Adaptive partitioning

Dennis Hofheinz*

Karlsruhe Institute of Technology
Dennis.Hofheinz@kit.edu

Abstract. We present a new strategy for partitioning proofs, and use it to obtain new tightly secure encryption schemes. Specifically, we provide the following two conceptual contributions:

- A new strategy for tight security reductions that leads to compact public keys and ciphertexts.
- A relaxed definition of non-interactive proof systems for non-linear ("OR-type") languages. Our definition is strong enough to act as a central tool in our new strategy to obtain tight security, and is achievable both in pairing-friendly and DCR groups.

We apply these concepts in a generic construction of a tightly secure public-key encryption scheme. When instantiated in different concrete settings, we obtain the following:

- A public-key encryption scheme whose chosen-ciphertext security can be tightly reduced to the DLIN assumption in a pairing-friendly group. Ciphertexts, public keys, and system parameters contain 6, 24, and 2 group elements, respectively. This improves heavily upon a recent scheme of Gay et al. (Eurocrypt 2016) in terms of public key size, at the cost of using a symmetric pairing.
- The first public-key encryption scheme that is tightly chosen-ciphertext secure under the DCR assumption. While the scheme is not very practical (ciphertexts carry 28 group elements), it enjoys constant-size parameters, public keys, and ciphertexts.

Keywords: public-key encryption, tight security proofs.

1 Introduction

Tight security. Ideally, the only way to attack a cryptographic scheme S should be to solve a well-investigated, presumably hard computational problem P (such as factoring large integers). In fact, most existing constructions of cryptographic schemes provide such security guarantees, by exhibiting a security reduction. A reduction shows that any attack that breaks the scheme with some probability ε_S implies a problem solver that succeeds with probability ε_P. Of course, we would like ε_P to be as large as possible, depending on ε_S.

Specifically, we could call the quotient $\ell := \varepsilon_S/\varepsilon_P$ the *security loss* of a reduction.[1] A small value of ℓ is desirable, since it indicates a tight coupling of

*Supported by DFG grants HO 4534/4-1 and HO 4534/2-2.

[1] Technically, we also need to take into account the complexity of the attacks on S and P. However, for this exposition, let us simply assume that the complexity of these attacks is comparable.

the security of the scheme to the hardness of the computational problem. It is also desirable that ℓ does not depend, e.g., on the number of considered instances of the scheme. Namely, when ℓ is linear in the number of instances, the scheme's security guarantees might vanish quickly in large settings. This can be a problem when being forced to choose concrete key sizes for schemes in settings whose size is not even known at setup time.

Hence, let us call a security reduction *tight* if its security loss ℓ only depends on a global security parameter (but not, e.g., on the number of considered instances, or the number of usages). Most existing cryptographic reductions are not tight. Specifically, it appears to be a nontrivial problem to construct tightly secure public-key primitives, such as public-key encryption, or digital signature schemes. (A high-level explanation of the arising difficulties can be found in [18].)

Existing work on tight security. The importance of a tight security reduction was already pointed out in 2000 by Bellare, Boldyreva, and Micali [4]. However, the first chosen-ciphertext secure (CCA secure) public-key encryption (PKE) scheme with a tight security reduction from a standard assumption was only proposed in 2012, by Hofheinz and Jager [18]. Their scheme is rather inefficient, however, with several hundred group elements in the ciphertext. A number of more efficient schemes were then proposed in [2, 7, 5, 26, 21, 27, 3, 14, 17, 12]. In particular, Chen and Wee [7] introduced a very useful partitioning strategy to conduct tight security reductions. Their strategy leads to very compact ciphertexts (of as few as 3 group elements [12], plus the message size), but also to large public keys. We will describe their strategy in more detail later, when explaining our techniques. Conversely, Hofheinz [17] presented a different partitioning strategy that leads to compact public keys, but larger ciphertexts (of 60 group elements). We give an overview over existing tightly secure PKE schemes (and some state-of-the-art schemes that are not known to be tightly secure for reference) in Fig. 1.

Our contribution. In this work, we propose a new strategy to obtain tightly secure encryption schemes. This strategy leads to new tightly secure PKE schemes with simultaneously compact public keys and compact ciphertexts (cf. Fig. 1). In particular, our technique yields a practical pairing-based PKE scheme that compares well even with the recent tightly secure PKE scheme of Gay, Hofheinz, Kiltz, and Wee [12]. However, we should also note that our scheme relies on a symmetric pairing (unlike the scheme of [12], which can be instantiated even in DDH groups). Hence, the price we pay for a significantly smaller public key is that the scheme of [12] is clearly superior to ours in terms of computational efficiency. Besides, the use of a symmetric pairing might entail larger group sizes for comparable security.

Our technique also yields the first PKE scheme whose security can be tightly reduced to the Decisional Composite Residuosity (DCR [29]) assumption in groups of the form $\mathbb{Z}_{N^2}^*$ for RSA numbers $N = PQ$. To obtain the DCR instance of our scheme, we also introduce a new type of "OR-proofs" (i.e., a proof system to show disjunctions of simpler statements) in the DCR setting. We give more details on these proofs below.

Scheme	$\lvert pk\rvert$	$\lvert C\rvert - \lvert M\rvert$	sec. loss	assumption	pairing
CS98 [8]	3	3	$\mathbf{O}(q)$	DDH	no
KD04, HK07 [25, 19]	$k+1$	$k+1$	$\mathbf{O}(q)$	k-LIN ($k \geq 1$)	no
HJ12 [18]	$\mathbf{O}(1)$	$\mathbf{O}(\lambda)$	$\mathbf{O}(1)$	DLIN	yes
ADKNO13 [2]	$\mathbf{O}(1)$	$\mathbf{O}(\lambda)$	$\mathbf{O}(1)$	DLIN	yes
HKS15 [21]	$\mathbf{O}(\lambda)$	2	$\mathbf{O}(\lambda)$	subgroup	yes
LPJY15 [26, 27]	$\mathbf{O}(\lambda)$	47	$\mathbf{O}(\lambda)$	DLIN	yes
AHY15 [3]	$\mathbf{O}(\lambda)$	12	$\mathbf{O}(\lambda)$	DLIN	yes
GCDCT16 [14]	$\mathbf{O}(\lambda)$	$6k+4$	$\mathbf{O}(\lambda)$	k-LIN ($k \geq 1$)	yes
H16 [17]	2	60	$\mathbf{O}(\lambda)$	SXDH	yes
GHKW16 [12]	$2k\lambda$	$3k$	$\mathbf{O}(\lambda)$	k-LIN ($k \geq 1$)	no
This work	$2k(k+5)$	$k+4$	$\mathbf{O}(\lambda)$	k-LIN ($k \geq 2$)	yes
CS02 [9]	9	2	$\mathbf{O}(q)$	DCR	—
CS03 [6]	3	2	$\mathbf{O}(q)$	DCR	—
This work	20	28	$\mathbf{O}(\lambda)$	DCR	—

Fig. 1. Comparison of CCA-secure public-key encryption schemes. λ is the security parameter, and q is the number of challenge ciphertexts. The sizes $|pk|$ and $|C|-|M|$ of public key (excluding public parameters) and ciphertext overhead are counted in group elements. For the ciphertext overhead $|C|-|M|$, we do not count smaller components (like MACs) inherited from the used symmetric encryption scheme.

We remark that our main scheme is completely generic, and can be instantiated both with prime-order groups, and in the DCR setting. Only some of our building blocks (such as the "OR-proofs" mentioned above) require setting-dependent instantiations, which we give both in a prime-order, and in the DCR setting.

Hence, we view our main contribution as conceptual. Indeed, in terms of computational efficiency, our encryption schemes do not outperform existing (non-tightly secure) schemes, even when taking into account our tight security reduction in the choice of key sizes. Still, we believe that specializations of our technique can lead to schemes whose efficiency is at least on par with that of existing non-tightly secure schemes.

1.1 Technical overview

Technical goal. To explain our approach, consider the following security game with an adversary $\mathcal{A}$. First, $\mathcal{A}$ obtains a public key, and then may ask for many encryptions of arbitrary messages. Depending on a single bit b chosen by the security game, $\mathcal{A}$ then either always gets an encryption of the desired message, or an encryption of a random message. Also, $\mathcal{A}$ has access to a decryption oracle, and is finally supposed to guess b (i.e., whether the encrypted ciphertexts contain the desired, or random messages). If no efficient $\mathcal{A}$ can predict b non-negligibly better than guessing, the used PKE scheme is considered CCA secure in the multi-challenge setting. Note that regular (i.e., single-challenge) CCA security implies CCA security in the multi-challenge setting using a hybrid argument

(over the challenge encryptions $\mathcal{A}$ gets), but this hybrid argument incurs a large security loss. Hence, the difficulty in proving multi-challenge security is to randomize many challenge ciphertexts in as few steps as possible.

General paradigm. All of the mentioned works on tightly secure PKE follow a general paradigm. Namely, in these schemes, each ciphertext $C = (c, \pi)$ carries some kind of "consistency proof" π that the plaintext message encrypted in c is intact. What this concretely means varies in different schemes. For instance, in some works [18, 2, 26, 27, 17], π is explicit and proves knowledge of the plaintext *or* of a valid signature on c. In other works [7, 5, 21, 3, 14, 12], π is implicit, and proves knowledge of the plaintext *or* of a special authentication tag for that ciphertext. All of these works, however, use π to enable the security reduction to get leverage over the adversary $\mathcal{A}$, as follows. For instance, in the signature-based works above, the security reduction will be able to produce proofs π for ciphertexts with unknown plaintexts (by proving knowledge of a signature), while an adversary can only construct proofs from which the plaintext can be extracted. This enables the security reduction to implement a decryption oracle, while being able to randomize plaintexts encrypted for $\mathcal{A}$.

Chen and Wee's approach. Chen and Wee [7] implement the above approach with an economic partitioning strategy (that in turn draws from an argument of Naor and Reingold [28]). Specifically, in their scheme, π implicitly proves knowledge of the plaintext *or* of a special tag T. Initially, T is constant, and committed to in the public key. In their security analysis, Chen and Wee introduce dependencies of T on the corresponding c. Specifically, in the i-th step of their analysis, they set $T = \mathbf{F}(\tau_{..i})$, where $\mathbf{F}$ is a random function, and $\tau_{..i}$ is the i-bit prefix of the hash τ of c. After a small number of such steps, T is a random value that is individual to each ciphertext. At this point, T is unpredictable for $\mathcal{A}$ on fresh ciphertexts, and hence $\mathcal{A}$'s decryption queries must prove knowledge of the respective plaintext. At the same time, the security game (which defines $\mathbf{F}$) can also prepare valid ciphertexts with unknown messages, and thus randomize all challenge ciphertexts at once.

We call the approach of Chen and Wee a partitioning strategy, since each hybrid step above proceeds as follows:

1. Partition the ciphertext space into two halves (in this case, according to the i-th bit of τ).
2. Change the definition of the "authentication tag" T for all ciphertexts from one half. (Keep the authentication tag for ciphertexts from the other half unchanged.)

In particular, the second step introduces an additional dependency of T on the bit τ_i. Most existing works use a partitioning strategy based on the individual bits of (the hash of) the ciphertext. An exception is the recent work [17], which implements a similar strategy based on an algebraic predicate of the ciphertext. This latter approach leads to shorter public keys, but requires relatively complex proofs π, and thus not only entails larger ciphertexts, but also requires a pairing.

Our approach. Here, we also follow the generic paradigm sketched above, but refine the partitioning strategy of Chen and Wee. Namely, instead of partitioning

the ciphertext space statically (e.g., through the hash of c), we add a special (encrypted) bit to π that determines the half in which the corresponding ciphertext is supposed to be. In contrast to previous works, that bit is not always known, not even to the security reduction itself. This change has several consequences:

- The bit that determines the partitioning in each ciphertext is easily accessible with a suitable decryption key, and so leads to a simple consistency proof π (and thus small ciphertexts). (This is in contrast to the scheme from [17], which proves complex statements in π.)
- The partitioning bit can by changed dynamically in challenge ciphertexts in different steps of the proof. Hence, a single "bit slot" can be used to partition the ciphertext space in many different ways during the proof. Eventually, this leads to compact public keys, since only few statements (about this single bit slot) need to be proven. (This is in contrast to partitioning schemes in which one proof for each bit position is generated.)
- However, since also the adversary can dynamically determine the partitioning of his ciphertexts from decryption queries, the security analysis becomes more complicated. Specifically, the reduction must cope with a situation in which an adversary submits a ciphertext for which the partitioning bit is not known.

In particular the last consequence will require additional measures in our security analysis. Namely, we will in some cases need to accept several authentication tags T in $\mathcal{A}$'s decryption queries, simply because we do not know in which half of the partitioning the corresponding ciphertext is. In fact, we will not be able to force $\mathcal{A}$ to use "the right" authentication tag in his decryption queries. We will only be able to force $\mathcal{A}$ to use an authentication tag T from a previous challenge ciphertext (since all other tags are unpredictable to $\mathcal{A}$). Hence, in order to eventually exclude that $\mathcal{A}$ produces ciphertexts without a proof of knowledge of the corresponding plaintext, we will need to work a bit more.

At this point, our main conceptual idea will be to introduce a dependency of T on a suitable value τ that is individual to each ciphertext. (While the construction in our scheme is slightly more complicated, one can think of τ as being simply the hash of the ciphertext.) Hence, in the first part of our analysis, we force $\mathcal{A}$ to reuse a tag T from a previous challenge ciphertext, while we tie this T to a ciphertext-unique value τ in the second part. When this is done, $\mathcal{A}$'s proofs π from decryption queries must prove knowledge of the encrypted plaintext message, *or* break the collision-resistance of the used hash function. Since the hash function will be assumed to be collision-resistant, $\mathcal{A}$ must prove knowledge of the respective plaintext in each decryption query. Hence, we can proceed with a proof of CCA security as in previous schemes.

Building blocks. To implement our strategy, we require a variety of building blocks. Specifically, like previous works, we require re-randomizable (chosen-plaintext-secure) encryption, and universal hash proof systems for linear languages. We also require tightly secure one-time signatures, for which we give the first construction in the DCR setting. However, apart from our new partitioning strategy, the main technical innovation from our work is the construction of a

non-interactive proof system for disjunctions (of simpler statements) in the DCR setting.

Namely, our proof system allows to prove that, given two ciphertexts c_1, c_2, at least one of them decrypts to zero. (In fact, the syntactics are a little more complicated, and in particular, honest proofs can only be formulated when the *first* ciphertext decrypts to zero. However, proofs that *one* of the two ciphertexts decrypts to zero can always be simulated using a special trapdoor, and we have soundness even in the presence of such simulated proofs.)

Such a proof system for disjunctions already exists in pairing-friendly groups (see [1]). A construction without pairings is far from obvious, though. Intuitively, the reason is that the language of pairs (c_1, c_2) as above (with at least one c_i that encrypts zero) is not closed under addition (of the respective plaintexts). Hence, disjunctions as above do not correspond to linear languages, and most common constructions (e.g., for universal hash proof systems [9, 23]) do not apply. Our DCR-based construction thus is not linear, and relies on new techniques.

Concretely, our proof system can be viewed as a randomized variant of a universal hash proof system. Namely, depending on how many of the c_i do not encrypt zero, a valid proof reveals zero, one, or two linear equations about the secret verification key of our system. However, proofs in our system are randomized, and the revealed equations are also blinded with precisely one random value. Hence, up to one equation about the secret key is completely blinded. But as soon as both c_i encrypt nonzero values, a valid proof contains nontrivial information about the secret key. Thus, such proofs cannot be produced by an adversary who only sees proofs for valid statements (with at least one c_i that encrypts zero). Soundness follows as with regular universal hash proof systems.

Roadmap and additional content in full version

In Section 2, we recall some basic notation and definitions. In Section 3, we formulate an algebraic setting that allows to express both our DLIN-based and DCR-based schemes in a generic way. In Section 4, we recall some existing and construct some new necessary tightly secure building blocks. In Section 5, we introduce our notion of "benign" proof systems, and our DCR-based benign proof system for "OR"-like languages. Finally, in Section 6, we describe our new generic key encapsulation scheme.

Unfortunately, our work requires several rather technical concepts, and we need to outsource several proofs and additional discussion into the full version [16] of this paper. In particular, in [16], we discuss the security of our scheme in the multi-user setting, analyze its performance, and suggest optimizations. In the full version, we also present a new DCR-based tightly secure one-time signature scheme (which constitutes a technical building block for our main encryption scheme). Moreover, we present details for "more conventional" benign proof systems, and full details of the proof for our encryption scheme.

Acknowledgements. I would like to thank Antonio Faonio for pointing out a problem in the formulation of Definition 8, and Dingding Jia and Ryo Nishimaki for a careful proofreading. In particular, Dingding spotted a mistake in

the description of honest key derivation. I am also indebted to Lin Lyu, who found a flaw in an earlier version of the DCR-based one-time signature scheme $\mathbf{OTS_{DCR}}$, a gap in the proof of a technical lemma from the main proof, and many smaller mistakes in an earlier version in a very thorough proofreading. Finally, I would like to thank the reviewers for helpful comments concerning the presentation.

2 Preliminaries

Notation and conventions. For a group $\mathbb{G}$ of order $|\mathbb{G}|$, a group element $g \in \mathbb{G}$, and a vector $\mathbf{u} = (u_1, \ldots, u_n)^\top \in \mathbb{Z}_{|\mathbb{G}|}^n$, we write $g^{\mathbf{u}} := (g^{u_1}, \ldots, g^{u_n})^\top \in \mathbb{G}^n$. Similarly, we define $g^{\mathbf{M}} \in \mathbb{G}^{n \times m}$ for matrices $\mathbf{M} \in \mathbb{Z}_{|\mathbb{G}|}^{n \times m}$. For integers $x, N \in \mathbb{Z}$ with $N > 0$, we define $[x]_N := x \bmod N$, and $[x]^N$ to be the unique integer with $x = [x]_N + N \cdot [x]^N$. Furthermore, we define the "absolute modular value" $|x|_N$ through

$$|x|_N \;:=\; \begin{cases} [x]_N & \text{if } [x]_N < N/2 \\ [-x]_N & \text{if } [x]_N \geq N/2, \end{cases}$$

such that $0 \leq |x|_N \leq N/2$ in any case. Finally, we let $\left(\frac{x}{N}\right)$ denote the Jacobi symbol of x modulo N. For a bit $b \in \{0,1\}$, we denote with $\overline{b} = 1 - b$ the complement of b. For a bitstring $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, we denote with $x_{..i} = (x_1, \ldots, x_i)$ the i-bit prefix of x, and with $x_{i..} = (x_i, \ldots, x_n)$ the $(n-i+1)$-bit postfix of x. For random variables $X, Y \in \{0,1\}^*$, we let $\mathbf{SD}\big(X\,;\,Y\big)$ denote their statistical distance, and $H_\infty(X)$ the min-entropy of X.

Global public parameters. To simplify notation, we assume that all algorithms in this work (including adversaries) implicitly receive public parameters pp as input. In our case, these public parameters will contain the description of algebraic groups and related algorithms, and a collision-resistant and a universal hash function. We give more details on these parameters when we discuss the algebraic setting, collision-resistant hashing, and our key extractor (which uses the universal hash function).

Collision-resistant hashing. We require collision-resistant hashing:

Definition 1 (Collision-resistant hashing). *A hash function generator is a PPT algorithm* **CRHF** *that, on input 1^λ, outputs (the description of) an efficiently computable function $H : \{0,1\}^* \to \{0,1\}^{\ell_H}$. We say that* **CRHF** *outputs collision-resistant hash functions H (or, slightly abusing notation, that* **CRHF** *is collision-resistant), if*

$$\mathrm{Adv}^{\mathrm{crhf}}_{\mathbf{CRHF},\mathcal{A}}(\lambda) \;=\; \Pr\left[x \neq x' \;\wedge\; H(x) = H(x') \;\middle|\; (x, x') \leftarrow \mathcal{A}(1^\lambda, H)\right]$$

(where $H \leftarrow \mathbf{CRHF}(1^\lambda)$) is negligible for every PPT adversary $\mathcal{A}$.

We assume that the public parameters pp contain a function H sampled with a hash function generator **CRHF**.

Universal hashing, and randomness extraction. We also assume a family $\mathbf{UHF} = \mathbf{UHF}_\lambda$ of universal hash functions $h : \{0,1\}^* \to \{0,1\}^\lambda$. Since universal hash functions are good randomness extractors, we in particular have that for any random variable X with min-entropy $H_\infty(X) \geq 3\lambda$,

$$\mathbf{SD}\big((h,\, h(X))\,;\,(h,R)\big) \;\leq\; 1/2^\lambda,$$

where $h \in \mathbf{UHF}_\lambda$ and $R \in \{0,1\}^\lambda$ are uniformly chosen.

Key encapsulation mechanisms, and multi-challenge security. A key encapsulation mechanism (KEM) **KEM** consists of PPT algorithms ($\mathbf{Gen}, \mathbf{Enc}$, $\mathbf{Dec}$). Key generation $\mathbf{Gen}(1^\lambda)$ outputs a public key pk and a secret key sk. Encapsulation $\mathbf{Enc}(pk)$ takes a public key pk, and outputs a ciphertext c, and a session key K. Decapsulation $\mathbf{Dec}(sk, c)$ takes a secret key sk, and a ciphertext c, and outputs a session key K. For correctness, we require that for all (pk, sk) in the range of $\mathbf{Gen}(1^\lambda)$, and all (c, K) in the range of $\mathbf{Enc}(pk)$, we always have $\mathbf{Dec}(sk, c) = K$. Security is defined as follows:

Definition 2 (Multi-challenge ciphertext indistinguishability). *Given a key encapsulation scheme* **KEM***, consider the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:*

1. *$\mathcal{C}$ samples a keypair through $(pk, sk) \leftarrow \mathbf{Gen}(1^\lambda)$, and chooses a uniform bit $b \leftarrow \{0,1\}$.*
2. *$\mathcal{A}$ is invoked on input $(1^\lambda, pk)$, and with (many-time) access to the following oracles:*
   - *$\mathcal{O}_{\mathbf{enc}}()$ runs $(c, K) \leftarrow \mathbf{Enc}(pk)$, sets $K_0 = K$, samples a fresh $K_1 \leftarrow \{0,1\}^\lambda$, and returns (c, K_b).*
   - *$\mathcal{O}_{\mathbf{dec}}(c)$ returns $\perp$ if c is a previous output of $\mathcal{O}_{\mathbf{enc}}$. Otherwise, $\mathcal{O}_{\mathbf{dec}}$ returns $K \leftarrow \mathbf{Dec}(sk, c)$.*
3. *Finally, $\mathcal{A}$ outputs a bit b', and $\mathcal{C}$ outputs 1 iff $b = b'$.*

Let

$$\mathrm{Adv}^{\mathrm{mcca}}_{\mathbf{KEM},\mathcal{A}}(\lambda) \;=\; \Pr\left[\mathcal{C} \text{ outputs } 1\right] - 1/2.$$

We say that **KEM** *has indistinguishable ciphertexts under chosen-ciphertext attacks in the multi-challenge setting (short: is IND-MCCA secure) if and only if $\mathrm{Adv}^{\mathrm{mcca}}_{\mathbf{KEM},\mathcal{A}}(\lambda)$ is negligible for all PPT $\mathcal{A}$.*

Secure KEM schemes imply secure PKE schemes [8], and the corresponding security reduction is tight also in the multi-challenge setting. Hence, like [12], we will focus on obtaining an IND-MCCA secure KEM scheme in the following.

3 The generic algebraic setting

3.1 The generic setting

Groups and public parameters In the following, let $\mathbb{G}$ be a group of order $|\mathbb{G}|$. We require that $|\mathbb{G}|$ is square-free, and only has prime factors larger than 2^λ. Furthermore, we assume two subgroups $\mathbb{G}_1, \mathbb{G}_2 \subseteq \mathbb{G}$ of order $|\mathbb{G}_1|$ and $|\mathbb{G}_2|$,

respectively, and such that $\mathbb{G}_1 \cdot \mathbb{G}_2 = \{h_1 \cdot h_2 \mid h_1 \in \mathbb{G}_1, h_2 \in \mathbb{G}_2\} = \mathbb{G}$. Note that we neither require nor exclude that $|\mathbb{G}|$ (or $|\mathbb{G}_1|$ or $|\mathbb{G}_2|$) is prime, or that $\mathbb{G}_1 \cap \mathbb{G}_2$ is trivial.

We assume that the global public parameters pp include

- (descriptions of) $\mathbb{G}$, $\mathbb{G}_1$, and $\mathbb{G}_2$,
- fixed generators g of $\mathbb{G}$, g_1 of $\mathbb{G}_1$ and g_2 of $\mathbb{G}_2$,
- the group order $|\mathbb{G}_2|$ of $\mathbb{G}_2$,
- a positive integer $\ell_{\mathbf{B}}$, and a matrix $g_1^{\mathbf{B}}$, for $\mathbf{B} \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}} \times \ell_{\mathbf{B}}}$.[2]

We stress that these parameters may depend on λ, and note that $|\mathbb{G}|$, $|\mathbb{G}_1|$, and $\mathbf{B}$ do not need to be public. However, we do require that there are efficient algorithms for the following tasks:

- performing the group operation in $\mathbb{G}$,
- sampling uniformly distributed $\mathbb{Z}_{|\mathbb{G}_1|}$-elements,
- recognizing $\mathbb{G}$ (i.e., deciding group membership in $\mathbb{G}$).

Since we assume $|\mathbb{G}_2|$ to be public, we also have algorithms for deciding membership in $\mathbb{G}_2$, and for uniformly sampling from $\mathbb{Z}_{|\mathbb{G}_2|}$ and $\mathbb{G}_2$, and thus also from $\mathbb{Z}_{|\mathbb{G}|}$ and $\mathbb{G}$.

Computational assumptions In our generic setting, we will use an assumption that can be seen as a combination of the Extended Decisional Diffie-Hellman assumption from [15], and the Matrix Decisional Diffie-Hellman assumption from [10].

Definition 3 (Generalized DDH, combining [15, 10]). *We say that the Generalized Decisional Diffie-Hellman (GDDH) assumption holds in our setting if the following advantage is negligible for every PPT adversary $\mathcal{A}$, and for uniformly chosen $\boldsymbol{\omega}, \mathbf{r} \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$:*

$$\mathrm{Adv}^{\mathrm{gddh}}_{\mathbb{G},\mathcal{A}}(\lambda) = \frac{1}{2}\big(\Pr\left[\mathcal{A}(1^\lambda, g_1^{\boldsymbol{\omega}^\top \mathbf{B}}, g_1^{\mathbf{B}\mathbf{r}}, g_1^{\boldsymbol{\omega}^\top \mathbf{B}\mathbf{r}}) = 1\right] - \Pr\left[\mathcal{A}(1^\lambda, g_1^{\boldsymbol{\omega}^\top \mathbf{B}}, g_1^{\mathbf{B}\mathbf{r}}, g_1^{\boldsymbol{\omega}^\top \mathbf{B}\mathbf{r}} g_2) = 1\right]\big).$$

Besides GDDH, we will also assume that it is infeasible to find a nontrivial element $g_2^u \in \mathbb{G}_2$ that does not already generate $\mathbb{G}_2$:

Definition 4 ($\mathbb{G}_2$-factoring assumption). *We say that the factoring $\mathbb{G}_2$ is hard in our setting if the following advantage is negligible for every PPT adversary $\mathcal{A}$ whose output $(g_2^{u_1}, \ldots, g_2^{u_q}) \in \mathbb{G}_2^q$ is always a vector of $\mathbb{G}_2$-elements:*

$$\mathrm{Adv}^{\mathrm{fact}}_{\mathbb{G}_2,\mathcal{A}}(\lambda) = \Pr\left[\exists i : \gcd(|\mathbb{G}_2|, u_i) \notin \{1, |\mathbb{G}_2|\} \;\middle|\; (g_2^{u_1}, \ldots, g_2^{u_q}) \leftarrow \mathcal{A}(1^\lambda)\right].$$

[2] How $\ell_{\mathbf{B}}$ and $\mathbf{B}$ are chosen depends in the concrete instance. In the prime-order setting, $\ell_{\mathbf{B}}$ and $\mathbf{B}$ determine what concrete computational problem is reduced to. Conversely, in the DCR setting, $\ell_{\mathbf{B}} = 1$, and $\mathbf{B} = 1$ is trivial.

Generalized ElGamal encryption To simplify our notation, and to structure our presentation, we consider the following generalized variant of ElGamal:

Keypairs. Keypairs (epk, esk) are of the form $(epk, esk) = (g_1^{\boldsymbol{\omega}^\top \mathbf{B}}, \boldsymbol{\omega})$ for $\boldsymbol{\omega} \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$.

Encryption. To encrypt $u \in \mathbb{Z}_{|\mathbb{G}_2|}$ with random coins $\mathbf{r} \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$, compute

$$\mathbf{E}_{epk}(u; \mathbf{r}) \;=\; \mathbf{c} \;=\; (\mathbf{c}_0, c_1) \;=\; (g_1^{\mathbf{Br}}, g_1^{\boldsymbol{\omega}^\top \mathbf{Br}} g_2^u) \;\in \mathbb{G}^{\ell_{\mathbf{B}}} \times \mathbb{G}.$$

If we omit $\mathbf{r}$ and only write $\mathbf{E}_{epk}(u)$, then $\mathbf{r}$ is implicitly chosen uniformly from $\mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$.

Decryption. A ciphertext $\mathbf{c} = (\mathbf{c}_0, c_1) = (g^{\boldsymbol{\gamma}}, g^{\delta})$ is decrypted to

$$\mathbf{D}_{esk}(\mathbf{c}) \;=\; g^{\delta - \boldsymbol{\omega}^\top \boldsymbol{\gamma}} \;\in \mathbb{G}.$$

Note that we encrypt exponents, while decryption only retrieves the respective group element.

It will also be useful to generalize this encryption to vectors of plaintexts with reused random coins: for $\mathbf{pk} = (epk_1, \ldots, epk_n)$ and $\mathbf{sk} = (esk_1, \ldots, esk_n)$ with $(epk_i, esk_i) = (g_1^{\boldsymbol{\omega}_i^\top \mathbf{B}}, \boldsymbol{\omega}_i)$, and $\mathbf{u} = (u_1, \ldots, u_n) \in \mathbb{Z}_{|\mathbb{G}_2|}^n$, let

$$\begin{aligned}
\mathbf{E}_{\mathbf{pk}}(\mathbf{u}; \mathbf{r}) \;&=\; (\mathbf{c}_0, (c_1, \ldots, c_n)) \\
&=\; (g_1^{\mathbf{Br}}, (g_1^{\boldsymbol{\omega}_1^\top \mathbf{Br}} g_2^{u_1}, \ldots, g_1^{\boldsymbol{\omega}_n^\top \mathbf{Br}} g_2^{u_n})) \;\in \mathbb{G}^{\ell_{\mathbf{B}}} \times \mathbb{G}^n \\
\mathbf{D}_{\mathbf{sk}}(\mathbf{c}) \;&=\; (g^{\delta_1 - \boldsymbol{\omega}_1^\top \boldsymbol{\gamma}}, \ldots, g^{\delta_n - \boldsymbol{\omega}_n^\top \boldsymbol{\gamma}}) \;\in \mathbb{G}^n \text{ for } \mathbf{c} = (g^{\boldsymbol{\gamma}}, (g^{\delta_1}, \ldots, g^{\delta_n})).
\end{aligned}$$

When no confusion is possible, we may write $(\mathbf{c}_0, c_1, \ldots, c_n)$ instead of the more cumbersome $(\mathbf{c}_0, (c_1, \ldots, c_n))$. Sometimes, it will also be convenient to write $\boldsymbol{\Omega} = (\boldsymbol{\omega}_1 || \ldots || \boldsymbol{\omega}_n) \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}} \times n}$, such that $\mathbf{pk} = g_1^{\boldsymbol{\Omega}^\top \mathbf{B}}$ and

$$\begin{aligned}
\mathbf{E}_{\mathbf{pk}}(\mathbf{u}; \mathbf{r}) \;&=\; (g_1^{\mathbf{Br}}, g_1^{\boldsymbol{\Omega}^\top \mathbf{Br}} g_2^{\mathbf{u}}) \\
\mathbf{D}_{\mathbf{sk}}(\mathbf{c}) \;&=\; g^{\boldsymbol{\gamma} - \boldsymbol{\Omega}^\top \boldsymbol{\delta}} \quad \text{for} \quad \mathbf{c} = (g^{\boldsymbol{\gamma}}, g^{\boldsymbol{\delta}}) \in \mathbb{G}^{\ell_{\mathbf{B}}} \times \mathbb{G}^n.
\end{aligned}$$

While this variant of ElGamal encryption will mainly be a notational tool, it is also a very simple tightly (chosen-plaintext) secure encryption scheme:

Definition 5 (IND-MCCPA security game for $(\mathbf{E}, \mathbf{D})$). *Consider the following game (which we call the IND-MCCPA security game, for "indistinguishability against multiple (partial) corruptions and chosen-plaintext attacks") between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:*

1. *$\mathcal{A}(1^\lambda)$ picks $n \in \mathbb{N}$, and an index $i^* \in \{1, \ldots, n\}$.*
2. *$\mathcal{C}$ samples $b \in \{0, 1\}$, and $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_n \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$, and sets $(epk_i, esk_i) = (g_1^{\boldsymbol{\omega}_i^\top \mathbf{B}}, \boldsymbol{\omega}_i)$, and $\mathbf{pk} = (epk_1, \ldots, epk_n)$ and $\mathbf{sk} = (esk_1, \ldots, esk_n)$.*
3. *Next, $\mathcal{A}$ is run on input $(epk_i)_{i=1}^{\ell_{\mathbf{B}}}$, $(esk_i)_{i \neq i^*}$, and with (many-time) access to the following oracle:*

– $\mathcal{O}_{\mathbf{enc}}(\mathbf{u}^{(0)}, \mathbf{u}^{(1)})$, *for* $\mathbf{u}^{(j)} = (u_1^{(j)}, \ldots, u_n^{(j)}) \in \mathbb{Z}_{|\mathbb{G}_2|}^n$ *(*$j \in \{0,1\}$*), first checks that* $u_i^{(0)} = u_i^{(1)}$ *for all* $i \neq i^*$*, and returns* $\perp$ *if not. Then,* $\mathcal{O}_{\mathbf{enc}}$ *computes and returns* $\mathbf{c} = \mathbf{E}_{\mathbf{pk}}(\mathbf{u}^{(b)})$.

4. If $\mathcal{A}$ *terminates with output* b'*, then* $\mathcal{C}$ *outputs* 1 *iff* $b = b'$.

Let

$$\mathrm{Adv}^{\mathrm{mccpa}}_{\mathbb{G},\mathcal{A}}(\lambda) = \Pr[\mathcal{C} \text{ outputs } 1] - 1/2.$$

Lemma 1 (Tight security of $(\mathbf{E}, \mathbf{D})$). *For every* $\mathcal{A}$*, there exists an adversary* $\mathcal{B}$ *(of essentially the same complexity as the IND-MCCPA game with* $\mathcal{A}$*) for which*

$$\mathrm{Adv}^{\mathrm{gddh}}_{\mathbb{G},\mathcal{B}}(\lambda) \;=\; \mathrm{Adv}^{\mathrm{mccpa}}_{\mathbb{G},\mathcal{A}}(\lambda). \tag{1}$$

Proof. $\mathcal{B}$ gets $epk^* = g_1^{\boldsymbol{\omega}^{*\top}\mathbf{B}}$ and $\mathbf{c}^* = (\mathbf{c}_0^*, c_1^*) = (g_1^{\mathbf{B}\mathbf{r}^*}, g_1^{\boldsymbol{\omega}^{*\top}\mathbf{B}\mathbf{r}^*} g_2^b)$ (for unknown $b \in \{0,1\}$) as input. Now $\mathcal{B}$ first runs $\mathcal{A}$ to obtain n and i^*. Then, $\mathcal{B}$ generates public and secret keys as follows:

– For $i \neq i^*$, $\mathcal{B}$ samples $\boldsymbol{\omega}_i \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$, and sets $(epk_i, esk_i) = (g^{\boldsymbol{\omega}_i^\top \mathbf{B}}, \boldsymbol{\omega}_i)$.

– $\mathcal{B}$ sets $epk_{i^*} = g_1^{\boldsymbol{\omega}^{*\top}\mathbf{B}}$, and thus implicitly defines $esk_{i^*} = \boldsymbol{\omega}_{i^*} = \boldsymbol{\omega}^*$.

Then, $\mathcal{B}$ runs $\mathcal{A}$, on input $\mathbf{pk} = (epk_i)_i$ and $(esk_i)_{i \neq i^*}$, and implements oracle $\mathcal{O}_{\mathbf{enc}}$ as follows:

– Upon an $\mathcal{O}_{\mathbf{enc}}(\mathbf{u}^{(0)}, \mathbf{u}^{(1)})$ query with $u_i^{(0)} = u_i^{(1)}$ for $i \neq i^*$, $\mathcal{B}$ first samples a fresh $\mathbf{r}' \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$, implicitly defines $\mathbf{r} = (u_{i^*}^{(1)} - u_{i^*}^{(0)})\mathbf{r}^* + \mathbf{r}'$, and sets up

$$\begin{aligned}
\mathbf{c}_0 &= g_1^{(u_{i^*}^{(1)} - u_{i^*}^{(0)})\mathbf{B}\mathbf{r}^* + \mathbf{B}\mathbf{r}'} = g_1^{\mathbf{B}((u_{i^*}^{(1)} - u_{i^*}^{(0)})\mathbf{r}^* + \mathbf{r}')} = g_1^{\mathbf{B}\mathbf{r}} \\
c_i &= g_1^{(u_{i^*}^{(1)} - u_{i^*}^{(0)})\boldsymbol{\omega}_i^\top \mathbf{B}\mathbf{r}^*} g_1^{\boldsymbol{\omega}_i^\top \mathbf{B}\mathbf{r}'} g_2^{u_i^{(0)}} = g_1^{\boldsymbol{\omega}_i^\top \mathbf{B}\mathbf{r}} g_2^{u_i^{(0)}} \qquad \text{for } i \neq i^* \\
c_i &= g_1^{(u_{i^*}^{(1)} - u_{i^*}^{(0)})\boldsymbol{\omega}^{*\top}\mathbf{B}\mathbf{r}^* + \boldsymbol{\omega}^{*\top}\mathbf{B}\mathbf{r}'} g_2^{(u_{i^*}^{(1)} - u_{i^*}^{(0)})\cdot b + u_{i^*}^{(0)}} = g_1^{\boldsymbol{\omega}_{i^*}^\top \mathbf{B}\mathbf{r}} g_2^{u_{i^*}^{(b)}}
\end{aligned}$$

For the resulting $\mathbf{c} = (\mathbf{c}_0, c_1, \ldots, c_n)$, we have that $\mathbf{c} = \mathbf{E}_{\mathbf{pk}}(\mathbf{u}^{(b)}; \mathbf{r})$ for (independently and uniformly distributed) random coins $\mathbf{r} = (u_{i^*}^{(1)} - u_{i^*}^{(0)})\mathbf{r}^* + \mathbf{r}'$. Hence, $\mathcal{O}_{\mathbf{enc}}$ returns $\mathbf{c}$.

Finally, $\mathcal{B}$ relays any guess b' from $\mathcal{A}$ as its own output.

Observe that $\mathcal{B}$ perfectly simulates the game from Lemma 1 (with the same challenge bit b). We obtain (1).

3.2 The prime-order setting

The groups. We consider two concrete instantiations of our generic setting. The first is a prime-order setting, in which $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$ has prime order $|\mathbb{G}| = |\mathbb{G}_1| = |\mathbb{G}_2|$. In these cases, we assume that $|\mathbb{G}| > 2^\lambda$ is public, and hence most syntactic requirements from Section 3.1 are trivially met. However, we will additionally need to assume that membership in $\mathbb{G}$ is efficiently decidable. We have numerous candidates for such groups (including, e.g., subgroups of $\mathbb{Z}_p^*$, or

elliptic curves). In such groups, plausible assumptions include the Decisional Diffie-Hellman (DDH) assumption, the k-Linear (k-LIN) assumption [30, 19], or a whole class of assumptions called Matrix-DDH assumptions [10].

Hardness of the GDDH and factoring problems. All of the mentioned assumptions imply our GDDH assumption for suitable $\ell_{\mathbf{B}}$ and $\mathbf{B}$. For instance, GDDH with $\ell_{\mathbf{B}} = 1$ and uniform $\mathbf{B}$ is nothing but a reformulation of the DDH assumption. More generally, GDDH with uniform $\mathbf{B}$ is actually the so-called $\mathcal{U}_{\ell_{\mathbf{B}}}$-MDDH assumption. In particular, this means that the k-LIN assumption implies GDDH with $\ell_{\mathbf{B}} = k$ and uniform $\mathbf{B}$ (see [10]). Additionally, we note that the $\mathbb{G}_2$-factoring assumption we make is trivially satisfied in prime-order settings (since $\mathrm{Adv}^{\mathrm{fact}}_{\mathbb{G}_2,\mathcal{A}}(\lambda) = 0$ for all $\mathcal{A}$ if $|\mathbb{G}_2| = |\mathbb{G}|$ is prime).

Pairing-friendly groups. In Section 5.4, we also exhibit a building block in the prime-order setting that uses a symmetric pairing $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ (for a suitable target group $\mathbb{G}_T$). Also for such pairing-friendly groups, we have a variety of candidates in case $\ell_{\mathbf{B}} \geq 2$. (Unfortunately, for $\ell_{\mathbf{B}} = 1$, a symmetric pairing can be used to trivially break the GDDH assumption.)

3.3 The DCR setting

The public parameters. The second setting we consider is compatible with the Decisional Composite Residuosity (DCR) assumption [29]. In this case, the global public parameters include an integer $N = PQ$, for distinct safe primes P, Q (i.e., such that $P = 2P'+1$ and $Q = 2Q'+1$ for prime $P', Q' > 2^{\lambda}$).[3] We also assume that P, Q, P', Q' are pairwise different, and that $\gcd(P + Q - 1, N) = 1$ (the latter of which ensures that N is invertible modulo $\varphi(N) = (P-1)(Q-1) = 4P'Q'$).

We implicitly set $\ell_{\mathbf{B}} = 1$, and the matrix $\mathbf{B} \in \mathbb{Z}_{|\mathbb{G}_1|\times|\mathbb{G}_1|}$ from Section 3.1 to be trivial (i.e., the identity matrix). Hence, neither $\ell_{\mathbf{B}}$ nor $g_1^{\mathbf{B}}$ will have to be included in the parameters. However, we also include a generator g_1 of $\mathbb{G}_1$ in the public parameters, chosen as described below.

The groups. We now define the groups $\mathbb{G}$, $\mathbb{G}_1$, and $\mathbb{G}_2$. Since $\mathbb{G}$ should only have large prime factors, we should avoid setting $\mathbb{G} = \mathbb{Z}^*_{N^2}$. Instead, we could set $\mathbb{G}_1$ and $\mathbb{G}_2$ to be the subgroups of order $\varphi(N)/4$ and N, respectively, and then $\mathbb{G} = \mathbb{G}_1 \cdot \mathbb{G}_2$. However, in this case, membership in $\mathbb{G}$ would not be efficiently decidable in an obvious way. So here, we define our groups in a slightly more complex way, following the approach of signed quadratic residues [13, 11, 20].

Equipped with the notation $|x|_N$ and $\left(\frac{x}{N}\right)$ from Section 2, we set

$$\mathbb{G}_1 = \left\{ |x^N|_{N^2} \;\middle|\; x \in \mathbb{Z}^*_{N^2}, \left(\frac{x^N}{N}\right) = 1 \right\} \subseteq \mathbb{Z}^*_{N^2}$$

$$\mathbb{G}_2 = \left\{ |(1+N)^e|_{N^2} \;\middle|\; e \in \mathbb{Z}_N \right\} \subseteq \mathbb{Z}^*_{N^2}$$

[3] We note that our DCR-based OR-proofs from Section 5.4 require P, Q to be somewhat larger, although still compatible with practical parameter choices.

$$\mathbb{G} \;=\; \{\; |y|_{N^2} \;\mid y \in \mathbb{Z}^*_{N^2},\; \left(\frac{y}{N}\right) = 1 \;\}.$$

These sets are groups, when equipped with the group operation $a \cdot b = |a \cdot b|_{N^2}$. Indeed, since $P, Q = 3 \bmod 4$, we have $\left(\frac{-1}{N}\right) = 1$, and thus $\left(\frac{|y_1 y_2|_{N^2}}{N}\right) = \left(\frac{y_1 y_2}{N}\right) = 1$ for $\left(\frac{y_1}{N}\right) = \left(\frac{y_2}{N}\right) = 1$. Hence, $\mathbb{G}_1$ and $\mathbb{G}$ are closed under group operation. It is then straightforward to check that $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}$ are groups.

A canonical generator g_2 of $\mathbb{G}_2$ is $|1+N|_{N^2}$, and a generator g_1 of $\mathbb{G}_1$ (to be included in the public parameters) can be randomly chosen as $|x^N|_{N^2}$ for a uniform $x \in \mathbb{Z}_{N^2}$.

Properties of the groups. We claim that $|\mathbb{G}_1| = \varphi(N)/4$. Indeed, we have that

$$\left|\{\; |x^N|_N \;\mid x \in \mathbb{Z}^*_{N^2} \;\}\right| \;=\; \left|\{\; |x^N|_{N^2} \;\mid x \in \mathbb{Z}^*_{N^2} \;\}\right| \;=\; \varphi(N)/2.$$

In other words, $|x^N|_N$ uniquely determines $|x^N|_{N^2}$. Furthermore, since N is invertible modulo $\varphi(N)$, the map $f : \mathbb{Z}^*_{N^2} \to \mathbb{Z}^*_N$ with $f(x) = x^N \bmod N$ is surjective. Hence, the set of all $|x^N|_N$ with $\left(\frac{x^N}{N}\right) = 1$ has cardinality $\varphi(N)/4$ (cf. [20, Lemma 1]). Using that $|x^N|_N$ fixes $|x^N|_{N^2}$, we obtain $|\mathbb{G}_1| = \varphi(N)/4$. Moreover, for $e \in \mathbb{Z}_N$, we can write $|(1+N)^e|_{N^2} = |1+eN|_{N^2} = e/|e| + |e|_N N$, and thus $|\mathbb{G}_2| = N$. Finally, we have $\mathbb{G} = \mathbb{G}_1 \cdot \mathbb{G}_2$, since every $|y|_{N^2} \in \mathbb{G}$ can be written as $|y|_{N^2} = |x^N(1+N)^e|_{N^2}$ with $\left(\frac{x^N}{N}\right) = 1$. Hence, since $|\mathbb{G}_1| = \varphi(N)/4 = P'Q'$ and $|\mathbb{G}_2| = N = PQ$ are coprime, $|\mathbb{G}| = |\mathbb{G}_1| \cdot |\mathbb{G}_2| = N \cdot \varphi(N)/4$ is square-free.

We also note that the discrete logarithm problem is easy in $\mathbb{G}_2$. Indeed, for $g_2^u \in \mathbb{G}_2$, we have

$$g_2^u \;=\; |(1+N)^e|_{N^2} \;=\; |1+eN|_{N^2} \;=\; \begin{cases} [e]_N N + 1 & \text{if } [e]_N < N/2 \\ [-e]_N N - 1 & \text{if } [e]_N > N/2. \end{cases}$$

A simple case distinction thus allows to compute $[e]_N$.

Membership testing and sampling exponents. It is left to note that membership in $\mathbb{G}$ can be efficiently decided (by checking that $y \in \mathbb{Z}_{N^2}$ is invertible, lies between $-N^2/2$ and $N^2/2$, and satisfies $\left(\frac{y}{N}\right) = 1$). However, since $|\mathbb{G}_1|$ will not be public, exponents $s \in \mathbb{Z}_{|\mathbb{G}_1|}$ can only be sampled approximatively, e.g., by uniformly sampling $s \in \mathbb{Z}_{\lfloor N/4 \rfloor}$. This incurs a statistical defect of $\mathbf{O}(1/2^\lambda)$ upon each such sampling. In the following, we will silently ignore these statistical defects (and assume that there is an algorithm that uniformly samples $s \in \mathbb{Z}_{\varphi(N)}$) in our generic constructions for simplicity and ease of presentation. However, we note that the concrete bound (8) also holds for such an approximative sampling in the DCR setting.

Hardness of the GDDH and factoring problems. We claim that in the setting described above, the Decisional Composite Residuosity (DCR) assumption [29] implies the GDDH assumption. This connection has already been established in [15, Theorem 2] for a slight variant of the groups $\mathbb{G}$, $\mathbb{G}_1$, $\mathbb{G}_2$ above.

(In their setting, $\mathbb{G}_1$ consists of elements $x^N \in \mathbb{Z}_{N^2}$ with $\left(\frac{x^N}{N}\right) = 1$, instead of elements $|x^N|_{N^2}$ with $\left(\frac{x^N}{N}\right) = 1$.) In fact, their proof applies also to our setting, and we obtain that the DCR assumption implies the GDDH assumption with $\ell = 1$ and trivial $\mathbf{B} = 1$ in $\mathbb{G}$ (as in Definition 3).

Furthermore, we note that the DCR assumption also implies the $\mathbb{G}_2$-factoring assumption (Definition 4). We sketch how any $\mathbb{G}_2$-factoring adversary $\mathcal{A}$ can be transformed into a DCR adversary $\mathcal{B}$. First, $\mathcal{B}$ runs $\mathcal{A}$, and obtains elements $g_2^{u_1}, \ldots, g_2^{u_q}$. Then, $\mathcal{B}$ uses that the discrete logarithm problem is easy in $\mathbb{G}_2$, and retrieves the corresponding $u_1, \ldots, u_q \in \mathbb{Z}_{|\mathbb{G}_2|}$. Now if $\gcd(|\mathbb{G}_2|, u_i) \notin \{1, |\mathbb{G}_2|\}$ for some u_i, then $\gcd(N, u_i) \in \{P, Q\}$ directly allows to factor N. Hence, if $\mathcal{A}$ succeeds, then $\mathcal{B}$ can factor N, and solve its own DCR challenge (e.g., by computing the order of its input).

4 Tightly secure building blocks

In this section, we describe two building blocks for our main KEM construction. The first, tightly secure one-time signature schemes, is fairly standard, but requires a new instantiation in the DCR setting to achieve tight security. The second is, key extractors, is new, but similar building blocks have been been used at least in the prime-order setting implicitly in previous works on tight security (e.g., [12]).

4.1 One-time signature schemes

Definition 6 (Digital signature scheme). *A digital signature scheme* $\mathbf{OTS} = (\mathbf{SGen}, \mathbf{SSig}, \mathbf{SVer})$ *consists of the following PPT algorithms:*

- $\mathbf{SGen}(1^\lambda)$ *outputs a keypair* (ovk, osk)*. We call ovk and osk the verification, resp. signing key.*
- $\mathbf{SSig}(osk, M)$*, for a message* $M \in \{0,1\}^*$*, outputs a signature* σ*.*
- $\mathbf{SVer}(ovk, M, \sigma)$*, outputs either* 0 *or* 1*.*

We require correctness in the sense that for all (ovk, osk) *that lie in the range of* $\mathbf{SGen}(1^\lambda)$*, all* $M \in \{0,1\}^*$*, and all* σ *in the range of* $\mathbf{SSig}(osk, M)$*, we always have* $\mathbf{SVer}(ovk, M, \sigma) = 1$*.*

We only require one-time security (and call a signature scheme secure in this sense also a one-time signature scheme):

Definition 7 (EUF-MOTCMA security). *Let* **OTS** *be a digital signature scheme as in Definition 6, and consider the following game between a challenger* $\mathcal{C}$ *and an adversary* $\mathcal{A}$*:*

1. $\mathcal{C}$ *runs* $\mathcal{A}$ *on input* 1^λ*, and with (many-time) oracle access to the following oracles:*
   - $\mathcal{O}_{\mathbf{gen}}()$ *samples a fresh keypair* $(ovk, osk) \leftarrow \mathbf{SGen}()$*, and returns ovk.*

– $\mathcal{O}_{\mathbf{sig}}(ovk, M)$ *first checks if* ovk *has been generated by* $\mathcal{O}_{\mathbf{gen}}$*, and returns* $\perp$ *if not. Next,* $\mathcal{O}_{\mathbf{sig}}$ *checks if there has been a previous* $\mathcal{O}_{\mathbf{sig}}(ovk, \cdot)$ *query (i.e., an* $\mathcal{O}_{\mathbf{sig}}$ *query with the same* ovk*), and returns* $\perp$ *if so. Let* osk *be the corresponding secret key generated alongside* ovk*. (If* ovk *has been generated multiple times by* $\mathcal{O}_{\mathbf{gen}}$*, take the first such* osk*.)* $\mathcal{O}_{\mathbf{sig}}$ *returns* $\sigma \leftarrow \mathbf{SSig}(osk, M)$*.*

2. *If* $\mathcal{A}$ *returns* (ovk^*, M^*, σ^*)*, such that* $\mathbf{SVer}(ovk^*, M^*, \sigma^*) = 1$*, and* ovk^* *has been returned by* $\mathcal{O}_{\mathbf{gen}}$*, but* σ^* *has not been returned by* $\mathcal{O}_{\mathbf{sig}}(ovk^*, M^*)$*, then* $\mathcal{C}$ *returns* 1*. Otherwise,* $\mathcal{C}$ *returns* 0*.*

Let $\mathrm{Adv}^{\mathrm{ots}}_{\mathbf{OTS},\mathcal{A}}(\lambda)$ *be the probability that* $\mathcal{C}$ *finally outputs* 1 *in the above game. We say that* **OTS** *is strongly existentially unforgeable under many one-time chosen-message attacks (EUF-MOTCMA secure) iff for every PPT* $\mathcal{A}$*, the function* $\mathrm{Adv}^{\mathrm{ots}}_{\mathbf{OTS},\mathcal{A}}(\lambda)$ *is negligible.*

We remark, however, that our security notion is "strong", in the sense that a forger is already successful when he manages to generate a new signature for an already signed message.

A construction in the prime-order setting In case $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$ with $|\mathbb{G}|$ prime and public, [18] already give a simple construction of a digital signature scheme that achieves EUF-MOTCMA security under the discrete logarithm assumption. Most importantly for our case, their security reduction is tight (i.e., only loses a constant factor). We refer to their paper for details.

A construction in the DCR setting In the DCR setting (as in Section 3.3), there exist simple and efficient EUF-MOTCMA secure signature schemes from the factoring [24] or RSA assumptions [22]. However, these schemes are not known to be tightly secure.

Hence, in the full version [16], we construct a new digital signature scheme whose EUF-MOTCMA security can be tightly reduced to the GDDH assumption in the DCR setting.

4.2 Key extractors

Intuition. Intuitively, a key extractor derives a pseudorandom key K from a given encryption $\mathbf{c} = \mathbf{E}(0; \mathbf{r})$ of 0. This K can be derived either publicly, using a public extraction key xpk and the witness $\mathbf{r}$, or secretly, using a secret extraction key xsk and only the ciphertext $\mathbf{c}$. We desire security in the sense keys derived secretly (i.e., using xsk) from random ciphertexts $\mathbf{c} = \mathbf{E}(R; \mathbf{r})$ for random R cannot be distinguished from truly random bitstrings K. This should hold even for many such challenges, and in the face of oracle access to xsk on "consistent" ciphertexts $\mathbf{c} = \mathbf{E}(0; \mathbf{r})$.

In this sense, key extractors give a computational form of the soundness guarantee provided by universal hash proof systems. We also note that a similar tool has been implicitly used in [12] for a similar purpose in the prime-order

setting. Hence, we abstract and generalize their construction in a straightforward way.

Definition. In the following, fix a function $\ell_{\text{ext}} = \ell_{\text{ext}}(\lambda)$. In the following definition, we will choose the value R encrypted in random ciphertexts uniformly from $\mathbb{Z}_{2^{\ell_{\text{ext}}}}$. Our generic construction of key extractors works for any $\ell_{\text{ext}} \geq 3\lambda$ (and $|\mathbb{G}_2| \geq 2^{3\lambda}$).

Definition 8 (Key extractor). *A key extractor* $\mathbf{EXT} = (\mathbf{ExtGen}, \mathbf{Ext}_{\text{pub}}, \mathbf{Ext}_{\text{priv}})$ *for* $\mathbb{G}$ *consists of the following PPT algorithms*

- $\mathbf{ExtGen}(1^\lambda, epk)$*, on input a public encryption key* $epk = g_1^{\boldsymbol{\omega}^\top \mathbf{B}} \in \mathbb{G}_1^{\ell_{\mathbf{B}}}$ *for* $(\mathbf{E}, \mathbf{D})$ *(as in Section 3.1), outputs a keypair* (xpk, xsk)*. We call* xpk *the public and* xsk *the private extraction key.*
- $\mathbf{Ext}_{\text{pub}}(xpk, \mathbf{c}, \mathbf{r})$*, for* $\mathbf{c} = \mathbf{E}_{epk}(0; \mathbf{r})$*, outputs a key* $K \in \{0,1\}^\lambda$*.*
- $\mathbf{Ext}_{\text{priv}}(xsk, \mathbf{c})$ *also outputs a session key* $K \in \{0,1\}^\lambda$*.*

We require the following:

Correctness. *For all* $epk = g_1^{\boldsymbol{\omega}^\top \mathbf{B}}$*, all keypairs* (xpk, xsk) *that lie in the range of* $\mathbf{ExtGen}(1^\lambda, epk)$*, all* $\mathbf{r} \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$*, and all* $\mathbf{c} = \mathbf{E}_{epk}(0; \mathbf{r})$*, we always have* $\mathbf{Ext}_{\text{pub}}(xpk, \mathbf{c}, \mathbf{r}) = \mathbf{Ext}_{\text{priv}}(xsk, \mathbf{c})$*.*

Indistinguishability. *Consider the following game between a challenger* $\mathcal{C}$ *and an adversary* $\mathcal{A}$*:*

1. $\mathcal{C}$ *uniformly samples* $\boldsymbol{\omega} \in \mathbb{Z}_{|\mathbb{G}_1|}^{\ell_{\mathbf{B}}}$ *and sets* $(epk, esk) = (g_1^{\boldsymbol{\omega}^\top \mathbf{B}}, \boldsymbol{\omega})$*. Then,* $\mathcal{C}$ *generates an* $\mathbf{EXT}$ *keypair* $(xpk, xsk) \leftarrow \mathbf{ExtGen}(1^\lambda, epk)$*, and finally samples* $b \in \{0,1\}$*.*
2. $\mathcal{A}$ *is run on input* $(1^\lambda, epk, xpk)$*, with (many-time) access to oracles* $\mathcal{O}_{\mathbf{cha}}$ *and* $\mathcal{O}_{\mathbf{ext}}$ *that operate as follows:*
   - $\mathcal{O}_{\mathbf{cha}}()$ *uniformly chooses a fresh* $R \in \mathbb{Z}_{2^{\ell_{\text{ext}}}}$*, computes* $\mathbf{c} \leftarrow \mathbf{E}_{epk}(R)$ *and* $K_0 = \mathbf{Ext}_{\text{priv}}(xsk, \mathbf{c})$*, and uniformly chooses* $K_1 \in \{0,1\}^\lambda$*. Finally,* $\mathcal{O}_{\mathbf{cha}}$ *returns* $(\mathbf{c}, K_b)$*.*
   - $\mathcal{O}_{\mathbf{ext}}(\mathbf{c})$ *first checks if* $\mathbf{D}_{esk}(\mathbf{c}) = g_2^0$*. If not, then we say that* $\mathcal{A}$ *fails, and* $\mathcal{C}$ *terminates with output* 0 *immediately. Otherwise,* $\mathcal{O}_{\mathbf{ext}}$ *computes and returns* $K = \mathbf{Ext}_{\text{priv}}(xsk, \mathbf{c})$*.*
   - *Finally,* $\mathcal{A}$ *outputs a bit* b'*, and* $\mathcal{C}$ *outputs* 1 *iff* $b = b'$ *(and* 0 *otherwise).*

Let $\text{Adv}^{\text{ext}}_{\mathbf{EXT},\mathcal{A}}(\lambda) = \Pr[\mathcal{C} \text{ outputs } 1] - 1/2$*. We require that for all PPT* $\mathcal{A}$*,* $\text{Adv}^{\text{snd}}_{\mathbf{PS},\mathcal{A}}(\lambda) \leq \varepsilon$ *for a negligible function* $\varepsilon = \varepsilon(\lambda)$*.*

A generic construction For our GDDH-based key extractor, we assume that a function h chosen from a family of universal hash functions $\mathbf{UHF}_\lambda$ is made public in the global public parameters pp. Then, our extractor $\mathbf{EXT}^{\text{gddh}} = (\mathbf{ExtGen}^{\text{gddh}}, \mathbf{Ext}^{\text{gddh}}_{\text{pub}}, \mathbf{Ext}^{\text{gddh}}_{\text{priv}})$ is defined as follows:

- $\mathbf{ExtGen}^{\text{gddh}}(1^\lambda, epk)$, for $epk = g_1^{\boldsymbol{\omega}^\top \mathbf{B}}$, uniformly samples $\mathbf{s} \in \mathbb{Z}_{|\mathbb{G}|}^{\ell_{\mathbf{B}}}$ and $t \in \mathbb{Z}_{|\mathbb{G}|}$, and computes $g_1^{\mathbf{w}^\top} := g_1^{\mathbf{s}^\top \mathbf{B} + t \cdot \boldsymbol{\omega}^\top \mathbf{B}} \in \mathbb{G}_1^{\ell_{\mathbf{B}}}$. The output of $\mathbf{ExtGen}^{\text{gddh}}$ is $xpk = g_1^{\mathbf{w}^\top}$ and $xsk = (\mathbf{s}, t)$.
- $\mathbf{Ext}^{\text{gddh}}_{\text{pub}}(xpk, \mathbf{c}, \mathbf{r})$, for xpk as above and $\mathbf{c} = \mathbf{E}_{epk}(0; \mathbf{r})$, outputs $K = h(g_1^{\mathbf{w}^\top \cdot \mathbf{r}})$.

– $\mathbf{Ext}^{\mathrm{gddh}}_{\mathrm{priv}}(xsk, \mathbf{c})$, for $\mathbf{c} = (g^{\boldsymbol{\gamma}}, g^{\delta}) \in \mathbb{G}^{\ell_{\mathbf{B}}} \times \mathbb{G}$, outputs $K = h(g^{\mathbf{s}^\top \boldsymbol{\gamma} + t \cdot \delta})$.

Given $g_1^{\mathbf{w}^\top} = g_1^{\mathbf{s}^\top \mathbf{B} + t \cdot \boldsymbol{\omega}^\top \mathbf{B}}$ and a ciphertext $\mathbf{c} = \mathbf{E}(0; \mathbf{r}) = (g^{\boldsymbol{\gamma}}, g^{\delta}) = (g_1^{\mathbf{Br}}, g_1^{\boldsymbol{\omega}^\top \mathbf{Br}})$, we have

$$g_1^{\mathbf{w}^\top \mathbf{r}} = g_1^{\mathbf{s}^\top \mathbf{Br} + t \cdot \boldsymbol{\omega}^\top \mathbf{Br}} = g^{\mathbf{s}^\top \boldsymbol{\gamma} + t \cdot \delta},$$

and correctness follows. Indistinguishability follows from the following lemma:

Lemma 2. *For $\ell_{\mathrm{ext}} \geq 3\lambda$ and $|\mathbb{G}_2| \geq 2^{3\lambda}$, $\mathbf{EXT}^{\mathrm{gddh}}$ above satisfies the indistinguishability property of Definition 8, assuming GDDH in $\mathbb{G}$. Specifically, for every adversary $\mathcal{A}$ that makes at most q oracle queries, there is an adversary $\mathcal{B}$ (with roughly the same complexity as the indistinguishability experiment with $\mathbf{EXT}^{\mathrm{gddh}}$ and $\mathcal{A}$), such that*

$$\mathrm{Adv}^{\mathrm{ext}}_{\mathbf{EXT}^{\mathrm{gddh}},\mathcal{A}}(\lambda) \;\leq\; \mathrm{Adv}^{\mathrm{gddh}}_{\mathbb{G},\mathcal{B}}(\lambda) + q/2^{\lambda}. \tag{2}$$

Due to lack of space, we outsource a proof of Lemma 2 to the full version [16].

Summing up, we obtain

Theorem 1. *Under the GDDH assumption, and for $\ell_{\mathrm{ext}} \geq 3\lambda$ and $|\mathbb{G}_2| \geq 2^{3\lambda}$, $\mathbf{EXT}^{\mathrm{gddh}}$ is a key extractor in the sense of Definition 8.*

5 Benign proof systems

Intuition. Benign proof systems are the central technical tool in our KEM construction. Intuitively, a benign proof system for some language $\mathcal{L}$ is a non-interactive designated-verifier zero-knowledge proof system with strong soundness guarantees. Concretely, the system guarantees soundness even if simulated proofs for potentially false statements $x \notin \mathcal{L}$ are known. However, we do not quite require "simulation-soundness", in the sense that this should hold for simulated proofs for arbitrary false statements. (We note that simulation-sound proof systems are extremely useful in the context of tight security proofs, but they are also very hard to construct.)

Instead, we only require that no adversary can forge proofs for statements $x \notin \mathcal{L}$ that are "more false" than any statement for which a simulated proof is known. A little more specifically, we require that even if simulated proofs for statements $x \in \mathcal{L}' \supseteq \mathcal{L}$ are known, an adversary cannot forge a proof for some $x \notin \mathcal{L}'$. The main benefit over existing soundness notions is that $\mathcal{L}'$ does not even have to be known during the construction of the scheme. (For instance, our first proof system provides a "graceful soundness degradation", in the sense that it is sound in this sense for arbitrary linear languages $\mathcal{L}' \supseteq \mathcal{L}$.)

Overview over our constructions. Apart from the abstraction, we also provide generic and setting-specific constructions of benign proof systems. Our generic constructions (for a linear, and a "dynamically parameterized" linear language) can be viewed as abstractions and generalizations of universal hash proof systems. For $\mathcal{L}' = \mathcal{L}$, soundness in the above sense follows immediately from the correctness property of hash proof systems. (Indeed, hash proofs for valid

instances $x \in \mathcal{L}$ are unique and completely determined by public information.) For $\mathcal{L}' \supsetneq \mathcal{L}$, we will use additional properties of specific (existing) hash proof systems. In fact, the mentioned "graceful degradation" guarantees have already been used implicitly in the work of [12].

However, we also consider a somewhat nonstandard (and in our application crucial) "OR-language". Here, we give a prime-order instantiation in pairing-friendly groups (which is directly implied by the universal hash proof systems for disjunctions from [1]), and a new instance in the DCR setting. This DCR instance will be the key to the DCR-based instantiation of our KEM.

5.1 Definition

Definition 9 (Proof system). *Let $\mathcal{L} = \{\mathcal{L}_{pars}\}$ be a family of languages[4] with $\mathcal{L}_{pars} \subseteq \mathcal{X}_{pars}$, and with efficiently computable witness relation $\mathcal{R}$. A non-interactive designated-verifier proof system (NIDVPS)* $\mathbf{PS} = (\mathbf{PGen}, \mathbf{PPrv}, \mathbf{PVer}, \mathbf{PSim})$ *for $\mathcal{L}$ consists of the following PPT algorithms:*

- $\mathbf{PGen}(1^\lambda, pars)$ *outputs a keypair (ppk, psk). We call ppk the public and psk the private key.*
- $\mathbf{PPrv}(ppk, x, w)$*, for $x \in \mathcal{L}$ and $\mathcal{R}(x, w) = 1$, outputs a proof π.*
- $\mathbf{PVer}(psk, x, \pi)$*, for $x \in \mathcal{X}$ and a proof π, outputs a verdict $b \in \{0, 1\}$.*
- $\mathbf{PSim}(psk, x)$*, for $x \in \mathcal{L}$, outputs a proof π.*

We require correctness in the following sense:

Completeness. *For all pars, all (ppk, psk) in the range of $\mathbf{PGen}(1^\lambda, pars)$, all $x \in \mathcal{L}$, and all w with $\mathcal{R}(pars, x, w) = 1$, we always have $\mathbf{PVer}(psk, x, \mathbf{PPrv}(ppk, x, w)) = 1$.*

All relevant security properties of a NIDVPS are condensed in the following definition.

Definition 10 (Benign proof system). *Let* **PS** *be an NIDVPS for $\mathcal{L}$ as in Definition 9, and let $\mathcal{L}^{\mathrm{sim}} = \{\mathcal{L}^{\mathrm{sim}}_{pars}\}$, $\mathcal{L}^{\mathrm{ver}} = \{\mathcal{L}^{\mathrm{ver}}_{pars}\}$, and $\mathcal{L}^{\mathrm{snd}} = \{\mathcal{L}^{\mathrm{snd}}_{pars}\}$ be families of languages. We say that* **PS** *is* $(\mathcal{L}^{\mathrm{sim}}, \mathcal{L}^{\mathrm{ver}}, \mathcal{L}^{\mathrm{snd}})$-benign *if the following properties hold:*

(Perfect) zero-knowledge. *For all pars, all (ppk, psk) that lie in the range of $\mathbf{PGen}(1^\lambda, pars)$, and all $x \in \mathcal{L}$ and w with $\mathcal{R}(pars, x, w) = 1$, we have the following equivalence of distributions:*

$$\mathbf{PPrv}(ppk, x, w) \quad \equiv \quad \mathbf{PSim}(psk, x).$$

(Statistical) $(\mathcal{L}^{\mathrm{sim}}, \mathcal{L}^{\mathrm{ver}}, \mathcal{L}^{\mathrm{snd}})$-soundness. *Consider the following game played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:*

1. *$\mathcal{A}$ is run on input 1^λ, and chooses pars.*
2. *$\mathcal{C}$ generates $(ppk, psk) \leftarrow \mathbf{PGen}(1^\lambda, pars)$.*
3. *$\mathcal{A}$ is run again on input $(1^\lambda, ppk)$, and with (many-time) access to oracles $\mathcal{O}_{\mathbf{sim}}$ and $\mathcal{O}_{\mathbf{ver}}$ that operate as follows:*

[4] These languages may also implicitly depend on the global public parameters pp.

– $\mathcal{O}_{\mathbf{sim}}(x)$ *checks if* $x \in \mathcal{L}^{\mathrm{sim}}_{pars}$*, and if yes, returns* $\mathbf{PSim}(psk, x)$*. Otherwise,* $\mathcal{O}_{\mathbf{sim}}$ *returns* $\perp$*.*
– $\mathcal{O}_{\mathbf{ver}}(x, \pi)$ *checks if* $x \in \mathcal{L}^{\mathrm{ver}}_{pars}$*, and, if so, returns* $\mathbf{PVer}(psk, x, \pi)$*. Otherwise,* $\mathcal{O}_{\mathbf{ver}}$ *returns* $\perp$*.*

Finally, $\mathcal{A}$ *wins iff it has queried* $\mathcal{O}_{\mathbf{ver}}$ *with* (x, π) *such that* $x \in \mathcal{X}_{pars} \setminus \mathcal{L}^{\mathrm{snd}}_{pars}$ *and* $\mathbf{PVer}(psk, x, \pi) = 1$*. Let* $\mathrm{Adv}^{\mathrm{snd}}_{\mathbf{PS},\mathcal{A}}(\lambda)$ *the probability that* $\mathcal{A}$ *wins. We require that for all (not necessarily computationally bounded)* $\mathcal{A}$ *that only make a polynomial number of oracle queries,* $\mathrm{Adv}^{\mathrm{snd}}_{\mathbf{PS},\mathcal{A}}(\lambda)$ *is negligible.*

Intuitively, the soundness condition of Definition 10 thus states that no proofs for $\mathcal{X} \setminus \mathcal{L}^{\mathrm{snd}}_{pars}$-statements can be forged, even when (simulated) proofs for $\mathcal{L}^{\mathrm{sim}}_{pars}$-statements are available, and proofs for $\mathcal{L}^{\mathrm{ver}}_{pars}$-statements can be verified.

5.2 The generic linear language

We will be interested in proof systems for "linear languages", in the sense that instances are vectors of group elements, and the language is closed under vector addition (i.e., componentwise group operation).

In the following, let $D \in \mathbb{N}$ and $\mathbf{pk} = (epk_1, \ldots, epk_D) = (g_1^{\boldsymbol{\omega}_1^\top \mathbf{B}}, \ldots, g_1^{\boldsymbol{\omega}_D^\top \mathbf{B}}) \in (\mathbb{G}_1^{\ell_{\mathbf{B}}})^D$. For a concise notation, write $\boldsymbol{\Omega} = (\boldsymbol{\omega}_1 || \ldots || \boldsymbol{\omega}_D) \in \mathbb{Z}^{\ell_{\mathbf{B}} \times D}_{|\mathbb{G}_1|}$. Also, fix a $\mathbb{Z}_{|\mathbb{G}_2|}$-module

$$\mathfrak{U} \;=\; \{\mathbf{M}\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^d_{|\mathbb{G}_2|}\} \;\subseteq\; \mathbb{Z}^D_{|\mathbb{G}_2|} \tag{3}$$

defined by a matrix $\mathbf{M} \in \mathbb{Z}^{D \times d}_{|\mathbb{G}_2|}$. Our languages are parameterized over $pars_{\mathrm{lin}} = (\mathbf{pk}, \mathbf{M})$, although $\mathcal{L}^{\mathrm{lin}}_{\mathbf{pk}}$ only depends on $\mathbf{pk}$, and not on $\mathbf{M}$. Namely, consider

$$\begin{aligned} \mathcal{L}^{\mathrm{lin}}_{\mathbf{pk}} \;&=\; \left\{\mathbf{E}_{\mathbf{pk}}(\mathbf{u}; \mathbf{r}) \;\middle|\; \mathbf{r} \in \mathbb{Z}^{\ell_{\mathbf{B}}}_{|\mathbb{G}_1|},\ \mathbf{u} = \mathbf{0} \in \mathbb{Z}^D_{|\mathbb{G}_2|}\right\} \\ \mathcal{L}^{\mathrm{lin}}_{\mathrm{sim},(\mathbf{pk},\mathbf{M})} \;&=\; \mathcal{L}^{\mathrm{lin}}_{\mathrm{ver},(\mathbf{pk},\mathbf{M})} \;=\; \mathcal{L}^{\mathrm{lin}}_{\mathrm{snd},(\mathbf{pk},\mathbf{M})} \\ &=\; \left\{\mathbf{E}_{\mathbf{pk}}(\mathbf{u}; \mathbf{r}) \;\middle|\; \mathbf{r} \in \mathbb{Z}^{\ell_{\mathbf{B}}}_{|\mathbb{G}_1|},\ \mathbf{u} \in \mathfrak{U}\right\} \\ \mathcal{X}^{\mathrm{lin}} \;&=\; \mathbb{G}^{\ell_{\mathbf{B}}+D}, \end{aligned} \tag{4}$$

and set $\mathcal{L}^{\mathrm{lin}} = \{\mathcal{L}^{\mathrm{lin}}_{\mathbf{pk}}\}$ and $\mathcal{L}^{\mathrm{lin}}_{\mathrm{sim}} = \mathcal{L}^{\mathrm{lin}}_{\mathrm{ver}} = \mathcal{L}^{\mathrm{lin}}_{\mathrm{snd}} = \{\mathcal{L}^{\mathrm{lin}}_{\mathrm{sim},(\mathbf{pk},\mathbf{M})}\}$. A witness for $x \in \mathcal{L}^{\mathrm{lin}}_{\mathbf{pk}}$ is $\mathbf{r}$.

In the full version [16], we present a simple GDDH-based construction (based upon hash proof systems) of an $(\mathcal{L}^{\mathrm{lin}}_{\mathrm{sim}}, \mathcal{L}^{\mathrm{lin}}_{\mathrm{ver}}, \mathcal{L}^{\mathrm{lin}}_{\mathrm{snd}})$-benign proof system for $\mathcal{L}^{\mathrm{lin}}$.

5.3 A dynamically parameterized linear language

In our scheme, we will also use a slight variant of the generic linear language above. Specifically, we will consider a simple "dynamically parameterized" linear language, where one parameter (i.e., coefficient) is determined by the language

instance. For a formal description, let $pars_{\text{hash}} = \mathbf{pk} = (epk_1, epk_2) \in (\mathbb{G}_1^{\ell_{\mathbf{B}}})^2$, and

$$\begin{aligned}
\mathcal{L}^{\text{hash}}_{\mathbf{pk}} &= \left\{\left(\mathbf{E}_{\mathbf{pk}}(\mathbf{u};\mathbf{r}),\ \tau\right) \mid \mathbf{u} = \mathbf{0} \in \mathbb{Z}^2_{|\mathbb{G}_2|}\right\} \\
\mathcal{L}^{\text{hash}}_{\text{sim},\mathbf{pk}} &= \mathcal{L}^{\text{hash}}_{\text{ver},\mathbf{pk}} = \mathcal{L}^{\text{hash}}_{\text{snd},\mathbf{pk}} \\
&= \left\{\left(\mathbf{E}_{\mathbf{pk}}(\mathbf{u};\mathbf{r}),\ \tau\right) \mid \mathbf{u} = (u_1, u_2)^\top \in \mathbb{Z}^2_{|\mathbb{G}_2|},\ u_2 = \tau u_1\right\} \\
\mathcal{X}^{\text{hash}}_{\mathbf{pk}} &= \left\{\left(\mathbf{E}_{\mathbf{pk}}(\mathbf{u};\mathbf{r}),\ \tau\right) \mid \mathbf{u} \in \mathbb{Z}^2_{|\mathbb{G}_2|}\right\},
\end{aligned} \tag{5}$$

where $\mathbf{r}$ and τ always range over $\mathbb{Z}^{\ell_{\mathbf{B}}}_{|\mathbb{G}_1|}$ and $\mathbb{Z}_{|\mathbb{G}_2|}$, respectively. A witness for $x \in \mathcal{L}^{\text{hash}}$ is $\mathbf{r}$. The families $\mathcal{L}^{\text{hash}}$, $\mathcal{L}^{\text{hash}}_{\text{sim}}$, $\mathcal{L}^{\text{hash}}_{\text{ver}}$, and $\mathcal{X}^{\text{hash}}$ are defined in the obvious way.

In the full version [16], we present a simple GDDH-based construction (based upon hash proof systems) of an $(\mathcal{L}^{\text{hash}}_{\text{sim}}, \mathcal{L}^{\text{hash}}_{\text{ver}}, \mathcal{L}^{\text{hash}}_{\text{snd}})$-benign proof system for $\mathcal{L}^{\text{hash}}$.

5.4 The generic OR-language

We will also be interested in the following family $\mathcal{L}^{\vee}$, together with its "simulation", "verification" and "soundness" counterparts $\mathcal{L}^{\vee}_{\text{sim}}$, $\mathcal{L}^{\vee}_{\text{ver}}$ and $\mathcal{L}^{\vee}_{\text{snd}}$. Here, the actual languages in $\mathcal{L}^{\vee}$ are linear like those in $\mathcal{L}^{\text{lin}}$. However, soundness also holds when $\mathcal{L}^{\vee}_{\text{sim}}$-instances are simulated, and those instances have an "OR flavor".

The language parameters are $pars_{\vee} = (\mathbf{pk}, \ell_{\vee})$ for $\mathbf{pk} = (epk_1, epk_2) \in (\mathbb{G}_1^{\ell_{\mathbf{B}}})^2$, and a function $\ell_{\vee} = \ell_{\vee}(\lambda)$. The families $\mathcal{L}^{\vee}$, $\mathcal{L}^{\vee}_{\text{sim}}$, $\mathcal{L}^{\vee}_{\text{ver}}$, $\mathcal{L}^{\vee}_{\text{snd}}$, and $\mathcal{X}^{\vee}$ are comprised of the following languages, where we consider all $\mathbf{r} \in \mathbb{Z}^{\ell_{\mathbf{B}}}_{|\mathbb{G}_1|}$, and $\mathbf{u} = (u_1, u_2) \in (\mathbb{Z}^*_{|\mathbb{G}_2|} \cup \{0\})^2$:

$$\begin{aligned}
\mathcal{L}^{\vee}_{\mathbf{pk}} = \mathcal{L}^{\vee}_{\text{ver},\mathbf{pk}} &= \left\{\mathbf{E}_{\mathbf{pk}}(\mathbf{u};\mathbf{r}) \mid u_1 = 0\right\} \\
\mathcal{L}^{\vee}_{\text{sim},(\mathbf{pk},\ell_{\vee})} &= \left\{\mathbf{E}_{\mathbf{pk}}(\mathbf{u};\mathbf{r}) \mid u_1 = 0 \ \vee\ (|u_1| < 2^{\ell_{\vee}} \wedge u_2 = 0)\right\} \\
\mathcal{L}^{\vee}_{\text{snd},\mathbf{pk}} &= \left\{\mathbf{E}_{\mathbf{pk}}(\mathbf{u};\mathbf{r}) \mid u_1 = 0 \vee u_2 = 0\right\} \\
\mathcal{X}^{\vee}_{\mathbf{pk}} &= \left\{\mathbf{E}_{\mathbf{pk}}(\mathbf{u};\mathbf{r})\right\}.
\end{aligned}$$

Here, the value $|u_1|$ (in the definition of $\mathcal{L}^{\vee}_{\text{sim},(\mathbf{pk},\ell_{\vee})}$ is to be understood simply as the absolute value for signed $\mathbb{Z}_{|\mathbb{G}_2|}$-values in the prime-order setting, and as $|u_1| = |u_1|_N$ in the DCR setting. Observe that $\mathcal{L}^{\vee}_{\mathbf{pk}} \subseteq \mathcal{L}^{\vee}_{\text{sim},(\mathbf{pk},\ell_{\vee})} \subseteq \mathcal{L}^{\vee}_{\text{snd},\mathbf{pk}} \subseteq \mathcal{X}^{\vee}_{\mathbf{pk}}$. A valid witness for $x \in \mathcal{L}^{\vee}$ is $\mathbf{r}$.

A construction in pairing-friendly groups Now assume that $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$ is a prime-order group equipped with a symmetric pairing. Then, a benign proof system for $\mathcal{L}^{\vee}$ can be constructed from the universal hash proof systems for disjunctions from [1]. Specifically, [1] construct universal hash proof systems for languages of the form $\mathcal{L} = \{(x_1, x_2) \mid x_1 \in \mathcal{L}_1 \vee x_2 \in \mathcal{L}_2\}$, where $\mathcal{L}_i \subseteq \mathbb{G}^{\ell}$

are linear languages (i.e., vector spaces over $\mathbb{Z}_{|\mathbb{G}|}$). In our case, given $\mathbf{pk} = (epk_1, epk_2)$, we can thus set

$$\begin{aligned} \mathcal{L}_1 &= \{\mathbf{E}_{epk_1}(0;\mathbf{r})\} \\ \mathcal{L}_2 &= \{\mathbf{E}_{epk_2}(0;\mathbf{r})\} \\ \mathcal{L} &= \{x = (\mathbf{c}_0, c_1, c_2) \mid (\mathbf{c}_0, c_1) \in \mathcal{L}_1 \vee (\mathbf{c}_0, c_2) \in \mathcal{L}_2\}. \end{aligned} \tag{6}$$

Invoking [1] with these languages yields a NIDVPS $\mathbf{PS}^{\vee}_{\mathbf{pair}}$ that achieves:

Theorem 2. $\mathbf{PS}^{\vee}_{\mathbf{pair}}$ *is an* $(\mathcal{L}^{\vee}_{\mathrm{sim}}, \mathcal{L}^{\vee}_{\mathrm{ver}}, \mathcal{L}^{\vee}_{\mathrm{snd}})$*-benign NIDVPS for* $\mathcal{L}^{\vee}$.

A construction in the DCR setting In the following, we assume an $N = PQ$, and groups $\mathbb{G}$, $\mathbb{G}_1$, $\mathbb{G}_2$ as in Section 3.3. In particular, we have $\ell_{\mathbf{B}} = 1$, and $\mathbf{B}$ is the trivial (identity) matrix. Furthermore, fix an $\ell_{\vee} = \ell_{\vee}(\lambda)$. We additionally assume that $P, Q > 2^{\ell_{\vee}+4\lambda}$. Recall that $g_1, epk_1, epk_2 \in \mathbb{G}_1$ are of order $|\mathbb{G}_1| = \varphi(N)/4$, and that $g_2 \in \mathbb{G}_2$ is of order $|\mathbb{G}_2| = N$.

Our $(\mathcal{L}^{\vee}_{\mathrm{sim}}, \mathcal{L}^{\vee}_{\mathrm{ver}}, \mathcal{L}^{\vee}_{\mathrm{snd}})$-benign proof system $\mathbf{PS}^{\vee}_{\mathbf{DCR}}$ for $\mathcal{L}^{\vee}$ is given by the following algorithms:

- $\mathbf{PGen}^{\vee}(1^{\lambda})$ uniformly chooses $s_1, s_2 \in \mathbb{Z}_{\lfloor N^2/4 \rfloor}$ and then outputs $ppk_{\vee} = (epk_1^{s_1}, epk_1^{s_2})$ and $psk_{\vee} = (s_1, s_2)$.
- $\mathbf{PPrv}^{\vee}(ppk_{\vee}, x, r)$ (with $ppk_{\vee} = (epk_1^{s_1}, epk_1^{s_2})$, and $x = (c_0, c_1, c_2) = (g_1^r, epk_1^r, epk_2^r g_2^{u_2})$) uniformly chooses $t_1, t_2 \in \mathbb{Z}_N$, and outputs

$$\pi_{\vee} = (\pi_0, \pi_1, \pi_2) = \left(c_2^{t_1 + N \cdot t_2},\ (epk_1^{s_1})^r \cdot g_2^{t_1},\ (epk_1^{s_2})^r \cdot g_2^{t_2}\right).$$

- $\mathbf{PVer}^{\vee}(psk_{\vee}, x, \pi_{\vee})$ (for $psk = (s_1, s_2)$, $x = (c_0, c_1, c_2)$, and $\pi_{\vee} = (\pi_0, \pi_1, \pi_2)$) first checks that $\pi_1 / c_1^{s_1} = g_2^{t_1}$ and $\pi_2 / c_1^{s_2} = g_2^{t_2}$ for some $t_1, t_2 \in \mathbb{Z}_N$ (and outputs 0 if not). $\mathbf{PVer}$ then computes[5] these t_1, t_2, and outputs 1 iff $\pi_0 = c_2^{t_1 + N \cdot t_2}$.
- $\mathbf{PSim}^{\vee}(psk_{\vee}, x)$ (for $psk = (s_1, s_2)$ and $x = (c_0, c_1, c_2)$) uniformly picks $t_1, t_2 \in \mathbb{Z}_{N^2}$ and outputs

$$\pi_{\vee} = (\pi_0, \pi_1, \pi_2) = \left(c_2^{t_1 + N \cdot t_2},\ c_1^{s_1} \cdot g_2^{t_1},\ c_1^{s_2} \cdot g_2^{t_2}\right).$$

The completeness and zero-knowledge properties of $\mathbf{PS}^{\vee}_{\mathbf{DCR}}$ follow directly from the fact that $c_1^{s_i} = (epk_1^r)^{s_i} = (epk_1^{s_i})^r$. To show the soundness of $\mathbf{PS}^{\vee}_{\mathbf{DCR}}$, we prove a helpful technical lemma:

Lemma 3. *Let* s_1, s_2, t_1, t_2 *be distributed as in* $\mathbf{PS}^{\vee}_{\mathbf{DCR}}$*, and fix any* $u \in \mathbb{Z}$ *with* $|u| < 2^{\ell_{\vee}}$. *Let*[6]

$$aux := \left([s_1]_{\varphi(N)/4},\ [s_2]_{\varphi(N)/4},\ [t_1 + N \cdot t_2]_{\varphi(N)/4},\ [us_1 + t_1]_N,\ [us_2 + t_2]_N\right),$$

[5] Here, we implicitly use that computing discrete logarithms in $\mathbb{G}_2$ is easy, see Section 3.3.

[6] In this lemma and its proof, we heavily rely on the notation of $[s]_N$ and $[s]^N$ from Section 2.

and write[7] $w_1 := [s_1/\alpha]_N$ *(with the division performed in* $\mathbb{Z}_N$*) for* $\alpha := [N]_{\varphi(N)/4}$. *Then, for an independently random* $R \in \mathbb{Z}_{2^\lambda}$, *we have*

$$\varepsilon \;:=\; \mathbf{SD}\big(([w_1]_{2^\lambda}, aux)\,;\,(R, aux)\big) \;\leq\; 3/2^\lambda.$$

In other words, w_1 (and thus s_1) is unpredictable, even given aux.

Proof. Without loss of generality, assume $u \geq 0$. (For $u < 0$, we can invoke the lemma with $-u$, $-t_1$, and $-t_2$ in place of u, t_1 and t_2.) We proceed in steps, in each step modifying aux, and bounding the impact on ε. Specifically, in the following, we will define a number of random variables aux_i, and abbreviate $\varepsilon_i := \mathbf{SD}\big(([w_1]_{2^\lambda}, aux_i)\,;\,(R, aux_i)\big)$. As a starting point, consider

$$aux_1 \;:=\; ([t_1 + N \cdot t_2]_{\varphi(N)/4},\; [us_1 + t_1]_N,\; [us_2 + t_2]_N).$$

Now note that $w_1 = [s_1/\alpha]_N$ and the $[us_i + t_i]_N$ (for $i \in \{1, 2\}$) only depend on $[s_i]_N$. However, our uniform choice of $s_i \in \mathbb{Z}_{\lfloor N^2/4 \rfloor}$ is statistically $2/2^{\ell_\vee + 4\lambda}$-close to a uniform choice of $s_i \in \mathbb{Z}_{N \cdot \varphi(N)/4}$ (in which case $[s_i]_N$ and $[s_i]_{\varphi(N)/4}$ are independently and uniformly random). Hence, the $[s_i]_{\varphi(N)/4}$ are essentially independent of w_1 and aux_1, and we obtain $\varepsilon \leq \varepsilon_1 + 4/2^{\ell_\vee + 4\lambda}$. Next, consider

$$aux_2 \;:=\; ([t_1]_\alpha,\; [t_2]^\alpha,\; [t_1]^\alpha + [t_2]_\alpha,\; [us_1 + t_1]_N,\; [us_2 + t_2]_N).$$

Since $t_1 + \alpha \cdot t_2 = [t_1]_\alpha + \alpha \cdot ([t_1]^\alpha + [t_2]_\alpha) + \alpha^2 \cdot [t_2]^\alpha$, we have that aux_1 is a function of aux_2, and so $\varepsilon_1 \leq \varepsilon_2$. Similarly, we can refine the last two components of aux_2 to obtain

$$aux_3 \;:=\; ([t_1]_\alpha,\; [t_2]^\alpha,\; [t_1]^\alpha + [t_2]_\alpha,\; [us_1 + \alpha \cdot [t_1]^\alpha]_N,\; [us_2 + [t_2]_\alpha]_N).$$

Again, $\varepsilon_2 \leq \varepsilon_3$ since aux_3 fully defines aux_2. Similar to our first step, now $[t_1]_\alpha$ and $[t_2]^\alpha$ are essentially independent of the remaining parts of aux (up to a statistical defect of at most $2/2^{\ell_\vee + 4\lambda}$ for each). Hence, for

$$aux_4 \;:=\; ([t_1]^\alpha + [t_2]_\alpha,\; [us_1 + \alpha \cdot [t_1]^\alpha]_N,\; [us_2 + [t_2]_\alpha]_N),$$

we get that $\varepsilon_3 \leq \varepsilon_4 + 4/2^{\ell_\vee + 4\lambda}$. Now let $w_2 := [s_2]_N$, and consider

$$aux_5 \;:=\; ([t_1]^\alpha + [t_2]_\alpha,\; uw_1 + [t_1]^\alpha,\; uw_2 + [t_2]_\alpha).$$

Since aux_4 can be computed from aux_5, we have $\varepsilon_4 \leq \varepsilon_5$. Next, we release $w_1 + w_2$ (over $\mathbb{Z}$):

$$aux_6 \;:=\; ([t_1]^\alpha + [t_2]_\alpha,\; w_1 + w_2,\; uw_1 + [t_1]^\alpha).$$

Again, aux_5 can be computed from aux_6, and hence $\varepsilon_5 \leq \varepsilon_6$. Since we consider the statistical distance between $[w_1]_{2^\lambda}$ and R, we can release (and then drop) $[w_1]^{2^\lambda}$. Concretely, consider

$$aux_7 \;:=\; ([t_1]^\alpha + [t_2]_\alpha,\; [w_1]_{2^\lambda} + w_2,\; u \cdot [w_1]_{2^\lambda} + [t_1]^\alpha,\; [w_1]^{2^\lambda}),$$

[7] Here, we use our assumption that $[N]_{\varphi(N)/4} = P + Q - 1$ and N are coprime.

$$aux_8 := ([t_1]^\alpha + [t_2]_\alpha,\ [w_1]_{2^\lambda} + w_2,\ u \cdot [w_1]_{2^\lambda} + [t_1]^\alpha)$$
$$aux_9 := ([t_1]^\alpha + [t_2]_\alpha,\ u \cdot [w_1]_{2^\lambda} + [t_1]^\alpha).$$

Here, $\varepsilon_6 \leq \varepsilon_7$ since aux_6 can be computed from aux_7. Moreover, recall that $N > 2^{2\ell_\vee+8\lambda}$ by our choice of $P, Q > 2^{\ell_\vee+4\lambda}$. Hence, $\varepsilon_7 \leq \varepsilon_8 + 1/2^{2\ell_\vee+7\lambda}$, since $[w_1]_{2^\lambda}$ and $[w_1]^{2^\lambda}$ are independent up to a statistical defect of at most $1/2^{2\ell_\vee+7\lambda}$. Finally, $\varepsilon_8 \leq \varepsilon_9 + 1/2^{2\ell_\vee+7\lambda}$, since w_2 is uniformly and independently random chosen from $\mathbb{Z}_N$.

Similarly, we can show that $[t_2]_\alpha$ blinds $[[t_1]^\alpha]_{2^{\ell_\vee+2\lambda}}$:

$$aux_{10} := ([[t_1]^\alpha]_{2^{\ell_\vee+2\lambda}} + [t_2]_\alpha,\ u \cdot [w_1]_{2^\lambda} + [[t_1]^\alpha]_{2^{\ell_\vee+2\lambda}},\ [[t_1]^\alpha]^{2^{\ell_\vee+2\lambda}}),$$
$$aux_{11} := ([[t_1]^\alpha]_{2^{\ell_\vee+2\lambda}} + [t_2]_\alpha,\ u \cdot [w_1]_{2^\lambda} + [[t_1]^\alpha]_{2^{\ell_\vee+2\lambda}},$$
$$aux_{12} := (u \cdot [w_1]_{2^\lambda} + [[t_1]^\alpha]_{2^{\ell_\vee+2\lambda}}).$$

With the same reasoning as in aux_7-aux_9 (and using that $\alpha, N/\alpha > 2^{\ell_\vee+4\lambda}/2$ by $P, Q > 2^{\ell_\vee+4\lambda}$), we get $\varepsilon_9 \leq \varepsilon_{10}$, as well as $\varepsilon_{10} \leq \varepsilon_{11} + 1/2^{2\lambda}$, and $\varepsilon_{11} \leq \varepsilon_{12} + 1/2^{2\lambda}$. Finally, if we set $aux_{13} := ()$ to be the empty sequence, we get $\varepsilon_{12} \leq \varepsilon_{13} + 1/2^\lambda + 2/2^{\ell_\vee+4\lambda}$, since $[t_1]^\alpha$ is $2/2^{\ell_\vee+4\lambda}$-close to uniform over $\mathbb{Z}_{\lceil N/\alpha \rceil}$ (which implies that $[[t_1]^\alpha]_{2^{\ell_\vee+2\lambda}}$ blinds $u \cdot [w_1]_{2^\lambda}$). It is left to observe that $\varepsilon_{13} = \mathbf{SD}\big([w_1]_{2^\lambda}\,;\, R\big) \leq 1/2^{2\ell_\vee+7\lambda}$, since $w_1 \in \mathbb{Z}_N$ is uniformly random. Summing up, we get $\varepsilon \leq 1/2^\lambda + 2/2^{2\lambda} + 10/2^{\ell_\vee+4\lambda} + 3/2^{2\ell_\vee+7\lambda} \leq 3/2^\lambda$, as desired.

We can now proceed to show the soundness of $\mathbf{PS}^\vee_{\mathbf{DCR}}$:

Lemma 4. *$\mathbf{PS}^\vee_{\mathbf{DCR}}$ is statistically $(\mathcal{L}^\vee_{\text{sim}}, \mathcal{L}^\vee_{\text{ver}}, \mathcal{L}^\vee_{\text{snd}})$-sound in the sense of Definition 10. Concretely, for an adversary $\mathcal{A}$ that makes at most $q = q(\lambda)$ oracle queries in the soundness game from Definition 10,*

$$\text{Adv}^{\text{snd}}_{\mathbf{PS}^\vee_{\mathbf{DCR}},\mathcal{A}}(\lambda) \ \leq\ 4q/2^\lambda. \tag{7}$$

Proof. Fix $\ell_\vee$ and $\mathbf{pk}$, and let $\mathbf{view}_\mathcal{A}$ be $\mathcal{A}$'s view in a run of the computational soundness game from Definition 10. Specifically, $\mathbf{view}_\mathcal{A}$ consists of $\mathcal{A}$'s input $ppk_\vee = (epk_1^{s_1}, epk_1^{s_2})$, as well as all oracle queries (and the corresponding answers). We first consider to what extent $\mathbf{view}_\mathcal{A}$ determines the secret key $psk_\vee = (s_1, s_2)$.

- $\mathcal{A}$'s input $ppk_\vee = (epk_1^{s_1}, epk_1^{s_2})$ only depends on the values$[s_1]_{\varphi(N)/4}$ and $[s_2]_{\varphi(N)/4}$ (since epk_1 has order $\varphi(N)/4$).
- Each $\mathcal{O}_{\mathbf{sim}}$ oracle query of $\mathcal{A}$ reveals a value $\pi_\vee = (\pi_0, \pi_1, \pi_2) = (c_2^{t_1+N\cdot t_2}, c_1^{s_1} \cdot g_2^{t_1}, c_1^{s_2} \cdot g_2^{t_2})$ for $\mathcal{A}$-supplied c_1, c_2 and fresh t_1, t_2. We may assume that $c_1 = epk_1^r \cdot g_2^{u_1}$ and $c_2 = epk_2^r \cdot g_2^{u_2}$ with $u_1 = 0$ or $|u_1|_N < 2^{\ell_\vee} \wedge u_2 = 0$ (since otherwise, $\mathcal{O}_{\mathbf{sim}}$ rejects the query). Hence, such a query reveals

  $$(\pi_0, \pi_1, \pi_2) \ =\ (epk_2^{r(t_1+N\cdot t_2)} g_2^{u_2 t_1},\ epk_1^{rs_1} \cdot g_2^{u_1 s_1 + t_1},\ epk_1^{rs_2} \cdot g_2^{u_1 s_2 + t_2}),$$

  which only depends on $[s_1]_{\varphi(N)/4}$, $[s_2]_{\varphi(N)/4}$, $[t_1 + N \cdot t_2]_{\varphi(N)/4}$, $[u_2 t_1]_N$, as well as $[u_1 s_1 + t_1]_N$ and $[u_1 s_2 + t_2]_N$. Thus, if $u_1 = 0$, the query reveals only $[s_1]_{\varphi(N)/4}$ and $[s_2]_{\varphi(N)/4}$ about (s_1, s_2). But if $u_1 \neq 0$ (and thus $u_2 = 0$),

we can apply Lemma 3 with $u := u_1$, where we represent $u_1 \in \mathbb{Z}_N$ as an integer between $-N/2$ and $N/2$. This yields that the query leaves $[w_1]_{2^\lambda}$ undetermined, up to a small statistical defect. A hybrid argument over all of $\mathcal{A}$'s $\mathcal{O}_{\mathbf{sim}}$ queries shows that the overall statistical defect is bounded by $3q/2^\lambda$.

– An $\mathcal{O}_{\mathbf{ver}}$ query on input $(x, \pi_\vee)$ yields $\perp$ unless $x \in \mathcal{L}^\vee_{\mathrm{ver},(\mathbf{pk},\ell_\vee)} = \mathcal{L}^\vee_{(\mathbf{pk},\ell_\vee)}$. But for $x = (c_0, c_1, c_2) = (g_1^r, epk_1^r, epk_2^r g_2^{u_2}) \in \mathcal{L}^\vee_{(\mathbf{pk},\ell_\vee)}$, we get that $\mathcal{O}_{\mathbf{ver}}$'s output only depends on $c_1^{s_i} = epk_1^{rs_i}$, and hence only on $[s_i]_{\varphi(N)/4}$ (for $i = 1, 2$).

To summarize, $\mathbf{view}_{\mathcal{A}}$ is essentially independent of $[w_1]_{2^\lambda}$, up to a statistical defect of $3q/2^\lambda$.

It remains to prove that any $\mathcal{O}_{\mathbf{ver}}$ query on some $(x, \pi_\vee)$ with $x \in \mathcal{X}^\vee \backslash \mathcal{L}^\vee_{\mathrm{snd},\mathbf{pk}}$ (i.e., an x with $x = (c_0, c_1, c_2) = (g_1^r, epk_1^r \cdot g_2^{u_1}, epk_2^r \cdot g_2^{u_2})$ for $u_1, u_2 \in \mathbb{Z}_N^*$) is invalid in the sense that $\mathbf{PVer}(psk_\vee, x, \pi_\vee) = 0$ with high probability. To this end, write

$$\pi_\vee \;=\; (\pi_0, \pi_1, \pi_2) \;=\; (epk_2^{\rho_0} \cdot g_2^{\alpha_0},\; epk_1^{\rho_1} \cdot g_2^{\alpha_1},\; epk_1^{\rho_2} \cdot g_2^{\alpha_2})$$

for suitable $\rho_0, \rho_1, \rho_2, \alpha_0, \alpha_1, \alpha_2$. Recall that $(x, \pi_\vee)$ is valid only if for $i = 1, 2$, we have $\pi_i / c_1^{s_i} = g_2^{t_i}$ for some $t_i \in \mathbb{Z}_N$, and if $\pi_0 = c_2^{t_1 + N \cdot t_2}$ for those t_i. Hence, if $(x, \pi_\vee)$ is valid, then the following holds for some t_1, t_2:

$$\begin{aligned} \rho_0 &= [r(t_1 + N \cdot t_2)]_{\varphi(N)/4} & \alpha_0 &= [u_2 t_1]_N \\ \rho_1 &= [r s_1]_{\varphi(N)/4} & \alpha_1 &= [u_1 s_1 + t_1]_N \\ \rho_2 &= [r s_2]_{\varphi(N)/4} & \alpha_2 &= [u_1 s_2 + t_2]_N. \end{aligned}$$

By assumption, $u_2 \in \mathbb{Z}_N^*$, and thus α_0 determines t_1. Using also $u_1 \in \mathbb{Z}_N^*$, hence α_0 and α_1 determine $[s_1]_N$, and thus also $w_1 = [s_1/\alpha]_N$. However, as we have argued above, $\mathbf{view}_{\mathcal{A}}$ is essentially independent of $[w_1]_{2^\lambda}$. The probability that $\mathcal{A}$ correctly guesses an independently and uniformly random $[w_1]_{2^\lambda}$ with a single query is exactly $1/2^\lambda$. Since $\mathcal{A}$ makes at most q guesses, the probability for a correct guess is bounded by $q/2^\lambda$. Taking into account the mentioned statistical defect in $\mathbf{view}_{\mathcal{A}}$, we obtain (7).

Taking things together, we obtain

Theorem 3. $\mathbf{PS}^\vee_{\mathbf{DCR}}$ *is an* $(\mathcal{L}^\vee_{\mathrm{sim}}, \mathcal{L}^\vee_{\mathrm{ver}}, \mathcal{L}^\vee_{\mathrm{snd}})$*-benign NIDVPS for* $\mathcal{L}^\vee$.

6 The key encapsulation scheme

In the following, we present our main construction of an IND-MCCA secure key encapsulation (KEM) scheme. (This directly implies a PKE scheme with the same security properties [8].)

6.1 The construction

Ingredients and public parameters. In our construction, we use the following ingredients:

- groups $\mathbb{G}$, $\mathbb{G}_1$, $\mathbb{G}_2$ with $|\mathbb{G}_2| > 2^{3\lambda}$ (see Section 3.1 for a description of the generic setting),
- the generalized ElGamal scheme $(\mathbf{E}, \mathbf{D})$ implicitly defined through $\mathbb{G}$, $\mathbb{G}_1$, $\mathbb{G}_2$ (Section 3.1),
- an EUF-MOTCMA secure one-time signature scheme $\mathbf{OTS} = (\mathbf{SGen}, \mathbf{SSig}, \mathbf{SVer})$ (Section 4.1),
- a key extractor $\mathbf{EXT} = (\mathbf{ExtGen}, \mathbf{Ext}_{\mathrm{pub}}, \mathbf{Ext}_{\mathrm{priv}})$ for $\mathbb{G}$ (see Section 4.2) with $\ell_{\mathrm{ext}} = 3\lambda$,
- an $(\mathcal{L}^{\mathrm{lin}}_{\mathrm{sim}}, \mathcal{L}^{\mathrm{lin}}_{\mathrm{ver}}, \mathcal{L}^{\mathrm{lin}}_{\mathrm{snd}})$-benign proof system denoted with $\mathbf{PS}^{\mathrm{lin}} = (\mathbf{PGen}^{\mathrm{lin}}, \mathbf{PPrv}^{\mathrm{lin}}, \mathbf{PVer}^{\mathrm{lin}}, \mathbf{PSim}^{\mathrm{lin}})$ for $\mathcal{L}^{\mathrm{lin}}$ (Section 5.2),
- an $(\mathcal{L}^{\mathrm{hash}}_{\mathrm{sim}}, \mathcal{L}^{\mathrm{hash}}_{\mathrm{ver}}, \mathcal{L}^{\mathrm{hash}}_{\mathrm{snd}})$-benign proof system denoted $\mathbf{PS}^{\mathrm{hash}} = (\mathbf{PGen}^{\mathrm{hash}}, \mathbf{PPrv}^{\mathrm{hash}}, \mathbf{PVer}^{\mathrm{hash}}, \mathbf{PSim}^{\mathrm{hash}})$ for $\mathcal{L}^{\mathrm{hash}}$ (Section 5.3),
- an $(\mathcal{L}^{\vee}_{\mathrm{sim}}, \mathcal{L}^{\vee}_{\mathrm{ver}}, \mathcal{L}^{\vee}_{\mathrm{snd}})$-benign proof system denoted $\mathbf{PS}^{\vee} = (\mathbf{PGen}^{\vee}, \mathbf{PPrv}^{\vee}, \mathbf{PVer}^{\vee}, \mathbf{PSim}^{\vee})$ for $\mathcal{L}^{\vee}$ (Section 5.4) with $\ell_{\vee} = 3\lambda$, and
- a collision-resistant hash function generator **CRHF** (Section 2) with $\ell_H = 2\lambda$.[8]

We can use the presented generic constructions for **EXT**, $\mathbf{PS}^{\mathrm{lin}}$, and $\mathbf{PS}^{\mathrm{hash}}$, and, in the prime-order and DCR settings, the presented concrete constructions for **OTS** and $\mathbf{PS}^{\vee}$. (We note, however, that the DCR-based proof system $\mathbf{PS}^{\vee}_{\mathbf{DCR}}$ additionally requires that $|\mathbb{G}|$ has no prime factors smaller than $2^{7\lambda}$.) Specifically, we obtain instantiations both in the prime-order (with symmetric pairing) and DCR settings.

We also assume public parameters pp that contain whatever public parameters our building blocks require. Specifically, pp defines groups $\mathbb{G}$, $\mathbb{G}_1$, and $\mathbb{G}_2$ (as described in Section 3.1), and contains a hash function H output by **CRHF**.

The algorithms. Now our KEM **KEM** is defined through the following algorithms:

- $\mathbf{Gen}(1^\lambda)$ first uniformly picks $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_4 \in \mathbb{Z}^{\ell_{\mathbf{B}}}_{|\mathbb{G}_1|}$, and sets $(\mathbf{pk}, \mathbf{sk}) = (epk_i, esk_i)_{i=1}^4 = (g_1^{\boldsymbol{\omega}_i^\top \mathbf{B}}, \boldsymbol{\omega}_i)_{i=1}^4$. Next, **Gen** samples

$$
\begin{aligned}
(ppk_{\mathrm{lin}}, psk_{\mathrm{lin}}) &\leftarrow \mathbf{PGen}^{\mathrm{lin}}(1^\lambda, \mathbf{pk}) \\
(ppk_{\mathrm{hash}}, psk_{\mathrm{hash}}) &\leftarrow \mathbf{PGen}^{\mathrm{hash}}(1^\lambda, (epk_1, epk_2/epk_3)) \\
(ppk_{\vee,1}, psk_{\vee,1}) &\leftarrow \mathbf{PGen}^{\vee}(1^\lambda, (epk_1, epk_1)) \\
(ppk_{\vee,2}, psk_{\vee,2}) &\leftarrow \mathbf{PGen}^{\vee}(1^\lambda, (epk_4, epk_4)) \\
(ppk_{\vee,3}, psk_{\vee,3}) &\leftarrow \mathbf{PGen}^{\vee}(1^\lambda, (epk_4, epk_1)) \\
(ppk_{\vee,4}, psk_{\vee,4}) &\leftarrow \mathbf{PGen}^{\vee}(1^\lambda, (epk_2/epk_3, epk_4))
\end{aligned}
$$

[8] Since we assume collision-resistance (and not only target collision-resistance), we will have to take into account, e.g., birthday attacks on the hash function. This unfortunately entails $\ell_H \geq 2\lambda$.

$$
\begin{aligned}
(ppk_{\vee,5}, psk_{\vee,5}) &\leftarrow \mathbf{PGen}^{\vee}(1^{\lambda}, (epk_2/epk_3, epk_4)) \\
(ppk_{\vee,6}, psk_{\vee,6}) &\leftarrow \mathbf{PGen}^{\vee}(1^{\lambda}, (epk_2/epk_3, epk_1)) \\
(xpk, xsk) &\leftarrow \mathbf{ExtGen}(1^{\lambda}, epk_2),
\end{aligned}
$$

sets $\mathbf{ppk} = (ppk_{\text{lin}}, ppk_{\text{hash}}, ppk_{\vee,1}, \ldots, ppk_{\vee,6})$ and $\mathbf{psk} = (psk_{\text{lin}}, psk_{\text{hash}}, psk_{\vee,1}, \ldots, psk_{\vee,6})$, and finally outputs

$$
pk = (\mathbf{pk}, \mathbf{ppk}, xpk) \qquad\qquad sk = (\mathbf{sk}, \mathbf{psk}, xsk).
$$

– $\mathbf{Enc}(pk)$ (for pk as above) selects a random $\mathbf{r}$, and computes

$$
\begin{aligned}
\mathbf{c} = (\mathbf{c}_0, c_1, \ldots, c_4) &= \mathbf{E}(\mathbf{pk}, \mathbf{0}; \mathbf{r}) \\
(ovk, osk) &\leftarrow \mathbf{SGen}() \\
\tau &= H(ovk) \\
\pi_{\text{lin}} &\leftarrow \mathbf{PPrv}^{\text{lin}}(ppk_{\text{lin}}, \mathbf{c}, \mathbf{r}) \\
\pi_{\text{hash}} &\leftarrow \mathbf{PPrv}^{\text{hash}}(ppk_{\text{hash}}, ((\mathbf{c}_0, c_1, c_2/c_3), \tau), \mathbf{r}) \\
\pi_{\vee,1} &\leftarrow \mathbf{PPrv}^{\vee}(ppk_{\vee,1}, (\mathbf{c}_0, c_1, c_1/g_2), \mathbf{r}) \\
\pi_{\vee,2} &\leftarrow \mathbf{PPrv}^{\vee}(ppk_{\vee,2}, (\mathbf{c}_0, c_4, c_4/g_2), \mathbf{r}) \\
\pi_{\vee,3} &\leftarrow \mathbf{PPrv}^{\vee}(ppk_{\vee,3}, (\mathbf{c}_0, c_4, c_1/g_2), \mathbf{r}) \\
\pi_{\vee,4} &\leftarrow \mathbf{PPrv}^{\vee}(ppk_{\vee,4}, (\mathbf{c}_0, c_2/c_3, c_4), \mathbf{r}) \\
\pi_{\vee,5} &\leftarrow \mathbf{PPrv}^{\vee}(ppk_{\vee,5}, (\mathbf{c}_0, c_2/c_3, c_4/g_2), \mathbf{r}) \\
\pi_{\vee,6} &\leftarrow \mathbf{PPrv}^{\vee}(ppk_{\vee,6}, (\mathbf{c}_0, c_2/c_3, c_1/g_2), \mathbf{r}) \\
\boldsymbol{\pi} &= (\pi_{\text{lin}}, \pi_{\text{hash}}, \pi_{\vee,1}, \ldots, \pi_{\vee,6}) \\
\sigma &\leftarrow \mathbf{SSig}(osk, (\mathbf{c}, \boldsymbol{\pi})) \\
K &= \mathbf{Ext}_{\text{pub}}(xpk, (\mathbf{c}_0, c_2), \mathbf{r}).
\end{aligned}
$$

Here, we interpret $\tau = (\tau_1, \ldots, \tau_{2\lambda}) \in \{0,1\}^{2\lambda}$ as an integer $\tau = \sum_{i=1}^{2\lambda} 2^{i-1}\tau_i \in \{0, \ldots, 2^{2\lambda} - 1\}$, with τ_1 being interpreted as the least significant bit.
The final output of **Enc** is $C = (\mathbf{c}, \boldsymbol{\pi}, ovk, \sigma)$ and K.

– $\mathbf{Dec}(sk, C)$ (for sk and C as above), first verifies σ and all proofs in $\boldsymbol{\pi}$ using ovk and $\mathbf{sk}$, and, if all are valid, returns

$$
K = \mathbf{Ext}_{\text{priv}}(xsk, (\mathbf{c}_0, c_2)).
$$

Explanation. The proofs in $\boldsymbol{\pi}$ require some explanation. They prove various (seemingly highly redundant) properties of the vector $\mathbf{u} = (u_i)_{i=1}^4 \in \mathbb{Z}_{|\mathbb{G}_2|}^4$ encrypted in $\mathbf{c}$. Some of these properties will be violated in different steps of our security analysis already by the security game, and we will then rely on the remaining properties. For instance, π_{lin} always guarantees that the vectors $\mathbf{u}$ encrypted in decryption queries lie in the subspace spanned by the vectors $\mathbf{u}$ encrypted in challenge ciphertexts. (That subspace is initially trivial, since honest encryptions contain $\mathbf{u} = \mathbf{0}$, but will be larger in later parts of the analysis.)

π_{hash} guarantees that $\tau u_1 = u_2 - u_3$ in $\mathcal{A}$'s decryption queries (unless generated challenge ciphertexts already violate that relation).

The $\mathbf{PS}^\vee$-proofs $\pi_{\vee,i}$ are a bit more delicate. First, $\pi_{\vee,1}$ and $\pi_{\vee,2}$ guarantee that $u_1, u_4 \in \{0,1\}$. The condition $u_1 \in \{0,1\}$ only simplifies the analysis, but $u_4 \in \{0,1\}$ is instrumental to enforce our partitioning strategy. In particular, u_4 will be the bit that determines the partitioning of ciphertexts in our partitioning argument. Depending on the value of u_4, $\pi_{\vee,4}$ and $\pi_{\vee,5}$ give further guarantees: $\pi_{\vee,4+b}$ guarantees $u_2 = u_3 \vee u_4 = b$. At each point in our analysis, at least one of these conditions (for one value of b) is never violated. Hence, $u_2 = u_3$ is guaranteed in decryption queries whenever $u_4 \neq b$. Finally, the proofs $\pi_{\vee,3}$ and $\pi_{\vee,6}$ ensure technical conditions ($u_4 = 0 \vee u_1 = 1$ and $u_2 = u_3 \vee u_1 = 1$) that will help to deal with the somewhat limited soundness guarantees of $\mathbf{PS}^\vee$. (In particular, these proofs help to cope with the fact that the soundness game of $\mathbf{PS}^\vee$only allows a limited type of verification queries.)

Correctness. The correctness of **KEM** follows directly from the correctness of the underlying primitives.

6.2 Security analysis

Theorem 4 (Security of KEM). *If the ingredients from Section 6.1 are secure, then* **KEM** *is IND-MCCA secure. Specifically, for every IND-MCCA adversary $\mathcal{A}$ that makes at most q oracle queries, there are adversaries $\mathcal{B}^{\text{crhf}}$, $\mathcal{B}^{\text{ots}}$, $\mathcal{B}^{\text{fact}}$, $\mathcal{B}^{\text{mccpa}}$, $\mathcal{B}^{\text{lin}}$, $\mathcal{B}^{\text{hash}}$, and $\mathcal{B}^\vee$ with*

$$\begin{aligned}|\text{Adv}^{\text{mcca}}_{\mathbf{KEM},\mathcal{A}}(\lambda)| \;\leq\; & \text{Adv}^{\text{crhf}}_{\mathbf{CRHF},\mathcal{B}^{\text{crhf}}}(\lambda) + \text{Adv}^{\text{ots}}_{\mathbf{OTS},\mathcal{B}^{\text{ots}}}(\lambda) \\ & + \mathbf{O}(\lambda)\text{Adv}^{\text{fact}}_{\mathbb{G}_2,\mathcal{B}^{\text{fact}}}(\lambda) + \mathbf{O}(\lambda)\text{Adv}^{\text{mccpa}}_{\mathbb{G},\mathcal{B}^{\text{mccpa}}}(\lambda) + \mathbf{O}(\lambda)\text{Adv}^{\text{snd}}_{\mathbf{PS}^{\text{lin}},\mathcal{B}^{\text{lin}}}(\lambda) \\ & + \mathbf{O}(\lambda)\text{Adv}^{\text{snd}}_{\mathbf{PS}^{\text{hash}},\mathcal{B}^{\text{hash}}}(\lambda) + \mathbf{O}(\lambda)\text{Adv}^{\text{snd}}_{\mathbf{PS}^\vee,\mathcal{B}^\vee}(\lambda) + \mathbf{O}(\lambda q)/2^\lambda. \quad (8)\end{aligned}$$

Outline. The goal of our proof will be to randomize all keys handed out by $\mathcal{O}_{\mathbf{enc}}$ along with challenge ciphertexts. In order to do so, we rely on the indistinguishability of the key extractor **EXT**. However, to apply **EXT**'s indistinguishability (Definition 8), we first need to establish a certain kind of "unfairness". Specifically, we will randomize the u_2 component of all challenge ciphertexts, while rejecting all decryption queries with $u_2 \neq 0$. (Note that this in particular means that the experiment does not need to be able to decrypt challenge ciphertexts.)

Establishing this unfairness thus is the key to proving chosen-ciphertext security. But it will also form the main difficulty of the proof, and we will outsource this process into several helper lemmas.

A little more specifically, we will proceed as hinted in the introduction. Namely, to show that all adversarial decryption queries with $u_2 \neq 0$ are rejected, we will gradually add more and more restrictive additional checks on decryption queries with $u_2 \neq 0$. In particular, if $u_2 \neq 0$, then we will require that u_2 "authenticates" the full ciphertext, in the sense that $u_2 = T$ for a ciphertext-dependent "authentication token" T (cf. also the description in the introduction). We will

change the definition of T in a series of transformations such that eventually $T = X + \tau$, where X is a fixed secret value, and τ is the (up to **OTS**-forgeries and **CRHF**-collisions ciphertext-unique) value of $\tau = H(ovk)$ from above. However, we will only be able to prove that the adversary must *reuse* a previously used authentication in his decryption queries. This means that the following rules apply:

- Any challenge ciphertext handed to the adversary satisfies $u_2 = X + \tau$.
- Any decryption query with $u_2 \neq 0$ must satisfy $u_2 = X + \tau^{(j)}$ for *some* $\tau^{(j)}$ from a challenge ciphertext. (Hence, the adversary must "reuse" an authentication tag.)

Additionally, all challenge ciphertexts will satisfy $u_1 = 1$ and $u_3 = X$. Hence, using the soundness of our benign proof systems $\mathbf{PS}^{\mathrm{lin}}$ and $\mathbf{PS}^{\vee}$, also any decryption query with $u_1 \neq 0$ will have to satisfy $u_1 = 1$ and $u_3 = X$ (or it is rejected). Finally invoking the soundness of $\mathbf{PS}^{\mathrm{hash}}$ (on the equation $u_2 = u_3 + \tau \cdot u_1$, which is fulfilled in all challenge ciphertexts), we obtain that also decryption queries will have to satisfy $u_2 = X + \tau$ for the respective value τ from that decryption query.

Hence, the requirements on adversarial decryption queries with $u_2 \neq 0$ are now that $u_2 = X + \tau$ and $u_2 = X + \tau^{(j)}$, and thus that $\tau = \tau^{(j)}$ for some $\tau^{(j)}$ from a previous challenge. Since the value τ is ciphertext-unique, we obtain a contradiction. (Thus, any decryption query with $u_2 \neq 0$ is rejected.)

Due to lack of space, we have to postpone our proof (and in particular the more complex argument for establishing the requirement $u_2 = X + \tau^{(j)}$ on adversarial decryption queries) to the full version [16].

References

[1] Michel Abdalla, Fabrice Benhamouda, and David Pointcheval. "Disjunctions for Hash Proof Systems: New Constructions and Applications". In: *Proc. EUROCRYPT (2) 2015*. Vol. 9057. Lecture Notes in Computer Science. Springer, 2015, pp. 69–100.

[2] Masayuki Abe, Bernardo David, Markulf Kohlweiss, Ryo Nishimaki, and Miyako Ohkubo. "Tagged One-Time Signatures: Tight Security and Optimal Tag Size". In: *Proc. Public Key Cryptography 2013*. Vol. 7778. Lecture Notes in Computer Science. Springer, 2013, pp. 312–331.

[3] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. "A Framework for Identity-Based Encryption with Almost Tight Security". In: *Proc. ASIACRYPT (1) 2015*. Vol. 9452. Lecture Notes in Computer Science. Springer, 2015, pp. 521–549.

[4] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. "Public-Key Encryption in a Multi-user Setting: Security Proofs and Improvements". In: *Proc. EUROCRYPT 2000*. Vol. 1807. Lecture Notes in Computer Science. Springer, 2000, pp. 259–274.

[5] Olivier Blazy, Eike Kiltz, and Jiaxin Pan. "(Hierarchical) Identity-Based Encryption from Affine Message Authentication". In: *Proc. CRYPTO (1) 2014*. Vol. 8616. Lecture Notes in Computer Science. Springer, 2014, pp. 408–425.

[6] Jan Camenisch and Victor Shoup. "Practical Verifiable Encryption and Decryption of Discrete Logarithms". In: *Proc. CRYPTO 2003*. Vol. 2729. Lecture Notes in Computer Science. Springer, 2003, pp. 126–144.

[7] Jie Chen and Hoeteck Wee. "Fully, (Almost) Tightly Secure IBE and Dual System Groups". In: *Proc. CRYPTO (2) 2013*. Vol. 8043. Lecture Notes in Computer Science. Springer, 2013, pp. 435–460.

[8] Ronald Cramer and Victor Shoup. "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack". In: *SIAM J. Comput.* 33.1 (2003), pp. 167–226.

[9] Ronald Cramer and Victor Shoup. "Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption". In: *Proc. EUROCRYPT 2002*. Vol. 2332. Lecture Notes in Computer Science. Springer, 2002, pp. 45–64.

[10] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge L. Villar. "An Algebraic Framework for Diffie-Hellman Assumptions". In: *Proc. CRYPTO (2) 2013*. Vol. 8043. Lecture Notes in Computer Science. Springer, 2013, pp. 129–147.

[11] Roger Fischlin and Claus-Peter Schnorr. "Stronger Security Proofs for RSA and Rabin Bits". In: *Proc. EUROCRYPT 1997*. Vol. 1233. Lecture Notes in Computer Science. Springer, 1997, pp. 267–279.

[12] Romain Gay, Dennis Hofheinz, Eike Kiltz, and Hoeteck Wee. "Tightly CCA-Secure Encryption Without Pairings". In: *Proc. EUROCRYPT (1) 2016*. Vol. 9665. Lecture Notes in Computer Science. Springer, 2016, pp. 1–27.

[13] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks". In: *SIAM J. Comput.* 17.2 (1988), pp. 281–308.

[14] Junqing Gong, Jie Chen, Xiaolei Dong, Zhenfu Cao, and Shaohua Tang. "Extended Nested Dual System Groups, Revisited". In: *Proc. Public Key Cryptography (1) 2016*. Vol. 9614. Lecture Notes in Computer Science. Springer, 2016, pp. 133–163.

[15] Brett Hemenway and Rafail Ostrovsky. "Extended-DDH and Lossy Trapdoor Functions". In: *Proc. Public Key Cryptography 2012*. Vol. 7293. Lecture Notes in Computer Science. Springer, 2012, pp. 627–643.

[16] Dennis Hofheinz. *Adaptive partitioning*. IACR ePrint Archive, report 2016/373. `http://eprint.iacr.org/2016/373`. 2016.

[17] Dennis Hofheinz. "Algebraic Partitioning: Fully Compact and (almost) Tightly Secure Cryptography". In: *Proc. TCC (A1) 2016*. Vol. 9562. Lecture Notes in Computer Science. Springer, 2016, pp. 251–281.

[18] Dennis Hofheinz and Tibor Jager. "Tightly Secure Signatures and Public-Key Encryption". In: *Proc. CRYPTO 2012*. Vol. 7417. Lecture Notes in Computer Science. Springer, 2012, pp. 590–607.

[19] Dennis Hofheinz and Eike Kiltz. "Secure Hybrid Encryption from Weakened Key Encapsulation". In: *Proc. CRYPTO 2007*. Vol. 4622. Lecture Notes in Computer Science. Springer, 2007, pp. 553–571.

[20] Dennis Hofheinz and Eike Kiltz. "The Group of Signed Quadratic Residues and Applications". In: *Proc. CRYPTO 2009*. Vol. 5677. Lecture Notes in Computer Science. Springer, 2009, pp. 637–653.

[21] Dennis Hofheinz, Jessica Koch, and Christoph Striecks. "Identity-Based Encryption with (Almost) Tight Security in the Multi-instance, Multi-ciphertext Setting". In: *Proc. Public Key Cryptography 2015*. Vol. 9020. Lecture Notes in Computer Science. Springer, 2015, pp. 799–822.

[22] Susan Hohenberger and Brent Waters. "Realizing Hash-and-Sign Signatures under Standard Assumptions". In: *Proc. EUROCRYPT 2009*. Vol. 5479. Lecture Notes in Computer Science. Springer, 2009, pp. 333–350.

[23] Eike Kiltz and Hoeteck Wee. "Quasi-Adaptive NIZK for Linear Subspaces Revisited". In: *Proc. EUROCRYPT (2) 2015*. Vol. 9057. Lecture Notes in Computer Science. Springer, 2015, pp. 101–128.

[24] Hugo Krawczyk and Tal Rabin. "Chameleon Signatures". In: *Proc. NDSS 2000*. The Internet Society, 2000.

[25] Kaoru Kurosawa and Yvo Desmedt. "A New Paradigm of Hybrid Encryption Scheme". In: *Proc. CRYPTO 2004*. Vol. 3152. Lecture Notes in Computer Science. Springer, 2004, pp. 426–442.

[26] Benoît Libert, Marc Joye, Moti Yung, and Thomas Peters. "Concise Multi-challenge CCA-Secure Encryption and Signatures with Almost Tight Security". In: *Proc. ASIACRYPT (2) 2014*. Vol. 8874. Lecture Notes in Computer Science. Springer, 2014, pp. 1–21.

[27] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. "Compactly Hiding Linear Spans - Tightly Secure Constant-Size Simulation-Sound QA-NIZK Proofs and Applications". In: *Proc. ASIACRYPT (1) 2015*. Vol. 9452. Lecture Notes in Computer Science. Springer, 2015, pp. 681–707.

[28] Moni Naor and Omer Reingold. "Number-theoretic constructions of efficient pseudo-random functions". In: *J. ACM* 51.2 (2004), pp. 231–262.

[29] Pascal Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes". In: *Proc. EUROCRYPT 1999*. Vol. 1592. Lecture Notes in Computer Science. Springer, 1999, pp. 223–238.

[30] Hovav Shacham. *A Cramer-Shoup Encryption Scheme from the Linear Assumption and from Progressively Weaker Linear Variants*. IACR ePrint Archive, report 2007/74. `http://eprint.iacr.org/2007/74`. 2007.