All-But-Many Encryption
A New Framework for Fully-Equipped UC Commitments

Eiichiro Fujisaki

NTT Secure Platform Laboratories
fujisaki.eiichiro@lab.ntt.co.jp

Abstract. We present a general framework for constructing non-interactive universally composable (UC) commitment schemes that are secure against adaptive adversaries in the non-erasure model under a re-usable common reference string. Previously, such "fully-equipped" UC commitment schemes have been known only in [5, 6], with *strict* expansion factor $O(\kappa)$; meaning that to commit λ bits, communication strictly requires $O(\lambda\kappa)$ bits, where κ denotes the security parameter. Efficient construction of a fully-equipped UC commitment scheme is a long-standing open problem. We introduce new abstraction, called *all-but-many encryption* (ABME), and prove that it captures a fully-equipped UC commitment scheme. We propose the first fully-equipped UC commitment scheme with *optimal expansion factor* $\Omega(1)$ from our ABME scheme related to the DCR assumption. We also provide an all-but-many lossy trapdoor function (ABM-LTF) [18] from our DCR-based ABME scheme, with a better lossy rate than [18].

1 Introduction

1.1 Motivating Application: Fully-Equipped UC Commitments

Universal composability (UC) framework [4] guarantees that if a protocol is proven secure in the UC framework, it remains secure even if it is run concurrently with arbitrary (even insecure) protocols. This composable property gives a designer a fundamental benefit, compared to the classic definitions, which only guarantee that a protocol is secure if it is run in the standalone setting. UC commitments are an essential ingredient to construct high level UC-secure protocols, which imply UC zero-knowledge protocols [5, 10] and UC oblivious transfer [6], thereby meaning that any UC-secure two-party and multi-party computations can be realized in the presence of UC commitments. Since UC commitments cannot be realized without an additional set-up assumption [5], the common reference string (CRS) model is widely used. A commitment scheme consists of a two-phase protocol between two parties, a committer and a receiver. In the commitment phase, a committer gives a receiver the digital equivalent of a *sealed envelope* containing value x, and, in the opening phase, the committer reveals x in a way that the receiver can verify it. From the original concept, it is required that a committer cannot change the value inside the envelope (*binding property*), whereas the receiver can learn nothing about x (*hiding property*) unless

the committer helps the receiver opens the envelope. Informally, a UC commitment scheme maintains the above binding and hiding properties under *any concurrent composition with arbitrary protocols*. To achieve this, a UC commitment scheme requires *equivocability* and *extractability* at the same time. Informally, equivocability of UC commitments in the CRS model can be interpreted as follows: An algorithm (called the simulator) that takes the secret behind the CRS string can generate an *equivocal* commitment that can be opened to any value. On the other hand, extractability can be interpreted as the ability of the simulator extracting the contents of a commitment generated by any adversarial algorithm, even after the adversary saw many equivocal commitments generated by the simulator.

Several factors as shown below feature UC commitments:

Non-Interactivity. If an execution of a commitment scheme is completed, simply by sending each one message from the committer to the receiver both in the commitment and opening phases, then it is called *non-interactive*; otherwise, interactive. From a practical viewpoint, non-interactivity is definitely favorable – non-interactive protocols are much easier to implement and more resilient to real threats such as denial of service attacks. Even from a theoretical viewpoint, non-interactive protocols generally make security proofs simpler.

CRS Re-usability. The CRS model assumes that CRS strings are generated in a trusted way and given to every party. For practical use, it is very important that a global single CRS string can be fixed beforehand and it can be *re-usable* in an unbounded number of executions of cryptographic protocols. Otherwise, a new CRS string must be set up in a trusted way every time when a new execution of a protocol is invoked.

Adaptive Security. If an adversary decides to corrupt parities only before a protocol starts, it is called a static adversary. On the other hand, if an adversary can decide to corrupt parties at any point in the executions of protocols, it is called an *adaptive* adversary. The attacks of adaptive adversaries are more realistic in the real world. So, adaptive UC security is more desirable.

Non-Erasure Model. When a party is corrupted, its complete inner state is revealed, including the randomness being used. Some protocols are only proven UC-secure under the assumption that the parties can securely erase their inner states at any point of an execution. However, reliable erasure is a difficult task on a real system. So, it is desirable that a *non-erasure* protocol is proven secure.

1.2 Previous Works

Canetti and Fischlin [5] presented the first UC secure commitment schemes. One of their proposals is "fully-equipped" – *non-interactive and adaptively secure in the non-erasure model under a global re-usable common reference string*. By construction, however, the proposal strictly requires, to commit to λ-bit secret, $O(\lambda\kappa)$ bits in communication and $O(\lambda)$ modular exponentiations in computation. Canetti et al. [6] also proposed another fully-equipped UC commitment scheme only from (enhanced) trapdoor permutations. It requires general non-interactive zero knowledge proofs and is simply inefficient.

So far, these two have been the only known fully-equipped UC commitment schemes. The known subsequent constructions of UC commitments [10, 8, 3, 22, 20, 13] have improved efficiency, but *sacrifice* at least one or a few requirements [1]. Efficient construction of a fully-equipped UC commitment scheme is a long-standing open problem.

1.3 Our Contribution

The UC framework is complicated with many subtleties. Therefore, it is desirable to translate the essence of basic UC secure protocols into simple cryptographic primitives. We introduce special tag-based public key encryption (Tag-PKE) that we call *all-but-many encryption* (ABME), and prove that it implies "fully-equipped" UC commitments. We propose a compact ABME scheme related to the DCR assumption and thereby the first fully-equipped UC commitment scheme with optimal expansion factor $\Omega(1)$. To commit λ bit, it requires $\Omega(\kappa)$ bits and a constant number of modular exponentiations. We also present an all-but-many lossy trapdoor function (ABM-LTF) [18] from our DCR-based ABME scheme, with a better lossy rate than [18].

In the full version [14], we present an ABME scheme from the DDH assumption with overhead $\Omega(\kappa/\log\kappa)$, which is slightly better than the prior work (with $\Omega(\kappa)$). We also present a fully-equipped UC commitment scheme from a *weak* ABME scheme under the general assumption (where (enhanced) trapdoor permutations exist), which is far more efficient than the prior scheme [6] under the same assumption.

Our Approach: All-But-Many Encryption. In an ABME scheme, a secret-key holding user (i.e., the simulator in the UC framework) can generate a *fake* ciphertext, which can be opened to any message with consistent randomness. On the other hand, it must be infeasible for a secret-key non-holding user (i.e., the adversary in the UC framework) (1) to distinguish a fake ciphertext from a *real* (honestly generated) ciphertext, even after the message and randomness are revealed, and (2) to produce a fake ciphertext (on a fresh tag) even given many fake ciphertexts.

To realize such a scheme, we divide its functionality into two primitives, called *probabilistic pseudo random functions* (PPRF) and *extractable sigma protocols* ($\mathsf{ext}\Sigma$). The former is a kind of a probabilistic version of a pseudo random function (family) in the public parameter model. The latter is special sigma (i.e., canonical 3-round public-coin HVZK) protocols [7] with some extractability. The concept of extractable sigma protocols is not completely new. A weaker notion, called *weak* extractable sigma protocols, appears in [15] to construct a few (interactive) simulation sound trapdoor commitment (SSTC) schemes. See also [16, 21, 17] for SSTC. This paper requires a stronger notion and its realization, which employed in a different framework. If two primitives are successfully combined, an ABME scheme can be constructed. We discuss more in the following.

[1] Only [22] and [13] satisfy all but one requirement. [22] does not satisfy CRS reusability, whereas [13] does not support the non-erasure model.

Probabilistic Pseudo-Random Function (PPRF). A PPRF $= (\mathsf{Gen}^{\mathsf{spl}}, \mathsf{Spl})$ is a probabilistic version of a pseudo random function family in the public parameter model. $\mathsf{Gen}_{\mathsf{spl}}(1^\kappa)$ generates a pair of public-key/seed (pk, w), and A PPT algorithm Spl takes (pk, w, t) and outputs (or *samples*) $u \leftarrow \mathsf{Spl}(pk, w, t)$. Let $L_{pk}(t) = \{u | \exists(w, v) : u = \mathsf{Spl}(pk, w, t, v)\}$. Informally, a PPRF requires that (a) u looks pseudo-random on any t (*pseudo randomness*) and (b) it is infeasible for any adversary to find u^* in *some super set*, $\widehat{L}_{pk}(t^*)$, of $L_{pk}(t^*)$ on any fresh t^*, even after it has access to oracle $\mathsf{Spl}(pk, w, \cdot)$ (*unforgeability on* $\widehat{L}_{pk}$), where $\widehat{L}_{pk} := \{(t, u) \,|\, u \in \widehat{L}_{pk}(t)\}$. The super set $\widehat{L}_{pk}$ will be clear later.

Extractable Sigma Protocols. An extractable sigma protocol is a special sigma protocol *associated with a language-generation algorithm and a decryption algorithm*. Recall the sigma protocols [7]. A sigma protocol Σ on NP language L is a canonical 3-round public coin interactive proof system such that the prover can convince the verifier that he knows the witness w behind common input $x \in L$, where the prover first sends commitment a; the verifier sends back challenge (public-coin) e; the prover responds with z; and the verifier finally accepts or rejects the conversation (a, e, z) on x. A sigma protocol is associated with a simulation algorithm $\mathsf{sim}\Sigma$ that takes x (regardless of whether $x \in L$ or not) and challenge e, and produces an accepting conversation $(a, e, z) \leftarrow \mathsf{sim}\Sigma(x, e)$ *without witness* w. It is guaranteed that, if $x \in L$, the distributions (a, e, z) produced by $\mathsf{sim}\Sigma(x, e)$ on random e is statistically indistinguishable from the transcript generated between two honest parties, called *honest-verifier statistically zero knowledge* (HVSZK). If $x \notin L$, for every a, there is unique e if there is an accepting conversation (a, e, z), which is called *special soundness*.

An extractable sigma protocol $\mathsf{ext}\Sigma = (\mathsf{Gen}^{\mathsf{ext}}, \Sigma, \mathsf{Dec})$ uses two more algorithms: The language-generation algorithm $\mathsf{Gen}^{\mathsf{ext}}$ outputs a pair of public/secret keys, (pk, sk), where pk determines two disjoint sets L_{pk} and L^{ext}_{pk}. Here sigma protocol Σ works on L_{pk} and the decryption algorithm Dec works on L^{ext}_{pk}, meaning that $\mathsf{Dec}(sk, x, a)$ outputs challenge e if $x \in L^{\mathsf{ext}}_{pk}$ and if an accepting conversation (a, e, z) exists on x. Due to special soundness, e is uniquely determined if $x \notin L_{pk}$. Therefore, the decryption algorithm is well defined.

Combining them. Suppose $\mathsf{ext}\Sigma$ and PPRF are so well combined that, for $(L_{pk}, L^{\mathsf{ext}}_{pk})$ generated by $\mathsf{Gen}^{\mathsf{ext}}$, L_{pk} is the language derived from PPRF and PPRF is unforgeable on $\widehat{L}_{pk}$ $(:= U'_{pk} \backslash L^{\mathsf{ext}}_{pk})$, where U'_{pk} denotes the entire set with respects to pk. We can then transform the extractable sigma protocol into an ABME scheme in the similar way that a sigma protocol is converted to an instance-dependent commitment scheme [2, 19]. To encrypt message e on tag t, a sender picks random u, runs $\mathsf{sim}\Sigma$ on instance (t, u) with challenge e, to get $(a, e, z) \leftarrow \mathsf{sim}\Sigma(pk, (t, u), e)$, and finally outputs (t, u, a). Due to unforgeability of PPRF, it holds that $(t, u) \in U'_{pk} \backslash \widehat{L}_{pk}$ with an overwhelming probability. Then, e is uniquely determined given $((t, u), a)$, as long as an accepting conversation (a, e, z) exists on (t, u). By our precondition, we can decrypt (t, u, a) using sk, as $e = \mathsf{Dec}(sk, (t, u), a)$ because $(t, u) \in L^{\mathsf{ext}}_{pk}$. On the other hand, a fake ciphertext on tag t is produced using (w, v) as follows: one sets $u := \mathsf{Spl}(pk, w, t; v)$, with random v, where $(t, u) \in L_{pk}$, and computes a, as similarly as an honest prover

computes the first message on common input (t, u) with witness (w, v). To open a to e, he produces the third message z in the sigma protocol. It is obvious by construction that he can open a to any e because $(t, u) \in L_{pk}$.

Realizing Extractable Sigma Protocols. Although sigma protocols (with HVSZK) exist on *many* NP languages, it is not known how to extract the challenge as discussed above. The following is our key observation to realize the functionality. Sigma protocols are often implemented on Abelian groups associated with homomorphic maps, in which the first message of such sigma protocols implies a system of linear equations with e and z. Hence, there is a matrix derived from the linear systems. Due to completeness and special soundness, there is an invertible (sub) square matrix if and only if $x \notin L_{pk}$ (provided that the linear system is defined in a finite field). Therefore, if one knows the contents of the matrix, one can solve the linear systems when $x \notin L_{pk}$ and obtain e if its length is logarithmic. Suppose for instance that L_{pk} is the DDH language – it does not form a PPRF, but a good toy case to explain how to extract the challenge. Let $(g_1, g_2, h_1, h_2) \notin L_{pk}$, meaning that $x_1 \neq x_2$ where $x_1 := \log_{g_1}(h_1)$ and $x_2 := \log_{g_2}(h_2)$. The first message (A_1, A_2) of a canonical sigma protocol on L_{pk} implies linear equations

$$\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ \alpha & \alpha x_2 \end{pmatrix} \begin{pmatrix} z \\ e \end{pmatrix} \tag{1}$$

where $A_1 = g_1^{a_1}$, $A_2 = g_2^{a_2}$, and $g_2 = g_1^{\alpha}$. The above matrix is invertible if and only if $(g_1, g_2, h_1, h_2) \notin L_{pk}$. We note that e is expressed as a linear combination of a_1 and a_2, i.e., $\beta_1 a_1 + \beta_2 a_2$, where the coefficients are determined by the matrix. Therefore, if the decryption algorithm takes (α, x_1, x_2) and the length of e is logarithmic, it can find out e by checking whether $g_1^e = A_1^{\beta_1} A_2^{\beta_2}$ or not. In the case when a partial information on the values of the matrix is given, the decryption algorithm can still find logarithmic-length e if the matrix is made so that e can be expressed as a *linear* combination of *unknown values* – the unknown values do not appear with a quadratic form or a more degree of forms in the equations.

In some case, we might be able to invert a homomorphic map, such as $f(a) = g^a$, using trapdoor f^{-1}. Then, the decryption algorithm can obtain (a_1, a_2) as well as the entire values of the matrix and hence extract the entire (polynomial-length) e. This happens in our DCR based implementation. In the case, the equivalent condition that the matrix is invertible is, not $x \notin L_{pk}$, but $x \notin \widehat{L}_{pk}$ for some superset $\widehat{L}_{pk}$, since the corresponding linear system is defined not on a finite field, but on a finite ring, such as $\mathbb{Z}_{n^d}$. This means that we require unforgeability on $\widehat{L}_{pk}$, so as to make an adversary output $x = (t, u)$ in $L_{pk}^{\mathsf{ext}} = U'_{pk} \backslash \widehat{L}_{pk}$.

1.4 Other Related Works

Simulation-based selective opening CCA (SIM-SO-CCA) secure PKE [12] is related to ABME, but both are incomparable. Indeed, the SIM-SO-CCA secure PKE scheme proposed in [12] does not satisfy the notion of ABME. On the other

hand, ABME does not satisfy SIM-SO-CCA PKE, because it does not support CCA security. Although the scheme in [12] could be tailored to a fully-equipped UC commitment scheme, it cannot overcome the barrier of expansion factor $O(\kappa)$, because it strictly costs $O(\lambda\kappa)$ bits to encrypt λ bit.

Hofheinz has presented the notion of all-but-many lossy trapdoor function (ABM-LTF) [18], mainly to construct indistinguishable-based selective opening CCA (IND-SO-CCA) secure PKE schemes. ABM-LTF is lossy trapdoor function (LTF) [25] with (unbounded) *many* lossy tags. The relation between ABM-LTF and ABME is a generalized analogue of LTF and lossy encryption [24, 1] with unbounded many loss tags. However, unlike the other primitives, ABME always enjoys an *efficient* "opening" algorithm that can open a ciphertext on a "lossy" tag to any message with consistent randomness. Hofheinz has proposed two instantiations. One is related to the DCR assumption and the other is based on pairing groups of a composite order. In the DCR-based ABM-LTF, lossy tags are an analogue of Waters signatures defined in DJ PKE. Such tags are carefully embedded in a matrix so that it can be non-invertible if tags are lossy; otherwise invertible. We were inspired by the lossy tag idea and have generalized it as PPRF. In the latest e-print version [18], Hofheinz has proven that his DCR-based ABM-LTF can be converted to a SIM-SO-CCA PKE scheme. To realize this, an opening algorithm for ABM-LTF is needed, and he converted his DCR-based ABM-LTF into one with an opening algorithm, by sacrificing efficiency. We note that ABM-LTF with an opening algorithm meets the notion of ABME. We will show in Sect. 8 how Hofheinz's DCR-based ABM-LTF is converted to an ABME scheme. Its expansion factor is $\Omega(1)$. However, compared to our DCR-based ABME scheme in Sect. 7, Hofheinz's ABM-LTF based ABME scheme is rather inefficient for practical use. Indeed, its expansion rate of ciphertext length per message length is ≥ 31. In addition, you must use a modulus of $\geq n^6$. On the other hand, our DCR-based ABME scheme has a small expansion rate of $(5 + 1/d)$ and you can use modulus of n^{d+1} for any $d \geq 1$. We compare them in Sect. 8. We remark that Hofheinz has not shown that his DCR-based ABM-LTF can be converted to a UC commitment scheme.

2 Preliminaries

We write PPT and DPT algorithms to denote probabilistic polynomial-time and deterministic poly-time algorithms, respectively. For random variables, X_κ and Y_κ, ranging over $\{0,1\}^\kappa$, the (statistical) distance between X_κ and Y_κ is defined as $\mathsf{Dist}(X_\kappa, Y_\kappa) \triangleq \frac{1}{2} \cdot |\Pr_{s\in\{0,1\}^\kappa}[X = s] - \Pr_{s\in\{0,1\}^\kappa}[Y = s]|$. We say that two probability ensembles, $X = \{X_\kappa\}_{\kappa\in\mathbb{N}}$ and $Y = \{Y_\kappa\}_{\kappa\in\mathbb{N}}$, are statistically indistinguishable (in κ), denoted $X \overset{\mathrm{s}}{\approx} Y$, if $\mathsf{Dist}(X_\kappa, Y_\kappa) = \mathsf{negl}(\kappa)$. We say that X and Y are computationally indistinguishable (in κ), denoted $X \overset{\mathrm{c}}{\approx} Y$, if for every PPT D (with one-bit output), $\{D(1^\kappa, X_\kappa)\}_{\kappa\in\mathbb{N}} \overset{\mathrm{s}}{\approx} \{D(1^\kappa, Y_\kappa)\}_{\kappa\in\mathbb{N}}$.

3 Building Blocks: Definitions

We now formally define probabilistic pseudo random functions and extractable sigma protocols.

3.1 Probabilistic Pseudo Random Function (PPRF)

$\mathsf{PPRF} = (\mathsf{Gen}^{\mathsf{spl}}, \mathsf{Spl})$ consists of the following two algorithms:

- $\mathsf{Gen}^{\mathsf{spl}}$, the key generation algorithm, is a PPT algorithm that takes 1^κ as input, creates pk and picks up $w \leftarrow \mathsf{KSP}^{\mathsf{spl}}_{pk}$ to outputs (pk, w), where pk uniquely determines $\mathsf{KSP}^{\mathsf{spl}}_{pk}$.
- Spl, the sampling algorithm, is a PPT algorithm that takes (pk, w) and $t \in \{0,1\}^\kappa$, picks up inner random coins $v \leftarrow \mathsf{COIN}^{\mathsf{spl}}$, and outputs u.

Here we require that pk determines set U_{pk}. Let us define $U'_{pk} = \{0,1\}^\kappa \times U_{pk}$, $L_{pk}(t) = \{u \in U_{pk} \,|\, \exists\, w, \exists\, v : u = \mathsf{Spl}(pk, w, t; v)\}$, and $L_{pk} = \{(t, u) \,|\, t \in \{0,1\}^\kappa \text{ and } u \in L_{pk}(t)\}$. We are only interested in the case that L_{pk} is relatively small in U'_{pk}, in order to avoid sampling from U'_{pk} by chance. We require that PPRFs satisfy the following security requirements:

Efficiently samplable and explainable domain: For every pk given by $\mathsf{Gen}^{\mathsf{spl}}$, set U is efficiently samplable and explainable [12], that is, there is an efficient sampling algorithm on U that takes pk and random coins R and outputs u uniformly from U_{pk}. In addition, for every $u \in U_{pk}$, there is an efficient explaining algorithm that takes pk and u and outputs random coins R behind u, where R is uniformly distributed subject to $sample(U_{pk}; R) = u$.

Pseudo randomness: Any adversary A, given pk generated by $\mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$, cannot distinguish whether it has had access to $\mathsf{Spl}(pk, w, \cdot)$ or $U(\cdot)$. Here U is the following oracle: If $\mathsf{Spl}(pk, w, \cdot)$ is a deterministic algorithm, $U : \{0,1\}^\kappa \to U_{pk}$ is a random oracle. (Namely, it returns the same (random) value on the same input.) If $\mathsf{Spl}(pk, w, \cdot)$ is probabilistic, then $U(\cdot)$ picks up a fresh randomness $u \xleftarrow{\mathsf{U}} U_{pk}$ for each query t. We say that PPRF is *pseudo random* if, for all non-uniform PPT A, $\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{PPRF},A}(\kappa) = \left|\Pr[\mathsf{Expt}^{\mathsf{prf}}_{\mathsf{PPRF},A}(\kappa) = 1] - \Pr[\mathsf{Expt}^{\mathsf{prf}}_{U,A}(\kappa) = 1]\right|$ is negligible in κ, where $\mathsf{Expt}^{\mathsf{prf}}_{\mathsf{PPRF},A}(\kappa)$ and $\mathsf{Expt}^{\mathsf{prf}}_{U,A}(\kappa)$ are defined in Fig. 1.

$\mathsf{Expt}^{\mathsf{prf}}_{\mathsf{PPRF},A}(\kappa)$:	$\mathsf{Expt}^{\mathsf{prf}}_{U,A}(\kappa)$:
$(pk, w) \leftarrow \mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$	$(pk, w) \leftarrow \mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$
$b \leftarrow A^{\mathsf{Spl}(pk,w,\cdot)}(pk)$	$b \leftarrow A^{U(\cdot)}(pk)$
return b.	return b.

Fig. 1. The experiments, $\mathsf{Expt}^{\mathsf{prf}}_{\mathsf{PPRF},A}(\kappa)$ and $\mathsf{Expt}^{\mathsf{prf}}_{U,A}(\kappa)$

Unforgeability (on $\widehat{L}_{pk}$): Let $\widehat{L}_{pk}(t)$ be some super set of $L_{pk}(t)$. Let $\widehat{L}_{pk} = \{(t,u) \mid t \in \{0,1\}^\kappa \text{ and } u \in \widehat{L}_{pk}(t)\}$. We define the game of unforgeability on $\widehat{L}_{pk}$ as follows: An adversary A takes pk generated by $\mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$ and may have access to $\mathsf{Spl}(pk, w, \cdot)$. The aim of the adversary is to output $(t^*, u^*) \in \widehat{L}_{pk}$ such that t^* has not been queried. We say that PPRF is *unforgeable* on $\widehat{L}_{pk}$ if, for all non-uniform PPT A, $\mathsf{Adv}^{\mathsf{euf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}(\kappa) = \Pr[\mathsf{Expt}^{\mathsf{euf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}(\kappa) = 1]$ (where $\mathsf{Expt}^{\mathsf{euf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}$ is defined in Fig. 2) is negligible in κ.

In some application, we require a stronger requirement, where in the same experiment above, it is difficult for the adversary to output (t^*, u^*) in $\widehat{L}_{pk}$, which did not appear in the query/answer list $\mathcal{QA}$. We say that PPRF is *strongly unforgeable* on $\widehat{L}_{pk}$ if, for all non-uniform PPT A, $\mathsf{Adv}^{\mathsf{seuf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}(\kappa) = \Pr[\mathsf{Expt}^{\mathsf{seuf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}(\kappa) = 1]$ (where $\mathsf{Expt}^{\mathsf{seuf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}$ is defined in Fig. 2) is negligible in κ.

We remark that (strong) unforgeability implies (1) that $\widehat{L}_{pk}$ should be small enough in U'_{pk} to avoid sampling from $\widehat{L}_{pk}$ by chance, and (2) that, if Spl is a DPT algorithm and $\widehat{L}_{pk} = L_{pk}$, it is implied by pseudo randomness.

$\mathsf{Expt}^{\mathsf{euf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}(\kappa)$:
$(pk, w) \leftarrow \mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$
$(t^*, u^*) \leftarrow A^{\mathsf{Spl}(pk,w,\cdot)}(pk)$
If t^* has not been queried and $u^* \in \widehat{L}_{pk}(t^*)$, return 1; otherwise 0.

$\mathsf{Expt}^{\mathsf{seuf\text{-}}\widehat{L}}_{\mathsf{PPRF},A}(\kappa)$:
$(pk, w) \leftarrow \mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$
$(t^*, u^*) \leftarrow A^{\mathsf{Spl}(pk,w,\cdot)}(pk)$
$(t^*, u^*) \notin \mathcal{QA}$ and $u^* \in \widehat{L}_{pk}(t^*)$, return 1; otherwise 0.

Fig. 2. The experiments of unforgeability (in the left) and strong unforgeability (in the right).

3.2 Extractable Sigma Protocol

An extractable sigma protocol, $\mathsf{ext}\Sigma = (\mathsf{Gen}^{\mathsf{ext}}, \mathsf{com}\Sigma, \mathsf{ch}\Sigma, \mathsf{ans}\Sigma, \mathsf{sim}\Sigma, \mathsf{Vrfy}, \mathsf{Dec})$ is a sigma protocol, associated with two algorithms, $\mathsf{Gen}^{\mathsf{ext}}$ and Dec, with the following properties.

- $\mathsf{Gen}^{\mathsf{ext}}$ is an PPT algorithm that takes 1^κ and outputs (pk, sk), such that pk defines the entire set U'_{pk}, and two sub disjoint sets, L_{pk} and L^{ext}_{pk}, i.e., $L_{pk} \cup L^{\mathsf{ext}}_{pk} \subset U'_{pk}$ and $L_{pk} \cap L^{\mathsf{ext}}_{pk} = \emptyset$. We also require that L_{pk} determines binary efficiently recognizable set R_{pk} such that $L_{pk} = \{x | \exists w : (x, w) \in R_{pk}\}$.
- $\mathsf{com}\Sigma$ is a PPT algorithm that takes pk and $(x, w) \in R_{pk}$, picks up inner coins r_a, and outputs a.
- $\mathsf{ch}\Sigma(pk)$ is a publicly-samplable set determined by pk.

– $\mathsf{ans}\Sigma$ is a DPT algorithm that takes (pk, x, r_a, e), where $e \in \mathsf{ch}\Sigma(pk)$, and outputs z.
– Vrfy is a DPT algorithm that accepts or rejects (pk, x, a, e, z).
– $\mathsf{sim}\Sigma$ is a PPT algorithm that takes (pk, x, e) and outputs $(a, e, z) = \mathsf{sim}\Sigma(pk, x, e; r_z)$, where $r_z \leftarrow \mathsf{COIN}^{\mathsf{sim}}$. *We additionally require that* $r_z = z$. Namely, $(a, e, r_z) = \mathsf{sim}\Sigma(pk, x, e; r_z)$.
– Dec is a DPT algorithm that takes (sk, x, a) and outputs e or $\perp$.

We require that $\mathsf{ext}\Sigma$ satisfies the following properties:

Completeness: For every $(pk, sk) \in \mathsf{Gen}^{\mathsf{ext}}(1^\kappa)$, every $(x, w) \in R_{pk}$, every r_a (in an appropriate specified domain) and every $e \in \mathsf{ch}\Sigma(pk)$, it always holds that $\mathsf{Vrfy}(x, \mathsf{com}\Sigma(x, w; r_a), e, \mathsf{ans}\Sigma(x, w, r_a, e)) = 1$.

Special Soundness: For every $(pk, sk) \in \mathsf{Gen}^{\mathsf{ext}}(1^\kappa)$, every $x \in U'_{pk} \backslash L_{pk}$ and every a, there is *unique* $e \in \mathsf{ch}\Sigma(pk)$ if there is an accepting conversation for a on x. We say that a pair of two different accepting conversations for the same a on x, i.e., (a, e, z) and (a, e', z'), with $e \neq e'$, is a *collision* on x.

Enhanced Honest-Verifier Statistical Zero-Knowledgeness (eHVSZK): For every $(pk, sk) \in \mathsf{Gen}^{\mathsf{ext}}(1^\kappa)$, every $(x, w) \in R_{pk}$, and every $e \in \mathsf{ch}\Sigma(pk)$, the following ensembles are statistically indistinguishable in κ:

$$\{(\mathsf{com}\Sigma(pk, x, w; r_a), e, \mathsf{ans}\Sigma(pk, x, w, r_a, e))\}_{\substack{(pk,\, sk) \in \mathsf{Gen}^{\mathsf{ext}}(1^\kappa),\ (x, w) \in R_{pk},\\ e \in \mathsf{ch}\Sigma(pk),\ \kappa \in \mathbb{N}.}}$$

$$\overset{\mathrm{s}}{\approx}\{\mathsf{sim}\Sigma(pk, x, e; r_z)\}_{(pk,sk)\in\mathsf{Gen}^{\mathsf{ext}}(1^\kappa),(x,w)\in R_{pk},e\in\mathsf{ch}\Sigma(pk),\kappa\in\mathbb{N}}$$

Here the probability of the left-hand side is taken over random variable r_z and the right-hand side is taken over random variable r_a. We remark that since $(a, e, r_z) = \mathsf{sim}\Sigma(pk, x, e; r_z)$, we have $\mathsf{Vrfy}(pk, x, a, e, z) = 1$ if and only if $(a, e, z) = \mathsf{sim}\Sigma(pk, x, e; z)$. *Therefore, one can instead use* $\mathsf{sim}\Sigma$ *to verify* (a, e, z) on x.

Extractability: For every $(pk, sk) \in \mathsf{Gen}^{\mathsf{ext}}(1^\kappa)$, every $x \in L^{\mathsf{ext}}_{pk}$, and every a such that there is an accepting conversation for a on x, Dec always outputs $e = \mathsf{Dec}(sk, x, a)$ such that (a, e, z) is an accepting conversation on x. We note that, when $x \notin L_{pk}$, e is unique given a, due to the special soundness property. Therefore, the extractability is well defined because $L_{pk} \cap L^{\mathsf{ext}}_{pk} = \emptyset$.

4 ABM Encryption

All-but-many encryption scheme $\mathsf{ABM.Enc}$ = ($\mathsf{ABM.gen}$, $\mathsf{ABM.spl}$, $\mathsf{ABM.enc}$, $\mathsf{ABM.dec}$, $\mathsf{ABM.col}$) consists of the following algorithms:

– $\mathsf{ABM.gen}$ is a PPT algorithm that takes 1^κ and outputs $(pk, (sk, w))$, where pk defines a set U_{pk}. We let $U'_{pk} = \{0,1\}^\kappa \times U_{pk}$. pk also determines two disjoint sets, L^{td}_{pk} and L^{ext}_{pk}, such that $L^{\mathsf{td}}_{pk} \cup L^{\mathsf{ext}}_{pk} \subset U'_{pk}$.
– $\mathsf{ABM.spl}$ is a PPT algorithm that takes (pk, w, t), where $t \in \{0,1\}^\kappa$, picks up inner random coins $v \leftarrow \mathsf{COIN}^{\mathsf{spl}}$, and computes $u \in U_{pk}$. We write $L^{\mathsf{td}}_{pk}(t)$ to denote the image of $\mathsf{ABM.spl}$ on t under pk, i.e.,

$$L^{\mathsf{td}}_{pk}(t) := \{u \in U_{pk} \mid \exists\, w, \exists\, v :\ u = \mathsf{ABM.spl}(pk, w, t; v)\}.$$

We require $L^{\mathsf{td}}_{pk} = \{(t,u) \,|\, t \in \{0,1\}^\kappa \text{ and } u \in L^{\mathsf{td}}_{pk}(t)\}$. We set $\widehat{L}^{\mathsf{td}}_{pk} := U'_{pk} \backslash L^{\mathsf{ext}}_{pk}$. Since $L^{\mathsf{td}}_{pk} \cap L^{\mathsf{ext}}_{pk} = \emptyset$, we have $L^{\mathsf{td}}_{pk} \subseteq \widehat{L}^{\mathsf{td}}_{pk} \subset U'_{pk}$.

- $\mathsf{ABM.enc}$ is a PPT algorithm that takes pk, $(t,u) \in U'_{pk}$, and message $x \in \mathsf{MSP}$, picks up inner random coins $r \leftarrow \mathsf{COIN}^{\mathsf{enc}}$, and computes $c = \mathsf{ABM.enc}^{(t,u)}(pk, x; r)$, where MSP denotes the message space uniquely determined by pk, whereas $\mathsf{COIN}^{\mathsf{enc}}$ denotes the inner coin space uniquely determined by pk *and* x [2].
- $\mathsf{ABM.dec}$ is a DPT algorithm that takes sk, (t,u), and ciphertext c, and outputs $x = \mathsf{ABM.dec}^{(t,u)}(sk, c)$.
- $\mathsf{ABM.col} = (\mathsf{ABM.col}_1, \mathsf{ABM.col}_2)$ is a pair of PPT and DPT algorithms, respectively, such that
  - $\mathsf{ABM.col}_1$ takes $(pk, (t,u), w, v)$ and outputs $(c, \xi) \leftarrow \mathsf{ABM.col}_1^{(t,u)}(pk, w, v)$, where $v \in \mathsf{COIN}^{\mathsf{spl}}$.
  - $\mathsf{ABM.col}_2$ takes $((t,u), \xi, x)$, with $x \in \mathsf{MSP}$, and outputs $r \in \mathsf{COIN}^{\mathsf{enc}}$.

We require that all-but-many encryption schemes satisfy the following properties:

1. **Adaptive All-but-many property:** $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ is a probabilistic pseudo random function (PPRF) as defined in Sect. 3.1 with unforgeability on $\widehat{L}^{\mathsf{td}}_{pk} (= U'_{pk} \backslash L^{\mathsf{ext}}_{pk})$.
2. **Dual mode property:**
   - **(Decryption mode)** For every $\kappa \in \mathbb{N}$, every $(pk, (sk, w)) \in \mathsf{ABM.gen}(1^\kappa)$, every $(t,u) \in L^{\mathsf{ext}}_{pk}$, and every $x \in \mathsf{MSP}$, it always holds that
   $$\mathsf{ABM.dec}^{(t,u)}(sk, \mathsf{ABM.enc}^{(t,u)}(pk, x)) = x.$$
   - **(Trapdoor mode)** Define the following random variables: $\mathsf{dist}^{\mathsf{enc}}(t, pk, sk, w, x)$ denotes random variable (u, c, r) defined as follows: $v \leftarrow \mathsf{COIN}^{\mathsf{spl}}$; $u = \mathsf{ABM.spl}(pk, w, t; v)$; $r \leftarrow \mathsf{COIN}^{\mathsf{enc}}$; $c = \mathsf{ABM.enc}^{(t,u)}(pk, x; r)$. $\mathsf{dist}^{\mathsf{col}}(t, pk, sk, w, x)$ denotes random variable (u, c, r) defined as follows: $v \leftarrow \mathsf{COIN}^{\mathsf{spl}}$; $u = \mathsf{ABM.spl}(pk, w, t; v)$; $(c, \xi) = \mathsf{ABM.col}_1^{(t,u)}(pk, w, v)$; $r = \mathsf{ABM.col}_2^{(t,u)}(\xi, x)$. Then, the following ensembles are statistically indistinguishable in κ:
   $$\left\{\mathsf{dist}^{\mathsf{enc}}(t, pk, sk, w, x)\right\}_{(pk,(sk,w))\in\mathsf{ABM.gen}(1^\kappa), t\in\{0,1\}^\kappa, x\in\mathsf{MSP}, \kappa\in\mathbb{N}}$$
   $$\stackrel{\mathrm{s}}{\approx} \left\{\mathsf{dist}^{\mathsf{col}}(t, pk, sk, w, x)\right\}_{(pk,(sk,w))\in\mathsf{ABM.gen}(1^\kappa), t\in\{0,1\}^\kappa, x\in\mathsf{MSP}, \kappa\in\mathbb{N}}$$

We say that a ciphertext c on (t,u) under pk is *valid* if there exist $x \in \mathsf{MSP}$ and $r \in \mathsf{COIN}^{\mathsf{enc}}$ such that $c = \mathsf{ABM.enc}^{(t,u)}(pk, x; r)$. We say that a valid

[2] We allow the inner coin space to depend on messages to be encrypted, in order to be consistent with our weak ABM encryption scheme from general assumption appeared in the full version [14]

ciphertext c on (t,u) under pk is *real* if $(t,u) \in L_{pk}^{\mathsf{ext}}$, otherwise *fake*. We remark that as long as c is a real ciphertext, regardless of how it is generated, there is only one consistent x in MSP and it is equivalent to $\mathsf{ABM.dec}^{(t,u)}(sk,c)$.

5 ABME from $\mathsf{ext}\Sigma$ on Language derived from PPRF

Let $\mathsf{PPRF} = (\mathsf{Gen}^{\mathsf{spl}}, \mathsf{Spl})$ be a PPRF and let $\mathsf{ext}\Sigma = (\mathsf{Gen}^{\mathsf{ext}}, \Sigma, \mathsf{Dec})$ be an extractable sigma protocol.

Assume the following conditions hold.

- The first output of $\mathsf{Gen}^{\mathsf{ext}}(1^\kappa)$ is distributed identically to the first output of $\mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$.
- For every L_{pk} generated by $\mathsf{Gen}^{\mathsf{ext}}$, L_{pk} is the language derived from PPRF; namely, $L_{pk} = \{(t,u) \mid \exists (w,v) : t \in \{0,1\}^\kappa, u = \mathsf{Spl}(pk,w,t;v)\}$.
- For $(L_{pk}, L_{pk}^{\mathsf{ext}}, U'_{pk})$ generated by $\mathsf{Gen}^{\mathsf{ext}}$, PPRF is unforgeable on $\widehat{L}_{pk}$, where $\widehat{L}_{pk} := U'_{pk} \backslash L_{pk}^{\mathsf{ext}}$.

Then, we can construct an ABME scheme as described in Fig. 3.

- $\mathsf{ABM.gen}(1^\kappa)$ runs $\mathsf{Gen}^{\mathsf{ext}}(1^\kappa)$ to output (pk,sk). It chooses $w \leftarrow \mathsf{KSP}_{pk}^{\mathsf{spl}}$ and finally outputs $(pk,(sk,w))$. We note that by a precondition the distribution of pk from $\mathsf{Gen}^{\mathsf{ext}}(1^\kappa)$ is identical to that of $\mathsf{Gen}^{\mathsf{spl}}(1^\kappa)$.
- $\mathsf{ABM.spl}(pk,w,t;v)$ outputs $u := \mathsf{Spl}(pk,w,t;v)$ where $v \xleftarrow{\cup} \mathsf{COIN}^{\mathsf{spl}}$.
- $\mathsf{ABM.enc}^{(t,u)}(pk,m;r)$ runs $(a,m,r) \leftarrow \mathsf{sim}\Sigma(pk,(t,u),m;r)$ to return the first output a, where $r \xleftarrow{\cup} \mathsf{COIN}_{pk}^{\mathsf{enc}}$ $(:= \mathsf{COIN}_{pk}^{\mathsf{sim}})$.
- $\mathsf{ABM.dec}^{(t,u)}(sk,c)$ outputs $m = \mathsf{Dec}(sk,(t,u),c)$.
- $\mathsf{ABM.col}_1^{(t,u)}(pk,w,v;r_a)$ outputs (c,ξ) such that $c := \mathsf{com}\Sigma(pk,(t,u),(w,v);r_a)$, and $\xi := (pk,t,u,w,v,r_a)$.
- $\mathsf{ABM.col}_2^{(t,u)}(\xi,m)$ outputs $r := \mathsf{ans}\Sigma(pk,(t,u),w,v,r_a,m)$, where $\xi = (pk,t,u,w,v,r_a)$.

Fig. 3. ABME from $\mathsf{ext}\Sigma$ on language derived from PPRF

By construction, the adaptive all-but-many property holds in the resulting scheme. The dual mode property also holds because: (a) If $(t,u) \in L_{pk}^{\mathsf{ext}}$, the first output of $\mathsf{sim}\Sigma(pk,(t,u),m)$ is perfectly binding to challenge m due to special soundness (because $L_{pk}^{\mathsf{ext}} \subset U'_{pk} \backslash L_{pk}^{\mathsf{td}}$, with $L_{pk}^{\mathsf{td}} := L_{pk}$), and m can be extracted given $(pk,(t,u),a)$ using sk due to extractability. (b) If $(t,u) \in L_{pk}^{\mathsf{td}}$, $\mathsf{ABM.col}$ runs the real sigma protocol with witness (w,v). Therefore, it can produce a fake commitment that can be opened in any way, while it is statistically indistinguishable from that of the simulation algorithm $\mathsf{sim}\Sigma$ (that is run by $\mathsf{ABM.enc}$), due to enhanced HVSZK. Therefore, the resulting scheme is ABME.

6 Fully-Equipped UC Commitment from ABME

We show that ABME implies fully-equipped UC commitment.

We work in the standard universal composability (UC) framework of Canetti [4]. We concentrate on the same model in [5] where the network is asynchronous, the communication is public but ideally authenticated, and the adversary is adaptive in corrupting parties and is active in its control over corrupted parties. Any number of parties can be corrupted and parties cannot erase any of their inner state. We consider UC commitment schemes that can be used repeatedly under a single common reference string. The multi-commitment ideal functionality $\mathcal{F}_{\mathsf{MCOM}}$ from [6] is the ideal functionality of such commitments, which is given in Figure 4.

A fully-equipped UC commitment scheme is constructed as follows: A trusted party chooses and puts pk of ABME in the common reference string. In the commit phase, committer P_i takes tag $t = (\mathtt{sid}, \mathtt{ssid}, P_i, P_j)$ and message x committed to. It then picks up random u from U_{pk} and compute an ABM encryption $c = \mathsf{ABM.enc}^{(t,u)}(pk, x; r)$ to send (t, u, c) to receiver P_j, which outputs $(\mathtt{receipt}, \mathtt{sid}, \mathtt{ssid}, P_i, P_j)$. In the reveal phase, P_i sends (x, r) to P_j and P_j accepts if and only if $c = \mathsf{ABM.enc}^{(t,u)}(pk, x; r)$. If P_j accepts, he outputs x, otherwise do nothing. The formal description is given in the full version [14].

Theorem 1. *The proposed commitment scheme from ABME UC-securely realizes the $\mathcal{F}_{\mathsf{MCOM}}$ functionality in the $\mathcal{F}_{\mathsf{CRS}}$-hybrid model in the presence of adaptive adversaries in the non-erasure model.*

Proof (Sketch). The formal proof is given in the full version [14]. We here sketch the essence. We consider the man-in-the-middle attack, where we show that the view of environment $\mathcal{Z}$ in the real world (in the CRS model) can be simulated in the ideal world. Let P_i, P_j be honest players and let $P_{i'}$ be a corrupted player controlled by adversary $\mathcal{A}$. In the man-in-the-middle attack, $P_{i'}$ (i.e., $\mathcal{A}$) is simultaneously participating in the left and right interactions. In the left interactions, $\mathcal{A}$ interacts with P_i, as playing the role of the receiver. In the right interactions, $\mathcal{A}$ interacts with P_j, as playing the role of the committer.

The following sketch corresponds to security proof in the (static) man-in-the-middle attack. It is not difficult to handle the adaptive case if this case has been proven secure.

In the ideal world, $\mathcal{A}$ actually interacts with simulator $\mathcal{S}$ in both interactions, where $\mathcal{S}$ pretends to be P_i and P_j respectively. In the left interactions, environment $\mathcal{Z}$ sends $(\mathtt{commit}, \mathtt{sid}, \mathtt{ssid}, P_i, P_{i'}, x)$ to the ideal commitment functionality $\mathcal{F}_{\mathsf{MCOM}}$ (via honest P_i). After receiving $(\mathtt{receipt}, \mathtt{sid}, \mathtt{ssid}, P_i, P_{i'})$ from $\mathcal{F}_{\mathsf{MCOM}}$, $\mathcal{S}$ starts the commitment protocol as the committer without given message x. It sends to $\mathcal{A}$ (u, c) on $t = (\mathtt{sid}, \mathtt{ssid}, P_i, P_{i'})$ as computed in Table 1. In the decommitment phase when $\mathcal{Z}$ sends $(\mathtt{open}, \mathtt{sid}, \mathtt{ssid})$ to $\mathcal{F}_{\mathsf{MCOM}}$ (via honest P_i), $\mathcal{S}$ receives x from $\mathcal{F}_{\mathsf{MCOM}}$ and then computes $r = \mathsf{ABM.col}_2^{(t,u)}(\xi, x)$ to send (t, x, r) to $\mathcal{A}$. In the right interactions, $\mathcal{S}$ receives (t', u', c') from $\mathcal{A}$ where $t' = (\mathtt{sid}', \mathtt{ssid}', P_{i'}, P_j)$. It then extracts $\tilde{x} = \mathsf{ABM.dec}^{(t',u')}(sk, c')$ to send to

$\mathcal{F}_{\mathsf{MCOM}}$. $\mathcal{F}_{\mathsf{MCOM}}$ then sends $(\mathtt{receipt}, \mathtt{sid}, \mathtt{ssid}, P_{i'}, P_j)$ to environment $\mathcal{Z}$ (via honest P_j). In the decommitment phase when $\mathcal{A}$ opens (t', u', c') correctly with (x', r'), $\mathcal{S}$ sends $(\mathtt{open}, \mathtt{sid}, \mathtt{ssid})$ to $\mathcal{F}_{\mathsf{MCOM}}$; otherwise, do nothing. Upon receiving $(\mathtt{open}, \mathtt{sid}, \mathtt{ssid})$, if the same $(\mathtt{sid}, \mathtt{ssid}, ..)$ was previously recorded, $\mathcal{F}_{\mathsf{MCOM}}$ sends **stored** $\tilde{x}$ to environment $\mathcal{Z}$ (via honest P_j); otherwise, do nothing. We note that in the ideal world, honest parties convey inputs from $\mathcal{Z}$ to the ideal functionalities and vice versa. The view of $\mathcal{Z}$ consists of the view of $\mathcal{A}$ plus the value sent by $\mathcal{F}_{\mathsf{MCOM}}$.

In $\mathsf{Hybrid}^{\mathcal{F}_{\mathsf{crs}}}$ (the real world in the CRS model), $\mathcal{A}$ interacts with real (committer) P_i in the left interactions, and real (receiver) P_j in the righ interactions. In the right interactions, at the end of the decommitment phase, P_j sends x' to $\mathcal{Z}$ if $\mathcal{A}$ has opened (t', u', c') correctly with (x', r'). The view of $\mathcal{Z}$ consists of the view of $\mathcal{A}$ plus the value sent by P_j.

The goal is to prove that the two views of $\mathcal{Z}$ above are computationally indistinguishable. As usual, we consider a sequence of hybrid games on which the probability spaces are identical, but we change the rules of games step by step. See Table 1 for summary.

Hybrid Game 1 is identical to the ideal world except that in the left interactions, at the beginning of the commitment phase, $\mathcal{S}$ (as P_i) is given message x on tag $t = (\mathtt{sid}, \mathtt{ssid}, P_i, P_{i'})$ by $\mathcal{F}_{\mathsf{MCOM}}$. $\mathcal{S}$ computes $u \leftarrow \mathsf{ABM.spl}(pk, w, t)$, and $c = \mathsf{ABM.enc}^{(t,u)}(pk, x; r)$, picking up random r, to send (t, u, c) to adversary $\mathcal{A}$. In the decommitment phase, $\mathcal{S}$ sends (t, x, r) to $\mathcal{A}$.

Hybrid Game 2 is identical to Hybrid Game 1 except that in the right interactions, after receiving (t', u', c'), $\mathcal{S}_2$ sends ϵ to $\mathcal{F}_{\mathsf{MCOM}}$. In the decommitment phase when $\mathcal{A}$ opens (t', u', c') correctly with (x, r), $\mathcal{S}$ sends $(\mathtt{open}, \mathtt{sid}, \mathtt{ssid}, x')$ to $\mathcal{F}_{\mathsf{MCOM}}$. $\mathcal{F}_{\mathsf{MCOM}}$ sends x' to environment $\mathcal{Z}$ (via ideal $\tilde{P}_j$), instead of sending ϵ.

Hybrid Game 3 is identical to Hybrid Game 2 except that in the left interactions, $\mathcal{S}$ instead picks up random $u \leftarrow U_{pk}$ and computes $c = \mathsf{ABM.enc}^{(t,u)}(pk, x; r)$, to send (t, u, c) to $\mathcal{A}$.

[$\mathsf{Ideal} \Rightarrow \mathsf{Hybrid}^1$] The two views of $\mathcal{Z}$ between the ideal world and Hybrid^1 are statistically close, due to the trapdoor mode property.

[$\mathsf{Hybrid}^1 \Rightarrow \mathsf{Hybrid}^2$] We note that the distance of the two views of $\mathcal{Z}$ between Hybrid^1 and Hybrid^2 is bounded by the following event. Let BD_I denote the event in Hybrid Game I ($I \in \{1, 2\}$) that $\mathcal{S}$ receives a fake ciphertext (t', u', c') from $\mathcal{A}$, i.e., $(t', u') \in L^{\mathsf{td}}_{pk}$, in the right intersections. If this event does not occur, the view of $\mathcal{Z}$ in both games are identical, which means $\neg\mathsf{BD}_1 = \neg\mathsf{BD}_2$. Hence, the distance of the views of $\mathcal{Z}$ in the two games is bounded by $\Pr[\mathsf{BD}]$, where $\mathsf{BD} := \mathsf{BD}_1 = \mathsf{BD}_2$. We then evaluate $\Pr[\mathsf{BD}]$ in Hybrid Game 2. (We note that we might not generally evaluate the probability in Hybrid Game 1, because $\mathcal{S}$ must decrypt (t', u', c'), which seems that it needs sk, but knowing sk implies some information on w.) We want to suppress $\Pr[\mathsf{BD}]$ by using the assumption that ($\mathsf{ABM.gen}$, $\mathsf{ABM.spl}$) is unforgeable on $\widehat{L}^{\mathsf{td}}_{pk}$. In Hybrid Game 2, we can construct an adversary B that breaks unforgeability of ($\mathsf{ABM.gen}, \mathsf{ABM.spl}$) on $\widehat{L}^{\mathsf{td}}_{pk}$ as follows. In the left and right interactions, B simulates the role of $\mathcal{S}$ and interacts with $\mathcal{A}$. B uses $\mathsf{ABM.spl}(pk, w, \cdot)$ as oracle to play the role of $\mathcal{S}$ in the left

interaction. After $\mathcal{A}$ halts, B outputs (t', u') at random from the communication with $\mathcal{A}$ in the right interactions. We note that, since the communication channel is fully authenticated, it holds that $t' \neq t$ for all t, t', because $t = (\star, \star, P_i, P_{i'})$ and $t' = (\star, \star, P_{i'}, P_j)$. If $(t', u') \in \widehat{L}^{\mathsf{td}}_{pk}$, B succeeds in breaking unforgeability on $\widehat{L}^{\mathsf{td}}_{pk}$, which is upper-bounded by some negligible function. Since event BD occurs at most with the success probability of B. Hence, its probability is negligible, too.

$[\mathsf{Hybrid}^2 \Rightarrow \mathsf{Hybrid}^3]$ It is obvious by construction that the distance of the two views of $\mathcal{Z}$ between Hybrid^2 and Hybrid^3 is bounded by the advantage of pseudo-randomness of $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$.

$[\mathsf{Hybrid}^3 \Rightarrow \mathsf{Hybrid}^{\mathcal{F}_{\mathsf{MCOM}}}]$ By construction, the two views of $\mathcal{Z}$ between Hybrid^3 and $\mathsf{Hybrid}^{\mathcal{F}_{\mathsf{MCOM}}}$ are identical.

Therefore, the two views of $\mathcal{Z}$ between the ideal world and $\mathsf{Hybrid}^{\mathcal{F}_{\mathsf{MCOM}}}$ are computationally close. ■

Games	$P_i(\mathcal{S}) \xrightarrow{(t,u,c)}$	Corr. $P_{i'}(\mathcal{A}) \xrightarrow{(t',u',c')}$	$P_j(\mathcal{S}) \xrightarrow{(t',\tilde{x})}$	$\mathcal{F}_{\mathsf{MCOM}}$
Ideal	$u = \mathsf{ABM.spl}(pk, w, t; v)$ $(c, \xi) = \mathsf{ABM.col}_1^{(t,u)}(pk, w, v)$ $\mathtt{open}\colon x,\ r = \mathsf{ABM.col}_2^{(t,u)}(\xi, x)$	(t', u', c') $\mathtt{open}\colon (x', r')$	$\tilde{x} =$ $\mathsf{ABM.dec}^{(t',u')}(sk, c')$	$\tilde{x}$
Hybrid^1	$u \leftarrow \mathsf{ABM.spl}(pk, w, t)$ $c = \mathsf{ABM.enc}^{(t,u)}(pk, x, r)$ $\mathtt{open}\colon x,\ r$	(t', u', c') $\mathtt{open}\colon (x', r')$	$\tilde{x} =$ $\mathsf{ABM.dec}^{(t',u')}(sk, c')$	$\tilde{x}$
Hybrid^2	$u \leftarrow \mathsf{ABM.spl}(pk, w, t)$ $c = \mathsf{ABM.enc}^{(t,u)}(pk, x, r)$ $\mathtt{open}\colon x,\ r$	(t', u', c') $\mathtt{open}\colon (x', r')$	$\tilde{x} = \epsilon$	x'
Hybrid^3	$u \leftarrow U_{pk}$ $c = \mathsf{ABM.enc}^{(t,u)}(pk, x, r)$ $\mathtt{open}\colon x,\ r$	(t', u', c') $\mathtt{open}\colon (x', r')$	$\tilde{x} = \epsilon$	x'
	$P_i \xrightarrow{(t,u,c)}$	Corr. $P_{i'}(\mathcal{A}) \xrightarrow{(t',u',c')}$	P_j	P_j
$\mathsf{Hybrid}^{\mathcal{F}_{\mathsf{crs}}}$	$u \leftarrow U_{pk}$ $c = \mathsf{ABM.enc}^{(t,u)}(pk, x, r)$ $\mathtt{open}\colon x,\ r$	(t', u', c') $\mathtt{open}\colon (x', r')$		x'

Table 1. The man-in-the-midle attack in the hybrid games

Here $t = (\mathtt{sid}, \mathtt{ssid}, P_i, P_{i'})$ and $t' = (\mathtt{sid}', \mathtt{ssid}', P_{i'}, P_j)$. The view of $\mathcal{Z}$ consists of the view of $\mathcal{A}$ plus the contents in the rightest column.

7 Compact ABME from Damgård-Jurik PKE

Damgård-Jurik PKE. Let $\Pi = (\mathbf{K}, \mathbf{E}, \mathbf{D})$ be a tuple of algorithms of Damgård-Jurik (DJ) PKE [9]. A public key of DJ PKE is $pk_{\mathsf{dj}} = (n, d)$ and the corresponding secret-key is $sk_{\mathsf{dj}} = (p, q)$ where $n = pq$ is a composite number of distinct odd primes, p and q, and $1 \leq d < p, q$ is a positive integer (when $d = 1$ it is Paillier PKE [23]). We often write $\Pi^{(d)}$ to clarify parameter d. We

let $g := (1+n)$. To encrypt message $x \in \mathbb{Z}_{n^d}$, one computes $\mathbf{E}_{pk_{\mathsf{dj}}}(x;R) = g^x R^{n^d} \pmod{n^{d+1}}$ where $R \leftarrow \mathbb{Z}_n^\times$ [3]. For simplicity, we write $\mathbf{E}(x)$ instead of $\mathbf{E}_{pk_{\mathsf{dj}}}(x)$, if it is clear. DJ PKE is enhanced additively homomorphic, meaning that, for every $x_1, x_2 \in \mathbb{Z}_{n^d}$ and every $R_1, R_2 \in \mathbb{Z}_n^\times$, one can efficiently compute R such that $\mathbf{E}(x_1+x_2;R) = \mathbf{E}(x_1;R_1)\cdot\mathbf{E}(x_2;R_2)$. Actually it can be done by computing $R = g^\gamma R_1 R_2 \pmod{n}$, where γ is an integer such that $x_1+x_2 = \gamma n^d + ((x_1+x_2) \bmod n^d)$. It is known that $\mathbb{Z}_{n^{d+1}}^\times$ is isomorphic to $\mathbb{Z}_{n^d} \times \mathbb{Z}_n^\times$ (the product of a cyclic group of order n^d and a group of order $\phi(n)$), and, for any $d < p, q$, element $g = (1+n)$ has order n^d in $\mathbb{Z}_{n^{d+1}}^\times$ [9]. Therefore, $\mathbb{Z}_{n^{d+1}}^\times$ is the image of $\mathbf{E}(\cdot;\cdot)$. We note that it is known that $\mathbb{Z}_{n^{d+1}}^\times$ is *efficiently samplable and explainable* [10, 12]. It is also known that DJ PKE is IND-CPA if the DCR assumption holds true [9].

Construction Idea. $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ below forms Waters-like signature scheme based on DJ PKE, where there is no verification algorithm and the signatures look pseudo random assuming that DJ PKE is IND-CPA. We then construct an extractable sigma protocol on the language derived from $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$, as discussed in Sect. 5. Here, the decryption algorithm works only when the matrix below in (3) is invertible, which is equivalent to that $(t,(u_r,u_t)) \in L_{pk}^{\mathsf{ext}}$, where $L_{pk}^{\mathsf{ext}} =$

$$\{(t,(u_r,u_t)) \mid \mathbf{D}(u_t) \not\equiv x_1x_2 + y(t)\mathbf{D}(u_r) \bmod p \wedge \mathbf{D}(u_t) \not\equiv x_1x_2 + y(t)\mathbf{D}(u_r) \bmod q\}.$$

Therefore, we require that $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ should be unforgeable on $\widehat{L}_{pk}^{\mathsf{td}} (= U'_{pk} \backslash L_{pk}^{\mathsf{ext}})$. To prove this, we additionally require two assumptions on DJ PKE, called *the non-multiplication assumption* and *the non-trivial divisor assumption*, described in Appendix C. The first one is an analogue of the DH assumption in an additively homomorphic encryption. If we consider unforgeability on L_{pk}^{td}, this assumption suffices, but we require unforgeability on $\widehat{L}_{pk}^{\mathsf{td}}$. Then we need the latter assumption, too. These two assumptions are originally introduced in [18] to obtain a DCR-based ABM-LTF.

7.1 ABME from Damgård-Jurik with Optimal Expansion Factor $\Omega(1)$

- $\mathsf{ABM.gen}(1^\kappa)$**:** It gets $(pk_{\mathsf{dj}}, sk_{\mathsf{dj}}) \leftarrow \mathbf{K}(1^\kappa)$ (the key generation algorithm for DJ PKE), where $pk_{\mathsf{dj}} = (n,d)$ and $sk_{\mathsf{dj}} = (p,q)$. It then picks up $x_1, x_2 \xleftarrow{\cup} \mathbb{Z}_{n^d}$, $R_1, R_2 \xleftarrow{\cup} \mathbb{Z}_{n^{d+1}}^\times$, and computes $g_1 = \mathbf{E}(x_1;R_1)$ and $g_2 = \mathbf{E}(x_2;R_2)$. It then picks up $\tilde{h} \leftarrow \mathbf{E}(1)$ and computes $\boldsymbol{h} = (h_0, \ldots, h_\kappa)$ such that $h_j := \tilde{h}^{y_j}$ where $y_j \xleftarrow{\cup} \mathbb{Z}_{n^{d+1}}$ for $j = 0, 1, \ldots, \kappa$. Let $H(t) = h_0 \prod_{i=1}^{\kappa} h_i^{t_i} \pmod{n^{d+1}}$ and let $y(t) = y_0 + \sum_{i=1}^{\kappa} y_i t_i \pmod{n^d}$, where $(t_0, \ldots, t_\kappa)$ represents the bit string of t. We note that $H(t) = \tilde{h}^{y(t)}$. It outputs $(pk, (sk, w))$ where

[3] In the original scheme, R is chosen from $\mathbb{Z}_{n^{d+1}}^\times$. However, since $\mathbb{Z}_n^\times$ is isomorphic to the cyclic group of order n^d in $\mathbb{Z}_{n^{d+1}}^\times$ by mapping $R \in \mathbb{Z}_n^\times$ to $R^{n^d} \in \mathbb{Z}_{n^{d+1}}^\times$, we can instead choose R from $\mathbb{Z}_n^\times$.

$pk := (n, d, g_1, g_2, \boldsymbol{h})$, $sk := (p, q)$ and $w := x_2$, where we define $U'_{pk} := \{0,1\}^\kappa \times (\mathbb{Z}^\times_{n^{d+1}})^2$ that contains the disjoint sets of L^{td}_{pk} and L^{ext}_{pk} as described below.

- $\mathsf{ABM.spl}(pk, x_2, t; (r, R_r, R_t))$**:** It chooses $r \leftarrow \mathbb{Z}_{n^d}$ and outputs $u := (u_r, u_t)$ such that $u_r := \mathbf{E}(r; R_r)$ and $u_t := g_1^{x_2}\mathbf{E}(0; R_t)\cdot H(t)^r$ where $R_r, R_t \leftarrow \mathbb{Z}^\times_{n^{d+1}}$. We let $L^{\mathsf{td}}_{pk} = \{(t, (u_r, u_t)) \mid \exists (x_2, (r, R_r, R_t)) : u_r = \mathbf{E}(r, ; R_r)$ and $u_t = g_1^{x_2}\mathbf{E}(0; R_t)H(t)^r\}$. We then define $L^{\mathsf{ext}}_{pk} = \{(t, (u_r, u_t)) \mid \mathbf{D}(u_t) \not\equiv x_1x_2 + y(t)\mathbf{D}(u_r) \bmod p \wedge \mathbf{D}(u_t) \not\equiv x_1x_2 + y(t)\mathbf{D}(u_r) \bmod q\}$. Since $(t, (u_r, u_t)) \in L^{\mathsf{td}}_{pk}$ holds if and only if $\mathbf{D}(u_t) \equiv x_1x_2 + y(t)\mathbf{D}(u_r) \pmod{n^d}$, it implies that $\mathbf{D}(u_t) \equiv x_1x_2 + y(t)\mathbf{D}(u_r) \pmod{n}$. Hence, $L^{\mathsf{td}}_{pk} \cap L^{\mathsf{ext}}_{pk} = \emptyset$.
- $\mathsf{ABM.enc}^{(t,(u_r,u_t))}(pk, m; (z, s, R_A, R_a, R_b))$**:** To encrypt message $m \in \mathbb{Z}_{n^d}$, it chooses $z, s \xleftarrow{\cup} \mathbb{Z}_{n^d}$ and computes $A := g_1^z H(t)^s u_t^m R_A^{n^d} \pmod{n^{d+1}}$, $a := \mathbf{E}(z; R_a) \cdot g_2^m \pmod{n^{d+1}}$ and $b := \mathbf{E}(s; R_b) \cdot u_r^m \pmod{n^{d+1}}$, where $R_A, R_a, R_b \xleftarrow{\cup} \mathbb{Z}^\times_{n^{d+1}}$. It outputs $c := (A, a, b)$ as the ciphertext of m on $(t, (u_r, u_t))$.
- $\mathsf{ABM.dec}^{(t,(u_r,u_t))}(sk, c)$**:** To decrypt $c = (A, a, b)$, it outputs

$$m := \frac{x_1\mathbf{D}(a) + y(t)\mathbf{D}(b) - \mathbf{D}(A)}{x_1x_2 - (\mathbf{D}(u_t) - y(t)\mathbf{D}(u_r))} \bmod n^d. \tag{2}$$

- $\mathsf{ABM.col}_1^{(t,(u_r,u_t))}(pk, x_2, (r, R_r, R_t))$**:** It picks up $\omega, \eta \xleftarrow{\cup} \mathbb{Z}_{n^d}$, $R'_A, R'_a, R'_b \xleftarrow{\cup} \mathbb{Z}^\times_{n^{d+1}}$. It then computes $A := g_1^\omega \cdot H(t)^\eta \cdot {R'_A}^{n^d} \pmod{n^{d+1}}$, $a := g^\omega {R'_a}^{n^d} \pmod{n^{d+1}}$, and $b := g^\eta {R'_b}^{n^d} \pmod{n^{d+1}}$. It outputs $c := (A, a, b)$ and $\xi := (x_2, (r, R_r, R_t), (u_r, u_t), \omega, \eta, R'_A, R'_a, R'_b)$.
- $\mathsf{ABM.col}_2(\xi, m)$**:** To open c to m, it computes $z = \omega - mx_2 \bmod n^d$, $s = \eta - mr \bmod n^d$, $\alpha = \lfloor(\omega - mx_2 - z)/n^d\rfloor$, and $\beta = \lfloor(\eta - mr - s)/n^d\rfloor$. It then sets $R_A := R'_A \cdot R_t^{-m} \cdot g_1^\alpha \cdot H(t)^\beta \pmod{n^{d+1}}$, $R_a := R'_a \cdot R_2^{-m} \cdot g^\alpha \pmod{n^{d+1}}$, and $R_b := R'_b \cdot R_r^{-m} \cdot g^\beta \pmod{n^{d+1}}$. It outputs (z, s, R_A, R_a, R_b), where $A = g_1^z H(t)^s u_t^m R_A^{n^d} \pmod{n^{d+1}}$, $a = \mathbf{E}(z; R_a) \cdot g_2^m \pmod{n^{d+1}}$, and $b = \mathbf{E}(s; R_b) \cdot u_r^m \pmod{n^{d+1}}$.

We note that $\mathsf{ABM.col}$ runs a canonical sigma protocol on L^{td}_{pk} to prove that the prover knows $(x_2, (r, R_r, R_t))$ such that $u_r = \mathbf{E}_{pk}(r; R_r)$ and $u_t = g_1^{x_2}\mathbf{E}_{pk}(0; R_t)H(t)^r$. Hence, the trapdoor mode works correctly when $(t, (u_r, u_t)) \in L^{\mathsf{td}}_{pk}$. On the contrary, $\mathsf{ABM.enc}$ runs a simulation algorithm of the sigma protocol with message (challenge) x. Notice that (A, a, b) implies the following linear system on $\mathbb{Z}_{n^d}$,

$$\begin{pmatrix} \mathbf{D}(A) \\ \mathbf{D}(a) \\ \mathbf{D}(b) \end{pmatrix} = \begin{pmatrix} x_1 & y(t) & \mathbf{D}(u_t) \\ 1 & 0 & x_2 \\ 0 & 1 & \mathbf{D}(u_r) \end{pmatrix} \begin{pmatrix} z \\ s \\ m \end{pmatrix} \tag{3}$$

The matrix is invertible if

$$\mathbf{D}(u_t) \neq (x_1x_2 + y(t)\mathbf{D}(u_r)) \pmod{p} \text{ and } \mathbf{D}(u_t) \neq (x_1x_2 + y(t)\mathbf{D}(u_r)) \pmod{q},$$

which means that $(t, (u_r, u_t)) \in L^{\mathsf{ext}}_{pk}$. Hence, the decryption mode works correctly.

Lemma 1 (Implicit in [18]). $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ *is PPRF with unforgeability on* $\widehat{L}^{\mathsf{td}}_{pk}(= U'_{pk} \backslash L^{\mathsf{ext}}_{pk})$*, under the assumptions, 3, 4, and 5.*

The proof is given in the full version [14]. By this lemma, we have:

Theorem 2. *The scheme constructed as above is an ABME scheme if the DCR assumption (Assumption 3), the non-tirvial divisor assmuption (Assumption 4), and the non-multiplication assumption (Assumption 5) hold true.*

This scheme has a ciphertext consisting of only 5 group elements (including (u_r, u_t)) and optimal expansion factor $\Omega(1)$. This scheme requires a public-key consisting of $\kappa + 3$ group elements along with some structure parameters.

8 ABM-LTF based ABME and Vice Versa

Hofheinz [18] has presented the notion of all-but-many lossy trapdoor function (ABM-LTF). We provide the definition in Appendix B. We remark that ABM-LTF requires that, in our words, $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ be *strongly* unforgeable, whereas ABME only requires it be unforgeable. However, as shown in [18], unforgeable PPRF can be converted into strongly unforgeable PPRF via a chameleon commitment scheme. Therefore, this difference is not important. We note that we can regard Hofheinz's DCR-based ABM-LTF (with only unforgeability) as a special case of our DCR-based ABME scheme by fixing a part of the coin space as $(R_A, R_a, R_b) = (1, 1, 1)$. Although the involved matrix of his original scheme is slightly different from ours, the difference is not essential. In the end, we can regard Hofheinz's DCR-based ABM-LTF as

$$\mathsf{ABM.eval}^{(t,(u_r,u_t))}(pk, (m, z, s)) := \mathsf{ABM.enc}^{(t,(u_r,u_t))}(pk, m; (z, s, 1, 1, 1)),$$

where (m, z, s) denotes a message. This ABM-LTF has $((d-1)\log n)$-lossyness. In the latest e-print version [18], Hofheinz has shown that his DCR-based ABM-LTF can be converted to SIM-SO-CCA PKE. To construct it, Hofheinz implicitly considered the following PKE scheme such that

$$\begin{aligned}\mathsf{ABM.enc}^{(t,(u_r,u_t))}(pk, M; (m, z, s)) :=& (\mathsf{ABM.eval}^{(t,(u_r,u_t))}(pk, (m, z, s)),\\ & M \oplus H(m, z, s)),\end{aligned}$$

where H is a suitable 2-universal hash function from $(\mathbb{Z}_{n^d})^3$ to $\{0,1\}^\kappa$ $(\kappa < n)$. According to the analysis in Sect. 7.2 in [18], if $d \geq 5$, it can open an ciphertext arbitrarily using Barvinok's alogorithm, when $(t, (u_r, u_t)) \in L^{\mathsf{loss}}$. Then it turns out ABME in our words. For practical use, it is rather inefficient, because its expansion rate of ciphertext length per message length is ≥ 31, and the modulus of $\geq n^6$ is required. The opening algorithm is also costly. Table 2 shows comparison.

On the contrary, our DCR-based ABME (strengthened with strong unforgeability) can be converted to ABM-LTF. Remember that $(A, a, b) = \mathsf{ABM.enc}^{(t,(u_r,u_t))}(pk, m; (z, s, R_A, R_a, R_b))$. It is obvious that we can extract not only message

ABME	expansion factor	ciphertext-length	message-length	pk-length
ABME from [18]	$\geq 31^*$	$(5(d+1)+1)\log n$	$\log n$	$(\kappa+3)d\log n$
Sect. 7.1 ($d \geq 1$)	$5+1/d$	$5(d+1)\log n$	$d\log n$	$(\kappa+3)d\log n$

Table 2. Comparison among ABMEs

$*$: $d \geq 5$ is needed.

m but (z, s) by inverting the corresponding matrix, but we point out that we can further retrieve (R_A, R_a, R_b), too. This mean that our DCR based ABME turns out ABM-LTF. Indeed, after extracting (m, z, s) from (A, a, b), we have $(R_A)^{n^d}, (R_a)^{n^d}, (R_b)^{n^d}$ in $\mathbb{Z}^{\times}_{n^{d+1}}$. We remark that R_A, R_a, R_b lie not in $\mathbb{Z}^{\times}_{n^{d+1}}$ but in $(\mathbb{Z}/n\mathbb{Z})^{\times}$. So, letting $\alpha = r^{n^d} \bmod n^{d+1}$ where $r \in (\mathbb{Z}/n\mathbb{Z})^{\times}$, $r = \alpha^{(n^d)^{-1}} \bmod n$ is efficiently solved by $\phi(n)$. Thus, our DCR based ABME turns out ABM-LTF with $(d\log n)$-lossyness for any $d \geq 1$, whereas Hofheinz's DCR based ABM-LTF is $((d-1)\log n)$-lossy.

ABM-LTF	exp. factor	output-length	input-length	lossyness
ABM-LTF [18]	5/3	$(5(d+1)+1)\log n$	$3d\log n$	$(d-1)\log n$
ABM-LTF from Sect. 7	5/3	$(5(d+1)+1)\log n$	$3(d+1)\log n$	$d\log n$

Table 3. Comparison among ABM-LTFs

9 Acknowledgments

We thank Kirill Morozov and his students for nice feedback in the early version of this work. We also thank Dennis Hofheinz for valuable discussion.

References

1. M. Bellare, D. Hofheinz, and S. Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 1–35. Springer-Verlag, 2009.
2. M. Bellare, S. Micali, and R. Ostrovsky. Perfect zero-knowledge in constant rounds. In *STOC '90*, pages 482–493. ACM, 1990.
3. J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In D. Boneh, editor, *CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 126–144. Springer-Verlag, 2003.

4. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2001)*, pages 136–145. IEEE Computer Society, 2001. The full version available at at Cryptology ePrint Archive `http://eprint.iacr.org/2000/067`.
5. R. Canetti and M. Fischlin. Universally composable commitments. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40. Springer-Verlag, 2001.
6. R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *STOC 2002*, pages 494–503. ACM, 2002. The full version is available at `http://eprint.iacr.org/2002/140`.
7. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Desmedt [11], pages 174–187.
8. I. Damgård and J. Groth. Non-interactive and reusable non-malleable commitment schemes. In *STOC 2003*, pages 426–437. ACM, 2003.
9. I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In K. Kim, editor, *PKC 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 125–140. Springer-Verlag, 2001.
10. I. Damgård and J. B. Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 581–596. Springer-Verlag, 2002. The full version is available at `http://www.brics.dk/RS/01/41/`.
11. Y. G. Desmedt, editor. *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.
12. S. Fehr, D. Hofheinz, E. Kiltz, and H. Wee. Encryption schemes secure against chosen-ciphertext selective opening attacks. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 381–402. Springer-Verlag, 2010.
13. M. Fischlin, B. Libert, and M. Manulis. Non-interactive and re-usable universally composable string commitments with adaptive security. In D. H. Lee and X. Wang, editors, *ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 468–485. Springer-Verlag, 2011.
14. E. Fujisaki. All-but-many encryption: A framework for efficient fully-equipped UC commitments. *IACR Cryptology ePrint Archive*, 2012:379, 2012.
15. E. Fujisaki. New constructions of efficient simulation-sound commitments using encryption and their applications. In O. Dunkelman, editor, *CT-RSA*, volume 7178 of *Lecture Notes in Computer Science*, pages 136–155. Springer-Verlag, 2012.
16. J. A. Garay, P. P.Mackenzie, and K. Yang. Strengthening zero-knowledge protocols using signatures. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 177–194. Springer-Verlag, 2003.
17. R. Gennaro. Multi-trapdoor commitments and their applications to proofs of knowledge secure under concurrent man-in-the-middle attacks. In M. K. Franklin, editor, *CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 220–236. Springer-Verlag, 2004. The full version available at at Cryptology ePrint Archive `http://eprint.iacr.org/2003/214`.
18. D. Hofheinz. All-but-many lossy trapdoor functions. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 209–227. Springer-Verlag, 2012. (last revised 18 Mar 2013 at http://eprint.iacr.org/2011/230).

19. T. Itoh, Y. Ohta, and H. Shizuya. Language dependent secure bit commitment. In Desmedt [11], pages 188–201.
20. Y. Lindell. Highly-efficient universally-composable commitments based on the DDH assumption. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 446–466. Springer-Verlag, 2011. The full version available at at Cryptology ePrint Archive `http://eprint.iacr.org/2011/180`.
21. P. MacKenzie and K. Yang. On simulation-sound trapdoor commitments. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 382–400. Springer-Verlag, 2004.
22. R. Nishimaki, E. Fujisaki, and K. Tanaka. An efficient non-interactive universally composable string-commitment scheme. *IEICE Transactions*, 95-A(1):167–175, 2012.
23. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, 1999.
24. C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer-Verlag, 2008.
25. C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In R. E. Ladner and C. Dwork, editors, *STOC 2008*, pages 187–196. ACM, 2008.

A Ideal Multi-Commitment Functionality

Functionality $\mathcal{F}_{\mathsf{MCOM}}$

$\mathcal{F}_{\mathsf{MCOM}}$ proceeds as follows, running with parties, $P_1, \ldots, P_n$, and an adversary $\mathcal{S}$:

- **Commit phase:** Upon receiving input $(\texttt{commit}, \texttt{sid}, \texttt{ssid}, P_i, P_j, x)$ from P_i, proceed as follows: If a tuple $(\texttt{commit}, \texttt{sid}, \texttt{ssid}, \ldots)$ with the same $(\texttt{sid}, \texttt{ssid})$ was previously recorded, does nothing. Otherwise, record the tuple $(\texttt{sid}, \texttt{ssid}, P_i, P_j, x)$ and send $(\texttt{receipt}, \texttt{sid}, \texttt{ssid}, P_i, P_j)$ to P_j and $\mathcal{S}$.
- **Reveal phase:** Upon receiving input $(\texttt{open}, \texttt{sid}, \texttt{ssid})$ from P_i, proceed as follows: If a tuple $(\texttt{sid}, \texttt{ssid}, P_i, P_j, x)$ was previously recorded, then send $(\texttt{reveal}, \texttt{sid}, \texttt{ssid}, P_i, P_j, x)$ to P_j and $\mathcal{S}$. Otherwise, does nothing.

Fig. 4. The ideal multi-commitment functionality

B All-But-Many Lossy Trapdoor Functions

We recall all-but-many lossy trapdoor functions (ABM-LTF) [18], by slightly modifying the notation to fit our purpose. All-but-many lossy trapdoor function $\mathsf{ABM.LTF} = (\mathsf{ABM.gen}, \mathsf{ABM.spl}, \mathsf{ABM.eval}, \mathsf{ABM.inv})$ consists of the following algorithms:

– $\mathsf{ABM.gen}$ is a PPT algorithm that takes 1^κ and outputs $(pk, (sk, w))$, where pk defines a set U_{pk}. We let $U'_{pk} = \{0,1\}^\kappa \times U_{pk}$. pk also determines two disjoint sets, L^{loss}_{pk} and L^{inj}_{pk}, such that $L^{\mathsf{loss}}_{pk} \cup L^{\mathsf{inj}}_{pk} \subset U'_{pk}$.
– $\mathsf{ABM.spl}$ is a PPT algorithm that takes (pk, w, t), where $t \in \{0,1\}^\kappa$, picks up inner random coins $v \leftarrow \mathsf{COIN}^{\mathsf{spl}}$, and computes $u \in U_{pk}$. We write $L^{\mathsf{loss}}_{pk}(t)$ to denote the image of $\mathsf{ABM.spl}$ on t under pk, i.e.,

$$L^{\mathsf{loss}}_{pk}(t) := \{u \in U_{pk} \mid \exists\, w,\, \exists\, v :\ u = \mathsf{ABM.spl}(pk, w, t; v)\}.$$

We require $L^{\mathsf{loss}}_{pk} = \{(t,u) \mid t \in \{0,1\}^\kappa \text{ and } u \in L^{\mathsf{loss}}_{pk}(t)\}$. We set $\widehat{L}^{\mathsf{loss}}_{pk} := U'_{pk} \backslash L^{\mathsf{inj}}_{pk}$. Since $L^{\mathsf{loss}}_{pk} \cap L^{\mathsf{inj}}_{pk} = \emptyset$, we have $L^{\mathsf{loss}}_{pk} \subseteq \widehat{L}^{\mathsf{loss}}_{pk} \subset U'_{pk}$.
– $\mathsf{ABM.eval}$ is a DPT algorithm that takes pk, (t,u), and message $x \in \mathsf{MSP}$ and computes $c = \mathsf{ABM.eval}^{(t,u)}(pk, x)$, where MSP denotes the message space uniquely determined by pk.
– $\mathsf{ABM.inv}$ is a DPT algorithm that takes sk, (t,u), and c, and computes $x = \mathsf{ABM.inv}^{(t,u)}(sk, c)$.

All-but-many encryption schemes require the following properties:

1. **Adaptive All-but-many property:** $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ is a probabilistic pseudo random function (PPRF), as defined in Sect. 3.1, with *strongly* unforgeability on $\widehat{L}^{\mathsf{loss}}_{pk} = U'_{pk} \backslash L^{\mathsf{inj}}_{pk}$. Strong unforgeability in this paper is called $\mathsf{evasiveness}$ in [18].
2. **Inversion** For every $\kappa \in \mathbb{N}$, every $(pk, (sk, w)) \in \mathsf{ABM.gen}(1^\kappa)$, every $(t,u) \in L^{\mathsf{inj}}_{pk}$, and every $x \in \mathsf{MSP}$, it always holds that

$$\mathsf{ABM.inv}^{(t,u)}(sk, \mathsf{ABM.eval}^{(t,u)}(pk, x)) = x.$$

3. **ℓ-Lossyness** For every $\kappa \in \mathbb{N}$, every $(pk, (sk, w)) \in \mathsf{ABM.gen}(1^\kappa)$, and every $(t,u) \in L^{\mathsf{loss}}_{pk}$, the image set $\mathsf{ABM.eval}^{(t,u)}(pk, \mathsf{MSP})$ is of size at most $|\mathsf{MSP}| \cdot 2^{-\ell}$.

Here L^{loss}_{pk} (resp. L^{inj}_{pk}) in ABM-LTFs corresponds to L^{td}_{pk} (resp. L^{ext}_{pk}) in ABMEs. We remark that ABM-LTFs [18] require that $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ should be *strongly* unforgeable, whereas ABMEs requires that $(\mathsf{ABM.gen}, \mathsf{ABM.spl})$ be just unforgeable.

C Assumptions and Some Useful Lemmas

Let us write $\Pi^{(d)}$ to denote DJ PKE with parameter d.

Assumption 3. *We say that the DCR assumption holds if for every PPT A, there exists a key generation algorithm $\mathbf{K}$ such that* $\mathsf{Adv}^{\mathsf{dcr}}_A(\kappa) =$

$$\Pr[\mathsf{Expt}^{\mathsf{dcr}-0}_A(\kappa) = 1] - \Pr[\mathsf{Expt}^{\mathsf{dcr}-1}_A(\kappa) = 1]$$

is negligible in κ, where

$$\begin{array}{l|l}
\mathsf{Expt}^{\mathsf{dcr}-0}_{A}(\kappa): & \mathsf{Expt}^{\mathsf{dcr}-1}_{d,A}(\kappa): \\
\quad n \leftarrow \mathbf{K}(1^\kappa);\ R \xleftarrow{\mathsf{U}} \mathbb{Z}^{\times}_{n^2} & \quad n \leftarrow \mathbf{K}(1^\kappa);\ R \xleftarrow{\mathsf{U}} \mathbb{Z}^{\times}_{n^2} \\
\quad c = R^n \bmod n^2 & \quad c = (1+n)R^n \bmod n^2 \\
\quad \textit{return } A(n,c). & \quad \textit{return } A(n,c).
\end{array}$$

Assumption 4 ([18]). *We say that the non-trivial divisor assumption holds on $\Pi^{(d)}$ if for every PPT A, $\mathsf{Adv}^{\mathsf{divisor}}_{A,\Pi^{(d)}}(\kappa) = \mathsf{negl}(\kappa)$ where*

$$\mathsf{Adv}^{\mathsf{divisor}}_{A,\Pi^{(d)}}(\kappa) = \Pr[(pk,sk) \leftarrow \mathbf{K}(1^\kappa);\ A(pk) = c :\ 1 < \gcd(\mathbf{D}(c), n) < n].$$

This assumes that an adversary cannot compute an encryption of a non-trivial divisor of n, i.e., $\mathbf{E}(p)$, under given public-key pk_{dj} only. Since the adversary is only given pk_{dj}, the assumption is plausible.

Lemma 2. *If A is an adversary against $\Pi^{(d)}$, there is adversary A' against $\Pi^{(1)}$ such that*

$$\mathsf{Adv}^{\mathsf{divisor}}_{A,\Pi^{(d)}}(\kappa) \leq \mathsf{Adv}^{\mathsf{divisor}}_{A',\Pi^{(1)}}(\kappa).$$

Assumption 5 ([18]). *We say that the non-multiplication assumption holds on DJ PKE $\Pi^{(d)}$ if for every PPT adversary A, the advantage of A, $\mathsf{Adv}^{\mathit{mult}}_{A,\Pi^{(d)}}(\kappa) = \mathsf{negl}(\kappa)$, where $\mathsf{Adv}^{\mathit{mult}}_{A,\Pi^{(d)}}(\kappa) = \Pr[(pk,sk) \leftarrow \mathbf{K}(1^\kappa);\ c_1, c_2 \leftarrow \mathbb{Z}^{\times}_{n^{d+1}};\ c^* \leftarrow A(pk, c_1, c_2) : \mathbf{D}_{sk}(c^*) = \mathbf{D}_{sk}(c_1) \cdot \mathbf{D}_{sk}(c_2)]$.*

This assumes that an adversary cannot compute $\mathbf{E}(x_1 \cdot x_2)$ for given $(pk_{\mathsf{dj}}, \mathbf{E}(x_1), \mathbf{E}(x_2))$. If the multiplicative operation is easy, DJ PKE turns out a fully-homomorphic encryption (FHE), which is unlikely. Although breaking the non-multiplication assumption does not mean that DJ PKE turns out a FHE, this connection gives us some feeling that this assumption is plausible.

Lemma 3. *If A is an adversary against DJ PKE $\Pi^{(d)}$, there is an adversary A' against $\Pi^{(1)}$ such that*

$$\mathsf{Adv}^{\mathit{mult}}_{A,\Pi^{(d)}}(\kappa) \leq \mathsf{Adv}^{\mathit{mult}}_{A',\Pi^{(1)}}(\kappa).$$