Blackbox Construction of a More Than Non-Malleable CCA1 Encryption Scheme from Plaintext Awareness

Steven Myers[1], Mona Sergi[2], and abhi shelat[2]

[1] Indiana University, Bloomington IN 47408, USA
[2] University of Virginia, Charlottesville VA 22904, USA

Abstract. We construct an NM-CCA1 encryption scheme from any CCA1 encryption scheme that is also plaintext aware and weakly simulatable. We believe this is the first construction of a NM-CCA1 scheme that follows strictly from encryption schemes with seemingly weaker or incomparable security definitions to NM-CCA1.

Previously, the statistical PA1 notion of plaintext awareness was only known to imply CCA1. Our result is therefore novel because unlike the case of CPA and CCA2, it is unknown whether a CCA1 scheme can be transformed into an NM-CCA1 scheme. Additionally, we show both the Damgård Elgamal Scheme (DEG) [Dam91] and the Cramer-Shoup Lite Scheme (CS-Lite) [CS03] are weakly simulatable under the DDH assumption. Since both are known to be statistical PA1 under the Diffie-Hellman Knowledge (DHK) assumption, they instantiate our scheme securely.

Next, in a partial response to a question posed by Matsuda and Matsuura [MM11], we define an extended version of the NM-CCA1, cNM-CCA1, in which the security definition is modified so that the adversary is permuted to ask a $c \geq 1$ number of parallel queries after receiving the challenge ciphertext. We extend our construction to yield a cNM-CCA1 scheme for any constant c. All of our constructions are black-box.

Keywords: Public-Key Encryption, Plaintext-Awareness, Non-Malleability.

1 Introduction

The standard security definition of an encryption scheme does not prevent an adversary who observes an encryption of the message m from producing an encryption of the message $f(m)$ for some function f (even though the value m remains private). The seminal work of Dolev, Dwork, and Naor [DDN03] addressed this security issue by introducing the area of non-malleable cryptographic primitives such as encryption schemes, commitment schemes, and zero-knowledge. Later, Pass, shelat and Vaikuntanathan [PSV06] strengthened the DDN definition and presented a construction from CPA to non-malleable CPA using non-blackbox use of the original encryption scheme. There have been many follow-up works that propose more efficient constructions of non-malleable primitives. A

I. Visconti and R. De Prisco (Eds.): SCN 2012, LNCS 7485, pp. 149–165, 2012.

notable achievement in this line of research has been the construction of non-malleable primitives using only black-box access to the standard version of the same primitive [CDSMW08, PW09, Wee10]. In particular, [CDSMW08] show how non-malleable CPA encryption can be constructed from standard versions of encryption in a black-box manner.

However, the question of whether an NM-CCA1 encryption scheme can be constructed from a CCA1 encryption scheme has remained open. This blemish on our understanding of the theory of encryption has remained despite multiple advances including many novel techniques for constructing encryption schemes. In this work, we present a black-box construction of an NM-CCA1 encryption scheme for a subset of CCA1 encryption schemes, namely those which are also plaintext aware under multiple keys and weakly simulatable (we will formally define these concepts later). Intuitively, an encryption scheme is plaintext aware (called **sPA1** in [BP04]) if the only way that a p.p.t. adversary can produce a valid ciphertext is to apply the (randomized) encryption algorithm to the public key and a message [BP04]. Notice that this definition does not imply non-malleability since there is no guarantee of what an adversary can do *when given a valid ciphertext.* In fact, both encryption schemes from [BP04] are multiplicatively homomorphic. The weakly simulatable property in our construction is required for technical reasons and roughly corresponds to the ability to to sample ciphertexts and pseudo-ciphertexts with random coins used to generate them.

Note that there exist encryption schemes that satisfy security notions that "sit between" standard notions. One such example from Cramer et al. [CHH$^+$07] consists of a black-box construction of a q-bounded CCA2 encryption scheme which is not NM-CPA, but which satisfies a stronger security notion than CPA. In particular, as a generalization of NM-CPA, Matsuda and Matsuura [MM11] put forth the challenge of constructing encryption schemes that can handle more than one parallel query after revealing the challenge ciphertext. They write:

> Since any (unbounded) CCA secure PKE construction from IND-CPA secure ones must first be secure against adversaries who make two or more parallel decryption queries, we believe that overcoming this barrier of two parallel queries is worth tackling.

In this spirit, we define an extension over NM-CCA1, cNM-CCA1, that is defined identically to NM-CCA1 except that the adversary can make c adaptive parallel decryption queries after seeing the challenge ciphertext, where each parallel decryption query can request that a polynomial number of ciphertexts be decrypted (excluding the challenge ciphertext). (Note that NM-CCA1 is cNM-CCA1 where the parameter c is set to be one.) Then we show how to construct a cNM-CCA1 secure encryption scheme for an arbitrary constant c. Unfortunately, the size of the ciphertext in a cNM-CCA1 encryption scheme is polynomially bigger than the size of the ciphertext in a (c-1)NM-CCA1 encryption scheme and thus the parameter c must be a constant to obtain an efficient construction.

About Knowledge Extraction Assumptions. Our constructions rely on encryption schemes that are plaintext aware ($\mathbf{sPA1}_\ell$) in the multi-key setup and are weakly

simulatable. In Appendix A.1, we show that such encryption schemes exist under a suitable extension of the Diffie-Hellman Knowledge (DHK) assumption that was originally proposed by Damgård, and modified to permit interactive extractors by Bellare and Palacio [BP04]. Dent [Den06b] has since shown that it is secure in the generic group model. We understand that there are some critics of the DHK assumption, due to its strength and the fact that it is not efficiently falsifiable. However, it is not our goal to argue whether or not it is an assumption which should be used in deployable systems. Instead we note it is seemingly a weaker assumption than the Random Oracle model (which is known to be incorrect in full generality, cf. [CGH04]), under which it is relatively easy to show that simple IND-CPA secure encryption schemes imply CCA2 secure ones. In contradistinction, there are no security definitions that seem weaker or incomparable to NM-CCA1 that are known to imply schemes which are NM-CCA1. Similarly, the gap between NM-CCA1 and CCA2 is poorly understood.

Techniques. Similar to the nested encryption construction in [MS09], both our NM-CCA1 and cNM-CCA1 constructions are based on the notion of double encryption. We first encrypt the message under one key (we refer to this ciphertext as the "inner layer"), and encrypt the resulting inner layer ciphertext repetitively under an additional k keys, where k is the security parameter (we refer to these k keys as the "outer keys", and the ciphertexts they produce as the "outer layer"). During decryption, all the outer layer ciphertexts are decrypted, and it is verified they all encode the same inner layer value. This is combined with the well studied notion of non-duplicatable set selection (in this case of public-keys used to encrypt the outer-layer encryptions), such that anyone attempting to maul a ciphertext has to perform their own independent outer layer encryption. Intuitively, anyone that can encrypt to a consistent outer layer encryption under a new key must have knowledge of the underlying inner-layer, and thus a valid ciphertext is not mauled.

On a more technical level, there are several challenges that need to be overcome. The traditional technical difficulty in proving weaker public-key encryption security notions imply stronger security notions is in showing how to simulate the decryption oracle. When beginning with a **sPA1**-secure encryption primitive, we can easily simulate the initial decryption oracle in the NM-CCA1 security definition, which is present before the challenge ciphertext is presented, by using the extractor guaranteed by the **sPA1** security definition. However, we cannot simply use the extractor to simulate the decryption oracle after receiving the challenge ciphertext in the NM-CCA1 security experiment. This is because the plaintext aware security does not guarantee that an extractor could decrypt ciphertexts where the underlying randomness is not known to the party that created the ciphertext. Generally, a party that mauls a ciphertext as in the case of non-malleability will not have access to this underlying randomness. To overcome this problem, we make use of a weak notion of simulatability.

To summarize, our contribution is twofold. Firstly, our work shows the first black-box construction of a non-malleable CCA1 encryption scheme in the standard model that is not CCA2 secure. Secondly, for the first time, we show how

to construct an encryption scheme that is not CCA2 secure but is secure against an adversary that can ask a bounded number of polynomial-parallel queries after receiving the challenge ciphertext, satisfying a natural extension to the notion of NM-CCA1 security. This might be of independent interest since the development of constructions that satisfy stronger notions than non-malleable CCA1 security but do not satisfy CCA2 security can provide insight in trying to understand the technical difficulties in understanding the larger relationship between CCA1 and CCA2.

2 Notations and Definitions

We use $[n]$ to denote the set $\{1, 2, \cdots, n\}$. We say a function $\mu : \mathbb{N} \to \mathbb{R}$ is negligible if for all polynomials p and all sufficiently large $n : \mu(n) \leq 1/p(n)$. Given two families of distributions $D_0 = \{D_{0,i}\}_{i\in\mathbb{N}}$ and $D_1 = \{D_{1,i}\}_{i\in\mathbb{N}}$, we denote that they are computationally indistinguishable by writing $D_0 \approx_c D_1$. For any vector v, we use $|v|$ to denote the number of elements in v.

Although non-malleability can be defined for any CPA, CCA1 or CCA2 secure encryption scheme (we use the standard definition for CPA/CCA1/CCA2 security),we only use and hence only define non-malleability for CCA1 secure encryption schemes. We use a definition similar to the non-malleability definition for CPA secure encryption schemes in [PSV06].

Definition 1 (NM-CCA1). *We say that* $\boldsymbol{E} = (\mathsf{nmg}, \mathsf{nme}, \mathsf{nmd})$ *is non-malleable* CCA1 *secure if for all p.p.t. adversaries and p.p.t. distinguishers* $\mathcal{A}$ *and* $\mathcal{D}$ *respectively and for all polynomials* $p(\cdot)$*, we have that* $\{\mathsf{NME}_0(\boldsymbol{E}, \mathcal{A}, \mathcal{D}, k, p(k))\}_k \approx_c \{\mathsf{NME}_1(\boldsymbol{E}, \mathcal{A}, \mathcal{D}, k, p(k))\}_k$ *where experiment* NME *is defined as follows:*

$\mathsf{NME}_b(\mathbf{E}, \mathcal{A}, \mathcal{D}, k, p(k))$
(1) $(\text{NPK}, \text{NSK}) \leftarrow \mathsf{nmg}(1^k)$
(2) $(m_0, m_1, S_1) \leftarrow \mathcal{A}_1^{\mathsf{nmd}_{\text{NSK}}}(\text{NPK})$ s.t. $|m_0| = |m_1|$
(3) $y \leftarrow \mathsf{nme}(\text{NPK}, m_b)$
(4) $(\boldsymbol{c}, S_2) \leftarrow \mathcal{A}_2(y, S_1)$ where $|\boldsymbol{c}| = p(k)$
(5) Output $\mathcal{D}(\boldsymbol{d}, S_2)$ where $d_i \leftarrow \mathsf{nmd}(\text{NSK}, c_i)$ if $c_i \neq y$ and $d_i \leftarrow \perp$ if $c_i = y$

2.1 Plaintext Awareness for Multiple Key Setup

We present a slight generalization to the definition of **sPA1** by [BP04] in which multiple keys are permitted to be constructed and given to the ciphertext creator, and the extractor must be able to decrypt relative to all of the keys. Notice that the **sPA1** definition is a special case of $\mathbf{sPA1}_\ell$ where $\ell(k) = 1$.

$\mathbf{sPA1}_\ell(\mathbf{E}, \mathbf{C}, \mathbf{C}^*, k)$
(1) Let $R[\mathbf{C}]$, $R[\mathbf{C}^*]$ be randomly chosen bit strings for $\mathbf{C}$ and $\mathbf{C}^*$.
(2) $((pk_i, sk_i))_{i\in[\ell(k)]} \leftarrow \mathsf{gen}(1^k)$
(3) $st \leftarrow \left((pk_i)_{i\in[\ell(k)]}, R[C]\right)$
(4) $\mathbf{C}^{\mathbf{C}^*(st,.)}\left((pk_i)_{i\in[\ell(k)]}\right)$
(5) Let $Q = \{(q_i = (pk_{j_i}, c_i), m_i)\}$ be the set of queries $\mathbf{C}$ made to $\mathbf{C}^*$ until it halted and $\mathbf{C}^*$'s responses to them. Return $\wedge_{i=1}^{|Q|}(m_i = \mathsf{dec}_{sk_{j_i}}(c_i))$.

In the above experiment, $\mathbf{C}$ is a ciphertext creator, and $\mathbf{C}^*$ is a stateful p.p.t. algorithm called the *extractor* that takes as input the state information st and a ciphertext given by the ciphertext creator $\mathbf{C}$, and will return the decryption of that ciphertext and the updated state st. The state information st is initially set to the public key pk and the adversary $\mathbf{C}$'s random coins. It gets updated by $\mathbf{C}^*$ as $\mathbf{C}^*$ answers each query that the adversary $\mathbf{C}$ submits. The above experiment returns 1 if all the extractor's answers to queries are the true decryption of those queries under sk. Otherwise, the experiment returns 0.

Definition 2 ($\mathbf{sPA1}_\ell$). *Let ℓ be a polynomial. Let $\boldsymbol{E} = (\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ be an asymmetric encryption scheme. Let the ciphertext-creator adversary $\boldsymbol{C}$ and the extractor $\boldsymbol{C}^*$ be p.p.t. algorithms. For $k \in \mathbb{N}$, the* **sPA1**-advantage *of $\boldsymbol{C}$ relative to $\boldsymbol{C}^*$ is defined as:*

$$\boldsymbol{Adv}^{\boldsymbol{sPA1}_\ell}(\boldsymbol{E}, \boldsymbol{C}, \boldsymbol{C}^*, k) = \Pr[\boldsymbol{sPA1}_\ell(\boldsymbol{E}, \boldsymbol{C}, \boldsymbol{C}^*, k) = 0]$$

The extractor $\boldsymbol{C}^*$ is a successful $\mathbf{sPA1}_\ell$-extractor *for the ciphertext-creator adversary $\boldsymbol{C}$ if for all $k \in \mathbb{N}$, the function $\boldsymbol{Adv}^{\boldsymbol{sPA1}_\ell}(\boldsymbol{E}, \boldsymbol{C}, \boldsymbol{C}^*, k)$ is negligible. The encryption scheme $\boldsymbol{E}$ is called* $\mathbf{sPA1}_\ell$ multi-key secure *if for any p.p.t. ciphertext creator there exists a successful $\boldsymbol{sPA1}_\ell$-extractor.*

We provide further discussion on the relationship between $\mathbf{sPA1}_\ell$ security and **sPA1** security in the full version of this paper. In Appendix A.1, we show that both the Damgård Elgamal encryption scheme (DEG) and the lite version of Cramer-Shoup encryption scheme (CS-lite) are $\mathbf{sPA1}_\ell$ secure under a suitable generalization of the DHK1 assumption.

2.2 Weakly Simulatable Encryption Scheme

Dent in [Den06a] introduced the notion of simulatability for an encryption scheme. Intuitively, an encryption scheme is simulatable if no attacker can distinguish valid ciphertexts from some family of pseudo-ciphertexts (which will include both valid encryptions and invalid encryptions). This family of pseudo-ciphertexts must be efficiently and publicly computable (i.e. without access to any private knowledge, say related to the secret-key), and somewhat invertible (given a pseudo-ciphertext,

one can find a random looking string that generates it). In Dent's definition, the attacker also has access to a decryption oracle to help it distinguish between pseudo-ciphertexts and legitimate ones, but it cannot query the decryption oracle on the challenges that it is trying to distinguish.

For our purposes, consider a restricted notion of simulatability where the attacker is not given access to the decryption oracle. If an encryption scheme satisfies this weaker notion of simulatability, we say it is weakly simulatable.

Definition 3. *(**Weakly Simulatable Encryption Scheme**) An asymmetric encryption scheme* $(\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ *is weakly simulatable if there exist two poly-time algorithms* (f, f^{-1})*, where* f *is deterministic and* f^{-1} *is probabilistic, such that for all* $k \in \mathbb{N}$ *there exists the polynomial function* $p(.)$ *where* $l = p(k)$*, we have the following correctness properties:*

1. *f on inputs of public key pk (in the range of gen) and a random string $r \in \{0,1\}^l$, returns elements in $\mathcal{C}$, where $\mathcal{C}$ is the set of all possible "cipher text"-strings that can be submitted to the decryption oracle (notice that $\mathcal{C}$ might not be a valid ciphertext).*
2. *f^{-1} on input of a public key pk (in the range of gen) and an element $C \in \mathcal{C}$, outputs elements of $\{0,1\}^l$.*
3. *$f(pk, f^{-1}(pk, C)) = C$ for all $C \in \mathcal{C}$.*

And the following security properties. No polynomial time attacker $\mathcal{A}$ has probability better than $1/2 + \mu(k)$ of winning in the following experiment, where μ is some negligible function.

1. *The challenger generates a random key pair $(pk, sk) \leftarrow \mathsf{gen}(1^k)$, and chooses randomly $b \in \{0,1\}$.*
2. *The attacker $\mathcal{A}$ executes on the input 1^k and the public key pk outputs $m \in \mathcal{M}$. The challenger sends $\mathcal{A}$ the pair $(f^{-1}(pk, c = \mathsf{enc}_{pk}(m)), c)$ if $b = 0$, or $(r, f(pk, r))$ for some randomly generated element $r \in \{0,1\}^l$ if $b = 1$. The attacker $\mathcal{A}$ terminates by outputting a guess b' for b.*
   $\mathcal{A}$ wins if $b = b'$ and its advantage is defined in the usual way.

In a scheme where you cannot distinguish legitimate ciphertexts from pseudo-ciphertexts that need not encode actual messages, CPA security is immediate. The converse need not hold, as ciphertexts might be hard to generate, and invalid ciphertexts might be easily distinguishable from illegitimate ones (for example, they might contain a zero-knowledge proof of validity). Notice that the weak simulatability notion is not equivalent to the Invertible Sampling notion introduced in [DN00] since the plaintext is not needed to compute the random looking string that generates the ciphertext.

Theorem 1. *If $\boldsymbol{E}$ is a weakly simulatable encryption scheme, then $\boldsymbol{E}$ is* CPA *secure.*

Proof. See the full version. □

Following the ideas of Dent, in the full version, we show how DEG and CS-lite schemes can both be weakly simulatable when instantiated in proper groups.

2.3 A Note on PA1$^+$

Dent [Den06a] also investigated an augmented notion of plaintext awareness in which he provides the ciphertext creator access to an oracle that produces random bits, PA1$^+$. The extractor receives the answers to any queries generated by the creator, but only at the time these queries are issued. The point of this oracle in the context of a plaintext awareness definition is to model the fact that the extractor might not receive all of the random coins used by the creator *at the beginning* of the experiment. Much in the spirit of "adaptive soundness" and "adaptive zero-knowledge", this oracle requires the extractor to work even when it receives the random coins at the same time as the ciphertext creator. Therefore, the extractor potentially needs to be able to extract some ciphertexts independent of future randomness. This modification has implications when the notion of plaintext awareness is computational—as in the case of Dent's work. However, in our case, we require statistical plaintext awareness, and as we argue below, allowing access to such an oracle does not affect the $\mathbf{sPA1}_\ell$ security.

We claim that any encryption scheme that is $\mathbf{sPA1}_\ell$ secure is also $\mathbf{sPA1}_\ell^+$ secure.

Definition 4. *Define the* $\boldsymbol{sPA1}_\ell^+$ *experiment in a similar way to the* $\boldsymbol{sPA1}_\ell$ *experiment.* The only difference *between the two is that during the* $\boldsymbol{sPA1}_\ell^+$ *experiment, the ciphertext creator has access to a random oracle* $\mathcal{O}$ *that* takes no input*, but returns independent uniform random strings upon each access. Any time the creator access the oracle, the oracle's response is forwarded to both the creator and extractor.*

If an encryption scheme would be deemed $\boldsymbol{sPA1}_\ell$ *secure, when we replace the* $\boldsymbol{sPA1}_\ell$ *experiment in the definition with the modified* $\boldsymbol{sPA1}_\ell^+$ *experiment, then the encryption scheme is said to be* $\boldsymbol{sPA1}_\ell^+$ *secure.*

Lemma 1. *If an encryption scheme* Π *is* $\boldsymbol{sPA1}_\ell$ *secure, then it is* $\boldsymbol{sPA1}_\ell^+$ *secure.*

Proof. See the full version. □

3 The Construction

Let $E = (\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ be any encryption scheme that is weakly simulatable and $\mathbf{sPA1}_\ell$ secure. Then we construct the encryption scheme Π represented in Fig. 1 that is a non-malleable CCA1 encryption scheme. Let $\Sigma = (\mathsf{GenKey}, \mathsf{Sign}, \mathsf{Verify})$ be a strong one-time signature scheme,[1] such that on security parameter k the verification keys that are constructed have length k.

As a first step, we define an encryption scheme $E' = (\mathsf{gen}', \mathsf{enc}', \mathsf{dec}')$ in which one encrypts the encryption of a message k times with k independently chosen public keys. More specifically:

[1] A strong one-time signature is a one-time signature where it is not even possible for an adversary to find an alternate signature to an already signed message.

— $\mathsf{gen}'(1^k)$: For $i \in [0,k]$, run $(pk_i, sk_i) \leftarrow \mathsf{gen}(1^k)$. Set the public and secret keys as $pk \stackrel{def}{=} (pk_0, pk_1, \ldots, pk_k)$ and $sk \stackrel{def}{=} (sk_0, sk_1, \ldots, sk_k)$
— $\mathsf{enc}'_{pk=pk_0,..,pk_k}(m)$: Output $[\mathsf{enc}_{pk_1}(\mathsf{enc}_{pk_0}(m; r_0); r_1), \ldots \mathsf{enc}_{pk_k}(\mathsf{enc}_{pk_0}(m; r_0); r_k)]$ using independently chosen coins r_i.
— $\mathsf{dec}'_{sk=sk_0,\ldots,sk_k}([c_1, c_2, \ldots, c_k])$: Compute $c'_i = \mathsf{dec}_{sk_i}(c_i)$. If all c'_i are not equal, output $\perp$, else output $\mathsf{dec}_{sk_0}(c'_1)$.

We are now ready to present our main construction Π defined in Fig. 1.

NMGen(1^k)
(1) $(pk_0, sk_0) \leftarrow \mathsf{gen}(1^k)$; $(pk_i^b, sk_i^b) \leftarrow \mathsf{gen}(1^k)$, $\forall i \in [k]$ and $b \in \{0,1\}$
(2) Output NPK $= \{pk, pk_0\}$ and NSK $= \{sk, sk_0\}$ where $pk = \{(pk_i^0, pk_i^1)\}_{i\in[k]}$ and $sk = \{(sk_i^0, sk_i^1)\}_{i\in[k]}$

NMEnc(NPK $= (pk, pk_0), m$)
(1) $(\mathsf{SigSK}, \mathsf{SigVK}) \leftarrow \mathsf{GenKey}(1^k)$
(2) $c \leftarrow \mathsf{enc}'_{pk_0, pk_1^{\mathsf{SigVK}_1}, \ldots, pk_k^{\mathsf{SigVK}_k}}(m)$
(3) $\sigma \leftarrow \mathsf{Sign}_{\mathsf{SigSK}}(c)$.
(4) Output $(c, \mathsf{SigVK}, \sigma)$

NMDec(NSK $= (sk, sk_0), C = (c, \mathsf{SigVK}, \sigma)$))
(1) **if** $\mathsf{Verify}_{\mathsf{SigVK}}(\sigma, c) = 0$ **then** Output $\perp$
(2) Output $\mathsf{dec}'_{sk_0, sk_1^{\mathsf{SigVK}_1}, \ldots, sk_k^{\mathsf{SigVK}_k}}(c_1)$

Fig. 1. The Non-malleable CCA1 Encryption Scheme Π

Lemma 2. *If* $\boldsymbol{E} = (\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ *is weakly simulatable, then* $\boldsymbol{E}' = (\mathsf{gen}', \mathsf{enc}', \mathsf{dec}')$ *is weakly simulatable as well.*

Proof. Via a standard hybrid argument. □

Theorem 2. *If* $\boldsymbol{E} = (\mathsf{gen}, \mathsf{enc}, \mathsf{dec})$ *is an encryption scheme that is weakly simulatable and also* $\boldsymbol{sPA1}_{\ell(k)=2k+1}$ *secure where* k *is the security parameter, then the encryption scheme* Π *as described in Fig. 1 is a non-malleable* CCA1 *encryption scheme.*

Proof. Recall that Lemma 1 shows that if $\mathbf{E}$ is $\mathbf{sPA1}_\ell$ secure, then it is also $\mathbf{sPA1}_\ell^+$ secure. In what follows, the $\mathbf{sPA1}_\ell^+$ ciphertext creator adversaries always have access to an oracle $\mathcal{O}$ that produces random strings upon access.

To prove that Π is a non-malleable CCA1 encryption scheme, we need to show that for any p.p.t. adversary $\mathcal{A}$ and p.p.t. distinguisher $\mathcal{D}$ and for all polynomials $p(k)$,

$$\{\mathsf{NME}_0(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k\in\mathbb{N}} \approx_c \{\mathsf{NME}_1(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k\in\mathbb{N}}$$

We show this by a hybrid argument. Consider the following experiments:

Experiment $\mathsf{NME}_b{}^{(1)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ modifies NME_b in two ways. First, instead of selecting vkSig^* when the challenge ciphertext is encrypted, choose this value as the first step of the experiment. Second, when processing decryption queries during the experiment, replace Verify with Verify^* as follows:

Verify* Let vkSig^* be the verification key in the challenge ciphertext $(c^*, \sigma^*, \text{vkSig}^*)$. Upon receiving a decryption query on $(c, \sigma, \text{vkSig})$, output $\perp$ if either $\text{vkSig} = \text{vkSig}^*$ or $\text{Verify}_{\text{vkSig}}(c, \sigma) = 0$.

Claim. For $b \in \{0,1\}$, $\{\mathsf{NME}_b(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k\in\mathbb{N}} \approx_c \{\mathsf{NME}_b{}^{(1)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k\in\mathbb{N}}$

Proof. Follows using standard techniques from the security of the signature scheme. □

Experiment $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ modifies $\mathsf{NME}_b{}^{(1)}$ to use an extractor to decrypt the inner layer cipher text for the decryptions in the final parallel decryption query. Specifically, in $\mathsf{NME}_b{}^{(2)}$ the calls are submitted to NMDec^* as described below. This is unlike $\mathsf{NME}_b{}^{(1)}$ where the final ciphertexts $d_1, ..., d_{p(k)}$ are presented by A_2 for parallel decryption via calls to NMDec, :

NMDec$^*(d_i = \boldsymbol{C}, \sigma, \textbf{vkSig})$ If $0 = \text{Verify}^*_{\text{vkSig}}(\boldsymbol{C}, \sigma)$ output $\perp$. For $i = 1 \ldots k$, do $C'_i \leftarrow \mathsf{dec}_{sk_i^{\text{vkSig}_i}}(C_i)$
If $\exists_j, C'_1 \neq C'_j$, output $\perp$. Use the extractor $C^*_{\mathcal{A}}$ (defined in Lemma 3) to extract C'_i, where i is the smallest value s.t. $\text{vkSig}_i \neq \text{vkSig}^*_i$, where vkSig^* is the verification key of the challenge ciphertext. Return the extracted plaintext.

Lemma 3. *For* $b \in \{0,1\}$, $\{\mathsf{NME}_b{}^{(1)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k\in\mathbb{N}} \approx_c \{\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k\in\mathbb{N}}$

This lemma might, on first glance, seem to follow immediately because the whole purpose of the extractor is that it be able to simulate a decryption oracle. However, since the adversary has (i) seen the challenge ciphertext, (ii) it is not aware of the randomness used to produce this ciphertext, and (iii) created final parallel decryption queries potentially based on the challenge ciphertext, there is no a priori reason to believe the **sPA1** extractor will "decrypt" properly. However, we are only extracting on the inner layers of ciphertexts, and the inner layer of the challenge ciphertext has been hidden by the encryptions on the outer layer. Further, the outer layer is weakly simulatable, so we can argue that these new ciphertexts issued for parallel decryption, described in point (iii) above, are not dependent on the randomness of the inner-layer of the challenge ciphertext. Therefore, the extractor will function correctly.

Proof. The experiments differ only if the extractor returns a result that is different from the of the decryption oracle. We define **badExtract** to capture this event, and show that it occurs with negligible probability. Assume (for contradiction) that there exists an adversary $\mathcal{A}$ which induces the event **badExtract** to occur with non-negligible probability. We show that $\mathbf{E}$ is not a weakly simulatable encryption scheme.

Note that the public- and secret-keys for Π are composed of $2k+1$ keys that are generated using the key generation algorithm for encryption scheme $\mathbf{E}$. To encrypt a message m, we first generate a pair of signing keys, (vkSig, skSig), and then encrypt m with a fixed public key, pk_0, in the set of $2k+1$ public keys (we refer to this ciphertext as the inner layer). Then, we select a subset of size k out of the $2k$ remaining keys determined by the bits of vkSig, and encrypt the inner layer using fresh random coins for k times under those k keys (we refer to these k ciphertexts as the outer layer). We refer to the key used to encrypt the inner layer as the inner key, and the remaining $2k$ keys as the outer keys.

The technical difficulty in showing that **badExtract** does not occur in the $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ experiment is that by providing the challenge ciphertext, we actually provide the adversary with ciphertexts that are encrypted using k keys out of the $2k$ outer keys. We must argue that even in this case, there should be a way to extract the plaintext of the queries submitted by the adversary on the spots in the outer layer that are encrypted under a new key from the k keys used in the outer layer of the challenge ciphertext.

To do so, we first construct an $\mathbf{sPA1}^+_\ell$ ciphertext creator $\mathbf{C}_\mathcal{A}$ using the adversary $\mathcal{A}$. Since the encryption scheme $\mathbf{E}$ is $\mathbf{sPA1}^+_\ell$ secure, there exists an extractor for $\mathbf{C}_\mathcal{A}$ which we call $\mathbf{C}^*_\mathcal{A}$. Then we define a series of hybrids using $\mathbf{C}_\mathcal{A}$ and $\mathbf{C}^*_\mathcal{A}$, that are indistinguishable assuming $\mathbf{E}$ is weakly simulatable. The last hybrid in that series perfectly simulates the $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ experiment for $\mathcal{A}$ up to the point when $\mathcal{A}$ returns the vector of the ciphertexts after receiving the challenge ciphertext. Based on the indistinguishability of the hybrids, we will argue that there exists an extractor that can decrypt the adversary's queries on the first spot i where $\text{vkSig}_i \neq \text{vkSig}^*_i$ with overwhelming probability. Notice that the extractor cannot be used to decrypt the outer layer on the spots where $\text{vkSig}_i = \text{vkSig}^*_i$, otherwise it could be argued that the encryption scheme $\mathbf{E}$ is indeed **PA2** secure (**PA2** security is defined in [BP04]) and hence $\mathsf{CCA2}$ secure.

First we construct an $\mathbf{sPA1}^+_\ell$ ciphertext creator $\mathbf{C}_\mathcal{A}$ from $\mathcal{A}$ where $\ell = 2k+1$. $\mathbf{C}_\mathcal{A}$ interacts with the $\mathbf{sPA1}^+_\ell$ experiment in "the outside" as follows:

- $\mathbf{C}_\mathcal{A}$ receives $2k+1$ public keys $\left(\{pk'_i\}_{i\in[0\ldots 2k]}\right)$ from the $\mathbf{sPA1}^+_\ell$ experiment. It generates a pair of signing keys $(\text{vkSig}^*, \text{skSig}^*) \leftarrow \mathsf{GenKey}(1^k)$ internally and sets $pk = \left(\{pk^\alpha_i\}_{i\in[0\ldots 2k],\alpha\in\{0,1\}}\right)$ as described (intuitively, $\mathbf{C}_\mathcal{A}$ arranges pk such that it can potentially sign a vector of ciphertexts that are supposed to be encrypted under the last k keys in $\boldsymbol{pk'}$, $\left(pk'_{k+1}, \ldots, pk'_{k+k}\right)$, to generate a valid ciphertext in the Π scheme):

$$\text{for } i \in [0\ldots k] \;\&\; \alpha \in \{0,1\}, pk^\alpha_i = \begin{cases} pk'_0 & \text{if } i = 0 \\ pk'_i & \text{else if } \text{vkSig}^*_i \neq \alpha \\ pk'_{i+k} & \text{otherwise} \end{cases}$$

$\mathbf{C}_{\mathcal{A}}$ runs $\mathcal{A}_1$ on the input of pk. Note that this rearrangement of keys is crucial to make the view of the adversary $\mathcal{A}$ on the arrangement of the keys identical to its view in a real $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ experiment. To see this, consider the following example. The adversary $\mathcal{A}$ might abort whenever the keys used in the outer layer of the challenge ciphertext are the last k keys in $\boldsymbol{pk}$. Such a coincidence occurs in the simulated $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ experiment with probability 1 if $\mathbf{C}_{\mathcal{A}}$ sets $\boldsymbol{pk}$ to be the same as $\boldsymbol{pk'}$, while this coincidence occurs in a real $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ experiment with negligible probability due to the security of the signature scheme.

- Whenever $\mathbf{C}_{\mathcal{A}}$ receives a query $\big(\{y_i\}_{i\in[k]}, \sigma, \mathrm{vkSig}\big)$ from $\mathcal{A}_1$, it first checks if the signature is valid. If not, it returns $\perp$ as the answer to this query. Next, it checks whether $\mathrm{vkSig} = \mathrm{vkSig}^*$. If so, it aborts. Otherwise, $\mathbf{C}_{\mathcal{A}}$ submits y_i's one by one to the extractor. If all of the queries do not get decrypted to the same value, $\mathbf{C}_{\mathcal{A}}$ returns $\perp$ to $\mathcal{A}_1$ as the answer to that query. But if all of the queries get decrypted to the same value y_0, $\mathbf{C}_{\mathcal{A}}$ then submits y_0 (which is supposed to be an encryption under pk_0^0) to the extractor and returns the result to $\mathcal{A}_1$. Eventually $\mathcal{A}_1$ returns (m_0, m_1, St) and halts. $\mathbf{C}_{\mathcal{A}}$ outputs (m_0, m_1).
- $\mathbf{C}_{\mathcal{A}}$ accesses its oracle $\mathcal{O}$ and generates k blocks of random bits of length l, giving the vector $\boldsymbol{x} = (x_1, \ldots, x_k)$. Let $\boldsymbol{y} = \big(f(pk'_{k+1}, x_1), \ldots, f(pk'_{k+k}, x_k)\big)$. $\mathbf{C}_{\mathcal{A}}$ then computes $\sigma^* = \mathsf{Sign}(\boldsymbol{y}, \mathrm{skSig}^*)$, and runs $\mathcal{A}_2$ on the input $y^* = (\boldsymbol{y}, \sigma^*, \mathrm{vkSig}^*)$ and St.
- $\mathcal{A}_2$ returns a vector of ciphertexts $\boldsymbol{Y}$ and the state information S and halts. For all $j \in [|\boldsymbol{Y}|]$, $\mathbf{C}_{\mathcal{A}}$ does the following: on the query $Y_j = (\{y_i\}_{i\in[k]}, \sigma, \mathrm{vkSig})$, it first checks if the signature is valid. If not, it moves to the next query. Otherwise, it checks whether $\mathrm{vkSig} = \mathrm{vkSig}^*$. If so, it aborts. Otherwise, $\mathbf{C}_{\mathcal{A}}$ finds the first index i where $\mathrm{vkSig}_i \neq \mathrm{vkSig}_i^*$, and submits y_i to its extractor to be "decrypted" under $pk_i^{\mathrm{vkSig}_i}$. $\mathbf{C}_{\mathcal{A}}$ then submits the answer from the extractor (which is supposed to be an encryption under pk_0^0) again to the extractor to be "decrypted" under pk_0^0. Denote the result m'_j.
  $\mathbf{C}_{\mathcal{A}}$ returns $\{Y_j, m'_j\}_{j\in[|\boldsymbol{Y}|]}$ and the state information S, it halts.

Since $\mathbf{C}_{\mathcal{A}}$ is a $\mathbf{sPA1}^+_\ell$ ciphertext creator adversary, the $\mathbf{sPA1}^+_\ell$ security of $\mathbf{E}$ implies there exists an extractor $\mathbf{C}^*_{\mathcal{A}}$ whose answers to the decryption queries submitted by $\mathbf{C}_{\mathcal{A}}$ are indistinguishable from their true decryptions. We call the above interaction **Game 1**. Let $\Pr[W_i]$ be the probability of the adversary $\mathbf{C}_{\mathcal{A}}$ inducing the event **badExtract** in the **Game** i. The $\mathbf{sPA1}^+_\ell$ security implies that $\Pr[W_1]$ is bounded by $\mathbf{Adv}^{\mathbf{sPA1}^+_\ell}(\mathbf{E}, \mathbf{C}_{\mathcal{A}}, \mathbf{C}^*_{\mathcal{A}}, k)$ which is negligible in k. Hence:

$$\Pr[W_1] \leq \mathbf{Adv}^{\mathbf{sPA1}^+_\ell}(\mathbf{E}, \mathbf{C}_{\mathcal{A}}, \mathbf{C}^*_{\mathcal{A}}, k) \tag{1}$$

We will define another game, **Game 2**, which is identical to **Game 1** with the difference that instead of a fake ciphertext, $\mathcal{A}_2$ is fed with a real ciphertext as the challenge ciphertext. The aborting probability of $\mathcal{A}_2$ in **Game 1** and **Game 2** is negligibly close otherwise it can be argued that $\mathbf{E}$ is not weakly simulatable. In what follows, we only deal with the probability of inducing the event

badExtract. Also notice that **Game 2** simulates $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ for the adversary $\mathcal{A}$ up to the point when the adversary $\mathcal{A}$ returns a vector of ciphertexts after seeing the challenge ciphertext. That is because $\mathbf{C}_{\mathcal{A}}$ only needs the vector of the ciphertext generated by $\mathcal{A}$ after revealing the challenge ciphertext to induce the event **badExtract** to occur. After receiving such a vector of ciphertexts, $\mathbf{C}_{\mathcal{A}}$ does not need to complete the simulation of the $\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))$ experiment for $\mathcal{A}$.

In **Game 2** we modify the oracle $\mathcal{O}$ as follows: when $\mathbf{C}_{\mathcal{A}}$ accesses the oracle $\mathcal{O}$ for the i^{th} time, instead of $r \in \{0,1\}^l$, $\mathcal{O}$ returns $f^{-1}(pk'_{k+i}, \mathsf{enc}_{pk'_{k+i}}(m_d))$ where m_d is picked randomly out of the two messages returned by $\mathcal{A}$. During **Game 2**, the random bit d is fixed. We argue that such a change does not affect the advantage of $\mathbf{C}_{\mathcal{A}}$ in inducing the event **badExtract** as otherwise $\mathbf{E}$ is not weakly simulatable countering our assumption. Using $\mathbf{C}_{\mathcal{A}}$ and $\mathbf{C}^*_{\mathcal{A}}$, we build the attacker $\mathcal{B}$ that distinguishes $(r, f(.,r))$ and $(f^{-1}(., c = \mathsf{enc}(.,.)), c)$ as follows:

1. The challenger samples k pairs of random keys $(pk_i, sk_i) \leftarrow \mathsf{gen}(1^k)$ for $1 \le i \le k$, and a random bit b.
2. The attacker $\mathcal{B}$ receives $\{pk_i\}_{i\in[k]}$. $\mathcal{B}$ then samples $k+1$ other random keys $(pk'_i, sk'_i) \leftarrow \mathsf{gen}(1^k)$ for $0 \le i \le k$. Let $\boldsymbol{pk''} = \big(pk'_0, pk'_1, \ldots, pk'_k, pk_1, pk_2, \ldots, pk_k\big)$. $\mathcal{B}$ samples random coins for $\mathbf{C}_{\mathcal{A}}$ and $\mathbf{C}^*_{\mathcal{A}}$ and sets $st \leftarrow (\boldsymbol{pk''}, R[\mathbf{C}_{\mathcal{A}}])$. $\mathcal{B}$ runs $\mathbf{C}_{\mathcal{A}}$ on the input $\boldsymbol{pk''}$, and $\mathbf{C}^*_{\mathcal{A}}$ on the input st. Eventually $\mathbf{C}_{\mathcal{A}}$ outputs (m_0, m_1). $\mathcal{B}$ randomly chooses $d \in \{0,1\}$ and outputs $c'_d = \mathsf{enc}_{pk'_0}(m_d)$. The challenger samples $r_i \in \{0,1\}^l$ for $1 \le i \le k$ and returns $\{(r_i, f(pk_i, r_i))\}_{i\in[k]}$ if $b = 0$, and $\big\{\big(f^{-1}(pk_i, c_i = \mathsf{enc}_{pk_i}(c'_d)), c_i\big)\big\}_{i\in[k]}$ if $b = 1$. Call the resulting vector (given by the challenger) $\boldsymbol{y}$. $\mathcal{B}$ then forwards $\big\{f^{-1}(pk_i, y_i)\big\}_{i\in[k]}$ to $\mathbf{C}_{\mathcal{A}}$ and $\mathbf{C}^*_{\mathcal{A}}$ when $\mathbf{C}_{\mathcal{A}}$ queries $\mathcal{O}$ for the i^{th} time. After $\mathbf{C}_{\mathcal{A}}$ halts, the attacker $\mathcal{B}$ checks if all the queries made by $\mathbf{C}_{\mathcal{A}}$ to the extractor after outputting m_0 and m_1 were answered correctly. This is done by using the extractor using $\boldsymbol{sk'}$ (notice that $\mathbf{C}_{\mathcal{A}}$ was made in a way that after returning m_0 and m_1, it always only asks the extractor on the ciphertexts encrypted under $\boldsymbol{pk'}$ which are the first $k+1$ keys in $\boldsymbol{pk}$). If so it outputs $b' = 0$ otherwise $b' = 1$.

When $b = 0$, **Game 1** is being simulated, and when $b = 1$, **Game 2** is being simulated. Therefore:

$$\begin{aligned}\Pr[b' = b] &= \Pr[b = 0] \cdot \Pr[b' = b | b = 0] + \Pr[b = 1] \cdot \Pr[b' = b | b = 1] \\ &= \frac{1}{2} \cdot (1 - \Pr[W_1]) + \frac{1}{2} \cdot \Pr[W_2]\end{aligned}$$

On the other hand, by Lemma 2, the advantage of the attacker $\mathcal{B}$ in guessing the bit b is negligible in k, and hence there exists a negligible function $\epsilon_1(.)$ such that $\Pr[b' = b] \le \frac{1}{2} + \epsilon_1(k)$. Therefore:

$$\Pr[b' = b] = \frac{1}{2} \cdot (1 - \Pr[W_1]) + \frac{1}{2} \cdot \Pr[W_2] \leq \frac{1}{2} + \epsilon_1(k)$$

$$\Longrightarrow \Pr[W_2] \leq 2 \cdot \epsilon_1(k) + \Pr[W_1] \tag{2}$$

$$\Longrightarrow \Pr[W_2] \leq 2 \cdot \epsilon_1(k) + \mathbf{Adv}^{\mathbf{sPA1}^{+}_{\ell}}(\mathbf{E}, \mathbf{C}_{\mathcal{A}}, \mathbf{C}^{*}_{\mathcal{A}}, k) \tag{3}$$

Inequality (3) follows from Inequalities (1) and (2). Therefore $\Pr[W_2]$ is negligible. Since $\Pr[W_2]$ is the probability that the event **badExtract** occurs, we conclude that there is a negligible chance that **badExtract** occurs. Hence:

$$\{\mathsf{NME}_b{}^{(1)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k \in \mathbb{N}} \approx_c \{\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k \in \mathbb{N}}$$

□

Lemma 4. *For every p.p.t. adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, there exists a p.p.t. adversary $\mathcal{B}$ such that for $b \in \{0,1\}$,*

$$\{\mathsf{NME}_b{}^{(2)}(\Pi, \mathcal{A}, \mathcal{D}, k, p(k))\}_{k \in \mathbb{N}} \equiv \{\mathsf{CPA}_b(\boldsymbol{E}, \mathcal{B}, k)\}_{k \in \mathbb{N}}$$

Proof. In the proof of Lemma 3, we showed how to construct the ciphertext creator $\mathbf{C}_{\mathcal{A}}$ that runs $\mathcal{A}$ internally and proved that there exists an extractor $\mathbf{C}^{*}_{\mathcal{A}}$ that can decrypt the queries submitted by $\mathbf{C}_{\mathcal{A}}$ with overwhelming probability.

We build the CPA adversary $\mathcal{B}$ that interacts with the CPA experiment. Having the algorithms for $\mathcal{A}$, $\mathbf{C}_{\mathcal{A}}$ and $\mathbf{C}^{*}_{\mathcal{A}}$, the CPA adversary $\mathcal{B}$ acts as follows: $\mathcal{B}$ receives the public key pk' from the CPA experiment, and generates $2k$ keys as $(pk_i^{''}, sk_i^{''}) \leftarrow \mathsf{gen}(1^k)$ for $i \in [2k]$. Let $\boldsymbol{pk} = \left(pk', \boldsymbol{pk}^{''}\right)$. $\mathcal{B}$ runs $\mathbf{C}_{\mathcal{A}}$ (that simulates $\mathcal{A}$ internally) on $\boldsymbol{pk}$ and its random coins. Whenever $\mathbf{C}_{\mathcal{A}}$ asks a query, $\mathcal{B}$ runs $\mathbf{C}^{*}_{\mathcal{A}}$ to answer them ($\mathbf{C}^{*}_{\mathcal{A}}$ gets to know the random coins of $\mathbf{C}_{\mathcal{A}}$ and all of its input as described in the proof of Lemma 3). Eventually $\mathbf{C}_{\mathcal{A}}$ outputs (m_0, m_1). $\mathcal{B}$ outputs m_0 and m_1 to the CPA experiment, and receives a ciphertext y. Remember that $\mathbf{C}_{\mathcal{A}}$ now accesses the oracle $\mathcal{O}$ k times. Using fresh random coins for each encryption, $\mathcal{B}$ computes $C_i = \mathsf{enc}_{pk_{k+i}}(y)$ and sends $f^{-1}(pk_{k+i}, C_i)$ to $\mathbf{C}_{\mathcal{A}}$ (and $\mathbf{C}^{*}_{\mathcal{A}}$) on the i^{th} access to $\mathcal{O}$. Eventually $\mathbf{C}_{\mathcal{A}}$ returns $\{Y_i, m'_i\}_{i \in [|\boldsymbol{Y}|]}$ and the state information S and halts. The only step left in determining the decryption of Y_i is to decrypt all the ciphertexts in the outer layer, and check that they all decrypt to the same value. $\mathcal{B}$ has the $2k$ secret keys for the outer layer, hence it can do the mentioned check. If the outer layer ciphertexts of Y_i do not decrypt to the same value, the decryption of Y_i is $\perp$, otherwise the decryption of Y_i is m'_i. After $\mathcal{B}$ decrypts all the Y_i, it submits the results along with the state information S to the distinguisher $\mathcal{D}$ and forwards $\mathcal{D}$'s output to the CPA experiment.

4 More Than Non-Malleable CCA1 Encryption Scheme

In the previous section, we showed how to build a non-malleable CCA1 encryption scheme from any encryption scheme that is weakly simulatable and $\mathbf{sPA1}_{\ell}$

Algorithm 1: DEG

function $\mathcal{G}(1^k)$
$(p, q, g) \leftarrow G(1^k)$
$x_1 \leftarrow \mathbb{Z}_q; X_1 \leftarrow g^{x_1} \mod p.$
$x_2 \leftarrow \mathbb{Z}_q; X_2 \leftarrow g^{x_2} \mod p.$
Return $(pk = (p, q, g, X_1, X_2),$
$sk = (p, q, g, x_1, x_2))$

function $\mathcal{E}(pk, M)$
$y \leftarrow \mathbb{Z}_q; Y \leftarrow g^y \mod p.$
$W \leftarrow X_1^y; V \leftarrow X_2^y \mod p.$
$U \leftarrow V \cdot M \mod p$
Return $C = (Y, W, U)$

function $\mathcal{D}(sk, C)$
if $W \neq Y^{x_1} \mod p$ **then** Return $\perp$.
else Return $M \leftarrow U \cdot Y^{-x_2} \mod p$

Algorithm 2: CS-Lite

function $\mathcal{G}(1^k)$
$(p, q, g_1) \leftarrow G(1^k); g_2 \leftarrow G_q \backslash \{1\}$
$x_1 \leftarrow \mathbb{Z}_q; x_2 \leftarrow \mathbb{Z}_q; z \leftarrow \mathbb{Z}_q.$
$X \leftarrow g_1^{x_1}.g_2^{x_2} \mod p; Z \leftarrow g_1^z \mod p.$
Return $(pk = (p, q, g_1, g_2, X, Z),$
$sk = (p, q, g_1, g_2, x_1, x_2, z))$

function $\mathcal{E}(pk, M)$
$r \leftarrow \mathbb{Z}_q.$
$R_1 \leftarrow g_1^r \mod p; R_2 \leftarrow g_2^r \mod p.$
$E \leftarrow Z^r \cdot M \mod p; V \leftarrow X^r \mod p$
Return $C = (R_1, R_2, E, V)$

function $\mathcal{D}(sk, C)$
if $V \neq R_1^{x_1} \cdot R_2^{x_2} \mod p$ **then** Return $\perp$.
else Return $M \leftarrow E \cdot R_1^{-z} \mod p$

secure. Define a parallel query as a query consisting of unbounded number of ciphertexts, none of which will be decrypted until all the ciphertexts in the query are submitted. In the NM-CCA1 game, the adversary is allowed to ask an unbounded number of queries before seeing the challenge ciphertext, and one parallel query afterwards. This compares with CCA2 secure encryption schemes, which are secure even if the adversary asks an unbounded number of queries before and after seeing the challenge ciphertext. The NM-CCA1 constructions seem to be much weaker primitives. However, between the extremes of the NM-CCA1 security and the CCA2 security, a range of security notions can be defined that distinguish themselves based on how many queries the adversary may ask after revealing the challenge ciphertext without sacrificing indistinguishability of ciphertexts.

Define cNM-CCA1 security identically to NM-CCA1 security except that the adversary can make $c \geq 1$ parallel queries after seeing the challenge ciphertext. We show how to extend our result to construct an encryption scheme that is cNM-CCA1 secure where c is a constant. The high level idea for constructing a cNM-CCA1 scheme is to add another c layers of encryption on top of the ciphertext from the previous section. Intuitively, with the first parallel query, the adversary can only ask queries that can help it to maul the first layer of encryption from the outside in the future. In other words, with the first parallel query, the adversary can gain no information about all the inner ciphertexts. Hence, to penetrate the innermost layer, the adversary has to ask at least c parallel queries. Notice also that this type of construction can only allow a constant c since each layer of encryption increases ciphertext size by a polynomial factor. For more details, see the full version.

References

[BP04] Bellare, M., Palacio, A.: Towards Plaintext-Aware Public-Key Encryption Without Random Oracles. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 48–62. Springer, Heidelberg (2004)

[CDSMW08] Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Black-Box Construction of a Non-malleable Encryption Scheme from Any Semantically Secure One. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 427–444. Springer, Heidelberg (2008)

[CGH04] Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM 51(4), 557–594 (2004)

[CHH+07] Cramer, R., Hanaoka, G., Hofheinz, D., Imai, H., Kiltz, E., Pass, R., Shelat, A., Vaikuntanathan, V.: Bounded CCA2-Secure Encryption. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 502–518. Springer, Heidelberg (2007)

[CS03] Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. 33(1), 167–226 (2003)

[Dam91] Damgård, I.: Towards Practical Public Key Systems Secure against Chosen Ciphertext Attacks. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 445–456. Springer, Heidelberg (1992)

[DDN03] Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIREV: SIAM Review 45 (2003)

[Den06a] Dent, A.W.: The Cramer-Shoup Encryption Scheme Is Plaintext Aware in the Standard Model. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 289–307. Springer, Heidelberg (2006)

[Den06b] Dent, A.W.: The hardness of the DHK problem in the generic group model. IACR Cryptology ePrint Archive 2006, 156 (2006)

[DN00] Damgård, I., Nielsen, J.B.: Improved Non-committing Encryption Schemes Based on a General Complexity Assumption. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 432–450. Springer, Heidelberg (2000)

[MM11] Matsuda, T., Matsuura, K.: Parallel Decryption Queries in Bounded Chosen Ciphertext Attacks. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 246–264. Springer, Heidelberg (2011)

[MS09] Myers, S., Shelat, A.: Bit encryption is complete. In: FOCS, pp. 607–616 (2009)

[PSV06] Pass, R., Shelat, A., Vaikuntanathan, V.: Construction of a Non-malleable Encryption Scheme from Any Semantically Secure One. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 271–289. Springer, Heidelberg (2006)

[PW09] Pass, R., Wee, H.: Black-box constructions of two-party protocols from one-way functions. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 403–418. Springer, Heidelberg (2009)

[Wee10] Wee, H.: Black-box, round-efficient secure computation via non-malleability amplification. In: FOCS, pp. 531–540. IEEE Computer Society (2010)

A Plaintext Awareness

A.1 sPA1$_\ell$ Secure Schemes

We argue that Cramer-Shoup Lite (CS-Lite) and Damgard's ElGammal (DEG) are **sPA1**$_\ell$ secure, based on a suitable modification of the Diffie-Hellman Knowl-

edge definition originally proposed by Damgård, and modified to permit interactive extractors by Bellare and Palacio [BP04].

$DHK1_\ell(k)$
$(p_i, q_i, g_i)_{i\in\ell[k]} \leftarrow G(1^k)$; $(a_i)_{i\in\ell[k]} \leftarrow \mathbb{Z}_q$; $A_i \leftarrow g_i^{a_i} \mod p$ for $i \in \ell[k]$
Let $R[H]$ and $R[H^*]$ be randomly selected strings for H and H^*.
$st \leftarrow ((p_i)_{i\in\ell[k]}, (q_i)_{i\in\ell[k]}, (g_i)_{i\in\ell[k]}, (A_i)_{i\in[\ell]}, R[H])$
while Simulate $H((p_i)_{i\in\ell[k]}, (q_i)_{i\in\ell[k]}, (g_i)_{i\in\ell[k]}, (A_i)_{i\in[\ell]}; R[H])$ **do**
  if H queries (i, B, W) **then**
    $(b, st) \leftarrow H^*((i, B, W), st; R[H^*])$
    if $W \equiv B^{a_i} \mod p$ and $B \not\equiv g_i^b \mod p$ **then** Return 1.
    else Return b.
Return 0.

We note that in the experiment, the change requires that the ciphertext creator be able to generate ciphertexts relative to a polynomial number of randomly chosen public-keys. It seems reasonable to conjecture that any extractor that could extract exponents with respect to single value $A = g^a$, could do so efficiently for many A_i.

We now argue that DEG is $\mathbf{sPA1}_\ell$ secure under the $DHK1_\ell$ definition.

Theorem 3. *For any polynomial ℓ, The DEG scheme is $\boldsymbol{sPA1}_\ell$ secure under the DHK_ℓ assumption.*

Proof. We build the DHK_ℓ adversary $\mathcal{B}$ that runs the $\mathbf{sPA1}_\ell$ adversary $\mathcal{A}$ internally and simulates the $\mathbf{sPA1}_\ell$ experiment for it. $\mathcal{B}$ receives $(p_i)_{i\in\ell[k]}$, $(q_i)_{i\in\ell[k]}$, $(g_i)_{i\in\ell[k]}$, $(A_i)_{i\in\ell[k]}$ and its random coins $R[H]$. For each $i \in [\ell]$, $\mathcal{B}$ samples $\hat{a}_i \leftarrow \mathbb{Z}_{q_i}$, computes $\hat{A}_i \leftarrow g_i^{\hat{a}_i} \mod p_i$ and sets $pk_i \leftarrow (q_i, g_i, A_i, \hat{A}_i)$. $\mathcal{B}$ then runs $\mathcal{A}$ on $(pk_i)_{i\in[\ell]}$ and the random coins $R[H]$ until $\mathcal{A}$ halts, answering to $\mathcal{A}$'s queries as follows: upon receiving the query $C = (i, Y, W, U)$ from $\mathcal{A}$, $\mathcal{B}$ submits (i, Y, W) to the DHK_ℓ extractor. The DHK_ℓ extractor returns the value b. If $b = 1$ then $\mathcal{B}$ returns $\perp$ as the decryption of C, otherwise $\mathcal{B}$ computes $M \leftarrow U.(\hat{A}_i^{\,b})^{-1} \mod p_i$ and return the result to $\mathcal{A}$.

Trivially, the integration of algorithm of $\mathcal{B}$ and its extractor(which depends on the algorithm of $\mathcal{A}$ and $\mathcal{B}$) is a potential extractor for the $\mathbf{sPA1}_\ell$ ciphertext creator adversary $\mathcal{A}$.

□

Theorem 4. *For any polynomial ℓ, The CS-Lite scheme is $\boldsymbol{sPA1}_\ell$ secure under the DHK_ℓ assumption.*

Proof. The proof is similar to the proof for Theorem 3. See the full version. □

B Weakly Simulatable Encryption Schemes

We argue that the Damgard ElGamal (DEG) scheme is weakly simulatable using an argument parallel to that of Dent[Den06a]. We remind the reader that the

definition for DEG is given on page 162. It has previously been shown that DEG is **sPA1** secure.

We use the notion of a simulatable group given by Dent [Den06a].

Definition 5. *(**Simulatable Group**) [Den06a] A Group G is simulatable if there exist two polynomial turing machines (f, f^{-1}) such that:*

- *f is a deterministic turing machine that takes a random element $r \in \{0,1\}^l$ as input, and outputs elements of G.*
- *f^{-1} is a probabilistic turing machine that takes elements of $h \in G$ as input, and outputs elements of $\{0,1\}^l$.*
- *$f(f^{-1}(C)) = C$ for all $h \in G$.*
- *There exists no polynomial time attacker $\mathcal{A}$ that has a non-negligible advantage in winning the following game:*
  1. *The challenger randomly chooses a bit $b \in \{0,1\}$.*
  2. *The attacker $\mathcal{A}$ executes on the input 1^k. The attacker has access to an oracle $\mathcal{O}_f$ that takes no input, generates a random element $r \in \{0,1\}^l$, and returns r if $b = 0$, and $f^{-1}(f(r))$ if $b = 1$. The attacker terminates by outputting a guess b' for b.*

  The attacker wins if $b = b'$ and its advantage is defined in the usual way.
- *There exists no polynomial-time attacker $\mathcal{A}$ that has a non-negligible advantage in winning the following game:*
  1. *The challenger randomly chooses $b \in \{0,1\}$.*
  2. *The attacker $\mathcal{A}$ executes on the input 1^k. The attacker has access to an oracle $\mathcal{O}_f$ that takes no input. If $b = 0$, then the oracle generates a random $r \in \{0,1\}^l$ and returns $f(r)$. Otherwise the oracle generates a random $h \in G$ and returns h. The attacker terminates by outputting a guess b' for b.*

  The attacker wins if $b = b'$ and its advantage is defined in the usual way.

Dent showed that groups in which the DDH assumptions are believed to hold are simulatable.

Lemma 5. *[Den06a] If q and p are primes such that $p = 2q + 1$, and G is the subgroup of $\mathbb{Z}_p^*$ of order q, then G is simulatable.*

Using this fact we show that DEG is weakly simulatable.

Theorem 5. *The DEG encryption scheme is weakly simulatable if it is instantiated on a simulatable group G (for the definition for simulatable groups, see [Den06a]) on which the DDH problem is hard.*

Proof. See the full version.

□

Theorem 6. *The Cramer-Shoup lite encryption scheme is weakly simulatable if it is instantiated on a simulatable group G on which the DDH problem is hard.*

Proof. Similar to the proof of Theorem 5 which is presented in full version of the paper. □