HAL open science

New and Improved Constructions for Partially Equivocable Public Key Encryption

Benoît Libert, Alain Passelègue, Mahshid Riahinia

To cite this version:

Benoît Libert, Alain Passelègue, Mahshid Riahinia. New and Improved Constructions for Partially Equivocable Public Key Encryption. SCN 2022 - 13th Conference on security and cryptography for networks, Sep 2022, Amalfi, Italy. pp.1-31. hal-03830141

HAL Id: hal-03830141

https://inria.hal.science/hal-03830141v1

Submitted on 26 Oct 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

New and Improved Constructions for Partially Equivocable Public Key Encryption

Benoît Libert[1,2], Alain Passelègue[2,3], and Mahshid Riahinia[2]

[1] CNRS, Laboratoire LIP, France
[2] ENS de Lyon, Laboratoire LIP (U. Lyon, CNRS, ENSL, Inria, UCBL), France
[3] Inria, France

Abstract. Non-committing encryption (NCE) is an advanced form of public-key encryption which guarantees the security of a Multi-Party Computation (MPC) protocol in the presence of an adaptive adversary. Brakerski et al. (TCC 2020) recently proposed an intermediate notion, termed Packed Encryption with Partial Equivocality (PEPE), which implies NCE and preserves ciphertext-rate (up to a constant factor). In this work, we propose three new constructions of rate-1 PEPE based on standard assumptions. In particular, we obtain the first constant ciphertext-rate NCE construction from the LWE assumption with polynomial modulus, and from the Subgroup Decision assumption. We also propose an alternative DDH-based construction with guaranteed polynomial running time. We also make clarifications on the requirements of PEPE.

Keywords. Non-committing encryption, standard assumptions, ciphertext rate, equivocable encryption.

1 Introduction

Non-committing encryption (NCE) was introduced by Canetti et al. [CFGN96] as a form of encryption that guarantees the security of an MPC protocol in the presence of an adaptive adversary. Informally, NCE is a form of public-key encryption that allows one to generate ciphertexts that can be later opened to an arbitrary message. Intuitively, by using NCE as the encryption tool in an MPC protocol, we can fool the adversary by opening the internal state of a newly corrupted party to an arbitrary message while being able to prove that this arbitrary internal state is consistent with the public transcript of the protocol.

An important property of an NCE scheme that determines its efficiency, like any other public-key encryption scheme, is its ciphertext rate, i.e., the ratio of ciphertext length to message length.

Prior Works. There is a large literature on NCE and we decide to focus on NCE constructions achieving optimal round complexity without random oracle. That is, we only discuss 2-round protocols. Recall that Nielsen [Nie02] proved that non-interactive NCE is impossible without random oracle.

Canetti et al. [CFGN96] presented the first NCE constructions based on the RSA or the computational Diffie-Hellman (CDH) assumptions. Each of their constructions achieves a ciphertext rate of $\mathcal{O}(\lambda^2)$. First improvements were only achieved after thirteen years by Choi et al. [CDSMW09]. In the latter work, the authors constructed an NCE scheme based on the factoring problem and achieving linear ciphertext rate $\mathcal{O}(\lambda)$. Hemenway et al. [HOR15] achieved sub-linear ciphertext-rate ($\mathcal{O}(\log \lambda)$) from the Φ-hiding assumption, which they later improved to $\mathsf{poly}(\log \lambda)$ [HORR16]. The latter construction is based on the (Ring) Learning-with-Errors (LWE) assumption with super-polynomial modulus-to-noise ratio. Canetti et al. [CPR17] obtained optimal ciphertext-rate using the power of indistinguishability obfuscation (iO) and in the CRS model. Yoshida et al. [YKT19] put forth an approach allowing to construct NCE with ciphertext rate $\mathcal{O}(\log \lambda)$ under the standard Decisional Diffie-Hellman (DDH) assumption. Under standard assumptions, constant ciphertext rate was recently achieved in two concurrent works: by Yoshida et al. in [YKXT20], from DDH or LWE with super-polynomial modulus; and by Brakerski et al. in [BBD$^+$20], under the DDH assumption, as well as a construction from Quadratic Residuosity (QR) assumption which requires oblivious sampling and CRS setup[1], and a construction from LWE with super-polynomial modulus-to-noise ratio.

1.1 Our Contributions

We follow the same approach as Brakerski et al. [BBD$^+$20] and obtain constant-rate NCE by constructing an intermediate primitive called *packed encryption with partial equivocality* (PEPE) that the latter authors proved to imply NCE with only a constant factor loss in the ciphertext rate. NCE is obtained by composing any PEPE scheme with a constant-rate error-correcting code (ECC), the latter being implied by the existence of one-way functions: to encrypt a message, first encode it using an ECC, and then encrypt the encoding using the PEPE encryption algorithm (decryption is done via decrypting-then-decoding). We refer the reader to [BBD$^+$20] for more details on this generic transform. We thus focus on constructing rate-1 PEPE from various assumptions, and rely on the transformation of [BBD$^+$20] to obtain constant ciphertext rate NCE. Specifically, we obtain three constructions of rate-1 PEPE, therefore of constant-ciphertext-rate NCE. Our first construction is secure assuming the hardness of the LWE problem with *polynomial* modulus and inverse-error rate, and our second construction is secure under the DDH assumption. We then explain why the QR-based PEPE construction of [BBD$^+$20] is flawed: beside a CRS setup, oblivious sampling seems necessary for their construction, and the latter is not known

[1] These requirements are overlooked in [BBD$^+$20]. Oblivious sampling is mentioned in [BBD$^+$20], but it is not known in the context of their QR construction, to the best of our knowledge. The need for a CRS is not mentioned. Both these requirements seem inherent for constructions relying on the hardness of factoring. We provide more details below.

in the group they work with, to the best of our knowledge. We then provide a substitute construction of rate-1 PEPE from Subgroup Decision (SD) assumption in pairing-free composite order groups. This construction requires a trusted setup (CRS) in order to generate the group order, and we provide evidence that the same requirement applies to any PEPE construction whose security requires hardness of factoring. In the process, we slightly modify the definition of PEPE that also presented a minor flaw. Table 1 displays a comparison between our results and prior ones.

	Ciphertext Rate	Assumption	Setup
[CFGN96]	$\mathcal{O}(\lambda^2)$	RSA, CDH	-
[CDSMW09]	$\mathcal{O}(\lambda)$	Factoring Blum integers	-
[HOR15]	$\mathcal{O}(\log \lambda)$	Φ-hiding	Oblivious sampling of RSA modulus
[HORR16]	$\mathsf{poly}(\log \lambda)$	LWE, Ring-LWE	superpolynomial LWE modulus-to-noise ratio
[CPR17]	$1 + o(1)$	iO	CRS
[YKT19]	$\mathcal{O}(\log \lambda)$	DDH	-
[BBD+20]	$\mathcal{O}(1)$	LWE, DDH, QR	superpolynomial LWE modulus-to-noise ratio
[YKXT20]	$\mathcal{O}(1)$	LWE, DDH	superpolynomial LWE modulus-to-noise ratio
This Work	$\mathcal{O}(1)$	LWE, DDH SD	- CRS

Table 1. Comparison between our NCE schemes and previously proposed NCE schemes. We only discuss 2-round constructions without random oracle.

LWE construction. We propose the first constant ciphertext-rate NCE scheme relying on the hardness of LWE with polynomial modulus-to-noise ratio. This result improves the recent works of Brakerski et al. [BBD+20] and Yoshida et al. [YKXT20], which rely on LWE with super-polynomial modulus-to-noise ratio. Our construction is similar to the one from [BBD+20], but we manage to avoid noise flooding that was used to equivocate the ciphertext randomness by convolving discrete Gaussians using a lemma from [LSSS17] to properly simulate the noise of simulated ciphertexts.

DDH construction. We present a simple construction for rate-1 PEPE from the DDH assumption which differs from the construction in [BBD+20] as follows. In the latter work, the authors use a ciphertext compression algorithm (that relies on sampling a PRF such that evaluation on some ciphertext-dependent messages satisfies some property) that runs in expected polynomial time. Instead, we apply a universal hash function to the encryption randomness and then encrypt the message using the output of the hash as a one-time pad. Thus, we preserve the constant ciphertext rate while avoiding the disadvantage of the compression algorithm. In particular, our encryption algorithm works in strict polynomial time.

SD construction. We propose the first constant ciphertext-rate NCE (via PEPE) based on the subgroup decision (SD) assumption [BGN05]. This construction uses a CRS, which seems inherent when relying on the hardness of factoring. The main purpose of proposing this construction, aside from broadening the family of assumptions implying NCE, is to provide a substitute for the QR-based construction of [BBD+20], which present several flaws, as mentioned above and discussed below in the technical overview.

1.2 Technical Overview

PEPE. As defined in [BBD+20], a *Packed Encryption with Partial Equivocality (PEPE)* scheme is a tuple $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EquivPK}, \mathsf{EquivCT})$ of efficient (PPT) algorithms such that $\mathsf{KeyGen}(b, I, r_G)$ algorithm takes as input a bit b, a subset $I \subset [\ell]$, where ℓ is the length of the messages that can be encrypted in the scheme, and it uses some randomness r_G to produce a public key pk and a secret key sk. This algorithm runs in two modes: real mode (if $b = 0$), or ideal mode (if $b = 1$). In the real mode, the scheme should satisfy the correctness property of a regular public-key encryption scheme restricted to the subset I. Namely, the output (of length ℓ) of the decryption algorithm on a ciphertext encrypting a message M should be equal to M_i, for $i \in I$.

The two other algorithms $\mathsf{EquivPK}$ and $\mathsf{EquivCT}$ can be used in the following way: $\mathsf{EquivPK}(\mathsf{sk}, b, (I, r_G), I')$ on input $I' \subset I$, and sk, where $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(b, I, r_G)$, outputs a new randomness r'_G, such that the pair (pk', r'_G) with $(\mathsf{pk}', \mathsf{sk}') \leftarrow \mathsf{KeyGen}(0, I', r'_G)$ is indistinguishable from an honestly generated (in the real mode) pair for set I'. That is, an efficient adversary cannot distinguish between pairs (pk', r'_G) obtained by equivocating from a larger set from honest pairs (pk', r'_G) obtained when running KeyGen in the real mode ($b = 0$) for the set I' with randomness r'_G. Here, we emphasize that for PEPE to actually imply NCE[1], we should require that the public key remains unchanged when equivocating, meaning that $\mathsf{pk}' = \mathsf{pk}$. The original definition of PEPE only required distributions of (honest and equivocated) randomness r'_G to be indistinguishable,

[1] In other words, in order for the transformation of PEPE to NCE, proposed in [BBD+20], to properly work.

unconditioned on $\mathsf{pk}' = \mathsf{pk}$. We fix this issue by requiring the joint distributions to be indistinguishable (See Definition 8).

The second equivocation algorithm allows to equivocate ciphertexts in the ideal mode: $\mathsf{EquivCT}(\mathsf{sk}, (M, r_E), \{M'_i\}_{i \notin I})$, on input an encryption randomness r_E, and messages M and M', such that $M_i = M'_i$ for $i \in I$, outputs a new encryption randomness r'_E. No efficient adversary should be able to tell apart whether the distribution (pk, M', r'_E) was obtained by equivocating an encryption of a different message M in the ideal mode or by honestly encrypting in the real mode.

We now dive into the details of our PEPE constructions.

LWE Construction. We recall the LWE-based PEPE scheme of [BBD+20] which requires a super-polynomial modulus-to-noise ratio, and then we show how we modify their scheme and obtain a rate-1 PEPE scheme from LWE with polynomial modulus. In the real mode, the public key consists of a random integer matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$, and vectors $\{\mathbf{v}_i\}_{i=1}^{\ell}$ that are either LWE samples $\mathbf{v}_i = \mathbf{A}\mathbf{s}_i + \mathbf{e}_i$ if $i \in I$, or random vectors $\mathbf{v}_i \xleftarrow{\$} \mathbb{Z}_q^n$ if $i \notin I$. In the equivocal mode, the public key has the same structure as in the real mode, except that the matrix $\mathbf{A}$ and random vectors $\{\mathbf{v}_i\}_{i \notin I}$ now come from the columns of a random matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times (k+\ell-|I|)}$ for which a lattice trapdoor $\mathsf{td}_{\mathbf{B}}$ [GPV08,MP12] is available. Other vectors $\{\mathbf{v}_i\}_{i \in I}$ are chosen as LWE samples as in the real mode. To encrypt a message $M \in \{0,1\}^{\ell}$, they use the packed Regev encryption scheme [Reg05] together with a masking noise as follows: first, it samples a random vector $\mathbf{r} \xleftarrow{\$} D_{\sigma}^n$ and computes $\mathbf{c}_1 \leftarrow \mathbf{r}^T\mathbf{A}$, and $w_{2,i} \leftarrow \mathbf{r}^T\mathbf{v}_i + e_i + \lfloor q/2 \rceil M_i \in \mathbb{Z}_q$, where $e_i \xleftarrow{\$} D_{\sigma'}$ for each $i \in [\ell]$. Next, it compresses $(\mathbf{c}_1, \{w_{2,i}\}_{i=1}^{\ell})$ using a technique introduced in [BDGM19]. This technique changes a ciphertext of the form $\mathsf{ct} = (\mathbf{c}_0, w_{2,1}, \cdots, w_{2,\ell}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^{\ell}$ into a tuple $(\mathbf{c}_0, \{c_i\}_{i \in [\ell]}, z)$, where $c_i \in \{0,1\}$ for each $i \in [\ell]$, and $z \in \mathbb{Z}_q$. Decryption also proceeds using the decryption algorithm of this compression procedure.

The equivocation of public key randomness for a subset $I' \subseteq I$ is done simply by outputting the secret keys for the indices that are in the subset I', and the unmodified public key elements for the rest. Note that here, for indices in $I \setminus I'$, we are claiming that $\mathbf{v}_i$'s which are formed as $\mathbf{A}\mathbf{s}_i + \mathbf{e}_i$ (as the output of $\mathsf{KeyGen}(b = 1, I, r_G)$), are indistinguishable from random vectors of $\mathbb{Z}_q^n$ (as the output of $\mathsf{KeyGen}(b = 0, I', r'_G)$). This holds assuming the hardness of LWE.

We now explain the ciphertext equivocation algorithm of [BBD+20]. Recall that $\mathsf{td}_{\mathbf{B}}$ allows sampling a short vector of the lattice $\Lambda_{\mathbf{y}}^{\perp}(\mathbf{B}) = \{\mathbf{r} \in \mathbb{Z}_q^n : \mathbf{B}^T \cdot \mathbf{r} = \mathbf{y}\}$, where $\mathbf{y} \in \mathbb{Z}_q^{k+\ell-|I|}$. Let us assume that $(\mathbf{c}_1, \{w_{2,i}\}_{i=1}^{\ell})$ is a packed Regev encryption of a message M with the encryption randomness $(\mathbf{r}, \{e_i^*\}_{i \in [\ell]})$ and we want to explain it as an encryption of a message M', where $M_i = M'_i$ for $i \in I$. Using the lattice trapdoor $\mathsf{td}_{\mathbf{B}}$, one can sample a short (in fact, Gaussian) $\bar{\mathbf{r}} \in \mathbb{Z}^n$ such that

$$\bar{\mathbf{r}}^T\mathbf{A} = \mathbf{r}^T\mathbf{A} \quad \text{and} \quad \bar{\mathbf{r}}^T\mathbf{v}_i = w_{2,i} - e'_i - \lfloor q/2 \rceil M'_i \ ,$$

for $i \notin I$, where e'_i is sampled from $D_{\sigma'}$. This gives a Gaussian vector $\bar{\mathbf{r}}$ and encryption noise e'_i, for indices $i \notin I$, that can explain $(\mathbf{c}_1, \{w_{2,i}\}_{i \notin I})$ as a valid encryption of bits $\{M'_i\}_{i \notin I}$. At this point, one still has to craft an encryption noise e'_i for all $i \in I$ such that

$$\begin{aligned}\bar{\mathbf{r}}^T \mathbf{v}_i + e'_i + \lfloor q/2 \rceil M'_i = \mathbf{r}^T \mathbf{v}_i + e^*_i + \lfloor q/2 \rceil M_i &\xRightarrow[M_i = M'_i]{i \in I} \bar{\mathbf{r}}^T \mathbf{v}_i + e'_i = \mathbf{r}^T \mathbf{v}_i + e^*_i \\ &\xRightarrow[\mathbf{v}_i = \mathbf{A}\mathbf{s}_i + \mathbf{e}_i]{i \in I} \bar{\mathbf{r}}^T \mathbf{e}_i + e'_i = \mathbf{r}^T \mathbf{e}_i + e^*_i \\ &\Longrightarrow e'_i = e^*_i + (\mathbf{r} - \bar{\mathbf{r}})^T \mathbf{e}_i.\end{aligned}$$

The security proof requires to make sure that the distribution of e'_i which is computed by this equation is indistinguishable from the distribution $D_{\sigma'}$ used in the real mode. To this end, Brakerski et al. [BBD+20] use the noise flooding technique, which is based on the property that, if B/σ' is negligible (where B is bound for $|e|$), then $D_{\sigma'} + e$ is statistically close to $D_{\sigma'}$. Thus, if the modulus-to-noise ratio is set to be super-polynomial, the equivocation algorithm of [BBD+20] outputs randomnesses within negligible statistical distance from the correct distribution.

We remove this issue by relying on a Lemma from [LSSS17] which states that, for matrices $\mathbf{B} \in \mathbb{Z}_q^{n \times k'}$ and $\mathbf{E} \in \mathbb{Z}^{n \times \ell'}$, given a tuple of the form $(\mathbf{r}^T\mathbf{B}, \mathbf{r}^T\mathbf{E} + \mathbf{e}^T)$, where $(\mathbf{r}, \mathbf{e})$ is sampled from $D_\sigma^n \times D_{\sigma'}^{\ell'}$, a trapdoor for $\mathbf{B}$ allows resampling a short vector $\bar{\mathbf{r}}$ such that $\bar{\mathbf{r}}^T\mathbf{B} = \mathbf{r}^T\mathbf{B}$ while obtaining $\bar{\mathbf{e}}$ with the correct distribution such that $\bar{\mathbf{r}}^T\mathbf{E} + \bar{\mathbf{e}}^T = \mathbf{r}^T\mathbf{E} + \mathbf{e}^T$. We then tweak the equivocation algorithm of [BBD+20] by resampling the Gaussian vector $\bar{\mathbf{r}}$ with an appropriate covariance matrix, making sure that the output of the equivocation algorithm is statistically indistinguishable from the real encryption randomness, without the super-polynomial modulus-to-noise requirement.

DDH Construction. Under the DDH assumption, we work with a group $\mathbb{G}$ of prime order p and generator g. Denote by $n \in \mathbb{N}$ some integer. In the real mode, the public key is of the form $(\{g_j\}_{j \in [n]}, \{h_{i,j}\}_{i \in [\ell], j \in [n]})$, where each g_j is a random element of the group $\mathbb{G}$, and $h_{i,j}$ is equal to $g_j^{s_i}$, where $s_i \xleftarrow{\$} \mathbb{Z}_p$ if $i \in I$, or it is sampled randomly from $\mathbb{G}$, otherwise. Here, the secret key consists of the subset I, and the secrets $\{s_i\}_{i \in I}$. The public key has the same structure in the ideal mode, except that we set each g_j to be of the form g^{a_j}, where $a_j \xleftarrow{\$} \mathbb{Z}_p$, and each $h_{i,j}$ is equal to $g^{z_{i,j}}$, where $z_{i,j} \xleftarrow{\$} \mathbb{Z}_p$, if $i \notin I$. The secret key in this case is of the form $\mathsf{sk} = (I, \{s_i\}_{i \in I}, \{a_j\}_{j \in [n]}, \{z_{i,j}\}_{i \notin I, j \in [n]})$.

To encrypt a message $M \in \{0,1\}^\ell$, we first sample randomness $r_1, \cdots, r_n \xleftarrow{\$} \mathbb{Z}_p$. We use a universal hash function, $H : \mathbb{G} \to \{0,1\}$, to compute a ciphertext as $(c_0, c_1, \cdots, c_\ell)$, where $c_0 = \prod_j^n g_j^{r_j}$, and $c_i = M_i \oplus H(\prod_{j=1}^n h_{i,j}^{r_j}) \in \{0,1\}$. Note that for each $i \in [\ell]$, we have:

(i) If $i \in I : c_i = M_i \oplus H\left(\prod_{j=1}^{n} h_{i,j}^{r_j}\right) = M_i \oplus H\left((\prod_{j=1}^{n} g_j^{r_j})^{s_i}\right) = M_i \oplus H(c_0^{s_i})$

(ii) If $i \in [\ell] \setminus I : c_i = M_i \oplus H\left(\prod_{j=1}^{n} h_{i,j}^{r_j}\right) = M_i \oplus H\left(\prod_{j=1}^{n} g^{z_{i,j} \cdot r_j}\right)$ (when $b = 1$)

To decrypt, each M_i can be computed as $c_i \oplus H(c_0^{s_i})$ for all $i \in I$.

The equivocation of public key randomness for a subset $I' \subseteq I$ is done simply by outputting the secret keys for the indices that are in the subset I', and the unmodified public key elements for the rest. Here again, for indices in $I \setminus I'$, we are claiming that $h_{i,j}$'s which are formed as $g_j^{s_i} = g^{a_j \cdot s_i}$ (as the output of $\mathsf{KeyGen}(b = 1, I, r_G)$), are indistinguishable from random elements of $\mathbb{G}$ (as the output of $\mathsf{KeyGen}(b = 0, I', r'_G)$). This holds assuming DDH in $\mathbb{G}$.

The idea behind the $\mathsf{EquivCT}$ algorithm is inspired by the construction of sender-equivocable lossy public-key encryption from the Matrix Diffie-Hellman assumptions [HJR16]. Suppose that $\mathsf{ct} = (c_0, c_1, \cdots, c_\ell)$ encrypts the message $M \in \{0,1\}^\ell$ using the encryption randomness $r_1, \cdots, r_n$. Let $M' \in \{0,1\}^\ell$ a targeted message such that $M_i = M'_i$ for all $i \in I$. To equivocate the ciphertext randomness for M', we have to find $\bar{r}_1, \cdots, \bar{r}_n$ such that $c_0 = \prod_j^n g_j^{\bar{r}_j}$, and

$$c_i \oplus M'_i = H\big(\prod_{j=1}^{n} g^{z_{i,j} \cdot \bar{r}_j}\big).$$

Note that, since M' agrees with M on indices $i \in I$, keeping c_0 unchanged guarantees that $c_i = M'_i \oplus H(c_0^{s_i})$ for all $i \in I$ (Equation (i)). In order to find such a randomness, we do as follows: we first find a random element t_i of $\mathbb{Z}_p$, for indices $i \notin I$, such that $c_i = M'_i \oplus H(g^{t_i})$. Since H is a universal hash function, due to the Leftover Hash Lemma, the distribution of $H(g^{t_i})$ is statistically close to uniform over $\{0,1\}$. The probability $\Pr[H(g^{t_i}) = c_i \oplus M'_i]$, for a random t_i, is thus $\geq 1/2 - \mathsf{negl}(\lambda)$, where λ is the security parameter. Hence, this task terminates in polynomial time with overwhelming probability if we repeatedly sample t_i until a suitable candidate is found. Then, we solve the following system of equations to find a vector $\bar{\mathbf{r}}$ over $\mathbb{Z}_q$ such that:

$$\begin{aligned} \mathbf{a} \cdot \bar{\mathbf{r}}^T &= \mathbf{a} \cdot \mathbf{r}^T \\ \mathbf{z}_{i_1} \cdot \bar{\mathbf{r}}^T &= t_{i_1} \\ &\vdots \\ \mathbf{z}_{i_\alpha} \cdot \bar{\mathbf{r}}^T &= t_{i_\alpha}, \end{aligned}$$

where $\{i_1, \cdots, i_\alpha\} = [\ell] \setminus I$, $\mathbf{z}_{i_k} = (z_{i_k,1}, \cdots, z_{i_k,n})$, $\mathbf{a} = (a_1, \cdots, a_n)$, and $\mathbf{r} = (r_1, \cdots, r_n)$. Note that, since $\mathbf{a}$ and each $\mathbf{z}_{i_k}$ are chosen uniformly at random, this system of equation is full rank with overwhelming probability. We can thus find suitable encryption randomness $\bar{\mathbf{r}}$ that explains the ciphertext ct as an encryption of the message M'.

On CRS and Oblivious Sampling Requirements. Let us first focus on oblivious sampling. As illustrated above, public key elements are of 2 types; (1) elements for positions in I, for which we know the underlying secrets (denoted by $\{s_i\}_{i \in I}$) in both modes, and (2) elements outside I for which the underlying secrets are *only* known in the ideal mode. To avoid relying on erasures, public key elements for indices outside I should be sampled *obliviously* in the real mode (i.e., without knowing the underlying secrets).

Recall that a PEPE scheme has to satisfy two properties: For indices in I, decryption must be correct while, for indices outside I, ciphertexts must be equivocable in ideal mode. In the ideal mode, the key components for indices in and outside I thus have to satisfy different properties. Elements within I should be associated with some secret information enabling decryption (e.g., their discrete logarithm). At the same time, in order to equivocate public key randomness, it should be possible to equivocate from a set I to a smaller set I' without changing the public key. To this end, one requirement is that the components for indices in $I \setminus I'$, which were originally sampled to enable decryption in either mode (e.g., as DDH tuples with known discrete logarithm), should be indistinguishable from elements that were originally sampled outside I in the real mode (e.g., random group elements). More specifically, they should be indistinguishable even given the randomness used to generate the key components. Since some computational assumption has to underlie the pseudorandomness of key components in $I \setminus I'$, some information should remain hidden about these elements given the randomness (e.g., in the DDH case, it would be easy to know whether they are DDH tuple or random group elements given their discrete log). For this reason, public key elements outside I should be sampled obliviously. Then, when equivocating, it is sufficient to directly include the public key elements for those indices in the key generation randomness, and use the hardness assumption to prove indistinguishability.

However, there should be a way to explain how obliviously sampled elements were chosen and this "explanation" should be compatible with a reduction from the computational assumption that underlies the indistinguishability of public key components. In the DDH case, for instance, there should be an inverse-sampling algorithm that, on input of a group element $h \in \mathbb{G}$ which is part of a DDH instance, samples uniformly from the set of random coins leading the real oblivious sampling algorithm to output h. While oblivious sampling is efficiently doable using standard techniques (see, e.g., [CF01, Section 4.3.2]), it is more problematic in groups of hidden order where standard inverse-sampling techniques require knowing the group order.

Now, regarding the need for a CRS, let us consider a construction relying on factoring-based assumption in which a composite integer $N = pq$ is sampled during the key generation in real mode (when $b = 0$). According to the definition of PEPE, the randomness used to sample N has to be included in the key generation randomness r_G. This randomness includes the factors p, and q. Then, when equivocating the key generation randomness, one can not longer rely on any hardness assumption related to hardness of factoring N, if p and q are included

in the randomness. As a result, we have to consider a separate Setup algorithm that generates the group order $N = pq$ and set it as a common reference string crs, for which a trapdoor (i.e., p and q) can be used by randomness equivocation algorithms.

It turns out that the QR-based construction of [BBD+20] would require both a trusted setup (which is easily fixed) and oblivious sampling to be correct. But it is not clear how the latter can be done in the group of quadratic residues modulo a safe prime product. Here, we consider the construction with a CRS setup that generates a safe prime $N = pq$, to focus only on the oblivious sampling problem. Consider $\mathbb{J}_N$ to be the multiplicative subgroup of $\mathbb{Z}_N^*$ with Jacobi symbol $+1$, and $\mathbb{QR}_N$ denotes the subgroup of quadratic residues modulo N. Letting g be a generator of $\mathbb{QR}_N$, it is not clear how to sample an element from $\mathbb{QR}_N$ without knowing its discrete log with respect to g and while remaining able to use the quadratic residuosity assumption in the security proof. The only obvious way to sample from this group is to pick a random element from $\mathbb{Z}_N^*$ and square it. Back to the construction, in the real mode, the public key is of the form $(\{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell],j\in[n]})$, where each g_j is a random element of $\mathbb{QR}_N$, and $h_{i,j}$ is equal to $g_j^{s_i}$, where $s_i \xleftarrow{\$} \mathbb{Z}_{(N-1)/2}$ if $i \in I$, or it is sampled from $\mathbb{QR}_N$, otherwise. The secret key is $\mathsf{sk} = (I, \{s_i\}_{i\in I})$. Here, the key generation randomness contains the sampling information of g_j's and $h_{i,j}$'s for $i \notin I$, which is the square root of them. The public key has the same structure in the equivocal mode, except that g_j's are of the form g^{a_j}, where $a_j \xleftarrow{\$} \mathbb{Z}_{\phi(N)/2}$, and each $h_{i,j}$ is a random element of $\mathbb{J}_N \setminus \mathbb{QR}_N$ if $i \notin I$. When equivocating the public key randomness in the equivocal mode with respect to a subset $I' \subseteq I$, we have to find a root of ideally-generated elements $h_{i,j}$'s for $i \notin I'$ in order to include them in r'_G. This is not possible because these elements are from $\mathbb{J}_N \setminus \mathbb{QR}_N$. Each element of $\mathbb{J}_N \setminus \mathbb{QR}_N$ can be written as $-g^\alpha$, where g is a square since it is a generator of $\mathbb{QR}_N$, but -1 is not a square when N is a safe prime product. Instead, if we knew how to oblivious sample from $\mathbb{QR}_N$, then we could follow the proof of [BBD+20] and directly include $h_{i,j} \xleftarrow{\$} \mathbb{J}_N \setminus \mathbb{QR}_N$ for all $i \notin I'$ in r'_G, in which case they would be indistinguishable from obliviously sampled elements of $\mathbb{QR}_N$ under the QR assumption. The same issue would also happen with a construction from the Decision Composite Residuoisity (DCR) assumption, since it is not clear how to obliviously sample elements from the subgroup of $2N$-th residues.

SD Construction. Our construction from the SD assumption avoids oblivious sampling difficulties (which arise in previous factoring-based constructions) because it only requires to obliviously sample from a public order group. It works over a multiplicative cyclic group $\mathbb{G} \approx \mathbb{G}_q \times \mathbb{G}_p$, where $\mathbb{G}$ is of order $N = pq$, $\mathbb{G}_p = \{x^q \mod N : x \in \mathbb{G}\}$ is the subgroup of $\mathbb{G}$ of order p, and $\mathbb{G}_q = \{x^p \mod N : x \in \mathbb{G}\}$ is the subgroup of $\mathbb{G}$ of order q, and p and q are sufficiently large prime numbers. Since the assumption is implied by the hardness of factoring, we require that there exists a trusted Setup algorithm that generates a group order $N = pq$, and provides the factorization p and q for randomness

equivocation algorithms. Let $\hat{g}$ and $\hat{h}$, be generators of $\mathbb{G}_p$, and $\mathbb{G}_q$, respectively. The public key contains $(N, \hat{g}, \hat{h}, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell],j\in[n]})$, where each $g_j = g^{a_j}$, for a random element $a_j \xleftarrow{\$} \mathbb{Z}_N$, and $h_{i,j} = g_j^{s_i}$, where $s_i \xleftarrow{\$} \mathbb{Z}_N$ if $i \in I$, or it is sampled randomly from $\mathbb{G}$, otherwise. Here, the secret key is $\mathsf{sk} = \{s_i\}_{i\in I}$. The public key has the same structure in the ideal mode, except that we set each $h_{i,j}$ to be equal to $h_i^{\delta_{i,j}} \cdot g_j^{s_i}$, where $h_i \xleftarrow{\$} \mathbb{G}_q$, and $s_i \xleftarrow{\$} \mathbb{Z}_N$, if $i \notin I$. The secret key in this case is $\mathsf{sk} = (I, \{s_i\}_{i\in[\ell]}, \{a_j\}_{j\in[n]}, \{h_i\}_{i\notin I})$. The encryption, decryption and public key randomness equivocation algorithms proceed more or less in the same way as in the DDH-based construction. The equivocation of the ciphertext randomness is achieved regarding the fact that $c_0 = \prod_j g_j^{r_j}$ is an element of the subgroup $\mathbb{G}_p$, so it only determines the randomness $\{r_i\}_{i\in[l]}$ modulo the order of the group, p. So, it is enough to find $r'_i = r_i \mod p$ such that $H(h_i^{r'_i} \cdot c_0^{s_i} \mod N^2) = c_i \oplus M'_i$ for all $i \notin I$. Since H is a universal hash function and r'_i is considered modulo p, a similar argument as in the DDH case shows that this task terminates in polynomial time with all but negligible probability.

2 Preliminaries

We use λ to denote the security parameter. For a natural integer $n \in \mathbb{N}$, the set $\{1, 2, \cdots, n\}$ is denoted by $[n]$. For $q \in \mathbb{N}$, we consider the rounding function $\lfloor \cdot \rceil_2 : \mathbb{Z}_q \to \mathbb{Z}_2$ to be $\lfloor x \rceil_2 = \lfloor x \cdot 2/q \rceil (\mod 2)$ for $x \in \mathbb{Z}_q$. We use bold lowercase letters (e.g., $\mathbf{v}$) to denote vectors and bold uppercase letters (e.g., $\mathbf{V}$) to denote matrices. For a vector $\mathbf{v} = (v_1, \cdots, v_n)$, the vector $(g^{v_1}. \cdots, g^{v_n})$ is denoted by $g^{\mathbf{v}}$. We write $\mathsf{poly}(\lambda)$ to denote an arbitrary polynomial function. We denote by $\mathsf{negl}(\lambda)$ a negligible function in λ, and PPT stands for probabilistic polynomial-time. Two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ are computationally (*resp.* statistically) indistinguishable if no PPT (*resp.* unbounded) adversary can distinguish them, and we write $\mathcal{D}_1 \approx_c \mathcal{D}_2$ (*resp.* $\mathcal{D}_1 \approx_s \mathcal{D}_2$). We write $\Delta(\mathcal{D}_1, \mathcal{D}_2)$ to denote the statistical distance between the distributions $\mathcal{D}_1$ and $\mathcal{D}_2$. For a finite set S, we write $x \xleftarrow{\$} S$ to denote that x is sampled uniformly at random from S. If $\mathcal{D}$ is a distribution over the set S, we write $x \leftarrow \mathcal{D}$ to denote that x is sampled from S according to $\mathcal{D}$. For an algorithm $\mathcal{A}$, we denote by $y \leftarrow \mathcal{A}(x)$ the output y after running $\mathcal{A}$ on input x. We use discrete Gaussian distributions, defined as follows.

Definition 1 ((Univariate) Discrete Gaussian Distribution). *For a real $\sigma > 0$, the discrete Gaussian distribution with variance σ is denoted by D_σ that is a probability distribution with support $\mathbb{Z}$ that assigns to each $x \in \mathbb{Z}$ a probability proportional to* $\exp(-\pi x^2/\sigma^2)$.

Definition 2. *Let Λ be a full-rank n-dimensional lattice. Let $\mathbf{\Sigma} \in \mathbb{R}^{n\times n}$ be a symmetric definite positive matrix, and $\mathbf{r}', \mathbf{c} \in \mathbb{R}^n$. $D_{\Lambda+\mathbf{r}',\mathbf{\Sigma},\mathbf{c}}$ denotes the discrete Gaussian distribution with support $\Lambda+\mathbf{r}'$, covariance $\mathbf{\Sigma}$, and mean $\mathbf{c}$ that assigns to each $\mathbf{x} \in \Lambda + \mathbf{r}'$ a probability proportional to* $\exp(-\pi(\mathbf{x}-\mathbf{c})^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{c}))$.

We also restate the definition of universal hash functions and the Leftover Hash Lemma:

Definition 3 (Universal Hash Function). *A family of hash functions* $\mathcal{H} = \{H : \mathcal{X} \to \{0,1\}^\ell\}$ *is universal if for all* $x_1, x_2 \in \mathcal{X}$*, where* $x_1 \neq x_2$*, we have* $\Pr[H \xleftarrow{\$} \mathcal{H}, H(x_1) = H(x_2)] \leq 1/2^\ell$.

Lemma 1 (Leftover Hash Lemma, [HILL99]). *Let* $\mathcal{H} = \{H : \mathcal{X} \to \{0,1\}^\ell\}$ *be a universal family of hash functions. Let* $\epsilon > 0$ *and* $\mathcal{D}$ *be a distribution over* $\mathcal{X}$ *with min-entropy* $H_\infty(\mathcal{D}) \geq \ell + 2\log(1/\epsilon)$*. Then*

$$\Delta((H, H(x)), (H, U)) = \epsilon \ ,$$

where $H \xleftarrow{\$} \mathcal{H}$*,* $x \xleftarrow{\$} \mathcal{D}$*, and* $U \xleftarrow{\$} \{0,1\}^\ell$.

2.1 Reminders on Standard Assumptions

We now provide a brief reminder on standard hardness assumptions and classical lattice results used throughout our work.

Lattices

Definition 4 (Learning With Errors assumption, [Reg05]). *Let* λ *be the security parameter,* $k, q \in \mathbb{Z}$*, and let* χ *be an an efficiently samplable distribution over* $\mathbb{Z}_q$*. The* $\mathsf{LWE}_{k,q,\chi}$ *assumption holds if for any polynomial* $n = n(\lambda)$ *we have*

$$(\mathbf{A}, \mathbf{As} + \mathbf{e}) \approx_c (\mathbf{A}, \mathbf{u}) \ ,$$

where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$*,* $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^k$*,* $\mathbf{e} \xleftarrow{\$} \chi^n$*, and* $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$.

In the above definition, if we set the error distribution χ to be a discrete Gaussian distribution $D_{\alpha.q}$, where $\alpha \in (\sqrt{k}/q, 1)$, then LWE is at least as hard as standard worst-case lattice problems ([Reg05,BLP+13]).

We also use the following lemma about lattice trapdoors.

Lemma 2 ([MP12]). *There exists a pair of PPT algorithms* $(\mathsf{TrapGen}, \mathsf{SampleD})$ *such that the* $\mathsf{TrapGen}$ *algorithm on receiving the security parameter* λ *and* $n, k', q \in \mathbb{Z}$ *as input, outputs a matrix* $\mathbf{B} \in \mathbb{Z}_q^{n \times k'}$ *and a trapdoor* td*, where* $\mathbf{B}$ *is* $2^{-k'}$ *close to uniform. Also, the* $\mathsf{SampleD}$ *algorithm takes as input a trapdoor* td*, a matrix* $\mathbf{B}$ *and a vector* $\mathbf{y} \in \mathbb{Z}_q^{k'}$ *and outputs* $\mathbf{r} \in \mathbb{Z}_q^n$ *such that* $\mathbf{r} \xleftarrow{\$} D_{\Lambda_{\mathbf{y}}^\perp(\mathbf{B}^T),\sigma}$.

We also need the following result about sampling from lattice Gaussians.

Lemma 3 ([BLP+13], Lemma 2.3). *There exists a PPT algorithm that, given a basis* $(b_i)_{i \leq k}$ *of a full-rank lattice* Λ*, vectors* $\mathbf{r}', \mathbf{c} \in \mathbb{R}^n$*, and a symmetric positive definite matrix* $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ *such that* $\Omega(\sqrt{\log n}).\max_i ||\mathbf{\Sigma}^{-1/2} \cdot b_i|| \leq 1$*, returns a sample from* $D_{\Lambda + \mathbf{r}', \mathbf{\Sigma}, \mathbf{c}}$.

We finally recall the following lemma which plays a central role in our LWE-based PEPE construction for resampling randomness and noise.

Lemma 4 ([LSSS17], Lemma 11). *Let* $\mathbf{B} \in \mathbb{Z}_q^{n \times k'}$ *and* $\mathbf{E} \in \mathbb{Z}^{n \times \ell'}$*. Sample* $(\mathbf{r}, \mathbf{e}) \leftarrow D_\sigma^n \times D_{\sigma'}^{\ell'}$ *and define* $(\mathbf{u}, \mathbf{f}) = (\mathbf{r}^T\mathbf{B}, \mathbf{r}^T\mathbf{E} + \mathbf{e}^T) \in \mathbb{Z}_q^{k'} \times \mathbb{Z}^{\ell'}$*. The conditional distribution of* $\mathbf{r}$ *given* $(\mathbf{u}, \mathbf{f})$ *is* $D^{S_{\mathbf{E},\mathbf{u},\mathbf{f}}}_{\Lambda^\perp(\mathbf{B}^T)+\mathbf{r}', \sqrt{\mathbf{\Sigma}}, \mathbf{c}}$ *with support*

$$S_{\mathbf{E},\mathbf{u},\mathbf{f}} = \{\bar{\mathbf{r}} \in \Lambda^\perp(\mathbf{B}^T) + \mathbf{r}' : \bar{\mathbf{r}} \in \mathbb{Z}_q^n, (\mathbf{f} - \bar{\mathbf{r}}^T\mathbf{E}) \in \mathbb{Z}_q^{\ell'}\} \ ,$$

where $\mathbf{r}'$ *is any solution to* $\mathbf{r}'^T\mathbf{B} = u$*, and*

$$\mathbf{\Sigma} = \sigma^2\sigma'^2 \cdot (\sigma^2 \cdot \mathbf{E}^T \cdot \mathbf{E} + \sigma'^2 \cdot \mathbf{I}_n)^{-1}, \quad \mathbf{c} = \sigma^2 \cdot (\sigma^2 \cdot \mathbf{E}^T \cdot \mathbf{E} + \sigma'^2 \cdot \mathbf{I}_n)^{-1} \cdot \mathbf{E}^T \cdot \mathbf{f} \ .$$

Decisional Diffie-Hellman

Let $\mathcal{G}$ be polynomial-time a group generator that takes the security parameter λ as input and outputs $(\mathbb{G}, p, g)$, where $\mathbb{G}$ is a multiplicative cyclic group of prime order p and g is a generator of the group.

Definition 5 (Decisional Diffie-Hellman problem). *Let* λ *be the security parameter. We say that the decisional Diffie-Hellman problem is hard relative to* $\mathcal{G}$ *if*

$$(\mathbb{G}, p, g, g^a, g^b, g^{ab}) \approx_c (\mathbb{G}, p, g, g^a, g^b, g^c) \ ,$$

where $(\mathbb{G}, p, g) \xleftarrow{\$} \mathcal{G}(1^\lambda)$ *and* $(a, b, c) \xleftarrow{\$} \mathbb{Z}_p$*.*

Subgroup Decision Assumption

Here we provide some reminders on the subgroup decision assumption over composite-order groups. The following definition is derived from the one of Boneh et al. [BGN05]. An important difference is that we do *not* require bilinear groups, which allows us to reveal generators of both subgroups.

Let $\mathcal{G}$ be a polynomial-time algorithm that takes the security parameter λ as input and outputs a tuple $(p, q, \mathbb{G}, g)$, where p and q are λ-bit prime numbers, and $\mathbb{G}$ is (multiplicative) cyclic group of order $N = pq$, generated by g. In particular, we have $\mathbb{G} \approx \mathbb{G}_q \times \mathbb{G}_p$ where $\mathbb{G}_p = \{x^q \mod N : x \in \mathbb{G}\}$ is the subgroup of $\mathbb{G}$ of order p and $\mathbb{G}_q = \{x^p \mod N : x \in \mathbb{G}\}$ is the subgroup of $\mathbb{G}$ of order q. Note also that g^p (resp. g^q) is a generator of $\mathbb{G}_q$ (resp. $\mathbb{G}_p$).

Definition 6 (Subgroup Decision, [BGN05]). *Let* λ *be the security parameter. We say that the subgroup decision assumption is hard relative to* $\mathcal{G}$ *if*

$$(N, \mathbb{G}, g_p, g_q, x^q) \approx_c (N, \mathbb{G}, g_p, g_q, x) \ ,$$

where $(p, q, \mathbb{G}, g) \xleftarrow{\$} \mathcal{G}(1^\lambda)$*,* $N = pq$*,* g_p *(resp.* g_q*) is a random element of* $\mathbb{G}_p$ *(resp.* $\mathbb{G}_q$*), and* x *is a random element of* $\mathbb{G}$*.*

Note that a cyclic group of composite order N can be obtained by considering the subgroup of a-th residues in $\mathbb{Z}_P^*$, where $a \in \mathbb{N}$ is chosen so that $P = a \cdot N + 1$ is prime. We insist that we use the assumption in groups without a pairing.

2.2 Non-Committing Encryption

Non-committing encryption (NCE) was first introduced in [CFGN96] as a central primitive for constructing adaptively-secure multi-party computation. Here we recall the formal definition of an NCE scheme.

Definition 7 (Non-Committing Encryption). *Let a security parameter λ. A Non-Committing Encryption (NCE) scheme consists of five PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that:*

- $\mathsf{KeyGen}(1^\lambda, r_G) \to (\mathsf{pk}, \mathsf{sk})$*: On input the security parameter λ and a randomness r_G, the key-generation algorithm outputs a public key* pk *and a secret key* sk*.*

- $\mathsf{Enc}(\mathsf{pk}, M, r_E) \to \mathsf{ct}$*: On input the public key* pk*, a message M, and some randomness r_E, the encryption algorithm outputs a ciphertext* ct*.*

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to M/\perp$*: On input the secret key* sk *and a ciphertext* ct*, the decryption algorithm outputs a message M or returns $\perp$.*

- $\mathsf{Sim}_1(1^\lambda) \to (\mathsf{pk}, \mathsf{ct}, \mathsf{st})$*: On input of the security parameter λ, the* Sim_1 *algorithm outputs a simulated public key* pk *and a ciphertext* ct *together with an internal state* st*.*

- $\mathsf{Sim}_2(M, \mathsf{st}) \to (r_G, r_E)$*: On input a message M and an internal state* st*, the* Sim_2 *algorithm outputs a key-generation and encryption randomness pair (r_G, r_E).*

  We require an NCE scheme to satisfy the following properties:

- ***Correctness.*** *For any message $M \in \{0,1\}^l$,*

$$\Pr\left[M \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) : \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, M) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda) \ .$$

- ***Simulatability.*** *The following two distributions should be computationally indistinguishable:*

$$\mathsf{REAL} = \left\{ (M, \mathsf{pk}, \mathsf{ct}, r_G, r_E) : \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, r_G) \\ M \leftarrow \mathcal{A}(\mathsf{pk}) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, M, r_E) \end{array} \right\} \ ,$$

  and

$$\mathsf{IDEAL} = \left\{ (M, \mathsf{pk}, \mathsf{ct}, r_G, r_E) : \begin{array}{l} (\mathsf{pk}, \mathsf{ct}, \mathsf{st}) \leftarrow \mathsf{Sim}_1(1^\lambda) \\ M \leftarrow \mathcal{A}(\mathsf{pk}) \\ (r_G, r_E) \leftarrow \mathsf{Sim}_2(M, \mathsf{st}) \end{array} \right\} \ ,$$

  where $\mathcal{A}$ denotes any PPT adversary.

Next, we recall the definition of PEPE, introduced in [BBD+20] by Brakerski *et al.*, which is proven to imply non-committing encryption.

Definition 8 (PEPE). *Let λ be a security parameter and $\{0,1\}^\lambda$ be the message space. A Packed Encryption with Partial Equivocality scheme consists of five PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{EquivPK}, \mathsf{EquivCT})$ *such that:*

- $\mathsf{KeyGen}(1^\lambda, b \in \{0,1\}, I, r_G) \to (\mathsf{pk}, \mathsf{sk})$*: On input the security parameter λ, a bit b, a subset of indices $I \subset [\ell]$, and a key generation randomness r_G, the key-generation algorithm outputs a public key* pk *and a secret key* sk*. If $b = 0$, we say that the keys are generated in the real mode. Otherwise, we say that the keys are generated in the ideal mode.*

- $\mathsf{Enc}(\mathsf{pk}, M \in \{0,1\}^\ell, r_E) \to \mathsf{ct}$*: On input the public key* pk*, a message $M \in \{0,1\}^\ell$ and an encryption randomness, the encryption algorithm outputs a ciphertext* ct*.*

- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to (M_i)_{i \in I}$*: On input the secret key* sk *and a ciphertext* ct*, the decryption algorithm outputs bits M_i for $i \in I$.*

- $\mathsf{EquivPK}(\mathsf{sk}, b, I, r_G, I') \to r'_G$*: On input the secret key* sk*, a bit b, subsets $I, I' \subset [n]$ and a (key-generation) randomness r_G, the public-key-equivocation algorithm outputs a randomness r'_G.*

- $\mathsf{EquivCT}(\mathsf{sk}, (M, r_E), \{M'_i\}_{i \notin I}) \to r'_E$*: On input the secret key* sk*, a pair of message and randomness (M, r_E), and some bits $\{M'_i\}_{i \notin I}$, the ciphertext-equivocation algorithm outputs a randomness r'_E.*

  We require a PEPE scheme to satisfy the following properties:

- ***Correctness.*** *For any message $M \in \{0,1\}^l$ and any subset $I \subset [l]$ we have*

$$\Pr\left[\{M_i\}_{i \in I} = \{M'_i\}_{i \in I} : \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 0, I, r_G) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, M, r_E) \\ (M'_i)_{i \in I} \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

- ***Public key randomness indistinguishability.*** *The random coins output by the* $\mathsf{EquivPK}$ *algorithm should be computationally indistinguishable from true random coins. Meaning that the two following joint distributions should be computationally indistinguishable:*

$$\mathsf{REAL}_{\mathsf{pk}} = \left\{(r_G, \mathsf{pk}) : \quad (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 0, I, r_G)\right\},$$

  and

$$\mathsf{IDEAL}_{\mathsf{pk}_b} = \left\{(r_G, \mathsf{pk}) : \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, b, I', r'_G) \\ r_G \leftarrow \mathsf{EquivPK}(\mathsf{sk}, b, (I', r'_G), I) \end{array}\right\},$$

for any subsets $I, I' \subset [\ell]$ such that $I \subset I'$ for any $b \in \{0,1\}$. This property is slightly different from the original one in [BBD+20], where the indistinguishability of the single variable r_G in the two cases is required. Here, we consider the joint distribution of (pk, r_G) instead, to impose that public key should remain the same after the randomness is equivocated. In other words, in the $\mathsf{IDEAL}_{\mathsf{pk}_b}$ case, if we run $\mathsf{KeyGen}(1^\lambda, 0, r_G) \to (\mathsf{pk}', \mathsf{sk}')$, then it should hold that $\mathsf{pk}' = \mathsf{pk}$. Although not mentioned in the original work [BBD+20], this extra requirement is necessary in order for the transformation of PEPE to NCE, to be correct.

- ***Ciphertext randomness indistinguishability.*** *The random coins output by the* $\mathsf{EquivCT}$ *algorithm should be statistically close to true random coins. Meaning that for any subset $I \subset [\ell]$ and any message $M' \in \{0,1\}^\ell$, the following two distributions should be statistically indistinguishable:*

$$\mathsf{IDEAL}_{\mathsf{ct}} = \left\{ (\mathsf{pk}, M, r_E) : \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1, I, r_G) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, M', r'_E) \\ M \leftarrow \mathcal{A}(\mathsf{pk}) \\ r_E \leftarrow \mathsf{EquivCT}(\mathsf{sk}, (M', r'_E), \{M_i\}_{i \notin I}) \end{array} \right\},$$

and

$$\mathsf{REAL}_{\mathsf{ct}} = \left\{ (\mathsf{pk}, M, r_E) : \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 0, I, r_G) \\ M \leftarrow \mathcal{A}(\mathsf{pk}) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, M, r_E) \end{array} \right\},$$

where $\mathcal{A}$ is an unbounded adversary which outputs M such that $M_i = M'_i$ for all $i \in I$.

PEPE implies NCE via an efficient transform incurring only a constant loss in the rate, thus we focus on building more efficient *rate*-1 PEPE schemes.

3 PEPE Constructions

We now present our different rate-1 PEPE constructions. As already mentioned, we propose the first construction from LWE with polynomial modulus, a DDH-based construction with strict polynomial running time, and the first PEPE construction from the subgroup decision assumptions. Proofs are detailed in Appendix A.

3.1 PEPE from LWE

We present a construction for PEPE from the LWE assumption. First, we recall the post-processing technique of ciphertext shrinking introduced in [BDGM19] which is used in our scheme.

Ciphertext Shrinking Algorithm for LWE-based Encryption Schemes.

Construction 1. Consider a public-key encryption scheme with ciphertexts of the form $(\mathbf{c}_1, w_{2,1}. \cdots, w_{2,\ell}) \in \mathbb{Z}_q^k \times \mathbb{Z}_q^\ell$ and secret key $\mathbf{S} \in \mathbb{Z}_q^{k\times\ell}$, where decryption is computed by $\lfloor (w_{2,1}, \cdots, w_{2,\ell}) - \mathbf{c}_1\mathbf{S} \rceil_2 = \lfloor M + e \rceil_2$ where e is sampled from a B-bounded distribution. We describe the shrinking algorithms in detail:

- $\mathsf{Shrink}\,(\mathsf{pk}, (\mathbf{c}_1, w_{2,1}. \cdots, w_{2,\ell}))$:
  - Choose $z \xleftarrow{\$} \mathbb{Z}_q \setminus U$, where
    $$U = \bigcup_{i=1}^{\ell} \left(\left[-\tfrac{q}{4} - w_{2,i} - B, -\tfrac{q}{4} - w_{2,i} + B \right] \cup \left[\tfrac{q}{4} - w_{2,i} - B, \tfrac{q}{4} - w_{2,i} + B \right] \right)$$
  - For all $i \in [\ell]$, compute $c_{2,i} = \lfloor w_{2,i} + z \rceil_2 \in \mathbb{Z}_2$.
  - Output $\mathsf{ct} = (\mathbf{c}_1, c_{2,1}, \cdots, c_{2,\ell}, z)$.
- $\mathsf{ShrinkDec}(\mathsf{sk} = \mathbf{S}, ct)$:
  - Parse ct as $(\mathbf{c}_1, c_{2,1}, \cdots, c_{2,\ell}, z)$.
  - Compute $M_i \leftarrow (c_{2,i} - \lfloor \mathbf{c}_1\mathbf{s}_i + z \rceil_2) \mod 2$ where $\mathbf{s}_i$ is the i-th row of $\mathbf{S}$.
  - Output $M = (M_1, \cdots, M_\ell)$.

  Note that we can give a subset $I \subset [\ell]$ as input to the $\mathsf{ShrinkDec}$ algorithm and only receive $\{M_i\}_{i\in I}$ as output. In [BDGM19], the authors prove that if $q > 4\ell B$, then this construction is correct.

We now detail our LWE-based PEPE scheme.

Construction 2. Let $(\mathsf{Shrink}, \mathsf{ShrinkDec})$ be the shrinking pair of algorithms described in Construction 1, and $\mathsf{TrapGen}$ the PPT algorithm for generating lattice trapdoors as proposed in [MP12].

- $\mathsf{KeyGen}(1^\lambda, b \in \{0,1\}, I, r_G)$:
  - if $b = 0$ do:
    - Choose $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times k}$.
    - For all $i \in I$, set $\mathbf{v}_i = \mathbf{A}\mathbf{s}_i + \mathbf{e}_i$, where $\mathbf{s}_i \xleftarrow{\$} \mathbb{Z}_q^k$, and $\mathbf{e}_i \xleftarrow{\$} D_\sigma^n$.
    - For all $i \notin I$, set $\mathbf{v}_i \xleftarrow{\$} \mathbb{Z}_q^n$.
    - Set $\mathsf{pk} = (\mathbf{A}, \{\mathbf{v}_i\}_{i\in[\ell]})$, and $\mathsf{sk} = (I, \{\mathbf{s}_i\}_{i\in I})$.
    - Set $r_G = (I, \mathbf{A}, \{\mathbf{s}_i\}_{i\in I}, \{\mathbf{e}_i\}_{i\in I}, \{\mathbf{v}_i\}_{i\notin I})$.
  - if $b = 1$ do:
    - Run $(\mathbf{B}, \mathsf{td}_\mathbf{B}) \leftarrow \mathsf{TrapGen}(1^\lambda, n, k + l - |I|, q)$ and parse $\mathbf{B}$ as $\mathbf{B} = \left[\mathbf{A} \middle| \mathbf{V}\right] \in \mathbb{Z}_q^{n\times(k+\ell-|I|)}$.
    - For all $i \in I$, set $\mathbf{v}_i = \mathbf{A}\mathbf{s}_i + \mathbf{e}_i$, where $\mathbf{s_i} \xleftarrow{\$} \mathbb{Z}_q^k$, and $\mathbf{e}_i \xleftarrow{\$} D_\sigma^n$.
    - For all $i \notin I$, set $\mathbf{v}_i := \mathbf{V}_i$, where $\mathbf{V}_i$ is the i-th column of matrix $\mathbf{V}$.
    - Set $\mathsf{pk} = (\mathbf{A}, \{\mathbf{v}_i\}_{i\in[\ell]})$, and $\mathsf{sk} = (I, \{\mathbf{s}_i\}_{i\in I}, \mathsf{td}_\mathbf{B})$.
    - Set $r_G = (I, \mathbf{A}, \{\mathbf{s}_i\}_{i\in I}, \{\mathbf{e}_i\}_{i\in I}, \{\mathbf{v}_i\}_{i\notin I})$.
- $\mathsf{Enc}(\mathsf{pk}, M \in \{0,1\}^\ell, r_E)$:
  - Parse $\mathsf{pk} = (\mathbf{A}, \{\mathbf{v}_i\}_{i\in[\ell]})$.

- Sample $\mathbf{r} \xleftarrow{\$} D_\sigma^n$.
- Compute $\mathbf{c}_1 \leftarrow \mathbf{r}^T\mathbf{A}$ and $w_{2,i} = \mathbf{r}^T\mathbf{v}_i + e_i + \lfloor q/2 \rceil \cdot M_i \in \mathbb{Z}_q, \forall i \in [\ell]$, where $e_i \xleftarrow{\$} D_{\sigma'}$.
- Compress $(\mathbf{c}_1, w_{2,1}, \cdots, w_{2,\ell})$ into

$$ct = (\mathbf{c}_1, c_{2,1}, \cdots, c_{2,\ell}, z) \leftarrow \mathsf{Shrink}(\mathsf{pk}, (\mathbf{c}_1, w_{2,1}, \cdots, w_{2,\ell})).$$

- Set $r_E = (\mathbf{r}, \{e_i\}_{i\in[\ell]}, z)$.
- Output ct.

- $\mathsf{Dec}(\mathsf{sk}, ct)$:
  - Use sk to run $\mathsf{ShrinkDec}(\mathsf{sk}, ct)$ and recover M.
  - Output $\{M_i\}_{i\in I}$.
- $\mathsf{EquivPK}(\mathsf{sk}, b, I, r_G, I')$:
  - If $I' \not\subseteq I$ abort.
  - Let $\mathsf{pk} = (\mathbf{A}, \{\mathbf{v}_i\}_{i\in[\ell]})$ be the output of KeyGen on input $(1^\lambda, b, I, r_G)$.
  - Set $r'_G = (I', \mathbf{A}, \{\mathbf{s}_i\}_{i\in I'}, \{\mathbf{e}_i\}_{i\in I'}, \{\mathbf{v}_i\}_{i\notin I'})$.
  - Output r'_G.
- $\mathsf{EquivCT}(\mathsf{sk}, (M, r_E), \{M'_i\}_{i\notin I})$:
  - Parse $\mathsf{sk} = (I, \{\mathbf{s}_i\}_{i\in I}, \mathsf{td}_\mathbf{B})$, $r_G = (I, \mathbf{A}, \{\mathbf{s}_i\}_{i\in I}, \{\mathbf{e}_i\}_{i\in I}, \{\mathbf{v}_i\}_{i\notin I})$. Let $(\mathbf{c}_1, w_{2,1}, \cdots, w_{2,\ell})$ be the LWE encryption of M w.r.t pk and $r_E = (\mathbf{r}, \{e^*_i\}_{i\in[\ell]}, z)$.
  - For $i \notin I$, sample $e'_i \xleftarrow{\$} D_{\sigma'}$.
  - Using $\mathsf{td}_\mathbf{B}$, sample $\bar{\mathbf{r}} \xleftarrow{\$} D^{S_{\mathbf{E},\mathbf{u},\mathbf{f}}}_{\Lambda^\perp(\mathbf{B}^T)+\mathbf{r}', \sqrt{\mathbf{\Sigma}}, \mathbf{c}}$, where

$$\begin{aligned}
\mathbf{E} &= \left[\mathbf{e}_{i_1} | \cdots | \mathbf{e}_{i_{|I|}}\right] \text{ for } i_1, \cdots, i_{|I|} \in I,\\
\mathbf{u} &= (\mathbf{c}_1, \{w_{2,i} - \lfloor q/2 \rceil M'_i - e'_i\}_{i\notin I}),\\
\mathbf{f} &= \mathbf{r}^T\mathbf{E} + \mathbf{e}^*, \text{ where } \mathbf{e}^* = (e^*_{i_1}, \cdots, e^*_{i_{|I|}}) \text{ for } i_1, \cdots, i_{|I|} \in I,\\
\mathbf{\Sigma} &= \sigma^2\sigma'^2 \cdot (\sigma^2 \cdot \mathbf{E}^T \cdot \mathbf{E} + \sigma'^2 \cdot \mathbf{I}_n)^{-1},\\
\mathbf{c} &= \sigma^2 \cdot (\sigma^2 \cdot \mathbf{E}^T \cdot \mathbf{E} + \sigma'^2 \cdot \mathbf{I}_n)^{-1} \cdot \mathbf{E}^T \cdot \mathbf{f}.
\end{aligned}$$

    Regarding Lemma 4 the output of the sampling would be a vector $\bar{\mathbf{r}}$ satisfying $\bar{\mathbf{r}}^T\mathbf{B} = \mathbf{u}$.
  - For $i \in I$, set $e'_i = e^*_i + (\mathbf{r} - \bar{\mathbf{r}})^T\mathbf{e}_i$.
  - Output $r'_E = (\bar{\mathbf{r}}, \{e'_i\}_{i\in[\ell]}, z)$.

3.2 PEPE from DDH

In this section, we detail our DDH-based construction that deviates from prior constructions by using a universal hash functions. Doing so, we obtain a simpler construction that additional features an encryption algorithm running in strict polynomial time.

Construction 3. Let $n \in \mathbb{N}$ and H be a universal hash function from $\mathbb{G}$ to $\{0, 1\}$.

- $\mathsf{KeyGen}(1^\lambda, b \in \{0,1\}, I, r_G)$:
  - Run $(\mathbb{G}, g, p) \leftarrow \mathcal{G}(1^\lambda)$.
  - if $b = 0$ do:
    - For all $j \in [n]$, choose $g_j \xleftarrow{\$} \mathbb{G}$.
    - For all $i \in I$ and $j \in [n]$, set $h_{i,j} = g_j^{s_i}$, for $s_i \xleftarrow{\$} \mathbb{Z}_p$.
    - For all $i \notin I$ and $j \in [n]$, choose $h_{i,j} \xleftarrow{\$} \mathbb{G}$.
    - Set $\mathsf{pk} = (\mathbb{G}, g, p, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell], j\in[n]})$, and $\mathsf{sk} = (I, \{s_i\}_{i\in I})$.
    - Set $r_G = (\mathbb{G}, g, p, \{g_j\}_{j\in[n]}, \{s_i\}_{i\in I}, \{h_{i,j}\}_{i\notin I, j\in[n]})$.
  - if $b = 1$ do:
    - For all $j \in [n]$, choose $a_j \xleftarrow{\$} \mathbb{Z}_p$ and set $g_j = g^{a_j}$.
    - For all $i \in I$ and $j \in [n]$, set $h_{i,j} = g_j^{s_i}$, for $s_i \xleftarrow{\$} \mathbb{Z}_p$.
    - For all $i \notin I$ and $j \in [n]$, set $h_{i,j} = g^{z_{i,j}}$, for $z_{i,j} \xleftarrow{\$} \mathbb{Z}_p$.
    - Define the public key as $\mathsf{pk} = \mathbb{G}, g, p, (\{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell], j\in[n]})$ while the secret key is defined to be $\mathsf{sk} = (I, \{s_i\}_{i\in I}, \{a_j\}_{j\in[n]}, \{z_{i,j}\}_{i\notin I, j\in[n]})$.
    - Set $r_G = (\mathbb{G}, g, p, \{g_j\}_{j\in[n]}, \{s_i\}_{i\in I}, \{a_j\}_{j\in[n]}, \{z_{i,j}\}_{i\notin I, j\in[n]})$.

- $\mathsf{Enc}(\mathsf{pk}, M \in \{0,1\}^\ell, r_E)$:
  - Parse $\mathsf{pk} = (g, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell], j\in[n]})$.
  - Sample $r_1, \cdots, r_n \xleftarrow{\$} \mathbb{Z}_p$.
  - Compute $c_0 = \prod_j^n g_j^{r_j}$, and for all $i \in [\ell]$ compute
$$c_i = M_i \oplus H\big(\prod_{j=1}^n h_{i,j}^{r_j}\big) \in \{0,1\}.$$
  - Set $r_E = (r_1, \cdots, r_n)$.
  - Output $ct = (c_0, c_1, \cdots, c_\ell)$.

- $\mathsf{Dec}(\mathsf{sk}, ct)$:
  - Parse $\mathsf{sk} = (I, \{s_i\}_{i\in I})$.
  - For all $i \in I$, compute $M_i = c_i \oplus H(c_0^{s_i})$.
  - Output $\{M_i\}_{i\in I}$.

- $\mathsf{EquivPK}(\mathsf{sk}, b, I, r_G, I')$:
  - If $I' \not\subseteq I$ abort.
  - Let $\mathsf{pk} = (\mathbb{G}, g, p, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell], j\in[n]})$ be the output of KeyGen on input $(1^\lambda, b, I, r_G)$.
  - Set $r'_G = (g, p, \{g_j\}_{j\in[n]}, \{s_i\}_{i\in I'}, \{h_{i,j}\}_{i\notin I', j\in[n]})$.

- $\mathsf{EquivCT}(\mathsf{sk}, (M, r_E), \{M'_i\}_{i\notin I})$:
  - Parse $\mathsf{sk} = (I, \{s_i\}_{i\in I}, \{a_j\}_{j\in[n]}, \{z_{i,j}\}_{i\notin I, j\in[n]})$. Let $(c_0, c_1, \cdots, c_\ell)$ be the encryption of M w.r.t pk and $r_E = (r_1, \cdots, r_n)$.
  - For all $i \notin I$, repeatedly sample $t_i \xleftarrow{\$} \mathbb{Z}_p$ until $H(g^{t_i}) = c_i \oplus M'_i$. If no such t_i is found after λ attempts, abort and output $\perp$.
  - Let $\mathbf{a} = (a_1, \cdots, a_n)$, $\mathbf{r} = (r_1, \cdots, r_n)$, and $\{i_1, \cdots, i_\alpha\} = [\ell] \setminus I$. Also, for all $i \notin I$, let $\mathbf{z}_i = (z_{i,1}, \cdots, z_{i,n})$. Now sample uniformly a solution $\bar{\mathbf{r}} \in \mathbb{Z}_p^n$ for

$$\begin{bmatrix} \mathbf{a} \\ \mathbf{z}_{i_1} \\ \vdots \\ \mathbf{z}_{i_\alpha} \end{bmatrix} \cdot \bar{\mathbf{r}}^T = \begin{bmatrix} \mathbf{a}\mathbf{r}^T \\ t_{i_1} \\ \vdots \\ t_{i_\alpha} \end{bmatrix}. \tag{1}$$

- Output $r'_E = \bar{\mathbf{r}}$.

3.3 PEPE from Subgroup Decision

Construction 4. Consider a group $\mathbb{G} \approx \mathbb{G}_q \times \mathbb{G}_p$ of order $N = pq$, where $\mathbb{G}_p = \{x^q \mod N : x \in \mathbb{G}\}$ is the subgroup of $\mathbb{G}$ of order p, and $\mathbb{G}_q = \{x^p \mod N : x \in \mathbb{G}\}$ is the subgroup of $\mathbb{G}$ of order q. Let $n \in \mathbb{N}$ such that $n > \ell$, and H be a universal hash function from $\mathbb{G}$ to $\{0,1\}$.

- $\mathsf{Setup}(1^\lambda, 1^n)$:
  - Sample $(p, q, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. Let $N = pq$.
  - Set $\hat{g} \leftarrow u^q$, where $u \xleftarrow{\$} \mathbb{G}$. Note that $\hat{g}$ is a random element (and a generator) of $\mathbb{G}_p$.
  - For all $j \in [n]$, choose $g_j \leftarrow \hat{g}^{a_j}$, where $a_j \xleftarrow{\$} \mathbb{Z}_N$. Note that each g_j is a random element of $\mathbb{G}_p$.
  - Set $\hat{h} \leftarrow v^p$, where $v \xleftarrow{\$} \mathbb{G}$. Note that $\hat{h}$ is a random element (and a generator) of $\mathbb{G}_q$.
  - Output $\mathsf{crs} = (N, \mathbb{G}, \hat{g}, \hat{h}, \{g_j\}_{j\in[n]})$, and $\mathsf{td} = (p, q)$.
- $\mathsf{KeyGen}(\mathsf{crs}, b \in \{0,1\}, I, r_G)$:
  - Parse $\mathsf{crs} = (N, \mathbb{G}, \hat{g}, \hat{h}, \{g_j\}_{j\in[n]})$.
  - if $b = 0$:
    - For all $i \in I$ and $j \in [n]$, set $h_{i,j} = g_j^{s_i} \in \mathbb{G}$, where $s_i \xleftarrow{\$} \mathbb{Z}_N$.
    - For all $i \notin I$ and $j \in [n]$, sample obliviously $h_{i,j} \xleftarrow{\$} \mathbb{G}$.
    - Set $\mathsf{sk} = (I, \{s_i\}_{i\in I})$, and $r_G = (N, \mathbb{G}, \hat{g}, \hat{h}, \{s_i\}_{i\in I}, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\notin I, j\in[n]})$.
  - if $b = 1$:
    - For all $i \in I$ and $j \in [n]$, set $h_{i,j} = g_j^{s_i} \in \mathbb{G}$, where $s_i \xleftarrow{\$} \mathbb{Z}_N$.
    - For all $i \notin I$ and $j \in [n]$, choose $h_i \leftarrow \hat{h}^{b_i}$, where $b_i \xleftarrow{\$} \mathbb{Z}_N$, and set $h_{i,j} = h_i^{\delta_{i,j}} \cdot g_j^{s_i} \in \mathbb{G}$, where $s_i \xleftarrow{\$} \mathbb{Z}_N$, and $\delta_{i,j}$ is the Kronecker delta.
    - Set $\mathsf{sk} = (I, \{s_i\}_{i\in[\ell]}, \{h_i\}_{i\notin I})$, and $r_G = (N, \mathbb{G}, \hat{g}, \hat{h}, \{s_i\}_{i\in[\ell]}, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\notin I, j\in[n]})$.
  - Set $\mathsf{pk} = (N, \mathbb{G}, \hat{g}, \hat{h}, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell], j\in[n]})$.
- $\mathsf{Enc}(\mathsf{pk}, M \in \{0,1\}^\ell, r_E)$:
  - Parse $\mathsf{pk} = (N, \mathbb{G}, \hat{g}, \hat{h}, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell], j\in[n]})$.

- Sample $r_1, \cdots, r_n \xleftarrow{\$} \mathbb{Z}_N$ and compute $c_0 = \prod_{j=1}^{n} g_j^{r_j} \in \mathbb{G}$. Then, for each $i \in [\ell]$, compute $c_i = M_i \oplus H(\prod_{j=1}^{n} h_{i,j}^{r_j}) \in \{0,1\}$.
- Set $r_E = (r_1, \cdots, r_n)$.
- Output $ct = (c_0, c_1, \cdots, c_\ell)$.

- $\mathsf{Dec}(\mathsf{sk}, ct)$:
  - Parse $\mathsf{sk} = (I, \{s_i\}_{i \in I})$.
  - For all $i \in I$, compute $M_i = c_i \oplus H(c_0^{s_i})$.
  - Output $\{M_i\}_{i \in I}$.
- $\mathsf{EquivPK}(\mathsf{sk}, \mathsf{td}, b, I, r_G, I')$:
  - If $I' \not\subseteq I$ abort.
  - Let $\mathsf{pk} = (N, \mathbb{G}, \hat{g}, \hat{h}, \{g_j\}_{j \in [n]}, \{h_{i,j}\}_{i \in [\ell], j \in [n]})$ be the output of KeyGen on input $(\mathsf{crs}, b, I, r_G)$.
  - Set $r'_G = (N, \mathbb{G}, \hat{g}, \hat{h}, \{a_j\}_{j \in [n]}, \{s_i\}_{i \in I'}, \{h_{i,j}\}_{i \notin I', j \in [n]})$.
  - Output r'_G.
- $\mathsf{EquivCT}(\mathsf{sk}, \mathsf{td}, (M, r_E), \{M'_i\}_{i \notin I})$:
  - Parse $\mathsf{sk} = (I, \{a_j\}_{j \in [n]}, \{s_i\}_{i \in [\ell]}, \{h_i\}_{i \notin I})$, and $\mathsf{td} = (p, q)$. Let $(c_0, c_1, \cdots, c_\ell)$ be the encryption of M w.r.t pk and $r_E = (r_1, \cdots, r_n)$.
  - For all $i \notin I$, repeatedly sample $r'_i \xleftarrow{\$} \mathbb{Z}_N$ such that $r'_i = r_i \mod p$ and $r'_i \neq r_i \mod q$ until

$$H(h_i^{r_i} \cdot c_0^{s_i}) = c_i \oplus M'_i. \tag{2}$$

    If no candidate is found after λ attempts, abort and output $\perp$.
  - For all $i \in I$, set $r'_i \leftarrow r_i$.
  - Output $r'_E = (r'_1, \cdots, r'_n)$.

On oblivious sampling in SD construction. Here, we explain how oblivious sampling can be done in the above construction. The construction works over the group $\mathbb{G} \approx \mathbb{G}_q \times \mathbb{G}_p$ of order $N = pq$. During the real-mode key generation, we require that some public key elements be obliviously sampled from $\mathbb{G}$. To see why it can be done, we use the sampling technique as done in [CF01].

Let $P = a \cdot N + 1$ be a prime number, where $\gcd(a, N) = 1$. Since N is public, then such P can be generated by a real-mode party. Now, since $\mathbb{G}$ is isomorphic to the subgroup of order N of $\mathbb{Z}_P^*$, it is enough to be able to obliviously sample from this subgroup. The idea for doing so is to generate a random element in $\mathbb{Z}_P^*$, and then raise it to the a-th power. To generate random elements in $\mathbb{Z}_P^*$, for a prime P, it is enough to pick a random bit string of length $2\log(P)$, and then reduce its decimal value modulo P. It turns out that the distribution of elements sampled in this way, is statistically close to the uniform distribution over $\mathbb{G}$.

Also, as explained in [CF01], the sampling is invertible, meaning that given a random element h of the subgroup $\mathbb{G}$, we can efficiently recover an underlying random element $h_P \in \mathbb{Z}_P^*$, such that $h_P^a = h(\mod P)$. This should be used in the $\mathsf{EquivPK}$ algorithm, and can be done as follows: First, we find the inverse of a

modulo N, which exists since $\gcd(a, N) = 1$. Let g_P be a generator of $\mathbb{Z}_P^*$. Now, given an element $h \in \mathbb{G}$, we set $h_p := h^{a^{-1}} \cdot g_P^{iN} \mod P$, where i is a random element from $\mathbb{Z}_a^*$. It is easy to see that $h_p^a = h(\mod P)$. Also, it is a random element among the elements of $\mathbb{Z}_P^*$ whose a-th power is equal to h. Generating a random decimal value (thus a random bit string) who is equal to h_P modulo P is easily done by choosing a random $k \in \mathbb{Z}_p$ and computing $h_P + kP$.

References

BBD+20. Z. Brakerski, P. Branco, N. Döttling, S. Garg, and G. Malavolta. Constant ciphertext-rate non-committing encryption from standard assumptions. In *TCC*, 2020.

BDGM19. Z. Brakerski, N. Döttling, S. Garg, and G. Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In *TCC*, 2019.

BGN05. D. Boneh, E. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *In TCC*, 2005.

BLP+13. Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *STOC*, 2013.

CDSMW09. S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Improved non-committing encryption with applications to adaptively secure protocols. In *ASIACRYPT*, 2009.

CF01. R. Canetti and M. Fischlin. Universally composable commitments. In *CRYPTO*, 2001.

CFGN96. R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *STOC*, 1996.

CPR17. R. Canetti, O. Poburinnaya, and M. Raykova. Optimal-rate non-committing encryption. In *ASIACRYPT*, 2017.

GPV08. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, 2008.

HILL99. J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 1999.

HJR16. D. Hofheinz, T. Jager, and A. Rupp. Public-key encryption with simulation-based selective-opening security and compact ciphertexts. In *TCC*, 2016.

HOR15. B. Hemenway, R. Ostrovsky, and A. Rosen. Non-committing encryption from ϕ-hiding. In *TCC*, 2015.

HORR16. B. Hemenway, R. Ostrovsky, S. Richelson, and A. Rosen. Adaptive security with quasi-optimal rate. In *TCC*, 2016.

LSSS17. B. Libert, A. Sakzad, D. Stehlé, and R. Steinfeld. All-but-many lossy trapdoor functions and selective opening chosen-ciphertext security from lwe. In *CRYPTO*, 2017.

MP12. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, 2012.

Nie02. J. B. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO*, 2002.

Reg05. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, 2005.

YKT19. Y. Yoshida, F. Kitagawa, and K. Tanaka. Non-committing encryption with quasi-optimal ciphertext-rate based on the ddh problem. In *ASIACRYPT*, 2019.

YKXT20. Y. Yoshida, F. Kitagawa, K. Xagawa, and K. Tanaka. Non-committing encryption with constant ciphertext expansion from standard assumptions. In *ASIACRYPT*, 2020.

Supplementary Material

A Proof of PEPE constructions

A.1 Proof of LWE-based PEPE

Theorem 1. *Assuming LWE, Construction 2 is a rate-1 PEPE scheme.*

Proof. We now prove correctness, public key randomness indistinguishability, and ciphertext randomness indistinguishability of the above construction.

Correctness. Correctness follows from the fact that in the ciphertext shrinking technique described in Construction 1 we have:

$$(c_{2,i} - \lfloor \mathbf{c}_1 \mathbf{s}_i + z \rceil_2) \mod 2 = \lfloor w_{2,i} - \mathbf{c}_1 \mathbf{s}_i \rceil_2 \ ,$$

where $i \in I$. Also, similarly to Regev's public-key encryption scheme [Reg05], for all $i \in I$ we have

$$\begin{aligned} \lfloor w_{2,i} - \mathbf{c}_1 \mathbf{s}_i \rceil_2 &= \lfloor \mathbf{r}^T (\mathbf{A}\mathbf{s}_i + \mathbf{e}_i) + e_i + \lfloor q/2 \rceil M_i - \mathbf{r}^T \mathbf{A}\mathbf{s}_i \rceil_2 \\ &= \lfloor \mathbf{r}^T \mathbf{e}_i + e_i + \lfloor q/2 \rceil M_i \rceil_2 = M_i \ . \end{aligned}$$

Thus, the scheme is correct.

Public key randomness indistinguishability. We prove the public key randomness indistinguishability using a sequence of games. Assume that $b = 1$ in the experiment $\mathsf{IDEAL}_{\mathsf{pk}_b}$. We start with $\mathcal{H}_0$ which is the experiment $\mathsf{IDEAL}_{\mathsf{pk}_1}$, and we end up at $\mathcal{H}_2$ which is the experiment $\mathsf{REAL}_{\mathsf{pk}}$. We show that the advantage of an adversary in distinguishing between each two successive games is negligible. Hence, the distribution of the public key randomness is indistinguishable in $\mathsf{IDEAL}_{\mathsf{pk}_1}$ and in $\mathsf{REAL}_{\mathsf{pk}}$. As a reminder, the key generation randomness with respect to a subset I' is $r_G = (I, \mathbf{A}, \{\mathbf{s}_i\}_{i \in I}, \{\mathbf{e}_i\}_{i \in I}, \{\mathbf{v}_i\}_{i \notin I})$.

Hybrid $\mathcal{H}_0$. This is the experiment $\mathsf{IDEAL}_{\mathsf{pk}_1}$:

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1, I', r'_G)$, where $\mathsf{pk} = (\mathbf{A}, \{\mathbf{v}_i\}_{i \in [\ell]})$ and $\mathsf{sk} = (I', \{\mathbf{s}_i\}_{i \in I'}, \mathsf{td}_{\mathbf{B}})$.
- Run $r_G \leftarrow \mathsf{EquivPK}(\mathsf{sk}, 1, (I', r'_G), I)$.
- $b \leftarrow \mathcal{A}(r_G)$.

Hybrid $\mathcal{H}_1$. Here, we replace the matrix $\mathbf{A}$ and the vectors $\{\mathbf{v}_i\}_{i \notin I'}$ to be uniform.

- Choose $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$ and for all $i \notin I'$, choose $\mathbf{v}_i \xleftarrow{\$} \mathbb{Z}_q^n$. For $i \in I'$, do as before.
- Run $r_G \leftarrow \mathsf{EquivPK}(\mathsf{sk}, 1, (I', r'_G), I)$.
- $b \leftarrow \mathcal{A}(r_G)$.

Indistinguishability of $\mathcal{H}_0$ and $\mathcal{H}_1$ follows from Lemma 2, which guarantees that matrix $\mathbf{B}$ is 2^{-k} close to uniform. Thus, $\mathbf{A}$ and $\{\mathbf{v}_i\}_{i \notin I'}$ are also statistically close to uniform.

Hybrid $\mathcal{H}_2$. Here, we replace each vector $\{\mathbf{v}_i\}_{i \in I' \setminus I}$ by a random vector from $\mathbb{Z}_q^n$.

- Choose $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$ and for all $i \notin I'$ and $i \in I' \setminus I$, choose $\mathbf{v}_i \xleftarrow{\$} \mathbb{Z}_q^n$. For $i \in I$, set $\mathbf{v}_i$ as before.
- Run $r_G \leftarrow \mathsf{EquivPK}(\mathsf{sk}, 1, (I', r'_G), I)$.
- $b \leftarrow \mathcal{A}(r_G)$.

Indistinguishability of $\mathcal{H}_2$ and $\mathcal{H}_1$ immediately follows from LWE. Since $\mathcal{H}_2$ is identical to experiment $\mathsf{REAL}_{\mathsf{pk}}$, this concludes the proof.
Note that hybrid $\mathcal{H}_1$ is identical to experiment $\mathsf{IDEAL}_{\mathsf{pk}_0}$. So, we also proved that $\mathsf{IDEAL}_{\mathsf{pk}_0} \approx_c \mathsf{REAL}_{\mathsf{pk}}$.

Ciphertext randomness indistinguishability. For a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1, I, r_G)$, let $ct = (\mathbf{c}_1, (c_{2,1}, \cdots, c_{2,\ell})) \leftarrow \mathsf{Enc}(\mathsf{pk}, M, r_E)$ with $r_E = (\mathbf{r}, \{e_i^*\}_{i \in [\ell]}, z)$, where $\mathsf{pk} = (\mathbf{A}, \{\mathbf{v}_i\}_{i \in [\ell]})$, $\mathsf{sk} = (I, \{\mathbf{s}_i\}_{i \in I}, \mathsf{td}_{\mathbf{B}})$. Now, let a message $M' \in \{0,1\}^\ell$ such that $M_i = M'_i$ for all $i \in I$, and $M_i \neq M'_i$ otherwise. After running $\mathsf{EquivCT}((\mathsf{sk}, r_G), (M, r_E), \{M'_i\}_{i \notin I})$ we obtain

$$r'_E = (\bar{\mathbf{r}}, \{e'_i\}_{i \in [\ell]}, z).$$

Let $ct' = (\mathbf{c}'_1, (c'_{2,1}, \cdots, c'_{2,\ell}))$ be the Regev encryption of M' w.r.t pk and r'_E. We have:

- $\mathbf{c}'_1 = \mathbf{c}_1$. In the third step of $\mathsf{EquivCT}$ algorithm, $\bar{\mathbf{r}}$ is sampled such that $\bar{\mathbf{r}}^T \mathbf{B} = \mathbf{u}$, so $\bar{\mathbf{r}}$ satisfies
$$\bar{\mathbf{r}}^T \mathbf{A} = \mathbf{c}_1 = \mathbf{r}^T \mathbf{A}$$
So, $\mathbf{c}_1 = \mathbf{c}'_1$.
- For all $i \notin I$, $c'_{2,i} = c_{2,i}$. Regarding how $\bar{\mathbf{r}}$ is sampled, for all $i \notin I$ we have
$$\bar{\mathbf{r}}^T \mathbf{v}_i = w_{2,i} - \lfloor q/2 \rceil M'_i - e'_i,$$
So,
$$c'_{2,i} = \lfloor \bar{\mathbf{r}}^T \mathbf{v}_i + \lfloor q/2 \rceil M'_i + e'_i \rceil_2 = \lfloor w_{2,i} \rceil_2 = c_{2,i}.$$
- For all $i \in I$, $c'_{2,i} = c_{2,i}$. For $i \in I$ we have
$$\begin{aligned} \mathbf{r}^T \mathbf{v}_i + \lfloor q/2 \rceil M_i + e_i^* &= \mathbf{r}^T \mathbf{v}_i + \lfloor q/2 \rceil M'_i + e_i^* \\ &= \bar{\mathbf{r}}^T \mathbf{v}_i + \lfloor q/2 \rceil M'_i + e'_i. \end{aligned}$$
So, $c'_{2,i} = c_{2,i}$.

Finally, we have to prove that $\{e_i^*\} \approx_s \{e'_i\}$. For all $i \notin I$, we pick e'_i from the same distribution as e_i^*. Also, for $i \in I$, we sample $\bar{\mathbf{r}}$ from a distribution with parameters according to Lemma 4. It guarantees that the distribution

of e'_i is the same as e^*_i. Also, given a sampler for $D_{\Lambda^\perp(\mathbf{A}^T)+\mathbf{r}',\sqrt{\mathbf{\Sigma}},\mathbf{c}}$ we can get independent samples from $D^{S_{\mathbf{E},\mathbf{u},\mathbf{f}}}_{\Lambda^\perp(\mathbf{A}^T)+\mathbf{r}',\sqrt{\mathbf{\Sigma}},\mathbf{c}}$ by rejection sampling. The former exists regarding Lemma 3.

This concludes the proof of Theorem 1. □

A.2 Proof of DDH-based PEPE

Theorem 2. *Assuming DDH, Construction 3 is a rate-*1 *PEPE scheme.*

Proof. We now prove correctness, public key randomness indistinguishability, and ciphertext randomness indistinguishability of the above construction.

Correctness. Let a public key $\mathsf{pk} = (g, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell],j\in[n]})$ and let $ct = (c_0, c_1, \cdots, c_\ell)$ be an encryption of $M \in \{0,1\}^\ell$ w.r.t encryption randomness $(r_1, \cdots, r_n)$. The decryption of ct w.r.t secret key $\mathsf{sk} = (I, \{s_i\}_{i\in I})$ is computed as $M_i = c_i \oplus H(c_0^{s_i})$. The correctness follows from the fact that when $b = 0$, we have $h_{i,j} = g_j^{s_i}$, where $i \in I$ and $j \in [n]$. Thus, each c_i can be computed as $M_i \oplus H(c_0^{s_i})$. Therefore, correctness follows.

Public key randomness indistinguishability. We start with $\mathcal{H}_0$ which is the experiment $\mathsf{IDEAL}_{\mathsf{pk}_b}$ for arbitrary bit b, and we end up at $\mathcal{H}_1$ which is the experiment $\mathsf{REAL}_{\mathsf{pk}}$. We show that the advantage of an adversary in distinguishing between two games is negligible. Hence, the distribution of the public key randomness is indistinguishable in $\mathsf{IDEAL}_{\mathsf{pk}_b}$ and in $\mathsf{REAL}_{\mathsf{pk}}$.

Hybrid $\mathcal{H}_0$. This is the experiment $\mathsf{IDEAL}_{\mathsf{pk}_b}$:

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, b, I', r'_G)$, where $\mathsf{pk} = (\mathbb{G}, g, p, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell],j\in[n]})$, and $\mathsf{sk} = (I', \{s_i\}_{i\in I'}, \{h_{i,j}\}_{i\notin I',j\in[n]})$, if $b = 0$, or
  $\mathsf{sk} = (I', \{s_i\}_{i\in I'}, \{a_j\}_{j\in[n]}, \{z_{i,j}\}_{i\notin I,j\in[n]})$, if $b = 1$.
- Run $r_G \leftarrow \mathsf{EquivPK}(\mathsf{sk}, 1, (I', r'_G), I)$.
- $b \leftarrow \mathcal{A}(r_G)$.

Hybrid $\mathcal{H}_1$. In this experiment we replace the vectors $\{h_{i,j}\}_{i\in I'\setminus I}$ to be uniform.

- For all $i \in I' \setminus I$, choose $h_{i,j} \xleftarrow{\$} \mathbb{G}$.
- Set $r_G = (g, p, \{g_j\}_{j\in[n]}, \{s_i\}_{i\in I}, \{h_{i,j}\}_{i\notin I,j\in[n]})$ and send it to $\mathcal{A}$.
- $b \leftarrow \mathcal{A}(r_G)$.

Note that in $\mathcal{H}_0$, we have $h_{i,j} = g^{a_j\cdot s_i}$, for $i \in I' \setminus I$ and $j \in [n]$, while in $\mathcal{H}_1$, they are random elements of $\mathbb{G}$. If the DDH assumption holds, $\mathcal{H}_0$ and $\mathcal{H}_1$ are computationally indistinguishable. Also, hybrid $\mathcal{H}_1$ is the same as experiment $\mathsf{REAL}_{\mathsf{pk}}$, which completes the proof.

Ciphertext randomness indistinguishability. In the ideal mode, let a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, 1, I, r_G)$, where $\mathsf{pk} = (g, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell],j\in[n]})$ and $\mathsf{sk} = (I, \{\mathbf{s}_i\}_{i\in I}, \{a_j\}_{j\in[n]}, \{z_{i,j}\}_{i\notin I, j\in[n]})$.

Let $ct = (c_0, c_1, \cdots, c_\ell) \leftarrow \mathsf{Enc}(\mathsf{pk}, M, r_E)$, where $r_E = \mathbf{r} = (r_1, \cdots, r_n)$. Now let $M' \in \{0,1\}^\ell$ such that $M_i = M'_i$ for all $i \in I$, and $M_i \neq M'_i$ otherwise. After running $\mathsf{EquivCT}((sk, r_G), (M, r_E), \{M'_i\}_{i\notin I})$ we obtain

$$r'_E = \bar{\mathbf{r}} = (\bar{r}_1, \cdots, \bar{r}_n).$$

Let $ct' = (c'_0, c'_1, \cdots, c'_\ell) \leftarrow \mathsf{Enc}(\mathsf{pk}, M', r'_E)$. We have:

- $c'_0 = c_0$. In the third step of $\mathsf{EquivCT}$ algorithm, $\bar{\mathbf{r}}$ is sampled such that $\mathbf{a}\bar{\mathbf{r}}^T = \mathbf{a}\mathbf{r}^T$, so $\bar{\mathbf{r}}$ satisfies

$$c'_0 = \prod_{j=1}^{n} g_j^{\bar{r}_j} = g^{\mathbf{a}\cdot\bar{\mathbf{r}}^T} = g^{\mathbf{a}\cdot\mathbf{r}^T} = \prod_{j=1}^{n} g_j^{r_j} = c_0.$$

- For all $i \notin I$, $c'_i = c_i$. Regarding how $\bar{\mathbf{r}}$ is sampled, for all $i \notin I$ we have

$$\mathbf{z}_i \cdot \bar{\mathbf{r}}^T = t_i,$$

  where $t_i \xleftarrow{\$} \mathbb{Z}_p$ such that $H(g^{t_i}) = c_i \oplus M'_i$. So,

$$c'_i = M'_i \oplus H(\prod_{j=1}^{n} g^{z_{i,j}\cdot\bar{r}_j}) = M'_i \oplus H(g^{\mathbf{z}_i\cdot\bar{\mathbf{r}}^T}) = M'_i \oplus H(g^{t_i}) = c_i.$$

- For all $i \in I$, $c'_i = c_i$. Note that $\mathbf{a}\cdot\bar{\mathbf{r}}^T = \mathbf{a}\cdot\mathbf{r}^T$. So, for all $i \in I$ we have:

$$c'_i = M'_i \oplus H(\prod_{j=1}^{n} g^{a_j\cdot s_i\cdot\bar{r}_j}) = M_i \oplus H(g^{s_i\cdot\mathbf{a}\cdot\bar{\mathbf{r}}^T}) = M_i \oplus H(g^{s_i\cdot\mathbf{a}\cdot\mathbf{r}^T}) = c_i.$$

  We now show that the sampling procedure of t_i for all $i \notin I$ does not abort. Note that, since H is a universal hash function, the Leftover Hash Lemma (Lemma 1) implies that the distribution of $H(g^{t_i})$ is statistically close to uniform over $\{0,1\}$. For a randomly chosen $t_i \xleftarrow{\$} \mathbb{Z}_p$, we thus have $\Pr[H(g^{t_i}) = c_i \oplus M'_i] \geq 1/2 - \mathsf{negl}(\lambda)$. With overwhelming probability, the sampling procedure successfully terminates after at most λ attempts.

  We are left with proving that the system of equation (1) has a solution. Since the system should be solved over $\mathbb{Z}_p$, it is enough to show that the matrix

$$\begin{bmatrix} \mathbf{a} \\ \mathbf{z}_{i_1} \\ \vdots \\ \mathbf{z}_{i_\alpha} \end{bmatrix}$$

has full rank with high probability (here, $(i_1, \cdots, i_\alpha) = [\ell] \setminus I$). Since each row of this matrix is chosen uniformly at random, it has full rank with probability $1 - \mathsf{negl}(\lambda)$.

Finally, the distribution of $\mathbf{r}$ which is the randomness used in the Enc algorithm when $b = 0$ is the same as the distribution of $\bar{\mathbf{r}}$, which is the randomness output by the $\mathsf{EquivCT}$ algorithm when $b = 1$, which completes the proof.

This concludes the proof of Theorem 2. □

A.3 Proof of SD-based PEPE

Theorem 3. *Assuming SD, Construction 4 is a rate-1 PEPE.*

Proof. The proof is similar to the proof of correctness for the DDH-based construction since the ciphertext is computed in the same way and the public key for the indices $i \in I$ has the same form as in the DDH-based construction.

Public key randomness indistinguishability. Let $(p, q, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$, and $N = pq$. Let $\hat{g} \leftarrow u^q$, and $\hat{h} \leftarrow v^p$, be two generators of $\mathbb{G}_p$, and $\mathbb{G}_q$, respectively, where $v, u \xleftarrow{\$} \mathbb{G}$. We use a hybrid argument to prove the lemma. We start with $\mathcal{H}_0$ which is the experiment $\mathsf{IDEAL}_{\mathsf{pk}_0}$, and we end up at $\mathcal{H}_{|I'|-|I|}$ which is the experiment $\mathsf{REAL}_{\mathsf{pk}}$. We show that if the subgroup decision assumption holds, then the advantage of an adversary in distinguishing between two successive games is negligible. Hence, the distribution of the public key randomness is indistinguishable in $\mathsf{IDEAL}_{\mathsf{pk}_0}$ and in $\mathsf{REAL}_{\mathsf{pk}}$.

Hybrid $\mathcal{H}_0$. This is the experiment $\mathsf{IDEAL}_{\mathsf{pk}_0}$:

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, 0, I', r'_G)$, where $\mathsf{crs} = (N, \mathbb{G}, \hat{g}, \hat{h}, \{g_j\}_{j\in[n]})$, $\mathsf{pk} = (N, \mathbb{G}, \hat{g},$
  $\hat{h}, \{g_j\}_{j\in[n]}, \{h_{i,j}\}_{i\in[\ell], j\in[n]})$.
- Run $r_G \leftarrow \mathsf{EquivPK}(\mathsf{sk}, \mathsf{td}, 0, (I', r'_G), I)$, where $\mathsf{td} = (p, q)$.
- $b \leftarrow \mathcal{A}(r_G)$.

Let $(i_1, \cdots, i_\alpha) = I' \setminus I$. We define a sequence of hybrid games $\mathcal{H}_{i_1}, \cdots, \mathcal{H}_{i_\alpha}$ such that $\mathcal{H}_{i_k}$ is as follows:

Hybrid $\mathcal{H}_{i_k}$. This is a modified version of the experiment $\mathsf{IDEAL}_{\mathsf{pk}_0}$ in which we have:

$$\forall j \in [n] : h_{i,j} = \begin{cases} g_j^{s_i} & i \in I \cup \{i_1, \cdots, i_{\alpha-k}\} \\ \xleftarrow{\$} \mathbb{G} & \text{Otherwise} \end{cases},$$

where s_i's are elements of sk as before. So, in each $\mathcal{H}_{i_k}$, we send the following key generation randomness to $\mathcal{A}$, where we claim that elements $h_{i,j}$ for the index $i = i_{\alpha-k+1} \in I' \setminus I$, and all indices $j \in [n]$, are random elements of $\mathbb{G}$:

- Set $r_G = (N, \mathbb{G}, \hat{g}, \hat{h}, \{s_i\}_{i\in I}, \{h_{i,j}\}_{i\notin I, j\in[n]})$, and send it to $\mathcal{A}$.

- $b \leftarrow \mathcal{A}(r_G)$.

Note that $\mathcal{H}_{i_\alpha}$ is the same as the $\mathsf{REAL}_{\mathsf{pk}}$ experiment with respect to the subset I. The difference between any two successive games is that we replace n elements of the form $g_j^{s_i}$ in the former game with random elements of the group $\mathbb{G}$ in the latter game for some index $i \in I' \setminus I$. In order to prove that two consecutive games are indistinguishable, we have to prove that the following two distributions are indistinguishable:

$$D_0 := (N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, g_p^{\mathbf{a}\cdot s}) \approx_c D_1 := (N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, \mathbf{y}),$$

where g_p and g_q are random generators of $\mathbb{G}_p$ and $\mathbb{G}_q$, respectively, $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_N^n$, $s \xleftarrow{\$} \mathbb{Z}_N$, and $\mathbf{y} \xleftarrow{\$} \mathbb{G}^n$.
Regarding the SD assumption, we have $(N, \mathbb{G}, g_p, g_q, x^q) \approx_c (N, \mathbb{G}, g_p, g_q, x)$. Thus, we have:

$$(N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, x^{q\cdot\mathbf{a}}) \approx_c (N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, x^{\mathbf{a}}), \tag{3}$$

for a random vector $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_N^n$. Note that we can write any $x \in \mathbb{G}$ as $x = g_p^\alpha \cdot g_q^\beta$ for some $\alpha \in \mathbb{Z}_p$ and $\beta \in \mathbb{Z}_q$. Thus, the left-hand side distribution simulates $D_{0,1}$. In the right-hand side distribution, after rewriting x with respect to g_p and g_q, we have

$$x^{a_j} = g_p^{\alpha\cdot(a_j \mod p)} \cdot g_q^{\beta\cdot(a_j \mod q)},$$

for each $j \in [n]$. Here, first of all, for each index $j \in [n]$, the exponents $(a_j \mod p)$ and $(a_j \mod q)$ are independent. Secondly, since $\alpha \cdot (a_j \mod p)$, is an element of the field $\mathbb{Z}_p$, then if α is random, this element will be random, too. The same holds for $\beta \cdot (a \mod q)$. So, we can replace x^{a_j} by an element of the form $g_p^{t_j} \cdot g_q^{k_j}$, where t_j and k_j are independent random elements of $\mathbb{Z}_p$, and $\mathbb{Z}_q$, respectively. In conclusion, we can replace x^{a_j} by a random element of $\mathbb{G}$. Also, since each a_i and a_j for $i \neq j$ are independent random elements, then we can replace the vector $x^{\mathbf{a}}$ by a random vector $\mathbf{y} \xleftarrow{\$} \mathbb{G}$. So, assuming SD for the group $\mathbb{G}$, the two distributions are indistinguishable.

Next, we prove that $\mathsf{IDEAL}_{\mathsf{pk}_1} \approx_c \mathsf{REAL}_{\mathsf{pk}}$. The distribution of r_G in $\mathsf{IDEAL}_{\mathsf{pk}_1}$, and $real_{\mathsf{pk}}$ are as follows:

$$\mathsf{REAL}_{\mathsf{pk}} = (N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, \{g_p^{\mathbf{a}\cdot s_i}\}_{i\in I}, \{h_{i,j}\}_{i\notin I, j\in[n]}),$$

and

$$\mathsf{IDEAL}_{\mathsf{pk}_1} = (N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, \{g_p^{\mathbf{a}\cdot s_i}\}_{i\in I}, \{\widetilde{h_{i,j}}\}_{i\notin I, j\in[n]}),$$

where for all $j \in [n]$, and $i \notin I$, $h_{i,j} \xleftarrow{\$} \mathbb{G}$ in $\mathsf{REAL}_{\mathsf{pk}}$, and

$$\forall j \in [n]: \quad \widetilde{h_{i,j}} = \begin{cases} g_j^{s_i} & i \in I' \setminus I \\ h_i^{\delta_{i,j}} \cdot g_j^{s_i} & i \notin I' \end{cases}$$

Recall that when proving $\mathsf{REAL}_{\mathsf{pk}} \approx_c \mathsf{IDEAL}_{\mathsf{pk}_0}$, we proved that for each index $i \notin I'$, $\{g_j^{s_i}\}_{j\in[n]}$ is indistinguishable from a random vector of n elements of $\mathbb{G}$. So, it's enough to prove that for each $i \notin I'$, $\{h_i^{\delta_{i,j}} \cdot g_j^{s_i}\}_{j\in[n]}$ is indistinguishable from a vector of the form $\{g_j^{s_i}\}_{j\in[n]}$. More specifically, we prove that assuming SD, the two following distributions are computationally indistinguishable:

$$D_0 := (N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, g_p^{\mathbf{a}\cdot s}) \approx_c D_2 := (N, \mathbb{G}, g_p, g_q, g_p^{\mathbf{a}}, \{h^{\delta_{i^*,j}} \cdot g_p^{a_j \cdot s}\}_{j\in[n]}),$$

where $\mathbf{a} = (a_1, \cdots, a_n)$ is a random vector of $\mathbb{Z}_N^n$, $i^* \in [\ell]$, $s \xleftarrow{\$} \mathbb{Z}_N$, and h is a random element of the subgroup $\mathbb{G}_q$. Now, we prove the following lemma, and use it to prove the indistinguishability.

Lemma 5. *Let $(p, q, \mathbb{G}, g) \leftarrow \mathcal{G}$. Then if the subgroup decision assumption holds in $\mathbb{G}$, the two following distributions are computationally indistinguishable:*

$$D_0 := (N, \mathbb{G}, g_p, g_q, g_p^t, g_p^s, g_p^{s\cdot t}) \approx_c D_1 := (N, \mathbb{G}, g_p, g_q, g_p^t, g_p^s, h \cdot g_p^{s\cdot t}) \ ,$$

where g_p and g_q are random elements of $\mathbb{G}_p$ and $\mathbb{G}_q$, respectively, and $t, s \xleftarrow{\$} \mathbb{Z}_N$, and $h \xleftarrow{\$} \mathbb{G}_q$.

Proof. We prove this lemma in two steps:

- *First we show that if the subgroup decision holds relative to $\mathcal{G}$, then the two following distributions are computationally indistinguishable:*

  $$D_0 := (N, \mathbb{G}, g_p, g_q, g_p^t, g_p^s, g_p^{s\cdot t}) \approx_c D_1' := (N, \mathbb{G}, g_p, g_q, g_p^t, u \cdot g_p^s, v \cdot g_p^{s\cdot t}),$$

  where $u, v \xleftarrow{\$} \mathbb{G}_q$.
  Regarding the SD assumption, we have $(N, \mathbb{G}, g_p, g_q, x^q) \approx_c (N, \mathbb{G}, g_p, g_q, x)$. Thus, we have

  $$(N, \mathbb{G}, g_p, g_q, g_p^t, x^q, x^{q\cdot t}) \approx_c (N, \mathbb{G}, g_p, g_q, g_p^t, x, x^t),$$

  where $t \xleftarrow{\$} \mathbb{Z}_N$. Note that we can write any random element $x \in \mathbb{G}$ as $x = g_p^a \cdot g_q^b$ for some $a, b \in \mathbb{Z}_N$. So, the left-hand-side distribution simulates the distribution D_0. And, the right-hand-side distribution will be equal to

  $$(N, \mathbb{G}, g_p, g_q, g_p^t, g_p^a \cdot g_q^b, g_p^{a\cdot t} \cdot g_q^{b\cdot t}).$$

  Note that g_p^t only depends on $(t \mod p)$, which is independent of $(t \mod q)$ in g_q^t that appears in the expression $x^t = g_p^{a\cdot t} \cdot g_q^{b\cdot t}$. So, we can consider $u = g_q^b$ and $v = g_q^{b\cdot(t \mod q)}$ and, since $(t \mod q)$ is independent of other components of the distribution, u and v are distributed independent random elements of $\mathbb{G}_q$.

- *We show that, if the subgroup decision holds relative to $\mathcal{G}$, then $D_1' \approx_c D_1$, or in other words:*

$$D_1' := (N, \mathbb{G}, g_p, g_q, g_p^t, u \cdot g_p^s, v \cdot g_p^{s \cdot t}) \approx_c D_1 := (N, \mathbb{G}, g_p, g_q, g_p^t, g_p^s, h \cdot g_p^{s \cdot t}),$$

  where $u, v, h \xleftarrow{\$} \mathbb{G}_q$.
  Regarding the SD assumption, we have

$$(N, \mathbb{G}, g_p, g_p^t, x, g_q^\alpha \cdot x^t) \approx_c (N, \mathbb{G}, g_p, g_p^t, x^q, g_q^\alpha \cdot x^{q \cdot t})$$

  where $t, \alpha \xleftarrow{\$} \mathbb{Z}_N$. Here again, since any $x \in \mathbb{G}$ can be written as $x = g_p^a \cdot g_q^b$ for some $a, b \in \mathbb{Z}_N$, the left-hand and right-hand side distributions simulate D_1' and D_1, respectively. A simple hybrid argument concludes that regarding the above steps, the statement of the lemma is correct and $D_0 \approx_c D_1$.

This concludes the proof of Lemma 5. □

Now we use Lemma 5 to prove that $D_0 \approx_c D_2$. Suppose there is an adversary, $\mathcal{A}$, that can distinguish between D_0 and D_2. We build an adversary $\mathcal{B}$ that given a tuple $(N, \mathbb{G}, g_p, g_p^{a^*}, g_p^{s^*}, x)$, uses $\mathcal{A}$ to decide whether $x = g_p^{a^* \cdot s^*}$ or $x = h \cdot g_p^{a^* \cdot s^*}$ for some $h \xleftarrow{\$} \mathbb{G}_q$. The adversary $\mathcal{B}$ first chooses an index i^*, and runs $\mathcal{A}$ on the following input

$$(N, \mathbb{G}, \{g_j\}_{j \in [n]}, \{h_{i^*,j}\}_{j \in [n], (j \neq i^*)}, h_{i^*,i^*}),$$

where

- $g_j \leftarrow {g_p}^{a_j}$, for a random $a_j \xleftarrow{\$} \mathbb{Z}_N$, for all $j \neq i^*$;
- $g_{i^*} \leftarrow g_p^{a^*}$;
- $\forall j \in [n]: \quad h_{i^*,j} = \begin{cases} (g_p^{s^*})^{a_j} & j \neq i^* \\ x & j = i^* \end{cases}$

If $x = g_p^{a^* \cdot s^*}$, then $\mathcal{A}$ obtains a distribution where $h_{i^*,i^*} = g_p^{a^* \cdot s^*}$. Otherwise, if $x = h \cdot g_p^{a^* \cdot s^*}$, for some $h \xleftarrow{\$} \mathbb{G}$, then $\mathcal{A}$ will see a distribution where $h_{i^*,i^*} = h \cdot g_p^{a^* \cdot s^*}$. Thus, if $\mathcal{A}$ succeeds, then $\mathcal{B}$ would succeed, which contradicts the SD assumption.

Ciphertext randomness indistinguishability. Let $ct = (c_0, c_1, \cdots, c_\ell) \leftarrow \mathsf{Enc}(\mathsf{pk}, M, r_E)$ for $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, 1, I, r_G)$ and $r_E = (r_1, \cdots, r_n)$, where $\mathsf{pk} = (N, \mathbb{G}, \hat{g}, \hat{h}, \{g_j\}_{j \in [n]}, \{h_{i,j}\}_{i \in [\ell], j \in [n]})$, $\mathsf{sk} = (I, \{s_i\}_{i \in [\ell]})$, and $\mathsf{td} = (p, q)$. Now let $M' \in \{0,1\}^\ell$ such that $M_i = M_i'$ for all $i \in I$, and $M_i \neq M_i'$ otherwise. After running $\mathsf{EquivCT}(\mathsf{sk}, \mathsf{td}, (M, r_E), \{M_i'\}_{i \notin I})$ we obtain $r_E' = (r_1', \cdots, r_n')$. Let $ct' = (c_0', c_1', \cdots, c_\ell') \leftarrow \mathsf{Enc}(\mathsf{pk}, M', r_E')$.

Note that when $b = 1$, the element $\hat{g}$ generates a subgroup of $\mathbb{G}$ of order p. So, $c_0 = \prod_{j=1}^{n} \hat{g}^{a_j \cdot r_j}$ only determines the randomness vector $\{r_j\}_{j \in [n]}$ modulo p, and choosing $r'_j = r_j \mod p$ ensures us that $c'_0 = c_0$. Also, the distribution of r'_E is the same as the distribution of r_E and equal to $\mathcal{U}(\mathbb{Z}_N)$.

Furthermore, as for Equation (2), for each $i \notin I$, we have

$$c_i = M'_i \oplus H(h^{r'_i} \cdot c_0^{s_i}) = c'_i.$$

We now show that with overwhelming probability, $\mathsf{EquivCT}$ can successfully resample the encryption randomness after at most λ attempts.

First, for each $i \notin I$, the distribution of $H(h^{r'_i}.c_0^{s_i})$ is statistically close to the uniform distribution over $\{0, 1\}$. This is because r'_i is sampled uniformely from $\mathbb{Z}_N$, so the distribution of $h^{r'_i}.c_0^{s_i}$ for a random r'_i has at least $\log(N)$ bits of min-entropy. Since H is a universal hash function, we have $1/N = \mathsf{negl}(\lambda)$. Thus, regarding the leftover hash lemma (Lemma 1), the distribution of $H(h^{r'_i}.c_0^{s_i})$ is statistically close to the uniform distribution over $\{0, 1\}$. So, for a random r'_i, the value of the hash $H(h^{r'_i}.c_0^{s_i})$ is equal to $c_i \oplus M'_i$ at least with probability $1/2 - \mathsf{negl}(\lambda)$. The probability of not finding such r'_i after λ attempts is $(\frac{1}{2})^{\lambda} - \mathsf{negl}(\lambda) = \mathsf{negl}(\lambda)$. For each $i \notin I$, with overwhelming probability, $\mathsf{EquivCT}$ obtains a randomness that satisfies Equation (2).

This concludes the proof of Theorem 3. $\square$