $N(n), N'(n)$ be polynomially bounded efficiently computable functions. Let A be an efficient adversary strategy for the N' user N challenge case. Then, there exists efficient adversary strategy A' for a single user and multiple challenges such that*

$$AdvCCA2_{N',N}(A, ENC) = N' * AdvCCA2_N(A', ENC)$$

Proof. Take algorithm A' to be the following:

1. Receive pk from CH.

2. Draw $l \leftarrow \{1, \ldots, N\}$.

3. For $i = 1, \ldots, l-1, l+1, \ldots N$, draw $(pk_i, sk_i) \leftarrow kg(1^n)$.

4. Pass $\vec{pk}$ to subroutine A.

5. Answer A's decryption queries for user l using CH. Answer decryption queries for other users using the corresponding secret key.

6. For $j = 1, \ldots, l-1$, for $i = 1, \ldots N$, answer the (j, i)-th of A's challenge requests $(m_{j,i,0}, m_{j,i,1})$, with $c_{j,i} \leftarrow enc(pk, m_{j,i,1})$.

7. For challenge requests to the l-th user, answer using CH.

8. For $j = l+1, \ldots, N'$, for $i = 1, \ldots N$, answer the (j, i)-th of A's challenge requests $(m_{j,i,0}, m_{j,i,1})$, with $c_{j,i} \leftarrow enc(pk, m_{j,i,0})$.

9. When A outputs b', output b'.

Then, A' participates in the single user N-challenge IND-CCA2 experiment, and, for $j = 1, \ldots, N'$,

$$Pr([A', CH(1^N)] = 1 | l = j) = Pr([A, CH((1^N)^j (0^N)^{N'-j})] = 1)$$
$$Pr([A', CH(0^N)] = 1 | l = j) = Pr([A, CH((1^N)^{j-1} (0^N)^{N'-j+1})] = 1)$$

Then,

$$\begin{aligned}
&AdvCCA2_N(A', ENC) \\
&= \sum_{j=1}^{N'} Pr(l = j)(Pr([A', CH(1^N)] = 1 | l = j) - Pr([A', CH(0^N)] = 1 | l = j)) \\
&= \frac{1}{N'} \sum_{j=1}^{N'} (Pr([A, CH((1^N)^j (0^N)^{N'-j})] = 1) - Pr([A, CH((1^N)^{j-1} (0^N)^{N'-j+1})] = 1)) \\
&= \frac{1}{N'} (Pr([A, CH(1^{N' \times N})] = 1) - Pr([A, CH(0^{N' \times N})] = 1)) \\
&= \frac{1}{N'} AdvCCA2_{N', N}(A, ENC)
\end{aligned}$$

which proves the theorem. □

3.3 The EUF-CMA security experiment with multiple keys

Let $MAC = (kg, tag, vrf)$ be a symmetric key message authentication scheme. Let $N(n)$ be a polynomially bounded efficiently computable function. Let A be an adversary strategy. The EUF-CMA security experiment with $N(n)$ users proceeds as follows:

1. For $i = 1, \ldots N$, draw $k_i \leftarrow kg(1^n)$.
2. A is allowed to submit queries to $tag(k_i), vrf(k_i)$ for $i = 1, \ldots N$. A transcript T of all queries is recorded.

For $i = 1, \ldots, N$ let $B_i(T)$ be the event that there is a query (m, t) to $vrf(k_i)$ that passes verification, and there is no prior query to $tag(k_i)$ with input m and output t. Let $B(T) = \cup_{i=1}^{N} B_i(T)$. The N-user EUF-CMA advatange of A is

$$AdvCMA_N(A, MAC) = Pr(T \leftarrow [A, CH] : B(T))$$

This time, the union bound allows us to relate the advantage in the multi-key experiment to the advantage against a single secret key.

Theorem 3. *Let $MAC = (kg, tag, vrf)$ be a symmetric key message authentication scheme. Let $N(n)$ be a polynomially bounded efficiently computable function. Let A be an efficient adversary strategy for the N-user EUF-CMA experiment. Then, there is an efficient adversary strategy A' for the single user EUF-CMA experiment such that*

$$AdvCMA_N(A, MAC) \leq N * AdvCMA(A', MAC)$$

Proof. Take A' to be the following

1. Draw $l \leftarrow \{1, \dots N\}$.
2. For $i = 1, \dots, l-1, l+1, \dots, N$, draw $k_i \leftarrow kg(1^n)$.
3. Use A as a subroutine. Answer A's queries for user l by passing them to the single user CH. Answer A's queries to any user $i \neq l$ using k_i.

This experiment generates two transcripts: a transcript T of the queries at the interface between A and A' and a transcript T' of the queries at the interface between A' and the single user CH. Moreover, we have, for $i = 1, \dots, N$,

$$Pr(B(T')|l = i) = Pr(B_i(T))$$

Then,

$$\begin{aligned} AdvCMA(A', MAC) &= \sum_{i=1}^{N} Pr(l = i) Pr(B(T')|l = i) \\ &= \frac{1}{N} \sum_{i=1}^{N} Pr(B_i(T)) \geq \frac{1}{N} Pr(B(T)) = \frac{1}{N} AdvCMA_N(A, MAC) \end{aligned}$$

which proves the theorem. □

4 Deniable Encryption

Here, we follow [4]. Alice wants to send an encrypted message to Bob. After observing the ciphertext, Eve demands to see the private random values that Alice used during encryption. Alice wants to be able to give fake random values at this point, and lie about the message that was actually transmitted. We give a specific construction that matches this intuition, and we state and prove formally the properties it achieves in Theorem 4.

We begin this section by defining a primitive called translucent sets (subsection 4.1), and showing how to achieve this primitive under the Decisional Diffie Hellman assumption (subsection 4.2). Then, we construct the parity scheme for deniable encryption from the translucent sets primitive (subsection 4.3), and formally state and prove the security and deniability properties of the parity scheme (subsection 4.4). We conclude this section with some results demonstrating the limits of the design approach underlying the parity scheme (subsection 4.5).

4.1 Translucent sets

The authors of [4] propose to achieve deniable encryption based on a primitive that they call translucent sets. We describe this primitive:

Definition 5. *The primitive translucent sets consists of the following functionalities:*

1. *Key generation: On input $t \in \mathbb{N}$, outputs a pair $(S_t(), V_t())$ at random from the set of acceptable pairs for security parameter t.*

2. *Generation of random translucent set elements: the algorithm $S_t()$ outputs an element of a certain subset $Im(S_t) \subset \{0,1\}^t$. The outputs of $S_t()$ have the uniform distribution over this set.*

3. *There exists a negligible function ϵ such that for all efficient distinguishers D, for all t,*

$$|Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow S_t() : D(x, S_t) = 1) \\ - Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow U_t() : D(x, S_t) = 1)| < \epsilon(t)$$

   where U_t is the algorithm that outputs uniformly random elements of $\{0,1\}^t$.

4. *There exists a negligible function δ such that for all t*

$$Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow S_t() : V_t(x) = 1) > 1 - \delta(t)$$
$$Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow U_t() : V_t(x) = 1) < \delta(t)$$

4.2 Construction of translucent sets

We present a simple construction of translucent sets based on the Decisional Diffie-Hellman assumption (subsection 2.3). Let *GroupGen* be the efficient generator that is conjectured to exist in the DDH assumption (Conjecture 1). Now, we construct translucent sets as follows:

1. Key generation proceeds as follows:
   (a) $(G, q, g) \leftarrow GroupGen(1^t)$.
   (b) $x \leftarrow Z_q, X \leftarrow g^x$.
   (c) $pk \leftarrow (G, q, g, X), sk \leftarrow (G, q, g, x)$
   (d) Output the algorithm S associated to pk and the algorithm V associated to sk (explained below).

2. The algorithm S associated to (G, q, g, X) proceeds as follows: $y \leftarrow Z_q$, output (g^y, X^y).

3. The algorithm V associated to (G, q, g, x) proceeds as follows: on input (h_1, h_2), if $h_1^x = h_2$ output 1, else output 0.

Property 3. of Definition 5 follows from equation (2) of the Decisional Diffie Hellman assumption. Property 4. of Definition 5 holds for any function $\delta(t)$ such that $\delta(t) > 1/min(q(t))$ for all t, where $min(q(t))$ denotes a lower bound on the size of primes output by *GroupGen* on input security parameter t.

4.3 The parity scheme

The parity scheme uses the translucent sets primitive to achieve deniable encryption. Choose odd $n \in \mathbb{N}$, a parameter that governs the level of deniability that is achieved. Then, the parity scheme works as follows:

- Key generation: Bob uses the key generation algorithm of translucent sets and obtains $(S_t, V_t) \leftarrow KG(t)$. S_t is Bob's public key, and V_t is Bob's secret key.

- Encryption: to send bit b to Bob, Alice obtains Bob's public key S_t and then does the following:

  1. If $b = 0$, choose a random even $k \in \{0, 2, \ldots, n-1\}$, and if $b = 1$ choose a random odd $k \in \{1, 3, \ldots, n\}$.
  2. For $i = 1, \ldots, k$, $c_i \leftarrow S_t()$, and for $i = k+1, \ldots, n$, $c_i \leftarrow U_t()$.
  3. Send $c = (c_1, \ldots c_n)$ to Bob.

- Decryption: on receiving c, Bob does the following:

  1. For $i = 1, \ldots, n$, $b_i \leftarrow V_t(c_i)$.
  2. $b \leftarrow \sum_i b_i \mod 2$.

- Sender deniability: to claim that c corresponds to an encryption of $1 - b$, Alice does the following:

  1. Instead of k, Alice reveals $k - 1$. If $k = 0$, the faking algorithm fails.
  2. Alice reveals the randomness $r_1, \ldots r_{k-1}$, used for generating $c_1, \ldots c_{k-1}$. Then, Alice claims that c_k was chosen uniformly at random from $\{0,1\}^t$, and honestly reveals that $c_{k+1}, \ldots, c_n$ are chosen uniformly at random from $\{0,1\}^t$.

4.4 Properties of the parity scheme

Theorem 4. *The parity scheme achieves the properties below. Each of the properties holds for all t.*

1. *Correctness:*

$$\forall b, \;\; Pr((S_t, V_t) \leftarrow KG(t), c \leftarrow Enc(S_t, b), b' \leftarrow Dec(V_t, c) : b' \neq b) < n\delta(t)$$

2. *IND-CPA security: for all efficient distinguishers D,*

$$\begin{aligned}|Pr((S_t, V_t) \leftarrow KG(t), c \leftarrow Enc(S_t, 0) : D(S_t, c) = 1)\\ - Pr((S_t, V_t) \leftarrow KG(t), c \leftarrow Enc(S_t, 1) : D(S_t, c) = 1)| < \epsilon(t)\end{aligned}$$

3. *$(2/(n+1)+\epsilon(t))$-sender deniability: first, Eve, cannot distinguish between an honest reveal of an encryption of 0 and a faked reveal of an encryption of 1 as an encryption of 0; that is, for all efficient distinguishers D,*

$$\begin{aligned}
&|Pr((S_t,V_t) \leftarrow KG(t), k \leftarrow Even_{\leq n}, r \leftarrow R^{\otimes k}(), c \leftarrow Enc(S_t, 0, k, r) \\
&\qquad : D(S_t, c, k, r) = 1) \\
&- Pr((S_t,V_t) \leftarrow KG(t), k' \leftarrow Odd_{\leq n}, r' \leftarrow R^{\otimes k'}(), c \leftarrow Enc(S_t, 1, k', r'), \\
&\qquad k \leftarrow k' - 1, r \leftarrow (r'_1, \ldots, r'_k) : D(S_t, c, k, r) = 1)| < \epsilon(t)
\end{aligned}$$

where $R()$ is the algorithm that tosses coins for the algorithm $S_t()$, and where $R^{\otimes k}()$ means running $R()$ independently k times.

Second, Eve cannot distinguish between an honest reveal of an encryption of 1 and a faked reveal of an encryption of 0 as an encryption of 1: for all efficient D,

$$\begin{aligned}
&|Pr((S_t,V_t) \leftarrow KG(t), k \leftarrow Odd_{\leq n}, r \leftarrow R^{\otimes k}(), c \leftarrow Enc(S_t, 1, k, r) \\
&\qquad : D(S_t, c, k, r) = 1) \\
&- Pr((S_t,V_t) \leftarrow KG(t), k' \leftarrow Even_{\leq n}, r' \leftarrow R^{\otimes k'}(), c \leftarrow Enc(S_t, 0, k', r'), \\
&\qquad k \leftarrow k' - 1, r \leftarrow (r'_1, \ldots, r'_k) : D(S_t, c, k, r) = 1)| < \frac{2}{n+1} + \epsilon(t)
\end{aligned}$$

Remark: Note that increasing n improves the deniability but makes worse the effort required for encryption and decryption and the correctness of the scheme.

Proof. Take any t.

First, we show correctness. The event that $b = b'$ contains the event that each c_i is decrypted correctly. Then,

$$\begin{aligned}
Pr(b' \neq b) &\leq Pr(\text{some } c_i \text{ is decrypted incorrectly}) \\
&\leq \sum_{i=1}^{n} Pr(c_i \text{ is decrypted incorrectly}) < n\delta(t)
\end{aligned}$$

where we have used the union bound, then the fourth property of translucent sets.

Next, we show secrecy. Given D that can distinguish between encryptions of zero and one, we construct D' that can distinguish between elements of the translucent set and random elements of $\{0,1\}^t$. Specifically, D' works as follows:

1. On input (x, S_t), choose random even $k \in \{0, 2, \ldots, n-1\}$.
2. For $i = 1, \ldots, k$, $c_i \leftarrow S_t()$.
3. $c_{k+1} \leftarrow x$.
4. For $i = k+2, \ldots, n$, $c_i \leftarrow U_t()$.

5. $b \leftarrow D(S_t, c)$.

6. Output b.

Note that the events

$$(S_t, V_t) \leftarrow KG(t), x \leftarrow U_t() : D'(x, S_t) = 1$$

and

$$(S_t, V_t) \leftarrow KG(t), c \leftarrow Enc(S_t, 0) : D(S_t, c) = 1$$

are equivalent. The same holds for the events

$$(S_t, V_t) \leftarrow KG(t), x \leftarrow S_t() : D'(x, S_t) = 1$$

and

$$(S_t, V_t) \leftarrow KG(t), c \leftarrow Enc(S_t, 1) : D(S_t, c) = 1$$

Therefore,

$$\begin{aligned}
&|Pr((S_t, V_t) \leftarrow KG(t), c \leftarrow Enc(S_t, 0) : D(S_t, c) = 1) - \\
&\qquad Pr((S_t, V_t) \leftarrow KG(t), c \leftarrow Enc(S_t, 1) : D(S_t, c) = 1)| \\
&\qquad = |Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow U_t() : D'(S_t, x) = 1) \\
&\qquad\qquad - Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow S_t() : D'(S_t, x) = 1)| < \epsilon(t)
\end{aligned}$$

as needed.

Next, we show $(2/(n+1) + \epsilon(t))$-sender deniability. Consider first the case of honest reveal of encryption of 0 and faked reveal of an encryption of 1 as an encryption of 0. The experiment corresponding to an honest reveal of an encryption of zero is the following:

1. $(S_t, V_t) \leftarrow KG(t)$.
2. $k \leftarrow \{0, 2, \ldots, n-1\}$.
3. $r \leftarrow R^{\otimes k}()$.
4. For $i = 1, \ldots, k$, $c_i \leftarrow S_t(r_i)$.
5. For $i = k+1, \ldots n$, $c_i \leftarrow U_t()$.
6. $b \leftarrow D(S_t, c, k, r)$.

Call this $Experiment_1$.

Next, we modify the first experiment so that the $k+1$-st element is chosen from the translucent set. Let $Experiment_2$ be

1. $(S_t, V_t) \leftarrow KG(t)$.
2. $k \leftarrow \{0, 2, \ldots, n-1\}$.

3. $r \leftarrow R^{\otimes k}()$.
4. For $i = 1, \ldots, k$, $c_i \leftarrow S_t(r_i)$.
5. $c_{k+1} \leftarrow S_t()$.
6. For $i = k+2, \ldots, n$, $c_i \leftarrow U_t()$.
7. $b \leftarrow D(S_t, c, k, r)$.

We want to bound

$$|Pr(Experiment_1 : b = 1) - Pr(Experiment_2 : b = 1)|$$

We consider the third property of translucent sets and the distinguisher D' given by: "on input (S_t, x),

1. $k \leftarrow \{0, 2, \ldots, n-1\}$.
2. $r \leftarrow R^{\otimes k}()$.
3. For $i = 1, \ldots, k$, $c_i \leftarrow S_t(r_i)$.
4. $c_{k+1} \leftarrow x$.
5. For $i = k+2, \ldots, n$, $c_i \leftarrow U_t()$.
6. $b \leftarrow D(S_t, c, k, r)$.
7. Output b.

Then,

$$\begin{aligned}
&|Pr(Experiment_1 : b = 1) - Pr(Experiment_2 : b = 1)| \\
&\qquad = |Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow U_t() : D(S_t, x) = 1) \\
&\qquad\qquad - Pr((S_t, V_t) \leftarrow KG(t), x \leftarrow S_t() : D(S_t, x) = 1)| < \epsilon(t)
\end{aligned}$$

Now, let $Experiment_3$ describe a faked reveal of an encryption of 1 as an encryption of zero. Specifically, $Experiment_3$ proceeds as follows:

1. $(S_t, V_t) \leftarrow KG(t)$.
2. $k' \leftarrow \{1, 3, \ldots, n\}$.
3. $r' \leftarrow R^{\otimes k'}()$.
4. For $i = 1, \ldots, k'$, $c_i \leftarrow S_t(r'_i)$.
5. For $i = k'+1, \ldots, n$, $c_i \leftarrow U_t()$.
6. $k \leftarrow k' - 1$.
7. $r \leftarrow (r'_1, \ldots r'_k)$.

8. $b \leftarrow D(S_t, c, k, r)$.

The joint distribution of the inputs to D is the same for $Experiment_2$ and $Experiment_3$. Therefore,

$$Pr(Experiment_2 : b = 1) = Pr(Experiment_3 : b = 1)$$

We conclude that Eve can distinguish an honest reveal of an encryption of 0 and a faked reveal of an encryption of 1 as an encryption of 0 with advantage at most $\epsilon(t)$.

Next, we consider an honest reveal of an encryption of 1 and a faked reveal of an encryption of 0 as an encryption of 1. Let $Experiment_1$ correspond to an honest reveal of an encryption of 1; specifically, $Experiment_1$ proceeds as follows:

1. $(S_t, V_t) \leftarrow KG(t)$.
2. $k \leftarrow \{1, 3, \ldots, n\}$.
3. $r \leftarrow R^{\otimes k}()$.
4. For $i = 1, \ldots, k$, $c_i \leftarrow S_t(r_i)$.
5. For $i = k+1, \ldots, n$, $c_i \leftarrow U_t()$.
6. $b \leftarrow D(S_t, c, k, r)$.

Next, we modify how c_{k+1} is computed. Let $Experiment_2$ be

1. $(S_t, V_t) \leftarrow KG(t)$.
2. $k \leftarrow \{1, 3, \ldots, n\}$.
3. $r \leftarrow R^{\otimes k}()$.
4. For $i = 1, \ldots, k$, $c_i \leftarrow S_t(r_i)$.
5. If $k < n$, $c_{k+1} \leftarrow S_t()$.
6. For $i = k+2, \ldots, n$, $c_i \leftarrow U_t()$.
7. $b \leftarrow D(S_t, c, k, r)$.

Next, we want to bound $|Pr(Experiment_1 : b = 1) - Pr(Experiment_2 : b = 1)|$. Let D' be given by: "on input (S_t, x), do the following:

1. $k \leftarrow \{1, 3, \ldots, n\}$.
2. $r \leftarrow R^{\otimes k}()$.
3. For $i = 1, \ldots, k$, $c_i \leftarrow S_t(r_i)$.
4. If $k < n$, $c_{k+1} \leftarrow x$.

5. For $i = k+2, \ldots, n$, $c_i \leftarrow U_t()$.
6. $b \leftarrow D(S_t, c, k, r)$.
7. Output b.

Then, the experiment $(S_t, V_t) \leftarrow KG(t), x \leftarrow U_t(), b \leftarrow D'(S_t, x)$ is equivalent to $Experiment_1$, while the experiment $(S_t, V_t) \leftarrow KG(t), x \leftarrow S_t(), b \leftarrow D'(S_t, x)$ is equivalent to $Experiment_2$. We conclude that

$$|Pr(Experiment_1 : b = 1) - Pr(Experiment_2 : b = 1)| < \epsilon(t)$$

Next, we look at a faked reveal of an encryption of 0 as an encryption of 1. Let $Experiment_3$ be:

1. $(S_t, V_t) \leftarrow KG(t)$.
2. $k' \leftarrow \{0, 2, \ldots, n-1\}$.
3. $r' \leftarrow R^{\otimes k'}()$.
4. For $i = 1, \ldots, k'$, $c_i \leftarrow S_t(r_i)$.
5. For $i = k'+1, \ldots, n$, $c_i \leftarrow U_t()$.
6. $k \leftarrow k' - 1$.
7. $r \leftarrow (r'_1, \ldots r'_k)$.
8. $b \leftarrow D(S_t, c, k, r)$.

Now, we want to bound $|Pr(Experiment_2 : b = 1) - Pr(Experiment_3 : b = 1)|$. We look at the inputs to D in the two experiments; let p, q denote the joint distributions of the inputs in $Experiment_2$ and $Experiment_3$ respectively. We look at the statistical distance between p and q:

$$\begin{aligned}\frac{1}{2}\|p - q\|_1 = \frac{1}{2} \sum_{S_t, c, k, r} |p(S_t, c, k, r) - q(S_t, c, k, r)| \\ = \frac{1}{2} \sum_{S_t, c, k, r} |p(k)p(S_t, c, r|k) - q(k)q(S_t, c, r|k)|\end{aligned}$$

Now, when $k \in \{1, 3, \ldots, n-2\}$, we have $p(k) = q(k) = 2/(n+1)$, and we have $p(S_t, c, r|k) = q(S_t, c, r|k)$ because in both cases S_t is independent of k, r has k elements, and c has $k+1$ random translucent set elements and $n-k-1$ random t-bit strings. When $k = n$, we have $p(k) = 2/(n+1)$, $q(k) = 0$. When $k = -1$, we have $p(k) = 0$, $q(k) = 2/(n+1)$. We conclude that

$$\frac{1}{2}\|p - q\|_1 = \frac{2}{n+1}$$

and therefore

$$|Pr(Experiment_2 : b = 1) - Pr(Experiment_3 : b = 1)| \leq \frac{2}{n+1}$$

From this and the previous discussion, we conclude that Eve can distinguish an honest reveal of an encryption of 1 from a faked reveal of an encryption of 0 as an encryption of 1 with an advantage at most $2/(n+1) + \epsilon(t)$. This completes the proof. $\square$

4.5 Lower bounds on the level of deniability

We see from the proof of Theorem 4 that for the parity scheme, there exists a distinguisher that can tell apart an honest reveal of an encryption of 1 from a faked reveal of an encryption of 0 as an encryption of 1 with advantage $2/(n+1)$. One such distinguisher is the following: "On input S_t, c, k, r, if $(k = n) \wedge (\wedge_{i=1}^{n} S_t(r_i) = c_i)$ output 1 else output 0".

In this section, we generalize this observation to a class of schemes that [4] calls separable. In performing the generalization, we want to keep the following feature of the parity scheme: the faking algorithm claims that the number of translucent set elements is lower than it actually is.

First, we generalize our idea of deniable encryption protocol. We want to model the publicly observable communication, the intended message, the private randomness of Alice and Bob, and the method that Alice uses to lie about the message and the private randomness. Let $\tau(b, r, s)$ denote the publicly observable transcript during execution of the protocol with desired message bit b, private randomness for Alice r, and private randomness for Bob s. Let $\phi(b, r, \tau)$ be the method by which Alice takes the actual message bit b, her actual private randomness r, and the publicly observable transcript τ and produces fake randomness to present to Eve or outputs a failure message indicating that no faking is possible in the given case. We arrive at the following

Definition 6. *A deniable encryption protocol is a tuple* $(\mathbb{R}, \mathbb{S}, \mathbb{T}, \tau, \phi)$*, where* $\mathbb{R}, \mathbb{S}$ *are the sets of possible private inputs for Alice and Bob,* $\mathbb{T}$ *is the set of possible public transcripts for the protocol,*

$$\tau : \{0, 1\} \times \mathbb{R} \times \mathbb{S} \to \mathbb{T}$$

is the function that takes a message bit, and private inputs for Alice and Bob to the resulting public transcript, and

$$\phi : \{0, 1\} \times \mathbb{R} \times \mathbb{T} \to \mathbb{R} \cup \{\perp\}$$

is the (possibly randomized) algorithm that takes the actual message bit, private input for Alice and public transcript and returns a fake private input for Alice or an indication of failure.

If it is desired to model also the role of the security parameter then one could consider a sequence of such tuples; however, this will not be necessary in this section.

We illustrate this definition using the parity scheme as example. In the parity scheme, the publicly observable communication consists of Bob's public key S_t and of the ciphertext c that Alice sends to Bob. The private randomness for Bob is the randomness he uses to generate his public key-secret key pair. The private randomness for Alice consists of the choice of the number k of translucent set elements, the values $d_1, \ldots, d_k$ she uses to generate the translucent set elements, and the values $c_{k+1}, \ldots, c_n$ she uses to generate the random elements of $\{0,1\}^t$. Note however that this natural representation for the private input of Alice has the following property: the sets of private inputs used to encrypt 0 and to encrypt 1 are disjoint.

In order to fit this to the generalized definition above and avoid certain technical issues (see also the remarks below), we need a uniform representation for the private inputs used to encrypt 0 and the private inputs used to encrypt 1. Therefore, think of elements of $\mathbb{R}$ as being tuples $(k, d_1, \ldots d_n, c_1, \ldots, c_n)$ where k is a number in $\{0, 2, \ldots, n-1\}$, $d_1, \ldots d_n$ are seeds for the generation of translucent set elements, and $c_1, \ldots c_n$ are elements of $\{0,1\}^t$. In this representation, Alice's encryption algorithm takes the first $k+b$ of the d's and the last $n-k-b$ of the c's in order to encrypt message bit b. In this representation, Alice's faking algorithm works as follows:

$$\begin{aligned}
&\phi(b, (k, d_1, \ldots, d_n, c_1, \ldots c_n), \tau) \\
&= \begin{cases} (k, d_1, \ldots, d_k, d'_{k+1}, d_{k+2}, \ldots d_n, c_1, \ldots, c_k, S_t(d_{k+1}), c_{k+2}, \ldots, c_n) & \text{if } b = 1 \\ (k-2, d_1, \ldots, d_{k-1}, d'_k, d_{k+1}, \ldots, d_n, c_1, \ldots c_{k-1}, S_t(d_k), c_{k+1}, \ldots, c_n) & \text{if } b = 0 \wedge k \geq 2 \\ \perp & \text{if } b = 0 \wedge k = 0 \end{cases}
\end{aligned}$$

where in the first line d'_{k+1} is chosen so that $S_t(d'_{k+1}) \neq S_t(d_{k+1})$ and in the second line d'_k is chosen so that $S_t(d'_k) \neq S_t(d_k)$.

Now, we want to introduce something analogous to the fact that in the parity scheme, Alice fools Eve by claiming a lower number of translucent set elements. The first step in this direction is the classification function.

Definition 7. *A classification function for a deniable encryption protocol*

$$(\mathbb{R}, \mathbb{S}, \mathbb{T}, \tau, \phi)$$

is a function

$$\gamma : \{0,1\} \times \mathbb{R} \times \mathbb{T} \to \{0, 1, \ldots, n\}$$

for some $n \in \mathbb{N}$.

In the case of the parity scheme, the classification function returns $k+b$, where b is the message bit and k is taken from the element of $\mathbb{R}$.

Based on the intuition from the parity scheme, we define separable schemes to be the ones where applying the faking method leads to a lower value of the classification function.

Definition 8. *A deniable encryption protocol* $(\mathbb{R}, \mathbb{S}, \mathbb{T}, \tau, \phi)$ *is called* n*-separable if there exists a classification function* $\gamma : \{0,1\} \times \mathbb{R} \times \mathbb{T} \to \{0,1,\ldots,n\}$ *with the property: there exists* $b \in \{0,1\}$ *such that*

$$\mathbb{E}\left[\gamma\Big(b, \phi\big(1-b, R, \tau(1-b,R,S)\big), \tau(1-b,R,S)\Big)\right] \leq \mathbb{E}[\gamma(b,R,\tau(b,R,S))] - 1$$

where R, S *are a random variable taking values in* $\mathbb{R}, \mathbb{S}$ *with the appropriate distributions, and where we take* $\gamma(b, \perp, \tau) = -1$ *in order to avoid having undefined expressions when* ϕ *returns an error.*

We illustrate this definition using the parity scheme. For the parity scheme, for all $r \in \mathbb{R}, s \in \mathbb{S}$, we have

$$\begin{aligned}
&\gamma(0, \phi(1,r,\tau(1,r,s)), \tau(1,r,s)) = k(r) = \gamma(0,r,\tau(0,r,s)) \\
&\gamma(1, \phi(0,r,\tau(0,r,s)), \tau(0,r,s)) = k(r) - 2 + 1 = \gamma(1,r,\tau(1,r,s)) - 1
\end{aligned}$$

and therefore the condition in the definition of separable schemes holds for $b = 1$.

Now, we are ready to show that the existence of a good distinguisher between honest and fake openings is not limited to the parity scheme, but extends to any separable scheme.

Theorem 5. *Let* $(\mathbb{R}, \mathbb{S}, \mathbb{T}, \tau, \phi)$ *be an* n*-separable deniable encryption protocol. Then, there exists* $b \in \{0,1\}$*, and there exists a distinguisher that can tell apart an honest opening of an encryption of* b *and a faked opening of an encryption of* $1-b$ *as an encryption of* b *with advantage at least* $1/(n+1)$*.*

Proof. We introduce the following shorthand notation. Let $X_b = (b, R, \tau(b,R,S))$ be the random variable describing the adversary view for an honest opening of an encryption of b, and let $Y_b = (b, \phi(1-b,R,\tau(1-b,R,S)), \tau(1-b,R,S))$ be the random variable describing the view of the adversary for a faked opening of an encryption of $1-b$ as an encryption of b.

Let b be the message bit such that $\mathbb{E}(\gamma(Y_b)) \leq \mathbb{E}(\gamma(X_b)) - 1$ (Definition 8). Define the set $Z \subset \{0,1,\ldots,n\}$ by

$$Z = \{z \in \{0,1,\ldots,n\} : \mathbb{P}(\gamma(X_b) = z) > \mathbb{P}(\gamma(Y_b) = z)\}$$

Then,

$$\begin{aligned}
1 \leq \mathbb{E}(\gamma(X_b) - \gamma(Y_b)) &= \sum_{z=-1}^{n} z(\mathbb{P}(\gamma(X_b) = z) - \mathbb{P}(\gamma(Y_b) = z)) \\
&\leq \mathbb{P}(\gamma(Y_b) = -1) + \sum_{z \in Z} z(\mathbb{P}(\gamma(X_b) = z) - \mathbb{P}(\gamma(Y_b) = z)) \\
&\leq \mathbb{P}(\gamma(Y_b) = -1) + nSD(\gamma(X_b), \gamma(Y_b)) \\
&\leq (n+1)SD(\gamma(X_b), \gamma(Y_b))
\end{aligned}$$

where $SD(\cdot,\cdot)$ denotes the statistical distance of two random variables. $\square$

Remark: [4, Definition 4] defines a scheme to be separable if for all r_A either one or the other of the two inequalities

$$\mathbb{E}_{R_B}\Bigg(\mathcal{C}\Big(\phi\big(0, r_A, \mathcal{COM}(0, r_A, R_B)\big)\Big)\Bigg) \leq \mathcal{C}(r_A) - 1$$

$$\mathbb{E}_{R_B}\Bigg(\mathcal{C}\Big(\phi\big(1, r_A, \mathcal{COM}(1, r_A, R_B)\big)\Big)\Bigg) \leq \mathcal{C}(r_A) - 1$$

holds.

However, it is not clear how with that version of the definition, the conclusion "either

$$\mathbb{E}(\mathcal{C}(R_A) - \mathcal{C}(\phi_0(R_A, R_B))) \geq 1/2$$

or

$$\mathbb{E}(\mathcal{C}(R_A) - \mathcal{C}(\phi_1(R_A, R_B))) \geq 1/2\text{”}$$

can be reached in the second paragraph of the proof on page 12.

To illustrate the problem with the argument on page 12 of [4], we present the following example: we will show that there exists random variable X and functions f, g, h such that

$$\forall x \in range(X), \text{ either } f(x) \leq h(x) - 1 \text{ or } g(x) \leq h(x) - 1$$

and in addition $\mathbb{E}(f(X)) > \mathbb{E}(h(X))$ and $\mathbb{E}(g(X)) > \mathbb{E}(h(X))$. Indeed, let X be a random 3 bit string and let f, g, h be given as in the table:

x	000	001	010	100	011	101	110	111
f	-1	3	0	3	1	3	1	3
g	3	0	3	0	3	1	3	2
h	0	1	1	1	2	2	2	3

Then, $\mathbb{E}(f(X)) = 13/8$, $\mathbb{E}(g(X)) = 15/8$, $\mathbb{E}(h(X)) = 12/8$.

Remark: Consider the parity scheme. The most natural representation of the random inputs for Alice is $r_A = (k, r_1, \ldots r_k, c_{k+1}, \ldots, c_n)$. With this representation, the random inputs used for an encryption of zero and the random inputs used for an encryption of one are members of disjoint sets. This leads to the following problem: the expressions $\mathcal{COM}(b, r_A, r_B)$ and $\phi(b, r_A, \mathcal{COM}(b, r_A, r_B))$ used in the proof of the claim on page 12 of [4] are not well-defined when r_A is a private input for the encryption of 1-b.

We have seen above that it is possible to encode the random input of Alice in such a way that the same set of random inputs is used for an encryption of zero and of one, but with this encoding, another problem arises: if the classification function is given only Alice's private input, it will not be able to distinguish whether it has $2l$ or $2l+1$ translucent set elements. Therefore, we have deviated from the exposition in Section 4 of [4] and have allowed the classification function to take also the message bit and the transcript as inputs.

5 Deniable Authentication

In this section, we examine the security and deniability of example (1) from the introduction. We reproduce the example here adding one additional layer of detail.

$$\begin{array}{ccc} \text{Alice} & & \text{Bob} \\ & \xrightarrow{m,s} & \\ & \xleftarrow{m,s,enc(pk_A,k)} & \\ & \xrightarrow{m,s,tag(k,(m,s))} & \end{array} \tag{5}$$

As before, $enc(pk_A, \cdot)$ is encryption under Alice's public key, and k is a randomly chosen key for a symmetric key message authentication scheme drawn by Bob as a challenge for Alice. The new detail is the string s: it allows Alice and Bob to associate incoming protocol flows to the appropriate session. We assume that Alice generates the session identifier s independently, uniformly at random from $\{0,1\}^n$ for each message.

In the remainder of this section, we first define a notion of security for interactive message authentication (subsection 5.1) that is analogous to the EUF-CMA security notion. Then, we show that our running example (5) satisfies this security notion (subsection 5.2). We proceed to define deniability for interactive authentication (subsection 5.3). Our first result on deniability is negative: we show that IND-CCA2 secure encryption and EUF-CMA secure message authentication are not sufficient for example (5) to be deniable (subsection 5.4). We then show that a stronger requirement on the encryption scheme, plaintext awareness, does suffice for deniability (subsection 5.5). One may wonder whether plaintext aware encryption schemes exist at all; we present one construction that has this property under a non-standard assumption called the Diffie Hellman Knowledge of Exponent assumption (subsection 5.6). The exposition in this section is influenced by the ideas of [6, 2].

5.1 Asymmetric key interactive message authentication protocols

An asymmetric key interactive message authentication protocol consists of three algorithms: $(kg, send, rec)$. kg is a probabilistic polynomial time key generation algorithm, that takes as input the security parameter in unary and outputs a pair (pk, sk). $send$ is an interactive probabilistic polynomial time algorithm that takes as input sk and reacts to external requests from the user or from the network to initiate and continue the interactive protocol execution as the message sender. rec is an interactive probabilistic polynomial time algorithm that takes as input pk and reacts to external requests to perform the interactive protocol as the message receiver.

We require the protocol $(kg, send, rec)$ to satisfy perfect correctness, i.e. when $(pk, sk) \leftarrow kg(1^n)$ and messages are faithfully forwarded between $send(sk)$ and $rec(pk)$, the receiver eventually accepts the message as coming from the sender.

We would like to have a notion of security for interactive authentication protocols. We take the main idea of the chosen message attack against non-interactive message authentication (subsection 2.2), and adapt it to the interactive case. Before we proceed, we remark on the difference between the interactive and non-interactive case. The main conceptual difference between the two cases concerns open sessions and interleaving (ordering in time) of messages.

In the non-interactive case, an adversary query to the *tag* or *sign* oracle opens a new session for the sender, but with the output of the response of *tag* or *sign*, this session is immediately closed. Similarly, a call to the vrf oracle opens a session for the receiver, and this session immediately closes with the output of the corresponding response. This means that the sender and receiver never have more than one session open at the same time, and implies that the order of sessions in time satisfies simple properties, such as "if one session starts before another, then it also ends before the other."

In the interactive case, the adversary can open a session of a sender or a receiver and keep it open for a long time; messages of other sessions may occur between the opening and closing. This means that the sender and receiver can have many open sessions at the same time, and need to be able to match external requests to the appropriate open session. It also means that simple properties of the time order such as our example "if one session starts before another, then it also ends before the other" are no longer true.

Now, we proceed to describe the chosen message attack security experiment for interactive authentication protocols. Given a protocol $IMA = (kg, send, rec)$ and adversary strategy A, the experiment proceeds as follows:

1. $(pk, sk) \leftarrow kg(1^n)$.
2. A is given pk. A can adaptively submit activation requests of its choice to $send(sk)$, $rec(pk)$ and see their response.

The interaction of A and IMA produces a transcript; we denote this by $T \leftarrow [A, IMA]$. T is a random variable.

We define $B'(T)$ to be the event that T contains a session of rec that accepts a message m without a corresponding session of $send$ that previously sent message m. $B'(T)$ captures our intuition about what it means for A to break IMA.

Definition 9. *Given an interactive message authentication protocol IMA and a probabilistic polynomial time adversary strategy A, we define the advantage of A against IMA in the chosen message attack experiment by*

$$AdvCMA'(A, IMA) = Pr(T \leftarrow [A, IMA](1^n) : B'(T))$$

We say that the interactive message authentication protocol IMA is secure against a chosen message attack if for all probabilistic polynomial time adversary strategies A, $AdvCMA'(A, IMA)$ is a negligible function of the security parameter n.

Finally, we remark that it is often convenient to think of a single computational entity, called a challenger, that gives pk to A and answers all of A's queries to $send(sk)$ sessions and $rec(pk)$ sessions.

5.2 Security proof for an interactive message authentication protocol

In this section, we show that our running example (5) satisfies the security notion of subsection 5.1. The exposition here was influenced by the ideas in [2].

Let IMA denote our running example protocol, and let ENC, MAC denote the underlying public key encryption and symmetric key message authentication schemes. Now, we show

Theorem 6. *Let A be a probabilistic polynomial time adversary strategy against IMA. Let N_r be an upper bound on the number of receiver sessions that A activates. Then, there exists probabilistic polynomial time A', A'' such that*

$$AdvCMA'(A, IMA) \leq AdvCCA2_{N_r}(A', ENC) + AdvCMA_{N_r}(A'', MAC)$$

Corollary 1. *If ENC is IND-CCA2 secure and MAC is EUF-CMA secure, then IMA is secure against a chosen message attack.*

Proof. Intuitively, the adversary can break the authentication protocol if it manages to break one of the receiver sessions. This in turn can happen in two cases: if the adversary can learn the ephemeral symmetric key used in that session, or if the adversary can create a correct message authentication tag without knowing the corresponding ephemeral key. The first case is prevented by the security of the encryption scheme, while the second case is prevented by the security of the symmetric-key message authentication scheme. We follow this intuition in constructing the proof.

We now proceed with the details. The security experiment against IMA can be described by the following pseudo-code:

1. $(pk, sk) \leftarrow kg(1^n)$
2. $T \leftarrow EmptyList$
3. While $A(pk)$ has not terminated
   (a) If A makes a query to $send(sk)$ with values $(Message_0, m)$,
      i. $s \leftarrow \{0,1\}^n$
      ii. give (m, s) to A, add $(Message_0, m, (m, s))$ to T
   (b) If A makes a query to $rec(pk)$ with values $(Message_1, m, s)$
      i. Draw $k \leftarrow kgmac(1^n)$. Associate k to (m, s).
      ii. $c \leftarrow enc(pk, k)$. Associate k to c.
      iii. Output (m, s, c) to A, add $(Message_1, (m, s), (m, s, c))$ to T.
   (c) If A makes a query to $send(sk)$ with values $(Message_2, m, s, c)$
      i. Check if there is an open session with associated message and session id (m, s). If not, output $\perp$ to A, add $(Message_2, (m, s, c), \perp)$ to T. Else, continue.

ii. If c was previously produced by $rec(sk)$, retrieve the associated k, draw $t \leftarrow tag(k, (m, s))$.

iii. Else $k \leftarrow dec(sk, c)$, $t \leftarrow tag(k, (m, s))$

iv. Give (m, s, t) to A, add $(Message_2, (m, s, c), (m, s, t))$ to T.

(d) If A makes a query to $rec(pk)$ with values $(Message_3, m, s, t)$

i. Check if there is an open session with associated message and session id (m, s). If not, output $\perp$ to A, add $(Message_3, (m, s, t), \perp)$ to T. Else, continue.

ii. Retrieve k associated to (m, s). If $vrf(k, (m, s), t) = 0$ output $\perp$ to A, and add $(Message_3, (m, s, t), \perp)$ to T, else if $vrf(k, (m, s), t) = 1$ output $SessionDone$ to A, and add

$$(Message_3, (m, s, t), SessionDone)$$

to T.

Now, we modify this experiment in small steps. Let $[A, CH_1]$ denote the interaction in the original experiment, as described above.

First, we limit the ability to learn the ephemeral symmetric keys used for authentication. Let the interaction $[A, CH_2]$ proceed as $[A, CH_1]$ except that lines 3.(b).i, 3.(b).ii change to

3.(b).i' Draw $k_0 \leftarrow kg(1^n)$. Draw $k_1 \leftarrow kg(1^n)$. Associate k_0 to (m, s).

3.(b).ii' Draw $c \leftarrow enc(pk, k_1)$. However, associate k_0 to c, so that k_0 is retrieved in line 3.(c).ii.

Now, we need to evaluate the change of adversary advantage from $[A, CH_1]$ to $[A, CH_2]$. For that purpose, we think of the of the instructions in lines 2, 3 (with all sub-items) as a single algorithm A' which takes input pk and participates in the N_r-challenge IND-CCA2 experiment (subsection 3.1). We interpret line 3.(c).iii as A' making a query to the decryption oracle, and we interpret lines 3.(b).i,3.(b).ii, respectively 3.(b).i',3.(b).ii', as A' requesting a new challenge cyphertext on message pair (k_0, k_1). Finally, we define the output of A' to be 0 if $B'(T)$ occurs and 1 otherwise. Thus, we have:

$$Pr(T \leftarrow [A, CH_1] : B'(T)) = 1 - Pr([A', MC - CCA2(0^{N_r})] = 1)$$
$$Pr(T \leftarrow [A, CH_2] : B'(T)) = 1 - Pr([A', MC - CCA2(1^{N_r})] = 1)$$

where we have used $MC-CCA2$ to denote the challenger in the multi-challenge IND-CCA2 security experiment. Thus, we have

$$Pr(T \leftarrow [A, CH_1] : B'(T)) - Pr(T \leftarrow [A, CH_2] : B'(T)) \\ = AdvCCA2_{N_r}(A', ENC) \quad (6)$$

Next, we want to evaluate the adversary advantage in the interaction $[A, CH_2]$. For that purpose, interpret the instructions in lines 1,2,3 (with all sub-items except the drawing of the keys k_0 in line 3.(b).i', and the retrieval of the keys in lines 3.(c).ii and 3.(d).ii) as a single algorithm A'' that participates in the N_r-key EUF-CMA security experiment (subsection 3.3). We interpret the keys k_0 drawn in line 3.(b).i as the secret keys of the EUF-CMA experiment. We interpret line 3.(c).ii as A'' making a query to the *tag* oracle. We interpret line 3.(d).ii as A'' making a query to the verify oracle.

Now we want to relate the advantage of A and A''. Before we do that, first we summarize the situation. We have an experiment that we can think of either as the interaction $[A, CH_2]$ or as the interaction $[A'', MK - MAC]$ where $MK - MAC$ denotes the challenger in the N_r-key EUF-CMA security experiment. We have two transcripts: the transcript T produced at the interface between A and CH_2, and the transcript T'' produced at the interface between A'' and $MK - MAC$.

We claim that the event $B'(T)$ implies the event $B(T'')$. To prove this, suppose that $B(T'')$ does not occur. If no tag verification succeeds, then $B'(T)$ also does not occur. If there are k, m, s, t such that $vrf(k, (m, s), t) = 1$ in line 3.(d).ii, then, for any such tuple, the tag t was produced by a $tag(k, (m, s))$ oracle query in line 3.(c).ii. Again, we see that $B'(T)$ does not occur. Thus, we have

$$\begin{aligned} Pr(T \leftarrow [A, CH_2] : B'(T)) \leq Pr(T'' \leftarrow [A'', MK - MAC] : B(T'')) \\ = AdvCMA_{N_r}(A'', MAC) \quad (7) \end{aligned}$$

Combining (6), (7) proves the theorem. □

5.3 Deniability for interactive authentication protocols

Let $IMA = (kg, send, rec)$ be an interactive message authentication protocol. Intuitively, the sender is able to deny participating in the protocol if the receiver could generate by himself a transcript that looks indistinguishable from a real interaction with the sender.

A further refinement of this idea is that such a property should hold not only against a receiver that follows the protocol, but also against a receiver that arbitrarily deviates from it. We can see immediately that our running example (5) is deniable against an honest receiver. However, we will see later on that the situation against arbitrary receiver is more complex. We postpone the details, and for now concentrate on defining deniability against an arbitrary receiver.

We model this situation with the following experiment: given a probabilistic polynomial time algorithm A,

1. $(pk, sk) \leftarrow kg(1^n)$.
2. $r \leftarrow Rand(1^n)$, where $Rand$ is a procedure that generates the random coins for A.
3. A takes inputs pk, r and has oracle access to $send(sk)$. A can start many sessions of *send*, interleaving them arbitrarily. The interaction of A and *send* produces a transcript; we denote this by $T \leftarrow A^{send(sk)}(r, pk)$

The view of A in the above experiment consists of the inputs (r, pk) that A receives and the transcript T. Now, we require that a computationally indistinguishable view can be produced without access to the $send(sk)$ oracle:

Definition 10. *The interactive message authentication protocol* $(kg, send, rec)$ *is deniable if for all efficient algorithms A, there exists efficient S such that for all efficient D,*

$$|Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow Rand(1^n), T \leftarrow A^{send(sk)}(r, pk) : D(r, pk, T) = 1) \\ - Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow Rand(1^n), T \leftarrow S(r, pk) : D(r, pk, T) = 1)|$$

is negligible.

5.4 IND-CCA2 secure encryption and EUF-CMA secure message authentication are not enough for our example protocol to be deniable

Now we start investigating whether our running example protocol (5) is deniable. Our first result is negative: we show that it is not sufficient to require that the encryption scheme used is IND-CCA2 secure and that the message authentication scheme used is EUF-CMA secure to ensure that the protocol is deniable. Intuitively, the problem is that a receiver that deviates from the protocol can submit an ill-formed ciphertext in the second flow of the protocol. We proceed with the details.

Suppose that $ENC = (kg, enc, dec)$ is an IND-CCA2 secure encryption scheme. Let f be a length preserving one-way function. We construct a new encryption scheme $ENC' = (kg', enc', dec')$. kg' proceeds as follows:

1. On input 1^n,
2. $(pk, sk) \leftarrow kg(1^n)$
3. $u_1 \leftarrow \{0,1\}^n, u_2 \leftarrow \{0,1\}^n$
4. $U_1 \leftarrow f(u_1), U_2 \leftarrow f(u_2)$
5. $pk' \leftarrow (pk, U_1, U_2), sk' \leftarrow (sk, u_1, u_2)$
6. Output (pk', sk')

enc' proceeds as follows: on input (pk, U_1, U_2, m), $c \leftarrow enc(pk, m)$, output $(0, c)$. dec' proceeds as follows:

1. On input $((sk, u_1, u_2), (b, c))$,
2. If $b = 0$, $d \leftarrow dec(sk, c)$,
3. Else if $b = 1$ and $c = (U_1, U_2)$, $d \leftarrow (u_1, u_2)$,
4. Else $d \leftarrow \perp$

5. Output d.

Thus, we have taken ENC and have added an independent mechanism involving a one-way function to it. Intuitively, ENC' should be as secure as ENC, and indeed we have

Claim 1. *Let A' be an efficient adversary strategy in the CCA2 experiment against ENC'. Then, there exists an efficient adversary strategy A in the CCA2 experiment against ENC such that $AdvCCA2(A, ENC) = AdvCCA2(A', ENC')$*

Proof. We construct an interactive algorithm A'' such that in the interaction

$$[[CH - CCA2(ENC), A''], A'] = [CH - CCA2(ENC), [A'', A']]$$

the combination $[CH-CCA2(ENC), A'']$ is equivalent to $CH-CCA2(ENC')$, and such that the combination $A = [A'', A']$ is an adversary strategy against ENC. Specifically, A'' operates as follows:

1. It receives pk from $CH - CCA2(ENC)$. It draws $u_1, u_2 \leftarrow \{0,1\}^n, U_1 \leftarrow f(u_1), U_2 \leftarrow f(u_2)$ and gives (pk, U_1, U_2) to A'

2. A'' answers decryption queries (b, c) from A' as follows

   (a) If $b = 0$, query $CH - CCA2(ENC)$ on c, and forward the response to A'.

   (b) If $b = 1$ and $c = (U_1, U_2)$, reply with (u_1, u_2).

   (c) Else reply with $\perp$.

3. When A' produces m_0, m_1, forward these to $CH - CCA2(ENC)$, obtain the challenge ciphertext c, and give $(0, c)$ to A'.

4. When A' outputs b', output b'.

Thus, it is clear that $AdvCCA2(A, ENC) = AdvCCA2(A', ENC')$. □

Similarly, suppose that $MAC = (kgmac, tag, vrf)$ is a EUF-CMA secure message authentication scheme. We construct a new scheme $MAC' = (kgmac', tag', vrf')$, which operates as follows

1. $kgmac'(1^n)$ draws independent $k_1, k_2 \leftarrow kgmac(1^n)$ and outputs (k_1, k_2).

2. $tag'(k_1, k_2, m)$ draws $t \leftarrow tag(k_1, m)$ and outputs (t, k_2).

3. $vrf'(k_1, k_2, m, t_1, t_2)$, takes $b_1 \leftarrow vrf(k_1, m, t_1)$, $b_2 \leftarrow (k_2 = t_2)$ and outputs $b_1 \wedge b_2$.

Thus, we have introduced an independent second key, and, intuitively, MAC' should be as secure as MAC. Indeed, we have:

Claim 2. *Let A' be an efficient adversary strategy against MAC'. Then, there exists an efficient adversary strategy A against MAC such that*

$$AdvCMA(A, MAC) = AdvCMA(A', MAC')$$

Proof. Similarly to the previous proof, we construct an interactive algorithm A'' such that in the interaction

$$[[MAC, A''], A'] = [MAC, [A'', A']]$$

the combination $[MAC, A'']$ is equivalent to MAC' from the point of view of A' and the combination $A = [A'', A']$ is an efficient strategy against MAC. Specifically, A'' operates as follows:

1. It draws $k_2 \leftarrow kgmac(1^n)$.
2. On (tag, m) query from A', A'' obtains $t_1 \leftarrow tag(k_1, m)$ from MAC and gives t_1, k_2 to A'.
3. On (vrf, m, t_1, t_2) query from A', A'' obtains $b_1 \leftarrow vrf(k_1, m, t_1)$ from MAC, computes $b_2 \leftarrow (k_2 = t_2)$ and returns $b_1 \wedge b_2$ to A'.

Thus, the interaction $[MAC, A'', A']$ produces two transcripts: a transcript T of the queries at the $MAC - A''$ interface and a transcript T' of the queries at the $A'' - A'$ interface. The queries in T and in T' are in a one-to-one correspondence, and the event $B(T)$ occurs if and only if the event $B(T')$ occurs. Thus, $AdvCMA(A, MAC) = AdvCMA(A', MAC')$. □

Now, we let IMA be our interactive message authentication protocol (5) instantiated with ENC and MAC, and let IMA' be the same protocol instantiated with ENC', MAC'. If ENC is IND-CCA2 secure and MAC is EUF-CMA secure, then IMA' is secure against a chosen message attack. However, we have the following:

Proposition 1. *IMA' is not deniable.*

Proof. We present a specific A, for which no simulator can produce a computationally indistinguishable view. A operates as follows:

1. It receives (pk, U_1, U_2).
2. It pick any m, and initializes a single *send* session, asking it to send message m. The sender session outputs (m, s), where s is a randomly chosen session ID.
3. A submits $(1, U_1, U_2)$ as challenge ciphertext. The sender session responds with $(tag(u_1, m), u_2)$, where u_2 is the pre-image of U_2 under the one-way function f. Then, A halts.

Intuitively, it should not be possible to simulate this transcript without access to the *send* oracle. Let D be the distinguisher which expects to see input of the form

$$(r, pk', T) = (r, pk, U_1, U_2, m, s, c, t, u_2)$$

and checks whether $f(u_2) = U_2$ and if so outputs 1, else outputs 0. Then,

$$\begin{aligned}&Pr((pk', sk') \leftarrow kg'(1^n), r \leftarrow Rand(1^n),\\ &\qquad\qquad\qquad T \leftarrow A^{send(sk')}(r, pk') : D(r, pk', T) = 1) = 1\end{aligned}$$

while for any efficient simulator S,

$$Pr((pk', sk') \leftarrow kg'(1^n), r \leftarrow Rand(1^n), T \leftarrow S(r, pk') : D(r, pk', T) = 1)$$

is negligible, because it is the probability that the algorithm S' given by

1. On input U_2
2. $u_1 \leftarrow \{0, 1\}^n$
3. $U_1 \leftarrow f(u_1)$
4. $(pk, sk) \leftarrow kg(1^n)$
5. $r \leftarrow Rand(1^n)$
6. $T \leftarrow S(r, pk, U_1, U_2)$
7. Output the last entry ot T

succeeds in the one-way function pre-image finding experiment

$$u_2 \leftarrow \{0, 1\}^n, U_2 \leftarrow f(u_2), w \leftarrow S'(U_2)$$

In summary, we have shown that $\exists A \exists D \forall S$

$$\begin{aligned}&|Pr((pk', sk') \leftarrow kg'(1^n), r \leftarrow Rand(1^n),\\ &\qquad\qquad T \leftarrow A^{send(sk')}(r, pk') : D(r, pk', T) = 1)\\ &- Pr((pk', sk') \leftarrow kg(1^n), r \leftarrow Rand(1^n), T \leftarrow S(r, pk') : D(r, pk', T) = 1)|\\ &\qquad\qquad\qquad\qquad = 1 - negl(n)\end{aligned}$$

and therefore IMA' is not deniable. □

5.5 Plaintext aware encryption and deniability of protocol (5)

We have seen in the previous subsection that IND-CCA2 security of the encryption scheme and EUF-CMA security of the message authentication scheme are not enough to guarantee that the interactive message authentication protocol is deniable. In this section, we will see that a stronger requirement on the encryption scheme: that it is plaintext aware, is sufficient to ensure that (5) is deniable.

The intuitive idea behind plaintext aware encryption is to require the following: if an adversary outputs a ciphertext, then it must know the corresponding plaintext. A further idea is that we capture "knows the plaintext" by requiring that the plaintext be efficiently computable from the view of the adversary. Thus, we arrive at the following:

Definition 11 ([3]). *Let $ENC = (kg, enc, dec)$ be an asymmetric encryption scheme. We say that ENC is PA0 if for every efficient algorithm A, there exists an efficient algorithm A^* such that*

$$Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow Rand(1^n), c \leftarrow A(r, pk) : A^*(r, pk, c) \neq dec(sk, c))$$

is negligible.

A useful question for gaining intuition at this point is: in what sense is A^* different from a decryption algorithm? Why doesn't A^*'s ability to decrypt without the secret key contradict the security of the encryption scheme? The answer is the following: A^* "knows" the algorithm that produced the ciphertext c and its random coins; this is captured by the order of quantifiers $\forall A \exists A^*$ and by giving r as input to A^*.

Before we proceed to the more advanced notions of plaintext awareness, we remark that this basic notion is enough to prove a limited kind of deniability for protocol (5), namely, deniability in the case of a single sender session.

Proposition 2. *Let $IMA = (kg, send, rec)$ be the protocol* (5) *using a PA0 encryption scheme. Let A be any efficient algorithm that interacts with only a single send session. Then, there exists efficient S such that for all efficient D,*

$$\begin{aligned} |Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow Rand(1^n), T \leftarrow A^{send(sk)}(r, pk) : D(r, pk, T) = 1) \\ - Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow Rand(1^n), T \leftarrow S(r, pk) : D(r, pk, T) = 1)| \end{aligned}$$

is negligible.

Proof. We first spell out in detail how the interaction of A with the single sender session proceeds:

1. A receives input its random coins r and the public key pk. A computes a message m and asks the sender session to send m. The sender session outputs m together with a session id $s \leftarrow \{0, 1\}^n$.

2. A computes a ciphertext c to submit to the sender session; we denote this by $c \leftarrow A(r, s, pk)$. By the PA0 property of the encryption scheme, there is an efficient algorithm A^* that on input (r, s, pk) decrypts c with negligible probability of failure.

3. The sender session computes $k \leftarrow dec(sk, c)$ and $t \leftarrow tag(k, (m, s))$.

This suggests the following simulator S:

1. On input (r, pk) do the following:

2. $m \leftarrow A(r, pk)$

3. Draw $s \leftarrow \{0, 1\}^n$

4. $c \leftarrow A(r, s, pk)$

5. $k' \leftarrow A^*(r, s, pk)$
6. $t' \leftarrow tag(k', (m, s))$
7. Output the three message transcript $T' \leftarrow ((m, s), (m, s, c), (m, s, t'))$.

Since the probability that k' in step 5 differs from $k \leftarrow dec(sk, c)$ is negligible, no distinguisher can tell apart (r, pk, T) from (r, pk, T') except with negligible advantage. □

Thus, we see that the PA0 property is intuitively clear, and easy to define and use. Unfortunately, we can also see why it is not sufficient for our purposes: if we try to prove deniability for many concurrent sender sessions, we will have to deal with adversaries that produce not one but many ciphertexts.

Thus, we are led to the property PA1. We follow the definitional approach of [3]. Instead of requiring that A^* decrypts each of a sequence of ciphertexts correctly, we require that A^* be able to serve as a decryption oracle for A, without A, or an external distinguisher, noticing the difference.

Definition 12. *We say that $ENC = (kg, enc, dec)$ is PA1 if for all efficient A there exists efficient A^* such that for all efficient D,*

$$|Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), x \leftarrow A^{dec(sk)}(r, pk) : D(x) = 1)$$
$$- Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), x \leftarrow A^{A^*(r,pk)}(r, pk) : D(x) = 1|$$

is negligible. We remark that in acting as a decryption oracle, A^* is allowed to keep internal state between queries, and to have its own internal coin tosses.

Now, we will see that the PA1 property is enough to show that protocol (5) is deniable according to definition 10.

Theorem 7. *Let $IMA = (kg, send, rec)$ be protocol (5) with a PA1 encryption scheme. Then, IMA is deniable according to definition 10.*

Proof. Take any efficient A.

We look in detail at how the experiment

$$(pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), T \leftarrow A^{send(sk)}(r, pk), b \leftarrow D(r, pk, T)$$

from definition 10 proceeds.

1. $(pk, sk) \leftarrow kg(1^n)$
2. $r \leftarrow R(1^n)$
3. $T \leftarrow EmptyList$
4. While $A(r, pk)$ has not terminated
   (a) If A makes a query to $send(sk)$ with message m,

i. $s \leftarrow \{0,1\}^n$

ii. give (m,s) to A, add $(StartSession, m, (m,s))$ to T

(b) If A makes a (m,s,c) challenge to $send(sk)$

i. Check if there is an open session with associated values (m,s). If not, output $\perp$ to A, add $(Challenge, (m,s,c), \perp)$ to T. Else, continue.

ii. $k \leftarrow dec(sk, c)$

iii. $t \leftarrow tag(k, (m,s))$

iv. Give (m,s,t) to A, add $(Challenge, (m,s,c), (m,s,t))$ to T.

5. $\bar{T} \leftarrow (r, pk, T)$

6. $b \leftarrow D(\bar{T})$

Now, we think of the random coins r generated in line 2. and the random coins $s_1, \ldots s_w$ (with w being some polynomial function of n) generated in line 4.(a).i. as being generated by an extended randomness generation procedure $\bar{R}$.

We also think of lines 3., 4. (with all sub-items except the generation of the s_i's), 5. as a single algorithm $\bar{A}$ that takes input $(r, s_1, \ldots s_w, pk)$, makes queries to a decryption oracle in line 4.(b).ii., and outputs $\bar{T}$ at the end.

Thus, we have rewritten the experiment

$$(pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), T \leftarrow A^{send(sk)}(r, pk), b \leftarrow D(r, pk, T)$$

as the experiment

$$(pk, sk) \leftarrow kg(1^n), (r, \vec{s}) \leftarrow \bar{R}(1^n), \bar{T} \leftarrow \bar{A}^{dec(sk)}(r, \vec{s}, pk), b \leftarrow D(\bar{T})$$

Now, from the assumption that the encryption scheme is PA1, we deduce that there exists efficient $\bar{A}^*$ such that for all efficient D,

$$|Pr((pk, sk) \leftarrow kg(1^n), (r, \vec{s}) \leftarrow \bar{R}(1^n), \bar{T} \leftarrow \bar{A}^{dec(sk)}(r, \vec{s}, pk) : D(\bar{T}) = 1)$$
$$-Pr((pk, sk) \leftarrow kg(1^n), (r, \vec{s}) \leftarrow \bar{R}(1^n), \bar{T} \leftarrow \bar{A}^{\bar{A}^*(r,\vec{s},pk)}(r, \vec{s}, pk) : D(\bar{T}) = 1)|$$

is negligible.

Now, we look at the details of the experiment

$$(pk, sk) \leftarrow kg(1^n), (r, \vec{s}) \leftarrow \bar{R}(1^n), \bar{T} \leftarrow \bar{A}^{\bar{A}^*(r,\vec{s},pk)}(r, \vec{s}, pk), b \leftarrow D(\bar{T})$$

They are the same as the pseudo-code above, but with line 4.(b).ii replaced by $k \leftarrow \bar{A}^*(r, \vec{s}, pk)$, and with all the session ids s_i drawn at the beginning.

Now we think of the instructions in line 3. and the modified line 4. (with all the sub-items) as forming a single algorithm S, that takes input (r, pk), draws the random session ids, and produces output T. Thus, we have rewritten the experiment

$$(pk, sk) \leftarrow kg(1^n), (r, \vec{s}) \leftarrow \bar{R}(1^n), \bar{T} \leftarrow \bar{A}^{\bar{A}^*(r,\vec{s},pk)}(r, \vec{s}, pk), b \leftarrow D(\bar{T})$$

as the experiment

$$(pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), T \leftarrow S(r, pk), b \leftarrow D(r, pk, T)$$

Combining all observations so far, we see that $\forall A \exists S \forall D$

$$\begin{aligned}|Pr((pk, sk) \leftarrow kg(1^n) r \leftarrow R(1^n), T \leftarrow A^{send(sk)}(r, pk) : D(r, pk, T) = 1)\\ - Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), T \leftarrow S(r, pk) : D(r, pk, T) = 1)|\end{aligned}$$

is negligible. Therefore, IMA is deniable according to definition 10. □

Having seen that PA1 encryption is sufficient for deniability, it may be worth revisiting the example ENC' of IND-CCA2 secure encryption that is not sufficient (subsection 5.4). For the scheme ENC' constructed there, and for the particular problematic cyphertext $(1, U_1, U_2)$, we see that no efficient algorithm A^* can decrypt that cyphertext without access to the secret key. Thus, the scheme ENC' from that example is not PA0 or PA1.

5.6 Candidate construction of a PA1 encryption scheme

One encryption scheme that is conjectured to have the PA1 property is Damgard's variant [5] of the El Gamal encryption scheme [8]. The key generation $kg(1^n)$ operates as follows:

1. $(G, q, g) \leftarrow GroupGen(1^n)$ where G is a cyclic group of prime order q, g is a generator of G, and $GroupGen$ is the group generator that is conjectured to exist in the Decisional Diffie Hellman assumption (conjecture 1).
2. $(x, y) \leftarrow \{0, 1, \ldots q-1\}^2, X \leftarrow g^x, Y \leftarrow g^y$.
3. $pk \leftarrow (G, q, g, X, Y), sk \leftarrow (G, q, g, x, y)$, output (pk, sk).

Encryption $enc(G, q, g, X, Y, m)$ operates on messages $m \in G$ as follows:

1. $r \leftarrow \{0, \ldots q-1\}$
2. Output $(g^r, X^r, Y^r * m)$

Decryption $dec(G, q, g, x, y, c_1, c_2, c_3)$ operates on cyphertexts $(c_1, c_2, c_3) \in G \times G \times G$ as follows:

1. Check $c_1^x = c_2$, and if it fails, output $\perp$.
2. Else, if the check passes, output $c_3 * c_1^{-y}$.

The intuition for the IND-CPA security of this encryption scheme is the same as for El Gamal encryption and Diffie-Hellman key exchange. The intuition for the PA1 property is the following: it seems hard to produce c_1, c_2 that would pass the check without knowing the exponent r such that $c_1 = g^r, c_2 = X^r$. It is possible to formalize this intuition into an assumption that is tailor made to prove the PA1 property of the above encryption scheme; for details, see [3].

6 Deniable key exchange

In this section we explore deniability of key exchange protocols. We begin by introducing our candidate deniable key exchange protocol in subsection 6.1. We then give a definition of security for key exchange (subsection 6.2) and prove our example secure according to this definition (subsection 6.3). Next, we make some preliminary remarks on the deniability of our example (subsection 6.4). Then, we discuss multi-user plaintext awareness in subsection 6.5. We conclude by discussing deniability of our example in subsection 6.6. The exposition in this section is influenced by the ideas of [2, 6].

6.1 The SKEME protocol

Our running example in this section will be the SKEME protocol [11] that is used in the Internet Key Exchange proposed standards [9, 10]. We use the three flow version of the protocol presented in [6]:

$$
\begin{array}{ccc}
\text{Alice} & & \text{Bob} \\
x \leftarrow Z_q, s_A \leftarrow \{0,1\}^n, & & \\
k_A \leftarrow kgmac(1^n) & \xrightarrow{A, s_A, g^x, enc(pk_B, k_A)} & y \leftarrow Z_q, s_B \leftarrow \{0,1\}^n, \\
 & \xleftarrow{B, s_B, A, s_A, g^y, enc(pk_A, k_B), tag(k_A, (g^y, g^x, B, s_B, A, s_A))} & k_B \leftarrow kgmac(1^n) \\
\text{Output } g^{xy} & \xrightarrow{A, s_A, B, s_B, tag(k_B, (g^x, g^y, A, s_A, B, s_B))} & \text{Output } g^{xy}
\end{array}
$$

Here, Alice and Bob each have a public key secret key pair. They use the interactive authentication protocol (5) to authenticate to each other the pair of Diffie-Hellman terms g^x, g^y that they use in this session. Finally, they compute the output as the Diffie Hellman term g^{xy}.

Remark: In [6], the session key is computed as $prf(k_A, g^{xy}) + prf(k_B, g^{xy})$, where prf is a keyed pseudo random function family. This approach creates problems when writing a security proof. To see why, consider the modification MAC' of a EUF-CMA secure scheme MAC described in subsection 5.4. In MAC', which is also EUF-CMA secure, half of the symmetric keys k_A, k_B are revealed in the authentication tags. Now, how does one deal with the prf when a substantial fraction of its secret key is known? The present author does not know what [6] had in mind, and that paper does not attempt a proof of security for their version of SKEME. The paper [11] that introduces SKEME also does not include security proofs. In [2], a security proof appears for a protocol that is essentially the same as we are considering here. The present exposition of the protocol and its security proof was influenced by [2].

6.2 Security definition for key exchange

We would like to argue that SKEME is secure as a key exchange protocol. We first need to explain the definition of security for key exchange that we will use. We use the game-based (also known as indistinguishability based) approach, and model concurrent executions of many instances of the protocol, full adversarial control of the network, revealing of established session keys, and static corruptions.

The security experiment for key exchange is an interaction between a challenger and an adversary. The challenger simulates to the adversary interaction with users of the protocol. The adversary can guide this interaction by submitting queries to the challenger. In more detail, the experiment proceeds as follows:

1. The security experiment is parametrized by the number of corrupt and honest users. Let $m(n), m'(n)$ be functions of the security parameter that are bounded by a polynomial in n. We will denote the honest users by $\{U_1, \ldots, U_m\}$, and the corrupt users by $\{V_1, \ldots, V_{m'}\}$.

2. CH, A receive input the security parameter 1^n in unary.

3. CH draws $(pk, sk) \leftarrow kg(1^n)$ for all users and gives all public keys and the secret keys of corrupt users to A.

4. CH draws $(G, g, q) \leftarrow GroupGen(1^n)$ for use in the Diffie Hellman part of the protocol and gives (G, g, q) to A.

5. A may submit queries to the simulated honest users $\{U_1, \ldots, U_m\}$:

   (a) A may ask user U_i to start a new protocol session with intended partner $X \in \{U_1, \ldots U_m, V_1, \ldots V_{m'}\}$.

   (b) A may ask user U_i to process an incoming protocol message.

   (c) A may ask to take a challenge on the key computed by a particular session. A may ask to take a challenge once at any point during the experiment (in particular, A is allowed to continue asking other queries to the challenger after taking the challenge). To specify which session is tested, A must submit a tuple (U_i, s, X, s') such that user U_i has a completed session with associated values (U_i, s, X, s') and X is an honest user. CH draws random $b \leftarrow \{0, 1\}$ (sometimes it is more convenient to think of b as drawn outside CH and supplied to it as input) and if $b = 0$ replies with the real session key associated to (U_i, s, X, s') by user U_i, and if $b = 1$, CH draws an independent random string of the same length and replies with that.

   (d) A may ask to reveal the session key of a particular session. A specifies a tuple (U_i, s, X, s') where honest user U_i has completed a session with values (U_i, s, X, s') and X is any user (honest or corrupt). CH replies with the session key computed by U_i associated to (U_i, s, X, s'). The only restriction is that A is not allowed to reveal the session key of the test session or its partner, where if (U_i, s, X, s') is the test session, we define its partner to be any instance of user X that has values (X, s', U_i, s), if such instance exists.

6. When A has finished making queries, it computes a guess b' for the value of b. A wins if $b = b'$. The advantage of A is

$$\begin{aligned} AdvKE(A, CH) &= 2Pr((b, b') \leftarrow [A, CH] : b = b') - 1 \\ &= Pr([A, CH(1)] = 1) - Pr([A, CH(0)] = 1) \end{aligned}$$

The protocol is deemed secure if for all efficient A, $AdvKE(A, CH)$ is negligible.

6.3 Proof of security for SKEME

In this subsection, we show:

Theorem 8. *Let SKEME be instantiated with encryption scheme ENC message authentication scheme MAC and cyclic group generator GroupGen. Let $m(n), m'(n)$ be functions of the security parameter that are bounded by a polynomial in n, and consider the security experiment with m honest and m' corrupt users. Let A be an efficient adversary that creates at most N sessions of honest users. Then, there exist efficient A', A'', A''' such that*

$$\begin{aligned} AdvKE(A, SKEME) \leq \frac{2N^2}{2^n} &+ 2AdvCCA2_{m,N}(A', ENC) \\ &+ 2AdvCMA_N(A'', MAC) + 2AdvDDH(A''', GroupGen) \end{aligned}$$

Corollary 2. *Suppose SKEME is instantiated with a IND-CCA2 secure asymmetric encryption scheme, a EUF-CMA secure symmetric message authentication scheme, and a group generator for which the Decisional Diffie-Hellman assumption holds. Then, SKEME is secure in the sense of subsection 6.2.*

Proof. Our intuition tells us that there are several methods by which an adversary may try to break the protocol. The first method is to attempt to mismatch sessions. The second method is by attempting to break the authentication of the protocol; as in the proof of Theorem 6, this can be broken down further into attempting to learn something about the ephemeral authentication key, or trying to break authentication without knowing anything about the ephemeral key. Finally, the adversary may attempt to learn something about the generated session key via an eavesdropping attack.

Out proof follows this intuition. We use the sequence of games technique. In each successive game, we limit the adversary's ability to pursue one of the above strategies. At a high level, the games are:

1. Let $[A, CH_0]$ denote the interaction in the security experiment for SKEME as described in subsection 6.2.

2. Next, we limit the adversary's ability to mismatch sessions. Let CH_1 operate as CH_0, except that whenever a new s value must be drawn for some user instance, CH_1 ensures that this value does not collide with any

previously observed s value, either adversary or user generated. In the interaction $[A, CH_1]$ for every honest user instance with associated values (X, s, Y, s'), there exists at most one honest instance with associated values (Y, s', X, s).

3. Next, we limit the adversary's ability to learn something about the ephemeral symmetric keys used for authentication. Let CH_2 be as CH_1, except that whenever a user instance has as partner one of the honest users $X \in \{U_1, \ldots U_m\}$ and needs to generate a symmetric key for authentication, it generates two independent keys $k_0, k_1 \leftarrow kgmac(1^n)^{\otimes 2}$. k_1 is used to generate the cyphertext $c \leftarrow enc(pk_X, k_1)$ that is sent out, and k_0 is used to verify incoming authentication tags. In addition, whenever c is delivered to user X, CH_2 knows to substitute k_0 instead of k_1 in the decryption. Thus, in effect, CH_2 ensures an "ideally secret" delivery of k_0 to X.

4. Next, we limit the adversary's ability to break the authentication mechanism. Let CH_3 be as CH_2, except that when a user instance receives an authentication tag, CH_3 not only verifies the correctness of the authentication tag, but also verifies that this authentication tag was produced by its partner instance.

5. Finally, we limit the ability of the adversary to obtain information about the session keys by breaking the Diffie Hellman mechanism. Let CH_4 be as CH_3, except that when two honest user instances have authenticated to each other the values (g^x, g^y, X, s, Y, s') and (g^y, g^x, Y, s', X, s), CH_4 generates their session key as an independent random group element instead of as g^{xy}.

Now, we proceed with the details. The interaction $[A, CH_0(b)]$ can be specified in pseudo-code as follows:

0. $b \leftarrow \{0, 1\}$.
1. On input the security parameter 1^n, and the secret bit b (alternatively, we can think of b as drawn inside the interaction; we switch between the two ways of thinking as needed).
2. For $i = 1, \ldots m$, $(pk_{U_i}, sk_{U_i}) \leftarrow kg(1^n)$.
3. $\vec{pk}_U \leftarrow (pk_{U_1}, \ldots, pk_{U_m})$, $\vec{sk}_U \leftarrow (sk_{U_1}, \ldots, sk_{U_m})$.
4. For $j = 1, \ldots m'$, $(pk_{V_j}, sk_{V_j}) \leftarrow kg(1^n)$.
5. $\vec{pk}_V \leftarrow (pk_{V_1}, \ldots, pk_{V_{m'}})$, $\vec{sk}_V \leftarrow (sk_{V_1}, \ldots, sk_{V_{m'}})$.
6. $T \leftarrow EmptyList$
7. While $A(\vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$ has not terminated:

(a) If A makes a $(message_0, X, Y)$ query instructing honest user $X \in \{U_1, \ldots U_m\}$ to start a new session with honest or corrupt user $Y \in \{U_1, \ldots, U_m, V_1, \ldots V_{m'}\}$ then

   i. Draw $s \leftarrow \{0,1\}^n$.

   ii. Draw $x \leftarrow Z_q$. Associate x to (X, s, Y).

   iii. Draw $k_0 \leftarrow kgmac(1^n), k_1 \leftarrow kgmac(1^n)$ independently. Associate k_0 to (X, s, Y). The key k_1 plays no further role in this game but will be used in subsequent games.

   iv. $c \leftarrow enc(pk_Y, k_0)$. Associate k_0 to (Y, c).

   v. Output (X, s, g^x, c) to A and append $((message_0, X, Y), (X, s, g^x, c))$ to T.

(b) If A makes a $(message_1, Y, (X, s, h, c))$ query to honest user $Y \in \{U_1, \ldots U_m\}$ then

   i. Draw $s' \leftarrow \{0,1\}^n$.

   ii. Draw $y \leftarrow Z_q$. Associate y to (Y, s', X, s). Also associate the peer DH element h to (Y, s', X, s).

   iii. Draw $k'_0, k'_1 \leftarrow kgmac(1^n)$ independently. Associate k'_0 to (Y, s', X, s). The key k'_1 plays no further role in this game but will be used in subsequent games.

   iv. $c' \leftarrow enc(pk_X, k'_0)$. Associate k'_0 to (X, c').

   v. If there is a key associated to the pair (Y, c) (i.e. if c was previously produced by an honest user using Y's public key), then retrieve the associated k.

   vi. Else $k \leftarrow dec(sk_Y, c)$.

   vii. $t' \leftarrow tag(k, (g^y, h, Y, s', X, s))$

   viii. Output $(Y, s', X, s, g^y, c', t')$ to A and append

$$((message_1, Y, (X, s, h, c)), (Y, s', X, s, g^y, c', t'))$$

   to T.

(c) If A makes a $(message_2, X, (Y, s', X, s, h', c', t'))$ to honest user $X \in \{U_1, \ldots U_m\}$ then

   i. If there is no open session with values (X, s, Y), output $\perp$ to A and append $((message_2, X, (Y, s', X, s, h', c', t')), \perp)$ to T.

   ii. Else retrieve the symmetric key k and the DH exponent x associated with (X, s, Y).

   iii. If $vrf(k, (h', g^x, Y, s', X, s), t') = 0$, output $\perp$ to A and append $((message_2, X, (Y, s', X, s, h', c', t')), \perp)$ to T, else continue.

   iv. Compute session key $(h')^x$. Associate this session key to (X, s, Y, s').

   v. If there is a key associated with (X, c') (i.e. if c' was previously produced by an honest user using X's public key), retrieve the corresponding k'.

vi. Else $k' \leftarrow dec(sk_X, c')$.

vii. $t \leftarrow tag(k', (g^x, h', X, s, Y, s'))$.

viii. Output (X, s, Y, s', t) to A and append

$$((message_2, X, (X, s, Y, s', h', c', t')), (X, s, Y, s', t))$$

to T.

(d) If A makes a $(message_3, Y, (X, s, Y, s', t))$ query to honest user $Y \in \{U_1, \ldots U_m\}$ then

i. If there is no open session with values (Y, s', X, s), output $\perp$ to A and append $((message_3, Y, (X, s, Y, s', t)), \perp)$ to T.

ii. Else retrieve the symmetric key k', the DH exponent y, and the peer DH group element h associated to values (Y, s', X, s).

iii. If $vrf(k', (h, g^y, X, s, Y, s'), t) = 0$, output $\perp$ to A and append $((message_3, Y, (X, s, Y, s', t)), \perp)$ to T, else continue.

iv. Compute session key h^y. Associate this session key to (Y, s', X, s).

v. Output $SessionDone$ to A and append

$$((message_3, Y, (X, s, Y, s', t)), SessionDone)$$

to T.

(e) If A makes a $(RevealSessionKey, (X, s, Y, s'))$ query to a session of honest user $X \in \{U_1, \ldots U_m\}$, then

i. If there is no completed session with values (X, s, Y, s'), then output $\perp$ to A and append $((RevealSessionKey, (X, s, Y, s')), \perp)$ to T.

ii. Else if session (X, s, Y, s') or (Y, s', X, s) is marked as tested, then output $\perp$ to A and append

$$((RevealSessionKey, (X, s, Y, s')), \perp)$$

to T.

iii. Else retrieve the session key $h \in G$ associated to values (X, s, Y, s'), mark (X, s, Y, s') and (Y, s', X, s) as revealed, output h to A and append $((RevealSessionKey, (X, s, Y, s')), h)$ to T.

(f) If A makes a $(Test, (X, s, Y, s'))$ query to a session of honest user $X \in \{U_1, \ldots U_m\}$ whose intended partner Y is also an honest user, then

i. Check that there was no prior $Test$ query in T. If there was, output $\perp$ to A and append $((Test, (X, s, Y, s')), \perp)$ to T.

ii. If there is no completed session with values (X, s, Y, s'), then output $\perp$ to A and append $((Test, (X, s, Y, s')), \perp)$ to T.

iii. Else if session (X, s, Y, s') or (Y, s', X, s) is marked as revealed, then output $\perp$ to A and append

$$((Test, (X, s, Y, s')), \perp)$$

to T.

iv. Else, if $b = 0$ retrieve the session key $h \in G$ associated to values (X, s, Y, s') and if $b = 1$ draw $h \leftarrow G$.

v. Mark (X, s, Y, s') and (Y, s', X, s) as tested, output h to A and append $((Test, (X, s, Y, s')), h)$ to T.

8. When A outputs b' output b'.

In the interaction $[A, CH_1]$ we modify the following lines:

7.(a).i' Draw $s \leftarrow \{0,1\}^n - \{\text{session ids that have appeared previously}\}$

7.(b).i' Draw $s' \leftarrow \{0,1\}^n - \{\text{session ids that have appeared previously}\}$

Now, we have to place upper bounds on the change in adversary advantage from one game to the next. We have

$$|Pr((b, b') \leftarrow [A, CH_0] : b = b') - Pr((b, b') \leftarrow [A, CH_1] : b = b')| \leq \frac{N^2}{2^n}$$

This is because the interactions $[A, CH_0], [A, CH_1]$ proceed identically unless in the first interaction there is a collision of a newly drawn s value with a previously observed one, and this occurs with probability at most $N^2 2^{-n}$. Then, we have

$$|AdvKE(A, CH_0) - AdvKE(A, CH_1)| \leq \frac{2N^2}{2^n} \tag{8}$$

Next, we consider the second game. In the interaction $[A, CH_2]$, the lines 7.(a).iv, 7.(b).iv are modified to:

7.(a).iv' If Y is honest $c \leftarrow enk(pk_Y, k_1)$. However, associate k_0 to (Y, c), so that k_0 is retrieved in lines 7.(b).v, 7.(c).v. If Y is corrupt, $c \leftarrow enk(pk_Y, k_0)$ and associate k_0 to (Y, c).

7.(b).iv' If X is honest, $c' \leftarrow enk(pk_X, k_1')$. However, associate k_0' to (X, c'), so that k_0' is retrieved in lines 7.(b).v, 7.(c).v. If X is corrupt, $c' \leftarrow enk(pk_X, k_0')$ and associate k_0' to (X, c').

Now, we need to give a bound for $|AdvKE(A, CH_1) - AdvKE(A, CH_2)|$. Think of the instructions in lines 0,1,3,4,5,6,7,8 as an algorithm A' that participates in the m-key N-challenge IND-CCA2 security experiment (subsection 3.2). We interpret the (pk, sk) pairs of honest users as the keys of the IND-CCA2 experiment. We interpret the honest intended partner case of lines 7.(a).iv, 7.(b).iv, 7.(a).iv', 7.(b).iv' as A' requesting a challenge cyphertext. We interpret the corrupt intended partner case as A' performing encryption by itself. We interpret lines 7.(b).vi, 7.(c).vi as A' querying the decryption oracle. At the end, A' outputs 1 if $b = b'$ and 0 otherwise. We have

$$Pr((b,b') \leftarrow [A, CH_1] : b = b') = Pr([A', MKMC - CCA2(0^{m \times N})] = 1)$$
$$Pr((b,b') \leftarrow [A, CH_2] : b = b') = Pr([A', MKMC - CCA2(1^{m \times N})] = 1)$$

Therefore,

$$|AdvKE(A, CH_1) - AdvKE(A, CH_2)| \leq 2AdvCCA2_{m,N}(A', ENC) \qquad (9)$$

Next, we consider the third game. In the interaction $[A, CH_3]$, the lines 7.(c).iii, 7.(d).iii are modified:

7.(c).iii' If the intended partner Y is corrupt, perform 7.(c).iii. Else: If

$$vrf(k, (h', g^x, Y, s', X, s), t') = 0$$

or T does not contain the session (Y, s', X, s) producing tag t' on values (h', g^x, Y, s', X, s), output $\perp$ to A and append

$$((message_2, X, (Y, s', X, s, h', c', t')), \perp)$$

to T, else continue.

7.(d).iii' If the intended partner X is corrupt, perform 7.(d).iii. Else: If

$$vrf(k', (h, g^y, X, s, Y, s'), t) = 0$$

or T does not contain the session (X, s, Y, s') producing tag t on values (h, g^y, X, s, Y, s'), output $\perp$ to A and append

$$((message_3, Y, (X, s, Y, s', t)), \perp)$$

to T, else continue.

Now, we need to give a bound for $|AdvKE(A, CH_2) - AdvKE(A, CH_3)|$. Think of the pseudo-code of the interaction $[A, CH_2]$ (with modifications explained below) as an algorithm A'' participating in the N-key EUF-CMA experiment (subsection 3.3). We interpret the keys k_0 drawn by honest user sessions whose intended partner is also an honest user as the secret keys of the N-key EUF-CMA experiment. If A'' needs to produce a tag under one of these keys in lines 7.(b).vii and 7.(c).vii, then A'' queries the corresponding tag oracle of the EUF-CMA experiment. If A'' needs to verify a tag under one of these keys in lines 7.(c).iii, 7.(d).iii, then it queries the corresponding vrf oracle of the EUF-CMA experiment.

Thus, we have shown that the same pseudo-code can be thought of either as the interaction $[A, CH_2]$ or as the interaction $[A'', MK - CMA]$, where $MK - CMA$ denotes the challenger for the N-key EUF-CMA security experiment.

Now, observe that the interaction $[A, CH_2]$ proceeds identically to the interaction $[A, CH_3]$, unless A'' wins the EUF-CMA experiment in the alternative interpretation $[A'', MK - CMA]$ of $[A, CH_2]$. Therefore,

$$|Pr((b, b') \leftarrow [A, CH_2] : b = b') - Pr((b, b') \leftarrow [A, CH_3] : b = b')|$$
$$\leq Pr(T'' \leftarrow [A'', MK - CMA] : B(T'')) = AdvCMA_N(A'', MAC) \quad (10)$$

and

$$|AdvKE(A, CH_2) - AdvKE(A, CH_3)| \leq 2AdvCMA_N(A'', MAC) \quad (11)$$

We have one more game left, game number 4. Before we describe the changes in game 4, we comment on the consequences of the design of game 3. In game 3, a session of honest user X, with values (X, s, Y, s'), whose intended partner is honest user Y accepts only if there is exactly one session of Y with values (Y, s', X, s), and moreover the two sessions $(X, s, Y, s'), (Y, s', X, s)$ are using the same pair of group elements for the Diffie-Hellman mechanism.

Now, we describe the game $[A, CH_4]$. The lines 7.(c).iv, 7.(d).iv are modified:

7.(c).iv' If the intended partner Y is corrupt, execute 7.(c).iv. Else, compute the session key as $\gamma \leftarrow G$ and associate it to session (X, s, Y, s').

7.(d).iv' If the intended partner X is corrupt, execute 7.(d).iv. Else, find the session key associated to session (X, s, Y, s'), and associate it also to session (Y, s', X, s).

Next, we claim there exists A''' such that

$$|Pr((b, b') \leftarrow [A, CH_3] : b = b') - Pr((b, b') \leftarrow [A, CH_4] : b = b')|$$
$$\leq AdvDDH(A''', GroupGen)$$

and therefore,

$$|AdvKE(A, CH_3) - AdvKE(A, CH_4)| \leq 2AdvDDH(A''', GroupGen) \quad (12)$$

To see this, let A''' be an algorithm trying to decide whether the triple of group elements (α, β, γ) it receives is a Diffie Hellman triple or three independent random group elements. A''' plays the role of CH_3 or CH_4 for A, with the following changes:

1. A''' gives the specification of the group (G, g, q) it receives from its own challenger to A.

2. When a new instance of some honest user X is created as protocol initiator with intended partner honest user Y, A''' draws $a \leftarrow Z_q$ and computes the Diffie Hellman term for that instance of X as αg^a.

3. If a new instance of honest user Y is created as protocol responder with intended partner honest user X, then A''' draws $b \leftarrow Z_q$ and computes the Diffie Hellman term for that instance of Y as βg^b.

4. If (X, s, Y, s') is a session of honest user X acting as protocol initiator with intended partner honest user Y, and if X has verified the authentication tag associating DH elements $(\beta g^b, \alpha g^a)$ to (Y, s', X, s), then A''' computes the session key associated to (X, s, Y, s') as $\gamma\alpha^b\beta^a g^{ab}$.

5. If (Y, s', X, s) is a session of honest user Y acting as protocol responder with intended partner honest user X, and if Y has verified the authentication tag associating DH elements $(\alpha g^a, \beta g^b)$ to (X, s, Y, s'), then A''' computes the session key associated to (Y, s', X, s) to be the same as the key associated to (X, s, Y, s') (i.e. $\gamma\alpha^b\beta^a g^{ab}$).

We see that if (α, β, γ) is drawn as a Diffie Hellman triple, then A''' provides the same view to A as CH_3, and if (α, β, γ) is drawn as a triple of independent random group elements, then A''' provides the same view to A as CH_4. This proves equation (12).

Finally, we see that in Game 4, the view of the adversary is independent of the hidden bit b. Therefore,

$$AdvKE(A, CH_4) = 0 \tag{13}$$

Combining equations (8), (9), (11), (12), (13) we complete the proof of the theorem. □

6.4 Preliminary remarks on the deniability of SKEME

We have already seen many of the ideas that allow us to define deniability for key exchange and prove that the SKEME protocol is deniable. These ideas are: the definition of deniability as the ability to simulate the view of an adversary interacting with an honest user (subsection 5.3), the observation that IND-CCA2 secure encryption and EUF-CMA secure authentication are not sufficient for deniability against an arbitrary receiver (subsection 5.4), and the observation that another requirement on encryption, plaintext awareness (subsection 5.5), does suffice for the authentication protocol underlying SKEME to be deniable.

Thus, we are almost ready to declare that SKEME is deniable. However, there is an additional complication in the case of SKEME: in the security experiment from subsection 6.2, there are multiple (pk, sk) pairs of honest users, while in the definition of PA1 encryption (definition 12) there is only one such pair. In the next subsection, we discuss this complication.

6.5 Multi-user plaintext awareness

We begin by defining multi-user PA1 encryption. Let $m(n)$ be a polynomially bounded function and let $(\vec{pk}, \vec{sk}) \leftarrow kg(1^n)^{\otimes m}$ denote independently drawing $m(n)$ (pk, sk) pairs.

Definition 13. *$ENC = (kg, enc, dec)$ is multi-user PA1 if for all polynomially bounded $m(n)$, for all efficient A there exists efficient A^* such that for all* *efficient D*

$$|Pr((\vec{pk}, \vec{sk}) \leftarrow kg(1^n)^{\otimes m}, r \leftarrow R(1^n), x \leftarrow A^{dec(\vec{sk})}(r, \vec{pk}) : D(x) = 1)$$
$$- Pr((\vec{pk}, \vec{sk}) \leftarrow kg(1^n)^{\otimes m}, r \leftarrow R(1^n), x \leftarrow A^{A^*(r,\vec{pk})}(r, \vec{pk}) : D(x) = 1)|$$

is negligible.

We note that A must specify $i \in \{1, \dots m\}$ when querying its oracle.

At this point, a natural question is this: does single-user PA1 imply multi-user PA1? Could, for example, one prove such an implication by a standard hybrid argument, replacing the decryption oracles one by one? The author of this paper attempted the hybrid argument approach, but encountered a problem. After the first decryption oracle is replaced by an A^* algorithm, one gets to a situation in which the ordinary PA1 definition no longer applies. To illustrate the difficulty more concretely, we write an attempted hybrid argument for the case of two key pairs, and point out where we get stuck.

Assume $ENC = (kg, enc, dec)$ is single-user PA1. Given algorithm A, the experiment in which two key pairs are drawn, and algorithm A is allowed to adaptively query two decryption oracles can be described in pseudo-code as follows:

1. $(pk_1, sk_1) \leftarrow kg(1^n)$.
2. $(pk_2, sk_2) \leftarrow kg(1^n)$.
3. $r \leftarrow R(1^n)$ (generation of the random coins for A).
4. While $A(r, pk_1, pk_2)$ has not terminated:
   (a) If A makes a query c to the first decryption oracle
      i. $m \leftarrow dec(sk_1, c)$.
      ii. Return m to A.
   (b) If A makes a query c to the second decryption oracle
      i. $m \leftarrow dec(sk_2, c)$.
      ii. Return m to A.
5. A outputs x and terminates.

Now, we want to apply the single user PA1 definition (definition 12) and replace the first decryption oracle by an algorithm that does not use the first secret key. First, we need to identify the algorithm that is making queries to the first decryption oracle: this algorithm consists of lines 2, 3, 4, 5, and makes decryption oracle queries in line 4.(a).i. The important point is that *the algorithm that makes queries to the first decryption oracle contains in its view the second secret key.*

Let $\bar{A}$ denote the algorithm consisting of lines 2,3,4,5. Algorithm $\bar{A}$ uses A as a subroutine and makes queries to the oracle $dec(sk_1)$. Let $\bar{r} \leftarrow \bar{R}(1^n)$

denote drawing the coins for $\bar{A}$. The coins $\bar{r}$ consist of two parts: the coins r for subroutine A, and the coins needed to draw the pair (pk_2, sk_2).

Thus, we have rewritten the experiment

$$(pk_1, sk_1) \leftarrow kg(1^n), (pk_2, sk_2) \leftarrow kg(1^n), r \leftarrow R(1^n), x \leftarrow A^{dec(sk_1), dec(sk_2)}(r, pk_1, pk_2)$$

as the experiment

$$(pk_1, sk_1) \leftarrow kg(1^n), \bar{r} \leftarrow \bar{R}(1^n), x \leftarrow \bar{A}^{dec(sk_1)}(\bar{r}, pk_1)$$

Now, we apply definition 12. There exists $\bar{A}^*$ such that for all D,

$$\begin{aligned}|Pr((pk_1, sk_1) \leftarrow kg(1^n), \bar{r} \leftarrow \bar{R}(1^n), x \leftarrow \bar{A}^{dec(sk_1)}(\bar{r}, pk_1) : D(x) = 1)\\ - Pr((pk_1, sk_1) \leftarrow kg(1^n), \bar{r} \leftarrow \bar{R}(1^n), x \leftarrow \bar{A}^{\bar{A}^*(\bar{r}, pk_1)}(\bar{r}, pk_1) : D(x) = 1)|\end{aligned}$$

is negligible.

Now, we look at the experiment

$$(pk_1, sk_1) \leftarrow kg(1^n), \bar{r} \leftarrow \bar{R}(1^n), x \leftarrow \bar{A}^{\bar{A}^*(\bar{r}, pk_1)}(\bar{r}, pk_1)$$

and write it in pseudo-code:

1. $(pk_1, sk_1) \leftarrow kg(1^n)$.
2. $(r, r_{kg}) \leftarrow \bar{R}(1^n)$.
3. $(pk_2, sk_2) \leftarrow kg(1^n, r_{kg})$.
4. While $A(r, pk_1, pk_2)$ has not terminated:
   (a) If A makes a query c to the first decryption oracle
      i. $m \leftarrow \bar{A}^*(r, r_{kg}, pk_1)$.
      ii. Return m to A.
   (b) If A makes a query c to the second decryption oracle
      i. $m \leftarrow dec(sk_2, c)$.
      ii. Return m to A.
5. A outputs x and terminates.

Now, we encounter a problem: there appears to be no way to view this pseudo-code as an algorithm that is entirely ignorant of sk_2 that makes queries to the oracle $dec(sk_2)$. This is because $\bar{A}^*$ takes as input r_{kg}, the coins that were used to generate (pk_2, sk_2).

The authors of [6] attempt to get around this problem by using PA2 encryption (see below for a definition) instead of PA1. However, the present author does not believe that PA2 encryption would solve the problem, as the definition of PA2 also involves a security experiment with a single (pk, sk) pair. Attempting a hybrid argument to show that single user PA2 implies multi-user PA2 seems to run into even more problems than the PA1 case. To be concrete, we give below definitions for single-user and multi-user PA2 plaintext awareness, attempt a hybrid argument to show that single-user PA2 implies multi-user PA2, and explain where we get stuck.

We begin with the definition of PA2 plaintext awareness. Intuitively, the requirement on PA2 encryption is the following: even if an algorithm has an external source of cyphertexts for which it does not know the plaintexts, that algorithm is not able to produce any new (not provided by the external source) cyphertext of which it does not know the corresponding plaintext. In the definition below, all details of the external source of cyphertexts are abstracted into a single oracle available to the adversary:

Definition 14 ([3]). *$ENC = (kg, enc, dec)$ is PA2 if for all efficient A there exists efficient A^* such that for all efficient P and for all efficient D*

$$|Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), x \leftarrow A^{dec(sk), enc(pk)\circ P}(r, pk) : D(x) = 1)$$
$$-Pr((pk, sk) \leftarrow kg(1^n), r \leftarrow R(1^n), x \leftarrow A^{A^*(r,pk), enc(pk)\circ P}(r, pk) : D(x) = 1)|$$

is negligible. Here, the oracle $enc(pk) \circ P$ operates as follows:

1. *A submits a query q.*
2. *P computes a plaintext m based on q. P is allowed to keep state between queries and is allowed its own internal coin tosses.*
3. *$c \leftarrow enc(pk, m)$ is given to A*
4. *A is not allowed to query c on the decryption oracle interface.*

This definition does not match what we need in the analysis of SKEME. The reason is that definition 14 involves only a single (pk, sk) pair, while the security experiment for key exchange (subsection 6.2) contains many (pk, sk) pairs. Thus, we present a modified definition with many (pk, sk) pairs.

Definition 15. *$ENC = (kg, enc, dec)$ is multi-user PA2 if for all polynomially bounded $m(n)$, for all efficient A there exists efficient A^* such that for all efficient P and for all efficient D*

$$|Pr((\vec{pk}, \vec{sk}) \leftarrow kg(1^n)^{\otimes m}, r \leftarrow R(1^n), x \leftarrow A^{dec(\vec{sk}), enc(\vec{pk})\circ P}(r, \vec{pk}) : D(x) = 1)$$
$$-Pr((\vec{pk}, \vec{sk}) \leftarrow kg(1^n)^{\otimes m}, r \leftarrow R(1^n), x \leftarrow A^{A^*(r,\vec{pk}), enc(\vec{pk})\circ P}(r, \vec{pk}) : D(x) = 1)|$$

is negligible.

We note that A must specify $i \in \{1, \dots m\}$ when querying its oracles.

Now, we consider the case of two key pairs and attempt a hybrid argument to replace the real decryption oracles one by one.

Assume $ENC = (kg, enc, dec)$ is single-user PA2. Take any A. Take any P. The experiment of drawing two key pairs, and allowing A adaptive queries to $dec(sk_1), dec(sk_2), enc(pk_1) \circ P, enc(pk_2) \circ P$ can be written in pseudo-code as follows:

1. $(pk_1, sk_1) \leftarrow kg(1^n)$.
2. $(pk_2, sk_2) \leftarrow kg(1^n)$.
3. $r \leftarrow R(1^n)$.
4. While $A(r, pk_1, pk_2)$ has not terminated:
   (a) If A asks $(dec, 1, c)$ query, and if c was not previously output by the oracle $enc(pk_1) \circ P$,
      i. $m \leftarrow dec(sk_1, c)$.
      ii. Return m to A.
   (b) If A asks $(dec, 2, c)$ query, and if c was not previously output by the oracle $enc(pk_2) \circ P$,
      i. $m \leftarrow dec(sk_2, c)$.
      ii. Return m to A.
   (c) If A asks $(enc, 1, q)$ query,
      i. $m \leftarrow P(1, q)$,
      ii. $c \leftarrow enc(pk_1, m)$,
      iii. Return c to A.
   (d) If A asks $(enc, 2, q)$ query,
      i. $m \leftarrow P(2, q)$,
      ii. $c \leftarrow enc(pk_2, m)$,
      iii. Return c to A.
5. A outputs x.

Now, we want to apply the single-user PA2 definition. To this end, we have to view the above pseudo-code as a single algorithm that makes queries to $dec(sk_1), enc(pk_1) \circ P$. We immediately encounter a problem: that algorithm would have to use P as a subroutine to handle queries $(enc, 2, q)$ by A.

We may try to avoid this first problem by considering only independent algorithms P_1, P_2 that handle $(enc, 1, q)$ and $(enc, 2, q)$ queries respectively. Thus, given A, P_1, P_2 let $\mathcal{A}(A, P_2)$ be the algorithm consisting of lines 2,3,4,5 above and making oracle queries to $dec(sk_1)$ and $enc(pk_1) \circ P_1$. The algorithm $\mathcal{A}(A, P_2)$ takes as input pk_1. It needs as input three different strings of random coins: (r, r_{kg}, r_P), where r are the random coins for subroutine A, r_{kg} are the random coins needed to generate the second key pair, and r_P are the random coins needed for subroutine P_2. Thus, we have rewritten the experiment above as

$$(pk_1, sk_1) \leftarrow kg(1^n), (r, r_{kg}, r_P) \leftarrow \bar{R}(1^n),$$
$$x \leftarrow \mathcal{A}(A, P_2)^{dec(sk_1), enc(pk_1) \circ P_1}(r, r_{kg}, r_P, pk_1)$$

Applying the single-user PA2 definition we obtain: for all A, for all P_2 there exists $\mathcal{A}^*$ such that for all P_1, for all D

$$\begin{aligned}|Pr((pk_1, sk_1) \leftarrow kg(1^n), (r, r_{kg}, r_P) \leftarrow \bar{R}(1^n),\\ x \leftarrow \mathcal{A}(A, P_2)^{dec(sk_1), enc(pk_1)\circ P_1}(r, r_{kg}, r_P, pk_1) : D(x) = 1)\\ - Pr((pk_1, sk_1) \leftarrow kg(1^n), (r, r_{kg}, r_P) \leftarrow \bar{R}(1^n),\\ x \leftarrow \mathcal{A}(A, P_2)^{\mathcal{A}^*(r, r_{kg}, r_P, pk_1), enc(pk_1)\circ P_1}(r, r_{kg}, r_P, pk_1) : D(x) = 1)|\end{aligned}$$

is negligible.

Now, our problems have multiplied. To begin with, we have the problem that we had in the PA1 case: that $\mathcal{A}^*$ takes as input the random coins r_{kg} that are used to generate the second key pair. In addition, we have the problem that $\mathcal{A}^*$ "knows everything" about P_2: this is because of the order of quantifiers $\forall P_2, \exists \mathcal{A}^*$, and because $\mathcal{A}^*$ takes the random coins r_P of P_2 as input. In such a situation, it is not clear how the next step of the hybrid argument could be taken.

After contemplating the above problems, one gets the feeling that the notion multi-user PA2 may be genuinely different than single-user PA2. Intuitively the difference is the following: in single user PA2, the claim is that one cannot change cyphertexts with unknown plaintext under one key pair into other cyphertexts with unknown plaintext under the same key pair. In multi-user PA2, this is strengthened to say that one cannot change cyphertexts with unknown plaintexts of any key pair into cyphertexts with unknown plaintexts of the same or any other key pair.

In conclusion, the present author does not know whether the multi-user versions of PA1 and PA2 plaintext awareness are equivalent or strictly stronger than the single-user versions. We leave this question for future work.

6.6 Deniability of SKEME

We now proceed to argue that SKEME is deniable if the underlying encryption scheme is multi-user PA1. Our first task is to state a definition of deniability for a key exchange protocol. Consider the key exchange security experiment from subsection 6.2. We are interested in the view of the adversary in this experiment; the view consists of the inputs, the transcript of the interaction with the challenger, and any output that A may produce. For the purposes of discussing deniability, the query with which the adversary asks to take a challenge is superfluous, so we eliminate it. We use the summary notation

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow [A(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V), CH]$$

to denote the generation of the adversary view; here, we use

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n)$$

to denote an initialization procedure which draws random coins for the adversary and vectors of public keys and secret keys for the honest users $\{U_1, \ldots U_m\}$ and for the corrupt users $\{V_1, \ldots V_{m'}\}$. With this notation, we can define deniability for key exchange as follows:

Definition 16. *A key exchange protocol is deniable if for all polynomially bounded $m(n), m'(n)$, for all efficient A, there exists efficient S such that for all efficient D*

$$\begin{aligned}|Pr((\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow [A(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V), CH] \\ : D(View) = 1) \\ - Pr((\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow S(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V) \\ : D(View) = 1)|\end{aligned}$$

is negligible.

We will show the following:

Theorem 9. *Let SKEME be initialized with a multi-user PA1 encryption scheme. Then, SKEME is deniable.*

Proof. Take any polynomially bounded $m(n), m'(n)$. Take any efficient A. We look in detail at how the experiment

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow [A(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V), CH]$$

proceeds.

1. $(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n)$
2. $View \leftarrow (r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$
3. While $A(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$ has not terminated:
   (a) If A makes a $(message_0, X, Y)$ query instructing honest user $X \in \{U_1, \ldots U_m\}$ to start a new session with honest or corrupt user $Y \in \{U_1, \ldots, U_m, V_1, \ldots V_{m'}\}$ then
      i. Draw $x \leftarrow Z_q$.
      ii. Draw $s \leftarrow \{0,1\}^n$.
      iii. Draw $k \leftarrow kgmac(1^n)$.
      iv. $c \leftarrow enc(pk_Y, k)$.
      v. Output (X, s, g^x, c) to A and append $((message_0, X, Y), (X, s, g^x, c))$ to $View$.
   (b) If A makes a $(message_1, Y, (X, s, h, c))$ query to honest user $Y \in \{U_1, \ldots U_m\}$ then
      i. Draw $y \leftarrow Z_q$.
      ii. Draw $s' \leftarrow \{0,1\}^n$.
      iii. Draw $k' \leftarrow kgmac(1^n)$.

iv. $c' \leftarrow enc(pk_Y, k')$.

v. If c was previously produced by an honest user using Y's public key, then retrieve the corresponding k.

vi. Else $k \leftarrow dec(sk_Y, c)$.

vii. $t' \leftarrow tag(k, (g^y, h, Y, s', X, s))$

viii. Output $(Y, s', X, s, g^y, c', t')$ to A and append

$$((message_1, Y, (X, s, h, c)), (Y, s', X, s, g^y, c', t'))$$

to $View$.

(c) If A makes a $(message_2, X, (Y, s', X, s, h', c', t'))$ to honest user $X \in \{U_1, \ldots U_m\}$ then

i. If there is no open session with values (X, s, Y), output $\perp$ to A and append $((message_2, X, (Y, s', X, s, h', c', t')), \perp)$ to $View$.

ii. Else retrieve the symmetric key k and the DH exponent x associated with (X, s, Y).

iii. If $vrf(k, (h', g^x, Y, s', X, s), t') = 0$, output $\perp$ to A and append $((message_2, X, (Y, s', X, s, h', c', t')), \perp)$ to $View$, else continue.

iv. Compute session key $(h')^x$.

v. If c' was previously produced by an honest user using X's public key, retrieve the corresponding k'.

vi. Else $k' \leftarrow dec(sk_X, c')$.

vii. $t \leftarrow tag(k', (g^x, h', X, s, Y, s'))$.

viii. Output (X, s, Y, s', t) to A and append

$$((message_2, X, (X, s, Y, s', h', c', t')), (X, s, Y, s', t))$$

to $View$.

(d) If A makes a $(message_3, Y, (X, s, Y, s', t))$ query to honest user $Y \in \{U_1, \ldots U_m\}$ then

i. If there is no open session with values (Y, s', X, s), output $\perp$ to A and append $((message_3, Y, (X, s, Y, s', t)), \perp)$ to $View$.

ii. Else retrieve the symmetric key k', the DH exponent y, and the peer DH group element h associated to values (Y, s', X, s).

iii. If $vrf(k', (h, g^y, X, s, Y, s'), t) = 0$, output $\perp$ to A and append $((message_3, Y, (X, s, Y, s', t)), \perp)$ to $View$, else continue.

iv. Compute session key h^y.

v. Output $SessionDone$ to A and append

$$((message_3, Y, (X, s, Y, s', t)), SessionDone)$$

to $View$.

(e) If A makes a $(RevealSessionKey, X, (X, s, Y, s'))$ query to honest user $X \in \{U_1, \ldots U_m\}$, then

i. If there is no completed session with values (X, s, Y, s'), then output $\perp$ to A and append $((RevealSessionKey, X, (X, s, Y, s')), \perp)$ to $View$.

ii. Else retrieve the session key $h \in G$ associated to values (X, s, Y, s'), output h to A and append $((RevealSessionKey, X, (X, s, Y, s')), h)$ to $View$.

4. Append any output that A produces to $View$.

5. Output $View$.

Now, we group the instructions above into two new algorithms. Let $\bar{r}$ be the random coins used in generating the session ids, DH exponents, and symmetric keys for lines 3.(a).i-iii, 3.(b).i-iii. Let $ExtInit$ be an extended initialization procedure that draws also the coins $\bar{r}$. Let $\bar{A}$ be an algorithm that takes as input $(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$, and performs the instructions on lines 2,3 (with all sub-items), 4, and finally outputs $View$, with the following modifications:

1. $\bar{A}$ uses the coins $\bar{r}$ to generate the session ids, DH exponents, and symmetric keys for lines 3.(a).i-iii, 3.(b).i-iii.

2. $\bar{A}$ makes queries on a decryption oracle interface for lines 3.(b).vi, 3.(c).vi.

Thus, we have rewritten the experiment

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow [A(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V), CH]$$

as the experiment

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r, \bar{r}) \leftarrow ExtInit(1^n), \\ View \leftarrow \bar{A}^{dec(\vec{sk}_U)}(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$$

Now, we apply the assumption that the encryption scheme is multi-user PA1. We get that there exists $\bar{A}^*$ such that for all efficient D

$$|Pr((\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r, \bar{r}) \leftarrow ExtInit(1^n), \\ View \leftarrow \bar{A}^{dec(\vec{sk}_U)}(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V) : D(View) = 1) \\ - Pr((\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r, \bar{r}) \leftarrow ExtInit(1^n), \\ View \leftarrow \bar{A}^{\bar{A}^*(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)}(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V) : D(View) = 1)|$$

is negligible.

Now we consider the experiment

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r, \bar{r}) \leftarrow ExtInit(1^n), \\ View \leftarrow \bar{A}^{\bar{A}^*(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)}(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$$

and convert it back to line-by-line instructions. The instructions are the same as before, except that in lines 3.(b).vi, 3.(c).vi, queries are made to $\bar{A}^*(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$ instead of to the decryption oracle.

Now, we consider a single algorithm S that takes input $(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$, draws by itself the random coins $\bar{r}$, performs the instructions in lines 2, 3 (with $\bar{A}^*$ instead of dec), 4, and outputs $View$. Thus, we have rewritten the experiment

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r, \bar{r}) \leftarrow ExtInit(1^n),$$
$$View \leftarrow \bar{A}^{\bar{A}^*(r,\bar{r},\vec{pk}_U,\vec{pk}_V,\vec{sk}_V)}(r, \bar{r}, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$$

as the experiment

$$(\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow S(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V)$$

Combining all observations so far, we conclude that for any polynomially bounded m, m', for any efficient A, there exists efficient S, such that for all efficient D,

$$\begin{aligned}|Pr((\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow [A(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V), CH] \\ : D(View) = 1) \\ - Pr((\vec{pk}_U, \vec{sk}_U, \vec{pk}_V, \vec{sk}_V, r) \leftarrow Init(1^n), View \leftarrow S(r, \vec{pk}_U, \vec{pk}_V, \vec{sk}_V) \\ : D(View) = 1)|\end{aligned}$$

is negligible. Therefore, SKEME instantiated with a multi-user PA1 encryption scheme is deniable. □

Acknowledgment

This work was supported by the Luxembourg National Research Fund, under CORE project Q-CoDe (Project ID 11689058).

References

[1] Better protection of whistle-blowers: new eu-wide rules to kick in in 2021. URL: `https://www.consilium.europa.eu/en/press/press-releases/2019/10/07/better-protection-of-whistle-blowers-new-eu-wide-rules-to-kick-in-in-2021/`

[2] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 419–428. ACM, 1998.

[3] Mihir Bellare and Adriana Palacio. Towards plaintext-aware public-key encryption without random oracles. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 48–62. Springer, 2004.

[4] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *Annual International Cryptology Conference*, pages 90–104. Springer, 1997.

[5] Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In *Annual International Cryptology Conference*, pages 445–456. Springer, 1991.

[6] Mario Di Raimondo, Rosario Gennaro, and Hugo Krawczyk. Deniable authentication and key exchange. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 400–409. ACM, 2006.

[7] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *Journal of the ACM (JACM)*, 51(6):851–898, 2004.

[8] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[9] Dan Harkins, Dave Carrel, et al. The internet key exchange (ike). Technical report, RFC 2409, november, 1998.

[10] Charlie Kaufman et al. Internet key exchange (ikev2) protocol. Technical report, RFC 4306, December, 2005.

[11] Hugo Krawczyk. Skeme: A versatile secure key exchange mechanism for internet. In *Proceedings of Internet Society Symposium on Network and Distributed Systems Security*, pages 114–127. IEEE, 1996.