A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks

Jan Camenisch[*] Nishanth Chandran[†] Victor Shoup[‡]

September 22, 2008

Abstract

Recently, at Crypto 2008, Boneh, Halevi, Hamburg, and Ostrovsky (BHHO) solved the long-standing open problem of "circular encryption," by presenting a public key encryption scheme and proving that it is semantically secure against key dependent chosen plaintext attack (KDM-CPA security) under standard assumptions (and without resorting to random oracles). However, they left as an open problem that of designing an encryption scheme that *simultaneously* provides security against both key dependent chosen plaintext *and* adaptive chosen ciphertext attack (KDM-CCA2 security). In this paper, we solve this problem. First, we show that by applying the Naor-Yung "double encryption" paradigm, one can combine any KDM-CPA secure scheme with any (ordinary) CCA2 secure scheme, along with an appropriate non-interactive zero-knowledge proof, to obtain a KDM-CCA2 secure scheme. Second, we give a concrete instantiation that makes use the above KDM-CPA secure scheme of BHHO, along with a generalization of the Cramer-Shoup CCA2 secure encryption scheme, and recently developed pairing-based NIZK proof systems. This instantiation increases the complexity of the BHHO scheme by just a small constant factor.

1 Introduction

Encryption is the oldest cryptographic primitive; indeed, cryptography used to be synonymous with encryption. Despite this, the right definition for the security of encryption schemes has still not been settled! The first formal definition of security for public key encryption was that of *semantic security* [GM82], which, loosely speaking, states that given an encryption of a message an adversary cannot learn any information about the message itself. As it turned out, this notion of security does not offer sufficient protection for most practical applications [Ble98], as it does not take into account that an adversary could learn (partial information about) some plaintext when he has access to a decryption oracle. The subsequent stronger notion of security against chosen ciphertext attacks (CCA2 security [RS91]) takes this into consideration and gives an adversary access to a decryption oracle that will decrypt any ciphertext except a particular "challenge ciphertext". CCA2 security was considered the final answer with regard to the security of public key encryption schemes.

However, none of the above notions of security allow an adversary to obtain encryptions of secret keys or, more generally, functions of secret keys. Black, Rogaway, and Shrimpton formally defined such a notion, calling it *Key Dependent Message (KDM)* security [BRS02]. A similar notion, called

[*] IBM Research, work funded by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 216483.

[†] UCLA, work done while visiting IBM Research.

[‡] NYU, work done while visiting IBM Research, supported by NSF award number CNS-0716690.

circular security, was earlier defined by Camenisch and Lysyanskaya [CL01] and used to prevent sharing of credentials. Both papers provided constructions in the random oracle model.

Without resorting to the use of random oracles, constructing a public key encryption scheme (practical or not) that is semantically secure against key dependent chosen plaintext attack (KDM-CPA) was a long-standing open problem. It was only recently that Boneh et al. [BHHO08] gave a construction of a KDM-CPA secure public key encryption scheme. They proved their scheme secure under the Decisional Diffie-Hellman (DDH) assumption. We will refer to their scheme as the BHHO scheme, which extends to obtain KDM-CPA security under the more general K-linear assumption [Sha07, Kil07] (which includes the DDH assumption for $K = 1$ and the DLIN assumption [BBS04] for $K = 2$). However, Boneh et al. left as an open problem the construction of an encryption scheme that is simultaneously secure against key dependent chosen plaintext *and* chosen ciphertext attack (KDM-CCA2).

Our Contribution. In this paper, we solve this problem by constructing the first KDM-CCA2 secure public key encryption scheme that can be proved secure under standard assumptions, and without random oracles. In fact, we show that a variation of the Naor-Yung paradigm [NY90] allows one to combine any KDM-CPA secure encryption scheme and any regular CCA2 secure encryption scheme, together with a non-interactive zero knowledge (NIZK) proof [BFM88], to obtain a KDM-CCA2 secure encryption scheme.

Moreover, we give a nearly practical instantiation of our general construction using the BBHO KDM-CPA scheme, a K-linear version [CS02, Sha07, HK07b] of the Cramer-Shoup [CS98] CCA2 scheme, and recently developed pairing-based NIZK proof systems [GOS06, Gro06, GS08]. In the BHHO scheme a ciphertext is a couple of hundred group elements and our construction blows this up only be a small constant factor (two or three, depending on the cryptographic assumption we employ). For our construction, we need a pairing $e : \mathbb{G} \times \Gamma \to \mathbb{G}_{\mathrm{T}}$, and we prove security under the K-linear assumption in $\mathbb{G}$ and the L-linear assumption in Γ, for appropriate constants K and L (and we also need a collision-resistant hash function).

Motivational Example: Key-Wrap. The "key-wrap" problem motivates the need for KDM-CCA2 secure encryption in practice. The key-wrap mechanism is found, for instance, in cryptographic coprocessors such as IBM's Common Cryptographic Architecture [IBM08] and RSA's Public Key Cryptographic Standards [RSA04]. Cryptographic coprocessors are tamper-proof hardware tokens that process requests from applications to perform cryptographic tasks such as encryption, signing and so on. One can view these tokens as trusted hardware that store keys of all users in the system. When an application (or user) wishes to perform a cryptographic task, it authenticates itself to the token and the token processes the request. For the purpose of creating backup of data or to transport keys from one token to another, it is often desired to encrypt keys (also known as "key wrapping"). Naturally, when we encrypt private keys with other keys it might lead to a circularity. In other words, an adversary might get to see an encryption of a secret key sk_1 with public key pk_2 as well as an encryption of a secret key sk_2 with public key pk_1 (such circularity can in general be more complicated). Although one can circumvent this problem by maintaining a hierarchy of keys and/or by maintaining separate keys for the purpose of wrapping other keys, this is not always convenient or possible. In addition, since the hardware token performs decryption, an adversary may effectively have access to a decryption oracle.

Labeled Encryption. In many applications in which one uses a CCA2 secure encryption, the notion of a *label* is very useful. Very briefly, a label consists of public data which is non-malleably attached to a ciphertext. In effect, it allows the encryptor to control the context in which a ciphertext is decrypted. This notion has been around for a long time, under various names, e.g., "indicator", "tag", "header", "associated data" [LL93, SG98, Sho01, CS03, MRY04, Kil06, RS06]. While one can always implement the label mechanism by appending the label to the plaintext, this is often not the most practical way to achieve this.

Coming back to the key-wrap problem, a label may be used to describe the type of message being encrypted: if it encrypts a key, who the key belongs to, etc. When the hardware token decrypts a ciphertext labeled as a key, it can restrict the usage of the decrypted key; in particular, the token can ensure that such a key is only used within the token in appropriate ways (e.g., decryption, further key-wrap). Even if a token restricts the usage in this way, an adversary may attempt a chosen ciphertext attack by submitting an encryption of a key that actually belongs to Alice, and make it look like it belongs to Bob; moreover, perhaps the adversary is authorized to decrypt ciphertexts under Bob's key, which in effect allows him to decrypt ciphertexts encrypted under Alice's key. However, if labels are used as described above, CCA2 security will prevent such attacks from succeeding.

Because of their utility, we include labels in our definition of KDM-CCA2 security, and implement them in our construction. Moreover, we exploit the label mechanism for plain CCA2 encryption in our general construction to bind together the two ciphertexts and NIZK proof of the Naor-Yung paradigm. In particular, we shall see that CCA2 encryption scheme we use directly support labels in a way that interacts very nicely with pairing-based NIZK techniques, leading to a conceptually simple and quite efficient concrete instantiation of our general construction.

Another use of labels is to enlarge the message space of a CCA2 encryption scheme: to encrypt a sequence of messages as a package, one can generate a key pair for a strongly secure one-time signature scheme, and then encrypt each message in the sequence using the verification key as a label, and then signing the whole sequence of ciphertexts. This application is actually quite important for us, because the BHHO scheme can only encrypt one bit of a secret key at a time.

Other related work. Backes, Pfitzmann and Scedrov [BPS07] and Backes, Dürmuth and Unruh [BDU08] considered KDM-CCA2 security of symmetric and asymmetric encryption schemes, respectively. They in fact define a notion of security stronger than we consider in our paper, by allowing the adversary to obtain some of the secret keys. They showed that RSA-OAEP ([BR94]) is secure in this sense in the random oracle model.

Halevi and Krawczyk [HK07a] studied key-dependent message security (under the name key-dependent input (KDI) security) with respect to primitives such as pseudo-random functions (PRFs) and block ciphers. They showed that in the ideal-cipher model, KDI secure PRFs can be built if one restricts the functions of the key to be independent of the ideal-cipher. Further, they showed that this goal cannot be achieved in the standard model. On the positive side, they show that if one allows the PRF construction to depend on a fixed public value, but do not allow the function of the key to depend on this value, then KDI secure PRFs can be constructed in the standard model. Hofheinz and Unruh [HU08], constructed a symmetric key encryption scheme that achieves KDM-CPA security when an adversary can only make a bounded number and length of encryptions. Haitner and Holenstein [HH08] proved negative results for KDM-CPA security of encryption schemes when an adversary can query encryptions of specific functions of the secret key.

Outline of the paper. In §2, we give and discuss the definitions of KDM-CCA2, NIZK proofs, strong one-time signatures, i.e., the ingredients of our generic construction, which is presented in §3.

In §4, we present concrete instantiations of our building blocks: We recall the BHHO KDM-

CPA encryption scheme, the K-linear version of the Cramer-Shoup CCA2 encryption scheme, and Groth's strongly secure one-time signature scheme. As a service to the reader, we give a self-contained exposition of a simplified version of the NIZK proof system of Groth and Sahai [GS08] as it applies to linear equations over a group. This allows us to completely describe the instantiation of our construction and analyze its complexity.

In the Appendix, we discuss an alternative construction of KDM-CCA2 encryption that uses a CPA secure encryption scheme instead of a CCA2 secure encryption scheme but requires an NIZK proof system that provides (unbounded) simulation soundness [Sah99, SCO+01]. Finally, we show how to make the general NIZK proofs of [GS08] (unbounded) simulation sound, given a CCA2 secure encryption scheme that supports ciphertexts with labels, which again illustrates the power labels. This exposition borrows techniques from Groth [Gro06].

2 Preliminaries

2.1 Notation

When we say that an algorithm is *efficient*, we mean that the algorithm runs in probabilistic polynomial time in the security parameter. All our algorithms and functions take as input an implicit security parameter. When we say that a function is *negligible*, we mean that it is negligible in the implicit security parameter. Let $a\|b$ denote the concatenation of string a with string b.

2.2 Definition of KDM-CCA2 Security

Let $\mathbf{E}$ be a public key encryption system that supports ciphertexts with labels, which consists of three (probabilistic) efficient algorithms $\mathsf{EncKeyGen}$, E and D. $\mathsf{EncKeyGen}$ is a randomized key generation algorithm, that outputs a public key/secret key pair (pk, sk). The algorithm E takes as input message m (from the message space $\mathcal{M}$), public key pk and label ℓ, and outputs a ciphertext $c := \mathsf{E}(pk, m, \ell)$. When we need to explicitly refer to the randomness in the encryption, we shall refer to an encryption of message m with randomness r by $\mathsf{E}(pk, m, \ell; r)$. The decryption algorithm D takes as input secret key sk, ciphertext c, and a label ℓ, and either outputs message m or `reject`. The (perfect) correctness condition is that (with probability one) $\mathsf{D}(sk, \mathsf{E}(pk, m, \ell), \ell) = m$ for all messages m, labels ℓ and all (pk, sk) pairs output by $\mathsf{EncKeyGen}$.

We extend the definition of key dependent message security from Black et al. [BRS02] to the notion of security against chosen ciphertext attack ([NY90, RS91, DDN91]). We will note that the standard definitions of public key encryption security are specific instances of this definition.

Let $\mathcal{S}$ denote the space of secret keys output by $\mathsf{EncKeyGen}$. As in [HK07a] and [BHHO08], key-dependence is defined with respect to a fixed set of functions $\mathcal{C}$. Let $n > 0$ be an integer and let $\mathcal{C}$ be a finite set of functions $\mathcal{C} := \{f : \mathcal{S}^n \to \mathcal{M}\}$. KDM-security is defined with respect to $\mathcal{C}$ through the following two experiments between a challenger and an adversary $\mathcal{A}$. Let $\mathfrak{d} \in \mathcal{M}$ be a fixed (dummy) message in $\mathcal{M}$. Experiment b (where $b = 0, 1$) is defined as follows:

1. **Initialization phase:** In both experiments the challenger runs $\mathsf{EncKeyGen}()$ n times and obtains the n key pairs $(pk_1, sk_1), (pk_2, sk_2), \cdots, (pk_n, sk_n)$. It sends the vector $(pk_1, pk_2, \cdots, pk_n)$ to $\mathcal{A}$.

2. **Query phase:** In both experiments, $\mathcal{A}$ may adaptively make the following two types of queries to the challenger.

(a) **Encryption queries:** $\mathcal{A}$ can make a query of the form (i, f, ℓ) where $1 \leq i \leq n$, $f \in \mathcal{C}$ and ℓ is a label. The challenger responds by setting $m := f(sk_1, sk_2, \cdots, sk_n) \in \mathcal{M}$.

In Experiment $b = 0$, it sets $c := \mathsf{E}(pk_i, m, \ell)$.

In Experiment $b = 1$, it sets $c := \mathsf{E}(pk_i, \mathfrak{d}, \ell)$.

In both experiments, the challenger sends c to $\mathcal{A}$.

When the adversary $\mathcal{A}$ submits (i, f, ℓ) as an encryption query and the response of the challenger is c, we call (i, c, ℓ) a *target tuple*.

(b) **Decryption queries:** In both experiments, $\mathcal{A}$ can make a query of the form (i, c, ℓ) where $1 \leq i \leq n, c$ is a string to be decrypted using secret key sk_i and ℓ is a label. The only restriction is that (i, c, ℓ) cannot be a (previous) target tuple. Note that c might not necessarily be a valid ciphertext. That is, c might not be an output of $\mathsf{E}(pk_j, m, \ell)$ for some $1 \leq j \leq n, m \in \mathcal{M}$ and ℓ.

In both experiments, the challenger responds with $\mathsf{D}(sk_i, c, \ell)$.

3. **Final phase:** Finally, the adversary outputs a bit $b' \in \{0, 1\}$.

We say that $\mathcal{A}$ is a $\mathcal{C}$-KDM-CCA2 adversary. Let W_b be the probability with which $\mathcal{A}$ outputs $b' = 1$ in Experiment b.

Definition 1 (KDM-CCA2) *A public key encryption scheme* **E** *is* KDM-CCA2 *secure with respect to* $\mathcal{C}$ *if* $\big|\Pr\big[W_0\big] - \Pr\big[W_1\big]\big|$ *is negligible for all efficient adversaries* $\mathcal{A}$.

Note that the standard definitions of public key encryption security can be viewed as specific instances of the above general definition.

KDM-CPA: By restricting $\mathcal{A}$ from making any decryption queries, we get the definition of key-dependent message semantic security (KDM-CPA) as defined in [BHHO08].

CCA2: When we restrict the set of functions $\mathcal{C}$ from which $\mathcal{A}$ can draw f to the set of all constant functions on $\mathcal{S}^n \rightarrow \mathcal{M}$, we get the security experiment for multiple message, multiple key CCA2 security, which is equivalent to the standard CCA2 security for a single message and single key (see [BBM00]). If we further restrict $\mathcal{A}$ from making any decryption queries, we obtain the standard definition for semantic security (also see [BBM00]).

Also note that, unlike regular CPA and CCA2 security, for both KDM-CPA and KDM-CCA2 security, one cannot reduce the attack game to a single encryption query and a single key pair.

We note that the definition of security by Backes et al. [BDU08] is somewhat stronger in that it allows the adversary to obtain some secret keys. To benefit from this in practice, the users need to carefully keep track of which keys were compromised and which are related, e.g., via key-wrap. In contrast, our definition pessimistically assumes that if one key is compromised then all potentially related keys should be considered compromised as well—which is probably more realistic.

When we use a public key encryption scheme **E** that does not support labels, we refer to the encryption and decryption algorithms of such a scheme by $\mathsf{E}(pk, m)$ and $\mathsf{D}(sk, c)$ respectively.

2.3 Non-interactive zero-knowledge proofs

Let R be a binary relation that is efficiently computable. For pairs of the form $(x, w) \in R$, x is referred to as the statement and w as the witness. Let $\mathcal{L} := \{x : (x, w) \in R \text{ for some } w\}$.

A non-interactive proof system for R consists of the following efficient algorithms: a common reference string (CRS) generation algorithm CRSGen, a prover P, and a verifier V. The CRSGen algorithm outputs the CRS denoted by $\mathfrak{C}$. P takes as input $\mathfrak{C}$, statement x and witness w. It produces a proof $\mathfrak{p}$ if $(x, w) \in R$ and outputs `failure` otherwise. V takes as input $\mathfrak{C}, x$ and $\mathfrak{p}$. V outputs `accept` if it accepts the proof and `reject` otherwise.

Definition 2 (NIZK[BFM88, FLS90]) $(\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ *is a* non-interactive zero-knowledge (NIZK) proof system *for R if it has the following properties described below:*

Perfect Completeness: *For all $\mathfrak{C}$ output by* $\mathsf{CRSGen}()$*, for all $(x, w) \in R$, and for all $\mathfrak{p} := \mathsf{P}(\mathfrak{C}, x, w)$,* $\Pr[\mathsf{V}(\mathfrak{C}, x, \mathfrak{p}) \text{ outputs } \texttt{reject}] = 0$.

Computational Soundness: *Consider the following game:*

1. $\mathsf{CRSGen}()$ *is run to obtain $\mathfrak{C}$. The adversary $\mathcal{A}$ is given $\mathfrak{C}$.*
2. *$\mathcal{A}$ responds with $(x, \mathfrak{p})$.*

$\mathcal{A}$ wins the game if $\mathsf{V}(\mathfrak{C}, x, \mathfrak{p}) = \texttt{accept}$ *and $x \notin \mathcal{L}$. Let W be the event that A wins the game. Then, for all efficient adversaries $\mathcal{A}$, we have* $\Pr[W]$ *is negligible.*

Computational Zero-knowledge: *Let $\mathsf{S} := (\mathsf{S}_1, \mathsf{S}_2)$ be a simulator running in polynomial time. Let W_i be the probability with which $\mathcal{A}$ outputs 1 in Experiment i, for $i = 0$ or 1. Consider the following two experiments:*

Experiment 0**:** $\mathsf{CRSGen}()$ *is run and the output $\mathfrak{C}$ is given to $\mathcal{A}$. $\mathcal{A}$ is then given oracle access to $\mathsf{P}(\mathfrak{C}, \cdot, \cdot)$.*

Experiment 1**:** $\mathsf{S}_1()$ *generates $\mathfrak{C}$ and trapdoor $\mathfrak{t}$. $\mathcal{A}$ is given $\mathfrak{C}$, and is then given oracle access to $\mathsf{S}'(\mathfrak{C}, \mathfrak{t}, \cdot, \cdot)$, where $\mathsf{S}'(\mathfrak{C}, \mathfrak{t}, x, w)$ is defined to be $\mathsf{S}_2(\mathfrak{C}, \mathfrak{t}, x)$ if $(x, w) \in R$ and* `failure` *if $(x, w) \notin R$.*

Then, for all efficient adversaries $\mathcal{A}$, we have $\left|\Pr[W_0] - \Pr[W_1]\right|$ *is negligible.*

Note that [BFM88] give a weaker, "one time" definition of computational zero-knowledge, in which the adversary is allowed to see only one fake proof. However, this is insufficient for our purposes.

2.4 Strongly secure one-time signatures

Let $\mathcal{M}$ be the message space. A *signature scheme* is a triple of (probabilistic) efficient algorithms $(\mathsf{SignKeyGen}, \mathsf{Sign}, \mathsf{Verify})$ with the following properties:

$\mathsf{SignKeyGen}$: outputs a signing key SK and a corresponding verification key VK.

Sign: takes as input a signing key SK and a message m and outputs a signature $\mathfrak{s} := \mathsf{Sign}_{SK}(m)$.

Verify: takes as input a verification key VK, a message m and a purported signature $\mathfrak{s}$. It outputs either $\texttt{reject}$ or $\texttt{accept}$. Denote the output as $\mathsf{Verify}_{VK}(m, \mathfrak{s})$.

We require that for all (VK, SK) output by $\mathsf{SignKeyGen}$, for all m in the message space and for all $\mathfrak{s}$ output by $\mathsf{Sign}_{SK}(m)$, we have $\mathsf{Verify}_{VK}(m, \mathfrak{s}) = \texttt{accept}$.

Definition 3 *A signature scheme* $(\mathsf{SignKeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is a* strongly secure one-time signature scheme *if the success probability of any efficient adversary $\mathcal{A}$ in the following experiment with a challenger is negligible:*

1. *The challenger runs* $\mathsf{SignKeyGen}()$ *to obtain* (VK, SK) *and gives* VK *to* $\mathcal{A}$*.*
2. *$\mathcal{A}$ may output a message m. In this case, the challenger responds with* $\mathfrak{s} := \mathsf{Sign}_{SK}(m)$*. The adversary may choose not to run this step and in this case* $(m, \mathfrak{s})$ *is not defined.*
3. *Following this, $\mathcal{A}$ outputs* $(m^*, \mathfrak{s}^*)$*.*

We say that $\mathcal{A}$ succeeds in the above experiment if $\mathsf{Verify}_{VK}(m^*, \mathfrak{s}^*) = \texttt{accept}$ *and additionally* $(m^*, \mathfrak{s}^*) \neq (m, \mathfrak{s})$ *(in the case when $(m, \mathfrak{s})$ is defined).*

3 Generic Construction of a KDM-CCA2 secure scheme

In this section, we give a generic construction of KDM-CCA2 secure public key encryption scheme. We require the following building blocks. Let $\mathbf{E}_{\text{kdm}}$ be a public key encryption scheme that is KDM-CPA secure with respect to the set of functions $\mathcal{C}$. Let $\mathbf{E}_{\text{cca}}$ be a regular CCA2 secure public key encryption scheme that supports ciphertexts with labels. Let $\mathcal{L}_{\text{eq}}$ be the language consisting of the set of all pairs of ciphertexts that encrypt the same message using $\mathbf{E}_{\text{kdm}}$ and $\mathbf{E}_{\text{cca}}$. Let $\mathbf{P}$ be an NIZK proof system for $\mathcal{L}_{\text{eq}}$ as in Definition 2 and let $\mathbf{S}$ be a one-time signature scheme as in Definition 3.

At a high level, our generic construction for a public key encryption scheme, $\mathbf{E}$, that is KDM-CCA2 secure with respect to $\mathcal{C}$ is similar to the construction of [NY90, DDN91]. To encrypt a message m, we generate a key-pair for the scheme $\mathbf{S}$, encrypt m using both $\mathbf{E}_{\text{kdm}}$ and $\mathbf{E}_{\text{cca}}$, where the label for $\mathbf{E}_{\text{cca}}$ will additionally contain the verification key generated above. Using $\mathbf{P}$, we give a proof that both ciphertexts contain the same plaintext. We then sign the two ciphertexts as well as the proof using $\mathbf{S}$. The ciphertext consists of the verification key, the two ciphertexts, the proof, and the signature.

3.1 Construction

We now formally describe the scheme $\mathbf{E} := (\mathsf{EncKeyGen}, \mathsf{E}, \mathsf{D})$ in detail. Let $\mathbf{E}_{\text{kdm}} := (\mathsf{EncKeyGen}_{\text{kdm}}, \mathsf{E}_{\text{kdm}}, \mathsf{D}_{\text{kdm}})$ (with key pair $(pk_{\text{kdm}}, sk_{\text{kdm}})$) and let $\mathbf{E}_{\text{cca}} := (\mathsf{EncKeyGen}_{\text{cca}}, \mathsf{E}_{\text{cca}}, \mathsf{D}_{\text{cca}})$ (with key pair $(pk_{\text{cca}}, sk_{\text{cca}})$). Let $\mathbf{S} := (\mathsf{SignKeyGen}, \mathsf{Sign}, \mathsf{Verify})$. Let

$$\mathcal{L}_{\text{eq}} := \{(c_1, c_2, \ell) : \exists m, r_1, r_2 \text{ s.t. } c_1 = \mathsf{E}_{\text{kdm}}(pk_{\text{kdm}}, m; r_1) \text{ and } c_2 = \mathsf{E}_{\text{cca}}(pk_{\text{cca}}, m, \ell; r_2)\}.$$

Let $\mathbf{P} := (\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ be an NIZK proof system for $\mathcal{L}_{\text{eq}}$. Note that there maybe be common system parameters that are used to define $\mathbf{E}_{\text{kdm}}, \mathbf{E}_{\text{cca}}$ and $\mathbf{P}$, and these are input to all associated algorithms.

The encryption scheme $\mathbf{E}$ comprises of the following three algorithms.

$\mathsf{EncKeyGen}()$:

1. Run $\mathsf{EncKeyGen}_{\text{kdm}}()$ and $\mathsf{EncKeyGen}_{\text{cca}}()$ to obtain key pairs $(pk_{\text{kdm}}, sk_{\text{kdm}})$ and $(pk_{\text{cca}}, sk_{\text{cca}})$ respectively.
2. Run $\mathsf{CRSGen}()$ to generate the CRS $\mathfrak{C}$ of the NIZK proof system $\mathbf{P}$.

The public key is $pk := (pk_{\text{kdm}}, pk_{\text{cca}}, \mathfrak{C})$. The secret key is $sk := sk_{\text{kdm}}$.

$\mathsf{E}(pk, m, \ell)$:

1. Let $c_{\text{kdm}} := \mathsf{E}_{\text{kdm}}(pk_{\text{kdm}}, m; r_{\text{kdm}})$.
2. Let $\mathsf{SignKeyGen}()$ generate key pair $(VK_{\text{ots}}, SK_{\text{ots}})$.
3. Let $c_{\text{cca}} := \mathsf{E}_{\text{cca}}(pk_{\text{cca}}, m, \ell \| VK_{\text{ots}}; r_{\text{cca}})$.
4. Let $\mathfrak{p}$ be the NIZK proof (using $\mathbf{P}$) for $(c_{\text{kdm}}, c_{\text{cca}}, \ell \| VK_{\text{ots}}) \in \mathcal{L}_{\text{eq}}$.
5. Let $c' := c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p}$ and let $\mathfrak{s} := \mathsf{Sign}_{SK_{\text{ots}}}(c')$.

Then $\mathsf{E}(pk, m, \ell) := c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p} \| VK_{\text{ots}} \| \mathfrak{s}$.

$\mathsf{D}(sk, c, \ell)$:

1. Parse c as $c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p} \| VK_{\text{ots}} \| \mathfrak{s}$. Output `reject` if the ciphertext is not of the right format.
2. If $\mathsf{Verify}_{VK_{\text{ots}}}(c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p}, \mathfrak{s}) = \texttt{reject}$ or $\mathsf{V}(\mathfrak{C}, (c_{\text{kdm}}, c_{\text{cca}}, \ell \| VK_{\text{ots}}), \mathfrak{p}) = \texttt{reject}$, then output `reject`. Otherwise, output $\mathsf{D}_{\text{kdm}}(sk, c_{\text{kdm}})$.

The (perfect) correctness of the public key encryption scheme $\mathbf{E}$ trivially follows from the (perfect) correctness of the scheme $\mathbf{E}_{\text{kdm}}$, (perfect) completeness of the proof system $\mathbf{P}$ and the (perfect) correctness of the signature scheme $\mathbf{S}$.

3.2 Proof of security

Theorem 1 *Let $\mathbf{E}_{\text{kdm}}$ be a* KDM-CPA *secure scheme with respect to the set of functions $\mathcal{C}$. Let $\mathbf{E}_{\text{cca}}$ be a* CCA2 *secure scheme, $\mathbf{S}$ a strong one-time signature scheme and $\mathbf{P}$ an* NIZK *proof system for $\mathcal{L}_{\text{eq}}$. Then $\mathbf{E}$, as constructed above, is a* KDM-CCA2 *secure scheme with respect to $\mathcal{C}$.*

Proof. The proof is through a sequence of games. We first present a schematic description of the sequence of games used to prove that $\mathbf{E}$ is KDM-CCA2 secure. The underlined parts indicate what has changed in each game.

Game	Process encryption query	Process decryption query	justification
0	encrypt (m, m); real $\mathfrak{p}$	decrypt c_{kdm}	
1	encrypt (m, m); real $\mathfrak{p}$	decrypt $\underline{c_{\text{cca}}}$	soundness for $\mathbf{P}$
2	encrypt (m, m); $\underline{\text{fake}}$ $\mathfrak{p}$	decrypt c_{cca}	ZK for $\mathbf{P}$
3	encrypt (m, m); fake $\mathfrak{p}$	decrypt c_{cca}; $\underline{\text{use special rejection}}$	strong one-time sig. $\mathbf{S}$
4	encrypt $(m, \underline{\mathfrak{d}})$; fake $\mathfrak{p}$	decrypt c_{cca}; use special rejection	CCA2 for $\mathbf{E}_{\text{cca}}$
5	encrypt $(m, \mathfrak{d})$; fake $\mathfrak{p}$	decrypt c_{cca}; $\underline{\text{stop special rejection}}$	strong one-time sig. $\mathbf{S}$
6	encrypt $(\underline{\mathfrak{d}}, \mathfrak{d})$; fake $\mathfrak{p}$	decrypt c_{cca}	KDM-CPA for $\mathbf{E}_{\text{kdm}}$
7	encrypt $(\mathfrak{d}, \mathfrak{d})$; $\underline{\text{real}}$ $\mathfrak{p}$	decrypt c_{cca}	ZK for $\mathbf{P}$
8	encrypt $(\mathfrak{d}, \mathfrak{d})$; real $\mathfrak{p}$	decrypt $\underline{c_{\text{kdm}}}$	soundness for $\mathbf{P}$

The sequence of games involving the challenger Ch and adversary $\mathcal{A}$ are more formally described below. Let W_i be the probability with which $\mathcal{A}$ outputs 1 in Game i.

Game 0: This game is the actual attack game, i.e., Experiment 0 in Definition 1.

When responding to an encryption query, Ch encrypts the actual message m using both encryption schemes. The label for E_{cca} additionally contains VK_{ots} which Ch picks using $\mathsf{SignKeyGen}()$. Ch gives a real proof $\mathfrak{p}$ that both encryptions contain the same message. It produces the signature $\mathfrak{s}$ using SK_{ots}.

More formally, Ch computes $c_{\text{kdm}} := \mathsf{E}_{\text{kdm}}(pk_{\text{kdm}}, m; r_{\text{kdm}})$, runs $\mathsf{SignKeyGen}()$ to generate key pair $(VK_{\text{ots}}, SK_{\text{ots}})$, computes $c_{\text{cca}} := \mathsf{E}_{\text{cca}}(pk_{\text{cca}}, m, \ell \| VK_{\text{ots}}; r_{\text{cca}})$ and real proof $\mathfrak{p}$ that $(c_{\text{kdm}}, c_{\text{cca}}, \ell \| VK_{\text{ots}}) \in \mathcal{L}_{\text{eq}}$ using witness $(m, r_{\text{kdm}}, r_{\text{cca}})$. Ch then uses SK_{ots} to sign $c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p}$.

When responding to a decryption query, Ch checks the proof and signature, and decrypts using secret key sk_{kdm}.

Game 1: This game is exactly like Game 0, except that when responding to a decryption query, Ch decrypts using secret key sk_{cca} instead of sk_{kdm}.

More formally, Ch checks the proof and signature but runs $\mathsf{D}(sk_{\text{cca}}, c_{\text{cca}}, \ell \| VK_{\text{ots}})$ to decrypt the message.

It follows from the soundness of the proof system $\mathbf{P}$ that $\mathcal{A}$ cannot distinguish Game 1 from Game 0 except with negligible probability. Hence $|W_1 - W_0|$ is negligible.

Game 2: This game is exactly like Game 1, except that when responding to an encryption query, Ch gives a simulated proof $\mathfrak{p}$ (using the trapdoor of the proof system) instead of a real proof.

More formally, Ch computes $c_{\text{kdm}} := \mathsf{E}_{\text{kdm}}(pk_{\text{kdm}}, m; r_{\text{kdm}})$, runs $\mathsf{SignKeyGen}()$ to generate key pair $(VK_{\text{ots}}, SK_{\text{ots}})$, computes $c_{\text{cca}} := \mathsf{E}_{\text{cca}}(pk_{\text{cca}}, m, \ell \| VK_{\text{ots}}; r_{\text{cca}})$, but computes a simulated proof $\mathfrak{p}$ that $(c_{\text{kdm}}, c_{\text{cca}}, \ell \| VK_{\text{ots}}) \in \mathcal{L}_{\text{eq}}$. In particular, it does not use the witness $(m, r_{\text{kdm}}, r_{\text{cca}})$ to generate $\mathfrak{p}$ but instead uses the trapdoor $(\mathfrak{t})$ of the NIZK proof system. Ch then uses SK_{ots} to sign $c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p}$.

It follows from the zero-knowledge property of $\mathbf{P}$ that $\mathcal{A}$ cannot distinguish Game 2 from Game 1 except with negligible probability. Hence $|W_2 - W_1|$ is negligible.

Game 3: This game is exactly like Game 2, except that when responding to a decryption query of the form (i, c, ℓ) from $\mathcal{A}$ such that $c = c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p} \| VK_{\text{ots}} \| \mathfrak{s}$, Ch first checks if there exists a target tuple of the form (i, c^*, ℓ) with $c^* = c^*_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p}^* \| VK_{\text{ots}} \| \mathfrak{s}^*$, for some $c^*_{\text{kdm}}, \mathfrak{p}^*$ and $\mathfrak{s}^*$. If this is the case, then let c^* be the first such response by Ch. Now if $c^* \neq c$, then Ch rejects the encryption query. We call this the *special rejection rule.*

It follows from the strong one-time security of the signature scheme $\mathbf{S}$ that Ch rejects via the special rejection rule only with negligible probability and hence $\mathcal{A}$ cannot distinguish Game 3 from Game 2 except with negligible probability. Hence $|W_3 - W_2|$ is negligible.

Now, note that if (i, c, ℓ), with $c = c_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p} \| VK_{\text{ots}} \| \mathfrak{s}$, does indeed match a target tuple of the form (i, c^*, ℓ), with $c^* = c^*_{\text{kdm}} \| c_{\text{cca}} \| \mathfrak{p}^* \| VK_{\text{ots}} \| \mathfrak{s}^*$, then either $c^* = c$, in which case Ch does not decrypt the query as it is a target tuple, or Ch rejects the encryption query by the special rejection rule. Hence, either way Ch never decrypts a ciphertext (c_{cca}) that was contained in a target tuple using sk_{cca}. We can now make use of the CCA2 security of $\mathbf{E}_{\mathrm{cca}}$ in the hybrid argument as shown below.

Game 4: This game is exactly like Game 3, except that when responding to an encryption query, Ch encrypts the dummy message $\mathfrak{d}$ using $\mathsf{E}_{\mathrm{cca}}$ but still encrypts the actual message m using $\mathsf{E}_{\mathrm{kdm}}$.

More formally, Ch computes $c_{\mathrm{kdm}} := \mathsf{E}_{\mathrm{kdm}}(pk_{\mathrm{kdm}}, m; r_{\mathrm{kdm}})$, runs $\mathsf{SignKeyGen}()$ to generate key pair $(VK_{\mathrm{ots}}, SK_{\mathrm{ots}})$, computes $c_{\mathrm{cca}} := \mathsf{E}_{\mathrm{cca}}(pk_{\mathrm{cca}}, \mathfrak{d}, \ell \| VK_{\mathrm{ots}}; r_{\mathrm{cca}})$, and computes a simulated proof $\mathfrak{p}$ that $(c_{\mathrm{kdm}}, c_{\mathrm{cca}}, \ell \| VK_{\mathrm{ots}}) \in \mathcal{L}_{\mathrm{eq}}$ using the trapdoor ($\mathfrak{t}$) of the NIZK proof system. Ch then uses SK_{ots} to sign $c_{\mathrm{kdm}} \| c_{\mathrm{cca}} \| \mathfrak{p}$.

It follows from the CCA2 security of $\mathbf{E}_{\mathrm{cca}}$ that $\mathcal{A}$ cannot distinguish Game 4 from Game 3 except with negligible probability. Hence $\left|W_4 - W_3\right|$ is negligible.

Game 5: This game is exactly like Game 4, except that when responding to a decryption query Ch no longer follows the special rejection rule that was defined in Game 3.

It follows from the strong one-time security of the signature scheme $\mathbf{S}$, that $\mathcal{A}$ cannot distinguish Game 5 from Game 4 except with negligible probability. Hence $\left|W_5 - W_4\right|$ is negligible.

Game 6: This game is exactly like Game 5, except that when responding to an encryption query, Ch encrypts the dummy message $\mathfrak{d}$ using both encryption schemes.

More formally, Ch computes $c_{\mathrm{kdm}} := \mathsf{E}_{\mathrm{kdm}}(pk_{\mathrm{kdm}}, \mathfrak{d}; r_{\mathrm{kdm}})$, runs $\mathsf{SignKeyGen}()$ to generate key pair $(VK_{\mathrm{ots}}, SK_{\mathrm{ots}})$, computes $c_{\mathrm{cca}} := \mathsf{E}_{\mathrm{cca}}(pk_{\mathrm{cca}}, \mathfrak{d}, \ell \| VK_{\mathrm{ots}}; r_{\mathrm{cca}})$, and computes a simulated proof $\mathfrak{p}$ that $(c_{\mathrm{kdm}}, c_{\mathrm{cca}}, \ell \| VK_{\mathrm{ots}}) \in \mathcal{L}_{\mathrm{eq}}$ using the trapdoor ($\mathfrak{t}$) of the NIZK proof system. Ch then uses SK_{ots} to sign $c_{\mathrm{kdm}} \| c_{\mathrm{cca}} \| \mathfrak{p}$.

It follows from the KDM-CPA security of $\mathbf{E}_{\mathrm{kdm}}$ that $\mathcal{A}$ cannot distinguish Game 6 from Game 5 except with negligible probability. Hence $\left|W_6 - W_5\right|$ is negligible.

Game 7: This game is exactly like Game 6, except that when responding to an encryption query, Ch gives a real proof $\mathfrak{p}$ that both encryptions contain the same message.

More formally, Ch computes $c_{\mathrm{kdm}} := \mathsf{E}_{\mathrm{kdm}}(pk_{\mathrm{kdm}}, \mathfrak{d}; r_{\mathrm{kdm}})$, runs $\mathsf{SignKeyGen}()$ to generate key pair $(VK_{\mathrm{ots}}, SK_{\mathrm{ots}})$, computes $c_{\mathrm{cca}} := \mathsf{E}_{\mathrm{cca}}(pk_{\mathrm{cca}}, \mathfrak{d}, \ell \| VK_{\mathrm{ots}}; r_{\mathrm{cca}})$ and real proof $\mathfrak{p}$ that $(c_{\mathrm{kdm}}, c_{\mathrm{cca}}, \ell \| VK_{\mathrm{ots}}) \in \mathcal{L}_{\mathrm{eq}}$ using witness $(\mathfrak{d}, r_{\mathrm{kdm}}, r_{\mathrm{cca}})$. Ch then uses SK_{ots} to sign $c_{\mathrm{kdm}} \| c_{\mathrm{cca}} \| \mathfrak{p}$.

It follows from the zero-knowledge property of $\mathbf{P}$ that $\mathcal{A}$ cannot distinguish Game 7 from Game 6 except with negligible probability. Hence $\left|W_7 - W_6\right|$ is negligible.

Game 8: This game is exactly like Game 7, except that when responding to a decryption query, Ch decrypts using secret key sk_{kdm} instead of sk_{cca}.

More formally, Ch checks the proof and signature and runs $\mathsf{D}(sk_{\mathrm{kdm}}, c_{\mathrm{kdm}})$ to decrypt the message.

It follows from the soundness of the proof system $\mathbf{P}$ that $\mathcal{A}$ cannot distinguish Game 8 from Game 7 except with negligible probability. Hence $\left|W_8 - W_7\right|$ is negligible. Game 8 is Experiment 1 in Definition 1.

Combining the different games, we get that $\left|W_8 - W_0\right|$ is negligible which proves Theorem 1. □

Note that we used the computational soundness property of the proof system $\mathbf{P}$ only in Games 1 and 8 and in both these games, Ch only gave real proofs for true statements. Hence "plain" soundness of $\mathbf{P}$ is sufficient and we do not require the proof system to be simulation sound ([Sah99]). In the definition of KDM-CCA2 security, one cannot reduce the attack game to a single encryption query and a single public key. Therefore, one-time zero-knowledge (see remark after Definition 2) would not be sufficient for our proof (one-time zero-knowledge does not imply multi-proof zero-knowledge). However, note that CCA2 security is sufficient, as the "single instance" definition implies the "multi-instance" definition (see remark after Definition 1).

4 Specific number-theoretic instantiation of a KDM-CCA2 secure scheme

In this section, we give specific efficient instantiations of the building blocks used to construct the generic scheme presented in §3. We introduce notation and the number-theoretic assumptions in §4.1. In §4.2, we describe the KDM-CPA scheme of Boneh et al. [BHHO08], while in §4.3, we describe the K-linear version of the Cramer-Shoup CCA2 encryption scheme that we need. In §4.4 and §4.5, we describe the NIZK proof system used to prove equality of plaintexts. We use the efficient strongly one-time signature scheme from [Gro06] (described in §4.6), to complete our instantiation of a KDM-CCA2 secure scheme. In §4.7, we discuss the size of the public key, system parameters and ciphertext of our encryption scheme.

4.1 General notation and Assumptions

Let $\mathbb{G}$ be a group of prime order q. We shall write $\mathbb{G}$ using multiplicative notation. One naturally views $\mathbb{G}$ as a vector space over $\mathbb{Z}_q$, where for $x \in \mathbb{Z}_q$ and $\mathbf{g} \in \mathbb{G}$, the "scalar product" of x and $\mathbf{g}$ is really the power $\mathbf{g}^x$. Because of this, we shall often employ concepts and terminology from linear algebra.

For vectors $\vec{\mathbf{g}} := (\mathbf{g}_1, \ldots, \mathbf{g}_R) \in \mathbb{G}^R$ and $\vec{x} := (x_1, \ldots, x_R) \in \mathbb{Z}_q^R$, define

$$\langle \vec{\mathbf{g}}, \vec{x} \rangle := \mathbf{g}_1^{x_1} \cdot \cdots . \, \mathbf{g}_R^{x_R} \in \mathbb{G}.$$

When we write $\prod_{i=1}^K \vec{\mathbf{g}}_i \in \mathbb{G}^R$ for vectors $\vec{\mathbf{g}}_i \in \mathbb{G}^R$, we mean the component wise product of each of the R terms.

Unless otherwise specified, there is no a priori relation between $\mathbf{g}, \vec{\mathbf{g}}, \mathbf{g}_i$ and $\vec{\mathbf{g}}_i$.

Definition 4 (K-linear assumption [Sha07, Kil07]) *Let $\mathbb{G}$ be a group of prime order q. For a constant $K \geq 1$, the K-linear assumption in $\mathbb{G}$ is defined through the following two experiments (0 and 1) between a challenger and an adversary $\mathcal{A}$. Let W_i be the probability with which $\mathcal{A}$ outputs 1 in Experiment i.*

Experiment 0: *The challenger picks $K+1$ random generators of $\mathbb{G}$: $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{K+1}$, picks random $x_1, \ldots, x_K \in \mathbb{Z}_q$ and sets $x_{K+1} = \sum_{i=1}^K x_i$. $\mathcal{A}$ is given $(\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{K+1}, \mathbf{g}_1^{x_1}, \mathbf{g}_2^{x_2}, \ldots, \mathbf{g}_{K+1}^{x_{K+1}})$ as input.*

Experiment 1: *The challenger picks $K+1$ random generators of $\mathbb{G}$: $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{K+1}$ and picks random $x_1, x_2, \ldots, x_{K+1} \in \mathbb{Z}_q$. $\mathcal{A}$ is given $(\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_{K+1}, \mathbf{g}_1^{x_1}, \mathbf{g}_2^{x_2}, \ldots, \mathbf{g}_{K+1}^{x_{K+1}})$ as input.*

The K-linear assumption holds in $\mathbb{G}$ if for all efficient adversaries $\mathcal{A}$, $\left|\Pr\left[W_0\right] - \Pr\left[W_1\right]\right|$ is negligible.

Another way to understand the K-linear assumption is as follows. Let us define group vectors $\vec{\mathbf{g}}_1, \ldots, \vec{\mathbf{g}}_K \in \mathbb{G}^{K+1}$:

$$\vec{\mathbf{g}}_1 := (\mathbf{g}_1, 1, 1, \ldots, 1, \mathbf{g}_{K+1}), \vec{\mathbf{g}}_2 := (1, \mathbf{g}_2, 1, \ldots, 1, \mathbf{g}_{K+1}), \ldots, \vec{\mathbf{g}}_K := (1, 1, \ldots, 1, \mathbf{g}_K, \mathbf{g}_{K+1}).$$

Let $\mathbb{T}$ denote the subspace of $\mathbb{G}^{K+1}$ generated by $\vec{\mathbf{g}}_1, \ldots, \vec{\mathbf{g}}_K$. The K-linear assumption says that it is hard to distinguish random elements of $\mathbb{T}$ from random elements of $\mathbb{G}^{K+1}$. Note that the standard Decisional Diffie-Hellman (DDH) assumption is the 1-linear assumption and the linear assumption (introduced in [BBS04]) is the 2-linear assumption.

Pairings. Let $\mathbb{G}, \Gamma$ and $\mathbb{G}_{\mathrm{T}}$ be groups of prime order q. We shall use Roman letters to denote elements in $\mathbb{G}$ and Greek letters to denote elements in Γ. A *pairing* is a map $e : \mathbb{G} \times \Gamma \to \mathbb{G}_{\mathrm{T}}$ that satisfies the following properties:

1. e is *bilinear*, which means that for all $\mathbf{a} \in \mathbb{G}$ and $\alpha \in \Gamma$, the maps $e(\mathbf{a}, \cdot) : \Gamma \to \mathbb{G}_{\mathrm{T}}$ and $e(\cdot, \alpha) : \mathbb{G} \to \mathbb{G}_{\mathrm{T}}$ are linear maps;
2. e is *non-degenerate*, which means that its image is not $\{1\}$;
3. e is *efficiently computable.*

4.2 KDM-CPA secure scheme based on the K-linear assumption

In this section, we describe the public key encryption scheme of Boneh et al. [BHHO08] based on the K-linear assumption. Let $N := \lceil (K+2) \log_2 q \rceil$. $\mathbf{E}_{\mathrm{kdm}} = (\mathsf{EncKeyGen}_{\mathrm{kdm}}, \mathsf{E}_{\mathrm{kdm}}, \mathsf{D}_{\mathrm{kdm}})$ is as described below.

$\mathsf{EncKeyGen}_{\mathrm{kdm}}$:

1. Pick random $\vec{\mathbf{g}}_1, \ldots, \vec{\mathbf{g}}_K \in \mathbb{G}^N$.
2. Pick random $\vec{s} \in \{0,1\}^N$.
3. Define $\mathbf{h}_i := \langle \vec{\mathbf{g}}_i, \vec{s} \rangle \in \mathbb{G}$ for $i = 1, \ldots, K$.
4. The secret key is $sk_{\mathrm{kdm}} := \vec{s}$ and the public key is $pk_{\mathrm{kdm}} := (\vec{\mathbf{g}}_1, \ldots, \vec{\mathbf{g}}_K,\ \mathbf{h}_1, \ldots, \mathbf{h}_K)$.

$\mathsf{E}_{\mathrm{kdm}}(pk_{\mathrm{kdm}}, m)$:

1. To encrypt $\mathbf{m} \in \mathbb{G}$, pick random $r_1, \ldots, r_K \in \mathbb{Z}_q$.
2. Compute the ciphertext

$$(\vec{\mathbf{g}}, \mathbf{h}) := \left(\prod_{i=1}^{K} \vec{\mathbf{g}}_i^{r_i},\ \mathbf{m} \cdot \prod_{i=1}^{K} \mathbf{h}_i^{r_i} \right) \in \mathbb{G}^N \times \mathbb{G}.$$

$\mathsf{D}_{\mathrm{kdm}}(sk_{\mathrm{kdm}}, (\vec{\mathbf{g}}, \mathbf{h}))$: Decryption works by setting $\mathbf{m}$ to the right hand side of the following equation:

$$\mathbf{m} = \frac{\mathbf{h}}{\langle \vec{\mathbf{g}}, \vec{s} \rangle} \ .$$

Note that the i^{th} bit s_i of the secret key $\vec{s}$ is encoded for the purpose of encryption as $\mathbf{g}^{s_i}$ for some random (but fixed) $\mathbf{g} \in \mathbb{G}$.

The key space (of encoded secret keys) is $\mathbb{G}^N$. Define a function $f_{\vec{t},\mathbf{b}} : \mathbb{G}^{nN} \to \mathbb{G}$ for fixed $\vec{t} \in \mathbb{Z}_q^{nN}$ and $\mathbf{b} \in \mathbb{G}$ to be the map $f_{\vec{t},\mathbf{b}}(\vec{\mathbf{u}}) := \langle \vec{\mathbf{u}}, \vec{t} \rangle \cdot \mathbf{b}$. Let $\mathcal{C}$ be the set of all functions $f_{\vec{t},\mathbf{b}}$ for all values of $\vec{t} \in \mathbb{Z}_q^{nN}$ and $\mathbf{b} \in \mathbb{G}$. $\mathbf{E}_{\text{kdm}}$ is KDM-CPA secure with respect to the set of functions $\mathcal{C}$ [BHHO08].

Note that [BHHO08] explicitly describes the above scheme in the case $K = 1$, and only briefly mentions its generalization to $K > 1$ (the explicit description of which has been obtained from the authors of [BHHO08] via personal communication).

4.3 CCA2 secure scheme based on the K-linear assumption

In this section, we describe a generalized version of the Cramer-Shoup encryption scheme based on the K-linear assumption. This generalization was described in [HK07b] and [Sha07]. However, given the K-linear decision problem, this scheme is essentially already implicit in [CS02] (based on Theorems 2 and 3, along with Example 1 in §7.4, of the full length version of that paper). This scheme is CCA2 secure and supports ciphertexts with labels. $\mathbf{E}_{\text{cca}} = (\mathsf{EncKeyGen}_{\text{cca}}, \mathsf{E}_{\text{cca}}, \mathsf{D}_{\text{cca}})$ is as described below.

$\mathsf{EncKeyGen}_{\text{cca}}$:

1. Pick random $\mathbf{f}_1, \ldots, \mathbf{f}_{K+1} \in \mathbb{G}$.
2. Pick random $\vec{x}, \vec{y}, \vec{z} \in \mathbb{Z}_q^{K+1}$.
3. Define group vectors $\vec{\mathbf{f}}_1, \ldots, \vec{\mathbf{f}}_K \in \mathbb{G}^{K+1}$:

$$\vec{\mathbf{f}}_1 := (\mathbf{f}_1, 1, 1, \ldots, 1, \mathbf{f}_{K+1}), \vec{\mathbf{f}}_2 := (1, \mathbf{f}_2, 1, \ldots, 1, \mathbf{f}_{K+1}), \ldots, \vec{\mathbf{f}}_K := (1, 1, \ldots, 1, \mathbf{f}_K, \mathbf{f}_{K+1}).$$

4. Define group elements $\mathbf{c}_1, \ldots, \mathbf{c}_K,\ \mathbf{d}_1, \ldots, \mathbf{d}_K,\ \mathbf{e}_1, \ldots, \mathbf{e}_K \in \mathbb{G}$:

$$\mathbf{c}_i := \langle \vec{\mathbf{f}}_i, \vec{x} \rangle,\ \mathbf{d}_i := \langle \vec{\mathbf{f}}_i, \vec{y} \rangle,\ \mathbf{e}_i := \langle \vec{\mathbf{f}}_i, \vec{z} \rangle \quad (i = 1, \ldots, K).$$

5. The secret key is $sk_{\text{cca}} := (\vec{x}, \vec{y}, \vec{z})$ and the public key is $pk_{\text{cca}} := \big(\{\mathbf{f}_j\}_{j=1}^{K+1},\ \{\mathbf{c}_i\}_{i=1}^{K},\ \{\mathbf{d}_i\}_{i=1}^{K},\ \{\mathbf{e}_i\}_{i=1}^{K}\big)$.

$\mathsf{E}_{\text{cca}}(pk_{\text{cca}}, m, \ell)$:

1. To encrypt $\mathbf{m} \in \mathbb{G}$ under label $\ell \in \{0,1\}^*$, pick random $w_1, \ldots, w_K \in \mathbb{Z}_q$.
2. Compute the ciphertext

$$(\vec{\mathbf{f}}, \mathbf{a}, \mathbf{b}) := \left(\prod_{i=1}^{K} \vec{\mathbf{f}}_i^{w_i},\ \mathbf{m} \cdot \prod_{i=1}^{K} \mathbf{c}_i^{w_i},\ \prod_{i=1}^{K} (\mathbf{d}_i \mathbf{e}_i^t)^{w_i} \right) \in \mathbb{G}^{K+1} \times \mathbb{G} \times \mathbb{G},$$

where

$$t := H(\vec{\mathbf{f}}, \mathbf{a}, \ell) \in \mathbb{Z}_q$$

and H is a collision resistant hash function.

$\mathsf{D}_{\text{cca}}(sk_{\text{cca}}, (\vec{\mathbf{f}}, \mathbf{a}, \mathbf{b}), \ell)$:

1. Verify that
$$\mathbf{b} = \langle \vec{\mathbf{f}}, \vec{y} + t\vec{z} \rangle.$$

2. Compute the message $\mathbf{m}$ as the right hand side of the following equation:
$$\mathbf{m} = \frac{\vec{\mathbf{a}}}{\langle \vec{\mathbf{f}}, \vec{x} \rangle}.$$

Note that the schemes in [CS02, Sha07, HK07b] do not explicitly support labels; however, the proof of security immediately generalizes to allow this, provided one assumes (as we do) that H is collision resistant.

4.4 NIZK proofs for satisfiable systems of linear equations over groups

In this section, we describe the NIZK proofs for proving that a system of linear equations over a group is satisfiable. These proofs are derived from Groth and Sahai [GS08]. The paper [GS08] deals with much more general systems of equations; for many applications, such as ours, we only need linear equations. For completeness, and concreteness, we describe how the methods of [GS08] apply to this setting. Our exposition is self contained, but brief.

Let $\mathbb{G}$ be a group of prime order q. A linear equation over $\mathbb{G}$ is an equation of the form
$$\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{X}_j},$$
where $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_W \in \mathbb{G}$ are constants and $\mathtt{X}_1, \ldots, \mathtt{X}_W$ are variables. An *assignment* to the variables is a tuple $(x_1, \ldots, x_W) \in \mathbb{Z}_q^W$, and such an assignment *satisfies* the equation if $\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{x_j}$. A set S of linear equations over $\mathbb{G}$ is called *satisfiable* if there exists an assignment to the variables that simultaneously satisfies each equation in S.

Let $\mathcal{L}_{\text{lsat}}$ be the language of all satisfiable sets of linear equations over $\mathbb{G}$. A witness for membership in $\mathcal{L}_{\text{lsat}}$ is a satisfying assignment. Our goal is to construct an efficient NIZK proof system for $\mathcal{L}_{\text{lsat}}$.

Our proof system for $\mathcal{L}_{\text{lsat}}$ requires a pairing $e : \mathbb{G} \times \Gamma \to \mathbb{G}_{\text{T}}$, where Γ and $\mathbb{G}_{\text{T}}$ are also groups of order q. In addition, we need to make the L-linear assumption in Γ, for some constant L (typically, L is a small constant like 1 or 2, depending on the assumption we make).

- The CRS generator works as follows:

  1. Pick random $\gamma_1, \ldots, \gamma_{L+1} \in \Gamma$.
  2. Define group vectors $\vec{\gamma}_1, \ldots, \vec{\gamma}_L \in \Gamma^{L+1}$:
  $$\vec{\gamma}_1 := (\gamma_1, 1, \ldots, 1, \gamma_{L+1}), \vec{\gamma}_2 := (1, \gamma_2, \ldots, 1, \gamma_{L+1}), \ldots, \vec{\gamma}_L := (1, 1, \ldots, \gamma_L, \gamma_{L+1}).$$
  3. Choose $\vec{\gamma} \in \Gamma^{L+1}$ at random.
  4. The common reference string is $(\gamma_1, \ldots, \gamma_{L+1}, \ \vec{\gamma})$.

- Given a set S of equations, along with a satisfying assignment $(x_1, \ldots, x_W)$, the prover works as follows:

1. Commit to $x_1, \ldots, x_W$:

$$\vec{\delta}_j := \vec{\gamma}^{x_j} \prod_{k=1}^{L} \vec{\gamma}_k^{r_{jk}} \quad (j = 1, \ldots, W)$$

   for random $r_{jk} \in \mathbb{Z}_q$ for $j = 1, \ldots, W$ and $k = 1, \ldots, L$.

2. The proof consists of the commitments $\vec{\delta}_1, \ldots, \vec{\delta}_W$, and in addition, for each equation $\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{x}_j}$ in S, the proof contains L corresponding "proof elements" $\mathbf{p}_1, \ldots, \mathbf{p}_L \in \mathbb{G}$, which are computed as:

$$\mathbf{p}_k := \prod_{j=1}^{W} \mathbf{g}_j^{r_{jk}} \quad (k = 1, \ldots, L).$$

- To verify such a proof, the verifier takes the commitments $\vec{\delta}_1, \ldots, \vec{\delta}_W$, and for each equation $\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{x}_j}$ in S, takes the corresponding proof elements $\mathbf{p}_1, \ldots, \mathbf{p}_L$, and checks that

$$\prod_{j=1}^{W} E(\mathbf{g}_j, \vec{\delta}_j) = E(\mathbf{g}_0, \vec{\gamma}) \prod_{k=1}^{L} E(\mathbf{p}_k, \vec{\gamma}_k). \tag{1}$$

  Here, $E : \mathbb{G} \times \Gamma^{L+1} \to \mathbb{G}_{\mathrm{T}}^{L+1}$ sends $(\mathbf{g}, (\alpha_1, \ldots, \alpha_{L+1}))$ to $(e(\mathbf{g}, \alpha_1), \ldots, e(\mathbf{g}, \alpha_{L+1}))$, which is also a bilinear map.

The CRS contains $2(L+1)$ elements of Γ, and a proof consists of $W(L+1)$ elements of Γ (for the commitments) and $|S|L$ elements of $\mathbb{G}$ (for the proof elements).

We now show that the above proof system has perfect completeness, (statistical) soundness, and computational zero-knowledge.

Perfect completeness

To argue perfect completeness, using bilinearity, one checks by a simple calculation that for any satisfying assignment $(x_1, \ldots, x_W)$, and for any choice of the r_{jk}'s, equation (1) will always be satisfied.

Soundness

A simple fact that will be useful in proving both the soundness and zero-knowledge property is the following, which the reader can easily verify using bilinearity:

Lemma 1 *If $\vec{\beta}_1, \ldots, \vec{\beta}_R \in \Gamma^{L+1}$ are linearly independent, then the map*

$$(\mathbf{h}_1, \ldots, \mathbf{h}_R) \mapsto E(\mathbf{h}_1, \vec{\beta}_1) \cdots E(\mathbf{h}_R, \vec{\beta}_R)$$

is an injective linear map from $\mathbb{G}^R$ into $\mathbb{G}_{\mathrm{T}}^{L+1}$.

To prove soundness, note that with overwhelming probability, the vectors $\vec{\gamma}, \vec{\gamma}_1, \ldots, \vec{\gamma}_L$ form a basis for Γ^{L+1}. Suppose a proof contains commitments $\vec{\delta}_1, \ldots, \vec{\delta}_W \in \Gamma^{L+1}$. Regardless of how these commitments were actually computed, each $\vec{\delta}_j$ can be expressed uniquely as $\vec{\delta}_j = \vec{\gamma}^{x_j} \prod_{k=1}^{L} \vec{\gamma}_k^{r_{jk}}$ for some $x_j, r_{j1}, \ldots, r_{jL} \in \mathbb{Z}_q$. Now consider any particular equation $\mathbf{g}_0^* = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{x}_j}$, and corresponding proof elements $\mathbf{p}_1^*, \ldots, \mathbf{p}_L^*$. Define $\mathbf{g}_0 := \prod_{j=1}^{W} \mathbf{g}_j^{x_j}$ and $\mathbf{p}_k := \prod_{j=1}^{W} \mathbf{g}_j^{r_{jk}}$ for $k = 1, \ldots, L$, using the

x_j's and r_{jk}'s determined as above by the commitments. On the one hand, by perfect completeness, we have

$$\prod_{j=1}^{W} E(\mathbf{g}_j, \vec{\delta}_j) = E(\mathbf{g}_0, \vec{\gamma}) \prod_{k=1}^{L} E(\mathbf{p}_k, \vec{\gamma}_k).$$

On the other hand, if the verification equation (1) holds for the given equation and proof elements, then we also must have

$$\prod_{j=1}^{W} E(\mathbf{g}_j, \vec{\delta}_j) = E(\mathbf{g}_0^*, \vec{\gamma}) \prod_{k=1}^{L} E(\mathbf{p}_k^*, \vec{\gamma}_k).$$

Thus, we have

$$E(\mathbf{g}_0, \vec{\gamma}) \prod_{k=1}^{L} E(\mathbf{p}_k, \vec{\gamma}_k) = E(\mathbf{g}_0^*, \vec{\gamma}) \prod_{k=1}^{L} E(\mathbf{p}_k^*, \vec{\gamma}_k).$$

Applying Lemma 1 to the linearly independent vectors $\vec{\gamma}, \vec{\gamma}_1, \ldots, \vec{\gamma}_L$, we conclude that $\mathbf{g}_0 = \mathbf{g}_0^*$ (and in fact, $\mathbf{p}_k = \mathbf{p}_k^*$ for $k = 1, \ldots, L$). It follows that if the proof verifies, then the assignment $x_1, \ldots, x_W$ determined by the commitments simultaneously satisfies all the given equations.

Zero Knowledge

The simulator generates a "fake CRS" as follows: it generates $\vec{\gamma}_1, \ldots, \vec{\gamma}_L$ as usual, but it computes $\vec{\gamma}$ as $\prod_{k=1}^{L} \vec{\gamma}_j^{s_j}$ for random $s_1, \ldots, s_L \in \mathbb{Z}_q$. The trapdoor for the fake CRS is $(s_1, \ldots, s_L)$.

In a fake CRS, $\vec{\gamma}_1, \ldots, \vec{\gamma}_L$ are linearly independent (with overwhelming probability), while $\vec{\gamma}$ is a random element of the subspace V generated by $\vec{\gamma}_1, \ldots, \vec{\gamma}_L$.

To simulate a proof for a satisfiable set S of linear equations, the simulator starts by setting

$$\vec{\delta}_j := \prod_{k=1}^{L} \vec{\gamma}_k^{r_{jk}} \quad (j = 1, \ldots, W)$$

for random $r_{jk} \in \mathbb{Z}_q$ for $j = 1, \ldots, W$ and $k = 1, \ldots, L$. For each equation $\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{x}_j}$ in S, the simulator generates proof elements $\mathbf{p}_1, \ldots, \mathbf{p}_L$ as follows:

$$\mathbf{p}_k := \mathbf{g}_0^{-s_k} \prod_{j=1}^{W} \mathbf{g}_j^{r_{jk}} \quad (k = 1, \ldots, L).$$

The reader may easily verify, using the bilinearity property, that the verification equation (1) is satisfied.

We now argue that fake proofs are computationally indistinguishable from real proofs. To this end, let us introduce a hybrid prover, which works exactly like a real prover, except that it uses a fake CRS. Such hybrid proofs are computationally indistinguishable from real proofs, under the L-linear assumption for Γ. Moreover, hybrid proofs are statistically indistinguishable from fake proofs. To see this, observe that with overwhelming probability, $\vec{\gamma}_1, \ldots, \vec{\gamma}_L$ are linearly independent. Assuming this is true, in both the hybrid and fake proofs, the distribution of the commitments are the same (uniformly and independently distributed over the subspace V). Additionally, in both types of proofs, the proof elements $\mathbf{p}_1, \ldots, \mathbf{p}_L$ for a given equation are uniquely determined in the same way by the equation, the commitments, and the CRS; indeed, both types of provers generate proof elements that satisfy the verification equation (1); moreover, applying Lemma 1 to the vectors $\vec{\gamma}_1, \ldots, \vec{\gamma}_L$, we see that for a fixed equation, commitments, and CRS, there exist unique $\mathbf{p}_1, \ldots, \mathbf{p}_L$ that satisfy (1).

4.5 NIZK proof for proving equality of plaintext

Given a ciphertext of $\mathbf{E}_{\mathrm{kdm}}$ (from §4.2)

$$(\vec{\mathbf{g}}, \mathbf{h}) \in \mathbb{G}^N \times \mathbb{G}$$

and a ciphertext of $\mathbf{E}_{\mathrm{cca}}$ (from §4.3)

$$(\vec{\mathbf{f}}, \mathbf{a}, \mathbf{b}) \in \mathbb{G}^{K+1} \times \mathbb{G} \times \mathbb{G}$$

with respect to a label $\ell \in \{0,1\}^*$, we want to prove that they are valid encryptions of the same message. This is done by proving that there exist

$$r_1, \dots, r_K,\ w_1, \dots, w_K\ \in \mathbb{Z}_q$$

such that

$$\vec{\mathbf{g}} = \prod_{i=1}^{K} \vec{\mathbf{g}}_i^{r_i},\ \vec{\mathbf{f}} = \prod_{i=1}^{K} \vec{\mathbf{f}}_i^{w_i},\ \mathbf{b} = \prod_{i=1}^{K} (\mathbf{d}_i \mathbf{e}_i^t)^{w_i},\ \mathbf{h}/\mathbf{a} = \prod_{i=1}^{K} \mathbf{h}_i^{r_i} / \prod_{i=1}^{K} \mathbf{c}_i^{w_i},$$

where

$$t := H(\vec{\mathbf{f}}, \mathbf{a}, \ell).$$

This translates into $N + (K+1) + 1 + 1 = N + K + 3$ equations in $2K$ variables. Using the proof system above, this means we need $(2K)(L+1)$ elements of Γ for commitments, and $(N+K+3)L$ elements of $\mathbb{G}$ for the proofs.

4.6 Strongly secure one-time signature scheme

We now describe the strongly secure one-time signature scheme $\mathbf{S}$ from Groth [Gro06].

$\mathsf{SignKeyGen}$: Let $\mathbb{G}$ be a group of prime order q with generator $\mathbf{g}$.

1. Pick $x, y \in \mathbb{Z}_q^*$ and set $\mathbf{f} = \mathbf{g}^x$ and $\mathbf{h} = \mathbf{g}^y$.
2. Pick $r, s \in \mathbb{Z}_q$ and set $\mathbf{c} = \mathbf{f}^r \mathbf{h}^s$.
3. Let H be a collision resistant hash function from $\{0,1\}^*$ to $\mathbb{Z}_q$.
4. The verification key is $VK = (q, \mathbb{G}, \mathbf{g}, \mathbf{f}, \mathbf{h}, \mathbf{c}, H)$ and the secret key $SK = (VK, x, y, r, s)$.

$\mathsf{Sign}_{SK}(m)$: To sign a message $m \in \{0,1\}^*$, pick t at random from $\mathbb{Z}_q$. The signature is $\mathfrak{s} = (t, (x(r-t) + ys - H(m))/y)$.

$\mathsf{Verify}_{VK}(m, \mathfrak{s})$: To verify the signature $\mathfrak{s} = (t, w)$, check that $\mathbf{c} = \mathbf{g}^{H(m)} \mathbf{f}^t \mathbf{h}^w$.

The scheme is strongly one-time secure assuming hardness of computing discrete logs in $\mathbb{G}$ (which follows from the K-linear assumption).

4.7 Size of public key, system parameters and ciphertext

Using the K-linear assumption for $\mathbb{G}$ and the L-linear assumption for Γ, the size of the public key, system parameters and ciphertext are as follows, where $N := \lceil (K+2)\log_2 q \rceil$.

The system parameters consists of the CRS which comprises $2(L+1)$ elements of Γ, the descriptions of $\mathbb{G}, \Gamma, \mathbb{G}_{\mathrm{T}}, e$ and the collision resistant hash function H for $\mathbf{E}_{\mathrm{cca}}$ and $\mathbf{S}$.

The public key of $\mathbf{E}$ consists of $(N+1)K$ elements of $\mathbb{G}$ for the public key pk_{kdm} and $4K+1$ elements of $\mathbb{G}$ for the public key pk_{cca}, for a total of $(N+5)K+1$ elements of $\mathbb{G}$.

The two ciphertexts (c_{kdm} and c_{cca}) require $(N+1)$ and $(K+3)$ elements of $\mathbb{G}$ respectively, giving a total of $N+K+4$ elements of $\mathbb{G}$. To prove equality of plaintexts, we require $(2K)(L+1)$ elements of Γ for commitments, and $(N+K+3)L$ elements of $\mathbb{G}$ for the proofs. Finally, to sign the resulting ciphertexts and proofs using the one-time signature scheme $\mathbf{S}$, we require 4 elements of $\mathbb{G}$ for the verification key VK of $\mathbf{S}$ and 2 elements of $\mathbb{Z}_q$ for the signature.

Note that we can make the public key shorter, by making pk_{cca} as part of the system parameters; indeed, since the secret key sk_{cca} is not needed (other than in the proof of security), one can simply generate all of the group elements appearing in pk_{cca} at random (yielding a distribution that is statistically close to the real distribution on public keys).

We emphasize that, typically, one would set $K = 1, 2$ and $L = 1, 2$, depending on the groups $\mathbb{G}$ and Γ. For example, at one extreme, if $\mathbb{G} = \Gamma$, then one could set $K = L = 2$; at the other extreme, if $\mathbb{G} \neq \Gamma$, and there is no (known) efficiently computable homomorphism from $\mathbb{G}$ to Γ or *vice versa*, then one could set $K = L = 1$.

References

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In *EUROCRYPT 2000*, pages 259–274, 2000.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *CRYPTO 2004*, pages 41–55, 2004.

[BDU08] Michael Backes, Markus Dürmuth, and Dominique Unruh. OAEP is secure under key-dependent messages. In *ASIACRYPT 2008*, December 2008. To appear.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC 1988*, pages 103–112, 1988.

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from Decision Diffie-Hellman. In *CRYPTO 2008*, pages 108–125, 2008.

[Ble98] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the rsa encryption standard pkcs #1. In *CRYPTO 1998*, pages 1–12, 1998.

[BPS07] Michael Backes, Birgit Pfitzmann, and Andre Scedrov. Key-dependent message security under active attacks - BRSIM/UC-soundness of symbolic encryption with key cycles. In *CSF*, pages 112–124, 2007.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *EUROCRYPT 1994*, pages 92–111, 1994.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *Selected Areas in Cryptography*, pages 62–75, 2002.

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT 2001*, pages 93–118, 2001.

[CS98] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO 1998*, pages 13–25, 1998.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for chosen ciphertext secure public key encryption. In *EUROCRYPT 2002*, 2002. Full length version at `http://eprint.iacr.org/2001/085`.

[CS03] Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *CRYPTO 2003*, pages 126–144, 2003.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography (extended abstract). In *STOC 1991*, pages 542–552, 1991.

[FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *FOCS 1990*, pages 308–317, 1990.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC 1982*, pages 365–377, 1982.

[GOS06] Jens Groth, Rafail Ostrovksy, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In *EUROCRYPT 2006*, pages 339–358, 2006.

[Gro06] Jens Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT 2006*, pages 444–459, 2006.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, pages 415–432, 2008.

[HH08] Iftach Haitner and Thomas Holenstein. On the (im)possibility of key dependent encryption. Cryptology ePrint Archive, Report 2008/164, 2008. `http://eprint.iacr.org/`.

[HK07a] Shai Halevi and Hugo Krawczyk. Security under key-dependent inputs. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 466–475, New York, NY, USA, 2007. ACM.

[HK07b] Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In *CRYPTO 2007*, pages 553–571, 2007.

[HU08] Dennis Hofheinz and Dominique Unruh. Towards key-dependent message security in the standard model. In *EUROCRYPT 2008*, pages 108–126, 2008.

[IBM08] IBM. *IBM CCA Basic Services Reference and Guide for the IBM 4758 PCI and IBM 4764 PCI-X Cryptographic Coprocessors: Releases 2.53, 2.54, 3.20, 3.23, 3.24, 3.25, 3.27, and 3.30*, 2008.

[Kil06] Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC 2006*, pages 581–600, 2006.

[Kil07] Eike Kiltz. Chosen-ciphertext secure key encapsulation based on hashed gap decisional Diffie-Hellman. In *PKC 2007*, pages 282–297, 2007.

[LL93] Chae Hoon Lim and Pil Joong Lee. Another method for attaining security against adaptively chosen ciphertext attacks. In *CRYPTO 1993*, pages 420–434, 1993.

[MRY04] Philip D. MacKenzie, Michael K. Reiter, and Ke Yang. Alternatives to non-malleability: Definitions, constructions, and applications (extended abstract). In *TCC 2004*, pages 171–190, 2004.

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC 1990*, pages 427–437, 1990.

[RS91] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO 1991*, pages 433–444, 1991.

[RS06] Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In *EUROCRYPT 2006*, pages 373–390, 2006.

[RSA04] RSA Laboratories. *PKCS #11 v2.20: Cryptographic Token Interface Standard*, 2004.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS 1999*, pages 543–553, 1999.

[SCO$^+$01] Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. Robust non-interactive zero knowledge. In *CRYPTO 2001*, pages 566–598, 2001.

[SG98] Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. In *EUROCRYPT 1998*, pages 1–16, 1998.

[Sha07] Hovav Shacham. A Cramer-Shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. `http://eprint.iacr.org/`.

[Sho01] Victor Shoup. A proposal for an ISO standard for public key encryption (version 2.1). Available on `http://shoup.net/papers/`, 2001.

A Appendix: Simulation Soundness

In our generic construction of a KDM-CCA2 encryption scheme, we combined a KDM-CPA secure encryption scheme with a CCA2 secure encryption scheme, along with an NIZK proof of equality of plaintexts.

In this section, we first briefly observe that instead of using a CCA2 secure encryption scheme, one could instead use just a semantically secure encryption scheme, provided the NIZK proof system is *simulation sound*. Informally, simulation soundness means that an adversary cannot generate a proof of a false statement, even after seeing several simulated proofs of false statements. Formally, the definition is as follows.

Definition 5 (Simulation Soundness [SCO+01]) *An NIZK proof provides* simulation soundness *if an adversary cannot prove a false statement even after seeing simulated proofs for arbitrary statements.*

Let $\mathsf{S} := (\mathsf{S}_1, \mathsf{S}_2)$ *be a simulator running in polynomial time. Consider the following game:*

1. $\mathsf{S}_1()$ *is run to generate CRS* $\mathfrak{C}$ *and trapdoor* $\mathfrak{t}$.
2. $\mathcal{A}$ *is given* $\mathfrak{C}$ *as well as oracle access to* $\mathsf{S}_2(\mathfrak{C}, \mathfrak{t}, \cdot)$.
3. $\mathcal{A}$ *outputs* $(x, \mathfrak{p})$.

Let $\mathcal{Q}$ *be the set of all simulation query and response pairs* $(x_i, \mathfrak{p}_i)$ *made by* $\mathcal{A}$ *to* $\mathsf{S}_2(\mathfrak{C}, \mathfrak{t}, \cdot)$*. We say that* $\mathcal{A}$ *wins the game if* $(x, \mathfrak{p}) \notin \mathcal{Q}$ *and* $x \notin \mathcal{L}$ *and* $\mathsf{V}(\mathfrak{C}, x, \mathfrak{p}) = 1$*. Let* W *be the event that* $\mathcal{A}$ *wins the game. Then for all efficient adversaries* $\mathcal{A}$*, we have* $\Pr[W]$ *is negligible.*

A weaker notion of simulation soundness, called *one-time* simulation soundness [Sah99], allows only a single query to the oracle $\mathsf{S}_2(\mathfrak{C}, \mathfrak{t}, \cdot)$. However, for our application to KDM-CCA2 secure encryption, this weaker notion of one-time simulation soundness is not sufficient, since in the definition of KDM-CCA2 security, one cannot reduce the attack game to a single encryption query and a single public key.

Designing efficient simulation sound NIZK proof systems is more challenging than designing ordinary NIZK proof systems, and given the state of the art in designing CCA2 secure encryption schemes, the construction we presented using NIZK with CCA2 secure encryption seems preferable from a practical point of view. Moreover, we would like to stress the importance of the role that the *label* played in making the construction quite straightforward.

The second goal of this section is to briefly sketch how CCA2 secure encryption can be used to easily construct efficient simulation sound NIZK proof systems for the language $\mathcal{L}_{\text{lsat}}$ of satisfiable systems of linear equations over a group (see §4.4). In fact, since it is no more difficult (either conceptually or practically), we construct a proof system for systems of equations that allow nonlinear as well as linear relations. The construction can also be applied to even more general systems of equations, using the techniques of Groth and Sahai [GS08].

Of course, (one-time) simulation sound NIZK proof systems were initially introduced to construct CCA2 secure encryption schemes from general assumptions (see [Sah99]). However, if we use an appropriate CCA2 secure scheme that supports labels, such as the one discussed in §4.3, we get a particularly simple and efficient simulation sound NIZK proof system for $\mathcal{L}_{\text{lsat}}$.

Finally, we apply the ideas used in the design of the above simulation sound NIZK proof system to obtain a simple and efficient signature based on the K-linear assumption.

A.1 Using simulation sound NIZK to obtain KDM-CCA2

We require the following building blocks. Let $\mathbf{E}_{\text{kdm}}$ be a public key encryption scheme that is KDM-CPA secure with respect to the set of functions $\mathcal{C}$. Let $\mathbf{E}_{\text{cpa}}$ be a regular CPA secure (i.e., semantically secure) encryption scheme. Let $\mathcal{L}_{\text{eq}}$ be the language consisting of the set of all pairs of ciphertexts that encrypt the same message using $\mathbf{E}_{\text{kdm}}$ and $\mathbf{E}_{\text{cpa}}$. Let $\mathbf{P}$ be an NIZK proof system with simulation soundness for $\mathcal{L}_{\text{eq}}$.

The public key for the KDM-CCA2 scheme $\mathbf{E}$ consists of a public key for $\mathbf{E}_{\text{kdm}}$, a public key for $\mathbf{E}_{\text{cpa}}$, and a CRS for $\mathbf{P}$. The encryption algorithm for $\mathbf{E}$ works as follows: a message m is encrypted twice, once using $\mathbf{E}_{\text{kdm}}$ and once using $\mathbf{E}_{\text{cpa}}$; let c_{kdm} and c_{cpa} be the resulting ciphertexts; then the encryption algorithm outputs $(c_{\text{kdm}}, c_{\text{cpa}}, \mathfrak{p})$, where $\mathfrak{p}$ is a proof that c_{kdm} and c_{cpa} encrypt the same plaintext. The decryption algorithm for $\mathbf{E}$ decrypts a ciphertext $(c_{\text{kdm}}, c_{\text{cpa}}, \mathfrak{p})$ as follows: it first verifies the proof $\mathfrak{p}$; if this fails, the decryption algorithm outputs `reject`; otherwise, it decrypts c_{kdm} and outputs the resulting message.

Here is a schematic description of the sequence of games used to prove that $\mathbf{E}$ is KDM-CCA2 secure:

Game	Process encryption query		Process decryption query	justification
0	encrypt (m, m);	real $\mathfrak{p}$	decrypt c_{kdm}	
1	encrypt (m, m);	$\underline{\text{fake}}$ $\mathfrak{p}$	decrypt c_{kdm}	ZK for $\mathbf{P}$
2	encrypt $(m, \underline{\mathfrak{0}})$;	fake $\mathfrak{p}$	decrypt c_{kdm}	CPA for $\mathbf{E}_{\text{cpa}}$
3	encrypt $(m, \mathfrak{0})$;	fake $\mathfrak{p}$	decrypt $\underline{c_{\text{cpa}}}$	sim. sound. for $\mathbf{P}$
4	encrypt $(\underline{\mathfrak{0}}, \mathfrak{0})$;	fake $\mathfrak{p}$	decrypt c_{cpa}	KDM-CPA for $\mathbf{E}_{\text{kdm}}$
5	encrypt $(\mathfrak{0}, \mathfrak{0})$;	$\underline{\text{real}}$ $\mathfrak{p}$	decrypt c_{cpa}	ZK for $\mathbf{P}$
6	encrypt $(\mathfrak{0}, \mathfrak{0})$;	real $\mathfrak{p}$	decrypt $\underline{c_{\text{kdm}}}$	soundness for $\mathbf{P}$

The scheme $\mathbf{E}$ does not support labels. To achieve this, one would need to use an extended notion of simulation sound NIZKs that also support labels.

A.2 Simulation sound NIZKs for satisfiable systems of polynomial equations over a group

In §4.4, we discussed a simple and efficient NIZK proof system for the language $\mathcal{L}_{\text{lsat}}$ of satisfiable systems of linear equations over a group $\mathbb{G}$ of prime order q. Using this proof system as a "black box," we show how to build a simulation sound NIZK for proving the satisfiability of systems of linear equations over a group. Since it is no more difficult, in addition to linear equations, we also allow nonlinear equations. Our system makes use of the CCA2 secure encryption scheme (with labels) discussed in §4.3, which requires the K-linear assumption for $\mathbb{G}$, along with a collision-resistant hash function (into $\mathbb{Z}_q$). We will also need a strongly secure one-time signature scheme, which can also be realized under the same assumptions (see §4.6 — in fact, we need just weaker DL assumption for $\mathbb{G}$). In addition, we will explicitly make use of the CDH assumption for $\mathbb{G}$, which is also implied by the K-linear assumption for $\mathbb{G}$.

Before presenting our main construction below, in §A.2.3, we begin with some preliminary constructions, in §A.2.1 and §A.2.2.

A.2.1 Non-linear equations

Let $\mathcal{L}_{\text{psat}}$ be the language consisting of all satisfiable sets of equations in variables $\mathtt{X}_1, \ldots, \mathtt{X}_W$ (taking values in $\mathbb{Z}_q$), where each equation is either a linear equation over $\mathbb{G}$, which is of the form

$$\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{X}_j},$$

where the $\mathbf{g}_j$'s are constants belonging to the group $\mathbb{G}$, or a nonlinear equation of the form

$$\mathtt{X}_i \mathtt{X}_j = \mathtt{X}_k.$$

Of course, using such nonlinear equations, one can, in effect, build up arbitrary polynomial equations in the variables.

We want to be build an NIZK proof system for $\mathcal{L}_{\text{psat}}$. This is easily done as follows, using an NIZK proof system $\mathbf{P}_{\text{lsat}}$ for $\mathcal{L}_{\text{lsat}}$ as a "black box." In addition to the CRS used by $\mathbf{P}_{\text{lsat}}$, we need an instance of the K-linear decision problem. This consists of random $\mathbf{f}_1, \ldots, \mathbf{f}_{K+1} \in \mathbb{G}$, along with random $\vec{\mathbf{f}} \in \mathbb{G}^{K+1}$.

Define group vectors $\vec{\mathbf{f}}_1, \ldots, \vec{\mathbf{f}}_K \in \mathbb{G}^{K+1}$:

$$\vec{\mathbf{f}}_1 := (\mathbf{f}_1, 1, 1, \ldots, 1, \mathbf{f}_{K+1}), \vec{\mathbf{f}}_2 := (1, \mathbf{f}_2, 1, \ldots, 1, \mathbf{f}_{K+1}), \ldots, \vec{\mathbf{f}}_K := (1, 1, \ldots, 1, \mathbf{f}_K, \mathbf{f}_{K+1}).$$

Our prover works as follows. For each nonlinear equation of the form $\mathtt{XY} = \mathtt{Z}$, we do the following. Suppose that the given satisfying assignment assigns (x, y, z) to $(\mathtt{X}, \mathtt{Y}, \mathtt{Z})$. The proof includes a "commitment" to x:

$$\vec{\mathbf{c}} := \vec{\mathbf{f}}^{x} \prod_{i=1}^{K} \vec{\mathbf{f}}_i^{x'_i} \in \mathbb{G}^{K+1},$$

where $x'_1, \ldots, x'_K$ are random elements of $\mathbb{Z}_q$. Setting $z'_i := x'_i y$ for $i = 1, \ldots, K$, the prover also needs to include a proof that

$$\vec{\mathbf{c}}^{y} = \vec{\mathbf{f}}^{z} \prod_{i=1}^{K} \vec{\mathbf{f}}_i^{z'_i}. \tag{2}$$

Thus, for each such equation, the prover essentially needs to introduce $K + 1$ new constants in $\mathbb{G}$ (for $\vec{\mathbf{c}}$), $2K$ new variables (representing the values $x'_1, \ldots, x'_K$ and $z'_1, \ldots, z'_K$) and $K + 1$ new linear equations over $\mathbb{G}$ (for proving (2)); it then invoke the prover for $\mathbf{P}_{\text{lsat}}$ on the new system of linear equations, using the given assignments to the original variables, and the new assignments to the new variables.

Completeness of the scheme is clear.

Soundness of the scheme follows easily from the fact that $\vec{\mathbf{f}}, \vec{\mathbf{f}}_1, \ldots, \vec{\mathbf{f}}_K$ are linearly independent (with overwhelming probability), and the soundness of $\mathbf{P}_{\text{lsat}}$.

The ZK simulator for the scheme simply replaces each commitment c above by a random element of $\mathbb{G}^{K+1}$, and then runs the ZK simulator for $\mathbf{P}_{\text{lsat}}$. The ZK property follows easily from the K-linear assumption for $\mathbb{G}$ and the ZK property for $\mathbf{P}_{\text{lsat}}$.

A.2.2 OR proofs

Suppose we have a proof system as above for $\mathbf{P}_{\text{psat}}$ for the language $\mathcal{L}_{\text{psat}}$ of satisfiable systems of linear and nonlinear equations over $\mathbb{G}$, as discussed in §A.2.1.

Define $\mathcal{L}_{\text{orsat}}$ to be the language consisting of pairs (S_0, S_1), where either $S_0 \in \mathcal{L}_{\text{psat}}$ or $S_1 \in \mathcal{L}_{\text{psat}}$. A witness for (S_0, S_1) is an assignment to the variables that either satisfies S_0 or satisfies S_1.

There is a trivial reduction from $\mathcal{L}_{\text{orsat}}$ to $\mathcal{L}_{\text{psat}}$. Namely, given a pair (S_0, S_1) of equations involving variables $\mathtt{X}_1, \ldots, \mathtt{X}_W$, we introduce $2W + 1$ new variables $\mathtt{X}_0$ and $\mathtt{X}_{bj}$ for $b = 0, 1$ and $j = 1, \ldots, W$, and generate a set S of equations in these new variables, as follows:

- S includes the nonlinear equation $\mathtt{X}_0^2 = \mathtt{X}_0$ (which forces $\mathtt{X}_0$ to be either 0 or 1);
- for each linear equation $\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{X}_j}$ in S_0, we add the linear equation $\mathbf{g}_0^{1-\mathtt{X}_0} = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{X}_{0j}}$ to S;
- for each nonlinear equation $\mathtt{X}_i\mathtt{X}_j = \mathtt{X}_k$ in S_0, we add the nonlinear equation $\mathtt{X}_{0i}\mathtt{X}_{0j} = \mathtt{X}_{0k}$ to S;
- for each linear equation $\mathbf{g}_0 = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{X}_j}$ in S_1, we add the linear equation $\mathbf{g}_0^{\mathtt{X}_0} = \prod_{j=1}^{W} \mathbf{g}_j^{\mathtt{X}_{1j}}$ to S;
- for each nonlinear equation $\mathtt{X}_i\mathtt{X}_j = \mathtt{X}_k$ in S_1, we add the nonlinear equation $\mathtt{X}_{1i}\mathtt{X}_{1j} = \mathtt{X}_{1k}$ to S.

The reader may verify that $(S_0, S_1) \in \mathcal{L}_{\rm orsat}$ iff $S \in \mathcal{L}_{\rm psat}$. Moreover, the reduction is "witness preserving," in the sense that a witness for an element (S_0, S_1) of $\mathcal{L}_{\rm orsat}$ is easily transformed into a witness for the corresponding element S of $\mathcal{L}_{\rm psat}$. Indeed, for $b = 0, 1$, given an assignment $(x_1, \ldots, x_W)$ to the variables $\mathtt{X}_1, \ldots, \mathtt{X}_W$ that satisfies S_b, we assign the value b to $\mathtt{X}_0$, for $j = 1, \ldots, W$, we assign x_j to $\mathtt{X}_{bj}$ and 0 to $\mathtt{X}_{(1-b)j}$, to obtain a satisfying assignment for S.

The proof system for $\mathcal{L}_{\rm orsat}$ simply transforms $(S_0, S_1) \in \mathcal{L}_{\rm orsat}$ to $S \in \mathcal{L}_{\rm psat}$, as above, and then proves that $S \in \mathcal{L}_{\rm psat}$.

The reductions in this section and §A.2.1 are very much like similar observations made by Groth [Gro06]. Moreover, if we trace through the reductions in this section and §A.2.1, we see that if a given instance (S_0, S_1) of $\mathcal{L}_{\rm orsat}$ involves W variables, $Q_{\rm l}$ linear equations, and $Q_{\rm nl}$ nonlinear equations, the proof system for $\mathcal{L}_{\rm lsat}$ has to deal with a system of equations involving $2W + 1 + 2K(Q_{\rm nl} + 1)$ variables and $Q_{\rm l} + (K + 1)(Q_{\rm nl} + 1)$ linear equations. The new CRS requires just $2K + 2$ elements of $\mathbb{G}$, in addition to the CRS for the proof system for $\mathcal{L}_{\rm lsat}$.

A.2.3 Simulation sound NIZK for $\mathcal{L}_{\rm psat}$

Our goal is to construct an NIZK proof system $\mathbf{P}$ for $\mathcal{L}_{\rm psat}$ that is simulation sound.

Let $\mathbf{P}_{\rm orsat}$ be a proof system for the language $\mathcal{L}_{\rm orsat}$, as discussed in §A.2.2.

We make use of the encryption scheme $\mathbf{E}_{\rm cca}$ described in §4.3. This scheme is CCA2 secure under the K-linear assumption for $\mathbb{G}$ (and a collision resistance assumption for a hash function). In addition, the scheme has a very simple algebraic structure which we can use to our advantage.

Let $\mathbf{S}$ a strong one-time signature scheme, which can be efficiently realized under the same assumptions as in the previous paragraph.

The CRS for $\mathbf{P}$ consists of the CRS for $\mathbf{P}_{\rm orsat}$, together with a public key for $\mathbf{E}_{\rm cca}$, along with three group elements $\mathbf{g}, \mathbf{h}, \mathbf{u} \in \mathbb{G}$. The CRS generator chooses $\mathbf{g}$ and $\mathbf{u}$ at random, and computes $\mathbf{h}$ by choosing $x \in \mathbb{Z}_q$ at random and setting $\mathbf{h} := \mathbf{g}^x$.

At a high level, the proof system works as follows.

Given $S \in \mathcal{L}_{\rm lsat}$, together with a satisfying assignment for S, the prover for $\mathbf{P}$ does the following:

1. generates a signing key SK and corresponding verification key VK for $\mathbf{S}$;
2. encrypts 1 with label VK, using the public key for $\mathbf{E}_{\rm cca}$ in the CRS, yielding a ciphertext c;
3. generates a proof $\mathfrak{p}$ that either
   (a) for some $x \in \mathbb{Z}_q$, $\mathbf{h} = \mathbf{g}^x$ and c is an encryption of $\mathbf{u}^x$ with label VK; or,
   (b) S is satisfiable;
4. computes a signature $\mathfrak{s}$ under SK of $(S, \mathfrak{p})$;
5. outputs the proof $(c, VK, \mathfrak{p}, \mathfrak{s})$.

Note that using the particular scheme $\mathbf{E}_{\rm cca}$, the condition specified in Step 3(a) is equivalent to the condition that there exist $x, w_1, \ldots, w_K \in \mathbb{Z}_q$ such that

$$\mathbf{h} = \mathbf{g}^x,\ \vec{\mathbf{f}} = \prod_{i=1}^{K} \vec{\mathbf{f}}_i^{w_i},\ \mathbf{b} = \prod_{i=1}^{K} (\mathbf{d}_i \mathbf{e}_i^t)^{w_i},\ \mathbf{a} = \mathbf{u}^x \prod_{i=1}^{K} \mathbf{c}_i^{w_i},$$

where the notation for the public key and ciphertext are as in §4.3, and $t := H(\vec{\mathbf{f}}, \mathbf{a}, VK)$. This is an equivalent statement about the satisfiability of $K + 4$ equations in $K + 1$ variables.

Note that to bind a label to a proof, the prover could sign the label, along with S and $\mathfrak{p}$.

To verify such a proof, the prover verifies the proof $\mathfrak{p}$ and signature $\mathfrak{s}$.

The ZK simulator works as follows. The fake CRS for $\mathbf{P}$ is generated in exactly the same way as the real CRS, but the exponent $x \in \mathbb{Z}_q$ is the trapdoor. To generate a fake proof, the fake prover for $\mathbf{P}$ works like the real prover for $\mathbf{P}$, except that in Step 2, it encrypts $\mathbf{u}^x$, and in Step 3, it generates a proof using the witness for the condition in Step 3(a), rather than Step 3(b).

We now argue briefly that this is a simulation sound NIZK proof system for $\mathcal{L}_{\text{psat}}$.

Completeness. This is clear.

Soundness. This follows from the soundness of $\mathbf{P}_{\text{orsat}}$ and the CDH assumption in $\mathbb{G}$: if an adversary could prove a false statement, then he must be able to encrypt a solution $\mathbf{u}^x$ to the instance $(\mathbf{g}, \mathbf{h}, \mathbf{u})$ of the CDH problem. Therefore, by setting up a CRS with known decryption key, we can use this adversary to solve the CDH problem.

Zero Knowledge. Consider a hybrid prover that works like the real prover, but encrypts $\mathbf{u}^x$ instead of 1. By the semantic security of the encryption scheme, the two are computationally indistinguishable. The only difference between the hybrid and fake provers is the choice of witness used to prove the statement in Step 3; as a general rule, zero knowledge implies witness indistinguishability, and so these two provers are also computationally indistinguishable.

Simulation soundness. The attack game here works as follows. The adversary $\mathcal{A}$ obtains fake proofs for statements of his choosing. Let $\mathcal{Q}$ denote the set of all tuples $(S, c, VK, \mathfrak{p}, \mathfrak{s})$, where S was a statement submitted, and $(c, VK, \mathfrak{p}, \mathfrak{s})$ was the corresponding fake proof. To win the game, $\mathcal{A}$ must come up with $S^* \notin \mathcal{L}_{\text{psat}}$, together with a valid proof $(c^*, VK^*, \mathfrak{p}^*, \mathfrak{s}^*)$, such that the tuple $(S^*, c^*, VK^*, \mathfrak{p}^*, \mathfrak{s}^*)$ does not belong to $\mathcal{Q}$. Let W be the event that $\mathcal{A}$ wins.

We want to show $\Pr[W]$ is negligible. To this end, let W_0 be the event that $\mathcal{A}$ wins by producing a tuple of the form $(S^*, c^*, VK^*, \mathfrak{p}^*, \mathfrak{s}^*)$, where $(c^*, VK^*) = (c, VK)$ for some $(S, c, VK, \mathfrak{p}, \mathfrak{s}) \in \mathcal{Q}$, and let W_1 be the event that he wins with a tuple that is not of this form. Clearly, $\Pr[W] = \Pr[W_0] + \Pr[W_1]$, and the assumption that $\mathbf{S}$ is a strongly secure one-time signature scheme implies that $\Pr[W_0]$ is negligible.

To show that $\Pr[W_1]$ is negligible, we define a sequence of games.

Game 1: The challenger generates the CRS as usual, but retains the decryption key for $\mathbf{E}_{\text{cca}}$, in addition to the usual trapdoor x. The challenger generates fake proofs as requested by the adversary, using the value x to encrypt $\mathbf{u}^x$. This process defines the set $\mathcal{Q}$ as above. Finally, when the adversary outputs a tuple $(S^*, c^*, VK^*, \mathfrak{p}^*, \mathfrak{s}^*)$, the challenger checks if $(c^*, VK^*) = (c, VK)$ for some $(S, c, VK, \mathfrak{p}, \mathfrak{s}) \in \mathcal{Q}$; if so, the challenger outputs `bad`; otherwise, using the decryption key, the challenger decrypts c^* with label VK^*, and, using the value x, checks whether the resulting plaintext is equal to $\mathbf{u}^x$; if so, the challenger outputs `good`, and otherwise outputs `bad`. Let W_1' be the event that the challenger outputs `good`. By the soundness property of $\mathbf{P}_{\text{orsat}}$, we have $\Pr[W_1]$ is at most $\Pr[W_1']$ plus a negligible amount.

Game 2: This is the same as Game 1, except that now the challenger uses the ZK simulator for $\mathbf{P}_{\text{orsat}}$ to generate fake proofs $\mathfrak{p}$. Let W_2' be the event that the challenger outputs `good` in this game. By the ZK property for $\mathbf{P}_{\text{orsat}}$, $|\Pr[W_1'] - \Pr[W_2']|$ is negligible.

Game 3: This is the same as Game 2, except that now the challenger encrypts 1 instead of $\mathbf{u}^x$. By the CCA2-security of $\mathbf{E}_{\text{cca}}$, $|\Pr[W_2'] - \Pr[W_3']|$ is negligible. Moreover, under the CDH assumption, $\Pr[W_3']$ is also negligible.

The above argument shows that $\Pr[W_1]$, and hence $\Pr[W]$, is negligible, which proves simulation soundness.

A.3 A signature scheme based on the K-linear assumption

The simulation sound NIZK proof system described in §A.2.3 suggests a simple signature scheme based on the K-linear assumption.

Let $\mathbf{P}_{\text{lsat}}$ be a proof system for the language $\mathcal{L}_{\text{lsat}}$, as discussed in §4.4. Let $\mathbf{E}_{\text{cca}}$ be the CCA2 encryptions scheme described in §4.3.

The key generation algorithm for the signature scheme runs as follows. We generate a CRS $\mathfrak{C}$ for $\mathbf{P}_{\text{lsat}}$, a public key pk for $\mathbf{E}_{\text{cca}}$. We also choose $\mathbf{g}, \mathbf{u} \in \mathbb{G}$ and $x \in \mathbb{Z}_q$ at random, and compute $\mathbf{h} := \mathbf{g}^x$. The public key is $(\mathfrak{C}, pk, \mathbf{g}, \mathbf{h}, \mathbf{u})$, and the secret key consists of x (along with the public key).

To sign a message $m \in \{0,1\}^*$, the signing algorithm encrypts $\mathbf{u}^x$ with label m, yielding a ciphertext c, and generates an NIZK proof $\mathfrak{p}$ that c was correctly computed. The signature consists of $(c, \mathfrak{p})$, and is verified by simply verifying the proof.

Security follows by an argument similar to that in proving the simulation soundness of the proof system in §A.2.3. The reduction to the K-linear assumption is very tight. This scheme is closely related to a signature scheme by Groth [Gro06]; however, by embedding the message to be signed into a label, and using the encryption scheme $\mathbf{E}_{\text{cca}}$, which directly supports labels in a particularly efficient manner, we get a simpler and somewhat more efficient signature scheme. In addition, the message space for Groth's signature scheme is the group $\mathbb{G}$ (or tuples of group elements), which may not always be convenient (in particular, designing hash functions that map arbitrary messages into $\mathbb{G}$ may not always be convenient).