Deniable Authentication when Signing Keys Leak*

Suvradip Chakraborty[1]** , Dennis Hofheinz[2], Ueli Maurer[2] and Guilherme Rito[2]

[1] Visa Research
suvradip1111@gmail.com
[2] Department of Computer Science, ETH Zurich, Zürich, Switzerland
{hofheinz,maurer,gteixeir}@inf.ethz.ch

Abstract. Deniable Authentication is a highly desirable property for secure messaging protocols: it allows a sender Alice to authentically transmit messages to a designated receiver Bob in such a way that only Bob gets convinced that Alice indeed sent these messages. In particular, it guarantees that even if Bob tries to convince a (non-designated) party Judy that Alice sent some message, and even if Bob gives Judy his own secret key, Judy will not be convinced: as far as Judy knows, *Bob could be making it all up!*

In this paper we study Deniable Authentication in the setting where Judy can additionally obtain Alice's secret key. Informally, we want that knowledge of Alice's secret key does not help Judy in learning whether Alice sent any messages, even if Bob does not have Alice's secret key and even if Bob cooperates with Judy by giving her his own secret key. This stronger flavor of Deniable Authentication was not considered before and is particularly relevant for Off-The-Record Group Messaging as it gives users stronger deniability guarantees.

Our main contribution is a scalable "MDRS-PKE" (Multi-Designated Receiver Signed Public Key Encryption) scheme—a technical formalization of Deniable Authentication that is particularly useful for secure messaging for its confidentiality guarantees—that provides this stronger deniability guarantee. At its core lie new MDVS (Multi-Designated Verifier Signature) and PKEBC (Public Key Encryption for Broadcast) scheme constructions: our MDVS is not only secure with respect to the new deniability notions, but it is also the first to be tightly secure under standard assumptions; our PKEBC—which is also of independent interest—is the first with ciphertext sizes and encryption and decryption times that grow only linearly in the number of receivers. This is a significant improvement upon the construction given by Maurer et al. (EUROCRYPT '22), where ciphertext sizes and encryption and decryption times are quadratic in the number of receivers.

---

* This is the full version of article [5], , https://doi.org/10.1007/978-3-031-30620-4_3.

** All of the work was done while author was at ETH Zurich.

Erratum

As noted by Maurer et al. in [20], it is not clear how to prove the off-the-record security of the MDRS-PKE construction given in [21]. Fortunately, also in [20], Maurer et al. show how to fix their construction so all security proofs, including off-the-record, go through. In an earlier version of this paper we considered the original MDRS-PKE construction given by Maurer el al. in [21], and claimed that the same security proofs with minor adaptations would work for the setting considered in this paper. While for the most part this is still the case, unfortunately for $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security the arguments given in [20] do not seem to apply for the setting we consider in this paper (where the adversary is given access to the secret key of honest senders). To fix this issue, in this new (full) version we introduce a new security notion for MDVS schemes—Message-Bound Validity—with which we can prove the security of the (modified) MDRS-PKE construction from [20] in the new setting considered in this paper. We also prove that our MDVS construction satisfies this new Message-Bound Validity notion.

While these are the main changes in this new full-version, we made other smaller fixes.

1 Introduction

Motivation. More than 3 billion people currently use messaging apps.[3] Naturally, there is a demand for *secure* messaging which guarantees, e.g., the secrecy of the transmitted contents, or the authenticity of senders. For point-to-point connections, combining standard cryptographic building blocks (like digital signature, public-key, and secret-key encryption schemes) may be sufficient. However, in particular for *group messaging* (in which groups of users communicate in a group chat), additional security properties are desirable. For instance, group members may want to be sure that all members receive the *same* messages (a property that, surprisingly, is not captured by traditional broadcast encryption definitions [10]).

Another security property that is generally desirable in messaging is *deniability*. Intuitively, it should be possible for a sender to deny having sent a message, or for a receiver to deny having received a particular message. Achieving deniability is even more challenging when considering that users may store copies of received (or even sent) messages on their communication device.

Here, we focus on a relatively mild (but still technically quite challenging) variant of deniability: "Off-The-Record" (OTR) messaging. Informally, with OTR security, received ciphertexts can be simulated, in the sense that it is easy to come up with ciphertexts for arbitrary messages that *look* as if they had been sent by a particular sender. In this sense, OTR security guarantees that third parties cannot be convinced of group-internal interactions. Of course, even OTR is relatively difficult to achieve, and becomes even harder so in the group messaging setting.

MDRS-PKE schemes. When translating desirable properties of such group messaging protocols into suitable cryptographic primitives (with associated properties), we end up with "Multi-Designated Receiver Signed Public Key Encryption" (MDRS-PKE, [21]). Informally, these protocols function like signed versions of broadcast encryption schemes with additional integrity properties (that guarantee, e.g., that all receivers receive the same message). A little more formally, MDRS-PKE schemes guarantee the following:

Syntax: A sender can prepare a single broadcast ciphertext c for a set $\mathcal{R}$ of intended receivers. Any intended receiver in $\mathcal{R}$ can decrypt c to retrieve the identity $\mathtt{pk}_S$ of the sender S, the encrypted message m, and the set $\mathcal{R}$.

Consistency: Not even a maliciously created c should decrypt to different sender identities, messages, or receiver sets for different intended receivers. Furthermore, if one receiver decrypts to $(\mathtt{pk}_S, m, \mathcal{R})$, then all receivers in $\mathcal{R}$ obtain the same $(\mathtt{pk}_S, m, \mathcal{R})$.

Unforgeability: Nobody except S can produce a ciphertext that decrypts to sender identity $\mathtt{pk}_S$ for any receiver.

Anonymity: c does not reveal the sender S or the set $\mathcal{R}$ of intended receivers (only its size $|\mathcal{R}|$).

[3] `https://www.businessofapps.com/data/messaging-app-market/`

Confidentiality: c does not reveal the encrypted message (only its length $|m|$).

Off-The-Record: Plausible-looking ciphertexts c can be simulated by any (subset of) intended receivers of that ciphertext. Intuitively, this guarantees that receivers cannot convince a third party of a received encrypted message.

MDRS-PKE is a complex primitive, and appears to require specific, case-tailored primitives to realize it. For instance, the combination of a *group* of designated receivers and the simulation properties required by OTR prevent the use of ordinary designated-verifier signatures (or even MACs) [8].

Fortunately, [21] shows how to construct MDRS-PKE schemes from a combination of suitable variants of signature and broadcast encryption schemes. Specifically, they require the following:

– A type of signature scheme called "Multi-Designated Verifier Signature" (MDVS [7,8,15]) with suitable consistency, unforgeability, and OTR properties. (Here, "OTR" means that valid-looking signatures can be simulated by designated receivers.) State-of-the-art MDVS constructions [8] exist from algebraic assumptions (like the combination of Diffie-Hellman and Paillier-like assumptions), and also from generic primitives (like the combination of non-interactive key exchange (NIKE), non-interactive zero-knowledge (NIZK), and a few other standard primitives).
– A type of broadcast encryption scheme called "Public-Key Encryption for Broadcast" (PKEBC [21]) that essentially has all the properties of an MDRS-PKE scheme except for authenticity. PKEBC schemes can be instantiated from a combination of public-key encryption, NIZKs, and commitments.
– Strongly Unforgeable (sEUF-CMA) One-Time Digital Signature Scheme. Such schemes can be instantiated from a multitude of standard assumptions.

The current situation. In summary, we do have tools that give meaningful security and privacy guarantees for group messaging even in face of corruptions. The current state of the art [8, 21] leaves a few questions unanswered, however:

Limited deniability guarantees. The deniability (technically: OTR) guarantees given by the combination of [8,21] are limited to the case where the secret keys of honest senders remain secret. In particular, simulated ciphertexts are only proven to look plausible when the corresponding sender key is unknown. However, current deniability notions do not provide any guarantees if an honest sender is forced (or blackmailed) to give away its secret key, in which case the sender might not be able to plausibly deny having sent a message.

Limited unforgeability guarantees. The MDVS constructions and analyses from [8] show unforgeability only in a setting in which an adversary has no verification oracle. (Intuitively, in such designated-verifier settings, signatures are not publicly verifiable, and hence typically adversaries are given access to an explicit verification oracle [18, 23].) This is undesirable, in particular because a constructive modeling of MDVS schemes [19] requires such a verification oracle. As a result, the resulting combined MDRS-PKE scheme from [8, 21] suffers from a similarly weak unforgeability guarantee.[4]

Limited scalability. The combined MDRS-PKE construction of [8, 21] has ciphertexts whose sizes are *quadratic* in the number of receivers. This is clearly undesirable for large groups. Furthermore, while the generic transformation of [21] itself is tightly secure, i.e., gives security guarantees that do not incur a loss in the number of parties or ciphertexts, the underlying primitives from [8,21] are not known to be. In particular, the (known) security guarantees of the final scheme degrade in the number of ciphertexts and users.

Gaps in some proofs. Unfortunately, some of the proofs in [8] appear incomplete. (See [6, Appendix C] for details.)

Our contribution. In this work, we construct a MDRS-PKE scheme that

- enjoys strong deniability guarantees (i.e. a strong OTR notion that takes into account leaked sender secret keys),
- likewise enjoys strong unforgeability properties (that take into account adversaries with a verification oracle),
- is scalable, in the sense that ciphertext sizes, encryption and decryption times are linear in the number of receivers, and we can prove it tightly secure based on primitives for which tightly secure instantiations are known.

Like [21], our MDRS-PKE scheme is based upon suitable MDVS and PKEBC schemes. In fact, we use the same generic MDRS-PKE construction as [21], but for more secure and more efficient MDVS and PKEBC schemes (that we also provide). In particular, we provide

- a conceptually simple MDVS scheme that achieves strong OTR and strong unforgeability guarantees (as explained above),
- a PKEBC scheme for which ciphertext sizes, and both encryption and decryption times only grow linearly with the number of receivers.

Both of these schemes can be proven tightly secure from primitives that have tightly secure instantiations from standard computational assumptions. In particular, unlike [8], we avoid the use of non-interactive key exchange, a primitive which is known to be difficult to prove tightly secure [2, 14].

2 Technical Overview

We now give an overview of the techniques used to construct our MDRS-PKE scheme. As aforementioned, our scheme is tightly secure under adaptive corruptions and satisfies the new (stronger) OTR notion considered in this paper. The

[4] It should be noted that this shortcoming appears to have gone unnoticed. In particular, [21] explicitly define and assume MDVS schemes that are unforgeable in the presence of a verification oracle, while [8] simply do not prove this property about their MDVS schemes. Technically speaking, this means the transformation of [21] cannot be directly applied to the MDVS schemes from [8].

main building blocks of our construction are: 1. a new MDVS scheme construction satisfying (the MDVS analogous of) the new OTR security notion which is tightly secure under adaptive corruptions; and 2. a new PKEBC scheme construction with linear-size ciphertexts, and linear-time encryption and decryption which is also tightly secure under adaptive corruptions. By following (a straightforward generalization of) the transformation given in [21] we then obtain the intended MDRS-PKE scheme. It is worth noting that, since the MDRS-PKE construction given in [21] uses the PKEBC scheme to encrypt a message whose size is already linear in the number of receivers, it is not sufficient for the underlying PKEBC scheme to have ciphertext sizes, encryption and decryption times that grow linearly with the number of receivers *times* the size of the message: it is necessary for the PKEBC's ciphertext sizes, encryption and decryption times to grow linearly with the number of receivers *plus* the size of the message. This is exactly what we achieve: when instantiated with our new MDVS and PKEBC constructions, the MDRS-PKE construction given in [21] yields the first (MDRS-PKE) scheme that satisfies the new stronger OTR notion, that has ciphertext size, encryption and decryption times that grow linearly with the number of receivers, and that is tightly secure under adaptive corruptions.

2.1 MDVS Construction

We now give an overview of our MDVS scheme construction. As a first step we consider the case of a single verifier and show how to construct a Designated Verifier Signature (DVS) scheme. This already conveys the main technical ideas of our construction. Then we discuss how to generalize the DVS to the case of multiple verifiers (MDVS), and, finally, we explain how to achieve tight security under adaptive corruptions. The building blocks of all our (M)DVS constructions are an IND-CPA secure PKE scheme, a One-Way Function (OWF) F and a Simulation-Sound (SS) NIZK.

The DVS *scheme.* Our signature scheme is of the following form: the public parameters $\texttt{pp}$ consist of a public key $\texttt{pk}$ of the PKE scheme, and a *Common Reference String* $\texttt{crs}$ of the NIZK argument system. The secret signing key $\texttt{ssk}$ is a pre-image x_S of the OWF F and the signer's public key $\texttt{spk}$ is the corresponding image (i.e. $\texttt{spk} = y_S = F(x_S)$). A verifier's key-pair is similar, except that it additionally includes a PKE key pair $(\texttt{pk}_V, \texttt{sk}_V)$: the verifier's secret key $\texttt{vsk}$ consists of a pre-image x_V of F together with the PKE secret key $\texttt{sk}_V$; the verifier's public key $\texttt{vpk}$ are the corresponding public keys, i.e. $\texttt{vpk} = (\texttt{pk}_V, y_V \coloneqq F(x_V))$. To sign a message m (using $\texttt{ssk} = x_S$, and $\texttt{vpk} = (\texttt{pk}_V, y_V)$), we first generate two ciphertexts, c and $c_{\texttt{pp}}$: c encrypts the bit 1 under the verifier's public key $\texttt{pk}_V$ (the role of this will be clear soon); $c_{\texttt{pp}}$ encrypts the tuple $(m, 1, \texttt{ssk})$ under the public key $\texttt{pk}$ included in the public parameters $\texttt{pp}$. Finally, we generate a NIZK proof π that binds the ciphertexts together: π proves that both $c_{\texttt{pp}}$ and c are well-formed and encrypt the same bit b, and that if $b = 1$ then $c_{\texttt{pp}}$ encrypts a pre-image (under F) of either y_S or y_V. The signature σ then consists of the tuple $(c_{\texttt{pp}}, c, \pi)$. To verify a signature the receiver first verifies the

NIZK proof π and then decrypts ciphertext c using its PKE secret key $\mathtt{sk}_V$; the signature is valid if π is a valid NIZK proof and the decryption of c is 1.

Simulating a signature works as follows: 1. for the case of a dishonest verifier, to simulate a signature one proceeds just like an honest signer would to generate a signature, the only difference being that $c_{\mathtt{pp}}$, instead of encrypting x_S—the pre-image of the signer's public key—encrypts x_V—the pre-image of the verifier's public key; 2. if the verifier is honest, one forges a signature by having c be an encryption of 0 under the verifier's public PKE key $\mathtt{pk}_V$, $c_{\mathtt{pp}}$ be an encryption of the triple $(m, 0, 0)$, and π be a NIZK proof. Note that, thanks to the NIZK relation we consider, in both cases one can compute a valid NIZK proof π: in the first case this is possible because $c_{\mathtt{pp}}$ encrypts a pre-image of the verifier's secret key; for the latter case this is possible because c is an encryption of 0.

To understand why the DVS scheme sketched above is unforgeable note first that if both the sender and the verifier are honest, by the one-wayness of F the adversary does not know a pre-image of neither $F(x_S)$ nor $F(x_V)$. On a high level the proof proceeds as follows: we begin by changing both the public parameter's $\mathtt{crs}$ and each signature's NIZK proof by simulated ones. We, next, further modify the signatures the adversary sees by making $c_{\mathtt{pp}}$ be an encryption of a "0" string—possible by the IND-CPA security of the underlying PKE scheme. Note that at this point all the adversary sees is independent of both $\mathtt{ssk} = x_S$ and $\mathtt{vsk} = x_V$.[5] Now suppose the adversary manages to come up with a forgery $(c^*_{\mathtt{pp}}, c^*, \pi^*)$ corresponding to some message m^* whose signature it has never seen: if the forgery is valid then on one hand c^* is encryption of bit 1 and on the other hand π^* is a valid NIZK proof; by (simulation) soundness this means that $c^*_{\mathtt{pp}}$ encrypts a pre-image of either y_S or y_V. However, at this point we can use the PKE secret key corresponding to the public parameter's public key to extract the pre-image, contradicting the one-wayness of F.

Understanding why the scheme sketched above satisfies the (stronger) OTR property is more involved (and refer the reader to the full proof of Theorem 7 for details). For simplicity, below we consider a weaker OTR notion—one where the adversary is not given access to a signature verification oracle: 1. If the verifier is dishonest the only differences between real and simulated signatures are that in the first case $c_{\mathtt{pp}}$ encrypts x_S and the NIZK proof π is generated using x_S as (part of the) witness, whereas in the latter case $c_{\mathtt{pp}}$ encrypts x_V and π is generated using x_V. If an adversary were able to distinguish real signatures from simulated ones then it would be either breaking the IND-CPA security of the underlying PKE scheme, or the Zero-Knowledge security of the NIZK (or both). 2. If the verifier is honest the differences between real signatures and simulated ones are that in the first case $c_{\mathtt{pp}}$ encrypts x_S, c is an encryption of 1 and π is generated using x_S, while in a simulated signature $c_{\mathtt{pp}}$ encrypts a "0" string, c is an encryption of 0 and π is no longer generated using a pre-image of neither y_S nor y_V. So, if an adversary were be able to distinguish real and simulated signatures then it could break the IND-CPA security of the underlying

[5] Here, independent is in the sense that all the adversary sees only depends on $y_S \coloneqq F(x_S)$ and $y_V \coloneqq F(x_V)$, but not on any pre-image of y_S or y_V.

PKE scheme—since it could distinguish either the $c_{\mathtt{pp}}$ or the c ciphertexts—or could break the Zero-Knowledge of the NIZK.

Generalizing for multiple verifiers. We now discuss how to extend the previous construction to the case of multiple designated verifiers. The main difference is that we additionally need to guarantee consistency—meaning that either all honest verifiers accept a signature, or they all reject.

Signatures in our MDVS construction consist of a vector of ciphertexts $\vec{c} = (c_1, \cdots, c_n)$ (one per receiver) and a ciphertext $c_{\mathtt{pp}}$. Each ciphertext c_i is the encryption of a bit b_i under the i-th receiver's public key $\mathtt{pk}_{V_i}$, and the ciphertext $c_{\mathtt{pp}}$ is an encryption of the tuple $(m, b_{\text{global}}, \vec{\alpha} = (\alpha_1, \cdots, \alpha_n))$, where $\alpha_i = (b_i, x_i)$, under the public parameter's public key $\mathtt{pk}$. Similarly to the DVS construction, signatures also contain a NIZK proof π that not only ensures ciphertexts are well-formed and signatures are unforgeable, but also consistency. In particular, π proves: 1. all ciphertexts in $\vec{c}$ and ciphertext $c_{\mathtt{pp}}$ are well-formed—in particular each ciphertext c_i of $\vec{c}$ encrypts the bit b_i that is in the α_i encrypted in $c_{\mathtt{pp}}$; 2. for each verifier, say the i-th, if $b_i = 1$ then the α_i encrypted in $c_{\mathtt{pp}}$ contains a pre-image of either y_S—the signer's public key—or y_{V_i}—the i-th verifier's public key—under F (this guarantees unforgeability); and 3. for each i-th verifier, if the value x_i in α_i that is encrypted under $c_{\mathtt{pp}}$ is not a pre-image of this verifier's public key y_{V_i} then $b_i = b_{\text{global}}$ (this guarantees consistency). Note that, if the verification of the NIZK proof is deterministic, the NIZK's soundness implies that if two verifiers disagree on a signature's validity, one of them is dishonest.

Achieving tight security under adaptive corruptions. While the MDVS construction above already satisfies correctness, consistency, unforgeability and OTR, we do not know how to prove it is tightly secure under adaptive corruptions. Our problem is that we do not know how a reduction could know in advance which parties the adversary will corrupt (and thus ask for their secret keys) and which ones it will not. Suppose for example we are reducing an adversary from breaking some security property of the MDVS construction to breaking the $\mathsf{IND\text{-}CPA}$ game of the underlying PKE scheme, and in particular consider a reduction that simply guesses whether the adversary will corrupt a party P_i: on one hand, if the reduction guesses incorrectly that P_i will be corrupted then it is not taking advantage of the adversary to win the underlying $\mathsf{IND\text{-}CPA}$ game; on the other hand, if the reduction incorrectly guesses P_i will not be corrupted—in which case it would set P_i's public key to be one output by the underlying $\mathsf{IND\text{-}CPA}$ game—then we do not know how the reduction could handle a query for the secret key of P_i—and so the reduction would again not be taking advantage of the adversary to win the underlying $\mathsf{IND\text{-}CPA}$ game. So although one could resort to this guessing technique to prove the security of the MDVS scheme under adaptive corruptions (via a hybrid argument), this leads to a reduction loss that grows linearly with the number of parties.

To void this reduction loss we follow the "two-key" technique already used in the context of tightly secure public-key encryption [1]. In the new scheme, and at a high level, the public key of each party P_i is a pair of public keys—say

$(\mathtt{pk}_0, \mathtt{pk}_1)$—from the previous scheme, and its secret key consists of a bit b—picked uniformly at random—and the secret key $\mathtt{sk}_b$ corresponding to $\mathtt{pk}_b$. Signatures then consist of $c_{\mathtt{pp}}$ as before, a vector of ciphertexts that includes two ciphertexts per verifier—one under each of the verifier's public keys—and the NIZK proof π—which now proves that $c_{\mathtt{pp}}$ encrypts a pre-image of one of the public keys of a party (rather than a single one as before). This technique allows to come up with tight security reductions to the underlying building blocks: having the two keys allows, on one hand, to embed challenges in the part of the public key whose corresponding secret key is "forgotten", i.e. $\mathtt{pk}_{1-b}$, where b is the bit in the party's secret key, and on the other hand to handle any possible queries the adversary may make, including ones where the party's secret key is leaked.

2.2 PKEBC Construction

We now give a high level overview of our PKEBC scheme's construction. We first explain how to achieve linear sized ciphertexts and linear time encryption (in the number of receivers), and then move towards making decryption time also linear. (We note that the ciphertext size and both the encryption and decryption times of the only prior PKEBC scheme construction (see [21]) all grow quadratically in the number of receivers.) Since the technique we use to obtain tight security reductions under adaptive corruptions is the same one we used in the MDVS construction, we do not include it in this overview.

As building blocks, we assume an IND-CPA and IK-CPA secure PKE scheme, a Simulation-Sound NIZK and a (one-time) IND-CPA secure Symmetric Encryption (SKE) scheme. The public parameters of our PKEBC schemes are the same as for the MDVS construction—comprising a public key of a PKE scheme and a $\mathtt{crs}$ for a NIZK, i.e. $\mathtt{pp} = (\mathtt{pk}, \mathtt{crs})$—and in the two constructions discussed below a PKEBC key-pair is simply a key-pair of the underlying PKE scheme.

Achieving linear ciphertext size and encryption time. As we now explain, the main idea to achieve linear ciphertext sizes and encryption time (in the number of receivers) is to use hybrid encryption.

To encrypt a message m to a vector of receiver public keys $\vec{v} = (\mathtt{pk}_1, \ldots, \mathtt{pk}_n)$ we first encrypt $(\vec{v}, m)$ under the public parameters' public key; let $c_{\mathtt{pp}}$ denote the resulting ciphertext and $r_{\mathtt{pp}}$ the sequence of random bits used for this encryption. Next we generate a symmetric key k for the SKE scheme and for each receiver public key $\mathtt{pk}_i$ in $\vec{v}$ we encrypt k under $\mathtt{pk}_i$, resulting in a vector of ciphertexts $(c_1, \ldots, c_n)$. Then we use k to encrypt not only $\vec{v}$ and m, but also $r_{\mathtt{pp}}$; let $c_{\mathtt{sym}}$ denote the resulting (symmetric) ciphertext. (Having $c_{\mathtt{sym}}$ encrypt $\vec{v}$, m and $r_{\mathtt{pp}}$ allows receivers to confirm they obtained the correct vector of receivers and message: since the public parameter's public key is honestly sampled, $c_{\mathtt{pp}}$ is a commitment to $(\vec{v}, m)$, and since $c_{\mathtt{sym}}$ also encrypts $r_{\mathtt{pp}}$, a receiver can simply recompute $c_{\mathtt{pp}}$; as we will see, this is key to guaranteeing correctness, robustness and consistency.) Finally, we create a NIZK proof π showing that: 1. $c_{\mathtt{pp}}$ is an encryption of $(\vec{v}, m)$ under the public parameters' public key using $r_{\mathtt{pp}}$ as the sequence of random encryption bits; 2. the symmetric key k was correctly sampled; 3. $c_{\mathtt{sym}}$ is an encryption under k of $(r_{\mathtt{pp}}, \vec{v}, m)$; and 4. for each ciphertext c_i of $\vec{c}$, c_i is an encryption of k under the i-th public key $\mathtt{pk}_i$ of $\vec{v}$. The final ciphertext is then the quadruple $c = (c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}, \pi)$. To decrypt a receiver first checks if π is a valid NIZK proof; if π is valid the receiver then starts trying to decrypt each ciphertext $c_i \in \vec{c}$; for each symmetric key k' the receiver obtains from successfully decrypting a ciphertext c_i, the receiver tries decrypting $c_{\mathtt{sym}}$. If the decryption of $c_{\mathtt{sym}}$ is successful, returning a triple $(r_{\mathtt{pp}}, \vec{v}, m)$, the receiver checks if $c_{\mathtt{pp}}$ indeed encrypts $(\vec{v}, m)$ under the public parameters' public key using $r_{\mathtt{pp}}$ as the random encryption coins, and if it does the receiver outputs $(\vec{v}, m)$ as the result of decryption. If it does not (or any of the decryption attempts failed) the receiver moves on to the next ciphertext c_j of $\vec{c}$, or returns the special error symbol $\perp$ if there are no more ciphertexts.

It is easy to see that for a vector of receivers $\vec{v}$ and message m both the ciphertext size and the encryption time of the scheme are $O(|\vec{v}| + |m|)$, exactly as we needed. Unfortunately, the scheme *does not* achieve linear time decryption: in the worst case the decryption of each ciphertext $c_i \in \vec{c}$ outputs a valid looking symmetric key k'[6], the decryption of $c_{\mathtt{sym}}$ is successful—which, given the size of $c_{\mathtt{sym}}$ is linear in the number of receivers, already takes time linear in the number of receivers—but then the triple $({r_{\mathtt{pp}}}', \vec{v}', m')$ resulting from $c_{\mathtt{sym}}$'s decryption does not match $c_{\mathtt{pp}}$, i.e. $c_{\mathtt{pp}}$ is not the encryption of $(\vec{v}', m')$ under the public key of the public parameters, and using ${r_{\mathtt{pp}}}'$ as the random encryption coins. Given the number of ciphertexts of $\vec{c}$ is linear in the number of receivers, the time to decrypt then grows *quadratically* in the number of receivers.

Achieving linear decryption time. To achieve linear time decryption receivers need a fast way of checking if any particular ciphertext $c_j \in \vec{c}$ is really meant for them without having to decrypt $c_{\mathtt{sym}}$, as this already takes linear time in the number of receivers. A first idea is adding, for each receiver, an encryption of a long enough 0 bitstring (and appropriately modifying the NIZK relation): to decrypt, a receiver would then first check if the decryption of this new ciphertext would output back the expected 0 bitstring, and if not the receiver would not have to attempt decrypting the (linear sized) $c_{\mathtt{sym}}$ ciphertext. Unfortunately, this approach only works for honestly generated ciphertexts. For instance, consider two key-pairs $(\mathtt{pk}, \mathtt{sk}), (\mathtt{pk}', \mathtt{sk}')$ of some arbitrary PKE scheme with $\mathtt{pk} \neq \mathtt{pk}'$: one cannot assume that an adversarially created encryption of a 0 bitstring under $\mathtt{pk}$ does not decrypt, under the non-matching secret key $\mathtt{sk}'$, to the same 0 bitstring (and, more generally, to any particular value). This means that a dishonest sender could potentially come up with "malformed" ciphertexts that would pass this first check, thus making a receiver have to decrypt the (large) $c_{\mathtt{sym}}$ ciphertext and then recompute $c_{\mathtt{pp}}$ to ensure consistency.

The way our scheme achieves linear time decryption is by pairing each ciphertext $c_i \in \vec{c}$ with: 1. a commitment to the i-th receiver's public key $\mathtt{pk}_i$; and 2. a ciphertext that encrypts, under $\mathtt{pk}_i$, the random coins used to generate

[6] For an arbitrary PKE scheme a receiver cannot *a priori* tell whether a given ciphertext is intended for itself.

the commitment. More concretely, in our scheme there are three ciphertexts per receiver, i.e. $\vec{c} = (c_1, \ldots, c_n)$ with $c_i = (c_{i,0}, c_{i,1}, c_{i,2})$, where: $c_{i,0}$ is an encryption, under the public parameter's public key, of the i-th receiver's public key $\mathtt{pk}_i$ using some sequence of random bits $r_{i,0}$; $c_{i,1}$ is an encryption, under $\mathtt{pk}_i$, of the random coins $r_{i,0}$; and $c_{i,2}$ is an encryption of the SKE key k used to encrypt $c_{\mathtt{sym}}$. As one might note, by appropriately modifying the NIZK statement, we can ensure that receivers no longer need to recompute $c_{\mathtt{pp}}$ to confirm they obtained the correct pair $(\vec{v} = (\mathtt{pk}_1, \ldots, \mathtt{pk}_n), m)$ from the decryption of $c_{\mathtt{sym}}$: first, note that the correctness of the underlying PKE scheme together with the soundness of the NIZK (for the modified NIZK statement) guarantee that ciphertext $c_{j,0}$ of each triple $c_j = (c_{j,0}, c_{j,1}, c_{j,2})$ of $\vec{c}$ binds the triple to a single receiver public key $\mathtt{pk}_j$; second, the PKE scheme's correctness with the NIZK's soundness further imply that ciphertext $c_{j,2}$ of every triple is an encryption of the same symmetric key k under the public key $\mathtt{pk}_j$ bound to the triple; third, the SKE's (perfect) correctness again with the NIZK's soundness imply that the decryption of $c_{\mathtt{sym}}$ using the aforementioned key k yields the same pair $(\vec{v} = (\mathtt{pk}_1, \ldots, \mathtt{pk}_n), m)$, where for each $i \in \{1, \ldots, n\}$, the triple $c_i \in \vec{c}$ is bound to the (corresponding) public key $\mathtt{pk}_i \in \vec{v}$.[7] Since, as explained above, receivers need not recompute $c_{\mathtt{pp}}$, in the new scheme $c_{\mathtt{sym}}$ no longer encrypts the random coins $r_{\mathtt{pp}}$. Furthermore, as each receiver's public key $\mathtt{pk}_i$ is already encrypted under the public parameter's public key in $c_{i,0}$, $c_{\mathtt{pp}}$ no longer needs to encrypt vector $\vec{v}$; in the new scheme $c_{\mathtt{pp}}$ encrypts only the message m.

3 Preliminaries

We denote the arity of a vector $\vec{x}$ by $|\vec{x}|$ and its i-th element by x_i. We write $\alpha \in \vec{x}$ to denote $\exists i \in \{1, \ldots, |\vec{x}|\}$ with $\alpha = x_i$. We write $\mathrm{Set}(\vec{x})$ to denote the set induced by vector $\vec{x}$, i.e. $\mathrm{Set}(\vec{x}) \coloneqq \{x_i \mid x_i \in \vec{x}\}$.

Throughout the paper we frequently use vectors. We use upper case letters to denote vectors of parties, and lower case letters to denote vectors of artifacts such as public keys, sequences of random coins, etc. Moreover, we use the convention that if $\vec{V}$ is a vector of parties, then $\vec{v}$ denotes $\vec{V}$'s corresponding vector of public keys. For example, for a vector of parties $\vec{V} \coloneqq (\mathrm{Bob}, \mathrm{Charlie})$, $\vec{v} \coloneqq (\mathtt{pk}_{\mathrm{Bob}}, \mathtt{pk}_{\mathrm{Charlie}})$ is $\vec{V}$'s corresponding vector of public keys. In particular, V_1 is Bob and v_1 is Bob's public key $\mathtt{pk}_{\mathrm{Bob}}$, and V_2 is Charlie and v_2 is Charlie's public key $\mathtt{pk}_{\mathrm{Charlie}}$. More generally, for a vector of parties $\vec{V}$ with corresponding vector of public keys $\vec{v}$, V_i's public key is v_i, for $i \in \{1, \ldots, |\vec{V}|\}$.

[7] In [17] Libert et al. introduce "anonymous hint systems": a type of scheme that can be seen as an abstraction of the technique that allows receivers to quickly check if a particular ciphertext is meant for them (and which is key to achieving linear time decryption).

4 Multi-Designated Verifier Signature Schemes with Enhanced Off-The-Record Security

An MDVS scheme Π is a 6-tuple of Probabilistic Polynomial Time Algorithms (PPTs) $\Pi = (S, G_S, G_V, Sig, Vfy, Forge)$, where:

- S: on input 1^k, generates public parameters $\texttt{pp}$;
- G_S: on input $\texttt{pp}$, generates a signer key-pair ($\texttt{spk}$, $\texttt{ssk}$);
- G_V: on input $\texttt{pp}$, generates a verifier key-pair ($\texttt{vpk}$, $\texttt{vsk}$);
- Sig: on input $(\texttt{pp}, \texttt{ssk}, \vec{v}, m)$, where $\texttt{ssk}$ is the signer's secret key, $\vec{v}$ is the vector of public verifier keys of the designated verifiers and m is the message, generates a signature σ;
- Vfy: on input $(\texttt{pp}, \texttt{spk}, \texttt{vsk}, \vec{v}, m, \sigma)$, where $\texttt{vsk}$ is a verifier's secret key, Vfy checks if σ is a valid signature on message m with respect to signer's public key $\texttt{spk}$ and vector of verifier public keys $\vec{v}$;
- $Forge$: on input $(\texttt{pp}, \texttt{spk}, \vec{v}, m, \vec{s})$, where $\texttt{spk}$ is the signer's public key, $\vec{v}$ is the vector of the designated verifiers' public keys, $\vec{s}$ is a vector of designated verifiers' secret keys—with $|\vec{s}| = |\vec{v}|$ and where for $i \in \{1, \dots, |\vec{v}|\}$, either $s_i = \perp$ or s_i is the secret key corresponding to the i-th public key of $\vec{v}$, i.e. v_i—and m is the message, generates a forged signature σ.

In this section we introduce a new (stronger) Off-The-Record security notion for MDVS schemes capturing the setting where the signer's secret key can leak (Definition 4) and give a new construction satisfying this stronger notion. (In this full-version we also introduce a new Message-Bound Validity notion for MDVS schemes that was not considered in the published version of this paper [5], and without which we do not know how to prove the $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security of the MDRS-PKE scheme from [21].)

4.1 Security Notions

Let $\Pi = (S, G_S, G_V, Sig, Vfy, Forge)$ be an MDVS scheme. The MDVS security games ahead have an implicitly defined security parameter k, and provide adversaries with access to the following oracles:

Public Parameter Generation Oracle: $\mathcal{O}_{PP}$

1. On the first call to $\mathcal{O}_{PP}$, compute $\texttt{pp} \leftarrow S(1^k)$; output $\texttt{pp}$;
2. On subsequent calls, simply output $\texttt{pp}$.

Signer Key-Pair Generation Oracle: $\mathcal{O}_{SK}(A_i)$

1. On the first call to $\mathcal{O}_{SK}$ on input A_i, compute $(\texttt{spk}_i, \texttt{ssk}_i) \leftarrow G_S(\texttt{pp})$, and output $(\texttt{spk}_i, \texttt{ssk}_i)$;
2. On subsequent calls, simply output $(\texttt{spk}_i, \texttt{ssk}_i)$.

Verifier Key-Pair Generation Oracle: $\mathcal{O}_{VK}(B_j)$

1. Analogous to the Signer Key-Pair Generation Oracle.

Signer Public-Key Oracle: $\mathcal{O}_{SPK}(A_i)$

1. $(\texttt{spk}_i, \texttt{ssk}_i) \leftarrow \mathcal{O}_{SK}(A_i)$; output $\texttt{spk}_i$.

Verifier Public-Key Oracle: $\mathcal{O}_{VPK}(B_j)$

1. Analogous to the Signer Public-Key Oracle.

Signing Oracle: $\mathcal{O}_S(A_i, \vec{V}, m)$

1. $(\mathtt{spk}_i, \mathtt{ssk}_i) \leftarrow \mathcal{O}_{SK}(A_i)$;
2. $\vec{v} = (\mathcal{O}_{VPK}(V_1), \ldots, \mathcal{O}_{VPK}(V_{|\vec{V}|}))$;
3. Output $\sigma \leftarrow \mathit{Sig}_{\mathsf{pp}}(\mathtt{ssk}_i, \vec{v}, m)$.

Verification Oracle: $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma)$

1. $\mathtt{spk}_i \leftarrow \mathcal{O}_{SPK}(A_i)$;
2. $\vec{v} = (\mathcal{O}_{VPK}(V_1), \ldots, \mathcal{O}_{VPK}(V_{|\vec{V}|}))$;
3. $(\mathtt{vpk}_j, \mathtt{vsk}_j) \leftarrow \mathcal{O}_{VK}(B_j)$;
4. Output $d \leftarrow \mathit{Vfy}_{\mathsf{pp}}(\mathtt{spk}_i, \mathtt{vsk}_j, \vec{v}, m, \sigma)$, where $d \in \{0, 1\}$.

Definition 1 (Correctness). *Game system $\mathbf{G}^{\mathsf{Corr}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{VK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$. $\mathbf{A}$ wins the game if there are two queries q_S and q_V to $\mathcal{O}_S$ and $\mathcal{O}_V$, respectively, where q_S has input $(A_i, \vec{V}, m)$ and q_V has input $(A_i', B_j, \vec{V}', m', \sigma)$, satisfying $(A_i, \vec{V}, m) = (A_i', \vec{V}', m')$, $B_j \in \vec{V}$, the input σ in q_V is the output of the oracle $\mathcal{O}_S$ on query q_S, and the output of the oracle $\mathcal{O}_V$ on the query q_V is 0. The advantage of $\mathbf{A}$ in winning the Correctness game, denoted $\mathit{Adv}^{\mathsf{Corr}}(\mathbf{A})$, is the probability that $\mathbf{A}$ wins game $\mathbf{G}^{\mathsf{Corr}}$ as described above.*

We say an adversary $\mathbf{A}$ (ε, t)-breaks the (n_V, q_S, q_V)-Correctness of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{VK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$ on at most n_V different verifiers, makes at most q_S and q_V queries to $\mathcal{O}_S$ and $\mathcal{O}_V$, respectively, and satisfies $\mathit{Adv}^{\mathsf{Corr}}(\mathbf{A}) \geq \varepsilon$.

Definition 2 (Consistency). *Game $\mathbf{G}^{\mathsf{Cons}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{VK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$. We say that $\mathbf{A}$ wins the game if it queries $\mathcal{O}_V$ on inputs $(A_i, B_j, \vec{V}, m, \sigma)$ and $(A_i', B_j', \vec{V}', m', \sigma')$ with $(A_i, \vec{V}, m, \sigma) = (A_i', \vec{V}', m', \sigma')$ and where $\{B_j, B_j'\} \subseteq \vec{V}$, the outputs of the two queries differ, and there is no $\mathcal{O}_{VK}$ query on either B_j or B_j'. The advantage of $\mathbf{A}$ in winning the Consistency game, denoted $\mathit{Adv}^{\mathsf{Cons}}(\mathbf{A})$, is the probability that $\mathbf{A}$ wins game $\mathbf{G}^{\mathsf{Cons}}$ as described above.*

An adversary $\mathbf{A}$ (ε, t)-breaks the (n_V, q_V)-Consistency of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{VK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$ on at most n_V different verifiers, makes at most q_V queries to $\mathcal{O}_V$ and satisfies $\mathit{Adv}^{\mathsf{Cons}}(\mathbf{A}) \geq \varepsilon$.

Definition 3 (Unforgeability). *Game system $\mathbf{G}^{\mathsf{Unforg}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{VK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$. $\mathbf{A}$ wins if it makes a query $\mathcal{O}_V(A_i^*, B_j^*, \vec{V}^*, m^*, \sigma^*)$ with $B_j^* \in \vec{V}^*$ that outputs 1, for every query $\mathcal{O}_S(A_i', \vec{V}', m')$, $(A_i^*, \vec{V}^*, m^*) \neq (A_i', \vec{V}', m')$, and there is no $\mathcal{O}_{SK}$ query on A_i^* nor $\mathcal{O}_{VK}$ query on B_j^*. The advantage of $\mathbf{A}$ in winning the Unforgeability game is the probability that $\mathbf{A}$ wins $\mathbf{G}^{\mathsf{Unforg}}$, and is denoted $\mathit{Adv}^{\mathsf{Unforg}}(\mathbf{A})$.*

An adversary $\mathbf{A}$ (ε, t)-breaks the (n_S, n_V, q_S, q_V)-Unforgeability of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{SK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$ on at most n_S different signers, $\mathcal{O}_{VK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$ on at most n_V different verifiers, makes at most q_S and q_V queries to $\mathcal{O}_S$ and $\mathcal{O}_V$, respectively, and satisfies $\mathit{Adv}^{\mathsf{Unforg}}(\mathbf{A}) \geq \varepsilon$.

4.1.1 New Off-The-Record Security Notion. We now present the new enhanced off-the-record security notion for MDVS schemes. As already mentioned, the main difference between our new notion and the existing one (see [8, 21]) is that in our new notion the adversary can query for the secret key of any sender (and still win the game). This is reflected in Definition 4 in that there is no restriction on which signer secret keys an adversary may query.

The off-the-record security notion defines two game systems, $\mathbf{G}_{\mathbf{0}}^{\mathsf{OTR}}$ and $\mathbf{G}_{\mathbf{1}}^{\mathsf{OTR}}$, which provide adversaries with access to a modified oracle $\mathcal{O}_S$ whose behavior varies depending on the underlying game system:

Signing Oracle: $\mathcal{O}_S(\mathtt{type} \in \{\mathtt{sig}, \mathtt{sim}\}, A_i, \vec{V}, m, \mathcal{C})$

For game system $\mathbf{G}_{\mathbf{b}}^{\mathsf{OTR}}$, the oracle behaves as follows:

1. $(\mathtt{spk}_i, \mathtt{ssk}_i) \leftarrow \mathcal{O}_{SK}(A_i)$;
2. Let $\vec{v} = (v_1, \ldots, v_{|\vec{V}|})$ and $\vec{s} = (s_1, \ldots, s_{|\vec{V}|})$, where, for $i \in \{1, \ldots, |\vec{V}|\}$:
   – $(v_i, s_i) = \begin{cases} \mathcal{O}_{VK}(V_i) & \text{if } V_i \in \mathcal{C} \\ (\mathcal{O}_{VPK}(V_i), \bot) & \text{otherwise;} \end{cases}$
3. $(\sigma_{\mathbf{0}}, \sigma_{\mathbf{1}}) \leftarrow (\mathit{\Pi}.\mathit{Sig}_{\mathtt{pp}}(\mathtt{ssk}_i, \vec{v}, m), \mathit{\Pi}.\mathit{Forge}_{\mathtt{pp}}(\mathtt{spk}_i, \vec{v}, m, \vec{s}))$;
4. If $\mathbf{b} = 0$, output $\sigma_{\mathbf{0}}$ if $\mathtt{type} = \mathtt{sig}$ and $\sigma_{\mathbf{1}}$ if $\mathtt{type} = \mathtt{sim}$; otherwise, if $\mathbf{b} = 1$, output $\sigma_{\mathbf{1}}$.

Definition 4 (Off-The-Record). *For $\mathbf{b} \in \{0, 1\}$, game $\mathbf{G}_{\mathbf{b}}^{\mathsf{OTR}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{VK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$. We say that $\mathbf{A}$ wins the game if it outputs a guess bit b' with $b' = \mathbf{b}$, and for every query $\mathcal{O}_S(\mathtt{type}, A_i, \vec{V}, m, \mathcal{C})$: 1. $\mathcal{C} \subseteq \mathit{Set}(\vec{V})$; 2. there is no query $\mathcal{O}_{VK}(B_j)$ with $B_j \in \mathit{Set}(\vec{V}) \setminus \mathcal{C}$; 3. letting σ be the output of the $\mathcal{O}_S$ query above, there is no query $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma)$ with $B_j \in \vec{V}$. The advantage of $\mathbf{A}$ in winning the Off-The-Record security game is*

$$\mathit{Adv}^{\mathsf{OTR}}(\mathbf{A}) := \Big|\Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{\mathsf{OTR}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{\mathsf{OTR}} = \mathtt{win}] - 1\Big|.$$

An adversary $\mathbf{A}$ (ε, t)-breaks the (n_V, d_S, q_S, q_V)-Off-The-Record security of $\mathit{\Pi}$ if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{VK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$ on at most n_V different verifiers, makes at most q_S and q_V queries to $\mathcal{O}_S$ and $\mathcal{O}_V$, respectively, with the sum of the verifier vectors' lengths input to $\mathcal{O}_S$ being at most d_S, and satisfies $\mathit{Adv}^{\mathsf{OTR}}(\mathbf{A}) \geq \varepsilon$.

4.1.2 New Security Notion: Message-Bound Validity. Another guarantee the definitions we have introduced so far do not give is that (honestly generated) MDVS signatures are bound to the single message they were generated for—in the sense that each MDVS signature only verifies as valid for the message it was created for—even if the message is chosen by an adversary who knows the secret key of the signer. Looking ahead, we introduce the Message-Bound Validity security notion because we do not know how to prove the $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ security of our MDRS-PKE construction if the underlying MDVS scheme does not give this guarantee.

Definition 5 (Message-Bound Validity). *Game* $\mathbf{G}^{\mathsf{Bound\text{-}Val}}$ *provides an adversary* $\mathbf{A}$ *with access to oracles* $\mathcal{O}_{PP}$,$\mathcal{O}_{SK}$,$\mathcal{O}_{VK}$,$\mathcal{O}_{SPK}$,$\mathcal{O}_{VPK}$,$\mathcal{O}_S$, *and* $\mathcal{O}_V$. $\mathbf{A}$ *wins the game if there are two queries* q_S *and* q_V *to* $\mathcal{O}_S$ *and* $\mathcal{O}_V$, *respectively, where* q_S *has input* $(A_i, \vec{V}, m)$ *and* q_V *has input* $(A_i', B_j, \vec{V}', m', \sigma)$, *satisfying 1.* $(A_i, \vec{V}) = (A_i', \vec{V}')$; *2.* $B_j \in \vec{V}$; *3.* $m \neq m'$; *4. the input* σ *in* q_V *is* $\mathcal{O}_S$*'s output on query* q_S; *and 5. the output of* $\mathcal{O}_V$ *on query* q_V *is* 1. $\mathbf{A}$*'s advantage is the probability that* $\mathbf{A}$ *wins* $\mathbf{G}^{\mathsf{Bound\text{-}Val}}$, *and is denoted* $Adv^{\mathsf{Bound\text{-}Val}}(\mathbf{A})$.

An adversary $\mathbf{A}$ (ε, t)-breaks the $(n_S, n_V, d_S, q_S, q_V)$-Message-Bound Validity of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{SK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$ on at most n_S different signers, $\mathcal{O}_{VK}$, $\mathcal{O}_{VPK}$, $\mathcal{O}_S$ and $\mathcal{O}_V$ on at most n_V different verifiers, makes at most q_S and q_V queries to $\mathcal{O}_S$ and $\mathcal{O}_V$, respectively, with the sum of the verifier vectors' lengths input to $\mathcal{O}_S$ being at most d_S, and satisfies $Adv^{\mathsf{Bound\text{-}Val}}(\mathbf{A}) \geq \varepsilon$.

Finally, we say that Π is

$$(\varepsilon_{\mathsf{Corr}}, \varepsilon_{\mathsf{Cons}}, \varepsilon_{\mathsf{Unforg}}, \varepsilon_{\mathsf{OTR}}, \varepsilon_{\mathsf{Bound\text{-}Val}}, t, n_S, n_V, d_S, q_S, q_V)\text{-secure}$$

if there is no adversary $\mathbf{A}$ that: 1. $(\varepsilon_{\mathsf{Corr}}, t)$-breaks Π's (n_V, q_S, q_V)-Correctness; 2. $(\varepsilon_{\mathsf{Cons}}, t)$-breaks Π's (n_V, q_V)-Consistency; 3. $(\varepsilon_{\mathsf{Unforg}}, t)$-breaks Π's (n_S, n_V, q_S, q_V)-Unforgeability; 4. $(\varepsilon_{\mathsf{OTR}}, t)$-breaks Π's (n_V, d_S, q_S, q_V)-Off-The-Record; or 5. $(\varepsilon_{\mathsf{Bound\text{-}Val}}, t)$-breaks Π's $(n_S, n_V, d_S, q_S, q_V)$-Message-Bound Validity.

4.2 DVS Construction

We present our MDVS construction incrementally.[8] We begin by giving a construction of a (single verifier) DVS scheme (see Algorithm 1) that is Correct (Definition 1), Unforgeable (Definition 3) and Off-The-Record (Definition 4); next, we generalize it into an MDVS scheme (which has to additionally satisfy consistency); finally, we use a technique first introduced by Bader et al. in [1] to make the scheme tightly secure under adaptive corruptions. The building blocks for all our constructions are a NIZK scheme $\Pi_{\mathrm{NIZK}} = (G, P, V, S \coloneqq (S_G, S_P))$, a PKE scheme $\Pi_{\mathrm{PKE}} = (G, E, D)$, and a One Way Function $\Pi_{\mathrm{OWF}} = (S, F)$.

For modularity, rather than introducing a single language/relation for the NIZK scheme used by our constructions, we will introduce different relations and then define the relation/language for our constructions as the intersection of these relations. For example, in Algorithm 1 we consider the language induced by a relation $R_{\mathrm{DVS}} \coloneqq R_{\mathrm{DVS\text{-}Match}} \cap R_{\mathrm{DVS\text{-}Unforg}}$, where

- $R_{\mathrm{DVS\text{-}Match}} \coloneqq \Big\{ \big((\mathtt{pk_{pp}}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}), (a, b, r, r_{\mathtt{pp}})\big) \mid$
  $\Big(c_{\mathtt{pp}} = \Pi_{\mathrm{PKE}}.E_{\mathtt{pk_{pp}}}\big((m, b, a); r_{\mathtt{pp}}\big)\Big) \wedge \big(c = \Pi_{\mathrm{PKE}}.E_{\mathtt{vpk.pk}}(b; r)\big) \Big\}$;
- $R_{\mathrm{DVS\text{-}Unforg}} \coloneqq \Big\{ \big((\mathtt{pk_{pp}}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}), (a, b, r, r_{\mathtt{pp}})\big) \mid$
  $(b = 1) \rightarrow (\Pi_{\mathrm{OWF}}.F(a) \in \{\mathtt{spk}.y, \mathtt{vpk}.y\}) \Big\}$.

[8] We only prove the security of the final MDVS construction given in Section 4.4.

The corresponding language is then defined as $L_{\text{DVS}} := \{(\mathtt{pk}_{\mathtt{pp}}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}) \mid \exists (a, b, r, r_{\mathtt{pp}}) : \big((\mathtt{pk}_{\mathtt{pp}}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}), (a, b, r, r_{\mathtt{pp}})\big) \in R_{\text{DVS}}\}$.

Algorithm 1 DVS scheme construction $\Pi_{\text{DVS}} = (S, G_S, G_V, \mathit{Sig}, \mathit{Vfy}, \mathit{Forge})$.

$S(1^k)$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\text{PKE}}.G(1^k)$
  return $\mathtt{pp} := (1^k, \mathtt{crs} \leftarrow \Pi_{\text{NIZK}}.G(1^k), \mathtt{pk})$

$G_S(\mathtt{pp})$
  $x \leftarrow \Pi_{\text{OWF}}.S(1^k)$
  return $(\mathtt{spk} := \Pi_{\text{OWF}}.F(x), \mathtt{ssk} := (\mathtt{spk}, x))$

$G_V(\mathtt{pp})$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\text{PKE}}.G(1^k)$
  $x \leftarrow \Pi_{\text{OWF}}.S(1^k)$
  return $(\mathtt{vpk} := (\Pi_{\text{OWF}}.F(x), \mathtt{pk}), \mathtt{vsk} := (\mathtt{vpk}, \mathtt{sk}, x))$

$\mathit{Sig}_{\mathtt{pp}}(\mathtt{ssk}, \mathtt{vpk}, m)$
  $c \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{vpk.pk}}(1; r)$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}\big((m, 1, \mathtt{ssk}.x); r_{\mathtt{pp}}\big)$
  $p \leftarrow \Pi_{\text{NIZK}}.P_{\mathtt{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}) \in L_{\text{DVS}}, (\mathtt{ssk}.x, 1, r, r_{\mathtt{pp}})\big)$
  return $\sigma := (p, c, c_{\mathtt{pp}})$

$\mathit{Vfy}_{\mathtt{pp}}(\mathtt{spk}, \mathtt{vsk}, m, \sigma := (p, c, c_{\mathtt{pp}}))$
  $b \leftarrow \Pi_{\text{NIZK}}.V_{\mathtt{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}) \in L_{\text{DVS}}, p\big)$
  return $b \wedge \Pi_{\text{PKE}}.D_{\mathtt{vsk.sk}}(c)$

$\mathit{Forge}_{\mathtt{pp}}(\mathtt{spk}, \mathtt{vpk}, m, \mathtt{vsk})$
  if $\mathtt{vsk} \neq \bot$ **then** ▷ Forge using verifier's secret key.
    $c \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{vpk.pk}}(1; r)$
    $c_{\mathtt{pp}} \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}\big((m, 1, \mathtt{vsk}.x); r_{\mathtt{pp}}\big)$
    $p \leftarrow \Pi_{\text{NIZK}}.P_{\mathtt{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}) \in L_{\text{DVS}}, (\mathtt{vsk}.x, 1, r, r_{\mathtt{pp}})\big)$
  else ▷ Forge without using verifier's secret key.
    $c \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{vpk.pk}}(0; r)$
    $c_{\mathtt{pp}} \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}\big((m, 0, 0); r_{\mathtt{pp}}\big)$
    $p \leftarrow \Pi_{\text{NIZK}}.P_{\mathtt{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \mathtt{vpk}, m, c, c_{\mathtt{pp}}) \in L_{\text{DVS}}, (0, 0, r, r_{\mathtt{pp}})\big)$
  return $\sigma := (p, c, c_{\mathtt{pp}})$

In our scheme a signature consists of two ciphertexts, c and $c_{\mathtt{pp}}$, together with a NIZK proof p which is the key for guaranteeing signature unforgeability. Informally, Π_{NIZK}'s soundness guarantees that, on one hand, since $R_{\text{DVS}} \subseteq R_{\text{DVS-Match}}$, ciphertexts $c_{\mathtt{pp}}$ and c encrypt the same bit b, and on the other hand, since $R_{\text{DVS}} \subseteq R_{\text{DVS-Unforg}}$, if this bit b is 1 (in which case the signature verification succeeds), $c_{\mathtt{pp}}$ encrypts either the signer's or the verifier's secret key. Regarding Message-Bound Validity, Π_{PKE}'s correctness together with Π_{NIZK}'s soundness guarantee that the ciphertext $c_{\mathtt{pp}}$ of each signature is bound to a single message (and thus so is the signature).

4.3 A Conceptually Simple MDVS Construction

We now show how to generalize the DVS scheme from before into an MDVS scheme. Our MDVS scheme construction is defined in Algorithm 2 and is analogous to the DVS scheme from before, but adapted to the multi-verifier case. The main difference is that MDVS schemes need to guarantee consistency.

In the following, let $\vec{\alpha} := ((b_1, a_1), \ldots, (b_{|\vec{\alpha}|}, a_{|\vec{\alpha}|}))$; we assume for simplicity that all vectors have matching lengths, i.e. $|\vec{v}| = |\vec{c}| = |\vec{\alpha}|$.

- $R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Match}} := \Big\{ \big((\mathtt{pk}_{\mathtt{pp}}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) :$
$$\Big[c_{\mathtt{pp}} = \Pi_{\mathrm{PKE}}.E_{\mathtt{pk}_{\mathtt{pp}}}\big((m, b, \vec{\alpha}); r_{\mathtt{pp}}\big)\Big] \wedge \Big[\bigwedge_{i \in \{1,\ldots,|\vec{v}|\}} \big(c_i = \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}}(b_i; r_i)\big)\Big]\Big\}$$
- $R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Unforg}} := \Big\{ \big((\mathtt{pk}_{\mathtt{pp}}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) :$
$$\bigwedge_{i \in \{1,\ldots,|\vec{v}|\}} \Big((b_i = 1) \rightarrow (\Pi_{\mathrm{OWF}}.F(a_i) \in \{\mathtt{spk}.y, v_i.y\})\Big)\Big\}$$
- $R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Cons}} := \Big\{ \big((\mathtt{pk}_{\mathtt{pp}}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) :$
$$\bigwedge_{i \in \{1,\ldots,|\vec{v}|\}} \Big((\Pi_{\mathrm{OWF}}.F(a_i) \neq v_i.y) \rightarrow (b_i = b)\Big)\Big\}.$$

Similarly to R_{DVS}, and for the sake of modularity, we define relation $R_{\mathrm{MDVS}^{\mathsf{static}}}$ as $R_{\mathrm{MDVS}^{\mathsf{static}}} := R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Match}} \cap R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Unforg}} \cap R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Cons}}$. In Algorithm 2, we consider the respective induced language $L_{\mathrm{MDVS}^{\mathsf{static}}} := \{(\mathtt{pk}_{\mathtt{pp}}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \mid \exists(\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b) : \big((\mathtt{pk}_{\mathtt{pp}}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) \in R_{\mathrm{MDVS}^{\mathsf{static}}}\}$.

Note that, since $R_{\mathrm{MDVS}^{\mathsf{static}}} \subseteq R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Match}} \cap R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Unforg}}$, Π_{NIZK}'s soundness guarantees that if for any $i \in \{1, \ldots, |\vec{v}|\}$, c_i is an encryption of 1, then $c_{\mathtt{pp}}$ contains either the signer's secret key or the i-th verifier's secret key. Similarly, since $R_{\mathrm{MDVS}^{\mathsf{static}}} \subseteq R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Match}} \cap R_{\mathrm{MDVS}^{\mathsf{static}}\text{-Cons}}$, Π_{NIZK}'s soundness implies that every designated verifier B_j whose secret key is not in $c_{\mathtt{pp}}$'s underlying plaintext will agree on whether the signature is valid.

4.4 Achieving Tight Security under Adaptive Corruptions

We now show how to transform the MDVS scheme from before into one that is tightly secure under adaptive corruptions. The main challenge here is finding a way to embed the challenges from the security games of the underlying PKE and OWF building blocks into the reductions (in such a way that the reduction is tight on the security of the underlying building blocks) while still being able to answer queries for the secret keys of signers and/or verifiers. To achieve this, we rely on a technique that was first introduced in [1]. Essentially, for each party two key-pairs are now sampled; the party's public key are the public keys of each of the underlying key-pairs, and the secret key is the secret key of one (and only one) of these key-pairs. This allows answering secret key queries by the adversary while still being able to embed challenges from the underlying security games into reductions.

Algorithm 2 $\Pi^{\mathsf{stat}}_{\mathrm{MDVS}}$.

$S(1^k)$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\mathrm{PKE}}.G(1^k)$
  return $\mathtt{pp} := (1^k, \mathtt{crs} \leftarrow \Pi_{\mathrm{NIZK}}.G(1^k), \mathtt{pk})$

$G_S(\mathtt{pp})$
  $x \leftarrow \Pi_{\mathrm{OWF}}.S(1^k)$
  return $(\mathtt{spk} := \Pi_{\mathrm{OWF}}.F(x), \mathtt{ssk} := (\mathtt{spk}, x))$

$G_V(\mathtt{pp})$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\mathrm{PKE}}.G(1^k)$
  $x \leftarrow \Pi_{\mathrm{OWF}}.S(1^k)$
  return $(\mathtt{vpk} := (\Pi_{\mathrm{OWF}}.F(x), \mathtt{pk}), \mathtt{vsk} := (\mathtt{vpk}, \mathtt{sk}, x))$

$Sig_{\mathtt{pp}}(\mathtt{ssk}, \vec{v} := (\mathtt{vpk}_1, \dots, \mathtt{vpk}_{|\vec{v}|}), m)$
  for each $i \in \{1, \dots, |\vec{v}|\}$ **do**
    $c_i \leftarrow \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}}(1; r_i)$
  $(\vec{c}, \vec{r}) \leftarrow ((c_1, \dots, c_{|\vec{v}|}), (r_1, \dots, r_{|\vec{v}|}))$
  $\vec{\alpha} \leftarrow (\alpha_1 := (1, \mathtt{ssk}.x), \dots, \alpha_{|\vec{v}|} = (1, \mathtt{ssk}.x))$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\mathrm{PKE}}.E_{\mathtt{pp.pk}}((m, 1, \vec{\alpha}); r_{\mathtt{pp}})$
  $p \leftarrow \Pi_{\mathrm{NIZK}}.P_{\mathtt{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{static}}}, (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, 1)\big)$
  return $\sigma := (p, \vec{c}, c_{\mathtt{pp}})$

$Vfy_{\mathtt{pp}}(\mathtt{spk}, \mathtt{vsk}, \vec{v}, m, \sigma := (p, \vec{c}, c_{\mathtt{pp}}))$
  if $\Pi_{\mathrm{NIZK}}.V_{\mathtt{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{static}}}, p\big) = 1$ **then**
    for $i = 1, \dots, |\vec{v}|$ **do**
      if $\mathtt{vsk.vpk} = v_i$ **then**
        return $\Pi_{\mathrm{PKE}}.D_{\mathtt{vsk.sk}}(c_i)$
  return 0

$Forge_{\mathtt{pp}}(\mathtt{spk}, \vec{v}, m, \vec{s} := (\mathtt{vsk}_1, \dots, \mathtt{vsk}_{|\vec{v}|}))$
  for each $i \in \{1, \dots, |\vec{v}|\}$ **do**
    if $s_i \neq \bot$ **then**
      $c_i \leftarrow \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}}(1; r_i)$
      $\alpha_i \leftarrow (1, s_i.x)$
    else
      $c_i \leftarrow \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}}(0; r_i)$
      $\alpha_i \leftarrow (0, 0)$
  $(\vec{c}, \vec{r}) \leftarrow ((c_1, \dots, c_{|\vec{v}|}), (r_1, \dots, r_{|\vec{v}|}))$
  $\vec{\alpha} \leftarrow (\alpha_1, \dots, \alpha_{|\vec{v}|})$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\mathrm{PKE}}.E_{\mathtt{pp.pk}}((m, 0, \vec{\alpha}); r_{\mathtt{pp}})$
  $p \leftarrow \Pi_{\mathrm{NIZK}}.P_{\mathtt{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{static}}}, (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, 0)\big)$
  return $\sigma := (p, \vec{c}, c_{\mathtt{pp}})$

Let $\vec{\alpha} \coloneqq ((b_1, a_1), \ldots, (b_{|\vec{\alpha}|}, a_{|\vec{\alpha}|}))$; in the following, vectors are assumed to have matching lengths:

- $R_{\mathrm{MDVS}^{\mathsf{adap}}\text{-Match}} \coloneqq \Big\{ \big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) :$
  $\big(c_{\mathtt{pp}} = \Pi_{\mathrm{PKE}}.E_{\mathtt{pp.pk}}((m, b, \vec{\alpha}); r_{\mathtt{pp}})\big) \bigwedge$
  $\Big[\bigwedge_{i \in \{1,\ldots,|\vec{v}|\}} \Big((c_{i,0} = \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_0}(b_i; r_{i,0})) \wedge (c_{i,1} = \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_1}(b_i; r_{i,1})) \Big) \Big] \Big\}$
- $R_{\mathrm{MDVS}^{\mathsf{adap}}\text{-Unforg}} \coloneqq \Big\{ \big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) :$
  $\bigwedge_{i \in \{1,\ldots,|\vec{\alpha}|\}} \Big((b_i = 1) \rightarrow (\Pi_{\mathrm{OWF}}.F(a_i) \in \{\mathtt{spk}.y_0, \mathtt{spk}.y_1, v_i.y_0, v_i.y_1\}) \Big) \Big\}$
- $R_{\mathrm{MDVS}^{\mathsf{adap}}\text{-Cons}} \coloneqq \Big\{ \big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) :$
  $\bigwedge_{i \in \{1,\ldots,|\vec{\alpha}|\}} \Big((\Pi_{\mathrm{OWF}}.F(a_i) \notin \{v_i.y_0, v_i.y_1\}) \rightarrow (b_i = b) \Big) \Big\}.$

As in Section 4.3, we define $R_{\mathrm{MDVS}^{\mathsf{adap}}} \coloneqq R_{\mathrm{MDVS}^{\mathsf{adap}}\text{-Match}} \cap R_{\mathrm{MDVS}^{\mathsf{adap}}\text{-Unforg}} \cap R_{\mathrm{MDVS}^{\mathsf{adap}}\text{-Cons}}$; in Algorithm 3, we consider the language $L_{\mathrm{MDVS}^{\mathsf{adap}}}$ that is induced by $R_{\mathrm{MDVS}^{\mathsf{adap}}}$, which is defined as: $L_{\mathrm{MDVS}^{\mathsf{adap}}} \coloneqq \{(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \mid \exists(\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b) : \big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}), (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, b)\big) \in R_{\mathrm{MDVS}^{\mathsf{adap}}}\}$.

4.4.1 Security Analysis of $\Pi^{\mathsf{adap}}_{\mathbf{MDVS}}$ The theorem below gives an informal summary of our construction's security properties. The formal security theorems (and the corresponding full proofs) are in the appendix (see Section B.2).

Theorem 1 (Informal). *If Π_{PKE} is correct and tightly multi-user and multi-challenge* IND-CPA *secure under* non-adaptive *corruptions, Π_{NIZK} is complete, sound, tightly multi-statement adaptive zero-knowledge and tightly multi-statement simulation sound, and Π_{OWF} is tightly multi-instance secure under* non-adaptive *corruptions, then $\Pi^{\mathsf{adap}}_{\mathrm{MDVS}}$ is tightly:*

1. *correct (Theorem 4);*
2. *consistent under adaptive corruptions (Theorem 5);*
3. *unforgeable under adaptive corruptions (Theorem 6);*
4. *off-the-record under adaptive corruptions (Theorem 7); and*
5. *message-bound validity secure under adaptive corruptions (Theorem 8).*

4.4.2 On Efficiently Instantiating the NIZK Relations All the relations we consider consist of checking a number of equations over a pairing-friendly group, when implemented with suitably algebraic primitives. (For instance, we can use ElGamal [9] as the PKE scheme, and a pairing with one fixed input as the

Algorithm 3 The $\Pi^{\mathsf{adap}}_{\mathrm{MDVS}}$ MDVS scheme.

$S(1^k)$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\mathrm{PKE}}.G(1^k)$
  return $\mathtt{pp} \coloneqq (1^k, \mathtt{crs} \leftarrow \Pi_{\mathrm{NIZK}}.G(1^k), \mathtt{pk})$

$G_S(\mathtt{pp})$
  $(x_0, x_1) \leftarrow (\Pi_{\mathrm{OWF}}.S(1^k), \Pi_{\mathrm{OWF}}.S(1^k))$
  $(y_0, y_1) \leftarrow (\Pi_{\mathrm{OWF}}.F(x_0), \Pi_{\mathrm{OWF}}.F(x_1))$
  $b \leftarrow \mathit{RandomCoin}$
  return $(\mathtt{spk} \coloneqq (y_0, y_1), \mathtt{ssk} \coloneqq (\mathtt{spk}, x \coloneqq x_b))$

$G_V(\mathtt{pp})$
  $((\mathtt{pk}_0, \mathtt{sk}_0), (\mathtt{pk}_1, \mathtt{sk}_1)) \leftarrow (\Pi_{\mathrm{PKE}}.G(1^k), \Pi_{\mathrm{PKE}}.G(1^k))$
  $(x_0, x_1) \leftarrow (\Pi_{\mathrm{OWF}}.S(1^k), \Pi_{\mathrm{OWF}}.S(1^k))$
  $(y_0, y_1) \leftarrow (\Pi_{\mathrm{OWF}}.F(x_0), \Pi_{\mathrm{OWF}}.F(x_1))$
  $b \leftarrow \mathit{RandomCoin}$
  return $(\mathtt{vpk} \coloneqq (\mathtt{pk}_0, y_0, \mathtt{pk}_1, y_1), \mathtt{vsk} \coloneqq (\mathtt{vpk}, b, \mathtt{sk} \coloneqq \mathtt{sk}_b, x \coloneqq x_b))$

$\mathit{Sig}_{\mathtt{pp}}(\mathtt{ssk}, \vec{v} \coloneqq (\mathtt{vpk}_1, \ldots, \mathtt{vpk}_{|\vec{v}|}), m)$
  for each $i \in \{1, \ldots, |\vec{v}|\}$ **do**
    $(c_{i,0}, c_{i,1}) \leftarrow (\Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_0}(1; r_{i,0}), \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_1}(1; r_{i,1}))$
  $(\vec{c}, \vec{r}) \leftarrow (((c_{1,0}, c_{1,1}), \ldots, (c_{|\vec{v}|,0}, c_{|\vec{v}|,1})), ((r_{1,0}, r_{1,1}), \ldots, (r_{|\vec{v}|,0}, r_{|\vec{v}|,1})))$
  $\vec{\alpha} \leftarrow (\alpha_1 \coloneqq (1, \mathtt{ssk}.x), \ldots, \alpha_{|\vec{v}|} \coloneqq (1, \mathtt{ssk}.x))$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\mathrm{PKE}}.E_{\mathtt{pp}.\mathtt{pk}}((m, 1, \vec{\alpha}); r_{\mathtt{pp}})$
  $p \leftarrow \Pi_{\mathrm{NIZK}}.P_{\mathtt{crs}}\big((\mathtt{pp}.\mathtt{pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}, (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, 1)\big)$
  return $\sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}})$

$\mathit{Vfy}_{\mathtt{pp}}(\mathtt{spk}, \mathtt{vsk}, \vec{v}, m, \sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}}))$
  if $\Pi_{\mathrm{NIZK}}.V_{\mathtt{crs}}\big((\mathtt{pp}.\mathtt{pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}, p\big) = 1$ **then**
    for $i = 1, \ldots, |\vec{v}|$ **do**
      if $\mathtt{vsk}.\mathtt{vpk} = v_i$ **then**
        return $\Pi_{\mathrm{PKE}}.D_{\mathtt{vsk}.\mathtt{sk}}(c_{i,\mathtt{vsk}.b})$
  return 0

$\mathit{Forge}_{\mathtt{pp}}(\mathtt{spk}, \vec{v}, m, \vec{s} \coloneqq (\mathtt{vsk}_1, \ldots, \mathtt{vsk}_{|\vec{v}|}))$
  for each $i \in \{1, \ldots, |\vec{v}|\}$ **do**
    if $s_i \neq \bot$ **then**
      $(c_{i,0}, c_{i,1}) \leftarrow (\Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_0}(1; r_{i,0}), \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_1}(1; r_{i,1}))$
      $\alpha_i \coloneqq (1, s_i.x)$
    else
      $(c_{i,0}, c_{i,1}) \leftarrow (\Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_0}(0; r_{i,0}), \Pi_{\mathrm{PKE}}.E_{v_i.\mathtt{pk}_1}(0; r_{i,1}))$
      $\alpha_i \coloneqq (0, 0)$
  $(\vec{c}, \vec{r}) \leftarrow (((c_{1,0}, c_{1,1}), \ldots, (c_{|\vec{v}|,0}, c_{|\vec{v}|,1})), ((r_{1,0}, r_{1,1}), \ldots, (r_{|\vec{v}|,0}, r_{|\vec{v}|,1})))$
  $\vec{\alpha} \leftarrow (\alpha_1, \ldots, \alpha_{|\vec{v}|})$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\mathrm{PKE}}.E_{\mathtt{pp}.\mathtt{pk}}((m, 0, \vec{\alpha}); r_{\mathtt{pp}})$
  $p \leftarrow \Pi_{\mathrm{NIZK}}.P_{\mathtt{crs}}\big((\mathtt{pp}.\mathtt{pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}, (\vec{\alpha}, \vec{r}, r_{\mathtt{pp}}, 0)\big)$
  return $\sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}})$

One Way Function.) Then, we can use a simulation-sound variant of Groth-Sahai proofs [12, 13] as a compatible NIZK scheme to prove these relations. This yields proofs that are only linear-sized in the number of witness variables and equations. Of course, this will result in an unoptimized solution that may not be quite practical yet.

5 PKEBC Scheme with Linear Ciphertext Size and Decryption Time

A PKEBC scheme Π is a quadruple $\Pi = (S, G, E, D)$ of PPTs, where:

- S: on input 1^k, generates public parameters $\texttt{pp}$;
- G: on input $\texttt{pp}$, generates a receiver key-pair $(\texttt{pk}, \texttt{sk})$;
- E: on input $(\texttt{pp}, \vec{v}, m)$, where $\vec{v}$ is a vector of public keys of the intended receivers and m is the message, generates a ciphertext c;
- D: on input $(\texttt{pp}, \texttt{sk}, c)$, where $\texttt{sk}$ is the receiver's secret key, D decrypts c using $\texttt{sk}$, and outputs the decrypted receiver-vector/message pair $(\vec{v}, m)$ (or $\perp$ if the ciphertext did not decrypt correctly).

In this section we introduce new security notions capturing the security of PKEBC schemes under adaptive corruptions and give a new construction of a PKEBC scheme that not only is tightly secure under these stronger notions, but also for which both the ciphertext size and the decryption time only grow linearly with the number of receivers.

5.1 Security Notions for Adaptive Corruptions

The security notions we now introduce are a strengthening of the original ones introduced by Maurer et al. in [21], but capturing the security of PKEBC schemes under adaptive corruptions. More concretely, in the Correctness, Robustness and Consistency notions adversaries are now allowed to query for the secret keys of any receiver and still win the game; in the $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security games—a combination of the original IND-CCA-2 and IK-CCA-2 security notions [21] capturing adaptive corruptions—adversaries can now corrupt parties adaptively. (Our $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security notion can also be interpreted as a variant of the notion introduced by Lee et al. in [16]—which captures the IND-CCA-2 security of PKE schemes under adaptive corruptions—but adapted for PKEBC schemes and also capturing anonymity.)

We now introduce some oracles that the game systems ahead provide to the adversaries. In the following, consider a PKEBC scheme $\Pi = (S, G, E, D)$ with message space $\mathcal{M}$. The oracles below are defined for a game-system with (an implicitly defined) security parameter k:

Public Parameters Oracle: $\mathcal{O}_{PP}$

1. On the first call, compute and store $\texttt{pp} \leftarrow S(1^k)$; output $\texttt{pp}$;
2. On subsequent calls, output the previously generated $\texttt{pp}$.

Secret Key Generation Oracle: $\mathcal{O}_{SK}(B_j)$

1. If $\mathcal{O}_{SK}$ was queried on B_j before, simply look up and return the previously generated key for B_j;
2. Otherwise, store $(\mathtt{pk}_j, \mathtt{sk}_j) \leftarrow G(\mathtt{pp})$ as B_j's key-pair, and output $(\mathtt{pk}_j, \mathtt{sk}_j)$.

Public Key Generation Oracle: $\mathcal{O}_{PK}(B_j)$

1. $(\mathtt{pk}_j, \mathtt{sk}_j) \leftarrow \mathcal{O}_{SK}(B_j)$;
2. Output $\mathtt{pk}_j$.

Encryption Oracle: $\mathcal{O}_E(\vec{V}, m)$

1. $\vec{v} \leftarrow (\mathcal{O}_{PK}(V_1), \ldots, \mathcal{O}_{PK}(V_{|\vec{V}|}))$;
2. Create and output a fresh encryption $c \leftarrow E_{\mathsf{pp},\vec{v}}(m)$.

Decryption Oracle: $\mathcal{O}_D(B_j, c)$

1. Query $\mathcal{O}_{SK}(B_j)$ to obtain the corresponding secret-key $\mathtt{sk}_j$;
2. Decrypt c using $\mathtt{sk}_j$, $(\vec{v}, m) \leftarrow D_{\mathsf{pp},\mathtt{sk}_j}(c)$, and then output the resulting receivers-message pair $(\vec{v}, m)$, or $\bot$ (if $(\vec{v}, m) = \bot$, i.e. the ciphertext is not valid with respect to B_j's secret key).

Definition 6 (Correctness). *Game* $\mathbf{G}^{\mathsf{Corr}}$ *provides an adversary* $\mathbf{A}$ *with access to oracles* $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{PK}$, $\mathcal{O}_E$ *and* $\mathcal{O}_D$. $\mathbf{A}$ *wins the game if there are two queries* q_E *and* q_D *to* $\mathcal{O}_E$ *and* $\mathcal{O}_D$, *respectively, where* q_E *has input* $(\vec{V}, m)$ *and* q_D *has input* (B_j, c), *satisfying* $B_j \in \vec{V}$, *the input* c *in* q_D *is the output of* q_E, *and the output of* q_D *is either* $\bot$ *or* $(\vec{v}', m')$ *with* $(\vec{v}, m) \neq (\vec{v}', m')$. *The advantage of* $\mathbf{A}$ *in winning the Correctness game, denoted* $Adv^{\mathsf{Corr}}(\mathbf{A})$, *is the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{Corr}}$ *as described above.*

An adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{Corr}}, t)$-breaks the (n, d_E, q_E, q_D)-Correctness of a PKEBC scheme Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{SK}$, $\mathcal{O}_{PK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$ on at most n different parties, makes at most q_E and q_D queries to $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, with the sum of lengths of the party vectors input to $\mathcal{O}_E$ being at most d_E, and satisfies $Adv^{\mathsf{Corr}}(\mathbf{A}) \geq \varepsilon_{\mathsf{Corr}}$.

Definition 7 (Robustness). *Game* $\mathbf{G}^{\mathsf{Rob}}$ *provides an adversary* $\mathbf{A}$ *with access to oracles* $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{PK}$, $\mathcal{O}_E$ *and* $\mathcal{O}_D$. $\mathbf{A}$ *wins the game if there are two queries* q_E *and* q_D *to* $\mathcal{O}_E$ *and* $\mathcal{O}_D$, *respectively, where* q_E *has input* $(\vec{V}, m)$ *and* q_D *has input* (B_j, c), *satisfying* $B_j \notin \vec{V}$, *the input* c *in* q_D *is the output of* q_E, *and the output of* q_D *is* $(\vec{v}', m')$ *with* $(\vec{v}', m') \neq \bot$. *The advantage of* $\mathbf{A}$ *in winning the Robustness game is the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{Rob}}$ *as described above, and is denoted* $Adv^{\mathsf{Rob}}(\mathbf{A})$.

An adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{Rob}}, t)$-breaks the Robustness of a PKEBC scheme Π if $\mathbf{A}$ runs in time at most t and satisfies $Adv^{\mathsf{Rob}}(\mathbf{A}) \geq \varepsilon_{\mathsf{Rob}}$.

Definition 8 (Consistency). *Game* $\mathbf{G}^{\mathsf{Cons}}$ *provides an adversary* $\mathbf{A}$ *with access to oracles* $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{PK}$ *and* $\mathcal{O}_D$. $\mathbf{A}$ *wins the game if there is a ciphertext* c *such that* $\mathcal{O}_D$ *is queried on inputs* (B_i, c) *and* (B_j, c) *for some* B_i *and* B_j *(possibly with* $B_i = B_j$*), query* $\mathcal{O}_D(B_i, c)$ *outputs some* $(\vec{v}, m)$ *satisfying* $(\vec{v}, m) \neq \bot$ *with* $\mathtt{pk}_j \in \vec{v}$ *(where* $\mathtt{pk}_j$ *is* B_j*'s public key), and query* $\mathcal{O}_D(B_j, c)$ *does not output* $(\vec{v}, m)$. *The advantage of* $\mathbf{A}$ *in winning the Consistency game is denoted* $Adv^{\mathsf{Cons}}(\mathbf{A})$ *and corresponds to the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{Cons}}$.

We say that an adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{Cons}}, t)$-breaks the (n, q_D)-Consistency of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{SK}$, $\mathcal{O}_{PK}$ and $\mathcal{O}_D$ on at most n different parties, makes at most q_D queries to $\mathcal{O}_D$ and satisfies $Adv^{\mathsf{Cons}}(\mathbf{A}) \geq \varepsilon_{\mathsf{Cons}}$.

Below we present the definition of $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ security. This notion is a combination of the original IND-CCA-2 and IK-CCA-2 security notions introduced in [21] that captures adaptive security (i.e. the adversary is allowed to corrupt parties adaptively). The games defined by this definition provide adversaries with access to the oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$ and $\mathcal{O}_{PK}$ defined above, as well as to oracles $\mathcal{O}_E$ and $\mathcal{O}_D$ defined below:

Encryption Oracle: $\mathcal{O}_E\big((\vec{V}_0, m_0), (\vec{V}_1, m_1)\big)$

1. For game system $\mathbf{G}_{\mathbf{b}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}$, encrypt $m_{\mathbf{b}}$ under $\vec{v}_{\mathbf{b}}$, the vector of public keys corresponding to $\vec{V}_{\mathbf{b}}$; output c.

Decryption Oracle: $\mathcal{O}_D(B_j, c)$

1. If c was the output of some query to $\mathcal{O}_E$, output `test`;
2. Otherwise, compute and output $(\vec{v}, m) \leftarrow D_{\mathtt{pp},\mathtt{sk}_j}(c)$, where $\mathtt{sk}_j$ is B_j's secret key.

Definition 9 ($(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ Security). *For* $\mathbf{b} \in \{0, 1\}$, *game system* $\mathbf{G}_{\mathbf{b}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}$ *provides an adversary* $\mathbf{A}$ *with access to oracles* $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{PK}$, $\mathcal{O}_E$ *and* $\mathcal{O}_D$. $\mathbf{A}$ *wins the game if it outputs a guess bit* b' *satisfying* $b' = \mathbf{b}$ *and for every query* $\mathcal{O}_E\big((\vec{V}_0, m_0), (\vec{V}_1, m_1)\big)$*: 1.* $|\vec{V}_0| = |\vec{V}_1|$*; 2.* $|m_0| = |m_1|$*; and 3. there is no query to* $\mathcal{O}_{SK}$ *on any* $B_j \in \mathit{Set}(\vec{V}_0) \cup \mathit{Set}(\vec{V}_1)$ *at any point during the game. We define the advantage of* $\mathbf{A}$ *in winning the* $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ *game as*

$$Adv^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}(\mathbf{A}) := \left|\Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}} = \mathtt{win}] - 1\right|.$$

We say that an adversary $\mathbf{A}$ (ε, t)-breaks the (n, d_E, q_E, q_D)-$(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ security of Π if $\mathbf{A}$ runs in time at most t, queries the oracles it has access to on at most n different parties, makes at most q_E and q_D queries to oracles $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, with the sum of lengths of all the party vectors input to $\mathcal{O}_E$ being at most d_E, and satisfies $Adv^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}(\mathbf{A}) \geq \varepsilon$. Finally, we say that Π is

$$(\varepsilon_{\mathsf{Corr}}, \varepsilon_{\mathsf{Rob}}, \varepsilon_{\mathsf{Cons}}, \varepsilon_{(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}, t, n, d_E, q_E, q_D, \mathsf{adap})\text{-secure},$$

if there is no adversary $\mathbf{A}$ that: 1. $(\varepsilon_{\mathsf{Corr}}, t)$-breaks Π's (n, d_E, q_E, q_D)-Correctness; 2. $(\varepsilon_{\mathsf{Rob}}, t)$-breaks Π's Robustness; 3. $(\varepsilon_{\mathsf{Cons}}, t)$-breaks Π's (n, q_D)-Consistency; or 4. $(\varepsilon_{(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}, t)$-breaks Π's (n, d_E, q_E, q_D)-$(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ security.

5.2 Achieving Linear Ciphertext Size

As before, we present our PKEBC construction incrementally (and only prove the security of the final PKEBC construction given Section 5.4). Our first PKEBC scheme is defined in Algorithm 4. Like Maurer et al.'s scheme [21], our construction is a generalization of Naor-Yung's PKE scheme for multiple receivers (see [22]). However, while Maurer et al.'s scheme encrypts, for each receiver, the vector of all receivers' public keys plus the message—leading not only to quadratic sized ciphertexts but also to quadratic encryption and decryption time—our scheme instead relies on a SKE scheme Π_{SKE} to encrypt the vector of all receivers plus the message under a key k that is then encrypted under each receiver's public key, resembling the hybrid encryption technique [25]. Furthermore, while Maurer et al.'s construction relies on a binding commitment scheme in order to achieve consistency, our scheme instead uses a PKE scheme: note that as long as a PKE key-pair $(\mathtt{pk}, \mathtt{sk})$ is sampled honestly, by the correctness of the PKE scheme, the encryption of any message m under $\mathtt{pk}$ also works as a commitment to m.[9] The building blocks of this first scheme consist of a PKE scheme $\Pi_{\text{PKE}} = (G, E, D)$, a SKE scheme $\Pi_{\text{SKE}} = (G, E, D)$ and a NIZK scheme $\Pi_{\text{NIZK}} = (G, P, V, S \coloneqq (S_G, S_P))$. In the following, vectors are assumed to have matching lengths; consider relation $R_{\text{PKEBC}^{\textsf{lin-ctxt}}}$ defined as

$$\begin{aligned} R_{\text{PKEBC}^{\textsf{lin-ctxt}}} \coloneqq \Big\{ &\big((1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}), (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}')\big) : \\ &\big(k_{\mathtt{sym}} = \Pi_{\text{SKE}}.G(1^k; r_{\mathtt{sym}})\big) \wedge \big(c_{\mathtt{sym}} = \Pi_{\text{SKE}}.E\big(k_{\mathtt{sym}}, (r_{\mathtt{pp}}, \vec{v}, m); r_{\mathtt{sym}}'\big)\big) \wedge \\ &\Bigg[\bigwedge_{j \in \{1, \dots, |\vec{c}|\}} \big(c_j = \Pi_{\text{PKE}}.E_{v_j}(k_{\mathtt{sym}}; r_j)\big) \Bigg] \wedge \big(c_{\mathtt{pp}} = \Pi_{\text{PKE}}.E_{\mathtt{pk}_{\mathtt{pp}}}((\vec{v}, m); r_{\mathtt{pp}})\big) \Big\}. \end{aligned} \tag{5.1}$$

In Algorithm 4, we consider the language $L_{\text{PKEBC}^{\textsf{lin-ctxt}}}$ that is induced by relation $R_{\text{PKEBC}^{\textsf{lin-ctxt}}}$: $L_{\text{PKEBC}^{\textsf{lin-ctxt}}} \coloneqq \{(1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \mid \exists (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}') : \big((1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}), (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}')\big) \in R_{\text{PKEBC}^{\textsf{lin-ctxt}}}\}$.

5.3 Achieving Linear Time Decryption

As discussed in Section 2.2, while the scheme given in Section 5.2 already achieves linear size ciphertexts and linear time encryption, it does not achieve linear time decryption. We now show how to modify $\Pi_{\text{PKEBC}}^{\textsf{lin-ctxt}}$ to achieve linear time decryption. The new scheme, denoted $\Pi_{\text{PKEBC}}^{\textsf{lin-dec}}$, is defined in Algorithm 5, and uses the same building blocks as $\Pi_{\text{PKEBC}}^{\textsf{lin-ctxt}}$. In the following, vectors are assumed to have matching lengths; furthermore, to simplify the definition of the relations below, we introduce the following predicate:

$$\begin{aligned} \textsf{CtxtMatch}(\mathtt{pk}, \mathtt{pk}', r_0, r_1, r_2, \alpha, k, c_0, c_1, c_2) \coloneqq \big((c_0, c_1, c_2) = \\ \big(\Pi_{\text{PKE}}.E_{\mathtt{pk}}(\alpha; r_0), \Pi_{\text{PKE}}.E_{\mathtt{pk}'}(r_0; r_1), \Pi_{\text{PKE}}.E_{\mathtt{pk}'}(k; r_2)\big)\big). \end{aligned} \tag{5.2}$$

[9] At a more technical level, replacing the binding commitment scheme of Maurer et al.'s PKEBC construction by a PKE scheme also serves the purpose of allowing the $(\textsf{IND} + \textsf{IK})$-CCA-2 security reductions to handle decryption queries.

Algorithm 4 Construction of PKEBC scheme $\Pi_{\text{PKEBC}}^{\text{lin-ctxt}} = (S, G, E, D)$.

$S(1^k)$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\text{PKE}}.G(1^k)$
  return $\mathtt{pp} := (1^k, \mathtt{crs} \leftarrow \Pi_{\text{NIZK}}.G(1^k), \mathtt{pk})$

$G(\mathtt{pp})$
  $(\mathtt{pk}', \mathtt{sk}') \leftarrow \Pi_{\text{PKE}}.G(1^k)$
  return $\big(\mathtt{pk} := \mathtt{pk}', \mathtt{sk} := (\mathtt{pk}, \mathtt{sk}')\big)$

$E_{\mathtt{pp}}(\vec{v} := \big(\mathtt{pk}_1, \ldots, \mathtt{pk}_{|\vec{v}|}\big), m)$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}\big((\vec{v}, m); r_{\mathtt{pp}}\big)$
  $k_{\mathtt{sym}} \leftarrow \Pi_{\text{SKE}}.G(1^k; r_{\mathtt{sym}})$
  $c_{\mathtt{sym}} \leftarrow \Pi_{\text{SKE}}.E_{k_{\mathtt{sym}}}\big((r_{\mathtt{pp}}, \vec{v}, m); r_{\mathtt{sym}}{}'\big)$
  for each $j \in \{1, \ldots, |\vec{v}|\}$ **do**
    $c_j \leftarrow \Pi_{\text{PKE}}.E_{v_j}(k_{\mathtt{sym}}; r_j)$
  $(\vec{r}, \vec{c}) := \big((r_1, \ldots, r_{|\vec{v}|}), (c_1, \ldots, c_{|\vec{v}|})\big)$
  $p \leftarrow \Pi_{\text{NIZK}}.P_{\mathtt{crs}}\big((1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{lin-ctxt}}}, (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}{}')\big)$
  return $(p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$

$D_{\mathtt{pp}}(\mathtt{sk}, c := (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}))$
  if $\Pi_{\text{NIZK}}.V_{\mathtt{crs}}\big((1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{lin-ctxt}}}, p\big) = \mathtt{valid}$ **then**
    for $j = 1, \ldots, |\vec{c}|$ **do**
      $k_{\mathtt{sym}} \leftarrow \Pi_{\text{PKE}}.D_{\mathtt{sk.sk}'}(c_j)$
      $\big(r_{\mathtt{pp}}, \vec{v}, m\big) \leftarrow \Pi_{\text{SKE}}.D_{k_{\mathtt{sym}}}(c_{\mathtt{sym}})$
      if $(r_{\mathtt{pp}}, \vec{v}, m) \neq \bot \wedge \mathtt{sk.pk} = v_j$ **then**
        if $c_{\mathtt{pp}} = \Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}\big((\vec{v}, m); r_{\mathtt{pp}}\big)$ **then**
          return $(\vec{v}, m)$
  return $\bot$

Consider relation $R_{\text{PKEBC}^{\text{lin-dec}}}$ defined as

$$R_{\text{PKEBC}^{\text{lin-dec}}} := \Bigg\{\big((1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}), (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}{}')\big) : \qquad (5.3)$$
$$\big(k_{\mathtt{sym}} = \Pi_{\text{SKE}}.G(1^k; r_{\mathtt{sym}})\big) \wedge \big(c_{\mathtt{sym}} = \Pi_{\text{SKE}}.E(k_{\mathtt{sym}}, (\vec{v}, m); r_{\mathtt{sym}}{}')\big) \wedge$$
$$\Bigg[\bigwedge_{j \in \{1,\ldots,|\vec{c}|\}} \mathsf{CtxtMatch}(\mathtt{pk}_{\mathtt{pp}}, v_j, r_{j,0}, r_{j,1}, r_{j,2}, v_j, k_{\mathtt{sym}}, c_{j,0}, c_{j,1}, c_{j,2})\Bigg] \wedge$$
$$\big(c_{\mathtt{pp}} = \Pi_{\text{PKE}}.E_{\mathtt{pk}_{\mathtt{pp}}}(m; r_{\mathtt{pp}})\big)\Bigg\}.$$

In Algorithm 5, we consider the language $L_{\text{PKEBC}^{\text{lin-dec}}}$ that is induced by relation $R_{\text{PKEBC}^{\text{lin-dec}}}$: $L_{\text{PKEBC}^{\text{lin-dec}}} := \{(1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \mid \exists(\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}{}') : \big((1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}), (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}{}')\big) \in R_{\text{PKEBC}^{\text{lin-dec}}}\}$.

5.4 Achieving Tight Security under Adaptive Corruptions

Finally, we modify $\Pi_{\text{PKEBC}}^{\text{lin-dec}}$ to get a PKEBC scheme that is tightly security under adaptive corruptions. Informally, we use the same two-key technique that we used for our MDVS scheme construction [1, 22]. In other words, in our scheme each party generates two key-pairs, $(\mathtt{pk}_0, \mathtt{sk}_0)$ and $(\mathtt{pk}_1, \mathtt{sk}_1)$, and then discards one

Algorithm 5 Construction of PKEBC scheme $\Pi^{\mathsf{lin\text{-}dec}}_{\text{PKEBC}}$.

$S(1^k)$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\text{PKE}}.G(1^k)$
  return $\mathtt{pp} := (1^k, \mathtt{crs} \leftarrow \Pi_{\text{NIZK}}.G(1^k), \mathtt{pk})$

$G(\mathtt{pp})$
  $(\mathtt{pk}', \mathtt{sk}') \leftarrow \Pi_{\text{PKE}}.G(1^k)$
  return $\big(\mathtt{pk} := \mathtt{pk}', \mathtt{sk} := (\mathtt{pk}, \mathtt{sk}')\big)$

$E_{\mathtt{pp}}(\vec{v} := \big(\mathtt{pk}_1, \ldots, \mathtt{pk}_{|\vec{v}|}\big), m)$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}(m; r_{\mathtt{pp}})$
  $k_{\mathtt{sym}} \leftarrow \Pi_{\text{SKE}}.G(1^k; r_{\mathtt{sym}})$
  $c_{\mathtt{sym}} \leftarrow \Pi_{\text{SKE}}.E_{k_{\mathtt{sym}}}((\vec{v}, m); r_{\mathtt{sym}}')$
  for each $j \in \{1, \ldots, |\vec{v}|\}$ **do**
    $(c_{j,0}, c_{j,1}, c_{j,2}) \leftarrow (\Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}(v_j; r_{j,0}), \Pi_{\text{PKE}}.E_{v_j}(r_{j,0}; r_{j,1}), \Pi_{\text{PKE}}.E_{v_j}(k_{\mathtt{sym}}; r_{j,2}))$
  $\vec{r} := \big((r_{1,0}, r_{1,1}, r_{1,2}), \ldots, (r_{|\vec{v}|,0}, r_{|\vec{v}|,1}, r_{|\vec{v}|,2})\big)$
  $\vec{c} := \big((c_{1,0}, c_{1,1}, c_{1,2}), \ldots, (c_{|\vec{v}|,0}, c_{|\vec{v}|,1}, c_{|\vec{v}|,2})\big)$
  $p \leftarrow \Pi_{\text{NIZK}}.P_{\mathtt{crs}}\big((1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\mathsf{lin\text{-}dec}}}, (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}')\big)$
  return $(p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$

$D_{\mathtt{pp}}(\mathtt{sk}, c := (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}))$
  if $\Pi_{\text{NIZK}}.V_{\mathtt{crs}}\big((1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\mathsf{lin\text{-}dec}}}, p\big) = \mathtt{valid}$ **then**
    for $j = 1, \ldots, |\vec{c}|$ **do**
      $r \leftarrow \Pi_{\text{PKE}}.D_{\mathtt{sk.sk}'}(c_{j,1})$
      if $r \neq \bot \land \Pi_{\text{PKE}}.E_{\mathtt{pp.pk}}(\mathtt{sk.pk}; r) = c_{j,0}$ **then**
        $k_{\mathtt{sym}} \leftarrow \Pi_{\text{PKE}}.D_{\mathtt{sk.sk}'}(c_{j,2})$
        return $\Pi_{\text{SKE}}.D_{k_{\mathtt{sym}}}(c_{\mathtt{sym}})$
  return $\bot$

of the secret keys $\mathtt{sk}_b$ picked uniformly at random. The new scheme is denoted $\Pi^{\mathsf{adap}}_{\text{PKEBC}}$ and is defined in Algorithm 6. Similarly to $\Pi^{\mathsf{lin\text{-}dec}}_{\text{PKEBC}}$, $\Pi^{\mathsf{adap}}_{\text{PKEBC}}$ uses the same building blocks as $\Pi^{\mathsf{lin\text{-}ctxt}}_{\text{PKEBC}}$. Consider relation $R_{\text{PKEBC}^{\mathsf{adap}}}$ defined as

$$R_{\text{PKEBC}^{\mathsf{adap}}} := \Big\{ \big((1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}), (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}')\big) \mid \big(k_{\mathtt{sym}} = \Pi_{\text{SKE}}.G(1^k; r_{\mathtt{sym}})\big) \land \big(c_{\mathtt{sym}} = \Pi_{\text{SKE}}.E\big(k_{\mathtt{sym}}, (\vec{v}, m); r_{\mathtt{sym}}'\big)\big) \land \big(c_{\mathtt{pp}} = \Pi_{\text{PKE}}.E_{\mathtt{pk}_{\mathtt{pp}}}(m; r_{\mathtt{pp}})\big) \land \Big[\bigwedge_{j \in \{1,\ldots,|\vec{c}|\},\, b \in \{0,1\}} \mathsf{CtxtMatch}(\mathtt{pk}_{\mathtt{pp}}, v_j.\mathtt{pk}_b, r_{j,0}, r_{j,b,1}, r_{j,b,2}, v_j, k_{\mathtt{sym}}, c_{j,0}, c_{j,b,1}, c_{j,b,2}) \Big] \Big\}, \tag{5.4}$$

where $\mathsf{CtxtMatch}$ is as in Equation 5.2. In Algorithm 6, we consider the following language: $L_{\text{PKEBC}^{\mathsf{adap}}} := \{(1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \mid \exists (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}') : \big((1^k, \mathtt{pk}_{\mathtt{pp}}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}), (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}')\big) \in R_{\text{PKEBC}^{\mathsf{adap}}}\}$.

5.4.1 Security Analysis of $\Pi^{\mathsf{adap}}_{\text{PKEBC}}$ The following theorem gives an informal overview of the security properties of our PKEBC scheme construction. The formal theorems and corresponding full proofs are in the appendix (see Section B.3).

Algorithm 6 Construction $\Pi_{\mathrm{PKEBC}}^{\mathsf{adap}}$.

$S(1^k)$
  $(\mathtt{pk}, \mathtt{sk}) \leftarrow \Pi_{\mathrm{PKE}}.G(1^k)$
  return $\mathtt{pp} := (1^k, \mathtt{crs} \leftarrow \Pi_{\mathrm{NIZK}}.G(1^k), \mathtt{pk})$

$G(\mathtt{pp})$
  $(\mathtt{pk}_0, \mathtt{sk}_0) \leftarrow \Pi_{\mathrm{PKE}}.G(1^k)$
  $(\mathtt{pk}_1, \mathtt{sk}_1) \leftarrow \Pi_{\mathrm{PKE}}.G(1^k)$
  $b \leftarrow RandomCoin$
  return $\big(\mathtt{pk} := (\mathtt{pk}_0, \mathtt{pk}_1), \mathtt{sk} := (\mathtt{pk}, b, \mathtt{sk}_b)\big)$

$E_{\mathtt{pp}}(\vec{v} := \big(\mathtt{pk}_1, \ldots, \mathtt{pk}_{|\vec{v}|}\big), m)$
  $c_{\mathtt{pp}} \leftarrow \Pi_{\mathrm{PKE}}.E_{\mathtt{pp}.\mathtt{pk}}\big(m; r_{\mathtt{pp}}\big)$
  $k_{\mathtt{sym}} \leftarrow \Pi_{\mathrm{SKE}}.G(1^k; r_{\mathtt{sym}})$
  $c_{\mathtt{sym}} \leftarrow \Pi_{\mathrm{SKE}}.E_{k_{\mathtt{sym}}}\big((\vec{v}, m); r_{\mathtt{sym}}'\big)$
  for each $j \in \{1, \ldots, |\vec{v}|\}$ **do**
    $c_{j,0} \leftarrow \Pi_{\mathrm{PKE}}.E_{\mathtt{pp}.\mathtt{pk}}(v_j; r_{j,0})$
    for each $b \in \{0,1\}$ **do**
      $(c_{j,b,1}, c_{j,b,2}) \leftarrow (\Pi_{\mathrm{PKE}}.E_{v_j.\mathtt{pk}_b}(r_{j,0}; r_{j,b,1}), \Pi_{\mathrm{PKE}}.E_{v_j.\mathtt{pk}_b}(k_{\mathtt{sym}}; r_{j,b,2}))$
    $(r_j, c_j) \leftarrow ((r_{j,0}, r_{j,0,1}, r_{j,0,2}, r_{j,1,1}, r_{j,1,2}), (c_{j,0}, c_{j,0,1}, c_{j,0,2}, c_{j,1,1}, c_{j,1,2}))$
  $(\vec{r}, \vec{c}) := \big((r_1, \ldots, r_{|\vec{v}|}), (c_1, \ldots, c_{|\vec{v}|})\big)$
  $p \leftarrow \Pi_{\mathrm{NIZK}}.P_{\mathtt{crs}}\big((1^k, \mathtt{pp}.\mathtt{pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\mathrm{PKEBC}^{\mathsf{adap}}}, (\vec{v}, m, r_{\mathtt{pp}}, \vec{r}, r_{\mathtt{sym}}, r_{\mathtt{sym}}')\big)$
  return $(p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$

$D_{\mathtt{pp}}(\mathtt{sk}, c := (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}))$
  if $\Pi_{\mathrm{NIZK}}.V_{\mathtt{crs}}\big((1^k, \mathtt{pp}.\mathtt{pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\mathrm{PKEBC}^{\mathsf{adap}}}, p\big) = \mathtt{valid}$ **then**
    for $j = 1, \ldots, |\vec{c}|$ **do**
      $r \leftarrow \Pi_{\mathrm{PKE}}.D_{\mathtt{sk}.\mathtt{sk}}(c_{j,\mathtt{sk}.b,1})$
      if $r \neq \bot \wedge \ \Pi_{\mathrm{PKE}}.E_{\mathtt{pp}.\mathtt{pk}}(\mathtt{sk}.\mathtt{pk}; r) = c_{j,0}$ **then**
        $k_{\mathtt{sym}} \leftarrow \Pi_{\mathrm{PKE}}.D_{\mathtt{sk}.\mathtt{sk}}(c_{j,\mathtt{sk}.b,2})$
        return $\Pi_{\mathrm{SKE}}.D_{k_{\mathtt{sym}}}(c_{\mathtt{sym}})$
  return $\bot$

Theorem 2 (Informal). *If Π_{PKE} is correct and tightly multi-user and multi-challenge* IND-CPA *and* IK-CPA *secure under* non-adaptive *corruptions, Π_{NIZK} is complete, sound, tightly multi-statement adaptive zero-knowledge and tightly multi-statement simulation sound, and Π_{SKE} is correct and tightly multi-instance* IND-CPA *secure, then $\Pi_{\mathrm{PKEBC}}^{\mathsf{adap}}$ is:*

1. *tightly correct (Theorem 9);*
2. *tightly robust (Theorem 10);*
3. *tightly consistent (Theorem 11); and*
4. *tightly* $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ *secure under adaptive corruptions (Theorem 12).*

6 Multi-Designated Receiver Signed Public Key Encryption Schemes

An MDRS-PKE scheme is a 6-tuple of PPTs $\Pi = (S, G_S, G_R, E, D, \mathit{Forge})$, where:

- S: on input 1^k, generates public parameters $\mathtt{pp}$;
- G_S: on input $\mathtt{pp}$, generates a sender key-pair $(\mathtt{spk}, \mathtt{ssk})$;
- G_R: on input $\mathtt{pp}$, generates a receiver key-pair $(\mathtt{rpk}, \mathtt{rsk})$;

– E: on input $(\mathtt{pp}, \mathtt{ssk}, \vec{v}, m)$, where $\mathtt{ssk}$ is the secret sending key, $\vec{v}$ is a vector of public keys of the intended receivers, and m is the message, generates a ciphertext c;
– D: on input $(\mathtt{pp}, \mathtt{rsk}, c)$, where $\mathtt{rsk}$ is the receiver's secret key, D decrypts c using $\mathtt{rsk}$, obtaining a triple sender/receiver-vector/message $(\mathtt{spk}, \vec{v}, m)$ (or $\perp$ if decryption fails) which it then outputs;
– *Forge*: on input $(\mathtt{pp}, \mathtt{spk}, \vec{v}, m, \vec{s})$, where $\mathtt{spk}$ is the sender's public key, $\vec{v}$ is a vector of public keys of the intended receivers, m is the message and $\vec{s}$ is a vector of designated receivers' secret keys—with $|\vec{s}| = |\vec{v}|$ and where for $i \in \{1, \ldots, |\vec{v}|\}$, either $s_i = \perp$ or s_i is the secret key corresponding to the i-th public key of $\vec{v}$, i.e. v_i—generates a ciphertext c.

Analogously to Section 4, in this section we introduce new (stronger) security notions for MDRS-PKE schemes (see Definitions 13 and 14). Then, we briefly describe how one can use the MDVS and PKEBC constructions from before, together with a strongly unforgeable one-time Digital Signature Scheme to obtain an MDRS-PKE scheme with the desired properties (by following the construction given by Maurer et al. in [20,21]), and argue why the scheme is secure with respect to our new stronger MDRS-PKE security notions. In particular, we will formally prove that the construction given by Maurer et al. in [20] is $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ secure.

6.1 Security Notions

Below we state the notions of Correctness, Consistency, Unforgeability, $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ and Off-The-Record for MDRS-PKE schemes. Analogously to the new MDVS Off-The-Record security notion we introduced in Section 4.1 (Definition 4), the $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ and Off-The-Record security notions we now present (Definitions 13 and 14, respectively), allow the adversary to obtain the sender's secret key; and analogously to the new PKEBC security notions we introduced in Section 5.1 (in particular Definition 9), our new MDRS-PKE security notions capture the setting where the adversary can adaptively corrupt parties (see Definition 13). The security notions we now present are thus an enhancement over the original ones given in [21].

Let $\Pi = (S, G_S, G_V, E, D, Forge)$ be an MDRS-PKE scheme with message space $\mathcal{M}$. The oracles below are defined for a game-system with (an implicitly defined) security parameter k:

Public Parameter Generation Oracle: $\mathcal{O}_{PP}$

1. On the first call, compute $\mathtt{pp} \leftarrow S(1^k)$; output $\mathtt{pp}$;
2. On subsequent calls, simply output $\mathtt{pp}$.

Sender Key-Pair Oracle: $\mathcal{O}_{SK}(A_i)$

1. On the first call on input A_i, compute and store $(\mathtt{spk}_i, \mathtt{ssk}_i) \leftarrow G_S(\mathtt{pp})$; output $(\mathtt{spk}_i, \mathtt{ssk}_i)$;
2. On subsequent calls, simply output $(\mathtt{spk}_i, \mathtt{ssk}_i)$.

Receiver Key-Pair Oracle: $\mathcal{O}_{RK}(B_j)$

1. Analogous to the Sender Key-Pair Oracle.

Sender Public-Key Oracle: $\mathcal{O}_{SPK}(A_i)$

1. $(\texttt{spk}_i, \texttt{ssk}_i) \leftarrow \mathcal{O}_{SK}(A_i)$; output $\texttt{spk}_i$.

Receiver Public-Key Oracle: $\mathcal{O}_{RPK}(B_j)$

1. Analogous to the Sender Public-Key Oracle.

Encryption Oracle: $\mathcal{O}_E(A_i, \vec{V}, m)$

1. $(\texttt{spk}_i, \texttt{ssk}_i) \leftarrow \mathcal{O}_{SK}(A_i)$;
2. $\vec{v} \leftarrow (\mathcal{O}_{RPK}(V_1), \ldots, \mathcal{O}_{RPK}(V_{|\vec{V}|}))$;
3. Output $c \leftarrow E_{\mathsf{pp}}(\texttt{ssk}_i, \vec{v}, m)$.

Decryption Oracle: $\mathcal{O}_D(B_j, c)$

1. $(\texttt{rpk}_j, \texttt{rsk}_j) \leftarrow \mathcal{O}_{RK}(B_j)$;
2. $(\texttt{spk}_i, \vec{v} := (\texttt{rpk}_1, \ldots, \texttt{rpk}_{|\vec{v}|}), m) \leftarrow D_{\mathsf{pp}}(\texttt{rsk}_j, c)$;
3. If, for each party A_i previously input to either $\mathcal{O}_{SK}$, $\mathcal{O}_{SPK}$ or $\mathcal{O}_E$, $\texttt{spk}_i \neq \mathcal{O}_{SPK}(A_i)$, then output $\perp$;
4. If, for some $l \in \{1, \ldots, |\vec{V}|\}$, there is no party B_j that was previously input to either $\mathcal{O}_{RK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ or $\mathcal{O}_D$ such that $v_l = \mathcal{O}_{RPK}(V_l)$, then output $\perp$;
5. Output $(\texttt{spk}, \vec{v}, m)$.

Definition 10 (Correctness). *Game system $\mathbf{G}^{\mathsf{Corr}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$. $\mathbf{A}$ wins the game if there are two queries q_E and q_D to $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, where q_E has input $(A_i, \vec{V}, m)$ and q_D has input (B_j, c), satisfying $B_j \in \vec{V}$, the input c in q_D is the output of q_E, the output of q_D is $(\texttt{spk}_i{}', \vec{v}', m')$ with $(\texttt{spk}_i{}', \vec{v}', m') = \perp$ or $(\texttt{spk}_i{}', \vec{v}', m') \neq (\texttt{spk}_i, \vec{v}, m)$—where $\texttt{spk}_i$ is A_i's public key and $\vec{v}$ is the corresponding vector of public keys of the parties of $\vec{V}$. The advantage of $\mathbf{A}$ in winning the Correctness game, denoted $Adv^{\mathsf{Corr}}(\mathbf{A})$, is the probability that $\mathbf{A}$ wins game $\mathbf{G}^{\mathsf{Corr}}$ as described above.*

Definition 11 (Consistency). *Game system $\mathbf{G}^{\mathsf{Cons}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$. $\mathbf{A}$ wins the game if there is a ciphertext c such that $\mathcal{O}_D$ is queried on inputs (B_i, c) and (B_j, c) for some B_i and B_j (possibly with $B_i = B_j$), there is no prior query on either B_i or B_j to $\mathcal{O}_{RK}$, query $\mathcal{O}_D(B_i, c)$ outputs some $(\texttt{spk}_l, \vec{v}, m)$ satisfying $(\texttt{spk}_l, \vec{v}, m) \neq \perp$, $\texttt{spk}_l$ is some party A_l's public sender key (i.e. $\mathcal{O}_{SPK}(A_l) = \texttt{spk}_l$) and $\texttt{rpk}_j \in \vec{v}$ (where $\texttt{rpk}_j$ is B_j's public key), and query $\mathcal{O}_D(B_j, c)$ does not output the same triple $(\texttt{spk}_l, \vec{v}, m)$. The advantage of $\mathbf{A}$ in winning the Consistency game is denoted $Adv^{\mathsf{Cons}}(\mathbf{A})$ and corresponds to the probability that $\mathbf{A}$ wins game $\mathbf{G}^{\mathsf{Cons}}$ as described above.*

Definition 12 (Unforgeability). *Game system $\mathbf{G}^{\mathsf{Unforg}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$. $\mathbf{A}$ wins if it makes a query $\mathcal{O}_D(B_j, c)$ that outputs $(\texttt{spk}_i, \vec{v}, m) \neq \perp$, there is a sender A_i and a vector of receivers $\vec{V}$ such that $\texttt{spk}_i$ is A_i's sender public key (i.e. $\mathcal{O}_{SPK}(A_i) = \texttt{spk}_i$) and $\vec{v}$ is the vector of receiver public keys corresponding to $\vec{V}$ (i.e. $|\vec{V}| = |\vec{v}|$ and for each $l \in \{1, \ldots, |\vec{v}|\}$, $\mathcal{O}_{RPK}(V_l) = v_l$), there was no query*

$\mathcal{O}_E(A_i{}', \vec{V}', m')$ *with* $(A_i, \vec{V}, m) = (A_i{}', \vec{V}', m')$*, and neither* $\mathcal{O}_{SK}$ *was queried on input* A_i *nor* $\mathcal{O}_{RK}$ *was queried on input* B_j*. The advantage of* $\mathbf{A}$ *in winning the Unforgeability game is the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{Unforg}}$ *as described above, and is denoted* $Adv^{\mathsf{Unforg}}(\mathbf{A})$*.*

We say that an adversary $\mathbf{A}$ (ε, t)-breaks the $(n_S, n_R, d_E, q_E, q_D)$-Correctness, Consistency, or Unforgeability of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{SK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$ on at most n_S different senders, queries $\mathcal{O}_{RK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$ on at most n_R different receivers, makes at most q_E and q_D queries to $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, with the sum of lengths of the party vectors input to $\mathcal{O}_E$ being at most d_E, and $\mathbf{A}$'s advantage in winning the (corresponding) security game is at least ε.

6.1.1 New $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ and Off-The-Record Notions.

Analogously to Section 4.1.1, in this section we present the new enhanced OTR and $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ security notions for MDRS-PKE schemes. As already mentioned, the main difference between our new notions and existing ones (see [21]) is that in our new notions the adversary can query for the secret key of any sender (see Definitions 13 and 14) and can corrupt parties adaptively.

The games defined by these notions provide adversaries with access to the oracles from before as well as to the oracles $\mathcal{O}_E$ and $\mathcal{O}_D$ defined below:

Encryption Oracle: $\mathcal{O}_E\big((A_{i,0}, \vec{V_0}, m_0), (A_{i,1}, \vec{V_1}, m_1)\big)$

1. For game system $\mathbf{G}_{\mathbf{b}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}$, encrypt $m_{\mathbf{b}}$ under $\mathtt{ssk}_{i,\mathbf{b}}$ ($A_{i,\mathbf{b}}$'s sender secret key) and $\vec{v_{\mathbf{b}}}$ ($\vec{V_{\mathbf{b}}}$'s corresponding vector of receiver public keys); output c.

Decryption Oracle: $\mathcal{O}_D(B_j, c)$

1. If c was the output of some query to $\mathcal{O}_E$, output $\mathtt{test}$;
2. Otherwise, compute $(\mathtt{spk}_i, \vec{v}, m) \leftarrow D_{\mathtt{pp},\mathtt{sk}_j}(c)$, where $\mathtt{sk}_j$ is B_j's secret key; output $(\mathtt{spk}_i, \vec{v}, m)$.

Definition 13 ($(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ **Security).** *For* $\mathbf{b} \in \{0, 1\}$*, game system* $\mathbf{G}_{\mathbf{b}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}$ *provides an adversary* $\mathbf{A}$ *with access to oracles* $\mathcal{O}_{PP}$*,* $\mathcal{O}_{SK}$*,* $\mathcal{O}_{RK}$*,* $\mathcal{O}_{SPK}$*,* $\mathcal{O}_{RPK}$*,* $\mathcal{O}_E$ *and* $\mathcal{O}_D$*.* $\mathbf{A}$ *wins the game if it outputs a guess bit* b' *with* $b' = \mathbf{b}$ *and for every query* $\mathcal{O}_E\big((A_{i,0}, \vec{V_0}, m_0), (A_{i,1}, \vec{V_1}, m_1)\big)$*: 1.* $|m_0| = |m_1|$*; 2.* $|\vec{V_0}| = |\vec{V_1}|$*; and 3. there is no query to* $\mathcal{O}_{RK}$ *on any* $B_j \in Set(\vec{V_0}) \cup Set(\vec{V_1})$ *at any point during the game. We define the advantage of* $\mathbf{A}$ *in winning the* $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ *game as*

$$Adv^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}(\mathbf{A}) := \left|\Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}} = \mathtt{win}] - 1\right|.$$

An adversary $\mathbf{A}$ (ε, t)-breaks the (n_R, d_E, q_E, q_D)-$(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ security of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{RK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$ on at most n_R different receivers, makes at most q_E and q_D queries to $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, with the sum of lengths of the party vectors input to $\mathcal{O}_E$ being at most d_E, and satisfies $Adv^{(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}(\mathbf{A}) \geq \varepsilon$.

The following notion defines two game systems, $\mathbf{G}_{\mathbf{0}}^{\mathsf{OTR}}$ and $\mathbf{G}_{\mathbf{1}}^{\mathsf{OTR}}$, which provide adversaries with access to an oracle $\mathcal{O}_E$, whose behavior varies depending on the underlying game system. For $\mathbf{b} \in \{0,1\}$, $\mathcal{O}_E$ behaves as follows:

Encryption Oracle: $\mathcal{O}_E(\mathtt{type} \in \{\mathtt{sig}, \mathtt{sim}\}, A_i, \vec{V}, m, \mathcal{C})$

For game system $\mathbf{G}_{\mathbf{b}}^{\mathsf{OTR}}$, the oracle behaves as follows:

1. Let $\vec{v} = (v_1, \ldots, v_{|\vec{V}|})$ and $\vec{s} = (s_1, \ldots, s_{|\vec{V}|})$, where, for $i \in \{1, \ldots, |\vec{V}|\}$:
   - $(v_i, s_i) = \begin{cases} \mathcal{O}_{RK}(V_i) & \text{if } V_i \in \mathcal{C} \\ (\mathcal{O}_{RPK}(V_i), \bot) & \text{otherwise;} \end{cases}$
2. $(c_{\mathbf{0}}, c_{\mathbf{1}}) \leftarrow (\Pi.E_{\mathsf{pp}}(\mathtt{ssk}_i, \vec{v}, m), \Pi.Forge_{\mathsf{pp}}(\mathtt{spk}_i, \vec{v}, m, \vec{s}))$;
3. If $\mathbf{b} = 0$, output $c_{\mathbf{0}}$ if $\mathtt{type} = \mathtt{sig}$ and $c_{\mathbf{1}}$ if $\mathtt{type} = \mathtt{sim}$; otherwise, if $\mathbf{b} = 1$, output $c_{\mathbf{1}}$.

Definition 14 (Off-The-Record). *For $\mathbf{b} \in \{0,1\}$, game system $\mathbf{G}_{\mathbf{b}}^{\mathsf{OTR}}$ provides an adversary $\mathbf{A}$ with access to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{RK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$. $\mathbf{A}$ wins the game if it outputs a guess bit b' with $b' = \mathbf{b}$ and for every query $(\mathtt{type}, A_i, \vec{V}, m, \mathcal{C})$ to $\mathcal{O}_E$, and letting c be the output of $\mathcal{O}_E$, all of the following hold: 1. $\mathcal{C} \subseteq Set(\vec{V})$; 2. for every query B_j to $\mathcal{O}_{VK}$, $B_j \notin Set(\vec{V}) \setminus \mathcal{C}$; 3. for all queries $\mathcal{O}_D(B_j, c')$, $c' \neq c$. $\mathbf{A}$'s advantage in winning the Off-The-Record security game is*

$$Adv^{\mathsf{OTR}}(\mathbf{A}) := \left| \Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{\mathsf{OTR}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{\mathsf{OTR}} = \mathtt{win}] - 1 \right|.$$

We say that an adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{OTR}}, t)$-breaks the $(n_S, n_R, d_E, q_E, q_D)$-Off-The-Record security of Π if $\mathbf{A}$ runs in time at most t, queries $\mathcal{O}_{SK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$ on at most n_S different senders, queries $\mathcal{O}_{RK}$, $\mathcal{O}_{RPK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$ on at most n_R different receivers, makes at most q_E and q_D queries to $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, with the sum of lengths of the party vectors input to $\mathcal{O}_E$ being at most d_E, and satisfies $Adv^{\mathsf{OTR}}(\mathbf{A}) \geq \varepsilon_{\mathsf{OTR}}$. Finally, we say that Π is

$$(\varepsilon_{\mathsf{Corr}}, \varepsilon_{\mathsf{Cons}}, \varepsilon_{\mathsf{Unforg}}, \varepsilon_{(\mathsf{IND+IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}, \varepsilon_{\mathsf{OTR}}, \\ t, n_S, n_R, d_E, q_E, q_D)\text{-secure},$$

if no adversary $\mathbf{A}$: 1. $(\varepsilon_{\mathsf{Corr}}, t)$-breaks the $(n_S, n_R, d_E, q_E, q_D)$-Correctness of Π; 2. $(\varepsilon_{\mathsf{Cons}}, t)$-breaks the $(n_S, n_R, d_E, q_E, q_D)$-Consistency of Π; 3. $(\varepsilon_{\mathsf{Unforg}}, t)$-breaks the $(n_S, n_R, d_E, q_E, q_D)$-Unforgeability of Π; 4. $(\varepsilon_{(\mathsf{IND+IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}}, t)$-breaks the (n_R, d_E, q_E, q_D)-$(\mathsf{IND}+\mathsf{IK})$-$\mathsf{CCA}$-$2^{\mathsf{adap}}$ security of Π; or 5. $(\varepsilon_{\mathsf{OTR}}, t)$-breaks the $(n_S, n_R, d_E, q_E, q_D)$-Off-The-Record security of Π.

6.2 Construction of MDRS-PKE with Short Ciphertexts

Maurer et al. give a black-box construction of an MDRS-PKE scheme from a PKEBC scheme, an MDVS scheme, and a strongly unforgeable one-time Digital

Signature Scheme [20, 21]. At a high level, the construction [21, Algorithm 2] consists of using the MDVS scheme to sign messages and the PKEBC scheme to encrypt messages and their MDVS signatures; the digital signature scheme is used to tie MDVS signatures and PKEBC ciphertexts together (in the sense that modifying either of them results in an invalid ciphertext). More concretely, in their construction a sender key-pair consists of an MDVS signer key-pair, whereas a receiver key-pair consists of an MDVS verifier key-pair and a PKEBC key-pair. To encrypt a message m, a sender first samples a fresh DSS key-pair $(\mathtt{vk}, \mathtt{sk})$ and then uses the MDVS scheme to sign the message, the vector of PKEBC public keys of the receivers, and the verification key $\mathtt{vk}$; next, it uses the PKEBC scheme to encrypt the message, the MDVS signature, its own MDVS public key and the vector of public MDVS verifier keys of the receivers; finally the signer signs the resulting (PKEBC) ciphertext using the initially sampled DSS secret key. The final ciphertext c is then a triple consisting of the DSS verification key $\mathtt{vk}$, the PKEBC ciphertext c' and the DSS signature σ' on c', so $c = (\mathtt{vk}, \sigma', c')$. Conversely, to decrypt an MDRS-PKE ciphertext c, a receiver first verifies the validity of the DSS signature σ' on the PKEBC ciphertext c' with respect to the DSS verification key $\mathtt{vk}$ included in c; if the verification is successful, the receiver then decrypts the PKEBC ciphertext, obtaining not only the vector of PKEBC public keys of the receivers, but also a signer's MDVS public key (of the sender), a vector of MDVS verifier public keys (of each of the receivers), a message m, and an MDVS signature σ; then, it uses its MDVS secret verification key to check if σ is a valid MDVS signature on the message m, vector of PKEBC public keys obtained from decryption and the DSS verification key $\mathtt{vk}$, with respect to all the MDVS public keys obtained from decrypting the PKEBC ciphertext. For completeness, we present Maurer et al.'s MDRS-PKE construction in Algorithm 7.

Security of the Resulting MDRS-PKE *Scheme.* The MDRS-PKE notions introduced in this paper capture the security of MDRS-PKE schemes in the setting where the adversary is allowed to corrupt parties (obtaining their secret keys) adaptively. While this is not the case for the analogous notions considered in [21], we note that, apart from the IND-CCA-2 and IK-CCA-2 security notions [21, Definitions 9 and 10] and corresponding security proofs, the remainder of the security notions they consider (or their security proofs) do allow the adversary to corrupt parties adaptively.

On the other hand, and in contrast to the $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ and OTR security notions for MDRS-PKE schemes considered in this paper, the original notions from [21] do not capture the setting where the adversary is given access to the secret keys of senders (see [21, Definitions 9, 10 and 11]). Regarding Off-The-Record, and as noted in [20, Remark 12], if one assumes the underlying MDVS scheme satisfies the stronger off-the-record notion we consider in this paper—wherein the adversary is given access to the sender's secret key, see Definition 4—then the resulting MDRS-PKE scheme also satisfies the corresponding stronger Off-The-Record notion (i.e. the one we consider in this paper, see Defi-

Algorithm 7 Construction of $\Pi_{\text{MDRS-PKE}}$.

$S(1^k)$
  $\mathtt{pp}_{\text{MDVS}} \leftarrow \Pi_{\text{MDVS}}.S(1^k)$
  $\mathtt{pp}_{\text{PKEBC}} \leftarrow \Pi_{\text{PKEBC}}.S(1^k)$
  return $\mathtt{pp} := (\mathtt{pp}_{\text{MDVS}}, \mathtt{pp}_{\text{PKEBC}}, 1^k)$

$G_S(\mathtt{pp})$
  $(\mathtt{spk}_{\text{MDVS}}, \mathtt{ssk}_{\text{MDVS}}) \leftarrow \Pi_{\text{MDVS}}.G_S(\mathtt{pp}_{\text{MDVS}})$
  return $(\mathtt{spk} := \mathtt{spk}_{\text{MDVS}}, \mathtt{ssk} := (\mathtt{spk}, \mathtt{ssk}_{\text{MDVS}}))$

$G_R(\mathtt{pp})$
  $(\mathtt{vpk}_{\text{MDVS}}, \mathtt{vsk}_{\text{MDVS}}) \leftarrow \Pi_{\text{MDVS}}.G_V(\mathtt{pp}_{\text{MDVS}})$
  $(\mathtt{pk}_{\text{PKEBC}}, \mathtt{sk}_{\text{PKEBC}}) \leftarrow \Pi_{\text{PKEBC}}.G(\mathtt{pp}_{\text{PKEBC}})$
  return $(\mathtt{rpk} := (\mathtt{vpk}_{\text{MDVS}}, \mathtt{pk}_{\text{PKEBC}}), \mathtt{rsk} := \big(\mathtt{rpk}, (\mathtt{vsk}_{\text{MDVS}}, \mathtt{sk}_{\text{PKEBC}})\big))$

$E_{\mathtt{pp}}(\mathtt{ssk}, \vec{v} := (\mathtt{rpk}_1, \dots, \mathtt{rpk}_{|\vec{v}|}), m)$
  $\vec{v}_{\text{PKEBC}} := (\mathtt{rpk}_1.\mathtt{pk}_{\text{PKEBC}}, \dots, \mathtt{rpk}_{|\vec{v}|}.\mathtt{pk}_{\text{PKEBC}})$
  $\vec{v}_{\text{MDVS}} := (\mathtt{rpk}_1.\mathtt{vpk}_{\text{MDVS}}, \dots, \mathtt{rpk}_{|\vec{v}|}.\mathtt{vpk}_{\text{MDVS}})$
  $(\mathtt{vk}, \mathtt{sk}) \leftarrow \Pi_{\text{DSS}}.G(\mathtt{pp}.1^k)$
  $\sigma \leftarrow \Pi_{\text{MDVS}}.Sig_{\mathtt{pp}_{\text{MDVS}}}(\mathtt{ssk}_{\text{MDVS}}, \vec{v}_{\text{MDVS}}, (\vec{v}_{\text{PKEBC}}, m, \mathtt{vk}))$
  $c \leftarrow \Pi_{\text{PKEBC}}.E_{\mathtt{pp}_{\text{PKEBC}}}\big(\vec{v}_{\text{PKEBC}}, (\mathtt{spk}_{\text{MDVS}}, \vec{v}_{\text{MDVS}}, m, \sigma)\big)$
  $\sigma' \leftarrow \Pi_{\text{DSS}}.Sig_{\mathtt{sk}}(c)$
  return $(\mathtt{vk}, \sigma', c)$

$D_{\mathtt{pp}}(\mathtt{rsk}, c := (\mathtt{vk}, \sigma', c'))$
  if $\Pi_{\text{DSS}}.Vfy_{\mathtt{vk}}(c', \sigma') = 0$ **then**
    return $\perp$
  $\big(\vec{v}_{\text{PKEBC}}, (\mathtt{spk} := \mathtt{spk}_{\text{MDVS}}, \vec{v}_{\text{MDVS}}, m, \sigma)\big) \leftarrow \Pi_{\text{PKEBC}}.D_{\mathtt{pp}_{\text{PKEBC}}}(\mathtt{rsk}.\mathtt{sk}_{\text{PKEBC}}, c')$
  if $\big(\vec{v}_{\text{PKEBC}}, (\mathtt{spk}, \vec{v}_{\text{MDVS}}, m, \sigma)\big) = \perp \quad \vee \quad |\vec{v}_{\text{PKEBC}}| \neq |\vec{v}_{\text{MDVS}}|$ **then**
    return $\perp$
  $\vec{v} := \big((v_{\text{MDVS}\,1}, v_{\text{PKEBC}\,1}), \dots, (v_{\text{MDVS}\,|\vec{v}_{\text{PKEBC}}|}, v_{\text{PKEBC}\,|\vec{v}_{\text{PKEBC}}|})\big)$
  if $\mathtt{rsk}.\mathtt{rpk} \notin \vec{v}$ **then**
    return $\perp$
  if $\Pi_{\text{MDVS}}.Vfy_{\mathtt{pp}_{\text{MDVS}}}(\mathtt{spk}, \mathtt{vsk}_{\text{MDVS}}, \vec{v}_{\text{MDVS}}, (\vec{v}_{\text{PKEBC}}, m, \mathtt{vk}), \sigma) \neq \mathtt{valid}$ **then**
    return $\perp$
  return $(\mathtt{spk}, \vec{v}, m)$

$Forge_{\mathtt{pp}}(\mathtt{spk}, \vec{v} := (\mathtt{rpk}_1, \dots, \mathtt{rpk}_{|\vec{v}|}), m, \vec{s} := (\mathtt{rsk}_1, \dots, \mathtt{rsk}_{|\vec{s}|}))$
  $\vec{v}_{\text{PKEBC}} := (\mathtt{rpk}_1.\mathtt{pk}_{\text{PKEBC}}, \dots, \mathtt{rpk}_{|\vec{v}|}.\mathtt{pk}_{\text{PKEBC}})$
  $\vec{v}_{\text{MDVS}} := (\mathtt{rpk}_1.\mathtt{vpk}_{\text{MDVS}}, \dots, \mathtt{rpk}_{|\vec{v}|}.\mathtt{vpk}_{\text{MDVS}})$
  $\vec{s}_{\text{MDVS}} := (\mathtt{rsk}_1.\mathtt{vsk}_{\text{MDVS}}, \dots, \mathtt{rsk}_{|\vec{s}|}.\mathtt{vsk}_{\text{MDVS}})$
  $(\mathtt{vk}, \mathtt{sk}) \leftarrow \Pi_{\text{DSS}}.G(\mathtt{pp}.1^k)$
  $\sigma \leftarrow \Pi_{\text{MDVS}}.Forge_{\mathtt{pp}_{\text{MDVS}}}(\mathtt{spk}_{\text{MDVS}}, \vec{v}_{\text{MDVS}}, (\vec{v}_{\text{PKEBC}}, m, \mathtt{vk}), \vec{s}_{\text{MDVS}})$
  $c \leftarrow \Pi_{\text{PKEBC}}.E_{\mathtt{pp}_{\text{PKEBC}}}\big(\vec{v}_{\text{PKEBC}}, (\mathtt{spk}_{\text{MDVS}}, \vec{v}_{\text{MDVS}}, m, \sigma)\big)$
  $\sigma' \leftarrow \Pi_{\text{DSS}}.Sig_{\mathtt{sk}}(c)$
  return $(\mathtt{vk}, \sigma', c)$

nition 14). In particular, the same argument used in the OTR security proof of the MDRS-PKE construction (see [20, Section I.6]) also applies.

Putting things together, this means that all we need to prove is that Maurer et al.'s MDRS-PKE construction [20] is (tightly) $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ secure. The following theorem gives an informal overview of the security properties of Maurer et al.'s MDRS-PKE construction [20] in the setting where the signer's secret key leaks.

Theorem 3 (Informal). *If*

- Π_{PKEBC} *is tightly correct, completely robust, consistent and is also* $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ *secure under adaptive corruptions;*
- Π_{MDVS} *is tightly correct, consistent, unforgeable, off-the-record, forgery invalidity secure and message-bound validity secure (all under adaptive corruptions) and has unique keys; and*
- Π_{DSS} *is tightly correct and* $1\text{-}\mathsf{sEUF}\text{-}\mathsf{CMA}$ *secure*

then $\Pi_{\mathrm{MDRS\text{-}PKE}}$ *is tightly:*

1. *correct (see [20, Theorem 6, Remark 10, Section I.1]);*
2. *consistent under adaptive corruptions (see [20, Theorem 7, Section I.2]);*
3. $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ *secure (Theorem 13);*
4. *unforgeable under adaptive corruptions (see [20, Theorem 8, Section I.3]); and*
5. *off-the-record under adaptive corruptions (see [20, Theorem 11, Remark 12, Section I.6]).*

We prove the $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security of the MDRS-PKE scheme in Appendix (Section B.4.1).

References

1. Bader, C., Hofheinz, D., Jager, T., Kiltz, E., Li, Y.: Tightly-secure authenticated key exchange. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part I. LNCS, vol. 9014, pp. 629–658. Springer, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-46494-6_26
2. Bader, C., Jager, T., Li, Y., Schäge, S.: On the impossibility of tight cryptographic reductions. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 273–304. Springer, Heidelberg (May 2016). https://doi.org/10.1007/978-3-662-49896-5_10
3. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-privacy in public-key encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (Dec 2001). https://doi.org/10.1007/3-540-45682-1_33
4. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (May / Jun 2006). https://doi.org/10.1007/11761679_25

5. Chakraborty, S., Hofheinz, D., Maurer, U., Rito, G.: Deniable authentication when signing keys leak. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part III. LNCS, vol. 14006, pp. 69–100. Springer, Heidelberg (Apr 2023). https://doi.org/10.1007/978-3-031-30620-4_3
6. Chakraborty, S., Hofheinz, D., Maurer, U., Rito, G.: Deniable authentication when signing keys leak. Cryptology ePrint Archive, Report 2023/213 (2023), https://eprint.iacr.org/2023/213
7. Chaum, D.: Designated confirmer signatures. In: Santis, A.D. (ed.) EUROCRYPT'94. LNCS, vol. 950, pp. 86–91. Springer, Heidelberg (May 1995). https://doi.org/10.1007/BFb0053427
8. Damgård, I., Haagh, H., Mercer, R., Nitulescu, A., Orlandi, C., Yakoubov, S.: Stronger security and constructions of multi-designated verifier signatures. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part II. LNCS, vol. 12551, pp. 229–260. Springer, Heidelberg (Nov 2020). https://doi.org/10.1007/978-3-030-64378-2_9
9. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO'84. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (Aug 1984)
10. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (Aug 1994). https://doi.org/10.1007/3-540-48329-2_40
11. Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Sciences **28**(2), 270–299 (1984)
12. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (Dec 2006). https://doi.org/10.1007/11935230_29
13. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (Apr 2008). https://doi.org/10.1007/978-3-540-78967-3_24
14. Hesse, J., Hofheinz, D., Kohl, L.: On tightly secure non-interactive key exchange. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 65–94. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0_3
15. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated verifier proofs and their applications. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (May 1996). https://doi.org/10.1007/3-540-68339-9_13
16. Lee, Y., Lee, D.H., Park, J.H.: Tightly cca-secure encryption scheme in a multi-user setting with corruptions. Des. Codes Cryptogr. **88**(11), 2433–2452 (2020). https://doi.org/10.1007/s10623-020-00794-z
17. Libert, B., Paterson, K.G., Quaglia, E.A.: Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 206–224. Springer, Heidelberg (May 2012). https://doi.org/10.1007/978-3-642-30057-8_13
18. Lombardi, A., Quach, W., Rothblum, R.D., Wichs, D., Wu, D.J.: New constructions of reusable designated-verifier NIZKs. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 670–700. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_22
19. Maurer, U., Portmann, C., Rito, G.: Giving an adversary guarantees (or: How to model designated verifier signatures in a composable framework). In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part III. LNCS, vol. 13092, pp. 189–219. Springer, Heidelberg (Dec 2021). https://doi.org/10.1007/978-3-030-92078-4_7

20. Maurer, U., Portmann, C., Rito, G.: Multi-designated receiver signed public key encryption. Cryptology ePrint Archive, Report 2022/256 (2022), https://eprint.iacr.org/2022/256
21. Maurer, U., Portmann, C., Rito, G.: Multi-designated receiver signed public key encryption. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 644–673. Springer, Heidelberg (May / Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_22
22. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC. pp. 427–437. ACM Press (May 1990). https://doi.org/10.1145/100216.100273
23. Quach, W., Rothblum, R.D., Wichs, D.: Reusable designated-verifier NIZKs for all NP from CDH. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part II. LNCS, vol. 11477, pp. 593–621. Springer, Heidelberg (May 2019). https://doi.org/10.1007/978-3-030-17656-3_21
24. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: 40th FOCS. pp. 543–553. IEEE Computer Society Press (Oct 1999). https://doi.org/10.1109/SFFCS.1999.814628
25. Shoup, V.: Using hash functions as a hedge against chosen ciphertext attack. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 275–288. Springer, Heidelberg (May 2000). https://doi.org/10.1007/3-540-45539-6_19
26. Shoup, V.: Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332 (2004), https://eprint.iacr.org/2004/332

Appendix

A Game-Based Security Definitions

A.1 One Way Function Schemes

A One Way Function (OWF) Π is a pair $\Pi = (S, F)$, where S is a Probabilistic Polynomial Time Algorithm (PPT) and F is a Polynomial Time Algorithm (PT). The role of S is sampling values from F's domain, whereas the role of F is to actually compute the function. Definition 15, which captures the security of OWF schemes, makes use of oracles $\mathcal{O}_Y$ and $\mathcal{O}_S$, which, for an OWF $\Pi = (S, F)$ are defined as:

Image Generation Oracle: $\mathcal{O}_Y(i \in \mathbb{N})$

1. On the first call on index $i \in \mathbb{N}$, compute $x \leftarrow S(1^k)$ and store $(i, x, y \coloneqq F(x))$; output y;
2. On subsequent calls, simply output y.

Submission Oracle: $\mathcal{O}_S(i \in \mathbb{N}, x)$

1. On the first call on i (to either this oracle or to $\mathcal{O}_Y$), compute $x \leftarrow S(1^k)$ and store $(i, x, y \coloneqq F(x))$; the oracle does not give any output;
2. On subsequent calls, the oracle simply does not perform any action nor give any output.

Definition 15. *Consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}^{\mathrm{OWF}}$*:*

1. $\mathbf{A}^{\mathcal{O}_Y, \mathcal{O}_S}$*.*

$\mathbf{A}$ *wins the game if it makes a query to* $\mathcal{O}_S$ *on an input* (i, x) *such that* $F(x) = \mathcal{O}_Y(i)$*.*

$\mathbf{A}$*'s advantage in winning the One Way Function security game is defined as*

$$Adv^{\mathrm{OWF}}(\mathbf{A}) \coloneqq \Pr[\mathbf{A}\mathbf{G}^{\mathrm{OWF}} = \mathtt{win}].$$

An adversary $\mathbf{A}$ $(\varepsilon_{\mathrm{OWF}}, t)$-breaks the (n)-One-Wayness of OWF Π if it runs in time t, queries oracles $\mathcal{O}_Y$ and $\mathcal{O}_S$ on at most n different indices $i \in \mathbb{N}$, and satisfies $Adv^{\mathrm{OWF}}(\mathbf{A}) \geq \varepsilon_{\mathrm{OWF}}$. We say Π is $(\varepsilon_{\mathrm{OWF}}, t, n)$-secure if there is no such adversary.

A.2 Public Key Encryption Schemes

A Public Key Encryption (PKE) scheme Π with message space $\mathcal{M}$ is a triple of PPTs $\Pi = (G, E, D)$. Below we state the multi-user multi-challenge variants of Correctness and IND-CPA and IK-CPA security for PKE schemes (first introduced in [11] and [3], respectively). Throughout the rest of this section, let $\Pi = (G, E, D)$ be a PKE scheme with message space $\mathcal{M}$. As before, we assume the game systems of the following definitions have (an implicitly defined) security parameter k.

Definition 16, which captures the correctness of PKE schemes, provides adversaries with access to oracles $\mathcal{O}_{PK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$:

Secret Key Generation Oracle: $\mathcal{O}_{SK}(B_j)$

1. On the first call on B_j, compute and store $(\mathtt{pk}_j, \mathtt{sk}_j) \leftarrow G(1^k)$; output $(\mathtt{pk}_j, \mathtt{sk}_j)$;
2. On subsequent calls, simply output $(\mathtt{pk}_j, \mathtt{sk}_j)$.

Public Key Generation Oracle: $\mathcal{O}_{PK}(B_j)$

1. $(\mathtt{pk}_j, \mathtt{sk}_j) \leftarrow \mathcal{O}_{SK}(B_j)$; output $\mathtt{pk}_j$.

Encryption Oracle: $\mathcal{O}_E(B_j, m; r)$

1. If r is given as input, encrypt m under $\mathtt{pk}_j$ (B_j's public key, as generated by $\mathcal{O}_{PK}$) using r as random tape; if r is not given as input create a fresh encryption of m under $\mathtt{pk}_j$;
2. Output the resulting ciphertext back to the adversary.

Decryption Oracle: $\mathcal{O}_D(B_j, c)$

1. Decrypt c using $\mathtt{sk}_j$ (B_j's secret key, as generated by $\mathcal{O}_{PK}$);
2. Output the resulting plaintext back to the adversary (or $\perp$ if decryption failed).

Definition 16. *Consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}^{\mathsf{Corr}}$*:*

– $\mathbf{A}^{\mathcal{O}_{SK}, \mathcal{O}_{PK}, \mathcal{O}_E, \mathcal{O}_D}$

$\mathbf{A}$ *wins the game if there are two queries* q_E *and* q_D *to* $\mathcal{O}_E$ *and* $\mathcal{O}_D$*, respectively, where* q_E *has input* $(B_j, m; r)$ *and* q_D *has input* $(B_j{}', c)$*, the input* c *in* q_D *is the output of* q_E*,* $B_j = B_j{}'$*, and the output of* q_D *is not* m*.*

The advantage of $\mathbf{A}$ *in winning the Correctness game, denoted* $Adv^{\mathsf{Corr}}(\mathbf{A})$*, is the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{Corr}}$ *as described above.*

A (computationally unbounded) adversary $\mathbf{A}$ ($\varepsilon_{\mathsf{Corr}}$)-breaks the ($n$)-Correctness of a PKE scheme Π if $\mathbf{A}$ queries $\mathcal{O}_{PK}$, $\mathcal{O}_E$ and $\mathcal{O}_D$ on at most n different parties and satisfies $Adv^{\mathsf{Corr}}(\mathbf{A}) \geq \varepsilon_{\mathsf{Corr}}$.

The IND-CPA game systems provide adversaries with access to oracle $\mathcal{O}_{PK}$ described above, and to an additional oracle $\mathcal{O}_E$ which behaves as follows:

Encryption Oracle: $\mathcal{O}_E(B_j, m_0, m_1)$

1. For game system $\mathbf{G}_{\mathbf{b}}^{\mathsf{IND\text{-}CPA}}$, the oracle encrypts $m_{\mathbf{b}}$ under B_j's public key, $\mathtt{pk}_j$, creating a fresh ciphertext c;
2. The oracle outputs the resulting ciphertext c back to the adversary.

Definition 17. *For* $\mathbf{b} \in \{0, 1\}$*, consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}_{\mathbf{b}}^{\mathsf{IND\text{-}CPA}}$*:*

– $b' \leftarrow \mathbf{A}^{\mathcal{O}_{PK}, \mathcal{O}_E}$

$\mathbf{A}$ *wins the game if* $b' = \mathbf{b}$ *and for every query* $\mathcal{O}_E(B_j, m_0, m_1)$*,* $|m_0| = |m_1|$*.*

We define the advantage of $\mathbf{A}$ *in winning the* IND-CPA *security game as*

$$Adv^{\mathsf{IND\text{-}CPA}}(\mathbf{A}) := \left| \Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{\mathsf{IND\text{-}CPA}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{\mathsf{IND\text{-}CPA}} = \mathtt{win}] - 1 \right|.$$

Similarly to the IND-CPA game systems, the IK-CPA game systems provide adversaries with access to oracle $\mathcal{O}_{PK}$ and to an oracle $\mathcal{O}_E$ which behaves as follows:

Encryption Oracle: $\mathcal{O}_E(B_{j,0}, B_{j,1}, m)$

1. For game system $\mathbf{G}_{\mathbf{b}}^{\textsf{IK-CPA}}$, encrypt m under $B_{j,\mathbf{b}}$'s public key, $\texttt{pk}_{j,\mathbf{b}}$, creating a fresh ciphertext c;
2. Output the resulting ciphertext c back to the adversary.

Definition 18. *Consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}_{\mathbf{b}}^{\textsf{IK-CPA}}$*, with* $\mathbf{b} \in \{0, 1\}$*:*

– $b' \leftarrow \mathbf{A}^{\mathcal{O}_{PK}, \mathcal{O}_E}$

$\mathbf{A}$ *wins the game if* $b' = \mathbf{b}$*.*

We define the advantage of $\mathbf{A}$ *in winning the* IK-CPA *security game as*

$$Adv^{\textsf{IK-CPA}}(\mathbf{A}) := \left| \Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{\textsf{IK-CPA}} = \texttt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{\textsf{IK-CPA}} = \texttt{win}] - 1 \right|.$$

We say $\mathbf{A}$ $(\varepsilon_{\textsf{IND-CPA}}, t)$-breaks (resp. $(\varepsilon_{\textsf{IK-CPA}}, t)$-breaks) the (n, q_E)-IND-CPA (resp. (n, q_E)-IK-CPA) security of a PKE scheme Π if $\mathbf{A}$ runs in time at most t, queries the oracles it has access to on at most n different parties, makes at most q_E queries to oracle $\mathcal{O}_E$, and satisfies $Adv^{\textsf{IND-CPA}}(\mathbf{A}) \geq \varepsilon_{\textsf{IND-CPA}}$ (resp. $Adv^{\textsf{IK-CPA}}(\mathbf{A}) \geq \varepsilon_{\textsf{IK-CPA}}$).

Finally, Π is $(\varepsilon_{\textsf{Corr}}, \varepsilon_{\textsf{IND-CPA}}, \varepsilon_{\textsf{IK-CPA}}, t, n, q_E)$-secure if no adversary $\mathbf{A}$ $(\varepsilon_{\textsf{IND-CPA}}, t)$-breaks the (n, q_E)-IND-CPA security of Π nor $(\varepsilon_{\textsf{IK-CPA}}, t)$-breaks the (n, q_E)-IK-CPA security of Π, and no (possibly computationally unbounded) adversary $(\varepsilon_{\textsf{Corr}})$-breaks the (n)-Correctness of Π.

A.3 Symmetric Encryption Schemes

A Symmetric Encryption (SKE) scheme Π with message space $\mathcal{M}$ is a triple of PPTs $\Pi = (G, E, D)$. Below we state the (Perfect) Correctness notion and the IND-CPA security notion for SKE schemes. Throughout the rest of this section, let $\Pi = (G, E, D)$ be an SKE scheme. As before, we assume the game systems of the following definitions have (an implicitly defined) security parameter k.

Definition 19. *Consider a* SKE *scheme* $\Pi = (G, E, D)$ *with message space* $\mathcal{M}$*. We say* Π *is correct if for every* $m \in \mathcal{M}$ *and every key* $k_{\texttt{sym}} \in Supp(G(1^k))$*:*

$$\Pr[D(k_{\texttt{sym}}, E(k_{\texttt{sym}}, m)) = m] = 1.$$

The (One Time) 1-IND-CPA game systems provide adversaries with access to oracle $\mathcal{O}_K$ described above, and to an additional oracle $\mathcal{O}_E$ which behaves as follows:

Encryption Oracle: $\mathcal{O}_E(i \in \mathbb{N}, m_0, m_1)$

1. For game system $\mathbf{G}_{\mathbf{b}}^{\textsf{1-IND-CPA}}$, the oracle encrypts $m_{\mathbf{b}}$ under the i-th key, $\texttt{k}_i$, creating a fresh ciphertext c;

2. The oracle outputs the resulting ciphertext c.

Definition 20. *For* $\mathbf{b} \in \{0,1\}$*, consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}_{\mathbf{b}}^{\textsf{1-IND-CPA}}$*:*

– $b' \leftarrow \mathbf{A}^{\mathcal{O}_E}$

$\mathbf{A}$ *wins the game if* $b' = \mathbf{b}$ *and for every query* $\mathcal{O}_E(i, m_0, m_1)$*,* $|m_0| = |m_1|$ *and there is no other query to* $\mathcal{O}_E$ *on the same index* i*.*

$\mathbf{A}$*'s advantage in winning the (One Time) 1-*IND-CPA *security game is defined as*

$$Adv^{\textsf{1-IND-CPA}}(\mathbf{A}) \coloneqq \left| \Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{\textsf{1-IND-CPA}} = \texttt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{\textsf{1-IND-CPA}} = \texttt{win}] - 1 \right|.$$

We say $\mathbf{A}$ $(\varepsilon_{\textsf{1-IND-CPA}}, t)$-breaks the (q_E)-1-IND-CPA security of an SKE scheme Π if $\mathbf{A}$ runs in time at most t, makes at most q_E queries to oracle $\mathcal{O}_E$ and satisfies $Adv^{\textsf{1-IND-CPA}}(\mathbf{A}) \geq \varepsilon_{\textsf{1-IND-CPA}}$.

Finally, we say Π is $(\varepsilon_{\textsf{1-IND-CPA}}, t, q_E)$-secure if Π is perfectly correct (see Definition 19) and no adversary $\mathbf{A}$ $(\varepsilon_{\textsf{1-IND-CPA}}, t)$-breaks the (q_E)-1-IND-CPA security of Π.

A.4 Non Interactive Zero Knowledge Schemes

For a binary relation R, let L_R be the language $L_R \coloneqq \{x \mid \exists w, (x, w) \in R\}$ induced by R. A *Non Interactive Proof System* (NIPS) for L_R is a triple of PPT algorithms $\Pi = (G, P, V)$ where:

- $G(1^k)$: given security parameter 1^k, outputs a common reference string $\texttt{crs}$;
- $P_{\texttt{crs}}(x, w)$: given a common reference string $\texttt{crs}$ and a statement-witness pair $(x, w) \in R$, outputs a proof p;
- $V_{\texttt{crs}}(x, p)$: given a common reference string $\texttt{crs}$, a statement x and a proof p, either accepts, outputting `valid` $(= 1)$ or rejects, outputting `invalid` $(= 0)$.

In the following definitions, let $\Pi = (G, P, V)$ be a NIPS for a relation R, and let k be the security parameter. The security notions below (Definitions 21 and 22) provide adversaries with access to oracles $\mathcal{O}_S$ and $\mathcal{O}_V$, defined as:

CRS Generation Oracle: $\mathcal{O}_S$
1. On the first call, compute and store $\texttt{crs} \leftarrow G(1^k)$; output $\texttt{crs}$;
2. On subsequent calls, output the previously generated $\texttt{crs}$.

Verify Oracle: $\mathcal{O}_V(x, p)$
1. Compute $b = V_{\texttt{crs}}(x, p)$; output b.

Definition 21 additionally provides adversaries with access to an oracle $\mathcal{O}_P$:

Prove Oracle: $\mathcal{O}_P(x, w)$
1. Compute $p = P_{\texttt{crs}}(x, w)$; output p.

Definition 21. *Consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}^{\textsf{Complete}}$*:*

– $\mathbf{A}^{\mathcal{O}_S,\mathcal{O}_P,\mathcal{O}_V}$

$\mathbf{A}$ *wins the game if there are two queries* q_P *and* q_V *to* $\mathcal{O}_P$ *and* $\mathcal{O}_V$*, respectively, where* q_P *has input* (x, w) *and* q_V *has input* (x', p)*, satisfying* $x = x'$*, the input* p *in* q_V *is the output of* q_P*, the output of* q_V *is* `invalid`*, and* $(x, w) \in R$*.*

The advantage of $\mathbf{A}$ *in winning the Completeness game, denoted* $Adv^{\mathsf{Complete}}(\mathbf{A})$*, corresponds to the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{Complete}}$ *as described above.*

We say that an adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{Complete}}, t)$-breaks the (q_P, q_V)-Completeness of a NIPS scheme Π if $\mathbf{A}$ runs in time at most t, makes at most q_P and q_V queries to oracles $\mathcal{O}_P$ and $\mathcal{O}_V$, respectively, and satisfies $Adv^{\mathsf{Complete}}(\mathbf{A}) \geq \varepsilon_{\mathsf{Complete}}$.

Definition 22. *Consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}^{\mathsf{Sound}}$*:*

– $\mathbf{A}^{\mathcal{O}_S,\mathcal{O}_V}$

$\mathbf{A}$ *wins the game if there is a query to* $\mathcal{O}_V$ *on input* (x, p)*, satisfying* $x \notin L_R$*, such that the oracle outputs* `valid`*.*

The advantage of $\mathbf{A}$ *in winning the Soundness game corresponds to the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{Sound}}$ *as described above and is denoted* $Adv^{\mathsf{Sound}}(\mathbf{A})$*.*

An adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{Sound}}, t)$-breaks the (q_V)-Soundness of a NIPS scheme Π if $\mathbf{A}$ runs in time at most t, makes at most q_V queries to $\mathcal{O}_V$ and satisfies $Adv^{\mathsf{Sound}}(\mathbf{A}) \geq \varepsilon_{\mathsf{Sound}}$.

A NIZK scheme $\Pi = (G, P, V, S = (S_G, S_P))$ for a relation R consists of a NIPS scheme $\Pi' = (G, P, V)$ for R and a simulator $S = (S_G, S_P)$, where:

– $S_G(1^k)$: given security parameter 1^k, outputs a pair $(\mathtt{crs}, \tau)$;
– $S_{P(\mathtt{crs},\tau)}(x)$: given a pair $(\mathtt{crs}, \tau)$ and a statement x, outputs a proof p.

Consider a NIZK scheme $\Pi = (G, P, V, S = (S_G, S_P))$. The following security notion, which defines game systems $\mathbf{G}_{\mathbf{0}}^{\mathsf{ZK}}$ and $\mathbf{G}_{\mathbf{1}}^{\mathsf{ZK}}$, provides adversaries with access to two oracles, $\mathcal{O}_S$ and $\mathcal{O}_P$, whose behavior depends on the underlying game system. For $\mathbf{G}_{\mathbf{b}}^{\mathsf{ZK}}$ (with $\mathbf{b} \in \{0, 1\}$):

CRS Generation Oracle: $\mathcal{O}_S$

1. On the first call, compute and store $\mathtt{crs} \leftarrow G(1^k)$ if $\mathbf{b} = \mathbf{0}$, and $(\mathtt{crs}, \tau) \leftarrow S_G(1^k)$ if $\mathbf{b} = \mathbf{1}$; output $\mathtt{crs}$;
2. On subsequent calls, output the previously generated $\mathtt{crs}$.

Prove Oracle: $\mathcal{O}_P(x, w)$

– If $\mathbf{b} = \mathbf{0}$, output $\pi \leftarrow P_{\mathtt{crs}}(x, w)$;
– If $\mathbf{b} = \mathbf{1}$, output $\pi \leftarrow S_{P(\mathtt{crs},\tau)}(x)$.

Definition 23. *For* $\mathbf{b} \in \{0, 1\}$*, consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}_{\mathbf{b}}^{\mathsf{ZK}}$*:*

– $b' \leftarrow \mathbf{A}^{\mathcal{O}_S,\mathcal{O}_P}$

$\mathbf{A}$ *wins the game if* $b' = \mathbf{b}$ *and every query* $\mathcal{O}_P(x, w)$ *satisfies* $(x, w) \in R$.

The advantage of $\mathbf{A}$ *in winning the Zero-Knowledge security game for* Π *is*

$$Adv^{\mathsf{ZK}}(\mathbf{A}) := \left| \Pr[\mathbf{A}\mathbf{G}_{\mathbf{0}}^{\mathsf{ZK}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_{\mathbf{1}}^{\mathsf{ZK}} = \mathtt{win}] - 1 \right|.$$

We say that an adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{ZK}}, t)$-breaks the (q_P)-ZK security of a NIZK scheme Π if it makes at most q_P queries to $\mathcal{O}_P$ and satisfies $Adv^{\mathsf{ZK}}(\mathbf{A}) \geq \varepsilon_{\mathsf{ZK}}$.

We now introduce Simulation Soundness for NIZK [24]. The game system defined by this notion provides adversaries with access to oracles $\mathcal{O}_S$, $\mathcal{O}_P$ and $\mathcal{O}_V$ defined as:

CRS Generation Oracle: $\mathcal{O}_S$

1. On the first call, compute and store $(\mathtt{crs}, \tau) \leftarrow S_G(1^k)$; output $\mathtt{crs}$;
2. On subsequent calls, output the previously generated $\mathtt{crs}$.

Prove Oracle: $\mathcal{O}_P(x)$

1. Compute $p = S_{P(\mathtt{crs},\tau)}(x)$; output p.

Verify Oracle: $\mathcal{O}_V(x, p)$

1. Compute $b = V_{\mathtt{crs}}(x, p)$; output b.

Definition 24. *Consider the following game played between an adversary* $\mathbf{A}$ *and game system* $\mathbf{G}^{\mathsf{SS}}$*:*

- $\mathbf{A}^{\mathcal{O}_S, \mathcal{O}_P, \mathcal{O}_V}$

$\mathbf{A}$ *wins the game if it makes a query* $\mathcal{O}_V(x, p)$ *with* $x \notin L_R$ *that outputs* `valid` *and no query* $\mathcal{O}_P(x)$ *output* p.

The advantage of $\mathbf{A}$ *in winning the Simulation Soundness game, denoted* $Adv^{\mathsf{SS}}(\mathbf{A})$*, is the probability that* $\mathbf{A}$ *wins game* $\mathbf{G}^{\mathsf{SS}}$ *as described above.*

An adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{SS}}, t)$-breaks the (q_P, q_V)-Simulation Soundness of a NIZK scheme Π if it makes at most q_P and q_V queries to $\mathcal{O}_P$ and $\mathcal{O}_V$, respectively, and satisfies $Adv^{\mathsf{SS}}(\mathbf{A}) \geq \varepsilon_{\mathsf{SS}}$.

Finally, we say that a NIZK scheme Π is $(\varepsilon_{\mathsf{Complete}}, \varepsilon_{\mathsf{Sound}}, \varepsilon_{\mathsf{ZK}}, \varepsilon_{\mathsf{SS}}, t, q_P, q_V)$-secure if no adversary $\mathbf{A}$ $(\varepsilon_{\mathsf{Complete}}, t)$-breaks the (q_P, q_V)-Completeness of Π, $(\varepsilon_{\mathsf{Sound}}, t)$-breaks the (q_V)-Soundness of Π, $(\varepsilon_{\mathsf{ZK}}, t)$-breaks the (q_P)-Zero-Knowledge of Π, or $(\varepsilon_{\mathsf{SS}}, t)$-breaks the (q_P, q_V)-Simulation Soundness of Π.

B Full Proofs

B.1 Helper Claims

We now establish two (straightforward) results that allow to simplify the MDVS and PKEBC security proofs ahead.

B.1.1 One Way Function Image Collision Resistance

Definition 25 (n-Instance ε-Image Collision-Resistance). *A* OWF $\Pi = (S, F)$ *is n-Instance ε-Image Collision-Resistant if*

$$\Pr\left[\left|\{\Pi.F(x_1), \ldots, \Pi.F(x_n)\}\right| < n \;\middle|\; \begin{matrix} x_1 \leftarrow \Pi.S(1^k) \\ \ldots \\ x_n \leftarrow \Pi.S(1^k) \end{matrix} \right] \leq \varepsilon.$$

Lemma 1. *If Π is $(\varepsilon_{\mathrm{OWF}}, t, n)$-secure, with $t \gtrapprox n \cdot (t_S + t_F)$—where t_S and t_F are, respectively, the times to run S and F—then Π is n-Instance ε-Image Collision-Resistant, with $\varepsilon \leq 2 \cdot \varepsilon_{\mathrm{OWF}}$.*

Proof. To prove this result we give an adversary $\mathbf{A}^n$ such that $2 \cdot Adv^{\mathrm{OWF}}(\mathbf{A}^n) \geq \varepsilon$. Since $Adv^{\mathrm{OWF}}(\mathbf{A}^n) \leq \varepsilon_{\mathrm{OWF}}$, it then follows that $\varepsilon \leq 2 \cdot \varepsilon_{\mathrm{OWF}}$.

Consider the following adversary $\mathbf{A}^n$. First, $\mathbf{A}^n$ samples an n bit long vector $\vec{b}$, each bit being picked independently and uniformly at random. For each index $i \in \{1, \ldots, n\}$, if $b_i = 0$ then $\mathbf{A}^n$ queries $\mathcal{O}_Y$ on input i, and sets $y_i = \mathcal{O}_Y(i)$, and if $b_i = 1$ then $\mathbf{A}^n$ samples an element x_i from the domain of the one way function $\Pi.S(1^k)$, saves x_i, and sets $y_i = \Pi.F(x_i)$. If there are no two indices $i, j \in \{1, \ldots, n\}$ such that $b_i \neq b_j$ and $y_i = y_j$, $\mathbf{A}^n$ aborts. Otherwise, for the least $i \in \{1, \ldots, n\}$ for which $b_i = 0$ and there exists $j \in \{1, \ldots, n\}$ with $b_j = 1$ and $y_i = y_j$, $\mathbf{A}^n$ makes a query $\mathcal{O}_S(i, x_j)$. Since algorithm F is deterministic, it follows that $\Pi.F(x_j) = y_j = y_i$, and so $\mathbf{A}^n$ wins the game.

Note that if there are two indices $i, j \in \{1, \ldots, n\}$ with $y_i = y_j$, the only case where $\mathbf{A}^n$ does not win the game is if $b_i = b_j$. Given this only happens with probability at most $\frac{1}{2}$ and this event is independent from the existence of two indices $i, j \in \{1, \ldots, n\}$ such that $y_i = y_j$, it follows $2 \cdot Adv^{\mathrm{OWF}}(\mathbf{A}^n) \geq \varepsilon$. □

B.1.2 Public Key Collision Resistance

Definition 26 (n-Party ε-Public Key Collision-Resistance). PKE *scheme* $\Pi = (G, E, D)$ *is n-Party ε-Public Key Collision-Resistant if*

$$\Pr\left[\left|\{\mathtt{pk}_1, \ldots, \mathtt{pk}_n\}\right| < n \;\middle|\; \begin{matrix} (\mathtt{pk}_1, \mathtt{sk}_1) \leftarrow \Pi.G(1^k) \\ \ldots \\ (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \Pi.G(1^k) \end{matrix} \right] \leq \varepsilon.$$

Lemma 2. *If Π is $(\varepsilon_{\mathsf{Corr}}, \varepsilon_{\mathsf{IND\text{-}CPA}}, \varepsilon_{\mathsf{IK\text{-}CPA}}, t, n, q_E)$-secure, with $t \gtrapprox n \cdot t_G + t_D$—where t_G and t_D are, respectively, the times to run $\Pi.G$ and $\Pi.D$—then Π is n-Party ε-Public Key Collision-Resistant, with $\varepsilon \leq 2 \cdot \varepsilon_{\mathsf{IND\text{-}CPA}} + \varepsilon_{\mathsf{Corr}}$.*

Proof. To prove this result we give two adversaries—$\mathbf{A}^{n,m_0,m_1}_{\mathsf{IND\text{-}CPA}}$ for the $\mathsf{IND\text{-}CPA}$ security games and $\mathbf{A}^{n,m_0,m_1}_{\mathsf{Corr}}$ for the Correctness game—such that for these adversaries

$$2 \cdot Adv^{\mathsf{IND\text{-}CPA}}(\mathbf{A}^{n,m_0,m_1}_{\mathsf{IND\text{-}CPA}}) + Adv^{\mathsf{Corr}}(\mathbf{A}^{n,m_0,m_1}_{\mathsf{Corr}}) \geq \varepsilon.$$

Since $Adv^{\textsf{IND-CPA}}(\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}) \leq \varepsilon_{\textsf{IND-CPA}}$ and $Adv^{\textsf{Corr}}(\mathbf{A}^{n,m_0,m_1}_{\textsf{Corr}}) \leq \varepsilon_{\textsf{Corr}}$, it then follows that $\varepsilon \leq 2 \cdot \varepsilon_{\textsf{IND-CPA}} + \varepsilon_{\textsf{Corr}}$.

Consider the following adversary $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ for the IND-CPA game of Π. First, $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ samples an n bit long vector $\vec{b}$, each bit being picked independently and uniformly at random. Next, $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ queries $\mathcal{O}_{PK}$ on each party B_i for which $b_i = 0$ and sets $\texttt{pk}_i = \mathcal{O}_{PK}(B_i)$. Similarly, $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ samples a key pair using $\Pi.G$ for each party B_i for which $b_i = 1$ and sets $(\texttt{pk}_i, \texttt{sk}_i) \leftarrow G(1^k)$ (where k is the security parameter). If there are no two indices $i, j \in \{1, \dots, n\}$ such that $b_i \neq b_j$ and $\texttt{pk}_i = \texttt{pk}_j$, $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ aborts. Otherwise, for the least $i \in \{1, \dots, n\}$ for which $b_i = 0$ and there exists $j \in \{1, \dots, n\}$ with $b_j = 1$ and $\texttt{pk}_i = \texttt{pk}_j$, $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ makes a query $\mathcal{O}_E(B_i, m_0, m_1)$; letting c be the output of this query, $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ tries decrypting c using $\texttt{sk}_j$ (here, j is assumed to be the least $j \in \{1, \dots, n\}$ such that $b_j = 1$ and $\texttt{pk}_i = \texttt{pk}_j$). Finally, $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ outputs 0 if the decryption resulted in m_0, 1 if the decryption resulted in m_1, and otherwise aborts.

The adversary $\mathbf{A}^{n,m_0,m_1}_{\textsf{Corr}}$ for the Correctness game of Π uses oracle $\mathcal{O}_{PK}$ to sample all the n parties' public keys. For each party B_i and each possible sequence r of random coins used by $\Pi.E$, $\mathbf{A}^{n,m_0,m_1}_{\textsf{Corr}}$ makes queries $\mathcal{O}_E(B_i, m_0; r)$ and $\mathcal{O}_E(B_i, m_1; r)$. Letting c_0 and c_1 be the respective outputs of the two oracle queries above, $\mathbf{A}^{n,m_0,m_1}_{\textsf{Corr}}$ makes queries $\mathcal{O}_D(B_i, c_0)$ and $\mathcal{O}_D(B_i, c_1)$.

Note that if there are two parties B_i and B_j with equal public keys, the only case where $\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}$ may not win the IND-CPA games is when either $b_i = b_j$ or the scheme does not work correctly. Given $\Pr[b_i = b_j] \leq \frac{1}{2}$ and the probability that the scheme does not work correctly is bounded by $Adv^{\textsf{Corr}}(\mathbf{A}^{n,m_0,m_1}_{\textsf{Corr}})$, it follows $2 \cdot Adv^{\textsf{IND-CPA}}(\mathbf{A}^{n,m_0,m_1}_{\textsf{IND-CPA}}) + Adv^{\textsf{Corr}}(\mathbf{A}^{n,m_0,m_1}_{\textsf{Corr}}) \geq \varepsilon$. □

B.2 MDVS Construction Security Proofs

In this section we give the formal security theorems and the (corresponding) full proofs for the MDVS construction given in Section 4.4.

B.2.1 Proof of Correctness

Theorem 4. *If Π_{PKE} is*

$$(\varepsilon_{\text{PKE-}\textsf{Corr}}, \varepsilon_{\text{PKE-}\textsf{IND-CPA}}, \varepsilon_{\text{PKE-}\textsf{IK-CPA}}, t_{\text{PKE}}, n_{\text{PKE}}, q_{E\,\text{PKE}})\text{-secure,} \tag{B.1}$$

and Π_{NIZK} is

$$(\varepsilon_{\text{NIZK-}\textsf{Complete}}, \varepsilon_{\text{NIZK-}\textsf{Sound}}, \varepsilon_{\text{NIZK-}\textsf{ZK}}, \varepsilon_{\text{NIZK-}\textsf{SS}}, t_{\text{NIZK}}, q_{P\,\text{NIZK}}, q_{V\,\text{NIZK}})\text{-secure,} \tag{B.2}$$

then no adversary $\mathbf{A}$ (ε, t)-breaks Π's

$$(n_V := n_{\text{PKE}}, q_S := q_{P\,\text{NIZK}}, q_V := q_{V\,\text{NIZK}})\text{-Correctness,}$$

with $\varepsilon > \varepsilon_{\text{NIZK-}\textsf{Complete}} + \varepsilon_{\text{PKE-}\textsf{Corr}}$ and with $t_{\text{NIZK}} \approx t + t_{\textsf{Corr}}$, where $t_{\textsf{Corr}}$ is the time to run Π's $\mathbf{G}^{\textsf{Corr}}$ game.

Proof. This proof proceeds in a sequence of games [4, 26].

$\mathbf{G}^{\mathsf{Corr}} \rightsquigarrow \mathbf{G}^1$**:** $\mathbf{G}^1$ is just like the original game $\mathbf{G}^{\mathsf{Corr}}$, except that in $\mathbf{G}^1$ for each signature $\sigma := (p, \vec{c}, c_{\mathsf{pp}})$ output by a query $\mathcal{O}_S(A_i, \vec{V}, m)$, if $\mathcal{O}_V$ is queried on any input $(A_i{}', B_j, \vec{V}', m', \sigma')$ with $(A_i, \vec{V}, m, \sigma) = (A_i{}', \vec{V}', m', \sigma')$ and $B_j \in \vec{V}$, it no longer verifies p's validity and simply proceeds as if it were valid.

Note that one can reduce distinguishing $\mathbf{G}^{\mathsf{Corr}}$ and $\mathbf{G}^1$ to breaking Π_{NIZK}'s completeness: since the reduction holds the secret keys of every signer and of every verifier, it can trivially handle any oracle queries. Furthermore, the reduction makes at most one Π_{NIZK}-$\mathcal{O}_P$ query for each $\mathcal{O}_S$ query, and at most one Π_{NIZK}-$\mathcal{O}_V$ query for each $\mathcal{O}_V$ query. Since $\mathbf{A}$ only makes up to $q_S \leq q_{P\,\mathrm{NIZK}}$ queries to $\mathcal{O}_S$ and $q_V \leq q_{V\,\mathrm{NIZK}}$ queries to $\mathcal{O}_V$, it follows from Equation B.2 that no adversary $(\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Complete}}, t_{\mathrm{NIZK}})$-breaks the $(q_{P\,\mathrm{NIZK}}, q_{V\,\mathrm{NIZK}})$-Completeness of Π_{NIZK}, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{\mathsf{Corr}} = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Complete}}.$$

$\mathbf{G}^1 \rightsquigarrow \mathbf{G}^2$**.** Game $\mathbf{G}^2$ is just like $\mathbf{G}^1$, except that now for each signature $\sigma := (p, \vec{c}, c_{\mathsf{pp}})$ output by a query $\mathcal{O}_S(A_i, \vec{V}, m)$, if $\mathcal{O}_V$ is queried on any input $(A_i{}', B_j, \vec{V}', m', \sigma')$, with $(A_i, \vec{V}, m, \sigma) = (A_i{}', \vec{V}', m', \sigma')$ and $B_j \in \vec{V}$, and letting $l \in \{1, \ldots, |\vec{V}|\}$ be the least index such that $B_j = V_l$, $\mathcal{O}_V$ no longer tries decrypting ciphertext $c_{l,b} \in (c_{l,0}, c_{l,1})$—where b is the secret bit in B_j's secret key and $(c_{l,0}, c_{l,1}) \in \vec{c}$—and instead simply outputs 1 as if the decryption of $c_{l,b}$ had output 1.

It is easy to see that one can reduce distinguishing these two games to breaking Π_{PKE}'s correctness: since the reduction holds all secret keys, it can handle any oracle queries. Noting that an adversary $\mathbf{A}$ can only query for the verifier public keys of at most $n_V \leq n_{\mathrm{PKE}}$ parties and since the reduction only has to rely on Π_{PKE}-$\mathcal{O}_{SK}$ oracle to generate at most one key-pair per party (namely, $(\mathtt{pk}_b, \mathtt{sk}_b)$), it follows from Equation B.1 that no adversary $(\varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}})$-breaks the (n_{PKE})-Correctness of Π_{PKE}, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}}.$$

To conclude, since no adversary can win game $\mathbf{G}^2$, $\Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}] = 0$. □

B.2.2 Proof of Consistency

Theorem 5. *If* Π_{PKE} *is*

$$\begin{aligned}(&\varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}}, \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IND}\text{-}\mathsf{CPA}}, \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IK}\text{-}\mathsf{CPA}},\\ &t_{\mathrm{PKE}}, n_{\mathrm{PKE}}, q_{E\,\mathrm{PKE}})\text{-}secure,\end{aligned} \tag{B.3}$$

with $n_{\mathrm{PKE}} \geq 1$*,* Π_{NIZK} *is*

$$\begin{aligned}(&\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Complete}}, \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Sound}}, \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{ZK}}, \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{SS}},\\ &t_{\mathrm{NIZK}}, q_{P\,\mathrm{NIZK}}, q_{V\,\mathrm{NIZK}})\text{-}secure,\end{aligned} \tag{B.4}$$

Π_{OWF} *is*

$$(\varepsilon_{\mathrm{OWF}}, t_{\mathrm{OWF}}, n_{\mathrm{OWF}})\text{-}secure, \tag{B.5}$$

and $\Pi_{\mathrm{NIZK}}.V$ *is a deterministic algorithm, then no adversary* $\mathbf{A}$ (ε, t)*-breaks* Π*'s*

$$(n_V \coloneqq \min(n_{\mathrm{PKE}}, n_{\mathrm{OWF}}), q_V \coloneqq q_{V\,\mathrm{NIZK}})\text{-}Consistency,$$

with $\varepsilon > 3 \cdot \varepsilon_{\text{PKE-Corr}} + \varepsilon_{\text{NIZK-Sound}} + 2 \cdot \varepsilon_{\mathrm{OWF}}$ *and with* $t_{\mathrm{NIZK}}, t_{\mathrm{OWF}} \approx t + t_{\mathsf{Cons}}$*, where* t_{Cons} *is the time to run* Π*'s* $\mathbf{G}^{\mathsf{Cons}}$ *game.*

Proof. We proceed via game hopping.

$\mathbf{G}^{\mathsf{Cons}} \rightsquigarrow \mathbf{G}^1$**:** $\mathbf{G}^1$ is just like $\mathbf{G}^{\mathsf{Cons}}$ except that in $\mathbf{G}^1$ the Π_{PKE} key pair $(\mathtt{pk}_0, \mathtt{sk}_0)$ sampled for each party B_j is assumed to be a correct one.

Note that one can reduce distinguishing these two games to breaking Π_{PKE}'s correctness: since the reduction holds all secret keys it can handle any oracle queries. Furthermore, given an adversary $\mathbf{A}$ can only query for the verifier public keys of at most $n_V \leq n_{\mathrm{PKE}}$ parties and given the reduction only has to rely on Π_{PKE}-$\mathcal{O}_{SK}$ oracle to generate at most one key-pair per party—namely, $(\mathtt{pk}_0, \mathtt{sk}_0)$—it follows from Equation B.3 that no adversary $(\varepsilon_{\text{PKE-Corr}})$-breaks the (n_{PKE})-Correctness of Π_{PKE}, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{\mathsf{Cons}} = \mathtt{win}]\right| \leq \varepsilon_{\text{PKE-Corr}}.$$

$\mathbf{G}^1 \rightsquigarrow \mathbf{G}^2$**:** This game hop is just like the previous one (i.e. $\mathbf{G}^{\mathsf{Cons}} \rightsquigarrow \mathbf{G}^1$), the only difference being that the key-pair which is assumed to be a correct one is now $(\mathtt{pk}_1, \mathtt{sk}_1)$. Hence,

$$\left|\Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}]\right| \leq \varepsilon_{\text{PKE-Corr}}.$$

$\mathbf{G}^2 \rightsquigarrow \mathbf{G}^3$**:** This step is similar to the previous ones, except that this time the key pair that is assumed to be a correct one is the public parameters' public key and the corresponding secret key (i.e. the key pair sampled by $\Pi.S$).

One can, once again, reduce distinguishing these games to breaking Π_{PKE}'s correctness: the reduction has all secret keys so it can handle any oracle queries. In contrast to the previous reductions, this one only has to rely on the Π_{PKE}-$\mathcal{O}_{PK}$ oracle to generate a single key-pair. Since $n_{\mathrm{PKE}} \geq 1$, it then follows from Equation B.3 that no adversary $(\varepsilon_{\text{PKE-Corr}})$-breaks the (1)-Correctness of Π_{PKE}, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^3 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}]\right| \leq \varepsilon_{\text{PKE-Corr}}.$$

$\mathbf{G}^3 \rightsquigarrow \mathbf{G}^4$**:** Game $\mathbf{G}^4$ is just $\mathbf{G}^3$ except that for each query $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma := (p, \vec{c}, c_{\mathsf{pp}}))$ in $\mathbf{G}^4$ it is assumed that if the NIZK proof p verifies as being valid then $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathsf{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$.

Once again, the reduction holds all secret keys and thus it can handle any oracle query. Moreover, because the reduction has a witness for every statement it has to produce a NIZK proof for, it can simply use $\Pi_{\mathrm{NIZK}}.P$ to generate the NIZK proofs. Regarding $\mathcal{O}_V$ queries, however, the reduction now relies on oracle Π_{NIZK}-$\mathcal{O}_V$ to verify the validity of each signature's the NIZK proof. Given the reduction only verifies at most one NIZK proof for each $\mathcal{O}_V$ query and since $q_V \le q_{V\,\mathrm{NIZK}}$, it follows from Equation B.4 that no adversary $(\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Sound}}, t_{\mathrm{NIZK}})$-breaks the $(q_{V\,\mathrm{NIZK}})$-Soundness of Π_{NIZK}, and so

$$\left|\Pr[\mathbf{AG}^4 = \mathtt{win}] - \Pr[\mathbf{AG}^3 = \mathtt{win}]\right| \le \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Sound}}.$$

To conclude we now prove the following claim:

Claim. $\Pr[\mathbf{AG}^4 = \mathtt{win}] \le 2 \cdot \varepsilon_{\mathrm{OWF}}$.

Proof. An adversary $\mathbf{A}$ can only win $\mathbf{G}^4$ if it makes two queries to $\mathcal{O}_V$ on inputs $(A, B_j, \vec{V}, m, \sigma := (p, \vec{c}, c_{\mathsf{pp}}))$ and $(A', {B_j}', \vec{V}', m', \sigma')$ satisfying $(A, \vec{V}, m, \sigma) = (A', \vec{V}', m', \sigma')$ and $B_j, {B_j}' \in \vec{V}$, and one of the queries outputs 1 while the other outputs 0.

Given $\Pi_{\mathrm{NIZK}}.V$ is a deterministic algorithm and one of the queries outputs 1, the NIZK proof p in the signature input to the $\mathcal{O}_V$ queries verifies as being valid both times. Furthermore, for the least $i, i' \in \{1, \ldots, |\vec{V}|\}$ such that $V_i = B_j$ and $V_{i'} = {B_j}'$, by the correctness of B_j's and ${B_j}'$'s key pairs, $c_{i,b}$ and $c_{i',b'}$ are encryptions of two different bits—b being the bit in B_j's secret key, and b' the bit in ${B_j}'$'s secret key. By the soundness of Π_{NIZK} it follows $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathsf{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$. On one hand, this implies c_{pp} is an encryption of a plaintext of the form $(m, b'', ((b_1, x_1), \ldots, (b_l, x_l)))$, and on the other hand, since $R_{\mathrm{MDVS}^{\mathsf{adap}}} \subseteq R_{\mathrm{MDVS}^{\mathsf{adap}}\text{-}\mathrm{Cons}}$ and $c_{i,b}$ and $c_{i',b'}$ are encryptions of two different bits, either x_i is such that $\Pi_{\mathrm{OWF}}.F(x_i) \in \{v_i.y_0, v_i.y_1\}$ or $x_{i'}$ is such that $\Pi_{\mathrm{OWF}}.F(x_{i'}) \in \{v_{i'}.y_0, v_{i'}.y_1\}$ (or both). The correctness of $\mathtt{pp.pk}$ implies that decrypting c_{pp} results in the plaintext above.

By Definition 2, if the adversary wins the game then it did not query $\mathcal{O}_{VK}$ on either B_j or ${B_j}'$. This in particular means that everything the adversary sees is independent of B_j's and ${B_j}'$'s secret key bits. Thus, if it is the case that $\Pi_{\mathrm{OWF}}.F(x_i) \in \{v_i.y_0, v_i.y_1\}$, with probability at least $\frac{1}{2}$, $\Pi_{\mathrm{OWF}}.F(x_i) = v_i.y_{\bar{b}}$, where $\bar{b} := 1 - b$ and b is the bit in B_j's secret key. Otherwise, if $\Pi_{\mathrm{OWF}}.F(x_{i'}) \in \{v_{i'}.y_0, v_{i'}.y_1\}$, with probability at least $\frac{1}{2}$, $\Pi_{\mathrm{OWF}}.F(x_{i'}) = v_{i'}.y_{\bar{b}}$, where $\bar{b} := 1 - b$ but this time b being the bit in ${B_j}'$'s secret key. It is easy to see that one can then reduce winning game $\mathbf{G}^4$ to breaking the security of the underlying Π_{OWF} by decrypting the ciphertext c_{pp} of signature σ. For each verifier B_l, and letting b be the bit in B_l's secret key, the $y_{\bar{b}}$ image in B_l's public key is now obtained via a query Π_{OWF}-$\mathcal{O}_Y$. Given $n_V \le n_{\mathrm{OWF}}$, it follows from Equation B.5 that no adversary $(\varepsilon_{\mathrm{OWF}}, t_{\mathrm{OWF}})$-breaks the (n_{OWF})-security of Π_{OWF}, implying $\Pr[\mathbf{AG}^4 = \mathtt{win}] \le 2 \cdot \varepsilon_{\mathrm{OWF}}$. □

B.2.3 Proof of Unforgeability

Theorem 6. *If $\mathit{\Pi}_{\mathrm{PKE}}$ is*

$$(\varepsilon_{\mathsf{PKE\text{-}Corr}}, \varepsilon_{\mathsf{PKE\text{-}IND\text{-}CPA}}, \varepsilon_{\mathsf{PKE\text{-}IK\text{-}CPA}}, \\ t_{\mathrm{PKE}}, n_{\mathrm{PKE}}, q_{E\,\mathrm{PKE}})\text{-}secure, \tag{B.6}$$

with $n_{\mathrm{PKE}} \geq 1$, $\mathit{\Pi}_{\mathrm{NIZK}}$ is

$$(\varepsilon_{\mathsf{NIZK\text{-}Complete}}, \varepsilon_{\mathsf{NIZK\text{-}Sound}}, \varepsilon_{\mathsf{NIZK\text{-}ZK}}, \varepsilon_{\mathsf{NIZK\text{-}SS}}, \\ t_{\mathrm{NIZK}}, q_{P\,\mathrm{NIZK}}, q_{V\,\mathrm{NIZK}})\text{-}secure, \tag{B.7}$$

and $\mathit{\Pi}_{\mathrm{OWF}}$ is

$$(\varepsilon_{\mathrm{OWF}}, t_{\mathrm{OWF}}, n_{\mathrm{OWF}})\text{-}secure, \tag{B.8}$$

with $t_{\mathrm{OWF}} \gtrapprox n_{\mathrm{OWF}} \cdot (t_S + t_F)$ (where t_S and t_F are, respectively, the times to run $\mathit{\Pi}_{\mathrm{OWF}}.S$ and $\mathit{\Pi}_{\mathrm{OWF}}.F$) and with $n_{\mathrm{OWF}} \geq 1$, then no adversary $\mathbf{A}$ (ε, t)-breaks $\mathit{\Pi}$'s

$$(n_S \coloneqq \max(n_{\mathrm{OWF}} - n_V, 0), n_V \coloneqq \min(n_{\mathrm{PKE}}, \max(n_{\mathrm{OWF}} - n_S, 0)), \\ q_S \coloneqq \min(q_{P\,\mathrm{NIZK}}, q_{E\,\mathrm{PKE}}), q_V \coloneqq q_{V\,\mathrm{NIZK}})\text{-}Unforgeability,$$

with $\varepsilon > (3 \cdot \varepsilon_{\mathsf{PKE\text{-}Corr}} + \varepsilon_{\mathsf{PKE\text{-}IND\text{-}CPA}}) + \varepsilon_{\mathsf{NIZK\text{-}ZK}} + \varepsilon_{\mathsf{NIZK\text{-}SS}} + 4 \cdot \varepsilon_{\mathrm{OWF}}$, with $t_{\mathrm{PKE}}, t_{\mathrm{OWF}} \approx t + t_{\mathsf{Unforg}} + q_S \cdot t_{S_P} + t_{S_G}$ and with $t_{\mathrm{NIZK}} \approx t + t_{\mathsf{Unforg}}$, where t_{Unforg} is the time to run $\mathit{\Pi}$'s $\mathbf{G}^{\mathsf{Unforg}}$ game and t_{S_P} and t_{S_G} are, respectively, the runtime of $\mathit{\Pi}_{\mathrm{NIZK}}.S_P$ and $\mathit{\Pi}_{\mathrm{NIZK}}.S_G$.

Proof. We proceed via a sequence of games [4, 26].

$\mathbf{G}^{\mathsf{Unforg}} \rightsquigarrow \mathbf{G}^1$: $\mathbf{G}^1$ is just like $\mathbf{G}^{\mathsf{Unforg}}$ except that in $\mathbf{G}^1$ the $\mathit{\Pi}_{\mathrm{PKE}}$ key pair $(\mathtt{pk}_0, \mathtt{sk}_0)$ sampled for each party B_j is assumed to be a correct one.

Note that one can reduce distinguishing these two games to breaking $\mathit{\Pi}_{\mathrm{PKE}}$'s correctness: since the reduction holds all secret keys it can handle any oracle queries. Furthermore, if an adversary $\mathbf{A}$ only queries for the verifier public keys of at most $n_V \leq n_{\mathrm{PKE}}$ parties and given the reduction only has to rely on $\mathit{\Pi}_{\mathrm{PKE}}$-$\mathcal{O}_{SK}$ oracle to generate at most one key-pair per party—namely, $(\mathtt{pk}_0, \mathtt{sk}_0)$—since from Equation B.6 no adversary $(\varepsilon_{\mathsf{PKE\text{-}Corr}})$-breaks the (n_{PKE})-Correctness of $\mathit{\Pi}_{\mathrm{PKE}}$, it follows

$$\left|\Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{\mathsf{Unforg}} = \mathtt{win}]\right| \leq \varepsilon_{\mathsf{PKE\text{-}Corr}}.$$

$\mathbf{G}^1 \rightsquigarrow \mathbf{G}^2$: This game hop is just like the previous one (i.e. $\mathbf{G}^{\mathsf{Unforg}} \rightsquigarrow \mathbf{G}^1$), the only difference being that the key-pair which is assumed to be a correct one is now $(\mathtt{pk}_1, \mathtt{sk}_1)$. Hence,

$$\left|\Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}]\right| \leq \varepsilon_{\mathsf{PKE\text{-}Corr}}.$$

$\mathbf{G}^2 \rightsquigarrow \mathbf{G}^3$**:** Game $\mathbf{G}^3$ is just like $\mathbf{G}^2$ except that in $\mathbf{G}^3$ both the $\texttt{crs}$ and the NIZK proofs in the signatures output by $\mathcal{O}_S$ are simulated (i.e. the $\texttt{crs}$ is now generated by $\Pi_{\mathrm{NIZK}}.S_G$ and the NIZK proofs are now generated by $\Pi_{\mathrm{NIZK}}.S_P$).

It is easy to see that one can reduce distinguishing these two games to breaking Π_{NIZK}'s Zero-Knowledge property, as the reduction holds all secret keys and thus can handle any oracle queries. (Although the reduction does not have the trapdoor for the $\texttt{crs}$, in case the $\texttt{crs}$ is a simulated one, since it only has to prove true statements it can rely on oracle Π_{NIZK}-$\mathcal{O}_P$ for generating the necessary NIZK proofs.) Noting the reduction only has to generate at most one NIZK proof for each $\mathcal{O}_S$ query, since $q_S \leq q_{P_{\mathrm{NIZK}}}$ it follows from Equation B.7 that no adversary $(\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{ZK}}, t_{\mathrm{NIZK}})$-breaks Π_{NIZK}'s $(q_{P_{\mathrm{NIZK}}})$-ZK security, implying

$$\left|\Pr[\mathbf{AG}^3 = \texttt{win}] - \Pr[\mathbf{AG}^2 = \texttt{win}]\right| \leq \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{ZK}}.$$

$\mathbf{G}^3 \rightsquigarrow \mathbf{G}^4$**:** The only difference between games $\mathbf{G}^3$ and $\mathbf{G}^4$ is that in game $\mathbf{G}^4$ ciphertext c_{pp} is an encryption of m followed by a 0-bitstring of appropriate length.

As before the reduction holds all secret keys and thus it can handle any oracle queries (note that the secret key corresponding to the public parameters' public key is never used by the scheme). Note that the reduction only relies on the Π_{PKE}-$\mathcal{O}_{PK}$ oracle to generate one key-pair and only queries Π_{PKE}-$\mathcal{O}_E$ at most once for each $\mathcal{O}_S$ query. Hence, as $n_{\mathrm{PKE}} \geq 1$ and $q_S \leq q_{E_{\mathrm{PKE}}}$, it follows from Equation B.6 that no adversary $(\varepsilon_{\mathrm{PKE}\text{-}\mathsf{IND}\text{-}\mathsf{CPA}}, t_{\mathrm{PKE}})$-breaks the $(n_{\mathrm{PKE}}, q_{E_{\mathrm{PKE}}})$-$\mathsf{IND}$-$\mathsf{CPA}$ security of Π_{PKE}, implying

$$\left|\Pr[\mathbf{AG}^4 = \texttt{win}] - \Pr[\mathbf{AG}^3 = \texttt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IND}\text{-}\mathsf{CPA}}.$$

$\mathbf{G}^4 \rightsquigarrow \mathbf{G}^5$**:** This step is similar to steps $\mathbf{G}^{\mathsf{Unforg}} \rightsquigarrow \mathbf{G}^1$ and $\mathbf{G}^1 \rightsquigarrow \mathbf{G}^2$, except that this time the key pair that is assumed to be a correct one is the public parameters' public key and the corresponding secret key (i.e. the key pair sampled by $\Pi.S$).

Once again one can reduce distinguishing these games to breaking Π_{PKE}'s correctness: the reduction has all secret keys so it can handle any oracle queries. In contrast to the reductions for the previous steps, however, this time the reduction only has to rely on Π_{PKE}-$\mathcal{O}_{PK}$ oracle to generate a single key-pair. Since $n_{\mathrm{PKE}} \geq 1$, it then follows from Equation B.6 that no adversary $(\varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}})$-breaks the (1)-Correctness of Π_{PKE}, implying

$$\left|\Pr[\mathbf{AG}^5 = \texttt{win}] - \Pr[\mathbf{AG}^4 = \texttt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}}.$$

$\mathbf{G}^5 \rightsquigarrow \mathbf{G}^6$**:** Game $\mathbf{G}^6$ is just like $\mathbf{G}^5$ except that now it is assumed that the OWF image y_0 in each party's public key is unique. From Lemma 1 and Equation B.8 it then follows

$$\left|\Pr[\mathbf{AG}^6 = \texttt{win}] - \Pr[\mathbf{AG}^5 = \texttt{win}]\right| \leq 2 \cdot \varepsilon_{\mathrm{OWF}}.$$

$\mathbf{G}^6 \rightsquigarrow \mathbf{G}^7$**:** Game $\mathbf{G}^7$ is just like game $\mathbf{G}^6$ except that in $\mathbf{G}^7$, for each query $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}}))$, if the NIZK proof p verifies as valid and was not output by a query $\mathcal{O}_S(A_i, \vec{V}, m)$ it is assumed that

$$(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}.$$

One can reduce distinguishing $\mathbf{G}^6$ and $\mathbf{G}^7$ to breaking the simulation soundness of Π_{NIZK}. On one hand, since the reduction holds all secret keys, it can handle any query to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{VK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{VPK}$ and $\mathcal{O}_V$. Regarding queries to $\mathcal{O}_S$, the reduction can rely on the Π_{NIZK}-$\mathcal{O}_P$ oracle to generate a simulated NIZK proof (even though the NIZK proof is for a false statement, see Definition 24). On the other hand, each query to $\mathcal{O}_V$ is handled by using the Π_{NIZK}-$\mathcal{O}_V$ oracle to verify the validity of NIZK proof p. We now argue that if there is a query $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}}))$ such that the NIZK proof p of σ verifies as valid for statement $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$, then either $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$, or σ was output by a query $\mathcal{O}_S({A_i}', \vec{V}', m')$ with $({A_i}', \vec{V}', m') = (A_i, \vec{V}, m)$: if NIZK proof p in σ was not output by $\mathcal{O}_S$ as part of a signature, then either $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$ or the reduction would win the simulation soundness game of the underlying Π_{NIZK}; if p was output as part of a signature $\sigma' = (p, \vec{c}', {c_{\mathtt{pp}}}')$ by some query $\mathcal{O}_S({A_i}', \vec{V}', m')$ such that $({A_i}', \vec{V}', m', \vec{c}', {c_{\mathtt{pp}}}') \neq (A_i, \vec{V}, m, \vec{c}, c_{\mathtt{pp}})$ then p was not generated for the same NIZK statement—in particular, note that we are assuming all parties have a distinct OWF image y_0 in their public key—implying that either $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$ or once again the reduction would win the simulation soundness game of the underlying Π_{NIZK}; it is easy to see it only remains the case where some query $\mathcal{O}_S({A_i}', \vec{V}', m')$ with $({A_i}', \vec{V}', m') = (A_i, \vec{V}, m)$ output signature σ. Note that the reduction generates at most one proof for each $\mathcal{O}_S$ query and verifies one NIZK proof for each $\mathcal{O}_V$ query. Because $q_S \leq q_{P\mathrm{NIZK}}$ and $q_V \leq q_{V\mathrm{NIZK}}$, it follows from Equation B.7 that no adversary $(\varepsilon_{\mathrm{NIZK\text{-}\mathsf{SS}}}, t_{\mathrm{NIZK}})$-breaks the $(q_{P\mathrm{NIZK}}, q_{V\mathrm{NIZK}})$-Simulation Soundness of Π_{NIZK}, and so

$$\left|\Pr[\mathbf{AG}^7 = \mathtt{win}] - \Pr[\mathbf{AG}^6 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{NIZK\text{-}\mathsf{SS}}}.$$

To conclude we now prove the following claim:

Claim. $\Pr[\mathbf{AG}^7 = \mathtt{win}] \leq 2 \cdot \varepsilon_{\mathrm{OWF}}$.

Proof. Recall that an adversary $\mathbf{A}$ can only win game $\mathbf{G}^7$ if it makes a query $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}}))$ that outputs 1, and $\mathbf{A}$ did not make any query $\mathcal{O}_S(A_i, \vec{V}, m)$, $\mathcal{O}_{SK}(A_i)$ nor any query $\mathcal{O}_{VK}(B_j)$ for $B_j \in \vec{V}$. This implies that an adversary can only win $\mathbf{G}^7$ if it forges a signature σ such that $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}}))$ outputs 1 and it did not query $\mathcal{O}_S$ on $(A_i, \vec{V}, m)$. In other words, for every query $\mathcal{O}_S({A_i}', \vec{V}', m')$, we have $(A_i, \vec{V}, m) \neq ({A_i}', \vec{V}', m')$. Note that all parties are assumed to have distinct public keys—since, as mentioned above, the OWF image y_0 in each party's public key is unique—and so for the adversary to win the game, the NIZK proof p in σ will have to verify as being valid with respect to a NIZK statement that was never proven by the $\mathcal{O}_S$ oracle. From the simulation soundness of Π_{NIZK} it then follows that $(\texttt{pp.pk}, \texttt{spk}_i, \vec{v}, m, \vec{c}, c_{\texttt{pp}}) \in L_{\text{MDVS}^{\text{adap}}}$, where $\texttt{spk}_i$ is A_i's signer public key and $\vec{v}$ is the vector of verifier public keys corresponding to vector of parties $\vec{V}$. Taking this one step further, note that $\mathcal{O}_V$ only outputs 1 if, in addition to the NIZK proof p being valid, for the least $i \in \{1, \ldots, |\vec{V}|\}$ such that $V_i = B_j$, $c_{i,b}$ is an encryption of 1 (by correctness, where b is the bit in B_j's secret key). Since by definition $R_{\text{MDVS}^{\text{adap}}} \subseteq R_{\text{MDVS}^{\text{adap}}\text{-Match}} \cap R_{\text{MDVS}^{\text{adap}}\text{-Unforg}}$, this implies $c_{\texttt{pp}}$ is an encryption of a plaintext $(m', b', ((b'_1, x'_1), \ldots, (b'_l, x'_l)))$ with x'_i being such that

$$\Pi_{\text{OWF}}.F(x'_i) \in \{\texttt{spk}.y_0, \texttt{spk}.y_1, v_i.y_0, v_i.y_1\}.$$

The correctness of $\texttt{pp.pk}$ further implies that decrypting $c_{\texttt{pp}}$ results in this plaintext. By Definition 3, since the query is a winning one, the adversary could not have queried $\mathcal{O}_{SK}$ on A_i nor $\mathcal{O}_{VK}$ on B_j. This implies that everything the adversary sees is now completely independent of A_i's bit b in its secret key and B_j's bit b in its secret key; thus, the probability that either $\Pi_{\text{OWF}}.F(x'_i) = \texttt{spk}.y_{\bar{b}}$—with $\bar{b} := 1-b$, b being the secret key in A_i's secret key—or $\Pi_{\text{OWF}}.F(x'_i) = v_i.y_{\bar{b}}$—with $\bar{b}$ this time being the complement of the bit in the secret key in B_j—is $\frac{1}{2}$. Given the correctness of $\texttt{pp.pk}$, one can then reduce winning $\mathbf{G}^7$ to breaking the security of the underlying Π_{OWF}. For each signer and each verifier, letting b be the bit in the party's secret key, the $y_{\bar{b}}$ image in the party's public key is now obtained via a query Π_{OWF}-$\mathcal{O}_Y$. Given $n_S \leq \max(n_{\text{OWF}} - n_V, 0)$ and $n_V \leq \max(n_{\text{OWF}} - n_S, 0)$, it follows by Equation B.8 that no adversary $(\varepsilon_{\text{OWF}}, t_{\text{OWF}})$-breaks the (n_{OWF})-security of Π_{OWF}, implying $\Pr[\mathbf{AG}^7 = \texttt{win}] \leq 2 \cdot \varepsilon_{\text{OWF}}$. □

B.2.4 Proof of Off-The-Record Security

Theorem 7. *If Π_{PKE} is*

$$(\varepsilon_{\text{PKE-Corr}}, \varepsilon_{\text{PKE-IND-CPA}}, \varepsilon_{\text{PKE-IK-CPA}}, t_{\text{PKE}}, n_{\text{PKE}}, q_{E\,\text{PKE}})\text{-}\textit{secure}, \tag{B.9}$$

with $n_{\text{PKE}} \geq 1$, Π_{NIZK} is

$$(\varepsilon_{\text{NIZK-Complete}}, \varepsilon_{\text{NIZK-Sound}}, \varepsilon_{\text{NIZK-ZK}}, \varepsilon_{\text{NIZK-SS}}, t_{\text{NIZK}}, q_{P\,\text{NIZK}}, q_{V\,\text{NIZK}})\text{-}\textit{secure}, \tag{B.10}$$

and Π_{OWF} is

$$(\varepsilon_{\text{OWF}}, t_{\text{OWF}}, n_{\text{OWF}})\text{-}\textit{secure}, \tag{B.11}$$

with $t_{\text{OWF}} \gtrapprox n_{\text{OWF}} \cdot (t_S + t_F)$ (where t_S and t_F are, respectively, the times to run $\Pi_{\text{OWF}}.S$ and $\Pi_{\text{OWF}}.F$) and with $n_{\text{OWF}} \geq 1$, then no adversary $\mathbf{A}$ (ε, t)-breaks Π's

$$(n_V := n_{\text{PKE}}, q_S := \min(q_{E\,\text{PKE}}, q_{P\,\text{NIZK}}), q_V := q_{V\,\text{NIZK}}, d_V := q_{E\,\text{PKE}})\text{-}\textit{Off-The-Record security},$$

with $\varepsilon > 10 \cdot \varepsilon_{\text{PKE-Corr}} + 6 \cdot \varepsilon_{\text{PKE-IND-CPA}} + 2 \cdot (\varepsilon_{\text{NIZK-ZK}} + \varepsilon_{\text{NIZK-SS}}) + 4 \cdot \varepsilon_{\text{OWF}}$*, with* $t_{\text{PKE}} \approx t + t_{\mathsf{OTR}} + q_S \cdot t_{S_P} + t_{S_G}$*, and with* $t_{\text{NIZK}} \approx t + t_{\mathsf{OTR}}$*, where* t_{OTR} *is the time to run* Π*'s* $\mathbf{G}_\beta^{\mathsf{OTR}}$ *game experiment (with* $\beta \in \{0,1\}$*), and* t_{S_P} *and* t_{S_G} *are, respectively, the runtime of* S_P *and* S_G*.*

The proof of Theorem 7 relies on an alternative signature verification procedure that is defined in Algorithm 8.

Algorithm 8 Alternative signature verification algorithm for the OTR security reductions. Below, $\mathtt{sk_{pp}}$ is the secret key corresponding to the public parameter's public key $\mathtt{pp.pk}$.

$Vfy_{\mathtt{pp}}(\mathtt{spk}, \mathtt{vpk}, \mathtt{sk}_{\mathtt{pp}}, \vec{v}, m, \sigma := (p, \vec{c}, c_{\mathtt{pp}}))$
  if $\Pi_{\text{NIZK}}.V_{\mathsf{crs}}\big((\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\text{MDVS}^{\mathsf{adap}}}, p\big) = 1$ **then**
    $(m', b', ((b'_1, x'_1), \ldots, (b'_l, x'_l))) \leftarrow \Pi_{\text{PKE}}.D_{\mathtt{sk_{pp}}}(c_{\mathtt{pp}})$
    for $i = 1, \ldots, l$ **do**
      if $\mathtt{vpk} = v_i$ **then**
        return b'_i
  return 0

Proof. As before, we proceed in a sequence of games.

For any given adversary $\mathbf{A}$, we bound $\mathbf{A}$'s advantage

$$Adv^{\mathsf{OTR}}(\mathbf{A}) := \left|\Pr[\mathbf{AG_0^{\mathsf{OTR}}} = \mathtt{win}] + \Pr[\mathbf{AG_1^{\mathsf{OTR}}} = \mathtt{win}] - 1\right|,$$

by bounding, for $\beta \in \{0,1\}$, the difference between the probability of $\mathbf{A}$ winning $\mathbf{G}_\beta^{\mathsf{OTR}}$ and winning $\mathbf{G}_\beta^1$, and, for $i \in \{1, \ldots, 12\}$, the difference between the probability of $\mathbf{A}$ winning $\mathbf{G}_\beta^i$ and winning $\mathbf{G}_\beta^{i+1}$. In other words, for $\beta \in \{0,1\}$, we bound

$$\left|\Pr[\mathbf{AG}_\beta^{\mathsf{OTR}} = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^1 = \mathtt{win}]\right|,$$

and bound, for $i \in \{1, \ldots, 12\}$,

$$\left|\Pr[\mathbf{AG}_\beta^i = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{i+1} = \mathtt{win}]\right|.$$

As we will see, games $\mathbf{G_0^{13}}$ and $\mathbf{G_1^{13}}$ are perfectly indistinguishable; it follows

$$\left|\Pr[\mathbf{AG_0^{13}} = \mathtt{win}] + \Pr[\mathbf{AG_1^{13}} = \mathtt{win}] - 1\right| = 0,$$

implying

$$\begin{aligned}
Adv^{\mathsf{OTR}}(\mathbf{A}) &\coloneqq \left|\Pr[\mathbf{AG}_{\mathbf{0}}^{\mathsf{OTR}} = \mathtt{win}] + \Pr[\mathbf{AG}_{\mathbf{1}}^{\mathsf{OTR}} = \mathtt{win}] - 1\right| \\
&\le \left|\Pr[\mathbf{AG}_{\mathbf{0}}^{\mathsf{OTR}} = \mathtt{win}] - \Pr[\mathbf{AG}_{\mathbf{0}}^{1} = \mathtt{win}]\right| \\
&+ \sum_{i=1,\ldots,12}\left|\Pr[\mathbf{AG}_{\mathbf{0}}^{i} = \mathtt{win}] - \Pr[\mathbf{AG}_{\mathbf{0}}^{i+1} = \mathtt{win}]\right| \\
&+ \sum_{i=1,\ldots,12}\left|\Pr[\mathbf{AG}_{\mathbf{1}}^{i} = \mathtt{win}] - \Pr[\mathbf{AG}_{\mathbf{1}}^{i+1} = \mathtt{win}]\right| \\
&+ \left|\Pr[\mathbf{AG}_{\mathbf{1}}^{\mathsf{OTR}} = \mathtt{win}] - \Pr[\mathbf{AG}_{\mathbf{1}}^{1} = \mathtt{win}]\right|.
\end{aligned}$$

For $\beta \in \{0,1\}$, game hops $\mathbf{G}_\beta^{\mathsf{OTR}} \rightsquigarrow \mathbf{G}_\beta^1$, $\mathbf{G}_\beta^1 \rightsquigarrow \mathbf{G}_\beta^2$, $\mathbf{G}_\beta^2 \rightsquigarrow \mathbf{G}_\beta^3$, $\mathbf{G}_\beta^3 \rightsquigarrow \mathbf{G}_\beta^4$, $\mathbf{G}_\beta^4 \rightsquigarrow \mathbf{G}_\beta^5$ and $\mathbf{G}_\beta^5 \rightsquigarrow \mathbf{G}_\beta^6$ are analogous to the ones given in the proof of Theorem 6 (see Section B.2.3), the only difference being that OTR game systems now give the adversary access to the modified $\mathcal{O}_S$ oracle instead of giving access to the original one. Nevertheless, it is trivial to adapt the reductions given in the proof of Theorem 6 to this proof. We now proceed with the remaining ones.

$\mathbf{G}_\beta^6 \rightsquigarrow \mathbf{G}_\beta^7$: Game $\mathbf{G}_\beta^7$ is just like game $\mathbf{G}_\beta^6$ except that in $\mathbf{G}_\beta^7$, for each $\mathcal{O}_V$ query where NIZK proof p in the input signature verifies as valid for the corresponding statement, it is assumed that $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$.

One can reduce distinguishing $\mathbf{G}_\beta^6$ and $\mathbf{G}_\beta^7$ to breaking the simulation soundness of Π_{NIZK}. On one hand, since the reduction holds all secret keys, it can handle any query to oracles $\mathcal{O}_{PP}$, $\mathcal{O}_{SK}$, $\mathcal{O}_{VK}$, $\mathcal{O}_{SPK}$, $\mathcal{O}_{VPK}$ and $\mathcal{O}_V$. Regarding queries to $\mathcal{O}_S$, the reduction can rely on the Π_{NIZK}-$\mathcal{O}_P$ oracle to generate a simulated NIZK proof (even though the NIZK proof is for a false statement, see Definition 24). On the other hand, each query to $\mathcal{O}_V$ is handled by using the Π_{NIZK}-$\mathcal{O}_V$ oracle to verify the validity of NIZK proof p. At this point, it only remains to show that distinguishing $\mathbf{G}_\beta^6$ and $\mathbf{G}_\beta^7$ implies the reduction would win the simulation soundness game for the underlying Π_{NIZK} scheme.

For every query $\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}}))$ where p verifies as a valid NIZK proof (for the corresponding statement), we will assume from now on that signature σ was not output by a query $\mathcal{O}_S(\mathtt{type}, A_i, \vec{V}, m, \mathcal{C})$ (as otherwise by definition the adversary does not win the game). In case p was not output as part of any signature output by $\mathcal{O}_S$ then it was not output by the underlying Π_{NIZK}-$\mathcal{O}_P$ oracle, and so, since it verifies as valid, either indeed $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$ or the adversary breaks the simulation soundness of Π_{NIZK}. In case p was output as part of a signature $\sigma' = (p, \vec{c}', c_{\mathtt{pp}}')$ generated by a query $\mathcal{O}_S(\mathtt{type}, A_i', \vec{V}', m', \mathcal{C})$, then by assumption we have $(A_i', \vec{V}', m', \vec{c}', c_{\mathtt{pp}}') \neq (A_i, \vec{V}, m, \vec{c}, c_{\mathtt{pp}})$. Given all parties have a distinct OWF image y_0 in their public key, it follows that if $(A_i', \vec{V}') \neq (A_i, \vec{V})$ then either $\mathtt{spk}_i \neq \mathtt{spk}_{i'}$ or $\vec{v} \neq \vec{v}'$. However, in this case the NIZK proof p verified as valid for a statement that is different from the one proven by $\mathcal{O}_S$—and thus also different from any statement proven by the underlying Π_{NIZK}-$\mathcal{O}_P$ oracle—and thus either $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$ or the adversary breaks the simulation soundness of Π_{NIZK}. If $(m', \vec{c}', c_{\mathtt{pp}}') \neq (m, \vec{c}, c_{\mathtt{pp}})$ then once again the NIZK proof p verified as valid for a statement that is different from the one proven by $\mathcal{O}_S$ (and thus either $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$ or the adversary breaks the simulation soundness of Π_{NIZK}).

To conclude, noting that the reduction only generates at most one NIZK proof for each query to oracle $\mathcal{O}_S$ and verifies at most one NIZK proof for each query to $\mathcal{O}_V$, it follows that since $q_S \leq q_{P\,\mathrm{NIZK}}$ and $q_V \leq q_{V\,\mathrm{NIZK}}$, by Equation B.10 no adversary $(\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{SS}}, t_{\mathrm{NIZK}})$-breaks Π_{NIZK}'s $(q_{P\,\mathrm{NIZK}}, q_{V\,\mathrm{NIZK}})$-Simulation Soundness, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^{6}_{\beta} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{7}_{\beta} = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{SS}}.$$

$\mathbf{G}^{7}_{\beta} \rightsquigarrow \mathbf{G}^{8}_{\beta}$: While in game $\mathbf{G}^{7}_{\beta}$ queries to oracle $\mathcal{O}_V$ are handled by following the normal signature verification procedure, in game $\mathbf{G}^{8}_{\beta}$ these queries are handled by following the alternative signature verification procedure given in Algorithm 8.

Since at this point we are assuming the perfect correctness of all Π_{PKE} key-pairs sampled by the game—which includes the public parameters public key (and corresponding secret key) as well as the two Π_{PKE} pairs sampled for each party—and furthermore are assuming that if a NIZK proof p verifies as valid for a statement $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}})$ then it must indeed be the case that $(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathtt{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}$, it follows that $\mathbf{G}^{7}_{\beta}$ and $\mathbf{G}^{8}_{\beta}$ are perfectly indistinguishable:

$$\left|\Pr[\mathbf{A}\mathbf{G}^{7}_{\beta} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{8}_{\beta} = \mathtt{win}]\right| = 0.$$

$\mathbf{G}^{8}_{\beta} \rightsquigarrow \mathbf{G}^{9}_{\beta}$: $\mathbf{G}^{9}_{\beta}$ is just like $\mathbf{G}^{8}_{\beta}$ except that in $\mathbf{G}^{9}_{\beta}$ the Π_{PKE} key pair $(\mathtt{pk}_0, \mathtt{sk}_0)$ sampled for each party B_j is assumed to be sampled according to the original Π_{PKE}'s key-pair distribution.

Note that one can reduce distinguishing these two games to breaking Π_{PKE}'s correctness: since the reduction holds all secret keys it can handle any oracle queries. If an adversary $\mathbf{A}$ only queries for the verifier public keys of at most $n_V \leq n_{\mathrm{PKE}}$ parties, and given the reduction only has to rely on Π_{PKE}-$\mathcal{O}_{SK}$ oracle to generate at most one key-pair per party—namely, $(\mathtt{pk}_0, \mathtt{sk}_0)$—since from Equation B.9 no adversary $(\varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}})$-breaks the (n_{PKE})-Correctness of Π_{PKE}, it follows

$$\left|\Pr[\mathbf{A}\mathbf{G}^{8}_{\beta} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{9}_{\beta} = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}}.$$

$\mathbf{G}^{9}_{\beta} \rightsquigarrow \mathbf{G}^{10}_{\beta}$: This game hop is just like the previous one (i.e. $\mathbf{G}^{8}_{\beta} \rightsquigarrow \mathbf{G}^{9}_{\beta}$), the only difference being that the key-pair which is assumed to be sampled according to the original Π_{PKE}'s key-pair distribution is now $(\mathtt{pk}_1, \mathtt{sk}_1)$. Hence,

$$\left|\Pr[\mathbf{A}\mathbf{G}^{9}_{\beta} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{10}_{\beta} = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}}.$$

$\mathbf{G}_\beta^{10} \rightsquigarrow \mathbf{G}_\beta^{11}$**:** The only difference between games $\mathbf{G}_\beta^{10}$ and $\mathbf{G}_\beta^{11}$ is that in $\mathbf{G}_\beta^{11}$ each signature $\sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}})$ output by a query $\mathcal{O}_S(\mathtt{type}, A_i, \vec{V}, m, \mathcal{C})$ is such that for all $i \in \{1, \ldots, |\vec{V}|\}$, ciphertext $c_{i,\bar{b}}$ in the vector of ciphertexts $\vec{c}$—where $\bar{b} \coloneqq 1 - b$, b being the secret bit of party V_i—is an encryption of bit 0.

Once again, one can reduce distinguishing the two games to breaking the IND-CPA security of the underlying scheme Π_{PKE}: since the reduction holds exactly the same secret information as it did in the last game, it can handle all queries as before. Furthermore, as for each verifier the reduction only has to rely on Π_{PKE}-$\mathcal{O}_{PK}$ to generate one public key and for each query $\mathcal{O}_S(\mathtt{type}, A_i, \vec{V}, m, \mathcal{C})$ the reduction queries Π_{PKE}-$\mathcal{O}_E$ at most $|\vec{V}|$ times, given $n_V \leq n_{\mathrm{PKE}}$ and $d_V \leq q_{E_{\mathrm{PKE}}}$, it follows by Equation B.9 that no adversary $(\varepsilon_{\text{PKE-IND-CPA}}, t_{\mathrm{PKE}})$-breaks Π_{PKE}'s $(n_{\mathrm{PKE}}, q_{E_{\mathrm{PKE}}})$-IND-CPA security, implying

$$\left|\Pr[\mathbf{AG}_\beta^{10} = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{11} = \mathtt{win}]\right| \leq \varepsilon_{\text{PKE-IND-CPA}}.$$

$\mathbf{G}_\beta^{11} \rightsquigarrow \mathbf{G}_\beta^{12}$**:** The difference between games $\mathbf{G}_\beta^{11}$ and $\mathbf{G}_\beta^{12}$ is that in $\mathbf{G}_\beta^{12}$ each signature $\sigma \coloneqq (p, \vec{c}, c_{\mathtt{pp}})$ output by a query $\mathcal{O}_S(\mathtt{type}, A_i, \vec{V}, m, \mathcal{C})$ is such that for all $i \in \{1, \ldots, |\vec{V}|\}$, ciphertext $c_{i,\bar{b}}$ in the vector of ciphertexts $\vec{c}$ returns to being an encryption of the same bit as it was in $\mathbf{G}_\beta^{11}$, whereas $c_{i,b}$ becomes an encryption of bit 0.

Note that for any party B_j the existence of a query $\mathcal{O}_S(\mathtt{type}, A_i, \vec{V}, m, \mathcal{C})$ with $B_j \in \vec{V}$ implies there is no query to $\mathcal{O}_{VK}$ on B_j (and vice-versa). Thus all the adversary sees is independent of B_j's secret key bit. It then follows that $\mathbf{G}_\beta^{11}$ is perfectly indistinguishable from $\mathbf{G}_\beta^{12}$, and hence

$$\left|\Pr[\mathbf{AG}_\beta^{11} = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{12} = \mathtt{win}]\right| = 0.$$

$\mathbf{G}_\beta^{12} \rightsquigarrow \mathbf{G}_\beta^{13}$**:** This step is analogous to step $\mathbf{G}_\beta^{10} \rightsquigarrow \mathbf{G}_\beta^{11}$, except that this time $c_{i,b}$ is an encryption of bit 0. It follows

$$\left|\Pr[\mathbf{AG}_\beta^{12} = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{13} = \mathtt{win}]\right| \leq \varepsilon_{\text{PKE-IND-CPA}}.$$

To conclude the proof, note that $\mathbf{G}_\mathbf{0}^{13}$ is perfectly indistinguishable from $\mathbf{G}_\mathbf{1}^{13}$, as everything an adversary sees when interacting with either game is exactly the same (independently of which game the adversary is actually interacting with). Also, note that each intermediate game simply has to emulate the original game towards $\mathbf{A}$—with a few tweaks that, apart from the generation of a simulated $\mathtt{crs}$ and the generation of simulated NIZK proofs, do not affect the time for emulating the game. Letting t_{OTR} be the time to run Π's $\mathbf{G}_\beta^{\mathsf{OTR}}$ game experiment (with $\beta \in \{0, 1\}$), t_{S_P} be the runtime of S_P and t_{S_G} be the runtime of S_G, it follows

$$\begin{aligned} t_{\mathrm{PKE}} &\approx t + t_{\mathsf{OTR}} + q_S \cdot t_{S_P} + t_{S_G}, \\ t_{\mathrm{NIZK}} &\approx t + t_{\mathsf{OTR}}. \end{aligned}$$

□

B.2.5 Proof of Message-Bound Validity

Theorem 8. *If Π_{PKE} is*

$$(\varepsilon_{\mathrm{PKE\text{-}Corr}}, \varepsilon_{\mathrm{PKE\text{-}IND\text{-}CPA}}, \varepsilon_{\mathrm{PKE\text{-}IK\text{-}CPA}}, \\ t_{\mathrm{PKE}}, n_{\mathrm{PKE}}, q_{E\,\mathrm{PKE}})\text{-secure,} \tag{B.12}$$

with $n_{\mathrm{PKE}} \geq 1$, and Π_{NIZK} is

$$(\varepsilon_{\mathrm{NIZK\text{-}Complete}}, \varepsilon_{\mathrm{NIZK\text{-}Sound}}, \varepsilon_{\mathrm{NIZK\text{-}ZK}}, \varepsilon_{\mathrm{NIZK\text{-}SS}}, \\ t_{\mathrm{NIZK}}, q_{P\,\mathrm{NIZK}}, q_{V\,\mathrm{NIZK}})\text{-secure,} \tag{B.13}$$

then no adversary $\mathbf{A}$ (ε, t)-breaks Π's

$$(q_V := q_{V\,\mathrm{NIZK}})\text{-Message-Bound Validity,}$$

with $\varepsilon > \varepsilon_{\mathrm{PKE\text{-}Corr}} + \varepsilon_{\mathrm{NIZK\text{-}Sound}}$ and with $t_{\mathrm{NIZK}} \approx t + t_{\mathsf{Bound\text{-}Val}}$, where $t_{\mathsf{Bound\text{-}Val}}$ is the time to run Π's $\mathbf{G}^{\mathsf{Bound\text{-}Val}}$ game.

Proof. We proceed in a sequence of games.

$\mathbf{G}^{\mathsf{Bound\text{-}Val}} \rightsquigarrow \mathbf{G}^1$: $\mathbf{G}^1$ is just like the original game $\mathbf{G}^{\mathsf{Bound\text{-}Val}}$, except that in $\mathbf{G}^1$ the key-pair consisting of the public parameters' public key and the corresponding secret key (i.e. the key-pair sampled by $\Pi.S$) is assumed to be a correct one. One can reduce distinguishing $\mathbf{G}^{\mathsf{Bound\text{-}Val}}$ and $\mathbf{G}^1$ to breaking Π_{PKE}'s correctness: the reduction has all secret keys so it can handle any oracle queries. Since $n_{\mathrm{PKE}} \geq 1$, it then follows from Equation B.12

$$\left|\Pr[\mathbf{A}\mathbf{G}^{\mathsf{Bound\text{-}Val}} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE\text{-}Corr}}.$$

$\mathbf{G}^1 \rightsquigarrow \mathbf{G}^2$: $\mathbf{G}^2$ is just like $\mathbf{G}^1$, except that in $\mathbf{G}^2$ for each query

$$\mathcal{O}_V(A_i, B_j, \vec{V}, m, \sigma := (p, \vec{c}, c_{\mathsf{pp}})),$$

if the NIZK proof p verifies as valid then it is assumed that

$$(\mathtt{pp.pk}, \mathtt{spk}, \vec{v}, m, \vec{c}, c_{\mathsf{pp}}) \in L_{\mathrm{MDVS}^{\mathsf{adap}}}.$$

One can reduce distinguishing these two games to breaking Π_{NIZK}'s soundness because a reduction holds all secret keys and thus can handle any oracle queries. Since each $\mathcal{O}_V$ query only requires the reduction to make one NIZK proof verification query and because the adversary only makes up to $q_V \leq q_{V\,\mathrm{NIZK}}$ signature verification queries, by Equation B.13 it follows

$$\left|\Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{NIZK\text{-}Sound}}.$$

To finish the proof we now prove the following claim:

Claim. For any adversary $\mathbf{A}$

$$\Pr[\mathbf{A}\mathbf{G}^2 = \texttt{win}] = 0.$$

Proof. An adversary can only win $\mathbf{G}^2$ if it makes a query

$$\mathcal{O}_V(A_i, B_j, \vec{V}, m', \sigma \coloneqq (p, \vec{c}, c_{\mathsf{pp}}))$$

that outputs 1 and σ was output by a query $\mathcal{O}_S(A_i, \vec{V}, m)$ satisfying $B_j \in \vec{V}$ and $m' \neq m$. By the definition of Π's *Sig* algorithm (Algorithm 3), c_{pp} is an encryption of a tuple $(m, 1, \vec{\alpha})$. The correctness of the public parameters public key (and corresponding secret key) then implies that there is no sequence of random coins r' such that c_{pp} is an encryption of a (different) tuple $(m', b', \vec{\alpha}')$ (since $m' \neq m$ the tuples are different). But then this implies

$$(\texttt{pp.pk}, \texttt{spk}, \vec{v}, m', \vec{c}, c_{\mathsf{pp}}) \notin L_{\mathrm{MDVS}^{\mathsf{adap}}},$$

and because signature verification oracles in $\mathbf{G}^2$ are assumed to output 0 in such case[10], it follows that no adversary can win $\mathbf{G}^2$. □

B.3 PKEBC Construction Security Proofs

In this section we give the formal security theorems and corresponding full proofs for the PKEBC construction given in Section 5.4.

B.3.1 Proof of Correctness

Theorem 9. *If Π_{PKE} is*

$$\begin{aligned}(\varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}, \varepsilon_{\mathrm{PKE\text{-}\mathsf{IND\text{-}CPA}}}, \varepsilon_{\mathrm{PKE\text{-}\mathsf{IK\text{-}CPA}}},\\ t_{\mathrm{PKE}}, n_{\mathrm{PKE}}, q_{E_{\mathrm{PKE}}}, \mathsf{Corr})\text{-secure,}\end{aligned} \tag{B.14}$$

with $t_{\mathrm{PKE}} \gtrapprox n_{\mathrm{PKE}} \cdot t_G + t_D$ (where t_G and t_D are, respectively, the times to run $\Pi_{\mathrm{PKE}}.G$ and $\Pi_{\mathrm{PKE}}.D$) and with $n_{\mathrm{PKE}} \geq 1$, Π_{NIZK} is

$$\begin{aligned}(\varepsilon_{\mathrm{NIZK\text{-}\mathsf{Complete}}}, \varepsilon_{\mathrm{NIZK\text{-}\mathsf{Sound}}}, \varepsilon_{\mathrm{NIZK\text{-}\mathsf{ZK}}}, \varepsilon_{\mathrm{NIZK\text{-}\mathsf{SS}}},\\ t_{\mathrm{NIZK}}, q_{P_{\mathrm{NIZK}}}, q_{V_{\mathrm{NIZK}}})\text{-secure,}\end{aligned} \tag{B.15}$$

and Π_{SKE} is

$$(\varepsilon_{\mathrm{SKE\text{-}\mathsf{1\text{-}IND\text{-}CPA}}}, t_{\mathrm{SKE}}, q_{E_{\mathrm{SKE}}}, \mathsf{Corr})\text{-secure,} \tag{B.16}$$

then no adversary $\mathbf{A}$ (ε, t)-breaks Π's

$$(n \coloneqq n_{\mathrm{PKE}}, q_E \coloneqq q_{P_{\mathrm{NIZK}}}, q_D \coloneqq q_{V_{\mathrm{NIZK}}})\text{-Correctness,}$$

with $\varepsilon > \varepsilon_{\mathrm{NIZK\text{-}\mathsf{Complete}}} + 2 \cdot \varepsilon_{\mathrm{PKE\text{-}\mathsf{IND\text{-}CPA}}} + 3 \cdot \varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}$, with $t_{\mathrm{NIZK}} \approx t + t_{\mathsf{Corr}}$— where t_{Corr} is the time to run Π's $\mathbf{G}^{\mathsf{Corr}}$ game—and with $t < t_{\mathrm{PKE}}$.

Proof. We proceed in a sequence of games.

[10] See the definition of Π's *Vfy* algorithm: Algorithm 3.

$\mathbf{G}^{\mathsf{Corr}} \rightsquigarrow \mathbf{G}^1$**:** Game $\mathbf{G}^1$ is just like the original game $\mathbf{G}^{\mathsf{Corr}}$, except that in $\mathbf{G}^1$ for each ciphertext $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ output by a query $\mathcal{O}_E(\vec{V}, m)$, if $\mathcal{O}_D$ is queried on input (B_j, c) it no longer verifies p's validity and simply proceeds as if p would verify as being valid.

Games $\mathbf{G}^{\mathsf{Corr}}$ and $\mathbf{G}^1$ are perfectly indistinguishable unless there is a query $\mathcal{O}_D(B_j, c)$ where c was output by some query $\mathcal{O}_E(\vec{V}, m)$ such that $B_j \in \vec{V}$, and the verification of the NIZK proof p in c fails. One can then reduce distinguishing these games to breaking Π_{NIZK}'s completeness: the reduction holds the secret keys of every party, and so it can trivially handle any oracle queries. Noting the reduction makes at most one Π_{NIZK}-$\mathcal{O}_P$ query for each $\mathcal{O}_S$ query and at most one Π_{NIZK}-$\mathcal{O}_V$ query for each $\mathcal{O}_V$ query, as $\mathbf{A}$ only makes up to $q_E \leq q_{P\,\text{NIZK}}$ queries to $\mathcal{O}_E$ and $q_D \leq q_{V\,\text{NIZK}}$ queries to $\mathcal{O}_D$, it follows from Equation B.15, that no adversary $(\varepsilon_{\text{NIZK-}\mathsf{Complete}}, t_{\text{NIZK}})$-breaks the $(q_{P\,\text{NIZK}}, q_{V\,\text{NIZK}})$-Completeness of Π_{NIZK}, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{\mathsf{Corr}} = \mathtt{win}]\right| \leq \varepsilon_{\text{NIZK-}\mathsf{Complete}}.$$

$\mathbf{G}^1 \rightsquigarrow \mathbf{G}^2$**:** $\mathbf{G}^2$ is just like $\mathbf{G}^1$, except that now there are no two parties with the same public key. It follows from Lemma 2 and Equation B.14 that

$$\left|\Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}]\right| \leq 2 \cdot \varepsilon_{\text{PKE-}\mathsf{IND\text{-}CPA}} + \varepsilon_{\text{PKE-}\mathsf{Corr}}.$$

$\mathbf{G}^2 \rightsquigarrow \mathbf{G}^3$**:** $\mathbf{G}^3$ is just like $\mathbf{G}^2$, except that now $\mathcal{O}_D$ behaves differently. For each query (B_j, c) to $\mathcal{O}_D$, where $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ is the output of a query $\mathcal{O}_E(\vec{V}, m)$, $\mathcal{O}_D$ now skips decryption attempts for every index $i \in \{1, \ldots, |\vec{V}|\}$ such that $V_i \neq B_j$.

Games $\mathbf{G}^3$ and $\mathbf{G}^2$ are perfectly indistinguishable unless there is a decryption query $\mathcal{O}_D(B_j, c)$ where $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ was the output of a query $\mathcal{O}_E(\vec{V}, m)$ and for some $i \in \{1, \ldots, |\vec{c}|\}$, ciphertext $c_{i,0} \in \vec{c}$ is an encryption of two different values under $\mathtt{pp.pk}$—one being $\mathtt{pk}_j$ and the other being the public key $\mathtt{pk}_i$ of party $V_i \in \vec{V}$. Note that one can reduce distinguishing these two games to breaking Π_{PKE}'s correctness. More concretely, the reduction only has to rely on the underlying oracle Π_{PKE}-$\mathcal{O}_{PK}$ to generate a single public key ($\mathtt{pp.pk}$), but otherwise can handle any oracle queries since it holds all necessary secret keys. Putting things together, since by Equation B.14 no adversary $(\varepsilon_{\text{PKE-}\mathsf{Corr}})$-breaks the (1)-Correctness of Π_{PKE}, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^3 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^2 = \mathtt{win}]\right| \leq \varepsilon_{\text{PKE-}\mathsf{Corr}}.$$

$\mathbf{G}^3 \rightsquigarrow \mathbf{G}^4$**:** Game $\mathbf{G}^4$ is just like $\mathbf{G}^3$ except that once again $\mathcal{O}_D$ behaves differently. Again, let $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ be the output of $\mathcal{O}_E$ when queried on some input $(\vec{V}, m)$. If $\mathcal{O}_D$ is queried on input (B_j, c) such that $B_j \in \vec{V}$

and letting $i \in \{1, \ldots, |\vec{V}|\}$ be the least index such that $V_i = B_j$ and letting b be the bit in B_j's secret key, $\mathcal{O}_D$ no longer follows the procedure of trying to decrypt $c_{i,b,1}$, reconstructing $c_{i,0}$ and then decrypting $c_{i,b,2}$ to obtain $k_{\mathtt{sym}}$—the Π_{SKE}'s symmetric key that was generated in the $\mathcal{O}_E$ query. Instead, $\mathcal{O}_D$ simply proceeds as if this check (i.e. reconstructing $c_{i,0}$) succeeded for index i, and $c_{i,b,2}$'s decryption resulted in $k_{\mathtt{sym}}$.

It is easy to see one can reduce distinguishing these two games to breaking the correctness of the underlying PKE scheme, similarly to the previous game hop (i.e. $\mathbf{G}^2 \rightsquigarrow \mathbf{G}^3$). The main difference is that now the reduction relies on Π_{PKE}-$\mathcal{O}_{SK}$ to generate a key-pair for each party: for each party B_j, the reduction uses Π_{PKE}-$\mathcal{O}_{SK}$ to generate key-pair $(\mathtt{pk}_b, \mathtt{sk}_b)$, where b is the bit in B_j's secret key. As before, the reduction has access to all the secret keys, and thus it can handle any oracle queries. Since $\mathbf{A}$ queries on at most $n \leq n_{\mathrm{PKE}}$ different parties, it follows from Equation B.14 that no adversary ($\varepsilon_{\mathsf{PKE\text{-}Corr}}$)-breaks the ($n_{\mathrm{PKE}}$)-Correctness property of Π_{PKE}, implying

$$\left|\Pr[\mathbf{AG}^4 = \mathtt{win}] - \Pr[\mathbf{AG}^3 = \mathtt{win}]\right| \leq \varepsilon_{\mathsf{PKE\text{-}Corr}}.$$

$\mathbf{G}^4 \rightsquigarrow \mathbf{G}^5$: $\mathbf{G}^5$ is just like $\mathbf{G}^4$ except that again $\mathcal{O}_D$ behaves differently. Let $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ be the output of $\mathcal{O}_E$ when queried on some input $(\vec{V}, m)$. If $\mathcal{O}_D$ is queried on input (B_j, c) such that $B_j \in \vec{V}$ and letting $i \in \{1, \ldots, |\vec{V}|\}$ be the least index such that $V_i = B_j$, $\mathcal{O}_D$ no longer tries decrypting $c_{\mathtt{sym}}$ using $k_{\mathtt{sym}}$, and instead simply proceeds as if the decryption had output the $(\vec{v}, m)$ pair that was encrypted by the $\mathcal{O}_E$ query.

As before, one can reduce distinguishing the two games to winning the correctness game of Π_{SKE}. It follows from Equation B.16 that Π_{SKE} is perfectly correct, which implies

$$\left|\Pr[\mathbf{AG}^5 = \mathtt{win}] - \Pr[\mathbf{AG}^4 = \mathtt{win}]\right| = 0.$$

Finally, noting that in $\mathbf{G}^5$ any query $\mathcal{O}_D(B_j, c)$—where c was output by a query $\mathcal{O}_E(\vec{V}, m)$ with $B_j \in \vec{V}$—must output $(\vec{v}, m)$—$\vec{v}$ being the vector of public keys corresponding to the vector of parties $\vec{V}$—it follows

$$\Pr[\mathbf{AG}^5 = \mathtt{win}] = 0.$$

□

B.3.2 Proof of Robustness

Theorem 10. *If Π_{PKE} is*

$$(\varepsilon_{\mathsf{PKE\text{-}Corr}}, \varepsilon_{\mathsf{PKE\text{-}IND\text{-}CPA}}, \varepsilon_{\mathsf{PKE\text{-}IK\text{-}CPA}}, t_{\mathrm{PKE}}, n_{\mathrm{PKE}}, q_{E\mathrm{PKE}})\text{-}secure, \tag{B.17}$$

with $t_{\mathrm{PKE}} \gtrapprox n_{\mathrm{PKE}} \cdot t_G + t_D$ (where t_G and t_D are, respectively, the times to run $\Pi_{\mathrm{PKE}}.G$ and $\Pi_{\mathrm{PKE}}.D$) and $n_{\mathrm{PKE}} \geq 1$, then no adversary $\mathbf{A}$ (ε, t)-breaks Π's $(n \coloneqq n_{E\mathrm{PKE}})$-Robustness, with $\varepsilon > 2 \cdot \varepsilon_{\mathsf{PKE\text{-}IND\text{-}CPA}} + 2 \cdot \varepsilon_{\mathsf{PKE\text{-}Corr}}$ and $t < t_{\mathrm{PKE}}$.

Proof. This result can be proven by following (some of) the arguments given in the proof of Theorem 9 (see Section B.3.1). More concretely, one would first hop from the original $\mathbf{G}^{\mathsf{Rob}}$ Robustness game to one where all parties are assumed to have distinct public keys (see $\mathbf{G}^2$), and then to one where decryption queries for ciphertexts not meant for the decrypting party would simply output $\bot$ (see $\mathbf{G}^3$). □

B.3.3 Proof of Consistency

Theorem 11. *If Π_{PKE} is*

$$(\varepsilon_{\text{PKE-}\mathsf{Corr}}, \varepsilon_{\text{PKE-}\mathsf{IND\text{-}CPA}}, \varepsilon_{\text{PKE-}\mathsf{IK\text{-}CPA}}, \\ t_{\mathrm{PKE}}, n_{\mathrm{PKE}}, q_{E\,\mathrm{PKE}})\text{-secure,} \tag{B.18}$$

with $n_{\mathrm{PKE}} \geq 1$, Π_{NIZK} is

$$(\varepsilon_{\text{NIZK-}\mathsf{Complete}}, \varepsilon_{\text{NIZK-}\mathsf{Sound}}, \varepsilon_{\text{NIZK-}\mathsf{ZK}}, \varepsilon_{\text{NIZK-}\mathsf{SS}}, \\ t_{\mathrm{NIZK}}, q_{P\,\mathrm{NIZK}}, q_{V\,\mathrm{NIZK}})\text{-secure,} \tag{B.19}$$

Π_{SKE} is

$$(\varepsilon_{\text{SKE-}\mathsf{1\text{-}IND\text{-}CPA}}, t_{\mathrm{SKE}}, q_{E\,\mathrm{SKE}})\text{-secure,} \tag{B.20}$$

and $\Pi_{\mathrm{NIZK}}.V$ is a deterministic algorithm, then no adversary $\mathbf{A}$ (ε, t)-breaks Π's

$$(n \coloneqq n_{\mathrm{PKE}}, q_D \coloneqq q_{V\,\mathrm{NIZK}})\text{-Consistency,}$$

with $\varepsilon > \varepsilon_{\text{NIZK-}\mathsf{Sound}} + 3 \cdot \varepsilon_{\text{PKE-}\mathsf{Corr}}$ and with $t_{\mathrm{NIZK}} \approx t + t_{\mathsf{Cons}}$, where t_{Cons} is the time to run Π's $\mathbf{G}^{\mathsf{Cons}}$ game.

Proof. We prove this result via game hopping.

$\mathbf{G}^{\mathsf{Cons}} \rightsquigarrow \mathbf{G}^1$: Game $\mathbf{G}^1$ is just like the original game $\mathbf{G}^{\mathsf{Cons}}$, except that whenever $\mathcal{O}_D$ is queried on an input (B_j, c), with $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ such that $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \notin L_{\mathrm{PKEBC}^{\mathsf{adap}}}$, the oracle outputs $\bot$.

It is easy to see that $\mathbf{G}^1$ is perfectly indistinguishable from $\mathbf{G}^{\mathsf{Cons}}$ unless $\mathbf{A}$ makes a decryption query on a ciphertext $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ such that the NIZK proof p verifies as being valid but the statement is not true (i.e. $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \notin L_{\mathrm{PKEBC}^{\mathsf{adap}}}$). One can then reduce distinguishing these two games to breaking the soundness of Π_{NIZK}, as the reduction holds the secrets of all parties, and thus it can handle any oracle queries. Since the reduction only has verify the validity of a NIZK proof for each query the adversary makes to $\mathcal{O}_D$—which it does using the Π_{NIZK}-$\mathcal{O}_V$ oracle of the underlying security game—and since the adversary can only make up to $q_D \leq q_{V\,\mathrm{NIZK}}$ decryption queries, it follows from Equation B.19 that no adversary $(\varepsilon_{\text{NIZK-}\mathsf{Sound}}, t_{\mathrm{NIZK}})$-breaks the $(q_{V\,\mathrm{NIZK}})$-Soundness of Π_{NIZK}, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}^1 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{\mathsf{Cons}} = \mathtt{win}]\right| \leq \varepsilon_{\text{NIZK-}\mathsf{Sound}}.$$

$\mathbf{G}^1 \rightsquigarrow \mathbf{G}^2$**:** The only difference between $\mathbf{G}^2$ and $\mathbf{G}^1$ is that in $\mathbf{G}^2$ the public key of the public parameters ($\texttt{pp.pk}$) and the corresponding secret key $\texttt{sk}_{\texttt{pp}}$ are assumed to be correct.

It is easy to see that one can reduce distinguishing the two games to breaking the correctness of Π_{PKE}: since the reduction holds all secret keys, it can handle any oracle queries. Noting that the reduction only queries the underlying game for the public key of a single party (which it does via the Π_{PKE}-$\mathcal{O}_{PK}$ oracle), it follows from Equation B.18 that no adversary ($\varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}$)-breaks the (1)-Correctness of Π_{PKE}, implying

$$\left|\Pr[\mathbf{AG}^2 = \texttt{win}] - \Pr[\mathbf{AG}^1 = \texttt{win}]\right| \leq \varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}.$$

$\mathbf{G}^2 \rightsquigarrow \mathbf{G}^3$**:** Game $\mathbf{G}^3$ only differs from $\mathbf{G}^2$ in that the key pair $(\texttt{pk}_0, \texttt{sk}_0)$ of each party B_j is assumed to be correct.

Similarly to the previous step, one can reduce distinguishing the two game systems to winning the correctness game of the underlying Π_{PKE}. Given the reduction queries for at most one key for each party, since $n \leq n_{\mathrm{PKE}}$ it follows from Equation B.18 that no adversary ($\varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}$)-breaks the ($n_{\mathrm{PKE}}$)-Correctness of Π_{PKE}, implying

$$\left|\Pr[\mathbf{AG}^3 = \texttt{win}] - \Pr[\mathbf{AG}^2 = \texttt{win}]\right| \leq \varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}.$$

$\mathbf{G}^3 \rightsquigarrow \mathbf{G}^4$**:** This game hop is analogous to the previous one ($\mathbf{G}^2 \rightsquigarrow \mathbf{G}^3$) except that now the key-pairs that are assumed to be correct are each party's $(\texttt{pk}_1, \texttt{sk}_1)$ key-pair.

$\mathbf{G}^4 \rightsquigarrow \mathbf{G}^5$**:** $\mathbf{G}^5$ differs from $\mathbf{G}^4$ in that in $\mathbf{G}^5$ it is assumed that Π_{SKE} is perfectly correct. It then follows from Equation B.20 that

$$\left|\Pr[\mathbf{AG}^5 = \texttt{win}] - \Pr[\mathbf{AG}^4 = \texttt{win}]\right| = 0.$$

To conclude this proof it only remains to prove the following claim:

Claim. For any adversary $\mathbf{A}$, $\Pr[\mathbf{AG}^5 = \texttt{win}] = 0$.

Proof. $\mathbf{A}$ wins $\mathbf{G}^5$ if it queries $\mathcal{O}_D$ on inputs (B_i, c) and (B_j, c) for some B_i and B_j and some ciphertext c, and the first query outputs $(\vec{v}, m) \neq \perp$ with $\texttt{pk}_j \in \vec{v}$ (where $\texttt{pk}_j$ is B_j's public key) whereas the second outputs either $\perp$ or some $(\vec{v}', m')$ with $(\vec{v}', m') \neq (\vec{v}, m)$. Consider any two queries $q_{D,i}$ and $q_{D,j}$ that $\mathbf{A}$ makes to $\mathcal{O}_D$ on inputs (B_i, c) and (B_j, c), respectively, such that $q_{D,i}$ outputs $(\vec{v}, m)$ with $(\vec{v}, m) \neq \perp$ and $\texttt{pk}_j \in \vec{v}$. (If $\mathbf{A}$ does not make any two queries satisfying these conditions, it does not win $\mathbf{G}^5$.) In the following, let $c \coloneqq (p, c_{\texttt{pp}}, \vec{c}, c_{\texttt{sym}})$ be the input ciphertext of both $q_{D,i}$ and $q_{D,j}$.

To prove this claim we will first show that since $q_{D,i}$ outputs a pair $(\vec{v}, m) \neq \perp$ then $c_{\texttt{pp}}$ must be an encryption of m under $\texttt{pp.pk}$—i.e. for some sequence of random coins $r_{\mathtt{pp}}$ we have $c_{\mathtt{pp}} = \Pi_{\text{PKE}}.E(\mathtt{pp.pk}, m; r_{\mathtt{pp}})$—and for $l \in \{1, \ldots, |\vec{v}|\}$ each ciphertext $c_{l,0}$ is an encryption of v_l under pp.pk—i.e. for some sequence of random coins $r_{l,0}$, we have $c_{l,0} = \Pi_{\text{PKE}}.E(\mathtt{pp.pk}, v_l; r_{l,0})$. Then, we will show that if $q_{D,j}$ outputs some pair $(\vec{v}', m') \neq \bot$, then it must be the case that $(\vec{v}', m') = (\vec{v}, m)$. Finally, we will show that $q_{D,j}$ does not output $\bot$, implying that it must output the same pair $(\vec{v}, m)$ that was output by $q_{D,i}$ (and so the adversary cannot win the game).

First, since query $q_{D,i}$ does not output $\bot$, NIZK proof p verified as being valid; the soundness of Π_{NIZK} implies $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{adap}}}$. By the definition of the decryption algorithm, it follows that for some $l \in \{1, \ldots, |\vec{c}|\}$, the oracle decrypted $c_{l,b,1}$ obtaining a sequence of random coins $r_{l,0}$ such that $c_{l,0} = \Pi_{\text{PKE}}.E(\mathtt{pp.pk}, \mathtt{pk}_i; r_{l,0})$. By the correctness of pp.pk, there is no sequence of random coins r such that $c_{l,0} = \Pi_{\text{PKE}}.E(\mathtt{pp.pk}, \mathtt{pk}'; r)$, for any $\mathtt{pk}' \neq \mathtt{pk}_i$. Since $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{adap}}}$, it then follows there are sequences of random coins $r_{l,b,2}$ and $r_{\mathtt{sym}}$ such that $c_{l,b,2} = \Pi_{\text{PKE}}.E(\mathtt{pk}_i, \Pi_{\text{SKE}}.G(1^k, r_{\mathtt{sym}}); r_{l,b,2})$. In the following, let $k_{\mathtt{sym}} = \Pi_{\text{SKE}}.G(1^k, r_{\mathtt{sym}})$. By the correctness of each party's $(\mathtt{pk}_b, \mathtt{sk}_b)$ key-pair (where b is the bit in the party's secret key) the decryption of $c_{l,b,2}$ outputs $k_{\mathtt{sym}}$. By the definition of the decryption algorithm, since $q_{D,i}$ outputs $(\vec{v}, m)$, then the decryption of $c_{\mathtt{sym}}$ resulted in this same pair $(\vec{v}, m)$. Again since $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{adap}}}$, and by the correctness of Π_{SKE} it follows that there is a sequence of random coins $r_{\mathtt{sym}}'$ such that $c_{\mathtt{sym}} = \Pi_{\text{SKE}}.E(k_{\mathtt{sym}}, (\vec{v}, m); r_{\mathtt{sym}}')$. To conclude the first step of the claim's proof, once again since $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{adap}}}$, $c_{\mathtt{pp}}$ is an encryption of m under pp.pk and for $k \in \{1, \ldots, |\vec{v}|\}$ each ciphertext $c_{k,0}$ is an encryption of v_k under pp.pk.

Recall that by assumption the output $(\vec{v}, m)$ of query $q_{D,i}$ is such that $\mathtt{pk}_j \in \vec{v}$. Consider any $l \in \{1, \ldots, |\vec{v}|\}$ with $v_l = \mathtt{pk}_j$. Given $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{adap}}}$ and letting $k_{\mathtt{sym}}$ be the same as above, it follows from the correctness of each party's $(\mathtt{pk}_b, \mathtt{sk}_b)$ key-pair (where b is the bit in the party's secret key) that if the oracle tries to decrypt $c_{l,b,2}$ using B_j's secret key, the decryption will yield $k_{\mathtt{sym}}$. By the correctness of Π_{SKE} and since $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{adap}}}$, any decryption of $c_{\mathtt{sym}}$ under $k_{\mathtt{sym}}$ results in the same pair $(\vec{v}, m)$. Noting that the correctness of pp.pk implies that the oracle will not attempt to decrypt any ciphertext $c_{k,b,2}$ with $v_k \neq \mathtt{pk}_j$ (where $k \in \{1, \ldots, |\vec{v}|\}$)—since there is no sequence of coins r such that $c_{k,0} = \Pi_{\text{PKE}}.E(\mathtt{pp.pk}, \mathtt{pk}_j; r)$—it then follows that if $q_{D,j}$ does not output $\bot$, then it outputs the same pair as query $q_{D,i}$.

To conclude the proof it only remains to show that query $q_{D,j}$ does not output $\bot$. Since Π_{NIZK}'s proof verification algorithm V is deterministic and because query $q_{D,i}$ does not output $\bot$, the NIZK proof p in ciphertext c also verifies as being valid in query $q_{D,j}$. Furthermore, on one hand, and as argued above, the correctness of pp.pk implies the oracle will skip decryption attempts for every index $k \in \{1, \ldots, |\vec{v}|\}$ such that $\mathtt{pk}_j \neq v_k$. On the other hand, since $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \in L_{\text{PKEBC}^{\text{adap}}}$ and due to the correctness of B_j's $(\mathtt{pk}_b, \mathtt{sk}_b)$ key-pair (where b is the bit in the B_j's secret key), for each $l \in \{1, \ldots, |\vec{v}|\}$ such that $v_l = \mathtt{pk}_j$, decrypting $c_{l,b,1}$ would result in a sequence r such that

$c_{l,0} = \mathit{\Pi}_{\mathrm{PKE}}.E(\mathtt{pp.pk}, \mathtt{pk}_j; r)$. This means that for the least $l \in \{1, \ldots, |\vec{v}|\}$ such that $v_l = \mathtt{pk}_j$ the oracle will attempt to decrypt $c_{l,b,2}$; as argued above, this implies that on query $q_{D,j}$ the oracle will output the same pair $(\vec{v}, m)$ as it did on query $q_{D,i}$, which concludes the proof of this claim. □

B.3.4 Proof of $(\mathsf{IND} + \mathsf{IK})$-$\mathsf{CCA}$-$\mathsf{2}^{\mathsf{adap}}$ Security

Theorem 12. *If $\mathit{\Pi}_{\mathrm{PKE}}$ is*

$$\begin{aligned}(\varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}}, &\varepsilon_{\mathrm{PKE}\text{-}\mathsf{IND}\text{-}\mathsf{CPA}}, \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IK}\text{-}\mathsf{CPA}}, \\ &t_{\mathrm{PKE}}, n_{\mathrm{PKE}}, q_{E\,\mathrm{PKE}})\text{-}secure,\end{aligned} \tag{B.21}$$

with $n_{\mathrm{PKE}} \geq 1$, $\mathit{\Pi}_{\mathrm{NIZK}}$ is

$$\begin{aligned}(\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Complete}}, &\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Sound}}, \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{ZK}}, \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{SS}}, \\ &t_{\mathrm{NIZK}}, q_{P\,\mathrm{NIZK}}, q_{V\,\mathrm{NIZK}})\text{-}secure,\end{aligned} \tag{B.22}$$

and $\mathit{\Pi}_{\mathrm{SKE}}$ is

$$(\varepsilon_{\mathrm{SKE}\text{-}\mathsf{1}\text{-}\mathsf{IND}\text{-}\mathsf{CPA}}, t_{\mathrm{SKE}}, q_{E\,\mathrm{SKE}})\text{-}secure, \tag{B.23}$$

then no adversary $\mathbf{A}$ (ε, t)-breaks $\mathit{\Pi}^{\mathsf{adap}}_{\mathrm{PKEBC}}$'s

$$\begin{aligned}&(n \coloneqq n_{\mathrm{PKE}} - 2, d_E \coloneqq q_{E\,\mathrm{PKE}}, q_E \coloneqq \min(q_{P\,\mathrm{NIZK}}, q_{E\,\mathrm{SKE}}), \\ &q_D \coloneqq q_{V\,\mathrm{NIZK}})\text{-}(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}\mathsf{2}^{\mathsf{adap}}\ security,\end{aligned}$$

with

$$\begin{aligned}\varepsilon &> (16 \cdot \varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}} + 12 \cdot \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IND}\text{-}\mathsf{CPA}} + 8 \cdot \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IK}\text{-}\mathsf{CPA}}) \\ &\quad + 2 \cdot (\varepsilon_{\mathrm{NIZK}\text{-}\mathsf{ZK}} + \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{Sound}} + \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{SS}}) \\ &\quad + 2 \cdot \varepsilon_{\mathrm{SKE}\text{-}\mathit{1}\text{-}\mathsf{IND}\text{-}\mathsf{CPA}}, \\ t_{\mathrm{PKE}}, t_{\mathrm{SKE}} &\approx t + t_{(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}\mathsf{2}^{\mathsf{adap}}} + q_E \cdot t_{S_P} + t_{S_G}, \\ t_{\mathrm{NIZK}} &\approx t + t_{(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}\mathsf{2}^{\mathsf{adap}}},\end{aligned}$$

where $t_{(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}\mathsf{2}^{\mathsf{adap}}}$ is the time to run the $\mathbf{G}_{\beta}^{(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}\mathsf{2}^{\mathsf{adap}}}$ game experiment of $\mathit{\Pi}^{\mathsf{adap}}_{\mathrm{PKEBC}}$'s (for any $\beta \in \{0, 1\}$), t_{S_P} is the runtime of $\mathit{\Pi}_{\mathrm{NIZK}}$'s S_P algorithm, and t_{S_G} is the runtime of $\mathit{\Pi}_{\mathrm{NIZK}}$'s S_G algorithm.

The proof of Theorem 12 relies on an alternative decryption procedure that is defined in Algorithm 9.

Algorithm 9 Alternative decryption algorithm for the $(\mathsf{IND}+\mathsf{IK})$-$\mathsf{CCA}$-$2$ security reductions. Below, $\mathtt{sk_{pp}}$ is the secret key corresponding to the public parameter's public key (i.e. $\mathtt{pp.pk}$) and $\mathtt{pk}$ is the public key of the party who is supposed to decrypt c.

$D_{\mathsf{pp}}(\mathtt{sk}_{\mathsf{pp}}, \mathtt{pk}, c := (p, c_{\mathsf{pp}}, \vec{c}, c_{\mathsf{sym}}))$
  if $\Pi_{\text{NIZK}}.V_{\mathsf{crs}}\big((1^k, \mathtt{pp.pk}, c_{\mathsf{pp}}, \vec{c}, c_{\mathsf{sym}}) \in L_{\text{PKEBC}^{\mathsf{adap}}}, p\big) = \mathtt{valid}$ **then**
    $m \leftarrow \Pi_{\text{PKE}}.D_{\mathsf{sk_{pp}}}(c_{\mathsf{pp}})$
    for $j = 1, \ldots, |\vec{c}|$ **do**
      $v_j \leftarrow \Pi_{\text{PKE}}.D_{\mathsf{sk_{pp}}}(c_{j,0})$
    $\vec{v} := (v_1, \ldots, v_{|\vec{v}|})$
    if $\mathtt{pk} \in \vec{v}$ **then** ▷ By Π_{PKE}'s Correctness and Π_{NIZK}'s Soundness, $(\vec{v}, m) \neq \bot$
      return $(\vec{v}, m)$
  return $\bot$

Table 1: Sequence of hybrids for proving the $(\mathsf{IND}+\mathsf{IK})$-CCA-2 security of our scheme. The first row specifies, for $\beta \in \{0,1\}$, $\mathbf{G}^{\mathsf{CCA}}_{\beta}$; each of the following rows specifies a hybrid game $\mathbf{G}^{i}_{\beta}$, with $\beta \in \{0,1\}$. The ε column indicates, for each row, an upper bound in an adversary's advantage in distinguishing between that row's game and the previous row's game. The non-shaded cells of each row specify how the hybrid differs from the previous game (except for the first row, which specifies $\mathbf{G}^{\mathsf{CCA}}_{\beta}$). Columns: NIZK: indicates whether the $\texttt{crs}$ output by $\mathcal{O}_{PP}$ and the proofs output by $\mathcal{O}_E$ queries are real ones (Real) or simulated ones (Sim); $\mathcal{O}_D$: indicates how the game handles decryption queries—$c_{\mathtt{sym}}$ indicates that the game handles decryption queries by following the normal decryption procedure, whereas $c_{\mathtt{pp}}$ indicates that the game handles decryption queries by following the alternative decryption procedure (see Algorithm 9); $\mathcal{O}_E$: indicates how $\mathcal{O}_E$ queries are handled—columns named Value indicate what value is encrypted (by the part of the ciphertext corresponding to that column), and columns named $\mathtt{pk}$ indicate what public key is used (by the part of the ciphertext corresponding to that column); Correctness: columns $\mathtt{pp.pk}$, $\mathtt{pk}_j.\mathtt{pk}_0$ and $\mathtt{pk}_j.\mathtt{pk}_1$ indicate whether the Π_{PKE} key pairs (corresponding to the columns public key) are correct; column Π_{SKE} indicates whether Π_{SKE} is assumed to be perfectly correct. In columns $c_{i,b,1}$, $c_{i,\bar{b},1}$, $c_{i,b,2}$ and $c_{i,\bar{b},2}$ under $\mathcal{O}_E$, b denotes the bit of the secret key of V_{β_i}, and $\bar{b}$ denotes the complement of b, i.e. $\bar{b} \coloneqq 1-b$. In the table, $\mathtt{pk}_0$ and $\mathtt{pk}_1$ are two Π_{PKE} public keys that are (honestly) sampled independently from all parties' public keys and independently from $\mathtt{pp.pk}$; these public keys are only used for the IK-CPA security reductions. Finally, all values 0 under columns Value are assumed to have appropriate length. For instance, in $\mathbf{G}^{9}_{\beta}$ the 0 is assumed to be of the same length as $k_{\mathtt{sym}}$, and in $\mathbf{G}^{14}_{\beta}$ the 0 is assumed to be of the same length as $(\vec{v}_\beta, m_\beta)$.

Hybrid	NIZK	$\mathcal{O}_D$	$\mathcal{O}_E\big((\vec{V}_0,m_0),(\vec{V}_1,m_1)\big)$											Correctness				ε
			$c_{\mathtt{pp}}$	$c_{\mathtt{sym}}$	$c_{i,0}$	$c_{i,b,1}$		$c_{i,\bar{b},1}$		$c_{i,b,2}$		$c_{i,\bar{b},2}$		$\mathtt{pp.pk}$	Π_{SKE}	$\mathtt{pk}_j.\mathtt{pk}_0$	$\mathtt{pk}_j.\mathtt{pk}_1$	
			Value	Value	Value	Value	$\mathtt{pk}$	Value	$\mathtt{pk}$	Value	$\mathtt{pk}$	Value	$\mathtt{pk}$					
$\mathbf{G}^{\mathsf{CCA}}_{\beta}$	Real	$c_{\mathtt{sym}}$	m_β	$(\vec{v}_\beta, m_\beta)$	v_{β_i}	$r_{i,0}$	$v_{\beta_i}.\mathtt{pk}_b$	$r_{i,0}$	$v_{\beta_i}.\mathtt{pk}_{\bar{b}}$	$k_{\mathtt{sym}}$	$v_{\beta_i}.\mathtt{pk}_b$	$k_{\mathtt{sym}}$	$v_{\beta_i}.\mathtt{pk}_{\bar{b}}$	Normal	Normal	Normal	Normal	
$\mathbf{G}^{1}_{\beta}$																	Correct	$\leq \varepsilon_{\mathrm{PKE\text{-}Corr}}$
$\mathbf{G}^{2}_{\beta}$																Correct		$\leq \varepsilon_{\mathrm{PKE\text{-}Corr}}$
$\mathbf{G}^{3}_{\beta}$															Correct			$= 0$
$\mathbf{G}^{4}_{\beta}$														Correct				$\leq \varepsilon_{\mathrm{PKE\text{-}Corr}}$
$\mathbf{G}^{5}_{\beta}$		$c_{\mathtt{pp}}$																$\leq \varepsilon_{\mathrm{NIZK\text{-}Sound}}$
$\mathbf{G}^{6}_{\beta}$	Sim																	$\leq \varepsilon_{\mathrm{NIZK\text{-}ZK}}$
$\mathbf{G}^{7}_{\beta}$																	Normal	$\leq \varepsilon_{\mathrm{PKE\text{-}Corr}}$
$\mathbf{G}^{8}_{\beta}$																Normal		$\leq \varepsilon_{\mathrm{PKE\text{-}Corr}}$
$\mathbf{G}^{9}_{\beta}$												0						$\leq \varepsilon_{\mathrm{PKE\text{-}IND\text{-}CPA}}$
$\mathbf{G}^{10}_{\beta}$													$\mathtt{pk}_{\bar{b}}$					$\leq \varepsilon_{\mathrm{PKE\text{-}IK\text{-}CPA}}$
$\mathbf{G}^{11}_{\beta}$										0	$\mathtt{pk}_b$	$k_{\mathtt{sym}}$	$v_{\beta_i}.\mathtt{pk}_{\bar{b}}$					$= 0$
$\mathbf{G}^{12}_{\beta}$												0						$\leq \varepsilon_{\mathrm{PKE\text{-}IND\text{-}CPA}}$
$\mathbf{G}^{13}_{\beta}$													$\mathtt{pk}_{\bar{b}}$					$\leq \varepsilon_{\mathrm{PKE\text{-}IK\text{-}CPA}}$
$\mathbf{G}^{14}_{\beta}$				0														$\leq \varepsilon_{\mathrm{SKE\text{-}1\text{-}IND\text{-}CPA}}$
$\mathbf{G}^{15}_{\beta}$								0										$\leq \varepsilon_{\mathrm{PKE\text{-}IND\text{-}CPA}}$
$\mathbf{G}^{16}_{\beta}$									$\mathtt{pk}_{\bar{b}}$									$\leq \varepsilon_{\mathrm{PKE\text{-}IK\text{-}CPA}}$

$\mathbf{G}_\beta^{17}$						0	$\mathtt{pk}_b$	$r_{i,\bar{b},0}$	$v_{\beta_i}.\mathtt{pk}_{\bar{b}}$						$= 0$
$\mathbf{G}_\beta^{18}$								0							$\leq \varepsilon_{\text{PKE-IND-CPA}}$
$\mathbf{G}_\beta^{19}$									$\mathtt{pk}_{\bar{b}}$						$\leq \varepsilon_{\text{PKE-IK-CPA}}$
$\mathbf{G}_\beta^{20}$														Correct	$\leq \varepsilon_{\text{PKE-Corr}}$
$\mathbf{G}_\beta^{21}$													Correct		$\leq \varepsilon_{\text{PKE-Corr}}$
$\mathbf{G}_\beta^{22}$		$c_{\mathtt{sym}}$													$\leq \varepsilon_{\text{NIZK-SS}}$
$\mathbf{G}_\beta^{23}$											Normal				$\leq \varepsilon_{\text{PKE-Corr}}$
$\mathbf{G}_\beta^{24}$					0										$\leq \varepsilon_{\text{PKE-IND-CPA}}$
$\mathbf{G}_\beta^{25}$			0												$\leq \varepsilon_{\text{PKE-IND-CPA}}$

Proof. This proof proceeds in a sequence of game hops.

For simplicity of notation, we will refer to $\mathbf{G}_\beta^{\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}$ as $\mathbf{G}_\beta^{\mathsf{CCA}}$, for $\beta \in \{0,1\}$. For any given adversary $\mathbf{A}$, we bound $\mathbf{A}$'s advantage

$$Adv^{\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}(\mathbf{A}) := \Big| \Pr[\mathbf{A}\mathbf{G}_\mathbf{0}^{\mathsf{CCA}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_\mathbf{1}^{\mathsf{CCA}} = \mathtt{win}] - 1 \Big|,$$

by bounding, for $\beta \in \{0,1\}$, the difference between the probability of $\mathbf{A}$ winning $\mathbf{G}_\beta^{\mathsf{CCA}}$ and winning $\mathbf{G}_\beta^1$, and, for $i \in \{1,\ldots,24\}$, the difference between the probability of $\mathbf{A}$ winning $\mathbf{G}_\beta^i$ and winning $\mathbf{G}_\beta^{i+1}$. In other words, for $\beta \in \{0,1\}$, we bound

$$\Big| \Pr[\mathbf{A}\mathbf{G}_\beta^{\mathsf{CCA}} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^1 = \mathtt{win}] \Big|,$$

and bound, for $i \in \{1,\ldots,24\}$,

$$\Big| \Pr[\mathbf{A}\mathbf{G}_\beta^i = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^{i+1} = \mathtt{win}] \Big|.$$

Since $\mathbf{G}_\mathbf{0}^{25}$ is perfectly indistinguishable from $\mathbf{G}_\mathbf{1}^{25}$,

$$\Big| \Pr[\mathbf{A}\mathbf{G}_\mathbf{0}^{25} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_\mathbf{1}^{25} = \mathtt{win}] - 1 \Big| = 0.$$

This then implies

$$\begin{aligned} Adv^{\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}(\mathbf{A}) &:= \\ &\Big| \Pr[\mathbf{A}\mathbf{G}_\mathbf{0}^{\mathsf{CCA}} = \mathtt{win}] + \Pr[\mathbf{A}\mathbf{G}_\mathbf{1}^{\mathsf{CCA}} = \mathtt{win}] - 1 \Big| \\ \le &\Big| \Pr[\mathbf{A}\mathbf{G}_\mathbf{0}^{\mathsf{CCA}} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\mathbf{0}^1 = \mathtt{win}] \Big| \\ + &\sum_{i=1,\ldots,24} \Big| \Pr[\mathbf{A}\mathbf{G}_\mathbf{0}^i = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\mathbf{0}^{i+1} = \mathtt{win}] \Big| \\ + &\sum_{i=1,\ldots,24} \Big| \Pr[\mathbf{A}\mathbf{G}_\mathbf{1}^i = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\mathbf{1}^{i+1} = \mathtt{win}] \Big| \\ + &\Big| \Pr[\mathbf{A}\mathbf{G}_\mathbf{1}^{\mathsf{CCA}} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\mathbf{1}^1 = \mathtt{win}] \Big|. \end{aligned}$$

The sequence of games is given in Table 1; by summing up the differences of the winning probabilities, one obtains

$$\begin{aligned} Adv^{\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}(\mathbf{A}) \le\ & 2 \cdot \varepsilon_{\mathrm{SKE\text{-}1\text{-}IND\text{-}CPA}} \\ & + (16 \cdot \varepsilon_{\mathrm{PKE\text{-}Corr}} + 12 \cdot \varepsilon_{\mathrm{PKE\text{-}IND\text{-}CPA}} + 8 \cdot \varepsilon_{\mathrm{PKE\text{-}IK\text{-}CPA}}) \\ & + 2 \cdot (\varepsilon_{\mathrm{NIZK\text{-}ZK}} + \varepsilon_{\mathrm{NIZK\text{-}Sound}} + 2 \cdot \varepsilon_{\mathrm{NIZK\text{-}SS}}). \end{aligned}$$

We now justify each game hop in Table 1.

$\mathbf{G}_\beta^{\mathsf{CCA}} \rightsquigarrow \mathbf{G}_\beta^1$**:** For $\beta \in \{0,1\}$, one can reduce distinguishing $\mathbf{G}_\beta^1$ and $\mathbf{G}_\beta^{\mathsf{CCA}}$ to breaking Π_{PKE}'s correctness: the reduction holds the secret key of every party and so it can trivially handle any oracle queries. Noting that the reduction only has to query oracle Π_{PKE}-$\mathcal{O}_{SK}$ on at most $n \leq n_{\mathrm{PKE}}$ key-pairs, since from Equation B.21 no adversary ($\varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}$)-breaks Π_{PKE}'s (n_{PKE})-Correctness, it follows

$$\left|\Pr[\mathbf{A}\mathbf{G}_\beta^{\mathsf{CCA}} = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^1 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}.$$

$\mathbf{G}_\beta^1 \rightsquigarrow \mathbf{G}_\beta^2$**:** Analogous to $\mathbf{G}_\beta^{\mathsf{CCA}} \rightsquigarrow \mathbf{G}_\beta^1$.

$\mathbf{G}_\beta^2 \rightsquigarrow \mathbf{G}_\beta^3$**:** By Equation B.23, Π_{SKE} is perfectly correct, implying

$$\left|\Pr[\mathbf{A}\mathbf{G}_\beta^2 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^3 = \mathtt{win}]\right| = 0.$$

$\mathbf{G}_\beta^3 \rightsquigarrow \mathbf{G}_\beta^4$**:** Similar to $\mathbf{G}_\beta^{\mathsf{CCA}} \rightsquigarrow \mathbf{G}_\beta^1$, except that this time the reduction only has to query oracle Π_{PKE}-$\mathcal{O}_{SK}$ on at most one key-pair; since $n_{\mathrm{PKE}} \geq 1$ it then follows by Equation B.21 that

$$\left|\Pr[\mathbf{A}\mathbf{G}_\beta^3 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^4 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE\text{-}\mathsf{Corr}}}.$$

$\mathbf{G}_\beta^4 \rightsquigarrow \mathbf{G}_\beta^5$**:** Note that, for $\beta \in \{0,1\}$, one can reduce distinguishing $\mathbf{G}_\beta^4$ and $\mathbf{G}_\beta^5$ to breaking the soundness of the underlying Π_{NIZK}: since at this point we are assuming perfect correctness of both Π_{SKE} and Π_{PKE}, an adversary can only distinguish $\mathbf{G}_\beta^4$ from $\mathbf{G}_\beta^5$ if it submits a decryption query for a ciphertext $c \coloneqq (p, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}})$ such that the NIZK proof p verifies as being valid but the corresponding statement is not true (i.e. $(1^k, \mathtt{pp.pk}, c_{\mathtt{pp}}, \vec{c}, c_{\mathtt{sym}}) \notin L_{\mathrm{PKEBC}^{\mathsf{adap}}}$). In particular, since the reduction holds the secret key of every party B_j—which consists of B_j's public key $\mathtt{pk}_j$, the secret bit b_j and the secret key $\mathtt{sk}_{b_j}$—it can handle any secret key queries. Moreover, it also holds the secret key corresponding to the public parameter's public key, it can handle any decryption oracle queries by using the alternative decryption procedure defined in Algorithm 9. Noting that the reduction queries the underlying Π_{NIZK}-$\mathcal{O}_V$ oracle only once for each $\mathcal{O}_D$ query, if adversary $\mathbf{A}$ makes at most $q_D \leq q_{V\,\mathrm{NIZK}}$ queries to $\mathcal{O}_D$, it follows from Equation B.22

$$\left|\Pr[\mathbf{A}\mathbf{G}_\beta^4 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^5 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{NIZK\text{-}\mathsf{Sound}}}.$$

$\mathbf{G}_\beta^5 \rightsquigarrow \mathbf{G}_\beta^6$**:** It is easy to see that one can reduce distinguishing these two games to breaking Π_{NIZK}'s Zero-Knowledge, as the reduction holds all secret keys (including the secret key corresponding to $\mathtt{pp.pk}$) and thus can handle any oracle queries. (Although the reduction does not have the trapdoor for the $\mathtt{crs}$, in case the $\mathtt{crs}$ is a simulated one, since it only has to prove true statements it can rely on oracle Π_{NIZK}-$\mathcal{O}_P$ for generating the necessary NIZK proofs.) Noting the reduction only has to generate at most one NIZK proof for each $\mathcal{O}_E$ query, if $q_E \leq q_{P\,\mathrm{NIZK}}$ then, since from Equation B.22 no adversary $(\varepsilon_{\mathrm{NIZK\text{-}\mathsf{ZK}}}, t_{\mathrm{NIZK}})$-breaks Π_{NIZK}'s $(q_{P\,\mathrm{NIZK}})$-ZK, it follows

$$\left|\Pr[\mathbf{A}\mathbf{G}_\beta^5 = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^6 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{NIZK\text{-}\mathsf{ZK}}}.$$

$\mathbf{G}_\beta^6 \rightsquigarrow \mathbf{G}_\beta^7$**:** Analogous to $\mathbf{G}_\beta^{\mathsf{CCA}} \rightsquigarrow \mathbf{G}_\beta^1$, but the $(\mathtt{pk}_1, \mathtt{sk}_1)$ key-pair of each party is now assumed to be sampled according to the original Π_{PKE}'s key-pair distribution (induced by $\Pi_{\mathrm{PKE}}.G$), instead of being assumed to be correct; it follows

$$\left|\Pr[\mathbf{AG}_\beta^6 = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^7 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{Corr}}.$$

$\mathbf{G}_\beta^7 \rightsquigarrow \mathbf{G}_\beta^8$**:** Analogous to $\mathbf{G}_\beta^6 \rightsquigarrow \mathbf{G}_\beta^7$.

$\mathbf{G}_\beta^8 \rightsquigarrow \mathbf{G}_\beta^9$**:** One can reduce distinguishing $\mathbf{G}_\beta^9$ and $\mathbf{G}_\beta^8$ to breaking the IND-CPA security of Π_{PKE}. In contrast to prior reductions, this time the reduction does not have, for each party B_j, both $\mathtt{sk}_0$ and $\mathtt{sk}_1$. However, since the scheme itself discards one of the secret keys, namely $\mathtt{sk}_{\bar{b}}$ (with $\bar{b} \coloneqq 1 - b$, b being the bit in the party's secret key), the reduction can still handle $\mathcal{O}_{SK}$ queries by generating itself the key-pair $(\mathtt{pk}_b, \mathtt{sk}_b)$ of each party and relying on the underlying oracle Π_{PKE}-$\mathcal{O}_{PK}$ to generate public key $\mathtt{pk}_{\bar{b}}$. This means the reduction can still handle queries to $\mathcal{O}_{SK}$. Regarding queries to $\mathcal{O}_D$, the reduction relies on the secret key $\mathtt{sk}_{\mathtt{pp}}$ corresponding to the public parameters public key as before. Regarding $\mathcal{O}_E$ queries, note that although the reduction now has to prove NIZK statements that it either does not have a witness for (in $\mathbf{G}_\beta^8$) or are even false (in $\mathbf{G}_\beta^9$), since the NIZK's $\mathtt{crs}$ is a simulated one generated by the reduction, the reduction holds the $\mathtt{crs}$ trapdoor τ that allows it to simulate NIZK proofs without a witness. Finally, if $\mathbf{A}$ only queries on at most $n \leq n_{\mathrm{PKE}}$ different parties and queries Π_{PKE}-$\mathcal{O}_E$ at most $d_E \leq q_{E\,\mathrm{PKE}}$ times, since from Equation B.21 no adversary $(\varepsilon_{\mathrm{PKE}\text{-}\mathsf{IND\text{-}CPA}}, t_{\mathrm{PKE}})$-breaks Π_{PKE}'s $(n_{\mathrm{PKE}}, q_{E\,\mathrm{PKE}})$-IND-CPA security, it follows

$$\left|\Pr[\mathbf{AG}_\beta^8 = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^9 = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IND\text{-}CPA}}.$$

$\mathbf{G}_\beta^9 \rightsquigarrow \mathbf{G}_\beta^{10}$**:** One can reduce distinguishing games $\mathbf{G}_\beta^{10}$ and $\mathbf{G}_\beta^9$ to breaking the IK-CPA security of Π_{PKE} in a very similar way to the previous step (i.e. $\mathbf{G}_\beta^8 \rightsquigarrow \mathbf{G}_\beta^9$). The only difference is in how to handle $\mathcal{O}_E$ queries: since now we have to swap the public key used for encryption, we cannot simply swap it with another party's public key: on one hand it is crucial the reduction has each party's secret key in order to handle $\mathcal{O}_{SK}$ queries as before—and thus the reduction cannot simply rely on Π_{PKE}-$\mathcal{O}_{PK}$ to create all public keys—and on the other hand, note that when reducing to the IK-CPA security of the underlying Π_{PKE} scheme, if two parties, say B_j and ${B_j}'$ have, respectively, bits b_j and ${b_j}'$ in their secret keys, and these bits are such that $b_j \neq {b_j}'$, then the reduction cannot simply change the encryption public key of a Π_{PKE} ciphertext from B_j's public key $\mathtt{pk}_{\bar{b_j}}$ to ${B_j}'$'s public key $\mathtt{pk}_{\bar{b_j}'}$. To avoid this issue we instead use two additional public keys, $\mathtt{pk}_0$ and $\mathtt{pk}_1$ in the reduction, that are generated by the underlying Π_{PKE}-$\mathcal{O}_{PK}$ oracle and which become the new encryption keys. Since the reduction queries on at most $n \leq n_{\mathrm{PKE}} - 2$ different parties and queries Π_{PKE}-$\mathcal{O}_E$ at most $d_E \leq q_{E\,\mathrm{PKE}}$ times, it follows from Equation B.21 that no adversary

$$\left|\Pr[\mathbf{AG}_\beta^9 = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{10} = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{PKE}\text{-}\mathsf{IK\text{-}CPA}}.$$

$\mathbf{G}_\beta^{10} \rightsquigarrow \mathbf{G}_\beta^{11}$**:** Note that, for any party B_j, if the adversary makes a query $\mathcal{O}_E\big((\vec{V}_0, m_0), (\vec{V}_1, m_1)\big)$, where $B_j \in \mathrm{Set}(\vec{V}_0) \cup \mathrm{Set}(\vec{V}_1)$, then it cannot make any query to $\mathcal{O}_{SK}$ on B_j, implying the adversary can never learn the secret bit in B_j's secret key. This implies that $\mathbf{G}_\beta^{11}$ is perfectly indistinguishable from $\mathbf{G}_\beta^{10}$. Hence

$$\left|\Pr[\mathbf{AG}_\beta^{10} = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{11} = \mathtt{win}]\right| = 0.$$

$\mathbf{G}_\beta^{11} \rightsquigarrow \mathbf{G}_\beta^{12}$**:** Analogous to $\mathbf{G}_\beta^{8} \rightsquigarrow \mathbf{G}_\beta^{9}$.

$\mathbf{G}_\beta^{12} \rightsquigarrow \mathbf{G}_\beta^{13}$**:** Analogous to $\mathbf{G}_\beta^{9} \rightsquigarrow \mathbf{G}_\beta^{10}$.

$\mathbf{G}_\beta^{13} \rightsquigarrow \mathbf{G}_\beta^{14}$**:** Note that one can reduce distinguishing these two games to breaking the 1-IND-CPA security of the underlying Π_{SKE} scheme: since the reduction holds exactly the same information as it did in the last few games, it can handle both decryption oracle queries and secret key queries as before. Given that for each query to $\mathcal{O}_E$ the reduction queries Π_{SKE}-$\mathcal{O}_E$ at most once, by Equation B.23 it follows

$$\left|\Pr[\mathbf{AG}_\beta^{13} = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{14} = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{SKE}\text{-}\mathsf{1\text{-}IND\text{-}CPA}}.$$

$\mathbf{G}_\beta^{14} \rightsquigarrow \mathbf{G}_\beta^{15}$**:** Analogous to $\mathbf{G}_\beta^{8} \rightsquigarrow \mathbf{G}_\beta^{9}$.

$\mathbf{G}_\beta^{15} \rightsquigarrow \mathbf{G}_\beta^{16}$**:** Analogous to $\mathbf{G}_\beta^{9} \rightsquigarrow \mathbf{G}_\beta^{10}$.

$\mathbf{G}_\beta^{16} \rightsquigarrow \mathbf{G}_\beta^{17}$**:** Analogous to $\mathbf{G}_\beta^{10} \rightsquigarrow \mathbf{G}_\beta^{11}$.

$\mathbf{G}_\beta^{17} \rightsquigarrow \mathbf{G}_\beta^{18}$**:** Analogous to $\mathbf{G}_\beta^{8} \rightsquigarrow \mathbf{G}_\beta^{9}$.

$\mathbf{G}_\beta^{18} \rightsquigarrow \mathbf{G}_\beta^{19}$**:** Analogous to $\mathbf{G}_\beta^{9} \rightsquigarrow \mathbf{G}_\beta^{10}$.

$\mathbf{G}_\beta^{19} \rightsquigarrow \mathbf{G}_\beta^{20}$**:** Analogous to $\mathbf{G}_\beta^{\mathsf{CCA}} \rightsquigarrow \mathbf{G}_\beta^{1}$.

$\mathbf{G}_\beta^{20} \rightsquigarrow \mathbf{G}_\beta^{21}$**:** Analogous to $\mathbf{G}_\beta^{\mathsf{CCA}} \rightsquigarrow \mathbf{G}_\beta^{1}$.

$\mathbf{G}_\beta^{21} \rightsquigarrow \mathbf{G}_\beta^{22}$**:** This step is similar to $\mathbf{G}_\beta^{4} \rightsquigarrow \mathbf{G}_\beta^{5}$, except that now the reduction has to prove a false NIZK statement each time $\mathcal{O}_E$ is queried. Although the reduction does not have the trapdoor to the simulated $\mathtt{crs}$, it can rely on the Π_{NIZK}-$\mathcal{O}_P$ oracle to obtain these proofs. As before, since the reduction holds the secret key of every party, it can handle any oracle queries. Given the reduction only has to generate one NIZK proof for each $\mathcal{O}_E$ query, and only has to verify one NIZK proof for each $\mathcal{O}_D$ query, since there are at most $q_E \leq q_{P_{\mathrm{NIZK}}}$ queries to $\mathcal{O}_E$ and $q_D \leq q_{V_{\mathrm{NIZK}}}$ queries to $\mathcal{O}_D$, it follows by Equation B.22

$$\left|\Pr[\mathbf{AG}_\beta^{21} = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^{22} = \mathtt{win}]\right| \leq \varepsilon_{\mathrm{NIZK}\text{-}\mathsf{SS}}.$$

$\mathbf{G}_\beta^{22} \rightsquigarrow \mathbf{G}_\beta^{23}$**:** Analogous to $\mathbf{G}_\beta^{3} \rightsquigarrow \mathbf{G}_\beta^{4}$.

$\mathbf{G}_\beta^{23} \rightsquigarrow \mathbf{G}_\beta^{24}$**:** Note that one can reduce distinguishing $\mathbf{G}_\beta^{24}$ and $\mathbf{G}_\beta^{23}$ to breaking the IND-CPA security of Π_{PKE}. Noting that the reduction only queries the underlying security game for one public key—namely pp.pk—and since the reduction queries the underlying Π_{PKE}-$\mathcal{O}_E$ oracle at most $d_E \leq q_{E\,\text{PKE}}$ times, it follows from Equation B.21

$$\left|\Pr[\mathbf{A}\mathbf{G}_\beta^{23} = \texttt{win}] - \Pr[\mathbf{A}\mathbf{G}_\beta^{24} = \texttt{win}]\right| \leq \varepsilon_{\text{PKE-}\mathsf{IND\text{-}CPA}}.$$

$\mathbf{G}_\beta^{24} \rightsquigarrow \mathbf{G}_\beta^{25}$**:** Analogous to $\mathbf{G}_\beta^{23} \rightsquigarrow \mathbf{G}_\beta^{24}$, except that the reduction only queries the underlying Π_{PKE}-$\mathcal{O}_E$ oracle at most $q_E \leq q_{E\,\text{PKE}}$ times.

To conclude the proof, note that each reduction simply has to emulate the original game towards $\mathbf{A}$—with a few tweaks that, apart from the generation of a simulated crs and the generation of simulated NIZK proofs, do not affect the time for emulating the game. Letting $t_{(\mathsf{IND+IK})\text{-}\mathsf{CCA\text{-}2^{adap}}}$ be the time to run Π's $\mathbf{G}_\beta^{\mathsf{CCA}}$ game experiment (with $\beta \in \{0,1\}$), t_{S_P} be the runtime of Π_{NIZK}'s S_P and t_{S_G} be the runtime of Π_{NIZK}'s S_G, it follows

$$t_{\text{PKE}}, t_{\text{SKE}} \approx t + t_{(\mathsf{IND+IK})\text{-}\mathsf{CCA\text{-}2^{adap}}} + q_E \cdot t_{S_P} + t_{S_G},$$
$$t_{\text{NIZK}} \approx t + t_{(\mathsf{IND+IK})\text{-}\mathsf{CCA\text{-}2^{adap}}}.$$

□

B.4 MDRS-PKE Construction Security Proofs

Below we give the $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA\text{-}2^{adap}}$ security proof of Maurer et al.'s MDRS-PKE construction [20] for the new setting considered in this paper.

B.4.1 Proof of $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA\text{-}2^{adap}}$ Security

Theorem 13. *If* Π_{PKEBC} *is*

$$(\varepsilon_{\text{PKEBC-}\mathsf{Corr}}, \varepsilon_{\text{PKEBC-}\mathsf{Rob}}, \varepsilon_{\text{PKEBC-}\mathsf{Cons}}, \varepsilon_{\text{PKEBC-}(\mathsf{IND+IK})\text{-}\mathsf{CCA\text{-}2^{adap}}}, t_{\text{PKEBC}}, n_{\text{PKEBC}}, d_{E\,\text{PKEBC}}, q_{E\,\text{PKEBC}}, q_{D\,\text{PKEBC}})\text{-}\textit{secure}, \tag{B.24}$$

Π_{MDVS} *is*

$$(\varepsilon_{\text{MDVS-}\mathsf{Corr}}, \varepsilon_{\text{MDVS-}\mathsf{Cons}}, \varepsilon_{\text{MDVS-}\mathsf{Unforg}}, \varepsilon_{\text{MDVS-}\mathsf{OTR}}, \varepsilon_{\text{MDVS-}\mathsf{Bound\text{-}Val}}, t_{\text{MDVS}}, n_{S\,\text{MDVS}}, n_{V\,\text{MDVS}}, d_{S\,\text{MDVS}}, q_{S\,\text{MDVS}}, q_{V\,\text{MDVS}})\text{-}\textit{secure}, \tag{B.25}$$

and Π_{DSS} *is*

$$(\varepsilon_{\text{DSS-}\mathsf{Corr}}, \varepsilon_{\text{DSS-}\mathsf{1\text{-}sEUF\text{-}CMA}}, t_{\text{DSS}}, n_{\text{DSS}}, q_{S\,\text{DSS}}, q_{V\,\text{DSS}})\text{-}\textit{secure}, \tag{B.26}$$

then no adversary $\mathbf{A}$ (ε, t)*-breaks* Π*'s*

$$\Big(n_S := n_{S\,\text{MDVS}},\quad n_R := \min(n_{\text{PKEBC}}, n_{V\,\text{MDVS}}),\quad d_E := \min(d_{E\,\text{PKEBC}}, d_{S\,\text{MDVS}}),\quad q_E := \min(q_{E\,\text{PKEBC}}, q_{S\,\text{MDVS}}, n_{\text{DSS}}, q_{S\,\text{DSS}}),\quad q_D := \min(q_{D\,\text{PKEBC}}, q_{V\,\text{MDVS}}, q_{V\,\text{DSS}})\Big)\text{-}(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$$ *security,*

with

$$\varepsilon > 2 \cdot (\varepsilon_{\text{DSS-}\mathsf{1\text{-}EUF\text{-}CMA}} + \varepsilon_{\text{MDVS-}\mathsf{Bound\text{-}Val}} + \varepsilon_{\text{PKEBC-}\mathsf{Rob}}) + 4 \cdot \varepsilon_{\text{PKEBC-}\mathsf{Corr}} + \varepsilon_{\text{PKEBC-}\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}},$$

and with $t_{\text{DSS}}, t_{\text{MDVS}}, t_{\text{PKEBC}} \approx t + t_{\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}$*, where* $t_{\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}$ *is the time to run* Π*'s* $\mathbf{G}^{\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}$ *games.*

Proof. Ciphertexts of our MDRS-PKE scheme are triples $c := (\mathtt{vk}, \sigma', c')$ where $\mathtt{vk}$ and σ' are, respectively, a verification key and a signature of the underlying Π_{DSS}, and c' is a Π_{PKEBC} ciphertext. At a high level, our goal is to reduce an adversary from distinguishing MDRS-PKE's $\mathbf{G_0}^{\mathsf{(IND+IK)\text{-}CCA\text{-}2}}$ and $\mathbf{G_1}^{\mathsf{(IND+IK)\text{-}CCA\text{-}2}}$ games to one distinguishing the analogous games for (the underlying) Π_{PKEBC}. If the adversary makes a decryption query

$$\mathcal{O}_D(B_j, c := (\mathtt{vk}, \sigma', c'))$$

where c was output by a challenge query to $\mathcal{O}_E$ the reduction can simply output `test` as this is a disallowed query; if c was not output by a query to $\mathcal{O}_E$ and the PKEBC ciphertext c' of c was also not output (as part of any ciphertext output by $\mathcal{O}_E$) then we can use the decryption oracle $\mathcal{O}_D$ of the $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ games of Π_{PKEBC}. However there is a problem when c was not output by any challenge query but its c' component was because this disallows us from using the decryption oracle $\mathcal{O}_D$ of Π_{PKEBC}'s $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security games. To get around this we will show—via a sequence of hybrids starting from $\mathbf{G}_\beta^{\mathsf{(IND+IK)\text{-}CCA\text{-}2}}$ and ending in $\mathbf{G}_\beta^5$, for $\beta \in \{0,1\}$—that such queries can be handled by simply outputting $\bot$, thus enabling a reduction to the $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security of the underlying Π_{PKEBC} scheme. Consider any query $\mathcal{O}_D(B_j, c := (\mathtt{vk}, \sigma', c'))$ such that c was not output by a challenge query to $\mathcal{O}_E$ but c' was (in the sense above); the following hybrids highlight the main steps of the proof:

$\mathbf{G}_\beta^1$: if any $\mathcal{O}_E$ query output $c^* := (\mathtt{vk}^*, {\sigma'}^*, {c'}^*)$ with $\mathtt{vk} = \mathtt{vk}^*$, $\mathcal{O}_D$ outputs $\bot$;

$\mathbf{G}_\beta^2$: if any query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ output $c^* := (\mathtt{vk}^*, {\sigma'}^*, {c'}^*)$ with $B_j \notin \vec{V}_\beta$ (and $\mathtt{vk} \neq \mathtt{vk}^*$), $\mathcal{O}_D$ outputs $\bot$;

$\mathbf{G}_\beta^5$: if any query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ output $c^* := (\mathtt{vk}^*, {\sigma'}^*, {c'}^*)$ with $B_j \in \vec{V}_\beta$ and $\mathtt{vk} \neq \mathtt{vk}^*$, $\mathcal{O}_D$ outputs $\bot$.

In $\mathbf{G}^5_\beta$ we know how to handle any decryption queries for ciphertexts that were not output by $\mathcal{O}_E$ but whose PKEBC component was, and hence we are (essentially) set to make the final reduction to the $(\mathsf{IND}+\mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security of Π_{PKEBC}. In the following, let $\beta \in \{0,1\}$.

$\mathbf{G}^{(\mathsf{IND+IK})\text{-}\mathsf{CCA}\text{-}2}_\beta \rightsquigarrow \mathbf{G}^1_\beta$: The difference between $\mathbf{G}^1_\beta$ and $\mathbf{G}^{(\mathsf{IND+IK})\text{-}\mathsf{CCA}\text{-}2}_\beta$ is that in $\mathbf{G}^1_\beta$ some decryption queries are handled differently: when $\mathcal{O}_D$ is queried on an input $(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ such that there is a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, {\sigma'}^*, {c'}^*)$ with $c \neq c^*$, $c' = {c'}^*$ and $\mathtt{vk} = \mathtt{vk}^*$, $\mathcal{O}_D$ simply outputs $\perp$.

One can reduce distinguishing the two games to breaking the 1-sEUF-CMA security of Π_{DSS}: since the reduction holds all MDVS and PKEBC secret keys and can sign ciphertexts using oracle $\mathcal{O}_S$ from Π_{DSS}'s $\mathbf{G}^{\text{1-sEUF-CMA}}$ game it can handle any oracle queries. If $\mathbf{A}$ only makes at most $q_E \leq \min(n_{\mathrm{DSS}}, q_{S_{\mathrm{DSS}}})$ and $q_D \leq q_{V_{\mathrm{DSS}}}$ queries to $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, since from Equation B.26 no adversary $(t_{\mathrm{DSS}}, \varepsilon_{\text{DSS-1-sEUF-CMA}})$-breaks the

$$(n_{\mathrm{DSS}}, q_{S_{\mathrm{DSS}}}, q_{V_{\mathrm{DSS}}})\text{-1-sEUF-CMA}$$

of Π_{DSS}, it follows

$$\left|\Pr[\mathbf{A}\mathbf{G}^1_\beta = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^{(\mathsf{IND+IK})\text{-}\mathsf{CCA}\text{-}2}_\beta = \mathtt{win}]\right| \leq \varepsilon_{\text{DSS-1-sEUF-CMA}}.$$

$\mathbf{G}^1_\beta \rightsquigarrow \mathbf{G}^2_\beta$: In $\mathbf{G}^2_\beta$ some decryption queries are once again handled differently; when $\mathcal{O}_D$ is queried on an input $(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ and there is a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, {\sigma'}^*, {c'}^*)$ with $c \neq c^*$, $c' = {c'}^*$, $\mathtt{vk} \neq \mathtt{vk}^*$, and $B_j \notin \vec{V}_\beta$, $\mathcal{O}_D$ outputs $\perp$.

One can reduce distinguishing $\mathbf{G}^2_\beta$ and $\mathbf{G}^1_\beta$ to breaking Π_{PKEBC}'s robustness (as defined in Definition 7). The main things to note are:

1. a reduction to Π_{PKEBC}'s robustness can access the secret keys of any party, and so it can handle any oracle queries that an adversary may make;
2. for a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$, the reduction can make a query $\mathcal{O}_E(A_{i,\beta}, \vec{V}_\beta, m_\beta)$ to the robustness game of Π_{PKEBC}; and
3. if a query $\mathcal{O}_D(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ does not output $\perp$, then the decryption by B_j of the PKEBC ciphertext c' (that is part of c) did not result in $\perp$ either.

If $\mathbf{A}$ only queries for at most $n_R \leq n_{\mathrm{PKEBC}}$ different receivers, the sum of lengths of the vectors input to $\mathcal{O}_E$ is at most $d_E \leq d_{E_{\mathrm{PKEBC}}}$, and $\mathbf{A}$ makes at most $q_E \leq q_{E_{\mathrm{PKEBC}}}$ and $q_D \leq q_{D_{\mathrm{PKEBC}}}$ queries to oracles $\mathcal{O}_E$ and $\mathcal{O}_D$, it follows

$$\left|\Pr[\mathbf{A}\mathbf{G}^2_\beta = \mathtt{win}] - \Pr[\mathbf{A}\mathbf{G}^1_\beta = \mathtt{win}]\right| \leq \varepsilon_{\text{PKEBC-Rob}}.$$

$\mathbf{G}_\beta^2 \rightsquigarrow \mathbf{G}_\beta^3$**:** In $\mathbf{G}_\beta^3$ some decryption queries are handled differently: when $\mathcal{O}_D$ is queried on an input $(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ such that there is a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$, $\mathtt{vk} \neq \mathtt{vk}^*$, $c' = c'^*$, and $B_j \in \vec{V}_\beta$, $\mathcal{O}_D$ works as follows: let

$$(\mathtt{spk}_{i,\beta}, \vec{v_\beta}_{\mathrm{MDVS}}, m_\beta, \sigma)$$

be the plaintext that was encrypted by $\Pi_{\mathrm{PKEBC}}.E$ under $\vec{v_\beta}_{\mathrm{PKEBC}}$ (which resulted in ciphertext c'), where $\mathtt{spk}_{i,\beta}$ is $A_{i,\beta}$'s public key,

$$\begin{aligned}\vec{v_\beta}_{\mathrm{MDVS}} &\coloneqq (\mathtt{vpk}_{\mathrm{MDVS}1,\beta}, \ldots, \mathtt{vpk}_{\mathrm{MDVS}|\vec{v}|,\beta}), \text{ and}\\ \vec{v_\beta}_{\mathrm{PKEBC}} &\coloneqq (\mathtt{pk}_{\mathrm{PKEBC}1,\beta}, \ldots, \mathtt{pk}_{\mathrm{PKEBC}|\vec{v}|,\beta})\end{aligned}$$

are, respectively, the vectors of public MDVS verifier keys and public PKEBC receiver keys corresponding to $\vec{V}_\beta$, and where

$$\sigma \leftarrow \Pi_{\mathrm{MDVS}}.Sig_{\mathsf{pp}_{\mathrm{MDVS}}}(\mathtt{ssk}_{\mathrm{MDVS}i,\beta}, \vec{v_\beta}_{\mathrm{MDVS}}, (\vec{v_\beta}_{\mathrm{PKEBC}}, m_\beta, \mathtt{vk})),$$

is an MDVS signature on $(\vec{v_\beta}_{\mathrm{PKEBC}}, m_\beta, \mathtt{vk})$, with $\mathtt{ssk}_{\mathrm{MDVS}i,\beta}$ being $A_{i,\beta}$'s secret MDVS signing key and $\mathtt{vk}$ being the DSS verification key in c; if the signature σ' verifies as valid for c' under verification key $\mathtt{vk}$, oracle $\mathcal{O}_D$ no longer decrypts c' using $\Pi_{\mathrm{PKEBC}}.D$ with B_j's PKEBC secret key, and instead simply assumes decryption outputs

$$(\vec{v_\beta}_{\mathrm{PKEBC}}, (\mathtt{spk}_{i,\beta}, \vec{v_\beta}_{\mathrm{MDVS}}, m_\beta, \sigma)).$$

One can reduce distinguishing the two games to breaking the correctness of Π_{PKEBC}: since the reduction holds all MDVS and DSS secret keys and has access to all PKEBC secret keys (through the $\mathcal{O}_{SK}$ oracle of Π_{PKEBC}'s Correctness game), it can handle any oracle queries; for each $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ query, the reduction makes a query $\mathcal{O}_E(A_{i,\beta}, \vec{V}_\beta, m_\beta)$ to Π_{PKEBC}'s Correctness game. If $\mathbf{A}$ only queries for at most $n_R \leq n_{\mathrm{PKEBC}}$ different receivers, the sum of lengths of the vectors input to $\mathcal{O}_E$ is at most $d_E \leq d_{E\mathrm{PKEBC}}$, and $\mathbf{A}$ makes at most $q_E \leq q_{E\mathrm{PKEBC}}$ and $q_D \leq q_{D\mathrm{PKEBC}}$ queries to oracles $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, since from Equation B.24 no adversary $(t_{\mathrm{PKEBC}}, \varepsilon_{\mathrm{PKEBC\text{-}Corr}})$-breaks the $(n_{\mathrm{PKEBC}}, d_{E\mathrm{PKEBC}}, q_{E\mathrm{PKEBC}}, q_{D\mathrm{PKEBC}})$-Correctness of Π_{PKEBC}, we have

$$\Big|\Pr[\mathbf{AG}_\beta^3 = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^2 = \mathtt{win}]\Big| \leq \varepsilon_{\mathrm{PKEBC\text{-}Corr}}.$$

$\mathbf{G}_\beta^3 \rightsquigarrow \mathbf{G}_\beta^4$**:** In $\mathbf{G}_\beta^4$ some decryption queries are handled differently: when $\mathcal{O}_D$ is queried on an input $(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ such that there is a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$, $c' = c'^*$, $\mathtt{vk} \neq \mathtt{vk}^*$, and $B_j \in \vec{V}_\beta$, $\mathcal{O}_D$ simply outputs $\perp$. Before moving to showing that

$$\Big|\Pr[\mathbf{AG}_\beta^4 = \mathtt{win}] - \Pr[\mathbf{AG}_\beta^3 = \mathtt{win}]\Big| \leq \varepsilon_{\mathrm{MDVS\text{-}Bound\text{-}Val}}$$

(by explaining why one can make a reduction to the Message-Bound Validity notion of the underlying Π_{MDVS} scheme), we want to note that, at this point, for any $\mathcal{O}_D(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ query the adversary makes such that there is a query to $\mathcal{O}_E$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$ and $c' = c'^*$, $\mathcal{O}_D$ simply outputs $\perp$ (i.e. we are essentially set to make the reduction to the $(\mathsf{IND} + \mathsf{IK})\text{-}\mathsf{CCA}\text{-}2^{\mathsf{adap}}$ security of the underlying Π_{PKEBC} scheme).

$\mathbf{G}^3_\beta$ already outputs $\perp$ for any query $\mathcal{O}_D(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ such that:

- there was a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$, $c' = c'^*$ and $\mathtt{vk} = \mathtt{vk}^*$ (see $\mathbf{G}_\beta^{(\mathsf{IND+IK})\text{-}\mathsf{CCA}\text{-}2} \rightsquigarrow \mathbf{G}^1_\beta$); or
- there was a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$, $c' = c'^*$, $\mathtt{vk} \neq \mathtt{vk}^*$ and $B_j \notin \vec{V}_\beta$ (see $\mathbf{G}^1_\beta \rightsquigarrow \mathbf{G}^2_\beta$).

Thus we only have to make sure that we can use the adversary to break the message-bound validity of Π_{MDVS} for the remainder of the queries, i.e. queries to $\mathcal{O}_D(B_j, c \coloneqq (\mathtt{vk}, \sigma', c'))$ such that:

1. there is no query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$, $c' = c'^*$ and $\mathtt{vk} = \mathtt{vk}^*$; and
2. there is no query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$, $c' = c'^*$, $\mathtt{vk} \neq \mathtt{vk}^*$ and $B_j \notin \vec{V}_\beta$; and
3. there is a query $\mathcal{O}_E\big((A_{i,0}, \vec{V}_0, m_0), (A_{i,1}, \vec{V}_1, m_1)\big)$ that output $c^* \coloneqq (\mathtt{vk}^*, \sigma'^*, c'^*)$ with $c \neq c^*$, $c' = c'^*$, $\mathtt{vk} \neq \mathtt{vk}^*$ and $B_j \in \vec{V}_\beta$.

Note that since $B_j \in \vec{V}_\beta$ and because we are assuming the correctness of Π_{PKEBC}, the reduction does not need to attempt to decrypt c', and instead can simply assume that the decryption is

$$(\vec{v_\beta}_{\mathrm{PKEBC}}, (\mathtt{spk}_{i,\beta}, \vec{v_\beta}_{\mathrm{MDVS}}, m_\beta, \sigma)),$$

as explained in step $\mathbf{G}^2_\beta \rightsquigarrow \mathbf{G}^3_\beta$—this is necessary because we need the MDVS keys the reduction obtains from the decryption of c' to match the ones from the underlying $\mathbf{G}^{\mathsf{Bound\text{-}Val}}$, as otherwise we cannot win Π_{MDVS}'s message-bound validity game;[11] on the other hand we are also assuming there was no query to $\mathcal{O}_E$ that output the same verification key $\mathtt{vk}$. Since the Π_{DSS} verification key is part of the messages that are signed and/or verified using Π_{MDVS}, then if σ verifies as being a valid signature on $(\vec{v_\beta}_{\mathrm{PKEBC}}, m_\beta, \mathtt{vk})$ with respect to sender public key $\mathtt{spk}_{i,\beta}$ and vector of verifier public keys $\vec{v_\beta}_{\mathrm{MDVS}}$ using B_j's secret verification key $\mathtt{rsk}_j.\mathtt{vsk}$, the reduction wins the $\mathsf{Bound\text{-}Val}$ game of Π_{MDVS} (see Definition 5). Finally, if $\mathbf{A}$ only queries for at most $n_S \leq n_{S\,\mathrm{MDVS}}$ (resp. $n_R \leq n_{V\,\mathrm{MDVS}}$) different sender keys (resp. different receiver keys), makes up to $q_E \leq q_{S\,\mathrm{MDVS}}$ queries to $\mathcal{O}_E$ and up to $q_D \leq q_{V\,\mathrm{MDVS}}$ queries to $\mathcal{O}_D$, and the

[11] Recall that an adversary can only win $\mathbf{G}^{\mathsf{Bound\text{-}Val}}$ if it makes a query to $\mathcal{O}_V$ on an input $(A_i, B_j, \vec{V}, m, \sigma)$ where A_i and B_j are parties and where $\vec{V}$ is a vector of parties.

sum of lengths of the party vectors input to $\mathcal{O}_E$ is at most $d_E \leq d_{S\text{MDVS}}$, since from Equation B.25 no adversary $(\varepsilon_{\text{MDVS-}\mathsf{Bound\text{-}Val}}, t_{\text{MDVS}})$-breaks the

$$(n_{S\text{MDVS}}, n_{V\text{MDVS}}, d_{S\text{MDVS}}, q_{S\text{MDVS}}, q_{V\text{MDVS}})\text{-Message-Bound Validity}$$

of Π_{MDVS}, it follows

$$\left|\Pr[\mathbf{AG}^4_\beta = \mathtt{win}] - \Pr[\mathbf{AG}^3_\beta = \mathtt{win}]\right| \leq \varepsilon_{\text{MDVS-}\mathsf{Bound\text{-}Val}}.$$

$\mathbf{G}^4_\beta \rightsquigarrow \mathbf{G}^5_\beta$: The only difference between these games is that in $\mathbf{G}^5_\beta$ the distribution of PKEBC key-pairs returns to being the distribution induced by Π_{PKEBC}'s G algorithm. (This step is necessary to allow us to reduce an adversary distinguishing $\mathbf{G}^5_\mathbf{0}$ and $\mathbf{G}^5_\mathbf{1}$ to one distinguishing Π_{PKEBC}'s $\mathbf{G}_\mathbf{0}^{\mathsf{(IND+IK)\text{-}CCA\text{-}2}}$ and $\mathbf{G}_\mathbf{1}^{\mathsf{(IND+IK)\text{-}CCA\text{-}2}}$ games.) It then follows

$$\left|\Pr[\mathbf{AG}^5_\beta = \mathtt{win}] - \Pr[\mathbf{AG}^4_\beta = \mathtt{win}]\right| \leq \varepsilon_{\text{PKEBC-}\mathsf{Corr}}.$$

Finally, we can now reduce an adversary distinguishing $\mathbf{G}^5_\mathbf{0}$ and $\mathbf{G}^5_\mathbf{1}$ to one distinguishing the $\mathbf{G}_\mathbf{0}^{\mathsf{(IND+IK)\text{-}CCA\text{-}2}}$ and $\mathbf{G}_\mathbf{1}^{\mathsf{(IND+IK)\text{-}CCA\text{-}2}}$ games of the underlying Π_{PKEBC} scheme because we can handle all decryption queries—either by using the $\mathcal{O}_D$ oracle of Π_{PKEBC}'s $\mathsf{(IND + IK)\text{-}CCA\text{-}2^{adap}}$ games, or by outputting $\bot$. So, if $\mathbf{A}$ only queries for at most $n_R \leq n_{\text{PKEBC}}$ different receivers, the sum of lengths of the vectors input to $\mathcal{O}_E$ is at most $d_E \leq d_{E\text{PKEBC}}$, and $\mathbf{A}$ makes at most $q_E \leq q_{E\text{PKEBC}}$ and $q_D \leq q_{D\text{PKEBC}}$ queries to oracles $\mathcal{O}_E$ and $\mathcal{O}_D$, respectively, since from Equation B.24 that no adversary $(t_{\text{PKEBC}}, \varepsilon_{\text{PKEBC-}\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}})$-breaks the

$$(n_{\text{PKEBC}}, d_{E\text{PKEBC}}, q_{E\text{PKEBC}}, q_{D\text{PKEBC}})\text{-}\mathsf{(IND + IK)\text{-}CCA\text{-}2^{adap}}\text{ security}$$

of Π_{PKEBC}, it follows

$$\left|\Pr[\mathbf{AG}^5_\mathbf{0} = \mathtt{win}] - \Pr[\mathbf{AG}^5_\mathbf{1} = \mathtt{win}]\right| \leq \varepsilon_{\text{PKEBC-}\mathsf{(IND+IK)\text{-}CCA\text{-}2^{adap}}}.$$

□

C Gaps in Security Proofs of Prior Work

In [8], Damgård et al. introduce an intermediate type of scheme, Provably Simulatable Designated Verifier Signature (PSDVS) schemes, from which they then construct a full-fledged MDVS scheme (see [8, Construction 1]). In this section we provide details on two proof gaps: one in the proof of [8, Theorem 2]—the theorem establishing the security of their PSDVS-based MDVS construction—and one in the proof of [8, Theorem 3]—the theorem establishing the security of their PSDVS construction based on standard primitives.

Issue with Consistency Proof of MDVS *Construction from Standard Primitives.* The Consistency security notion given in [8, Definition 2] provides adversaries with access to a signature verification oracle. Unfortunately, in the proof of [8, Theorem 2]—the security proof establishing the security of their MDVS scheme construction from PSDVS schemes (see [8, Construction 1])—it is not mentioned how the reduction could handle signature verification queries. Furthermore, it is also not clear how such queries could be handled, as the security notions for the PSDVS scheme on which the consistency proof relies (see [8, Definitions 10, 13 and 15]) do not themselves provide an adversary with access to a verification oracle either. Finally, we would like to note that it is desirable to provide the adversary with access to such an oracle since signatures are not publicly verifiable; in particular, the composable treatment of MDVS schemes given in [19] requires the consistency game to provide access to a signature verification oracle.

Issue with Verifier Signature Simulation Indistinguishability Proof of the PSDVS *Construction from Standard Primitives.* In the proof of [8, Theorem 3]—which establishes the security of [8, Construction 2], the PSDVS construction from standard primitives—and in particular in paragraph "VerSigSim Indistinguishability, (Definition 11)", it is argued that verifier simulated signatures are indistinguishable from real signatures generated by the signer due to the pseudorandomness of the PRF underlying their construction. Unfortunately, the PRF's secret seed is part of the verifier's secret key, and, according to [8, Definition 11], the adversary has access to the verifier's secret key, making it unclear how one could actually make a reduction to the pseudorandomness of the underlying PRF.