J. Cryptol. (2014) 27: 139–180
DOI: 10.1007/s00145-012-9141-6

Journal of
CRYPTOLOGY

Security Models and Proof Strategies for Plaintext-Aware Encryption

James Birkett
Information Security Group, Royal Holloway, University of London, London, UK

Alexander W. Dent
Information Security Group, Royal Holloway, University of London, London, UK
alexander.dent@gmail.com
and
Qualcomm Research, Qualcomm Technologies Inc., San Diego, USA

Communicated by Phillip Rogaway

Received 1 August 2010
Online publication 9 January 2013

Abstract. Plaintext-aware encryption is a simple concept: a public-key encryption scheme is plaintext aware if no polynomial-time algorithm can create a ciphertext without "knowing" the underlying message. However, the formal definitions of plaintext awareness are complex. This paper analyses these formal security definitions and presents the only known viable strategy for proving a scheme is PA2 plaintext aware. At the heart of this strategy is a new notion called PA1+ plaintext awareness. This security notion conceptually sits between PA1 and PA2 plaintext awareness (although it is formally distinct from either of these notions). We show exactly how this new security notion relates to the existing notions and how it can be used to prove PA2 plaintext awareness.

Key words. Public-key encryption, Provable security, Plaintext-aware encryption, Plaintext awareness.

1. Introduction

The concept of plaintext awareness was initially developed as a tool to aid attempts to prove the security of public-key encryption schemes. The basic idea is that a scheme should be deemed plaintext aware if no polynomial-time algorithm can create a valid ciphertext without "knowing" the underlying plaintext. This mitigates the usefulness of the decryption oracle in the IND-CCA2 security model: an attacker automatically knows the decryption of any ciphertext that attacker could submit to the decryption oracle. Thus we would hope that a scheme which is IND-CPA secure [16] and plaintext aware would be IND-CCA2 secure [23].

Initially, the formal model for plaintext awareness was developed in the mid-1990s by Bellare and Rogaway [3] in an attempt to create a framework to prove the security of

encryption schemes in the random oracle model. Unfortunately, the original formulation only sufficed to prove IND-CCA1 security [21] rather than the preferred IND-CCA2 security [23]. A revised random oracle definition was proposed by Bellare et al. [6] in the late 1990s that did imply IND-CCA2 security, but at the cost of a more complex definition. Moreover, this formulation could still not be easily translated from the random oracle model into the standard model. The cryptographic community had to wait for a decade until a formal model which could be achieved in the standard model was proposed by Bellare and Palacio [2]. The key insight which allowed this new model to be developed was that the security motion could not be formulated in a black-box manner.

The non-black-box security models for plaintext awareness are unusual and independently interesting. This paper highlights the important differences that can be introduced into a model through the decision to use a priori and a posteriori generated random coins. The existing security models for plaintext awareness assume an adversary with a pre-generated random tape—i.e. a priori generated random coins. We propose a novel model for plaintext awareness which uses random values that can be generated by the adversary during their execution—i.e. a posteriori random coins. A key insight into the proof strategies in this work is that an a posteriori choice of randomness provides a stronger and more useful security model. The work on the models for plaintext awareness contained in this model are essentially an investigation in to the relationship between non-black-box models which assume a priori and a posteriori random coins and therefore may be of independent interest.

While earlier work formalised a standard-model interpretation of plaintext awareness, there was no known generic strategy for proving (full) plaintext awareness. This paper provides the first such methodology and uses this methodology to prove that the Cramer–Shoup encryption scheme [10] is plaintext aware under the well-known (but non-falsifiable) Diffie–Hellman Knowledge (DHK) assumption.

Unfortunately, this methodology is not useful in proving IND-CCA2 security, as one of the properties that the encryption scheme is required to fulfil in order to use this methodology implies IND-CCA2 security. We claim that this fact does not seriously undermine the contributions of this paper. We believe that the development of new models for plaintext awareness may be of interest even if no solid methodology exists to prove that a scheme meets the new security definition. Furthermore, we believe that the discovery that an encryption scheme is plaintext aware is of theoretical interest beyond the chore of proving IND-CCA2 security. Plaintext awareness gives an interesting insight into why certain schemes achieve IND-CCA2 security and why certain schemes which would appear to be IND-CCA2 secure have not been proven to achieve that security level.

One of the main challenges of proving the IND-CCA2 security of a public-key encryption scheme is the difficulty in simulating the decryption oracle without full knowledge of the private decryption key. This challenge is made harder by the possibility of trivial decryption oracle queries in which an attacker submits a ciphertext to the oracle for which it already "knows" the underlying message. The simulated decryption oracle must return the correct underlying message, as otherwise the attacker would be able to distinguish the simulated decryption oracle from a real decryption oracle, despite the fact that the query does not actually help the attacker break the security of the underlying scheme. Plaintext awareness can be interpreted as a formalisation of the idea that all

decryption oracle queries that a polynomial-time attacker can generate must be trivial. IND-CCA2 proofs for encryption schemes which are not plaintext aware must either (i) answer decryption queries using private information unknown to the attacker and show that this does not leak "too much" of the private key to the attacker; (ii) show that the scheme is plaintext aware for a large portion of the ciphertext space and that it is unlikely that the attacker will be able to generate a valid ciphertext for which he does not "know" the underlying plaintext; or (iii) show that the ability to decrypt non-trivial ciphertexts does not enable an attacker to determine any information about the challenge ciphertext.

Furthermore, plaintext awareness has been shown to be a useful property in its own right and one that allows the construction of simple provably-secure encryption-based protocols. For example, Di Raimondo et al. use plaintext awareness to analyse the deniability of the SKEME key exchange protocol [13]. Di Raimondo et al.'s work considers protocols which use a message transmission system of the following form:

1. Bob generates a symmetric session key K and encrypts it using Alice's public key pk_A to form a ciphertext $C \xleftarrow{\$} \mathcal{E}(pk_A, K)$. Bob sends this ciphertext to the sender.
2. Alice decrypts the ciphertext to recover the session key K. Alice uses this to encrypt a message m using an authenticated encryption system $\mathcal{AE}_K(m)$ and sends this to Bob.

Loosely speaking, this protocol is (sender) deniable if every possible legitimate protocol transcript between Alice and Bob could have been produced by Bob alone. Bob may try to break deniability by producing the ciphertext C in a non-standard way and showing that Alice must have been involved in decrypting the ciphertext. This proves Alice's involvement in the protocol and removes Alice's deniability. Di Raimondo et al. use plaintext awareness to argue that Bob must know the symmetric key K encrypted by any valid ciphertext C and so Bob can always simulate a protocol transcript. They conclude that protocols of this form, including SKEME, are deniable if they are instantiated using plaintext-aware encryption. A similar argument is made by Ventre and Visconti in the construction of two-round extractable commitment schemes [30].

Models for Plaintext Awareness The concept of plaintext-aware encryption may be simple, but the formal definition is complex and subtle. The problem lies in the difficulty of formally defining what is meant by the phrase 'an attacker knows the underlying plaintext'. The initial definition of plaintext awareness, introduced by Bellare and Rogaway [3] and extended by Bellare, Desai, Pointcheval and Rogaway [6], relies on the random oracle model. A public-key encryption scheme was said to be plaintext aware if it is possible to deduce the decryption of a ciphertext produced by an attacker by observing the random oracle queries made by the attacker during the construction of the ciphertext. A class of public-key encryption schemes, which make use of a hash-based "checksum" value, can easily be proven plaintext aware in this model. However, this definition relies on the random oracle model and has no obvious counterpart in the standard model.

A more general treatment was introduced by Bellare and Palacio [2] who introduced a definition of "knowledge" similar to the definition which appears in the analysis of zero-knowledge protocols [17]. For a zero-knowledge protocol, an algorithm $\mathcal{A}$ is deemed

to "know" a value if it could be altered to give an algorithm $\mathcal{A}^*$ of similar complexity which could output that value. For example, assuming the hardness of discrete logarithm problem, a polynomial-time algorithm which is given a group generator g and a randomly chosen element g^x does not know x, since no polynomial-time algorithm could compute x, but would know g^{x+1} since we could alter $\mathcal{A}$ to give an algorithm $\mathcal{A}^*(g, g^x)$ which outputs g^{x+1}. In the context of plaintext-aware encryption, Bellare and Palacio proposed that an encryption scheme be deemed plaintext-aware if (roughly) for any polynomial-time algorithm $\mathcal{A}$ which can output ciphertexts, there exists a polynomial-time algorithm $\mathcal{A}^*$ which will output the underlying messages of those ciphertexts given the same inputs as original algorithm. The algorithm $\mathcal{A}$ is known as the "ciphertext creator" and the algorithm $\mathcal{A}^*$ is known as the "plaintext extractor" for obvious reasons. It should be noted that the phrase "given the same inputs" is a very strong requirement: crucially, the plaintext extractor $\mathcal{A}^*$ should even be given the random coins of the ciphertext creator $\mathcal{A}$.

From a model point of view, the approach proposed by Bellare and Palacio is very appealing. The plaintext extractor $\mathcal{A}^*$ can be thought of as a polynomial-time "spy" which sits on the shoulder of an IND-CCA2 attacker $\mathcal{A}$ as it creates a ciphertext, deduces the underlying message, and feeds this information back to the attacker when it makes a decryption oracle query.

This model for plaintext awareness is known as PA1 plaintext awareness. Unfortunately, PA1 plaintext awareness is not sufficient to allow us to prove the IND-CCA2 security of an IND-CPA secure encryption scheme. This is because the IND-CCA2 game allows the attacker to make use of an extra piece of information to help them create a ciphertext: the challenge ciphertext itself. It may be possible for an attacker to use the challenge ciphertext to create a ciphertext which the plaintext extractor cannot successfully decrypt. In order to use the concept of plaintext awareness to prove IND-CCA2 security, Bellare and Palacio allowed the ciphertext creator $\mathcal{A}$ to query an encryption oracle which would return the encryption of a randomly chosen message output by a polynomial-time "plaintext creator" algorithm $\mathcal{P}$. A scheme which remains plaintext aware in the presence of any plaintext creator is deemed to be PA2 plaintext aware, and a scheme which is both PA2 plaintext aware and IND-CPA secure is necessarily IND-CCA2 secure.

Our Contribution[1] The core of our approach to plaintext-aware public-key encryption is the concept of PA1+ plaintext awareness. One major difference between PA1 and PA2 plaintext awareness is that, in the PA1 model of plaintext awareness, the plaintext extractor can determine every action that the ciphertext creator will take during its entire execution, since the plaintext extractor knows all of the inputs that the ciphertext creator will ever receive. In the definition of PA2 plaintext awareness, the plaintext extractor is only able to determine the actions of the ciphertext creator up to the point at which it queries the encryption oracle, since it cannot a priori know the ciphertext that oracle will return.

This means, for a suitably random encryption scheme, the encryption oracle can actually be used to provide two services: it gives the ciphertext creator access to ciphertexts and it gives the ciphertext creator access to a source of new random bits. PA1+ plain-

[1] This paper presents, extends, and corrects errors in [12] and [8].

text awareness is similar to PA1 plaintext awareness except that the plaintext creator is also given access to a source of new random bits. Conceptually, the PA1+ model sits between the PA1 and PA2 models: it gives the ciphertext creator access to a source of randomness, but does not give the ciphertext creator access to a full encryption oracle.

We show how this new notion of PA1+ plaintext awareness relates to Bellare and Palacio's existing notions of PA1 and PA2 plaintext awareness; in particular, we show that it is a weaker notion than PA2 plaintext awareness in the sense that any public-key encryption scheme which is PA2 plaintext aware and IND-CPA secure is necessarily PA1+ plaintext aware. We also show that a scheme which is PA1+ plaintext aware and which presents ciphertexts which appear essentially random is necessarily PA2 plaintext aware. This gives the only known viable strategy for proving PA2 plaintext awareness in the standard model. We demonstrate the usefulness of this strategy by proving that the Cramer–Shoup public-key encryption scheme is PA2 plaintext aware [10].

Related Work Obviously, the work in this paper builds heavily on the standard-model definitions presented by Bellare and Palacio [2]. Bellare and Palacio also proved the fundamental result that PA2 plaintext awareness and IND-CPA security implies IND-CCA2 security. This result was improved by Teranishi and Ogata [29] who proved that PA2 plaintext awareness and one-way security implies IND-CCA2 security. Jiang and Wang investigated notions of the plaintext awareness of hybrid (KEM-DEM) encryption schemes [19] and used the proof strategy in this paper to prove that the hash-proof-based public-key encryption scheme of Kurosawa and Desmedt [20] is PA2 plaintext aware. This latter result was independently obtained by Birkett [7] who also proved that the hash-proof-based public-key encryption scheme of Cramer and Shoup [10] is PA2 plaintext aware.

2. Preliminaries

We let $\leftarrow$ and $\xleftarrow{\$}$ denote assignment. If $\mathcal{A}$ is a deterministic algorithm, then $y \leftarrow \mathcal{A}(x)$ denotes the assignment to y of the output of running $\mathcal{A}$ on x. If $\mathcal{A}$ is a probabilistic algorithm, then $y \xleftarrow{\$} \mathcal{A}(x)$ denotes the assignment to y of the output of running $\mathcal{A}$ on x with a fresh set of random coins. We write $y \leftarrow \mathcal{A}(x; R)$ to denote the assignment to y of the output of running $\mathcal{A}$ on x with the random coins R and we let $R[\mathcal{A}]$ denote the random coins of $\mathcal{A}$. An algorithm $\mathcal{A}$ is (strict) polynomial-time if there exists a polynomial p such that the running time of $\mathcal{A}(x)$ is bounded by $p(|x|)$. A function $f(k, x)$ is negligible in k if $f(k, x) \in O(k^{-n})$ for all $x \in \{0, 1\}^*$ and $n \in \mathbb{N}$.

If S is a finite set, then $y \xleftarrow{\$} S$ denotes the assignment to y of a randomly chosen element of S. If X is a distribution over a finite set S, then $y \xleftarrow{\$} X$ denotes the assignment to y of an element of S chosen according to the distribution X. If X and Y are distributions over some common finite set S, then the statistical distance between X and Y is

$$\Delta[X, Y] = \frac{1}{2} \sum_{s \in S} \big|\Pr[X = s] - \Pr[Y = s]\big|.$$

The output of a probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$ with input x and access to an oracle $\mathcal{O}$ is written $\mathcal{A}^{\mathcal{O}}(x)$. If we are describing an algorithm $\mathcal{A}$ then we

will use the phrase "Query $\beta \leftarrow \mathcal{O}(\alpha)$" to denote that $\mathcal{A}$ queries the oracle $\mathcal{O}$ on α and receives the response β. We will frequently encounter a situation where an algorithm $\mathcal{A}^{\mathcal{O}}(x)$ will be running an algorithm $\mathcal{B}^{\mathcal{O}'}(y)$ as a subroutine. In this situation $\mathcal{A}$ will be expected to simulate the oracle $\mathcal{O}'$ for $\mathcal{B}$. In order to describe this, we will use a pseudocode description of the form

$z \xleftarrow{\$} \mathcal{B}^{\mathcal{O}'}(y)$
  If $\mathcal{B}$ queries $\mathcal{O}'(\alpha)$
    Perform calculations
    Return β

to denote that $\mathcal{B}$ is run on y, may query the "oracle" $\mathcal{O}'$ with an input α and will receive a response β in return, and finally outputs z. We also make use of pseudocode of the form

Repeat (k times) until E
  Perform first calculations
Else
  Perform second calculations

to mean that the set of first calculations is repeated until either E occurs or the loop has been executed k times. If E has not occurred after k executions of the loop, then the second set of calculations is performed.

We refer to a list of bitstrings using the form ALIST for some identifier A. All lists are initially initialised to be empty (ALIST $\leftarrow \varepsilon$). Elements of the list are addressed as ALIST[i] with the first entry addressed as ALIST[0]; the number of the entries in the list is written as |ALIST|.

A public-key encryption scheme is a triple of PPT algorithms $(\mathcal{G}, \mathcal{E}, \mathcal{D})$. The key generation algorithm takes as input a security parameter 1^k and outputs a public/private key pair $(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$. The public key implicity defines the message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$. The encryption algorithm takes a message $m \in \mathcal{M}$ and a public key pk as input, and outputs a ciphertext $C \xleftarrow{\$} \mathcal{E}(pk, m)$ in the ciphertext space $\mathcal{C}$. The (deterministic) decryption algorithm takes a ciphertext $C \in \mathcal{C}$ and a private key sk as input, and output either a message $m \leftarrow \mathcal{D}(sk, C)$ in the message space $\mathcal{M}$ or the distinguished error symbol $\perp$. For correctness, we require that for all $(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$ and messages $m \in \mathcal{M}$, we have that $\mathcal{D}(sk, \mathcal{E}(pk, m)) = m$ with probability 1.

We define five notions of security for a public-key encryption scheme: IND-ATK, LH-IND-ATK, and OW-CPA for ATK $\in$ {CPA, CCA2} [16,23]. These security notions are defined via the experiments in Fig. 1. For the IND/LH-IND security notions, a PPT attacker $\mathcal{A}$ is deemed to have advantage

$$Adv_{\mathcal{A}}^{\text{(LH-) IND}}(k) = \left|\Pr\left[\text{EXPT}_{\mathcal{A}}^{\text{(LH-) IND-1}} = 1\right] - \Pr\left[\text{EXPT}_{\mathcal{A}}^{\text{(LH-) IND-0}} = 1\right]\right|.$$

For the OW security notion, a PPT attacker $\mathcal{A}$ is deemed to have advantage

$$Adv_{\mathcal{A}}^{\text{OW}}(k) = \Pr\left[\text{EXPT}_{\mathcal{A}}^{\text{OW}} = 1\right].$$

A public-key encryption scheme is OW/IND/LH-IND secure if the corresponding advantage is negligible. We will often prove results about the security of encryption

$\text{EXPT}^{\text{IND-}b}_{\mathcal{A}}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$(m_0, m_1, \omega) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathcal{D}}}(1^k, pk)$
$C^* \xleftarrow{\$} \mathcal{E}(pk, m_b)$
If $|m_0| \neq |m_1|$ then $C^* \leftarrow \perp$
$b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_{\mathcal{D}}}(C^*, \omega)$
If $b' = 1$ then output 1
Else output 0

$\text{EXPT}^{\text{LH-IND-}b}_{\mathcal{A}}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$(m_0, m_1, \omega) \xleftarrow{\$} \mathcal{A}_1^{\mathcal{O}_{\mathcal{D}}}(1^k, pk)$
$C^* \xleftarrow{\$} \mathcal{E}(pk, m_b)$
$b' \xleftarrow{\$} \mathcal{A}_2^{\mathcal{O}_{\mathcal{D}}}(C^*, \omega)$
If $b' = 1$ then output 1
Else output 0

$\text{EXPT}^{\text{OW}}_{\mathcal{A}}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$m^* \xleftarrow{\$} \mathcal{M}$
$C^* \xleftarrow{\$} \mathcal{E}(pk, m^*)$
$m \xleftarrow{\$} \mathcal{A}(1^k, pk, C^*)$
If $m = m^*$ then output 1
Else output 0

Fig. 1. Security notions for public-key encryption: IND (*left*), LH-IND (*centre*), and OW (*right*). In the IND and LH-IND security games, the oracle $\mathcal{O}_{\mathcal{D}}$ depends on ATK. If ATK = CPA then the decryption oracle $\mathcal{O}_{\mathcal{D}}$ returns the empty string ε on all inputs. If ATK = CCA2 then the decryption oracle $\mathcal{O}_{\mathcal{D}}$ returns $\mathcal{D}(sk, C)$ on input C. The attacker $\mathcal{A}_2$ is forbidden from querying the decryption oracle $\mathcal{O}_{\mathcal{D}}$ on C^*.

schemes using game-hopping techniques [4,26] for which a short introduction is given in Appendix A.

The LH-IND (length-hiding indistinguishable) security notion perhaps merits a few words of explanation. A scheme is deemed to be IND secure if a ciphertext hides all partial information about a message except perhaps its length. A scheme is deemed to be LH-IND secure if hides all partial information about a ciphertext *including* its length. Clearly, any scheme which has $\mathcal{M} = \{0,1\}^\ell$ is LH-IND-ATK secure if and only if it is IND-ATK secure. Equally clearly, any scheme which has $\mathcal{M} = \{0,1\}^*$ is not LH-IND-ATK secure even if it is IND-ATK secure (as one can break LH-IND-CPA security by setting m_0 to be a very short message and m_1 to be a very long message and observing the difference in ciphertext length).

Lastly, we will occasionally write

$$\Pr[\textit{condition} : \textit{experiment}]$$

to denote the probability that *condition* is true after executing the steps described by *experiment*.

3. Relations Between Notions of Plaintext Awareness

3.1. *The Bellare–Palacio Notions of Plaintext Awareness*

The standard-model definitions of plaintext awareness were provided by Bellare and Palacio [2]. As we have already discussed, the main idea behind the Bellare–Palacio definitions are the concepts of a ciphertext creator $\mathcal{A}$ and a plaintext extractor $\mathcal{A}^*$. The ciphertext creator will be able to submit ciphertexts to a "decryption oracle". This oracle will either be a real decryption oracle or the plaintext extractor attempting to play the part of the real decryption oracle. The plaintext extractor is given all of the inputs of the ciphertext creator, including the ciphertext creator's random coins. Thus, the plaintext extractor can be thought of as a modified version of the ciphertext creator which outputs both ciphertexts and their underlying messages, thus realising the idea that the ciphertext creator must "know" the underlying message in the sense introduced by zero-knowledge protocols.

$\text{EXPT}^{\text{REAL-PA1}}_{\mathcal{A},D}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}(C)$:
Return $\mathcal{D}(sk, C)$
$b \xleftarrow{\$} D(x)$
Output b

$\text{EXPT}^{\text{FAKE-PA1}}_{\mathcal{A},\mathcal{A}^*,D}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}(C)$:
Return $\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}])$
$b \xleftarrow{\$} D(x)$
Output b

Fig. 2. The PA1 plaintext awareness security games. We assume that $\mathcal{A}^*$ is a stateful algorithm (that retains its internal state between invocations) and recall that $R[\mathcal{A}]$ represents the random coins of $\mathcal{A}$.

The following two definitions are taken from the work of Bellare and Palacio [2]. We define plaintext awareness using two games in Fig. 2 which are played by a ciphertext creator $\mathcal{A}$, a plaintext extractor $\mathcal{A}^*$ and a distinguisher algorithm D. The advantage is defined to be

$$Advv^{\text{PA1}}_{\mathcal{A},\mathcal{A}^*,D}(k) = \left|\Pr\left[\text{EXPT}^{\text{REAL-PA1}}_{\mathcal{A},D} = 1\right] - \Pr\left[\text{EXPT}^{\text{FAKE-PA1}}_{\mathcal{A},\mathcal{A}^*,D} = 1\right]\right|.$$

Definition 3.1 (PA1). A public-key encryption scheme is PA1 plaintext aware if for every PPT ciphertext creator $\mathcal{A}$ there exists a PPT plaintext extractor $\mathcal{A}^*$ such that for all polynomial-time distinguisher algorithms D we have that $Advv^{\text{PA1}}_{\mathcal{A},\mathcal{A}^*,D}(k)$ is negligible.

As we mentioned previously, PA1 plaintext awareness is not sufficient to prove IND-CCA2 security, since the model for PA1 plaintext awareness does not capture situations in which the ciphertext creator may be able to obtain ciphertexts for which they do not know the underlying message. This motivates the definition of PA2 plaintext awareness, which is defined by the two games in Fig. 3. The central difference between PA1 and PA2 plaintext awareness is the introduction of the encryption oracle $\mathcal{O}_{\mathcal{E}}$. This oracle gives the ciphertext creator access to a source of ciphertexts by returning the encryption of a message output by a stateful, PPT plaintext creator algorithm $\mathcal{P}$. This architecture is shown in Fig. 4. The advantage of a ciphertext creator $\mathcal{A}$, plaintext extractor $\mathcal{A}^*$, plaintext creator $\mathcal{P}$, and a distinguisher algorithm D is given by

$$Advv^{\text{PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}(k) = \left|\Pr\left[\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\mathcal{P},D} = 1\right] - \Pr\left[\text{EXPT}^{\text{FAKE-PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D} = 1\right]\right|.$$

Definition 3.2 (PA2). A public-key encryption scheme is PA2 plaintext aware if for every PPT ciphertext creator $\mathcal{A}$ there exists a PPT plaintext extractor $\mathcal{A}^*$ such that for all PPT plaintext creators $\mathcal{P}$ and PPT distinguisher algorithms D we have that $Advv^{\text{PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}(k)$ is negligible.

Theorem 3.3 (Bellare–Palacio). *A public-key encryption scheme that is (LH-)IND-CPA secure and PA2 plaintext aware is (LH-)IND-CCA2 secure.*

Theorem 3.4 (Teranishi–Ogata). *A public-key encryption scheme that is OW-CPA secure and PA2 plaintext aware is (LH-)IND-CPA secure.*

$\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\mathcal{P},D}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}},\mathcal{O}_{\mathcal{E}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(\alpha)$:
$m \xleftarrow{\$} \mathcal{P}(\alpha)$
$C \xleftarrow{\$} \mathcal{E}(pk, m)$
Append C to CLIST
Return C
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}$ on C:
If $C \in$ CLIST then return $\perp$
Else return $\mathcal{D}(sk, C)$
$b \xleftarrow{\$} D(x)$
Output b

$\text{EXPT}^{\text{FAKE-PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}},\mathcal{O}_{\mathcal{E}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(\alpha)$:
$m \xleftarrow{\$} \mathcal{P}(\alpha)$
$C \xleftarrow{\$} \mathcal{E}(pk, m)$
Append C to CLIST
Return C
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}$ on C:
If $C \in$ CLIST then return $\perp$
Else return $\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}],$ CLIST)
$b \xleftarrow{\$} D(x)$
Output b

Fig. 3. The PA2 plaintext awareness security games. We assume that $\mathcal{A}^*$ and $\mathcal{P}$ are stateful algorithms (that retain their internal state between invocations).

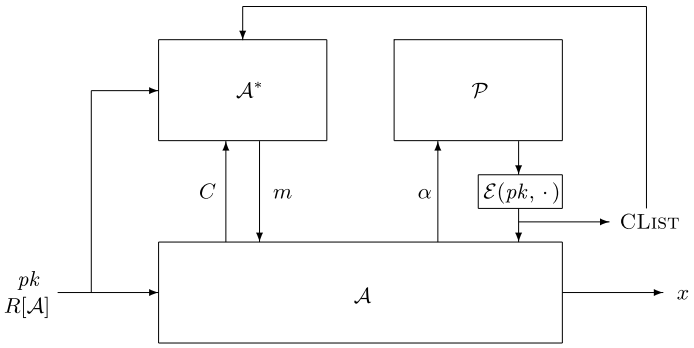

Fig. 4. The $\text{EXPT}^{\text{FAKE-PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}$ model.

Technically, both of these results are only proven for the IND security notions, but an examination of the proofs shows that neither relies on the fact that $|m_0| = |m_1|$ in the IND security game. Thus, the proofs hold for the LH-IND security notions as well. This allows us to prove the following result (which motivates our consideration of LH-IND security at all):

Corollary 3.5. *A public-key encryption scheme that is IND-CPA secure, PA2 plaintext aware, and has the property that $|\mathcal{M}|$ grows faster than any polynomial in the security parameter k is LH-IND-CPA secure.*

Proof. An encryption scheme which is IND-CPA scheme and which has the property that $|\mathcal{M}|$ grows faster than any polynomial is OW-CPA secure. Thus, by the Teranishi–Ogata result, the scheme is LH-IND-CPA secure. □

Conceptually, this minor observation proves that we should only consider schemes which are LH-IND-CPA secure as candidates for PA2 plaintext awareness. Practically, this means that no scheme with $\mathcal{M} = \{0,1\}^*$ can ever be PA2 plaintext aware, which eliminates most hybrid schemes. Interestingly, we will show that the Cramer–Shoup hybrid encryption scheme is PA2 plaintext aware *if* we restrict the message space to $\mathcal{M} = \{0,1\}^k$ (and so, with the use of a suitable padding scheme, if $\mathcal{M} = \{0,1\}^{\leq k}$).

3.2. *PA1+ Plaintext Awareness*

The crux of our approach to plaintext awareness is a new definition which we term PA1+ plaintext awareness. We claim that it lies conceptually between PA1 and PA2 plaintext awareness, although it is formally distinct from either of them. In the definition of PA1 plaintext awareness, when the ciphertext creator $\mathcal{A}$ first invokes the plaintext extractor $\mathcal{A}^*$, the plaintext extractor can determine every action that the ciphertext creator will take during its entire execution, since the ciphertext creator is given no independently generated input during its execution.

In the PA2 model for plaintext awareness, the ciphertext creator can use the encryption oracle for two purposes:

- It can use the encryption oracle to generate encryptions of messages drawn from polynomial-time distributions.
- For suitably randomised encryption schemes, the encryption oracle can be used as a source of random bits (e.g. by hashing the ciphertext).

The ability to obtain "new" random bits means that the plaintext extractor cannot determine the ciphertext creator's complete execution in advance. This is a significant difference between PA1 and PA2 plaintext awareness. PA1+ plaintext awareness explicitly gives the ciphertext creator one of the two abilities that it gets in the PA2 model: the ability to access a source of new random bits. In order to allow the plaintext extractor $\mathcal{A}^*$ to track the execution of the ciphertext creator $\mathcal{A}$, we give the plaintext extractor access to all the random bits that have been generated up to the point that it is queried.

Definition 3.6 (PA1+). For any definition of plaintext awareness PA $\in$ {PA1/PA2}, we give a new definition PA+ in which the ciphertext creator $\mathcal{A}$ is given access to a randomness oracle $\mathcal{R}$, which takes no input and returns a randomly-chosen bit. The plaintext extractor $\mathcal{A}^*$ is altered so that it takes a list RLIST of the random bits returned by the randomness oracle as an additional input, i.e. the plaintext extractor is run as $\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$.

Another interpretation of the difference between the PA and PA+ definitions relates to the way one defines a probabilistic Turing machine. If one defines a probabilistic Turing machine as a Turing machine with access to infinite random tape, then one automatically derives the PA notions of plaintext awareness. If one defines a probabilistic Turing machine as a Turing machine with access to an oracle which will return a randomly chosen bit, then one derives the PA+ notions of plaintext awareness. It is interesting that these two definitions of probabilistic Turing machine, which are usually equivalent, lead to different definitions of plaintext awareness.

Interpreting the Difference Between PA1 and PA1+ The difference between PA1 and PA1+ plaintext awareness seem to be fairly minor—PA1 provides the plaintext extractor all the ciphertext creator's random coins at the start of the game whereas PA1+ provides the plaintext extractor the ciphertext creator's random coins as they are used.

In order to highlight the conceptual differences between a priori and a posteriori randomness in the PA1 and PA1+ definitions, we consider an analogous situation that may occur in zero-knowledge protocols. Consider the standard "cut-and-choose" protocol for proving that an element $x^2 = y \in \mathbb{Z}_N^*$ is a quadratic residue [14]:

- A prover P presents the verifier with ℓ quadratic residues $(R_1, R_2, \ldots, R_\ell)$ in $\mathbb{Z}_N^*$ where $R_i = r_i^2$.
- The verifier V sends the prover ℓ random bits $(b_1, b_2, \ldots, b_\ell)$.
- The prover returns the square roots $(z_1, z_2, \ldots, z_\ell)$ where $z_i = r_i \cdot x^{b_i}$ in $\mathbb{Z}_N^*$.
- The verifier checks that $z_i^2 = R_i \cdot y^{b_i}$ for each $1 \leq i \leq \ell$ and accepts the proof if all checks pass.

Now consider a simulator S which wishes to fool the verifier V into accepting a non-residue by observing V's randomness. (In our analogy, the verifier V is analogous to the ciphertext creator $\mathcal{A}$ and the simulator S is analogous to the plaintext extractor $\mathcal{A}^*$.) If the simulator S is working in a model in which the verifier's randomness is an a priori fixed random tape, i.e. using a security definition similar to PA1 plaintext awareness, then it is simple for the simulator to fool the verifier by choosing the values of R_i so that the it can compute the square roots of $R_i \cdot y^{b_i}$. However, if the verifier has the ability to obtain new random bits during its execution, i.e. using a security definition similar to PA1+ plaintext awareness, then the simulator cannot fool the verifier as it does not know the bits $(b_1, b_2, \ldots, b_\ell)$ when it has to commit to the values of $(R_1, R_2, \ldots, R_\ell)$.

The PA1+ definition prevents the plaintext extractor $\mathcal{A}^*$ from returning values that are based on the ciphertext creator $\mathcal{A}$'s future actions (which prevents the use of certain proof techniques such as rewinding). This makes the PA1+ definition harder to achieve.

3.3. *Simplifying PA2 Plaintext Awareness*

The full definition for PA2 plaintext awareness models a very general scenario: the plaintext creator can be used to represent any number of (stateful) communicating users which wish to generate any number of ciphertexts using any polynomial-time distribution. This definition appears stronger than is required to prove Theorem 3.3, which only requires that we consider two plaintext creators, $\mathcal{P}_0$ and $\mathcal{P}_1$, defined as follows:[2]

[2] The original version of this result [8] claimed that plaintext awareness with respect to a single plaintext creator $\mathcal{P}_I$, which chooses a random bit b and then consistently outputs m_b when given $\alpha = (m_0, m_1)$, was sufficient to prove IND-CCA2 security. We called this property PA2I plaintext awareness. Sadly, this result is not correct. We will sketch a counter-example here. Suppose $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ is IND-CCA2 secure and PA2I plaintext aware. Let $\Pi' = (\mathcal{G}, \mathcal{E}', \mathcal{D}')$ be the encryption scheme with $\mathcal{E}'(pk, m) = 0\|\mathcal{E}(pk, m)$ and $\mathcal{D}'(sk, \delta\|C) = \mathcal{D}(sk, C)$ for any $\delta \in \{0, 1\}$. This scheme is IND-CPA secure but not IND-CCA2 secure. We claim that this scheme is still PA2I plaintext aware. We may build a plaintext extractor $\mathcal{A}^*$ for a ciphertext creator $\mathcal{A}$ working against Π' by noting two properties of the scheme. (a) Since Π is PA2I there exists a plaintext extractor which can decrypt all decryption queries of the form $\delta\|C$ where $0\|C \notin$ CLIST. (b) If $\mathcal{A}$ requests the decryption of $1\|C$ where $0\|C \in$ CLIST then the correct response must be either m_0 or m_1, which can be determined by observing the original encryption oracle query. Furthermore, since Π is IND-

$\mathcal{P}_0(\alpha)$:
  Parse α as (m_0, m_1)
    If parsing fails, return 0
    Else return m_0

$\mathcal{P}_1(\alpha)$:
  Parse α as (m_0, m_1)
    If parsing fails, return 0
    Else return m_1

We will show that, under certain conditions, plaintext awareness with respect to these two plaintext creators is equivalent to the general PA2 plaintext awareness.

Definition 3.7 (2PA2). A public-key encryption scheme is 2PA2 plaintext aware if for every PPT ciphertext creator $\mathcal{A}$ there exists a PPT plaintext extractor $\mathcal{A}^*$ such that for the plaintext creators $\mathcal{P} \in \{\mathcal{P}_0, \mathcal{P}_1\}$ and PPT distinguisher algorithms D we have that $Adv^{\text{PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},\text{D}}(k)$ is negligible.

We may now begin to relate 2PA2 and PA2 plaintext awareness for LH-IND-CPA and LH-IND-CCA2 schemes. We aim to show that a scheme which is LH-IND-CPA and 2PA2 plaintext aware is PA2. We will do this by showing a stronger theorem; we show that for a scheme which is LH-IND-CCA2 (e.g. as it is IND-CPA and 2PA2 plaintext aware) we have that a scheme that is plaintext aware with respect to any fixed plaintext creator (e.g. the $\mathcal{P}_0$ plaintext creator in the 2PA2 definition) is plaintext aware with respect to all plaintext creators (i.e. PA2 plaintext aware).

Theorem 3.8. *Let $\hat{\mathcal{P}}$ be some fixed, stateless polynomial-time plaintext creator (i.e. $\hat{\mathcal{P}}$ does not pass state between invocations). If a public-key encryption scheme Π is LH-IND-CCA2 secure and PA2 plaintext aware* with respect to the plaintext creator $\hat{\mathcal{P}}$, *then Π is PA2 plaintext aware.*

Proof. Let $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ and $\mathcal{A}$ be any PA2 ciphertext creator. Then, since Π is PA2 plaintext aware for the plaintext creator $\hat{\mathcal{P}}$, there exists a plaintext extractor $\mathcal{A}^*$ which "simulates" the decryption oracle in $\text{EXPT}^{\text{FAKE-PA2}}_{\mathcal{A},\mathcal{A}^*,\hat{\mathcal{P}},D}$. We show that $\mathcal{A}^*$ "simulates" the decryption oracle for any plaintext creator $\mathcal{P}$.

Suppose $\mathcal{A}$ makes at most n queries to the plaintext creator oracle. Consider $\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\mathcal{P},D}$ instantiated with an arbitrary PPT plaintext creator $\mathcal{P}$. Define the games $\text{EXPT}^{\text{REAL-}i}_{\mathcal{A},\mathcal{P},D}$ to be identical to $\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\mathcal{P},D}$ except that the final i plaintext creator oracle queries are answered using the plaintext creator $\hat{\mathcal{P}}$. Hence, $\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\mathcal{P},D} = \text{EXPT}^{\text{REAL-0}}_{\mathcal{A},\mathcal{P},D}$ and $\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\hat{\mathcal{P}},D} = \text{EXPT}^{\text{REAL-}n}_{\mathcal{A},\mathcal{P},D}$. Suppose that

$$\left|\Pr\left[\text{EXPT}^{\text{REAL-0}}_{\mathcal{A},\mathcal{P},D} = 1\right] - \Pr\left[\text{EXPT}^{\text{REAL-}n}_{\mathcal{A},\mathcal{P},D}\right]\right|$$

is non-negligible. Then, by a hybrid argument, there exists a sequence of values i_k such that

$$\left|\Pr\left[\text{EXPT}^{\text{REAL-}i_k}_{\mathcal{A},\mathcal{P},D} = 1\right] - \Pr\left[\text{EXPT}^{\text{REAL-}(i_k+1)}_{\mathcal{A},\mathcal{P},D}\right]\right|$$

CCA2 secure, no algorithm can distinguish between the case where $\mathcal{A}^*$ returns the correct decryption of $1\|C$ and the case where $\mathcal{A}^*$ returns the message m_d (for some value d which was randomly chosen by $\mathcal{A}^*$ during its first execution). Thus Π' is IND-CPA secure and PA2I plaintext aware, but not IND-CCA2 secure, which contradicts the claimed result of [8].

is non-negligible. We use this to build an LH-IND-CCA2 attacker $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ as follows:

$\mathcal{B}_1^{\mathcal{D}}(1^k, pk)$:
Pick $j \xleftarrow{\$} \{1, \ldots, n\}$
$ctr \leftarrow 0$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}}, \mathcal{O}_{\mathcal{E}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}(C)$
Query $m \leftarrow \mathcal{D}(C)$
Return m
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(\alpha)$
$ctr \leftarrow ctr + 1$
If $ctr \neq j$ then return $\mathcal{E}(pk, \mathcal{P}(\alpha))$
If $ctr = j$ then $m_0 \xleftarrow{\$} \mathcal{P}(\alpha)$,
$m_1 \xleftarrow{\$} \hat{\mathcal{P}}(\alpha)$
Pause $\mathcal{A}$ and output (m_0, m_1)

$\mathcal{B}_2^{\mathcal{D}}(C^*)$:
Resume $x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}}, \mathcal{O}_{\mathcal{E}}}(1^k, pk)$
by returning C^*
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}(C)$
Query $m \leftarrow \mathcal{D}(C)$
Return m
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(\alpha)$
Return $\mathcal{E}(pk, \hat{\mathcal{P}}(\alpha))$
Output $D(x)$

Thus,

$$Adv_{\mathrm{B}}^{\text{LH-IND}}(k) = \left|\Pr\left[\text{EXPT}_{\mathcal{B}}^{\text{LH-IND-1}} = 1\right] - \Pr\left[\text{EXPT}_{\mathcal{B}}^{\text{LH-IND-0}} = 1\right]\right|$$

$$\geq \frac{1}{n}\left|\Pr\left[\text{EXPT}_{\mathcal{A},\mathcal{P},D}^{\text{REAL-}i_k} = 1\right] - \Pr\left[\text{EXPT}_{\mathcal{A},\mathcal{P},D}^{\text{REAL-}(i_k+1)}\right]\right|.$$

The fact that $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is LH-IND-CCA2 secure means that $Adv_{\mathcal{B}}^{\text{LH-IND}}(k)$ is negligible, which is a contradiction. A similar argument (based on LH-IND-CPA security) shows that

$$\left|\Pr\left[\text{EXPT}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}^{\text{FAKE-PA2}} = 1\right] - \Pr\left[\text{EXPT}_{\mathcal{A},\mathcal{A}^*,\hat{\mathcal{P}},D}^{\text{FAKE-PA2}} = 1\right]\right|$$

is negligible. However, since Π is 2PA2 plaintext aware, we have that

$$\left|\Pr\left[\text{EXPT}_{\mathcal{A},\hat{\mathcal{P}},D}^{\text{REAL-PA2}} = 1\right] - \Pr\left[\text{EXPT}_{\mathcal{A},\mathcal{A}^*,\hat{\mathcal{P}},D}^{\text{FAKE-PA2}} = 1\right]\right|$$

is negligible. Thus, we conclude that

$$\left|\Pr\left[\text{EXPT}_{\mathcal{A},\mathcal{P},D}^{\text{REAL-PA2}} = 1\right] - \Pr\left[\text{EXPT}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}^{\text{FAKE-PA2}} = 1\right]\right|$$

is negligible, which means that Π is PA2 plaintext aware. □

Corollary 3.9. *A public-key encryption scheme Π which is LH-IND-CPA secure and 2PA2 plaintext aware is PA2 plaintext aware.*

Proof. If Π is LH-IND-CPA secure and 2PA2 plaintext aware, then Π is LH-IND-CCA2 secure by Theorem 3.3. And so, by Theorem 3.8, it is PA2 plaintext aware. □

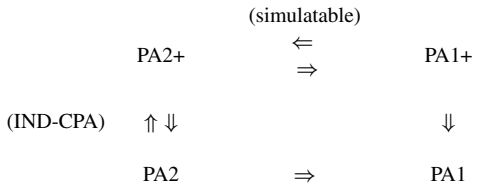

Fig. 5. The relationship between notions of plaintext awareness (including the result of Theorem 5.1).

3.4. *Relations between Notions of Plaintext Awareness*

It is trivial to see that PA2+ $\Rightarrow$ PA1+ $\Rightarrow$ PA1 and PA2+ $\Rightarrow$ PA2 $\Rightarrow$ PA1 (where X $\Rightarrow$ Y means that any public-key encryption scheme which satisfies notion X must also satisfy notion Y). We would hope that, for sufficiently random encryption schemes, we also have that PA2 $\Rightarrow$ PA2+, and this does indeed turn out to be the case. We will also see that, for encryption schemes whose ciphertexts resemble strings of random bits, PA1+ $\Rightarrow$ PA2+. The relationship between the notions of plaintext awareness is shown in Fig. 5.

In this section, we will prove that schemes which are IND-CPA and PA2 plaintext aware are necessarily PA2+ plaintext aware (and therefore trivially PA1+ plaintext aware). Our proof uses a universal hash family [9] to extract the randomness inherent in the ciphertext produced by an IND-CPA encryption scheme.

Definition 3.10 (Universal Hash Family). A family (H, K, A, B) of functions $(H_k)_{k\in K}$, where $H_k : A \to B$ for all $k \in K$, is universal if for all $x, y \in A$ satisfying $x \neq y$ we have

$$\Pr\left[H_k(x) = H_k(y) : k \xleftarrow{\$} K\right] \leq 1/|B|.$$

Definition 3.11 (Collision Probability). For a random variable x which takes values on a set X, the collision probability is defined to be

$$\kappa(x) = \sum_{y\in X} \Pr[x = y]^2.$$

Lemma 3.12. *The following results will be useful in later proofs*:

1. *If x is a random variable on a set X, then* $\max_{y\in X} \Pr[x = y] \leq \sqrt{\kappa(x)}$.
2. *Let $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ be a public-key encryption scheme and M be a deterministic polynomial-time algorithm which outputs a message $M(pk) \in \mathcal{M}$. If x is the random variable on $\mathcal{C}$ distributed according to $\mathcal{E}(pk, M(pk))$, then there exists an IND-CPA adversary $\mathcal{B}$ such that $\kappa(x) = Adv^{\text{IND}}_{\mathcal{B}}(k)$.*

The first part follows trivially from the definition of collision probability. The second part can easily be proven by noting that if $C_1, C_2 \xleftarrow{\$} \mathcal{E}(pk, M(pk))$ then the probability that $C_1 = C_2$ is $\kappa(x)$. The IND-CPA attacker $\mathcal{B}$ is constructed by setting $\mathcal{B}_1$ to output $m_0 \leftarrow M(pk)$ and some arbitrary $m_1 \neq m_0$, and setting $\mathcal{B}_2$ to output 0 if $C^* = \mathcal{E}(pk, M(pk))$ and 1 otherwise.

The leftover hash lemma was originally proven by Håstad et al. [18]. We present the version given by Shoup [27, Theorem 6.21]:

Lemma 3.13 (Leftover Hash Lemma). *Let (H, K, A, B) be a family of universal hash functions. If $k \xleftarrow{\$} K$, $x_1, \ldots, x_\ell$ are random variables on A which are independent of k, and $y_1, \ldots, y_\ell \xleftarrow{\$} B$, then*

$$\Delta\big[\big(k, H_k(x_1), \ldots, H_k(x_\ell)\big), (k, y_1, \ldots, y_\ell)\big] \leq \ell\sqrt{|B|\kappa}/2 \quad \text{where } \kappa = \max_{i=1}^{\ell}\big\{\kappa(x_i)\big\}.$$

We are now in a position to state and prove the main theorem of this section. This technical theorem essentially proves that a scheme which is IND-CPA secure and PA2 plaintext aware is PA2+ plaintext aware (and therefore PA1+ plaintext aware).

Theorem 3.14. *If a public-key encryption scheme Π is IND-CPA secure and 2PA2 plaintext aware, then it is 2PA2+ plaintext aware.*

Proof. Suppose $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$. We intend to simulate the randomness oracle by hashing ciphertexts $\mathcal{E}(pk, 0)$ using a universal hash function. Let $\ell(k)$ be a (polynomial) upper-bound on the length of $\mathcal{E}(pk, 0)$ (such a bound exists as $\mathcal{E}$ is a strict polynomial-time algorithm). Let A be the set of all strings of length at most $\ell(k)$ and $B = \{0, 1\}$. We can construct a universal hash family from A to B without computational assumptions [9], so let $(H_k)_{k \in K}$ be such a family.

Let $\mathcal{A}$ be a 2PA2+ ciphertext creator (i.e. $\mathcal{A}$ expects to have access to encryption, decryption, and randomness oracles). Let q_D and q_R be a bound on the number of decryption and randomness oracle queries respectively. We construct a 2PA2 ciphertext creator $\mathcal{B}$ (i.e. $\mathcal{B}$ has access to an encryption and decryption oracle) as in Fig. 6. (Note that we may assume that there exists a polynomial-bound $t(k)$ on the number of random bits on $\mathcal{A}$'s initial random tape as $\mathcal{A}$ is polynomial time.) Since $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is 2PA2 plaintext aware, there exists a plaintext extractor $\mathcal{B}^*$ for $\mathcal{B}$. We use $\mathcal{B}^*$ to construct a 2PA2+ plaintext extractor $\mathcal{A}^*$ for $\mathcal{A}$ as in Fig. 6.

We now prove that $\mathcal{A}^*$ simulates the decryption oracle for $\mathcal{A}$. Fix a PPT distinguisher algorithm D. We will define a series of games G_i in which $\mathcal{A}$ outputs a variable x and let W_i be the event that $D(x) = 1$ in game G_i. The games are summarised in Figs. 7 and 8.

Game G_1: G_1 is the $\text{EXPT}^{\text{FAKE-2PA2+}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}$ game.

Game G_2: G_2 is similar to G_1 except that some of the components of the plaintext extractor $\mathcal{A}^*$ are moved into the randomness extractor $\mathcal{O}_{\mathcal{R}}$. G_1 and G_2 are identical except for one (subtle) exception. The ciphertext creator $\mathcal{A}$ is forbidden from submitting a ciphertext $C \in \text{CLIST}$ to the decryption oracle. This means that, in G_2, the ciphertext creator is forbidden from querying the decryption oracle on $C \in \text{RCLIST}$. The two games are identical if this is does not occur. Throughout this proof, z will be the distribution of $\mathcal{E}(pk, 0)$ on $\mathcal{C}$. The elements of RCLIST are distributed according to z and are unknown to $\mathcal{A}$. By Lemma 3.12, the probability that some specific decryption oracle query C is equal to some specific ciphertext $C' \in \text{RCLIST}$ is bounded by

$\mathcal{B}^{\mathcal{O}'_{\mathcal{E}},\mathcal{O}'_{\mathcal{D}}}(1^k, pk)$:
$k \xleftarrow{\$} K$
$R[\mathcal{A}] \xleftarrow{\$} \{0,1\}^{t(k)}$
$x \leftarrow \mathcal{A}^{\mathcal{O}_{\mathcal{E}},\mathcal{O}_{\mathcal{D}},\mathcal{O}_{\mathcal{R}}}(1^k, pk; R[\mathcal{A}])$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(m_0, m_1)$
Query $C \leftarrow \mathcal{O}'_{\mathcal{E}}(m_0, m_1)$
Return C
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}(C)$
Query $m \leftarrow \mathcal{O}'_{\mathcal{D}}(C)$
Return m
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{R}}$
Query $C \leftarrow \mathcal{O}'_{\mathcal{E}}(0,0)$
$\rho \leftarrow H_k(C)$
Return ρ
Output x

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$:
On first invocation
$k \xleftarrow{\$} K$
$n_R \leftarrow 0$
$\text{RCLIST} \leftarrow \varepsilon$
For every $\mathcal{O}_{\mathcal{R}}$ query since last invocation
Repeat (k times) until $H_k(C') = \text{RLIST}[n_R]$
$C' \xleftarrow{\$} \mathcal{E}(pk, 0)$
Else
$C' \leftarrow 0$
Append C' to RCLIST
$n_R \leftarrow n_R + 1$
Interleave CLIST and RCLIST to give BCLIST
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, k \| R[\mathcal{A}], \text{BCLIST})$
Return m

Fig. 6. The ciphertext creator $\mathcal{B}$ and the plaintext extractor $\mathcal{A}^*$. Note that $R[\mathcal{B}] = k \| R[\mathcal{A}]$ as $\mathcal{B}$ generates no other random values. The variable n_R in $\mathcal{A}^*$ counts the total number of randomness oracle queries made by $\mathcal{A}$ and the list RCLIST gives a list of ciphertexts that, when hashed, give the bits in RLIST. The interleave operation in $\mathcal{A}^*$ interleaves CLIST and RCLIST according to the ordering of encryption oracle and randomness oracle queries made by $\mathcal{A}$. This creates a list BCLIST which is suitable for use with $\mathcal{B}^*$.

$\sqrt{\kappa(z)}$ and $\kappa(z)$ is bounded by $Adv^{\text{IND}}_{\mathcal{B}'}(k)$ for some IND-CPA adversary $\mathcal{B}'$. Therefore, $|\Pr[W_1] - \Pr[W_2]| \leq q_D q_R \sqrt{Adv^{\text{IND}}_{\mathcal{B}'}(k)}$.

Game G_3: G_3 is similar G_2 except that $\mathcal{O}_{\mathcal{R}}$ continues to generate ciphertexts C until it generates one for which $H_k(C) = \rho$. Obviously, G_2 and G_3 are identical as long as $\mathcal{O}_{\mathcal{R}}$ doesn't "abort" and set $C \leftarrow 0$ in G_2. By the Leftover Hash Lemma (Lemma 3.13), if $C' \xleftarrow{\$} \mathcal{E}(pk, 0)$ and $\rho \xleftarrow{\$} \{0,1\}$, then $\Delta[(k, H_k(C')), (k, \rho)] \leq \sqrt{\kappa(z)/2}$. Since $\kappa(z) = Adv^{\text{IND}}_{\mathcal{B}'}(k)$ and Π is IND-CPA secure, we have that $\Pr[H_k(C') = 0], \Pr[H_k(C') = 1] \leq 2/3$ for large enough k. Therefore, for large enough values of k, we have that the probability that we "abort" in G_2 is bounded by $q_R(2/3)^k$ and so $|\Pr[W_2] - \Pr[W_3]| \leq q_R(2/3)^k$.

Game G_4: G_4 is similar to G_3 except that we again alter the randomness oracle. In G_4 we compute $\rho \leftarrow H_k(C')$ for $C' \xleftarrow{\$} \mathcal{E}(pk, 0)$ rather than $\rho \xleftarrow{\$} \{0,1\}$. Suppose that $\mathcal{O}_{\mathcal{R}}$ outputs random bits $\rho_1, \ldots, \rho_{q_R}$ in G_3 and $\rho'_1, \ldots, \rho'_{q_R}$ in G_4. By the Leftover Hash Lemma, we have that

$$\Delta\left[(k, \rho_1, \ldots, \rho_{q_R}), \left(k, \rho'_1, \ldots, \rho'_{q_R}\right)\right] \leq q_R \sqrt{\kappa(z)/2} = q_R \sqrt{Adv^{\text{IND}}_{\mathcal{B}'}(k)/2}.$$

This is a distributional step in game hopping, see Appendix A, and so $|\Pr[W_4] - \Pr[W_3]| \leq q_R \sqrt{Adv^{\text{IND}}_{\mathcal{B}'}(k)/2}$.

Game G_5: A close examination of the inputs to $\mathcal{B}$ in G_4 shows that the event W_4 is the same as the event $\text{EXPT}^{\text{FAKE-2PA2}}_{\mathcal{B},\mathcal{B}^*,\mathcal{P},D} = 1$. G_5 is the game in which decryption oracle queries are answered by the real decryption algorithm. So W_5 is the same as $\text{EXPT}^{\text{REAL-2PA2}}_{\mathcal{B},\mathcal{P},D} = 1$. Thus, $|\Pr[W_5] - \Pr[W_4]|$ is negligible as Π is 2PA2 plaintext aware.

Game G_6: G_6 and G_7 reverse the previous changes to the randomness oracle. In G_6 the randomness oracle is changed so that it no longer adds ciphertexts to CLIST. Since decryption oracle queries are answered by $\mathcal{D}(sk, \cdot)$, rather than the plaintext extractor

G_1: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$
Return m

$\mathcal{O}_{\mathcal{R}}$:
$\rho \xleftarrow{\$} \{0,1\}$
Return ρ

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$:
On first invocation
$k \xleftarrow{\$} K$
$n_R \leftarrow 0$
$\text{RCLIST} \leftarrow \varepsilon$
For every $\mathcal{O}_{\mathcal{R}}$ query since last invocation
Repeat (k times) until $H_k(C') = \text{RLIST}[n_R]$
$C' \xleftarrow{\$} \mathcal{E}(pk, 0)$
Else
$C' \leftarrow 0$
Append C' to RCLIST
$n_R \leftarrow n_R + 1$
Interleave CLIST and RCLIST to give BCLIST
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, k \| R[\mathcal{A}], \text{BCLIST})$
Return m

G_2: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$
Return m

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$:
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, k \| R[\mathcal{A}], \text{CLIST})$
Return m

$\mathcal{O}_{\mathcal{R}}$:
On first invocation
$k \xleftarrow{\$} K$
$\rho \xleftarrow{\$} \{0,1\}$
Repeat (k times) until $H_k(C) = \rho$
$C \xleftarrow{\$} \mathcal{E}(pk, 0)$
Else
$C \leftarrow 0$
Append C to CLIST
Return ρ

G_3: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$
Return m

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$:
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, k \| R[\mathcal{A}], \text{CLIST})$
Return m

$\mathcal{O}_{\mathcal{R}}$:
On first invocation
$k \xleftarrow{\$} K$
$\rho \xleftarrow{\$} \{0,1\}$
Repeat until $H_k(C) = \rho$
$C \xleftarrow{\$} \mathcal{E}(pk, 0)$
Append C to CLIST
Return ρ

G_4: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$
Return m

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST}, \text{RLIST})$:
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, k \| R[\mathcal{A}], \text{CLIST})$
Return m

$\mathcal{O}_{\mathcal{R}}$:
On first invocation
$k \xleftarrow{\$} K$
$C \xleftarrow{\$} \mathcal{E}(pk, 0)$
$\rho \leftarrow H_k(C)$
Append C to CLIST
Return ρ

Fig. 7. Definition of games $G_1 - G_4$ for the proof of Theorem 3.14.

$\mathcal{A}^*$, the only affect of this change is to allow the ciphertext creator $\mathcal{A}$ to query the decryption oracle on ciphertexts in (the list previously described as) RCLIST. By a similar argument to G_2, we have that $|\Pr[W_6] - \Pr[W_5]| \leq q_D q_R \sqrt{Adv^{\text{IND}}_{\mathcal{B}'}(k)}$.

G_5: $\mathcal{O}_{\mathcal{D}}(C)$:
$\boxed{m \leftarrow \mathcal{D}(sk, C)}$
Return m

$\mathcal{O}_{\mathcal{R}}$:
On first invocation
$k \xleftarrow{\$} K$
$C \xleftarrow{\$} \mathcal{E}(pk, 0)$
$\rho \leftarrow H_k(C)$
Append C to CLIST
Return ρ

G_6: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \leftarrow \mathcal{D}(sk, C)$
Return m

$\mathcal{O}_{\mathcal{R}}$:
On first invocation
$k \xleftarrow{\$} K$
$C \xleftarrow{\$} \mathcal{E}(pk, 0)$
$\rho \leftarrow H_k(C)$
Return ρ

G_7: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \leftarrow \mathcal{D}(sk, C)$
Return m

$\mathcal{O}_{\mathcal{R}}$:
$\boxed{\rho \xleftarrow{\$} \{0, 1\}}$
Return ρ

Fig. 8. Definition of games $G_5 - G_7$ for the proof of Theorem 3.14.

Game G_7: G_7 changes the randomness oracle so that it simply returns a random bit. By a similar argument to G_4, we have that $|\Pr[W_7] - \Pr[W_6]| \leq q_R\sqrt{Adv^{\text{IND}}_{\text{B}'}(k)/2}$. Now, an examination of G_7 shows that it is identical to the $\text{EXPT}^{\text{REAL-2PA2+}}_{\mathcal{A},\mathcal{P},D}$ game. Therefore, we have that

$$\left|\Pr\left[\text{EXPT}^{\text{FAKE-2PA2+}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D} = 1\right] - \Pr\left[\text{EXPT}^{\text{REAL-2PA2+}}_{\mathcal{A},\mathcal{P},D} = 1\right]\right| = \left|\Pr[W_1] - \Pr[W_7]\right|$$

is negligible and so Π is 2PA2+ is plaintext aware. □

Corollary 3.15. *If a public-key encryption scheme* Π *is PA*2 *plaintext aware, IND-CPA secure and has that* $|\mathcal{M}|$ *grows faster than any polynomial in the security parameter, then* Π *is PA*2+ *plaintext aware* (*and therefore also PA*1+ *plaintext aware*).

Proof. Since Π is PA2 plaintext aware, it is 2PA2 plaintext aware (by inclusion). Since Π is IND-CPA secure and 2PA2 plainext aware, it is 2PA2+ plaintext aware (by Theorem 3.14). We also have that, since Π is PA2 plaintext aware, IND-CPA secure and has that $|\mathcal{M}|$ grows faster than any polynomial, Π is LH-IND-CPA (by Corollary 3.5). Lastly, since Π is LH-IND-CPA and 2PA2+, it is PA2+ (by an argument analogous to Corollary 3.9). □

4. Simulatable Sets and Algorithms

We now introduce a novel generalisation of dense sets: simulatable sets and algorithms. Roughly speaking, a set is dense if a randomly chosen element of the set is

indistinguishable from a randomly chosen element of the set $\{0,1\}^{\ell}$. In essence, a set is *simulatable* if it is computationally indistinguishable from the image of $\{0,1\}^{\ell}$ under an invertible polynomial-time map. Since one can easily generate random elements in $\{0,1\}^{\ell}$, one can also easily simulate a random element of the set by generating a random element of $\{0,1\}^{\ell}$ and applying the polynomial-time map (and vice versa).

We consider a family of sets indexed by a security parameter $k \in \mathbb{N}$. We may wish to simulate different sets at the same security level, so we allow for the possibility of further indexing at each security level. Hence, we consider families of sets of the form $S = ((S_i)_{i \in I_k})_{k \in \mathbb{N}}$ or $(S_k)_{k \in \mathbb{N}}$. A simulator for a family of sets $((S_i)_{i \in I_k})_{k \in \mathbb{N}}$ is a tuple (f, f^{-1}, ℓ) where $\ell(k)$ is a polynomial and (f, f^{-1}) is a pair of PPT algorithms with the following properties:

- f is a deterministic algorithm which takes as input a security parameter 1^k, an index $i \in I_k$, and a seed $r \in \{0,1\}^{\ell(k)}$, and outputs an element $s \in S_i$.
- f^{-1} is a probabilistic algorithm which takes as input a security parameter 1^k, an index i, and an element $s \in S$, and outputs a seed $r \in \{0,1\}^{\ell(k)}$.
- For all $k \in \mathbb{N}$, $i \in I_k$, $s \in S_i$, we have that $f(1^k, i, f^{-1}(1^k, i, s)) = s$.

In situations where the security parameter is clear by context, we will write $f(i,r)$ and $f^{-1}(i,s)$ for $f(1^k,i,r)$ and $f^{-1}(1^k,i,s)$, respectively.

We require that $f^{-1}(1^k, i, f(1^k,i,r))$ appears to be random. Let U_k be the uniform distribution on $\{0,1\}^{\ell(k)}$ and let Y_k be the distribution on $\{0,1\}^{\ell(k)}$ given by $f^{-1}(1^k, i, f(1^k,i,r))$ where $r \xleftarrow{\$} \{0,1\}^{\ell(k)}$. We define the simulator to be "statistically random-set simulatable" if $\Delta[U_k, Y_k]$ is negligible as a function of k. We define a simulator to be "computationally random-set simulatable" if the distributions U_k and Y_k are computationally indistinguishable. This is defined by the games in Fig. 9. A PPT algorithm $\mathcal{A}$ has advantage

$$Adv^{\text{RAND}}_{\mathcal{A}} = \left|\Pr\left[\text{EXPT}^{\text{RAND-1}}_{\mathcal{A}} = 1\right] - \Pr\left[\text{EXPT}^{\text{RAND-0}}_{\mathcal{A}} = 1\right]\right|$$

in breaking the random-set simulatable property of the simulator. A simulator is "computationally random-set simulatable" if every PPT algorithm $\mathcal{A}$ has negligible advantage.

We also require that $f(1^k, i, r)$ looks like a random element of the set S_i; however, the exact form of this definition depends upon whether we are talking about a simulatable set or a simulatable algorithm (of various different types). Let U_k be the uniform distribution on S_i and let Y_k be the distribution on S_i given by $f(1^k, i, r)$ for $r \xleftarrow{\$} \{0,1\}^{\ell(k)}$. We say that the set is "statistically image-set simulatable" if $\Delta[U_k, Y_k]$ is negligible as a function of k. A set is "computationally image-set simulatable" is U_k and Y_k are computationally indistinguishable. This is defined by the games in Fig. 10. A PPT attacker has advantage

$$Adv^{\text{IM}}_{\mathcal{A}}(k) = \left|\Pr\left[\text{EXPT}^{\text{IM-1}}_{\mathcal{A}} = 1\right] - \Pr\left[\text{EXPT}^{\text{IM-0}}_{\mathcal{A}} = 1\right]\right|$$

in breaking the image-set simulatable property of the simulator. A simulator is "computationally image-set simulatable" if every PPT algorithm $\mathcal{A}$ has negligible advantage.

$\text{EXPT}_{\mathcal{A}}^{\text{RAND-1}}$:
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{R}}}(1^k, i)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{R}}$
$r \xleftarrow{\$} \{0,1\}^{\ell}$
Return r
Output b'

$\text{EXPT}_{\mathcal{A}}^{\text{RAND-0}}$:
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{R}}}(1^k, i)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{R}}$
$r \xleftarrow{\$} \{0,1\}^{\ell}$
Return $f^{-1}(1^k, i, f(1^k, i, r))$
Output b'

Fig. 9. The security games for the random-set simulatable property. It is sufficient to restrict the attacker $\mathcal{A}$ to a single $\mathcal{O}_{\mathcal{R}}$ query as this is equivalent to the case where multiple $\mathcal{O}_{\mathcal{R}}$ queries are allowed.

$\text{EXPT}_{\mathcal{A}}^{\text{IM-1}}$:
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(1^k, i)$
If $\mathcal{A}$ queries $\mathcal{O}_S$
$s \xleftarrow{\$} S_i$
Return s
Output b'

$\text{EXPT}_{\mathcal{A}}^{\text{IM-0}}$:
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_S}(1^k, i)$
If $\mathcal{A}$ queries $\mathcal{O}_S$
$r \xleftarrow{\$} \{0,1\}^{\ell(k)}$
$s \leftarrow f(1^k, pk, r)$
Return s
Output b'

Fig. 10. The security games for the image-set simulatable property. It is sufficient to restrict the attacker $\mathcal{A}$ to a single $\mathcal{O}_S$ query as this is equivalent to the multi-query case.

Definition 4.1 (Simulatable Set). A family of sets is statistically simulatable if there exists a simulator which is both statistically random-set and statistically image-set simulatable. A family of sets is computationally simulate if there exists a simulator which is both computationally random-set and computationally image-set simulatable.

A simple hybrid argument/probability argument shows the following:

Lemma 4.2. *If the sets A and B are computationally/statistically simulatable, then the set $A \times B$ is computationally/statistically simulatable.*

We will require the use of public-key encryption schemes with computationally simulatable ciphertext spaces; however, since we are going to attempt to simulate ciphertexts in the presence of a decryption oracle, we require a stronger notion of simulatability than for sets. Note that we may define the ciphertext space of a public-key encryption scheme as a family of sets $((\mathcal{C}_{pk})_{pk \in PK_k})_{k \in \mathbb{N}}$ where PK_k is the set of all public keys that could be produced for the security parameter 1^k. We define ciphertext-space simulatability using the games in Fig. 11. We define the attacker's advantage $Adv_{\mathcal{A}}^{\text{PKE}}(k)$ in the usual way and define a public-key encryption scheme to be computationally ciphertext-space simulatable if every PPT algorithm $\mathcal{A}$ has negligible advantage.

Definition 4.3 (Simulate PKE). A public-key encryption scheme is simulatable if there exists a simulator for the family $((\mathcal{C}_{pk})_{pk \in PK_k})_{k \in \mathbb{N}}$ which is computationally random-set simulatable and computationally ciphertext-space simulatable.

$\text{EXPT}_{\mathcal{A}}^{\text{PKE-1}}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\mathcal{E}, \mathcal{D}(sk,\cdot)}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(m^*)$
$C^* \xleftarrow{\$} \mathcal{E}(pk, m^*)$
Return m^*
Output b'

$\text{EXPT}_{\mathcal{A}}^{\text{PKE-0}}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\mathcal{E}, \mathcal{D}(sk,\cdot)}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(m^*)$
$r \xleftarrow{\$} \{0,1\}^{\ell(k)}$
$C^* \leftarrow f(1^k, pk, r)$
Return m^*
Output b'

Fig. 11. The security games for the ciphertext-space simulatable property. The attacker may not submit any response from the encryption oracle $\mathcal{O}_{\mathcal{E}}$ to the decryption oracle $\mathcal{D}(sk, \cdot)$. It is sufficient to restrict the attacker $\mathcal{A}$ to a single $\mathcal{O}_{\mathcal{E}}$ query as this is equivalent to the multi-query case.

The following property was first noted by Stam [28].

Lemma 4.4. *If a public-key encryption scheme is simulatable, then it is IND-CCA2 secure.*

Sketch Proof. By the definition of a simulatable encryption scheme, one can replace the generation of the challenge ciphertext $C^* \xleftarrow{\$} \mathcal{E}(pk, m_b)$ with the generation of a simulated ciphertext $C^* \leftarrow f(1^k, pk, r)$ in the IND-CCA2 model. However, this is independent of the bit b, and so the probability that an attacker recovers b in this model is $1/2$. □

This lemma represents one of the major challenges of plaintext awareness: all known proofs for plaintext awareness require that the public-key encryption scheme is simulatable. Thus, these proof techniques can only ever be applied to schemes *which are already known to be IND-CCA2 secure* and so plaintext awareness is not an effective tool for proving IND-CCA2 security.

5. A Strategy for Proving PA2 Plaintext Awareness

In this section, we will prove that a public-key encryption scheme which is PA1+ plaintext aware and simulatable is PA2 plaintext aware. This is logical: if the encryption scheme is simulatable then ciphertexts are indistinguishable from random bitstrings. Hence, access to an encryption oracle is indistinguishable from access to a randomness oracle. This gives the only known strategy for proving that schemes are PA2 plaintext aware: one proves that the scheme is both simulatable and that it is PA1+ plaintext aware.

Theorem 5.1. *If Π is simulatable and PA1+ plaintext aware, then it is PA2 plaintext aware.*

Proof. Suppose that $\Pi = (\mathcal{G}, \mathcal{E}, \mathcal{D})$ is simulatable with simulator (f, f^{-1}, ℓ) and let $\mathcal{A}$ be a PA2 ciphertext creator. We define a PA1+ ciphertext creator $\mathcal{B}$ in Fig. 12. Since

$\mathcal{B}^{\mathcal{O}'_{\mathcal{D}},\mathcal{O}'_{\mathcal{R}}}(1^k, pk)$:
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{E}},\mathcal{O}_{\mathcal{D}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}$ on C
Query $m \leftarrow \mathcal{O}'_{\mathcal{D}}(C)$
Return m
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}$ on α
Query $r \leftarrow \mathcal{O}'_{\mathcal{R}}$
$C \leftarrow f(pk, r)$
Return C
Output x

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
On first invocation:
$\text{RLIST} \leftarrow \varepsilon$
$n_C \leftarrow 0$
Repeat until $n_C = |\text{CLIST}|$
$r' \xleftarrow{\$} f^{-1}(pk, \text{CLIST}[n_C])$
Append r' to RLIST
$n_C \leftarrow n_C + 1$
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$
Return m

Fig. 12. The ciphertext creator $\mathcal{B}$ and plaintext extractor $\mathcal{A}^*$. Without loss of generality, we assume that the randomness oracle $\mathcal{O}'_{\mathcal{R}}$ returns blocks of $\ell(k)$ random bits. This could be achieved by querying a one-bit oracle $\ell(k)$ times. The variable n_C counts the total number of ciphertexts on CLIST and RLIST contains a list of random bits which could have given rise to CLIST under the action of f.

Π is PA1+ plaintext aware, there exists a PA1+ plaintext extractor $\mathcal{B}^*$ for $\mathcal{B}$. We use $\mathcal{B}^*$ to construct a PA2 plaintext extractor $\mathcal{A}^*$ for $\mathcal{A}$ in Fig. 12.

This proof is similar to, but slightly simpler than, Theorem 3.14. Fix a PPT distinguisher D and let W_i be the event that $D(x) = 1$ in game G_i. The games are shown in Fig. 13.

Game G_1: G_1 is the $\text{EXPT}^{\text{FAKE-PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D}$ game.

Game G_2: G_2 is similar to G_1 except that the encryption oracle $\mathcal{O}_{\mathcal{E}}$ returns $f(pk, r)$ for $r \xleftarrow{\$} \{0,1\}^{\ell}$ rather than $\mathcal{E}(pk, \mathcal{P}(\alpha))$. Any difference between $\Pr[W_1]$ and $\Pr[W_2]$ gives rise to an attacker $\mathcal{B}'$ against the computational ciphertext-space simulatable property of the encryption scheme. The attacker $\mathcal{B}'$ is defined as follows:

$\mathcal{B}'^{\mathcal{O}'_{\mathcal{E}},\mathcal{D}(sk,\cdot)}(1^k, pk)$:
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{E}},\mathcal{O}_{\mathcal{D}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}$ on α
$m \xleftarrow{\$} \mathcal{P}(\alpha)$
Query $C \xleftarrow{\$} \mathcal{O}'_{\mathcal{E}}(m)$
Return C
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}$ on C
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
Return m
Output $D(x)$

The event W_1 is equivalent to $\text{EXPT}^{\text{PKE-1}}_{\mathcal{B}'} = 1$ and the event W_2 is equivalent to $\text{EXPT}^{\text{PKE-0}}_{\mathcal{B}'} = 1$. Hence, $|\Pr[W_1] - \Pr[W_2]| \leq Adv^{\text{PKE}}_{\mathcal{B}'}(k)$ is negligible. (Note that $\mathcal{B}'$ does not make use of its decryption oracle in this step. This will not be the case when we make an analogous game hop in G_5.)

Game G_3: G_3 is similar to G_2 except that the randomness oracle constructs RLIST rather than the plaintext extractor $\mathcal{A}^*$. In G_2, the element $f^{-1}(pk, f(pk, r))$ is added to RLIST. In G_3, the element r is added to RLIST. Any difference between $\Pr[W_2]$ and $\Pr[W_3]$ gives rise to an attacker $\mathcal{B}^*$ against the computational random-set simulatable property of the encryption scheme. The attacker $\mathcal{B}^*$ is defined as follows:

G_1: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
Return m

$\mathcal{O}_{\mathcal{E}}(\alpha)$:
$m \xleftarrow{\$} \mathcal{P}(\alpha)$
$C \xleftarrow{\$} \mathcal{E}(pk, C)$
Return C

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
On first invocation:
RLIST $\leftarrow \varepsilon$
$n_C \leftarrow 0$
Repeat until $n_C = |\text{CLIST}|$
$r' \xleftarrow{\$} f^{-1}(pk, \text{CLIST}[n_C])$
Append r' to RLIST
$n_C \leftarrow n_C + 1$
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$
Return m

G_2: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
Return m

$\mathcal{O}_{\mathcal{E}}(\alpha)$:
$\boxed{r \xleftarrow{\$} \{0,1\}^{\ell(k)}}$
$\boxed{C \leftarrow f(pk, r)}$
Return C

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
On first invocation:
RLIST $\leftarrow \varepsilon$
$n_C \leftarrow 0$
Repeat until $n_C = |\text{CLIST}|$
$r' \xleftarrow{\$} f^{-1}(pk, \text{CLIST}[n_C])$
Append r' to RLIST
$n_C \leftarrow n_C + 1$
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$
Return m

G_3: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
Return m

$\mathcal{O}_{\mathcal{E}}(\alpha)$:
$r \xleftarrow{\$} \{0,1\}^{\ell(k)}$
$C \leftarrow f(pk, r)$
$\boxed{\text{Append } r \text{ to RLIST}}$
Return C

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{CLIST})$
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$
Return m

G_4: $\mathcal{O}_{\mathcal{D}}(C)$:
$\boxed{m \leftarrow \mathcal{D}(sk, C)}$
Return m

$\mathcal{O}_{\mathcal{E}}(\alpha)$:
$r \xleftarrow{\$} \{0,1\}^{\ell(k)}$
$C \leftarrow f(pk, r)$
Return C

G_5: $\mathcal{O}_{\mathcal{D}}(C)$:
$m \leftarrow \mathcal{D}(sk, C)$
Return m

$\mathcal{O}_{\mathcal{E}}(\alpha)$:
$\boxed{m \xleftarrow{\$} \mathcal{P}(\alpha)}$
$\boxed{C \xleftarrow{\$} \mathcal{E}(pk, m)}$
Return C

Fig. 13. Definition of games G_1 and G_5 for the proof of Theorem 5.1.

$\mathcal{B}^{*\mathcal{O}'_{\mathcal{R}},\mathcal{D}(sk,\cdot)}(1^k, pk)$:
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{E}},\mathcal{O}_{\mathcal{D}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}$ on α
Query $r \leftarrow \mathcal{O}'_{\mathcal{R}}$
$C \leftarrow f(pk, r)$
Append r to RLIST
Return C
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}$ on C
$m \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$
Return m
Output $D(x)$

The randomness oracle $\mathcal{O}'_{\mathcal{R}}$ returns either $r \xleftarrow{\$} \{0,1\}^\ell$ or $f^{-1}(pk, f(pk, r))$ for $r \xleftarrow{\$} \{0,1\}^\ell$ depending on the random-set simulatable game. In both cases, $f(pk, r)$ is returned to the ciphertext creator $\mathcal{A}$, as $f(pk, f^{-1}(pk, f(pk, r))) = f(pk, r)$ since $f(pk, f^{-1}(pk, C)) = C$ for all $C \in \mathcal{C}$. The event W_2 is identical to $\text{EXPT}^{\text{RAND-0}}_{\mathcal{B}^*} = 1$ and the event W_3 is identical to $\text{EXPT}^{\text{RAND-1}}_{\mathcal{B}^*} = 1$. Thus, $|\Pr[W_3] - \Pr[W_2]| \leq Adv^{\text{RAND}}_{\mathcal{B}^*}(k)$ is negligible by the random-set simulatable property.

Game G_4: An examination of G_3 shows that it is identical to $\text{EXPT}^{\text{FAKE-PA1+}}_{\mathcal{B},\mathcal{B}^*,D}$. G_4 changes the decryption oracle so that it answers queries using the real decryption algorithm, i.e. G_4 is $\text{EXPT}^{\text{REAL-PA1+}}_{\mathcal{B},D}$. Since Π is PA1+, we have that $|\Pr[W_4] - \Pr[W_3]| \leq Adv^{\text{PA1+}}_{\mathcal{A},\mathcal{A}^*,\text{D}}(k)$ is negligible.

Game G_5: G_5 changes the action of the encryption oracle $\mathcal{O}_{\mathcal{E}}$ back to its original state, i.e. the encryption oracle returns $\mathcal{E}(pk, \mathcal{P}(\alpha))$. A similar argument to that in G_2 shows that any difference between $\Pr[W_4]$ and $\Pr[W_5]$ leads to an attacker $\mathcal{B}^\dagger$ against the computational ciphertext-space simulatable property of the encryption scheme (although in this case decryption oracle queries made by $\mathcal{A}$ are handled using $\mathcal{B}^\dagger$'s decryption oracle). Thus, $|\Pr[W_5] - \Pr[W_4]| \leq Adv^{\text{PKE}}_{\mathcal{B}^\dagger}(k)$ is negligible.

However, G_5 is identical to $\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\mathcal{P},D}$. So

$$\left|\Pr\left[\text{EXPT}^{\text{FAKE-PA2}}_{\mathcal{A},\mathcal{A}^*,\mathcal{P},D} = 1\right] - \Pr\left[\text{EXPT}^{\text{REAL-PA2}}_{\mathcal{A},\mathcal{P},D} = 1\right]\right| = \left|\Pr[W_1] - \Pr[W_5]\right|$$

is negligible and we conclude that Π is PA2 plaintext aware. □

A similar argument (with more complex book-keeping) can be used to prove that a public-key encryption scheme which is simulatable and PA1+ plaintext aware is necessarily PA2+ plaintext aware. However, we will be content with the basic result.

6. Extending the PA2 Proof Strategy to Hybrid Encryption Schemes

Section 5 provides a basic strategy for proving PA2 plaintext awareness; however, a proof which applies this methodology directly can quickly become complex in practice. In this section, we break the proof methodology down into several steps for hybrid encryption schemes.

6.1. *Hybrid Encryption Using KEMs and DEMs*

One common method to construct a practical public-key encryption scheme is to use hybrid encryption. This technique separates the public-key encryption scheme into an asymmetric key encapsulation mechanism (KEM) and a symmetric data encapsulation mechanism (DEM). Owing to the use of the symmetric DEM, the resulting public-key encryption scheme has the advantage of being able to encrypt messages of any length.

The notion of a hybrid encryption scheme was formalised using KEMs and DEMs by Cramer and Shoup [10,25]. A KEM is a triple of PPT algorithms $(Gen, Encap, Decap)$. The key generation algorithm takes a security parameter 1^k and outputs a public/private key pair $(pk, sk) \xleftarrow{\$} Gen(1^k)$. The public key defines a ciphertext space $\mathcal{C}$ and a symmetric key space $\mathcal{K}$. The encapsulation algorithm takes a public key pk as input, and outputs a ciphertext and a symmetric key $(C, K) \xleftarrow{\$} Encap(pk)$. The (deterministic) decapsulation algorithm takes a private key sk and a ciphertext $C \in \mathcal{C}$, and outputs either a symmetric key $K \leftarrow Decap(sk, C)$ or the distinguished error symbol $\perp$. For correctness we require that for all $(pk, sk) \xleftarrow{\$} Gen(1^k)$ and $(C, K) \xleftarrow{\$} Encap(pk)$, we have that $Decap(sk, C) = K$ with probability 1.

A DEM is defined by a pair of deterministic polynomial-time algorithms (ENC, DEC) and these algorithms implicitly define a key space $\mathcal{K}$, a message space $\mathcal{M}$, and a ciphertext space $\mathcal{C}$. The encapsulation algorithm takes a symmetric key $K \in \mathcal{K}$ and a message $m \in \mathcal{M}$ as input, and outputs a ciphertext $C \leftarrow \text{ENC}_K(m)$. The decapsulation algorithm takes a symmetric key $K \in \mathcal{K}$ and a ciphertext $C \in \mathcal{C}$ as input, and outputs either a message $m \leftarrow \text{DEC}_K(C)$ or the distinguished error symbol $\perp$. For correctness we require that for all $K \in \mathcal{K}$ and $m \in \mathcal{M}$, we have that $\text{DEC}_K(\text{ENC}_K(m)) = m$.

A public-key encryption scheme may be formed from a KEM and a DEM by setting the PKE key generation algorithm $\mathcal{G}$ to be the KEM key generation algorithm Gen, and computing encryption and decryption as in Fig. 14. Cramer and Shoup proved that the combination of a sufficiently secure KEM and a sufficiently secure DEM leads to an IND-CCA2 public-key encryption scheme [10,25], although the exact form of that result is not relevant to this paper. However, we will note that a "sufficiently secure" DEM can be constructed through an "encrypt-then-MAC" construction using, for example, counter mode encryption with a CBC-MAC algorithm.

6.2. *Simulatable KEMS and DEMs*

Our strategy requires us to extend the notion of simulatable encryption to KEMs and DEMs. These notions are very similar to that of a simulatable encryption scheme. For a simulatable KEM, we consider the family of sets which define the ciphertext space $((\mathcal{C}_{pk})_{pk \in PK_k})_{k \in \mathbb{N}}$ as in Sect. 4 and define ciphertext-space simulatability using

$\mathcal{E}(pk, m)$:
$(C_1, K) \xleftarrow{\$} Encap(pk)$
$C_2 \leftarrow \text{ENC}_K(m)$
$C \leftarrow (C_1, C_2)$
Return C

$\mathcal{D}(sk, C)$:
Parse C as (C_1, C_2)
$K \leftarrow Decap(sk, C_1)$
$m \leftarrow \text{DEC}_K(C_2)$
Return m

Fig. 14. Construction of a public-key encryption scheme from a KEM and a DEM.

$\text{EXPT}_{\mathcal{A}}^{\text{KEM-1}}$:
$(pk, sk) \xleftarrow{\$} Gen(1^k)$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\mathcal{E}, \mathcal{D}(sk,\cdot)}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}$
$(C^*, K^*) \xleftarrow{\$} Encap(pk)$
Return (C^*, K^*)
Return b'

$\text{EXPT}_{\mathcal{A}}^{\text{KEM-0}}$:
$(pk, sk) \xleftarrow{\$} Gen(1^k)$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\mathcal{E}, \mathcal{D}(sk,\cdot)}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}$
$r \xleftarrow{\$} \{0,1\}^{\ell(k)}$
$C^* \leftarrow f(1^k, pk, r)$
$K^* \xleftarrow{\$} \mathcal{K}$
Return (C^*, K^*)
Return b'

Fig. 15. The security games for the KEM ciphertext-space simulatable property. The attacker may not submit any ciphertext C^* received from the encapsulation oracle $\mathcal{O}_{\mathcal{E}}$ to the decapsulation oracle $\mathcal{D}(sk, \cdot)$. It is sufficient to restrict the attacker $\mathcal{A}$ to a single $\mathcal{O}_{\mathcal{E}}$ query as this is equivalent to the multi-query case.

$\text{EXPT}_{\mathcal{A}}^{\text{DEM-1}}$:
$K \xleftarrow{\$} \mathcal{K}$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\mathcal{E}, \text{DEC}_K(\cdot)}(1^k)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(m^*)$
$C^* \leftarrow \text{ENC}_K(m^*)$
Return C^*
Return b'

$\text{EXPT}_{\mathcal{A}}^{\text{DEM-0}}$:
$K \xleftarrow{\$} \mathcal{K}$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\mathcal{E}, \text{DEC}_K(\cdot)}(1^k)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{E}}(m^*)$
$r \xleftarrow{\$} \{0,1\}^{\ell(k)}$
$C^* \leftarrow f(1^k, r)$
Return C^*
Return b'

Fig. 16. The security games for the DEM ciphertext-space simulatable property. We assume that the attacker does not submit the same message to the encryption oracle twice. The attacker may not submit any ciphertext C^* received from the encryption oracle $\mathcal{O}_{\mathcal{E}}$ to the decryption oracle $\text{DEC}_K(\cdot)$. It is sufficient to restrict the attacker $\mathcal{A}$ to a single $\mathcal{O}_{\mathcal{E}}$ query as this is equivalent to the multi-query case.

the games defined in Fig. 15. We define the attacker's advantage $Adv_{\mathcal{A}}^{\text{KEM}}(k)$ in the usual way and define a KEM to be computationally ciphertext-space simulatable if every PPT algorithm $\mathcal{A}$ has negligible advantage.

Definition 6.1 (Simulatable KEM). A KEM is simulatable if there exists a simulator for the family $((\mathcal{C}_{pk})_{pk \in PK_k})_{k \in \mathbb{N}}$ which is computationally random-set simulatable and computationally ciphertext-space simulatable.

For a simulatable DEM, we consider the family of ciphertext spaces $(\mathcal{C}_k)_{k \in \mathbb{N}}$ and define ciphertext-space simulatability via the games in Fig. 16. We define the attacker's advantage $Adv_{\mathcal{A}}^{\text{DEM}}(k)$ in the usual way and define a DEM to be computationally ciphertext-space simulatable if every PPT algorithm $\mathcal{A}$ has negligible advantage.

Definition 6.2 (Simulatable DEM). A DEM is simulatable if there exists a simulator for the family $(C)_{k \in \mathbb{N}}$ which is computationally random-set simulatable and computationally ciphertext-space simulatable.

There is a strong similarity between the notion of simulatable DEMs and the notion of IND-R-CCA2 security for symmetric encryption schemes [24]. Using similar

techniques, it is relatively easy to show that an encrypt-then-MAC DEM composed of counter model encryption and CBC-MAC (each using an independent ideal cipher) is simulatable when $\mathcal{M} = \{0,1\}^n$ [5,22]. The following lemma follows easily from the definitions:

Lemma 6.3. *The composition of a simulatable KEM (with a super-polynomially-sized symmetric key space) and a simulatable DEM is a simulatable hybrid encryption scheme.*

Remark 6.4. We have given the original definitions of simulatable sets/algorithms [12] in which the attacker is given access to an oracle which supplies multiple set elements/ciphertexts. A simple hybrid argument can be used to show that the definitions are equivalent to a simpler definition in which an attacker can only obtain a single set element/ciphertext. These definitions can be phrased in terms of an attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in which $\mathcal{A}_1$ outputs the single oracle query and $\mathcal{A}_2$ processes this oracle's response. We will use both version of these definitions in this paper. To prove a scheme is simulatable, we will use the single-query version of the definition; however, when making use of a simulatable set/scheme in a larger construction, we will assume the multi-query definition.

6.3. *Proving PA2 Plaintext Awareness for Hybrid Encryption*

We begin by defining a notion of PA1+ plaintext awareness for KEMs via the two games given in Fig. 17 (which make use of a ciphertext creator $\mathcal{A}$, a plaintext extractor $\mathcal{A}^*$ and a distinguisher algorithm D). The advantage is defined as

$$Advv^{\text{KEM-PA1+}}_{\mathcal{A},\mathcal{A}^*,D}(k) = \left|\Pr\left[\text{EXPT}^{\text{REAL-KEM-PA1+}}_{\mathcal{A},D} = 1\right] - \Pr\left[\text{EXPT}^{\text{FAKE-KEM-PA1+}}_{\mathcal{A},\mathcal{A}^*,D} = 1\right]\right|.$$

Definition 6.5. A KEM Π_K is PA1+ plaintext aware if for every PPT ciphertext creator $\mathcal{A}$ there exists a PPT plaintext extractor $\mathcal{A}^*$ such that for all PPT distinguisher algorithms D we have that $Advv^{\text{KEM-PA1+}}_{\mathcal{A},\mathcal{A}^*,D}(k)$ is negligible.

$\text{EXPT}^{\text{REAL-KEM-PA1+}}_{\mathcal{A},D}$:
$(pk, sk) \xleftarrow{\$} Gen(1^k)$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_\mathcal{D},\mathcal{O}_\mathcal{R}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_\mathcal{R}$
$\rho \xleftarrow{\$} \{0,1\}$
Return ρ
If $\mathcal{A}$ queries $\mathcal{O}_\mathcal{D}(C)$
$K \leftarrow Decap(sk, C)$
Return K
$b \xleftarrow{\$} D(x)$
Output b

$\text{EXPT}^{\text{FAKE-KEM-PA1+}}_{\mathcal{A},\mathcal{A}^*,D}$:
$(pk, sk) \xleftarrow{\$} Gen(1^k)$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_\mathcal{D},\mathcal{O}_\mathcal{R}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_\mathcal{R}$
$\rho \xleftarrow{\$} \{0,1\}$
Append ρ to RLIST
Return ρ
If $\mathcal{A}$ queries $\mathcal{O}_\mathcal{D}(C)$
$K \xleftarrow{\$} \mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$
Return K
$b \xleftarrow{\$} D(x)$
Output b

Fig. 17. The PA1+ plaintext awareness security games for KEMs.

We now simplify the process of proving PA1+ plaintext awareness in the case of hybrid encryption schemes by showing that a hybrid encryption scheme Π is PA1+ plaintext aware if and only if its KEM Π_K is PA1+ plaintext aware. This proof technique does not immediately extend to PA2 plaintext awareness and this issue is further discussed by Jiang and Wang [19].

Lemma 6.6. *Suppose Π is a public-key encryption scheme composed of a KEM scheme Π_K and a DEM scheme Π_D. If Π_K is PA1+ plaintext aware, then Π is PA1+ plaintext aware.*

Proof. Let $\mathcal{A}$ be a PA1+ ciphertext creator for Π and suppose that the DEM Π_D is a (deterministic) DEM of the form (ENC, DEC). We may define a PA1+ ciphertext creator $\mathcal{B}$ for Π_K as in Fig. 18. Since Π_K is PA1+, there exists a plaintext extractor $\mathcal{B}^*$ for $\mathcal{B}$. We define a plaintext extractor $\mathcal{A}^*$ for Π in Fig. 18. An examination of the games easily shows that for any distinguisher algorithm D, we have $\text{EXPT}^{\text{REAL-PA1+}}_{\mathcal{A},D} = \text{EXPT}^{\text{REAL-KEM-PA1+}}_{\mathcal{B},D}$ and $\text{EXPT}^{\text{FAKE-PA1+}}_{\mathcal{A},\mathcal{A}^*,D} = \text{EXPT}^{\text{FAKE-KEM-PA1+}}_{\mathcal{B},\mathcal{B}^*,D}$, which means that Π if PA1+ if Π_K is PA1+. □

This now gives a complete strategy to prove that a hybrid encryption scheme, consisting of a KEM Π_K and a DEM Π_D, is PA2 plaintext aware:

- Prove that Π_K is PA1+ plaintext aware.
- Prove that both Π_K and Π_D are simulatable.

Recall that several DEMs have already been shown to be simulatable [5,22] and so we may concentrate on proving that there are PA1+ and simulatable KEMs.

7. The Cramer–Shoup Encryption Scheme

In this section, we will use our strategy to show the hybrid Cramer–Shoup encryption scheme [10] is PA2 under a variety of assumptions including the interactive Diffie–Hellman Knowledge (DHK) assumption.[3]

$\mathcal{B}^{\mathcal{O}'_{\mathcal{D}},\mathcal{O}'_{\mathcal{R}}}(1^k, pk)$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}},\mathcal{O}_{\mathcal{R}}}$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{R}}$
Query $r \leftarrow \mathcal{O}'_{\mathcal{R}}$
Return r
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}(C)$
Parse C as (C_1, C_2)
Query $K \leftarrow \mathcal{O}'_{\mathcal{D}}(C_1)$
$m \leftarrow \text{DEC}_K(C_2)$
Return m
Output x

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$
Parse C as (C_1, C_2)
$K \xleftarrow{\$} \mathcal{B}^*(1^k, pk, C_1, R[\mathcal{A}], \text{RLIST})$
$m \leftarrow \text{DEC}_K(C_2)$
Output m

Fig. 18. The ciphertext creator $\mathcal{B}$ and plaintext extractor $\mathcal{A}^*$.

[3] This is also sometimes known as the Knowledge-of-Exponent assumption (KEA).

7.1. *Assumptions*

The Cramer–Shoup encryption scheme is defined over a group $\mathbb{G}$ with a generator g of prime order q. We will require that the group is statistically simulatable (as a set), that the DDH problem is hard on $\mathbb{G}$, and that the DHK assumption holds on $\mathbb{G}$.

Definition 7.1 (DDH). The DDH advantage of an attacker $\mathcal{A}$ is defined as

$$Adv^{\text{DDH}}_{\mathcal{A}}(k) = \left| \Pr\left[b' = 1 : \begin{array}{c} x, y \xleftarrow{\$} \mathbb{Z}_q, \\ b' \xleftarrow{\$} \mathcal{A}(g, g^x, g^y, g^{xy}) \end{array} \right] - \Pr\left[b' = 1 : \begin{array}{c} x, y, z \xleftarrow{\$} \mathbb{Z}_q, \\ b' \xleftarrow{\$} \mathcal{A}(g, g^x, g^y, g^z) \end{array} \right] \right|.$$

The DDH problem is hard if every PPT attacker $\mathcal{A}$ has negligible advantage.

The DHK assumption was first introduced by Dåmgard [11]. It is a very strong, interactive assumption which is designed to capture the intuition that it does not seem to be possible to create a Diffie–Hellman tuple (g, a, g^x, a^x) from (g, a) without knowing x. Let $DH = \{(g, g^x, g^y, g^{xy}) : x, y \in \mathbb{Z}_q\}$. We define DHK security via the security game shown in Fig. 19.

Definition 7.2 (DHK). The DHK advantage of an attacker $\mathcal{A}$ and an extractor $\mathcal{A}^*$ is defined as $Adv^{\text{DHK}}_{\mathcal{A},\mathcal{A}^*}(k) = \Pr[\text{EXPT}^{\text{DHK}}_{\mathcal{A},\mathcal{A}^*} = 1]$. The DHK assumption holds if for every PPT attacker $\mathcal{A}$ there exists a PPT extractor algorithm $\mathcal{A}^*$ such that $Adv^{\text{DHK}}_{\mathcal{A},\mathcal{A}^*}(k)$ is negligible.

In order to prove that the Cramer–Shoup encryption scheme is plaintext aware, we require groups which are simulatable and on which the DDH and DHK assumptions are believed to hold. We will aim to show that a q-order finite field group $\mathbb{G}_k \subseteq \mathbb{F}_{p_k}$ is simulatable. Our strategy will be to show that $\mathbb{F}_{p_k}$ is simulatable and that sufficiently large subsets of simulatable sets are also simulatable.

We begin by showing that families of sets $S = ((S_i)_{i \in I_k})_{k \in \mathbb{N}}$ with $S_i = \mathbb{Z}_{N_i}$ and $2^{k-1} < N_i < 2^k$ are statistically simulatable. We define a simulator $\Pi_S = (f, f^{-1}, \ell)$

$\text{EXPT}^{\text{DHK}}_{\mathcal{A},\mathcal{A}^*}$:
$a \xleftarrow{\$} \mathbb{G}$
$\mathcal{A}^{\mathcal{O}}(1^k, g, a)$
If $\mathcal{A}$ queries $\mathcal{O}(b, c)$
$x \xleftarrow{\$} \mathcal{A}^*(g, a, b, c, R[\mathcal{A}])$
If $(g, a, b, c) \in DH$ and $b \neq g^x$ then output 1
Else return x
Output 0

Fig. 19. The security game for the Diffie–Hellman Knowledge (DHK) assumption. $\mathcal{A}^*$ is run as a stateful algorithm (which retains state between invocations).

where $\ell = 2k$ and

$f(i, r)$:
  $x \leftarrow r \bmod N_i$
  Output x

$f^{-1}(i, s)$:
  $\alpha_i \leftarrow \lfloor 2^{2k}/N_i \rfloor$
  $t \xleftarrow{\$} \{0, \ldots, \alpha_i - 1\}$
  $r \leftarrow tN_i + s$
  Output r

Lemma 7.3. *The simulator Π_S witnesses the fact that S is simulatable, i.e. Π_S is statistically random-set and statistically image-set simulatable.*

Image-Set Simulatable Proof. Let $\alpha_i = \lfloor 2^{2k}/N_i \rfloor$ and $\beta_i = 2^{2k} \bmod N_i$ (i.e. $2^{2k} = \alpha_i N_i + \beta_i$). Let U be the uniform distribution on S_i and let Y be the distribution given by $f(i, r)$ for $r \xleftarrow{\$} \{0, 1\}^\ell$. Split S_i into two disjoint sets S_i^+ and S_i^- where $S_i^+ = \{0, \ldots, \beta_i - 1\}$ and $S_i^- = \{\beta_i, \ldots, N_i - 1\}$. Hence,

$$\big|\Pr[Y = s] - \Pr[U = s]\big| = \begin{cases} \left|\frac{\alpha_i + 1}{2^{2k}} - \frac{1}{N_i}\right| & \text{if } s \in S_i^+, \\ \left|\frac{\alpha_i}{2^{2k}} - \frac{1}{N_i}\right| & \text{if } s \in S_i^-. \end{cases}$$

Since $2^{2k} - \alpha_i N_i = \beta_i$, and $\beta_i < N_i < 2^k$, we have that $|\Pr[Y = s] - \Pr[U = s]| \leq 1/2^{2k}$ for all $s \in S_i$. Thus, $\Delta[U, Y] \leq 1/2^k$ and S is statistically image-set simulatable. □

Random-Set Simulatable Proof. Now let U' be the uniform distribution on $\{0, 1\}^\ell$ and Y' be the distribution of $f^{-1}(i, f(i, r))$ for $r \xleftarrow{\$} \{0, 1\}^\ell$. We have

$$\big|\Pr[Y' = r'] - \Pr[U' = r']\big| = \begin{cases} \left|\frac{\alpha_i + 1}{\alpha_i 2^{2k}} - \frac{1}{2^{2k}}\right| & \text{if } r' < \alpha_i N_i \text{ and } r' \bmod N_i \in S_i^+, \\ \left|\frac{\alpha_i}{\alpha_i 2^{2k}} - \frac{1}{2^{2k}}\right| & \text{if } r' < \alpha_i N_i \text{ and } r' \bmod N_i \in S_i^-, \\ \frac{1}{2^{2k}} & \text{if } r' \geq \alpha_i N_i. \end{cases}$$

There exists $\alpha_i \beta_i \leq \alpha_i N_i$ values for r' with $r' < \alpha_i N_i$ and $r' \bmod N_i \in S_i^+$ and there exists $\beta_i < N_i$ values r' with $r' \geq \alpha_i N_i$. Thus, $\Delta[U', Y'] \leq 1/2^k$ and S is statistically random-set simulatable. □

This demonstrates that $\mathbb{F}_{p_k}$ is statistically simulatable when p_k is a k-bit prime. However, this does not imply that a prime order group $\mathbb{G} \subseteq \mathbb{F}_{p_k}$ is necessarily simulatable. To prove that we use the following lemma:

Lemma 7.4 (Simulatable Subset Lemma). *Suppose $S = ((S_i)_{i \in I_k})_{k \in \mathbb{N}}$ is statistically simulatable family of finite sets and that $S' = ((S'_i)_{i \in I_k})_{k \in \mathbb{N}}$ satisfies $S'_i \subseteq S_i$ for all $i \in I_k$ and $k \in \mathbb{N}$. Let $\mu(k, i) = |S'_i|/|S_i|$ and $\nu(k, i) = 1 - \mu(k, i)$. Suppose that there exists a polynomial $p(k)$ such that $\mu(k, i), \nu(k, i) \geq |1/p(k)|$ for all sufficiently large k and $i \in I_k$ and that there exists polynomial-time algorithm $Ber_\mu(1^k, i)$ which outputs a bit from the Bernoulli distribution with probability $\mu(k, i)$. Suppose further that membership of S_i and S'_i are polynomial-time decideable. Then S' is a statistically simulatable family of sets.*

$f'(i, r)$:
  $bad\_f \leftarrow \texttt{false}$
  Parse r as $r_1 \| \ldots \| r_{\ell'}$
    where $r_j \in \{0,1\}^{\ell'}$
  For $j = 1, 2, \ldots, kp(k)$:
    If $f(i, r_j) \in S'_i$
      Output $f(i, r_j)$
  $bad\_f \leftarrow \texttt{true}$
  Output 0

$f'^{-1}(i, s)$:
  $r \leftarrow \varepsilon$
  $bad\_finv \leftarrow \texttt{false}$
  $found \leftarrow \texttt{false}$
  For $j = 1, 2, \ldots, kp(k)$:
    $\rho \xleftarrow{\$} Ber_\mu(1^k, i)$
    If $\rho = 1$ and
    $found = \texttt{false}$
      $r_j \xleftarrow{\$} f^{-1}(i, s)$
      Append r_j to r
      $found \leftarrow \texttt{true}$
    If $\rho = 0$ and
    $found = \texttt{false}$
      Repeat ($kp(k)$ times)
      until $s'_j \notin S'_i$
        $r_j \xleftarrow{\$} \{0,1\}^\ell$
        $s'_j \leftarrow f(i, r_j)$
      Else
        $s'_j \leftarrow f(i, 0^\ell)$
        $bad\_finv \leftarrow \texttt{true}$
      Append $f^{-1}(i, s'_j)$ to r
    If $found = \texttt{true}$
      $r_j \xleftarrow{\$} \{0,1\}^{\ell(k)}$
      Append r_j to r
  Output r

Fig. 20. Simulator f' and f'^{-1} for sets $S' \subseteq S$.

Image-Set Simulatable Proof. Suppose (f, f^{-1}, ℓ) is a statistical simulator for S. We define a new simulator (f', f'^{-1}, ℓ') where $\ell'(k) = kp(k)\ell(k)$ and the remaining algorithms are given in Fig. 20.

This proof can thought of as "distributional game hopping" and makes liberal use of the results of Appendix A. We define a number of distributions – un-prime distributions (e.g. X) are associated with (f, f^{-1}, ℓ) and prime distributions (e.g. X') are associated with (f', f'^{-1}, ℓ'):

- U_ℓ, $U_{\ell'}$, U_T refer to the uniform distributions over the sets $\{0,1\}^\ell$, $\{0,1\}^{\ell'}$, T respectively. Note that T could be any set including S_i, S'_i and $S_i \setminus S'_i$.
- X refers to the distribution $f(i, r)$ for $r \xleftarrow{\$} \{0,1\}^\ell$.
- Y refers to the distribution $f^{-1}(i, f(i, r))$ for $r \xleftarrow{\$} \{0,1\}^\ell$.
- Z refers to the distribution $f^{-1}(i, s)$ for $s \xleftarrow{\$} S_i$.
- X' refers to the distribution $f'(i, r)$ for $r \xleftarrow{\$} \{0,1\}^{\ell'}$ and $\bar{X}'$ refers to the distribution $f'(i, r)$ for $r \xleftarrow{\$} \{0,1\}^{\ell'}$ conditioned on the fact that f' does not set bad_f to be true. Note that $\Delta[X', \bar{X}'] \leq 1/2^k$.
- Y' refers to the distribution $f'^{-1}(i, f(i, r))$ for $r \xleftarrow{\$} \{0,1\}^{\ell'}$ and $\bar{Y}'$ refers to the distribution $f'^{-1}(i, f(i, r))$ for $r \xleftarrow{\$} \{0,1\}^{\ell'}$ conditioned on the fact that f' does not set bad_f to be true. Note that $\Delta[Y', \bar{Y}'] \leq 1/2^k$.
- Z' refers to the distribution $f'^{-1}(i, s)$ for $s \xleftarrow{\$} S'_i$ and $\bar{Z}'$ refers to the distribution $f'^{-1}(i, s)$ for $s \xleftarrow{\$} S'_i$ conditioned on the fact that f'^{-1} does not set bad_finv to be $\texttt{true}$. Note that $\Delta[Z', \bar{Z}'] \leq 1/2^k$.

In order to show that S' is statistically image-set simulatable, we need to show that $\Delta[U_{S'_i}, X']$ is negligible. This follows simply as

$$\Delta\left[U_{S'_i}, X'\right] \leq \Delta\left[U_{S'_i}, \bar{X}'\right] + \Delta\left[\bar{X}', X'\right] \leq \Delta\left[U_{S'_i}, \bar{X}'\right] + 1/2^k \leq p(k)\Delta[U_S, X] + 1/2^k$$

and as $\Delta[U_{S_i}, X]$ is negligible (by definition). □

Random-Set Simulatable Proof. It is more difficult to show that S' is statistically random-set simulatable. We make use of the following simple fact:

$$\Delta[U_\ell, Z] \leq \Delta[U_\ell, Y] + \Delta[Y, Z] \leq \Delta[U_\ell, Y] + \Delta[X, U_{S_i}].$$

Thus, $\Delta[U_\ell, Z]$ is negligible (as S is statistically image-set and random-set simulatable). This can be interpreted as saying that the distribution $f^{-1}(i, s)$ for $s \xleftarrow{\$} S_i$ is statistically close to uniform on $\{0,1\}^\ell$.

Consider $r' \xleftarrow{\$} f'^{-1}(i, f'(i, r))$ for $r \xleftarrow{\$} \{0,1\}^{\ell'}$. We may split r' into "blocks" $r'_1 \| \ldots \| r'_{kp(k)}$ where each $r'_j \in \{0,1\}^\ell$. Let λ be the smallest integer such that $f(i, r'_\lambda) \in S'_i$ and note that the value λ is correctly distributed—i.e. if we chose $s_j \xleftarrow{\$} S_i$ and set λ to be the smallest value for which $s_\lambda \in S'_i$ then λ would have the same binomial distribution as is produced by f^{-1}. Therefore, it is sufficient to show that each "block" r_j is statistically close to an independent copy of the uniformly random distribution U_ℓ. This is trivially true for blocks with $j > \lambda$.

In order to show that S' is statistically random-set simulatable, we have to show that $\Delta[U_{\ell'}, Y']$ is negligible. We begin by noting that

$$\begin{aligned}
\Delta\big[U_{\ell'}, Y'\big] &\leq \Delta\big[U_{\ell'}, \bar{Y}'\big] + \Delta\big[\bar{Y}', Y'\big] \\
&\leq \Delta\big[U_{\ell'}, \bar{Y}'\big] + 1/2^k && \big(\text{as } \Delta\big[\bar{Y}', Y'\big] \leq 1/2^k\big) \\
&\leq \Delta\big[U_{\ell'}, Z'\big] + \Delta\big[Z', \bar{Y}'\big] + 1/2^k \\
&\leq \Delta\big[U_{\ell'}, Z'\big] + \Delta\big[U_{S'_i}, \bar{X}'\big] + 1/2^k && (*) \\
&\leq \Delta\big[U_{\ell'}, Z'\big] + \Delta\big[U_{S'_i}, X'\big] + \Delta\big[X', \bar{X}'\big] + 1/2^k \\
&\leq \Delta\big[U_{\ell'}, Z'\big] + \Delta\big[U_{S'_i}, X'\big] + 2/2^k && \big(\text{as } \Delta\big[\bar{X}', X'\big] \leq 1/2^k\big) \\
&\leq \Delta\big[U_{\ell'}, \bar{Z}'\big] + \Delta\big[\bar{Z}', Z'\big] + \Delta\big[U_{S'_i}, X'\big] + 2/2^k \\
&\leq \Delta\big[U_{\ell'}, \bar{Z}'\big] + \Delta\big[U_{S'_i}, X'\big] + 3/2^k && \big(\text{as } \Delta\big[\bar{Z}', Z'\big] \leq 1/2^k.\big)
\end{aligned}$$

The inequality $(*)$ holds because the only difference between Z' and $\bar{Y}'$ is in the distribution of the element s' input to f'^{-1}. We can characterise $\bar{Y}'$ as $f'^{-1}(i, \bar{X}')$ and Z' as $f'^{-1}(i, U_{S'_i})$. Hence, $\Delta[Z', \bar{Y}'] \leq \Delta[\bar{X}', U_{S'_i}]$. We have already shown that $\Delta[U_{S'_i}, X']$ is negligible and so it suffices to bound $\Delta[U_{\ell'}, \bar{Z}']$. In other words, it suffices to show that the distribution $f'^{-1}(i, s)$ for $s \xleftarrow{\$} S'_i$ (conditioned on the fact that f'^{-1} does not set *bad_finv* to be `true`) is statistically close to uniform.

In order to show that $\Delta[U_{\ell'}, \bar{Z}']$ is negligible, we compare the simulator $f'^{-1}(i,s)$ to a perfect (non-polynomial-time) version $\hat{f}^{-1}(i,s)$:

$\hat{f}^{-1}(i,s)$:
$r \leftarrow \varepsilon$
$found \leftarrow \texttt{false}$
For $j = 1, 2, \ldots, kp(k)$:
$\rho \xleftarrow{\$} Ber_\mu(1^k, i)$
If $\rho = 1$ and $found = \texttt{false}$
$r_j \xleftarrow{\$} f^{-1}(i,s)$
Append r_j to r
$found \leftarrow \texttt{true}$
If $\rho = 0$ and $found = \texttt{false}$
$\boxed{s'_j \xleftarrow{\$} S_i \setminus S'_i}$
Append $f^{-1}(i, s'_j)$ to r
If $found = \texttt{true}$
$r_j \xleftarrow{\$} \{0,1\}^{\ell(k)}$
Append r_j to r
Output r

Let $\hat{Z}'$ be the distribution of $\hat{f}^{-1}(i,s)$ for $s \xleftarrow{\$} S'_i$. We have

$$\Delta\big[U_{\ell'}, \bar{Z}'\big] \leq \Delta\big[U_{\ell'}, \hat{Z}'\big] + \Delta\big[\hat{Z}', \bar{Z}'\big].$$

We bound the term $\Delta[\hat{Z}', \bar{Z}']$ by noting that the only difference between the two distributions is in the distribution of the value s'_j. We can characterise $\bar{Z}'$ as $f'^{-1}(i, \bar{X}')$ conditioned on the fact that f'^{-1} does not set the flag *bad_finv* to be `true` and $\hat{Z}'$ as $f'^{-1}(U_{S_i \setminus S'_i})$. In other words, we require that the distribution of $\bar{X}'$ on $S_i \setminus S'_i$ is sufficiently close to uniform. By an argument that is almost identical to the proof that S' is statistically image-set simulatable, we have that these two distributions are statistically indistinguishable. Hence, $\Delta[\hat{Z}', \bar{Z}']$ is negligible.

It therefore only remains to prove that $\Delta[U_{\ell'}, \hat{Z}']$ is negligible. In $\hat{Z}'$, each "block" r'_j output by $\hat{f}^{-1}(i,s)$ is either uniformly randomly distributed or distributed as $f^{-1}(i,s)$ for $s \xleftarrow{\$} S$ (i.e. distributed according to Z). (This is because Z outputs a random element of S'_i with probability μ and a random element of $S_i \setminus S'_i$ with probability ν.) Therefore, we have that

$$\Delta\big[U_{\ell'}, \hat{Z}'\big] \leq kp(k)\Delta[U_\ell, Z]$$

which is negligible. Thus, S' is statistically random-set simulatable. □

Note that the majority of groups used in cryptography (e.g. prime-order finite field groups and elliptic curve groups) can be shown to be subgroups of $\mathbb{Z}_p$ or $\mathbb{Z}_p^2$ which satisfy the conditions of Theorem 7.4. Hence, most cryptographic groups are statistically simulatable.

The Cramer–Shoup also requires the use of a target collision resistant hash function:

Definition 7.5 (TCR). Consider a family of hash functions $TCR_k : A_k \to B_k$. We define the TCR advantage of an attacker $\mathcal{A}$ as follows:

$$Adv^{\text{TCR}}_{\mathcal{A}}(k) = \Pr\left[\begin{array}{c} x \neq y \quad \wedge \\ TCR_k(x) = TCR_k(y) \end{array} : \begin{array}{c} x \xleftarrow{\$} A_k \\ y \xleftarrow{\$} \mathcal{A}(1^k, x) \end{array}\right].$$

$Gen(1^k)$:
$w \xleftarrow{\$} \mathbb{Z}_q$; $g_2 \leftarrow g_1^w$
$x, y, z \xleftarrow{\$} \mathbb{Z}_q$
$e \leftarrow g_1^x$; $f \leftarrow g_1^y$
$h \leftarrow g_1^z$
$pk \leftarrow (g_1, g_2, h, e, f)$
$sk \leftarrow (w, x, y, z)$
Output (pk, sk)

$Encap(pk)$:
$r \xleftarrow{\$} \mathbb{Z}_q$
$c_1 \leftarrow g_1^r$; $c_2 \leftarrow g_2^r$
$t \leftarrow TCR(c_1, c_2)$
$\pi \leftarrow e^{rt} f^r$
$C \leftarrow (c_1, c_2, \pi)$
$K \leftarrow h^r$
Output (C, K)

$Decap(C)$:
Parse C as (c_1, c_2, π)
$t \leftarrow TCR(c_1, c_2)$
If $c_2 = c_1^w$ and $\pi = c_1^{xt+y}$
Output c_1^w
Else
Output $\perp$

Fig. 21. The hybrid Cramer–Shoup encryption scheme (CS1b).

A hash function *TCR* is target collision resistant if every PPT attacker $\mathcal{A}$ has negligible advantage.

We will write *TCR* in place of TCR_k if the hash function family is implicit by context.

7.2. *The Hybrid Cramer–Shoup Encryption Scheme*

The hybrid Cramer–Shoup encryption scheme is defined on a group $\mathbb{G}$ of prime order q and with generator g_1, and uses a target collision-resistant hash function $TCR : \mathbb{G}^2 \to \mathbb{Z}_q$. The scheme was famously proven IND-CCA2 secure by Cramer and Shoup in 1998 [10]. We present the scheme as a KEM in Fig. 21. The scheme has a symmetric key-space $\mathbb{G}$ and so has to be used with a DEM with key-space $\mathbb{G}$. However, a DEM can with key space $\mathcal{K}$ can be converted into a DEM with key space $\mathbb{G}$ using a smooth hash function[4] $H : \mathbb{G} \to \mathcal{K}$.

In accordance with the strategy in Sect. 6.3, we prove that this KEM is PA1+ and simulatable. Recall that, according to Remark 6.4, we are only required to show that the KEM is simulatable for attackers which obtain a single challenge ciphertext. In order to evaluate the simulatability of the Cramer–Shoup KEM, we need to introduce a new definition of security (which is very similar to the IND-R notion of security for symmetric encryption schemes).[5] The IND-R security notion for a KEM is defined using the two games given in Fig. 22.

Definition 7.6 (IND-R-CCA2). A KEM $\Pi = (Gen, Encap, Decap)$ is IND-R-CCA2 secure if for every PPT attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ the advantage

$$Adv_{\mathcal{A}}^{\text{IND-R}}(k) = \left|\Pr\left[\text{EXPT}_{\mathcal{A}}^{\text{IND-R-1}} = 1\right] - \Pr\left[\text{EXPT}_{\mathcal{A}}^{\text{IND-R-0}} = 1\right]\right|$$

is negligible.

Cramer and Shoup (essentially) proved the following theorem [10]:

[4] A smooth hash function has the property that the distribution of $H(x)$, for $x \xleftarrow{\$} \mathbb{G}$, is computationally indistinguishable from the uniform distribution on $\mathcal{K}$

[5] The original proof that the Cramer–Shoup encryption scheme is simulatable [12] has a (minor) flaw. The proof does not "reset" the decryption oracle to its correct operation at the end of proof. This presentation sidesteps the issue by using a simplified proof.

$Encap'(pk, m)$:
$C \xleftarrow{\$} \mathcal{C}$
$K \xleftarrow{\$} \mathcal{K}$
Output (C, K)

$\text{EXPT}_{\mathcal{A}}^{\text{IND-R-1}}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$\omega \xleftarrow{\$} \mathcal{A}_1^{Decap(sk,\cdot)}(1^k, pk)$
$(C^*, K^*) \xleftarrow{\$} Encap(pk)$
$b' \xleftarrow{\$} \mathcal{A}_2^{Decap(sk,\cdot)}$
(C^*, K^*, ω)
Return b'

$\text{EXPT}_{\mathcal{A}}^{\text{IND-R-0}}$:
$(pk, sk) \xleftarrow{\$} \mathcal{G}(1^k)$
$\omega \xleftarrow{\$} \mathcal{A}_1^{Decap(sk,\cdot)}(1^k, pk)$
$(C^*, K^*) \xleftarrow{\$} Encap'(pk)$
$b' \xleftarrow{\$} \mathcal{A}_2^{Decap(sk,\cdot)}$
(C^*, K^*, ω)
Return b'

Fig. 22. The IND-R-CCA2 security notion for KEMs. In G_1 and G_2 the attacker $\mathcal{A}_2$ is forbidden from querying the decryption oracle on the ciphertext C^*.

Theorem 7.7 (Cramer–Shoup). *Suppose the DDH problem is hard in the group* $\mathbb{G}$ *and the hash function TCR is target collision resistant. Then the Cramer–Shoup KEM is IND-R-CCA2 secure*

There is a clear relationship between IND-R-CCA2 security and simulatability. A simple hybrid argument can show that the following:

Theorem 7.8. *If a KEM* Π *is IND-R-CCA2 secure and both* $\mathcal{C}$ *and* $\mathcal{K}$ *are computationally simulatable as sets, then* Π *is simulatable* (*as a KEM*).

Thus, we can easily conclude:

Corollary 7.9. *If* $\mathbb{G}$ *is a group on which the DDH problem is hard and which is simulatable as a set, and TCR is target collision resistant, then the Cramer–Shoup KEM is simulatable* (*as a KEM*).

Proof. Since $\mathbb{G}$ is computationally simulatable, we have that both $\mathcal{C} = \mathbb{G}^3$ and $\mathcal{K} = \mathbb{G}$ are computationally simulatable (by Lemma 4.2). Hence, by a combination of Theorem 7.7 and Theorem 7.8, we have that the Cramer–Shoup KEM is simulatable. □

7.3. *The Cramer–Shoup KEM is PA1+ Plaintext Aware*

The proof that the Cramer–Shoup KEM is simulatable is heavily based on the proof that it is IND-CCA2 secure. We introduce new techniques to prove that the Cramer–Shoup KEM is PA1+.[6]

Theorem 7.10. *Suppose* $\mathbb{G}$ *is a statistically simulatable group on which the DHK assumption holds. Then the Cramer–Shoup KEM is PA1+ plaintext aware.*

[6] The original proof of this result claimed that it was sufficient for the group $\mathbb{G}$ to be computationally simulatable. This is incorrect: we require that the group $\mathbb{G}$ is statistically simulatable. This error is relatively minor in practice as all the common groups which are used for cryptography are statistically simulatable (see Sect. 7.1). The original proof also failed to deal effectively with the differences in the way that the DHK extractor $\mathcal{B}^*$ and the PA1+ plaintext extractor $\mathcal{A}^*$ handle the random inputs of their respective underlying algorithms $\mathcal{B}$ and $\mathcal{A}$. The DHK extractor $\mathcal{B}^*$ requires all random bits of $\mathcal{B}$ to be known in advance, while the PA1+ extractor $\mathcal{A}^*$ must allow for the possibility that fresh random bits will be generated by $\mathcal{A}$ after $\mathcal{A}^*$'s execution.

$\mathcal{B}^{\mathcal{O}}(1^k, g, a)$:
$g_1 \leftarrow g$; $g_2 \leftarrow a$
$r_e, r_f, r_h \xleftarrow{\$} \{0,1\}^\ell$
$e \leftarrow Gf(r_e)$; $f \leftarrow Gf(r_f)$
$h \leftarrow Gf(r_h)$
$pk \leftarrow (g_1, g_2, e, f, h)$
$R[\mathcal{A}] \xleftarrow{\$} \{0,1\}^{t(k)}$
$x \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathcal{D}}, \mathcal{O}_{\mathcal{R}}}(1^k, pk)$
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{R}}$
$\rho \xleftarrow{\$} \{0,1\}$
Return ρ
If $\mathcal{A}$ queries $\mathcal{O}_{\mathcal{D}}(C)$
Parse C as (c_1, c_2, π)
Query $r \leftarrow \mathcal{O}(c_1, c_2)$
$t \leftarrow TCR(c_1, c_2)$
If $c_1 = g_1^r$ and $c_2 = g_2^r$
and $\pi = e^{rt} f^r$
Return h^r
Else
Return $\perp$
Output 0

$\mathcal{A}^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$:
On first invocation
$\text{DLIST} \leftarrow \varepsilon$
$n_D \leftarrow 0$
Else
$n_D \leftarrow n_D + 1$
$r_e \xleftarrow{\$} Gf^{-1}(e)$; $r_f \xleftarrow{\$} Gf^{-1}(f)$
$r_h \xleftarrow{\$} Gf^{-1}(h)$
$\text{RLIST}' = \{0,1\}^{q_R - |\text{RLIST}|}$
$R[\mathcal{B}] \leftarrow r_e \| r_f \| r_h \| R[\mathcal{A}] \| \text{RLIST} \| \text{RLIST}'$
Reset $\mathcal{B}^*$ (i.e. delete its internal state)
Append C to DLIST
For $j = 0, 1, \ldots, n_D$
$C_j \leftarrow \text{DLIST}[j]$
Parse C_j as (c_1, c_2, π)
$r_j \xleftarrow{\$} \mathcal{B}^*(g_1, g_2, c_1, c_2, R[\mathcal{B}])$
$t \leftarrow TCR(c_1, c_2)$
If $c_1 = g_1^r$ and $c_2 = g_2^r$ and $\pi = e^{rt} f^r$
$m_j \leftarrow h^r$
Else
$m_j \leftarrow \perp$
Output m_{n_D}

Fig. 23. The DHK algorithm $\mathcal{B}$ and the plaintext extractor $\mathcal{A}^*$. Since $\mathcal{A}$ runs in strict polynomial-time attacker, there exists a polynomial bound $t(k)$ on the running time of $\mathcal{A}$, and so we may assume that $R[\mathcal{A}]$ is $t(k)$-bits long. We also assume that $\mathcal{A}$ makes at most q_R queries to its randomness oracle $\mathcal{O}_{\mathcal{R}}$. Recall that $\mathcal{A}^*$ and $\mathcal{B}^*$ run as stateful algorithms. DLIST contains a list of ciphertexts which have been submitted to the decryption oracle. The counter n_D gives the index for the current ciphertext C in DLIST.

Proof. Let (Gf, Gf^{-1}, ℓ) be a simulator for the group $\mathbb{G}$. Let $\mathcal{A}$ be an arbitrary PPT PA1+ ciphertext creator. We define a DHK attacker $\mathcal{B}$ in Fig. 23. Since the DHK assumption holds on $\mathbb{G}$, there exists a DHK extractor $\mathcal{B}^*$ for $\mathcal{B}$, and we use this to define a plaintext extractor $\mathcal{A}^*$ in Fig. 23. One of the key points about the plaintext extractor $\mathcal{A}^*$ is that for every ciphertext C submitted to the decryption oracle, the plaintext extractor runs the DHK extractor $\mathcal{B}^*$ from scratch over all of the ciphertexts submitted to the decryption oracle up to this point. This is to cope with fact that the plaintext extractor $\mathcal{A}^*$ is forced to guess the results of future randomness oracle queries RLIST$'$; if the DHK extractor $\mathcal{B}^*$ is run as a stateful algorithm in the obvious way (with one execution of $\mathcal{B}^*$ for each execution of $\mathcal{A}^*$) then we cannot argue that $\mathcal{B}^*$'s responses are correct since the list of random coins RLIST$'$ generated in the previous execution is unlikely to be correct. Thus, the proof of this theorem revolves centrally around an argument that $\mathcal{B}^*$'s responses are identical when it is repeatedly run on the same ciphertexts in different executions of the plaintext extractor $\mathcal{A}^*$ (and therefore when it potentially has different internal states).

Let D be an arbitrary polynomial-time distinguisher algorithm and W_i be the event that $D(x)$ outputs 1 in game G_i.

Game G_1: G_1 is the $\text{EXPT}^{\text{FAKE-PA1+}}_{\mathcal{A},\mathcal{A}^*,D}$ game.

Game G_2: Note that the group elements e, f, h are uniformly distributed in $\mathbb{G}$ in G_1. In G_2 these elements are generated by choosing $\tilde{r}_e, \tilde{r}_f, \tilde{r}_h \xleftarrow{\$} \{0,1\}^\ell$ and setting

$\mathcal{A}_4^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$:
Parse C as (c_1, c_2, π)
If there exists no r such that $c_1 = g_1^r$ and $c_2 = g_2^r$
  Return $\perp$
Find r such that $c_1 = g_1^r$ and $c_2 = g_2^r$
$t \leftarrow TCR(c_1, c_2)$
If $\pi = e^{rt} f^r$
  Return h^r
Else
  Return $\perp$

$\mathcal{A}_6^*(1^k, pk, C, R[\mathcal{A}], \text{RLIST})$:
Parse C as (c_1, c_2, π)
If $c_2 \neq c_1^w$
  Return $\perp$
$t \leftarrow TCR(c_1, c_2)$
If $\pi = c_1^{xt+y}$
  Return c_1^z
Else
  Return $\perp$

Fig. 24. The plaintext extractors $\mathcal{A}_4^*$ and $\mathcal{A}_6^*$ used in G_4 and G_6.

$e \leftarrow Gf(\tilde{r}_e)$, $f \leftarrow Gf(\tilde{r}_f)$, and $h \leftarrow Gf(\tilde{r}_h)$. This is a distributional game hopping step (see Appendix A). Let U be the uniform distribution on $\mathbb{G}$ and Y be the distribution of $Gf(\tilde{r})$ for $\tilde{r} \xleftarrow{\$} \{0,1\}^\ell$. So $|\Pr[W_2] - \Pr[W_1]| \leq 3\Delta[U, Y]$ which is negligible by the assumption that $\mathbb{G}$ is statistically simulatable.

Game G_3: Note that in G_2 the plaintext extractor $\mathcal{A}^*$ re-computes the values r_e, r_f, r_h as $Gf^{-1}(Gf(\tilde{r}_e)), Gf^{-1}(Gf(\tilde{r}_f)), Gf^{-1}(Gf(\tilde{r}_h))$. G_3 alters $\mathcal{A}^*$ to use the values $\tilde{r}_e, \tilde{r}_f, \tilde{r}_h$ that were introduced in G_2 as part of the key generation process. This is a distributional game hopping step. Let U be the uniform distribution on $\{0,1\}^\ell$ and let Y be the distribution of $Gf^{-1}(Gf(r))$ for $r \xleftarrow{\$} \{0,1\}^\ell$. So $|\Pr[W_3] - \Pr[W_2]| \leq 3\Delta[U, Y]$ which is negligible by the assumption that $\mathbb{G}$ is statistically simulatable.

Game G_4: G_4 responds to decryption oracle using the plaintext extractor $\mathcal{A}_4^*$ rather than $\mathcal{A}^*$. $\mathcal{A}_4^*$ is almost identical to $\mathcal{A}^*$ except that it no longer uses $\mathcal{B}^*$ but instead uses computes the Diffie–Hellman value r exactly (even though this makes a $\mathcal{A}_4^*$ a non-polynomial-time algorithm). The complete description is given in Fig. 24.

Let E be the event that the decryption oracle returns a different response in G_3 and G_4 and let E_i be the event that this *first* occurs on the ith decryption oracle query. Suppose that the ith decryption oracle query occurs on a ciphertext (c_1, c_2, π). If (g_1, g_2, c_1, c_2) is not a Diffie–Hellman tuple, then the decryption oracle will return $\perp$ in G_3 and G_4. Hence, E_i only occurs if $\mathcal{A}$ submits a ciphertext to the decryption oracle for which (g_1, g_2, c_1, c_2) is a valid Diffie–Hellman tuple but for which $\mathcal{A}^*$ fails to compute a value r_i such that $c_1 = g_1^{r_i}$ and $c_2 = g_2^{r_i}$.

We will bound $\Pr[E_i]$. Consider the ith decryption oracle query. $\mathcal{A}^*$ computes r_i by simulating an execution of the DHK game using $\mathcal{B}$ and $\mathcal{B}^*$ for i DHK oracle queries. This simulation doesn't actually run $\mathcal{B}$ but supplies $\mathcal{B}^*$ with the ciphertexts that $\mathcal{B}$ would have computed using DLIST. Within this execution, let E'_j be the event that $\mathcal{B}^*$ fails to determine the correct value $\tilde{r}_j$ when queried with the jth ciphtertext (i.e. the jth DHK oracle query is a pair (c_1, c_2) such that $c_1 = g_1^r$ and $c_2 = g_2^r$ but $\mathcal{B}^*$ fails to return the correct value of r).

– If any E'_j occurs, with $1 \leq j < i$, then we cannot determine whether $\mathcal{B}^*$ will correctly determine the value r_i that $\mathcal{A}^*$ requires. This is because the internal state of $\mathcal{B}^*$ would be consistent with an execution of $\mathcal{B}$ which computed an incorrect decryption of the jth ciphertext. Hence, the list of ciphertexts DLIST may be inconsistent with the ciphertexts output by the algorithm $\mathcal{B}$ that $\mathcal{A}^*$ is trying to simulate.

(Recall that E_i represents the first time the decryption oracle incorrectly decrypts a ciphertext; hence, all previous ciphertexts must have been decrypted correctly.)
– If E'_i occurs, then the value r_i computed by $\mathcal{A}^*$ is incorrect by the definition of E'_i.

Thus, we can conclude

$$\Pr[E_i] \leq \Pr\left[E'_1 \vee E'_2 \vee \cdots \vee E'_{i-1} \vee E'_i\right].$$

However, by the definition of the DHK game, this value is bounded by $Adv^{\text{DHK}}_{\mathcal{B},\mathcal{B}^*}(k)$. Thus, $\Pr[E_i] \leq Adv^{\text{DHK}}_{\mathcal{A},\mathcal{A}^*}(k)$ and so $|\Pr[W_4] - \Pr[W_3]| \leq \Pr[E] \leq q_D Adv^{\text{DHK}}_{\mathcal{A},\mathcal{A}^*}(k)$ which is negligible.

Game G_5: Game G_5 changes the distribution of e, f, h so that they are generated as $e, f, h \xleftarrow{\$} \mathbb{G}$ rather than by using Gf. This "undoes" the change made in G_2 and by the same argument we have that $|\Pr[W_5] - \Pr[W_4]|$ is negligible since $\mathbb{G}$ is statistically image-set simulatable. (Note that we require *statistical* image-set simulatability here as the extractor is not polynomial-time and so there is no guarantee that computational image-set simulatability would suffice.) We do not have to "undo" the change made in G_3 since $\mathcal{A}^*_4$ does not make use of the elements r_e, r_f, r_h.

Game G_6: G_6 changes the distribution of g_2, e, f, h and uses the plaintext extractor $\mathcal{A}^*_6$ given in Fig. 24. The elements g_2, e, f, h are computed by selecting $w, x, y, z \xleftarrow{\$} \mathbb{Z}_q$ and setting $g_2 \leftarrow g_1^w, e \leftarrow g_1^x, f \leftarrow g_1^y, h \leftarrow g_1^z$. Obviously, this doesn't change their distribution, so we concentrate on showing that the action of $\mathcal{A}^*_6$ is identical to the action of $\mathcal{A}^*_4$ used in G_5. However, this can trivially be seen to be true by noting that $c_2 = c_1^w$ if and only if there exist an r such that $c_1 = g_1^r$ and $c_2 = g_2^r$, and that if $c_1 = g_1^r$ then $e^{rt} f^r = c_1^{xt} c_1^y$ and $h^r = c_1^z$. Thus, $\Pr[W_6] = \Pr[W_5]$.

But an examination of $\mathcal{A}^*_6$ shows that it is functionally identical to the real decryption algorithm $\mathcal{D}(sk, \cdot)$. Thus, the event W_6 is equivalent to the event $\text{EXPT}^{\text{REAL-PA1+}}_{\mathcal{A},D} = 1$. And so, $Adv^{\text{PA1+}}_{\mathcal{A},\text{D}}(k) \leq |\Pr[W_1] - \Pr[W_6]|$ is negligible and so the Cramer–Shoup KEM is PA1+. □

Corollary 7.11. *Suppose* $\mathbb{G}$ *is a statistically simulatable group on which both the DDH problem is hard and the DHK assumption holds, and that TCR is a target collision resistant hash function. Then the Cramer–Shoup KEM is PA*2 *plaintext aware.*

Therefore, we have that the hybrid Cramer–Shoup encryption scheme (with $\mathcal{M} = \{0,1\}^n$) is PA2 plaintext aware. (Recall that the condition that $\mathcal{M} = \{0,1\}^n$ is required in order to prove that the DEM is simulatable.)

8. Conclusions and Open Problems

This paper gave the first proof strategy for proving PA2 plaintext awareness and used this strategy to prove that the hybrid Cramer–Shoup encryption scheme is PA2 plaintext aware. At the heart of this strategy are the notions of simulatability and PA1+ plaintext awareness. Both are strong requirements: a scheme which is simulatable is necessarily IND-CCA2 secure and a proof of PA1+ plaintext awareness seems to require strong extractor assumptions like the Diffie–Hellman Knowledge (DHK) assumption.

Nevertheless, the fact that PA2 plaintext aware encryption schemes can be proven to exist at all is theoretically interesting and the fact that the Cramer–Shoup encryption scheme is PA2 plaintext aware gives some explanation as why the scheme achieves IND-CCA2 security. The approach demonstrated here has been used to show that all Cramer–Shoup hash-proof encryption schemes [7] and Kurosawa–Desmedt hash-proof encryption schemes [7,19] are also plaintext aware (under certain extractor-based assumptions).

There are many open problems in this area. The most important ones are:

- Is it possible to show any other schemes are plaintext aware? All the schemes that are currently known to be PA2 plaintext aware are of the "hash-proof" variety. It would be interesting to see if any other schemes could achieve this level of security.
- Is it possible to use plaintext-aware encryption to reason about protocols and attack strategies? This approach was taken in the work of Di Raimondo, Gennaro and Krawczyk [13] and Ventre and Visconti [30], but it may be useful in other scenarios, e.g. in the analysis of protocols using formal methods.
- Is it possible to develop a strategy for proving PA2 plaintext awareness that does not rely on simulatable encryption schemes? Since one of the major motivations for the study of plaintext-aware encryption is its use in proving IND-CCA2 security, the reliance of the strategy on simulatable encryption schemes, which are already necessarily IND-CCA2 secure, is a disadvantage.
- Is it possible to prove that a scheme is PA1/PA2 plaintext aware without the use of an extractor-based assumption like the DHK assumption? Alternatively, can it be shown that plaintext awareness cannot be proven under any non-interactive assumption?

It is also interesting to examine other ways in which the extraction methodology can be applied. One interesting extension is the concept of secret-key awareness, which moves the knowledge assumption from the ciphertext to the public key, i.e. a scheme is secret-key aware if it is infeasible for any PPT algorithm to create a public key without "knowing" the underlying private key [1]. This has applications to proving complete non-malleability [15] and may have applications towards protocol analysis.

Acknowledgements

The majority of research for this paper was performed while both authors were members of the Information Security Group at Royal Holloway, University of London. The authors would like to thank the referees of Eurocrypt 2006, PKC 2008 and the Journal of Cryptology for their helpful comments. The authors would also like to thank Martijn Stam and Nigel Smart for their detailed and insightful comments.

Appendix A. Game Hopping

Game hopping is a proof technique that involves adapting the environment in which an attacker runs until it is clear that an attacker has a negligible change of "winning". Every time the environment (game) is changed, the proof has to show that the attacker

does not have a significantly greater probability of winning in the second game than in the first—i.e. that the "game hop" does not significantly increase the attacker's chance of winning. Hence, the main game hopping proof techniques involve showing that an attacker's output is practical identical when running in two similar environments. We will often make use of game hopping techniques when working with statistically similar distributions, hence we begin by proving a few simple lemmas:

Lemma A.1. *Let X, Y, Z be distributions over some common finite set S. Then $\Delta[X, Z] \leq \Delta[X, Y] + \Delta[Y, Z]$. Furthermore, if $\mathcal{A}$ is any (possibly probabilistic) algorithm, then $\Delta[\mathcal{A}(X), \mathcal{A}(Y)] \leq \Delta[X, Y]$.*

A proof of these facts can be found in, for example, Shoup [27]. We restate the difference lemma [26] in terms of statistical distance:

Lemma A.2. *Suppose that X is the random variable describing the output of a randomised algorithm $\mathcal{A}$ (on a finite set). Let E be some event that may occur during $\mathcal{A}$'s execution. Let Y be the random variable which describes the output of $\mathcal{A}$ given that E does not occur (i.e. $\Pr[Y = s] = \Pr[X = s \mid \neg E]$). Then $\Delta[X, Y] \leq \Pr[E]$.*

Proof. The result follows from a simple probability calculation:

$$\begin{aligned}
\Delta[X, Y] &= \frac{1}{2} \sum_{s \in S} \big|\Pr[X = s] - \Pr[Y = s]\big| \\
&= \frac{1}{2} \sum_{s \in S} \big|\Pr[X = s \mid E] \Pr[E] + \Pr[X = s \mid \neg E] \Pr[\neg E] - \Pr[Y = s]\big| \\
&= \frac{1}{2} \sum_{s \in S} \big|\Pr[X = s \mid E] \Pr[E] + \Pr[Y = s]\big(1 - \Pr[E]\big) - \Pr[Y = s]\big| \\
&= \frac{1}{2} \sum_{s \in S} \big|\Pr[X = s \mid E] \Pr[E] - \Pr[Y = s] \Pr[E]\big| \\
&\leq \frac{1}{2} \Pr[E] \sum_{s \in S} \Pr[X = s \mid E] + \frac{1}{2} \Pr[E] \sum_{s \in S} \Pr[Y = s] \\
&\leq \Pr[E].
\end{aligned}$$

□

The computational distance between families of distributions X_k and Y_k for an algorithm $\mathcal{A}$ is

$$dist_{\mathcal{A}}^{X,Y}(k) = \big|\Pr\big[\mathcal{A}(s) = 1 : s \xleftarrow{\$} X_k\big] - \Pr\big[\mathcal{A}(s) = 1 : s \xleftarrow{\$} Y_k\big]\big|.$$

X_k and Y_k are computationally indistinguishable if $dist_{\mathcal{A}}^{X,Y}(k)$ is negligible for all probabilistic polynomial-time algorithms $\mathcal{A}$.

A game G describes the distributions of a series of inputs to an algorithm $\mathcal{A}$ and a "win condition" for that algorithm. Let W_i denote the event that $\mathcal{A}$ achieves the win

condition. A game hop is relationship between the probability that $\mathcal{A}$ wins game G_0 and game G_1. We make use of three type of game hop:

1. *Bridging Steps*: The distribution of inputs to $\mathcal{A}$ in G_0 and G_1 are identical (although the inputs may be generated in a different way). Thus, $\Pr[W_0] = \Pr[W_1]$.
2. *Distributional Steps*: If the only difference between G_0 and G_1 is that an input s to $\mathcal{A}$ is drawn from distribution X in G_0 and from distribution Y in G_1, then

$$\left|\Pr[W_0] - \Pr[W_1]\right| \leq \Delta[X, Y].$$

   Furthermore, if $\mathcal{A}$ is a probabilistic, polynomial-time (PPT) algorithm, and W_0 and W_1 are polynomial-time decidable events, then

$$\left|\Pr[W_0] - \Pr[W_1]\right| \leq dist_{\mathcal{A}}^{X,Y}(k).$$

3. *Small Error Steps*: If G_0 and G_1 proceed identically unless some event E occurs, then

$$\left|\Pr[W_0] - \Pr[W_1]\right| \leq \Pr[E].$$

The reader is referred to Bellare and Rogaway [4] and Shoup [26] for more information.

References

[1] M. Barbosa, P. Farshim, Strong knowledge extractors for public-key encryption schemes, in *Australasian Conference on Information Security and Privacy—ACISP 2010*, ed. by P. Hawkes, R. Steinfeld. Lecture Notes in Computer Science, vol. 6168 (Springer, Berlin, 2010), pp. 164–181

[2] M. Bellare, A. Palacio, Towards plaintext-aware public-key encryption without random oracles, in *Advances in Cryptology—Asiacrypt 2004*, ed. by P.J. Lee. Lecture Notes in Computer Science, vol. 3329 (Springer, Berlin, 2004), pp. 48–62

[3] M. Bellare, P. Rogaway, Optimal asymmetric encryption, in *Advances in Cryptology—Eurocrypt '94*, ed. by A. De Santis. Lecture Notes in Computer Science, vol. 950 (Springer, Berlin, 1994), pp. 92–111

[4] M. Bellare, P. Rogaway, The security of triple encryption and a framework for code-based game-playing proofs, in *Advances in Cryptology—Eurocrypt 2006*, ed. by S. Vaudenay. Lecture Notes in Computer Science, vol. 4004 (Springer, Berlin, 2006), pp. 409–426

[5] M. Bellare, A. Desai, E. Jokipii, P. Rogaway, A concrete security treatment of symmetric encryption, in *Foundations of Computer Science—FOCS 1997* (IEEE Computer Society, Los Alamitos, 1997), pp. 394–403

[6] M. Bellare, A. Desai, D. Pointcheval, P. Rogaway, Relations among notions of security for public-key encryption schemes, in *Advances in Cryptology—Crypto 1998*, ed. by H. Krawczyk. Lecture Notes in Computer Science, vol. 1462 (Springer, Berlin, 1998), pp. 26–45

[7] J. Birkett, On plaintext-aware public-key encryption schemes. PhD thesis, Royal Holloway, University of London, 2010

[8] J. Birkett, A.W. Dent, Relations among notions of plaintext awareness, in *Public Key Cryptography—PKC 2008*, ed. by R. Cramer. Lecture Notes in Computer Science, vol. 4939 (Springer, Berlin, 2008), pp. 47–65

[9] J.L. Carter, M.N. Wegman, New classes and applications of hash functions, in *Foundations of Computer Science—FOCS 1979* (IEEE Computer Society, Los Alamitos, 1979), pp. 175–182

[10] R. Cramer, V. Shoup, Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.* **33**(1), 167–226 (2004)

[11] I.B. Damgård, Towards practical public key systems secure against chosen ciphertext attacks, in *Advances in Cryptology—Crypto '91*, ed. by J. Feigenbaum. Lecture Notes in Computer Science, vol. 576 (Springer, Berlin, 1991), pp. 445–456

[12] A.W. Dent, The Cramer-Shoup encryption scheme is plaintext aware in the standard model, in *Advances in Cryptology—Eurocrypt 2006*, ed. by S. Vaudenay. Lecture Notes in Computer Science, vol. 4004 (Springer, Berlin, 2006), pp. 289–307

[13] M. Di Raimondo, R. Gennaro, H. Krawczyk, Deniable authentication and key exchange, in *ACM Conference on Computer and Communications Security—ACM CCS '06* (ACM, New York, 2006), pp. 400–409

[14] U. Feige, A. Fiat, A. Shamir, Zero-knowledge proofs of identity. *J. Cryptol.* **1**(2), 77–94 (1988)

[15] M. Fischlin, Completely non-malleable schemes, in *Automata, Languages and Programming—ICALP 2005*, ed. by L. Caires, G.F. Italiano, L. Monteiro, C. Palamidessi, M. Yung. Lecture Notes in Computer Science, vol. 3580 (Springer, Berlin, 2005), pp. 779–790

[16] S. Goldwasser, S. Micali, Probabilistic encryption. *J. Comput. Syst. Sci.* **28**, 270–299 (1984)

[17] S. Goldwasser, S. Micali, C. Rackoff, The knowledge complexity of interactive proof systems. *SIAM J. Comput.* **18**(1), 186–208 (1989)

[18] J. Håstad, R. Impagliazzo, L.A. Levin, M. Luby, A pseudorandom generator from any one-way function. *SIAM J. Comput.* **18**(1), 1364–1396 (1999)

[19] S. Jiang, H. Wang, Plaintext-awareness of hybrid encryption, in *Topics in Cryptology—CT-RSA 2010*, ed. by J. Pieprzyk. Lecture Notes in Computer Science, vol. 5985 (Springer, Berlin, 2010), pp. 57–72

[20] K. Kurosawa, Y. Desmedt, A new paradigm of hybrid encryption scheme, in *Advances in Cryptology—Crypto 2004*, ed. by M. Franklin. Lecture Notes in Computer Science, vol. 3152 (Springer, Berlin, 2004), pp. 426–442

[21] M. Naor, M. Yung, Public-key cryptosystems provably secure against chosen ciphertext attacks, in *Proc. 22nd ACM Symposium on the Theory of Computing—STOC '90* (ACM, New York, 1990), pp. 427–437

[22] E. Petrank, C. Rackoff, CBC MAC for real-time data sources. *J. Cryptol.* **13**(3), 315–339 (2000)

[23] C. Rackoff, D. Simon, Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack, in *Advances in Cryptology—Crypto '91*, ed. by J. Feigenbaum. Lecture Notes in Computer Science, vol. 576 (Springer, Berlin, 1991), pp. 434–444

[24] P. Rogaway, M. Bellare, J. Black, T. Krovetz, OCB: a block-cipher mode of operation for efficient authenticated encryption, in *ACM Conference on Computer and Communications Security—ACM CCS '01* (ACM, New York, 2001), pp. 196–205

[25] V. Shoup, Using hash functions as a hedge against chosen ciphertext attack, in *Advances in Cryptology—Eurocrypt 2000*, ed. by B. Preneel. Lecture Notes in Computer Science, vol. 1807 (Springer, Berlin, 2000), pp. 275–288

[26] V. Shoup, Sequences of games: a tool for taming complexity in security proofs. Available from http://eprint.iacr.org/2004/332, 2004

[27] V. Shoup, *A Computational Introduction to Number Theory and Algebra* (Cambridge University Press, Cambridge, 2005)

[28] M. Stam, Personal e-mail correspondence, 2005

[29] I. Teranishi, W. Ogata, Relationship between standard model plaintext awareness and message hiding, in *Advances in Cryptology—Asiacrypt 2006*, ed. by X. Lai, K. Chen. Lecture Notes in Computer Science, vol. 4284 (Springer, Berlin, 2006), pp. 226–240

[30] C. Ventre, I. Visconti, 2-round extractable commitments: definitions, constructions and applications in the plain model. Unpublished manuscript, 2010