Strong Post-Compromise Secure Proxy Re-Encryption

Alex Davidson[1,2,†], Amit Deo[1,†], Ela Lee[1,†,*], and Keith Martin[1]

[1]ISG, Royal Holloway University of London, UK
[2]Cloudflare, London

Abstract. Proxy Re-Encryption (PRE), introduced by Blaze et. al in [BBS98], allows a ciphertext encrypted using a key pk_i to be re-encrypted by a third party so that it is an encryption of the same message under a new key pk_j, without revealing the message. Post-Compromise Security (PCS) was first introduced for messaging protocols, and ensures that a ciphertext remains confidential even when past keys have been corrupted. We define PCS in the context of PRE, which ensures that an adversary cannot distinguish which ciphertext a re-encryption was created from even given the old secret key, potential old ciphertexts and update token used to perform the re-encryption. We argue that this formal notion accurately captures the most intuitive form of PCS. We give separating examples demonstrating how our definition is stronger than existing ones, before showing that PCS can be met using a combination of existing security definitions from the literature. In doing so, we show that there are existing PRE schemes that satisfy PCS. We also show that natural modifications of more practical PRE schemes can be shown to have PCS without relying on this combination of existing security definitions. Finally, we discuss the relationship between PCS with selective versus adaptive key corruptions, giving a theorem that shows how adaptive security can be met for certain re-encryption graphs.

1 Introduction

Cloud storage has become increasingly popular in recent years, evolving from acting as a source of backup data to becoming the default storage for many applications and systems. For example, popular media streaming platforms such as Netflix and Spotify allow clients to subscribe to on-demand access for media files as opposed to storing them locally. This has increased the number for devices which no longer need much in-built storage as long as they have an internet connection.

Since the cloud is usually a third party, to ensure confidentiality clients should encrypt their files. This poses problems when a client wants to change the key for their encrypted files as a means of satisfying compliance directives or to enforce access control policies. For the former, NIST recommends regular key rotation [BBB+12], as does the Payment Card Industry Data Security Standard [PCI18] and the Open Web Application Security Project (OWASP) [OWA18]. For the latter, an organisation can choose which of its employees can access specific files by having those files encrypted under those keys. Should access need to be granted or revoked from someone, the files should be re-encrypted to a new key accordingly. One trivial solution has the client download, decrypt, encrypt using the new key, then re-upload the file. However, this can be very expensive, particularly for modern applications involving large databases, or if the client has limited processing capability.

*Contact author: `Ela.Lee.2010@live.rhul.ac.uk`

† Author was supported by the EPSRC and the UK government as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (EP/K035584/1).

The primitive of Proxy Re-Encryption (PRE), introduced by Blaze et al. [BBS98], presents a more elegant solution. In a PRE scheme, the client creates an *update token* $\Delta_{i,j}$ using the current secret key sk_i and a new public key pk_j. The server can then use this token to re-encrypt the ciphertext, transforming it into an encryption of the same message which can now be decrypted using sk_j. The most basic security notion for PRE states that the server performing the re-encryption learns nothing about the underlying message.

Post-Compromise Security (PCS). The notion of PCS was first used in [CCG16] for messaging protocols and is informally defined as follows:

Definition 1 ([CCG16]). *A protocol between Alice and Bob provides* Post-Compromise Security (PCS) *if Alice has a security guarantee about communication with Bob, even if Bob's secrets have already been compromised.*

As this definition only provides the bare intuition, the 'security guarantee' is left open so that it can be tailored to the application, which is what we will for for PCS in PRE in this work. Note that PCS differs from *forward security*, which conveys that the compromise of future states does not affect the security of past ones. PCS conveys a scheme's ability to regain security after a session or party has been compromised, which has clear applications involving revocation and key rotation (key life-cycles), where having access to the old key should not affect the security of a re-encrypted ciphertext.

Motivation for PCS PRE. One particular application of post-compromise PRE of interest is to complement PCS of messages in transit, by giving PCS security to backed-up messages stored in the cloud. The Signal Protocol for encrypting messages in transit between two devices provides strong, modern, provable security guarantees including PCS [CCG16]. However, most users expect to keep their messages when they lose their device or buy a new one; thus, popular Signal implementations such as WhatsApp back up client messages to public cloud services. Unfortunately, this backup is encrypted using a static encryption key. This means that while messages in transit have PCS, these properties are lost once messages are backed up. If an adversary compromises a device and obtains the static cloud backup key, they can retain this to compromise future messages once they are backed up.

Assuming that all message history is stored locally, the updated message history could be encrypted under a new key and re-uploaded at regular time intervals, but this will have a huge cost both in terms of computation and bandwidth, particularly as much messaging is done via smart-phones. A PRE scheme with PCS could be used instead, so that the PCS of messages in transit is extended to message backups.

1.1 Contributions

In this paper we set out the first formalisation of PCS for PRE schemes. In our model, the adversary cannot distinguish a re-encrypted ciphertext given the old key, old ciphertexts and the token used to perform the re-encryption. In other words, we view a compromise as the loss of all previous public and secret states associated with a given ciphertext, and limit the information that must remain secret to the current secret key alone. To date there is no security definition that gives the adversary the update token used in the challenge re-encryption. Our definition implies unidirectionality (meaning update tokens can only be used to re-encrypt from pk_i to pk_j and not from pk_j to pk_i) as opposed to treating unidirectional and bidirectional schemes differently, and that additional randomness beyond that given in the update token is added upon re-encryption. Since we do not make as many assumptions concerning which algorithms are deterministic or on the flow of re-encryption operations, our security model can be applied to more general PRE schemes and applications than similar definitions in the literature (see Section 3).

We analyse our model, proving several results that associate PCS with existing security models for PRE and related primitives such as updatable encryption [LT18], and provide separating examples that distinguish PCS as a separate security characteristic in its own right. One of our major contributions is to show that a PRE scheme that is both *source-hiding* and secure against *chosen plaintext attacks* also has PCS, meaning that one of the PRE schemes given by Fuchsbauer et al. [FKKP18] immediately satisfies PCS.

In particular, the only known technique used to make a scheme source-hiding also significantly reduces the correctness bound – the number of times a ciphertext can be re-encrypted and still decrypt correctly. This means the lattice-based source-hiding schemes of [FKKP18] are forced into sub-optimal parameter choices that render their assumptions much stronger (with respect to the approximation factors of solving worst-case lattice problems) and much less efficient. We therefore give a new PRE scheme, pcBV-PRE, adapted from BV-PRE – the practical RLWE-based scheme of Polyakov et al. [PRSV17]. Our new scheme is much more efficient than those given in [FKKP18] since we leverage the speed of the original construction in [PRSV17] with minimal changes. Whilst our adaptation is not source-hiding, we prove that it achieves PCS and is Indistinguishable against Chosen Plaintext Attacks (PRE-IND-CPA) (often referred to as IND-CPA security) via a tighter reduction than using the combination of properties previously mentioned, meaning this combination of properties, particularly source-hiding, is sufficient, but not necessary for achieving PCS.

Finally, we show that achieving PCS with adaptive key compromises is possible via the transformation of [FKKP18] using adaptive PRE-IND-CPA security (commonly called IND-CPA[1]) and source-hiding PRE schemes, where sub-exponential security loss is restricted to certain re-encryption graphs (particularly trees and chains).

Paper structure. We begin by reviewing necessary preliminaries in Section 2 before reviewing related work in Section 3. In Section 4 we define Post-Compromise Security (PCS) and show how our definition relates to those already in the literature. In particular, in Section 4.4 we demonstrate how a combination of properties already defined in the literature can be used to demonstrate that a PRE scheme has PCS. In Section 5, we give an explicit, efficient construction, pcBV-PRE, by modifying the lattice-based BV-PRE scheme [PRSV17] to show that our notion of PCS can be satisfied by natural extensions of current practical PRE schemes. Finally, in Section 6 we discuss the relationship between selective and adaptive security for PCS using the work of [FKKP18].

Changes w. r. t the ACISP 2019 version. *The security proofs for our construction,* pcBV-PRE, *have been updated to correct some errors. We also revert to the single-challenge definition of source-hiding presented in [FKKP18] for easier comparison, which has affected the security bound of some of our theorems but not the overall correctness of those theorems. Clear comments as to what has changed are provided in the appropriate sections.*

2 Preliminaries

In this section, we give the preliminaries for Proxy Re-Encryption (PRE), including some common security definitions and an explanation of directed re-encryption graphs. Whilst we stick to the asymmetric setting in the body of this work, we give symmetric variants

[1]We use different terminology to avoid confusion with CPA security for PKE schemes, which we refer to as Indistinguishable against Chosen Plaintext Attacks (PKE-IND-CPA)

for important definitions in Appendix A for easier comparison with related work in the symmetric setting such as updatable encryption [BLMR13,LT18] and key rotation [EPRS17].

In general, we use $y \leftarrow \mathsf{F}(x)$ to indicate that y is the output of a deterministic algorithm F with input x, we use $y \overset{\$}{\leftarrow} \mathsf{F}(x)$ to indicate that F is probabilistic (samples some randomness) and $y \overset{(\$)}{\leftarrow} \mathsf{F}(x)$ for the general case where F may or may not be deterministic.

Definition 2. *A* Proxy Re-Encryption (PRE) *scheme consists of the following algorithms:*

- $\mathsf{Setup}(1^\lambda) \rightarrow$ *params: Outputs a set of public parameters, including the message space and ciphertext space. Note that params is input to every subsequent algorithm, but we leave it out for compactness of notation. We often omit the* Setup *algorithm for the same reason.*
- $\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{sk})$*: Generates a public-private key pair.*
- $\mathsf{Enc}(\mathsf{pk}, m) \overset{\$}{\rightarrow} C$*: Encrypts a message* m *using a public key* pk*, producing a ciphertext* C.[2]
- $\mathsf{Dec}(\mathsf{sk}, C) \rightarrow m' \cup \perp$*: Decrypts a ciphertext* C *to produce either an element of the message space* m' *or an error symbol* $\perp$.
- $\mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j) \overset{(\$)}{\rightarrow} \Delta_{i,j} \cup \perp$*: Takes a secret key* sk_i *and public key* pk_j *and outputs an* update token $\Delta_{i,j}$*, or* $\perp$ *when* $i = j$*. This last condition is often left out of constructions for compactness.*
- $\mathsf{ReEnc}(\Delta_{i,j}, C) \overset{(\$)}{\rightarrow} C'$*: Takes a ciphertext* C *under* pk_i *and outputs a new ciphertext* C' *under* pk_j.

A PRE scheme is correct *if, for all* $m \in \mathcal{M}, (\mathsf{pk}, \mathsf{sk}) \overset{\$}{\leftarrow} \mathsf{KeyGen}(1^\lambda)$*, then:*

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) \rightarrow m$$

and if, for all $C \in \mathcal{C}$ *such that* $\mathsf{Dec}(\mathsf{sk}_i, C) \rightarrow m$*, then:*

$$\mathsf{Dec}(\mathsf{sk}_j, \mathsf{ReEnc}(\Delta_{i,j}, C)) \rightarrow m$$

where $(\mathsf{pk}_i, \mathsf{sk}_i), (\mathsf{pk}_j, \mathsf{sk}_j) \overset{\$}{\leftarrow} \mathsf{KeyGen}(1^\lambda)$ *and* $\Delta_{i,j} \overset{(\$)}{\leftarrow} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$.

Note that some PRE constructions have a *correctness bound* – a limit on the number of re-encryptions that are possible before the resulting ciphertext fails to decrypt properly. We shall see this in Section 5.

Definition 3. *If an update token* $\Delta_{i,j} \overset{(\$)}{\leftarrow} \mathcal{PRE}.\mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$ *computed using a PRE scheme* $\mathcal{PRE}$ *can be used to derive a token* $\Delta_{j,i}$ *that can re-encrypt ciphertexts from* pk_j *to* pk_i *then we say the scheme* $\mathcal{PRE}$ *is* bidirectional*. If* $\mathcal{PRE}$ *is not bidirectional then it is* unidirectional.

Directionality is often used in security games to determine the adversary's limitations.

We note that in some existing work including [BBS98, ID03, AFGH06, LV08], new elements are added to a ciphertext when it is re-encrypted. In such schemes, decryption is different depending on whether (or how many times) a ciphertext has been re-encrypted. This motivates the following definition:

[2]Note that some definitions of a PRE scheme have an additional input ℓ to indicate a *level* the ciphertext should be at. In this work, we leave out ℓ unless discussing schemes and results that use levelling explicitly.

Definition 4. *A PRE scheme has* transparency *if ciphertexts have the same format regardless of how many times they have been re-encrypted.*

Transparency is considered advantageous since it means that decryption is the same for re-encrypted ciphertexts as it is for fresh ciphertexts.

We now move on to giving definitions for message confidentiality in PRE. Indistinguishability against Chosen Plaintext Attacks (IND-CPA) is a well-known notion in public-key encryption which states that given a ciphertext, an adversary cannot distinguish which of two messages it is an encryption of. In this work we refer to it as PKE-IND-CPA, to make the distinction between this and what is commonly referred to as IND-CPA for PRE in the literature.

```
$\mathsf{PKE\text{-}IND\text{-}CPA}^{b,\mathcal{PKE}}_{\mathcal{A}}(1^\lambda)$
    $\kappa = 0$
    $b' \overset{(\$)}{\leftarrow} \mathcal{A}_1^{\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{challenge}}}(1^\lambda)$
    return $b'$

$\mathsf{O}_{\mathsf{KeyGen}}(1^\lambda)$
    $\kappa = \kappa + 1$
    $(\mathsf{pk}_\kappa, \mathsf{sk}_\kappa) \overset{\$}{\leftarrow} \mathsf{KeyGen}(1^\lambda)$
    return $\mathsf{pk}_\kappa$

$\mathsf{O}^{\mathsf{PKE\text{-}IND\text{-}CPA}}_{\mathsf{challenge}}(i, m_0, m_1)$
    if $|m_0| \neq |m_1|$ : return $\perp$
    $C \overset{\$}{\leftarrow} \mathsf{Enc}(\mathsf{pk}_i, m_b)$
    return $C$
```

Fig. 1: The PKE-CPA game. This is essentially the same as the IND-CPA game for PKE schemes, but we make the distinction here for indistinguishability of chosen-plaintext attacks for PKE schemes and PRE schemes. We give a multi-key, multi-challenge version.

Definition 5. *A Public Key Encryption (PKE) scheme $\mathcal{PKE}$ is ϵ-*Indistinguishable against Chosen Plaintext Attacks (ϵ-PKE-IND-CPA-secure) *if for all Probabilistic Polynomial-Time (*PPT*) adversaries $\mathcal{A}$:*

$$\left|\Pr\left[\mathsf{PKE\text{-}IND\text{-}CPA}^{0,\mathcal{PKE}}_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{PKE\text{-}IND\text{-}CPA}^{1,\mathcal{PKE}}_{\mathcal{A}}(1^\lambda) = 1\right]\right| \leq \epsilon$$

where PKE-IND-CPA *is defined in Figure 1. If ϵ is negligible as parameterised by the security parameter λ, then we say the scheme is* Indistinguishable against Chosen Plaintext Attacks (PKE-IND-CPA-secure)*.*
A PRE scheme $\mathcal{PRE}$ is ϵ-PKE-IND-CPA secure if the PKE scheme given by $\mathcal{PKE} = \{\mathcal{PRE}.\mathsf{KeyGen}, \mathcal{PRE}.\mathsf{Enc}, \mathcal{PRE}.\mathsf{Dec}\}$ is ϵ-PKE-IND-CPA-secure.

2.1 Re-encryption graphs

We often use a directed re-encryption graph (DRG) when discussing the security of PRE schemes. A DRG tracks queries the adversary $\mathcal{A}$ makes during a security game to represent re-encryptions that $\mathcal{A}$ can make locally. Using update tokens, the adversary can locally re-encrypt ciphertexts. Therefore, if a challenge ciphertext is an encryption under pk_i, and there exists a sequence of tokens going from i to j, then both sk_i and sk_j are considered challenge keys. The DRG consists of nodes v_i that represent key pairs, and directed[3] edges $\overrightarrow{e}_{i,j}$ which represent re-encryptions from pk_i to pk_j. The DRG is often used to enforce the condition that $\mathcal{A}$ cannot query oracles in such a way that reveals a challenge under a corrupted key, which we call the *trivial win condition.* This is a standard condition in all PRE security definitions. If $\mathcal{DRG}$ contains a path from v_i to v_j and sk_i is a challenge key, then so is sk_j. Figure 2 gives a pictorial representation of this.

[3]If a scheme is bidirectional, then edges added would be directionless. In this work we mainly focus on unidirectional schemes.

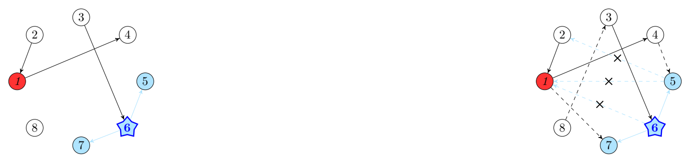

Fig. 2: An example directed re-encryption graph, $\mathcal{DRG}$. If a **challenge** is learned under pk_6 and $\mathcal{A}$ has learned tokens $\Delta_{6,5}$ and $\Delta_{6,7}$, then $\mathsf{pk}_5, \mathsf{pk}_6$ are also considered challenge keys. Therefore, token queries that lead to paths from challenge keys to corrupted ones such as $\Delta_{6,1}$ or $\Delta_{5,2}$ result in a trivial win. The graph on the right gives some examples of what tokens the adversary can and cannot learn next.

Re-encryption graphs often reflect applications. For example, for simply rotating keys the resulting graph will be a chain, as is assumed in *updatable encryption* [BLMR13, LT18] and *key rotation* [EPRS17], whereas some access control hierarchies may lead to trees. Some results such as those given in [FKKP18] between selective and adaptive security mainly apply to some types of graph. Note that some existing work assumes DRGs are acyclic.

2.2 Common oracles

Security games usually have a key generation oracle $\mathsf{O}_{\mathsf{KeyGen}}$, key corruption oracle $\mathsf{O}_{\mathsf{Corrupt}}$, update token generation oracle $\mathsf{O}_{\mathsf{ReKeyGen}}$, re-encryption oracle $\mathsf{O}_{\mathsf{ReEnc}}$, and finally a challenge oracle $\mathsf{O}_{\mathsf{challenge}}$. Recent work also gives an encryption oracle $\mathsf{O}_{\mathsf{Enc}}$, as we shall explain in Section 3.1. We give most of these oracles in Figure 3 for compactness.

The main variations between definitions are how the challenge oracle $\mathsf{O}_{\mathsf{challenge}}$ is defined, and how $\mathsf{O}_{\mathsf{ReEnc}}$ affects the DRG. We therefore define these in each individual game. Games often keep track of lists updated by the oracles, namely a list of challenge keys $\mathcal{K}_{\mathsf{chal}}$, corrupted keys $\mathcal{K}_{\mathsf{corrupted}}$, oracle-generated ciphertexts $\mathcal{C}_{\mathsf{honest}}$, challenge ciphertexts $\mathcal{C}_{\mathsf{chal}}$ and oracle-generated tokens $\mathcal{T}_{\mathsf{honest}}$.

```
$\mathsf{O}_{\mathsf{KeyGen}}(1^\lambda)$
    $\kappa = \kappa + 1$
    $(\mathsf{pk}_\kappa, \mathsf{sk}_\kappa) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
    $\boxed{\mathcal{DRG}.\mathsf{add}\{v_\kappa\}}$
    return $\mathsf{pk}_\kappa$
```

```
$\mathsf{O}_{\mathsf{Corrupt}}(i)$
    $\mathcal{K}_{\mathsf{corrupted}}.\mathsf{add}\{\mathsf{sk}_i\}$
    return $\mathsf{sk}_i$
```

```
$\mathsf{O}_{\mathsf{Enc}}(i, m)$
    $C \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, m)$
    $\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{(i, C)\}$
    return $C$
```

```
$\mathsf{O}_{\mathsf{ReKeyGen}}(i, j)$
    $\Delta_{i,j} \xleftarrow{(\$)} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
    $\mathcal{T}_{\mathsf{honest}}.\mathsf{add}\{i, j, \Delta_{i,j}\}$
    $\boxed{\mathcal{DRG}.\mathsf{add}\{\vec{e}_{i,j}\}}$
    return $\Delta_{i,j}$
```

Fig. 3: Common oracles used in security games for PRE, where κ is the number of keys in the game and $\boxed{\text{boxed values}}$ indicate steps to update lists that may not be used depending on the game. The lists a particular game uses are indicated in the game's setup phase.

$\mathsf{O}_{\mathsf{KeyGen}}(1^\lambda)$ increments the number of keys κ, generates a new key pair and adds a new vertex to $\mathcal{DRG}$ to represent this key pair before returning the public key. $\mathsf{O}_{\mathsf{Corrupt}}(i)$ adds sk_i to the set of corrupted keys $\mathcal{K}_{\mathsf{corrupted}}$ and returns it to the adversary. $\mathsf{O}_{\mathsf{Enc}}(i, m)$ encrypts the message under pk_i, appends the list of honestly-created ciphertexts $\mathcal{C}_{\mathsf{honest}}$ with (i, C) and returns the ciphertext. $\mathsf{O}_{\mathsf{ReKeyGen}}(i, j)$ creates the update token, appends the list of honestly-generated update tokens $\mathcal{T}_{\mathsf{honest}}$ with $(i, j, \Delta_{i,j})$, adds the edge $\vec{e}_{i,j}$ to $\mathcal{DRG}$ and returns the update token.

In our syntax, the restrictions on what tokens the adversary is allowed to learn is not enforced by oracles (as in other work), but instead by the list of challenge keys $\mathcal{K}_{\mathsf{chal}}$ being updated using the graph $\mathcal{DRG}$ after the adversary has output its guess. Since the adversary can locally re-encrypt the challenge ciphertext using any relevant update tokens it has learned, we therefore use the following function to update the set of challenge keys:

```
UpdateChallengeKeys($\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG}$)
$\forall i$ such that $\mathsf{sk}_i \in \mathcal{K}_{\mathsf{chal}}$ :
  $\forall j$ such that $\exists$ a path from $v_i$ to $v_j$ in $\mathcal{DRG}$ :
    $\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\mathsf{sk}_j\}$
return $\mathcal{K}_{\mathsf{chal}}$
```

We enforce the trivial win condition by calling UpdateChallengeKeys at the end of the game, and checking that no challenge keys have been corrupted ($\mathcal{K}_{\mathsf{chal}} \cap \mathcal{K}_{\mathsf{corrupted}} = \emptyset$).

3 Related work

We begin by stating definitions of PRE schemes that imply confidentiality in Section 3.1, before stating other properties of PRE schemes already defined in the literature that are relevant to our work in Section 3.2.

3.1 Confidentiality definitions

The basic security definition for PRE was first given in [BBS98] for bidirectional PRE schemes. Informally, it states the scheme should still be PKE-IND-CPA-secure when given the additional functionality of re-encryption. This means the proxy should not learn the message during the re-encryption process. Unidirectional PRE schemes were introduced by Ateniese et al. [AFGH06] together with an equivalent security definition. Similar definitions conveying this notion appear in all work on PRE. We refer to such notions as PRE-IND-CPA.

Definition 6. *A PRE scheme $\mathcal{PRE}$ is said to be* (selectively) ϵ-Indistinguishable against Chosen Plaintext Attacks-secure (ϵ-PRE-IND-CPA-secure) *if for all* PPT *adversaries* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$*:*

$$\left|\Pr\left[\mathsf{PRE\text{-}IND\text{-}CPA}^0_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{PRE\text{-}IND\text{-}CPA}^1_{\mathcal{A}}(1^\lambda) = 1\right]\right| \le \epsilon,$$

where PRE-IND-CPA *is defined in Figure 4.*

If ϵ is negligible as parameterised by the security parameter, then we say the scheme is (selectively) Indistinguishable against Chosen Plaintext Attacks (PRE-IND-CPA-secure)*.*

Whilst Definition 6 is based on the one given in [FKKP18], our formulation is slightly different as we account for there being multiple possible tokens per key pair, meaning $\mathsf{O}_{\mathsf{ReEnc}}$ allows $\mathcal{A}$ to input an honestly-generated update token as opposed to only having indexes as input. Note that the $\mathcal{DRG}$ is created by adding an edge whenever $\mathsf{O}_{\mathsf{ReEnc}}$ is called.

It was an open problem for many years to create a PRE scheme which is both unidirectional and multi-hop (can be re-encrypted more than once), with single-hop (can only be re-encrypted once) constructions emerging as the means of providing unidirectionality in the meantime. Multi-hop schemes are necessary for both key rotation and dynamic access control. Unidirectional schemes are necessary for applications where the trust relationship

```
$\mathsf{PRE\text{-}IND\text{-}CPA}_{\mathcal{A}}^{b,\mathcal{PRE}}(1^\lambda)$
$\mathcal{K}_{\mathsf{chal}}, \mathcal{K}_{\mathsf{corrupted}}, \mathcal{T}_{\mathsf{honest}}, \mathcal{DRG} = \emptyset$
$\kappa = 0, \mathsf{called} = \mathsf{false}$
$state \overset{(\$)}{\leftarrow} \mathcal{A}_0^{\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Corrupt}}}(1^\lambda)$
$b' \overset{(\$)}{\leftarrow} \mathcal{A}_1^{\mathsf{O}_{\mathsf{Enc}}, \mathsf{O}_{\mathsf{ReKeyGen}}, \mathsf{O}_{\mathsf{ReEnc}}^{\mathsf{PRE\text{-}IND\text{-}CPA}}, \mathsf{O}_{\mathsf{challenge}}^{\mathsf{PRE\text{-}IND\text{-}CPA}}}(1^\lambda, state)$
$\mathcal{K}_{\mathsf{chal}} \leftarrow \mathsf{UpdateChallengeKeys}(\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG})$
if $\mathcal{K}_{\mathsf{chal}} \cap \mathcal{K}_{\mathsf{corrupted}} \neq \emptyset$ : return $0$
return $b'$
```

```
$\mathsf{O}_{\mathsf{ReEnc}}^{\mathsf{PRE\text{-}IND\text{-}CPA}}(C, i, j, [\Delta_{i,j}])$
if $\Delta_{i,j}$ given AND $(i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}}$ :
  return $\perp$
if $\Delta_{i,j}$ not given :
  $\Delta_{i,j} \overset{(\$)}{\leftarrow} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
$\mathcal{DRG}.\mathsf{add}\{\overrightarrow{e}_{i,j}\}$
$C' \overset{(\$)}{\leftarrow} \mathsf{ReEnc}(\Delta_{i,j}, C)$
return $C'$
```

```
$\mathsf{O}_{\mathsf{challenge}}^{\mathsf{PRE\text{-}IND\text{-}CPA}}(i, m_0, m_1)$
if $\mathsf{called} = \mathsf{true}$ OR $|m_0| \neq |m_1|$ :
  return $\perp$
$C \overset{\$}{\leftarrow} \mathsf{Enc}(\mathsf{pk}_i, m_b)$
$\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\mathsf{sk}_i\}$
$\mathsf{called} \leftarrow \mathsf{true}$
return $C$
```

Fig. 4: The PRE-IND-CPA game – an extension of PKE-IND-CPA which accounts for re-encryption. $\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Enc}}, \mathsf{O}_{\mathsf{ReKeyGen}}$ are as defined in Figure 3. We note that an equivalent way to enforce the trivial win condition is for $\perp$ to be returned whenever $\mathsf{O}_{\mathsf{ReKeyGen}}$ or $\mathsf{O}_{\mathsf{ReEnc}}$ are called from an uncorrupted key sk_i to a corrupted key sk_j.

between Alice and Bob is not symmetrical, such as if Bob is more senior to Alice and therefore has access to more sensitive files. Whilst most existing literature considers directionality as a class of PRE schemes, it was been defined explicitly as a formal security definition in [Lee17]. The first constructions for PRE-IND-CPA secure unidirectional, multi-hop PRE schemes were given using program obfuscation [HRSV07,CCV12], until Gentry [Gen09] gave a generic construction using Fully Homomorphic Encryption (FHE).

Much of the research on PRE in recent years has focused on CCA security. A definition of IND-CCA security for PRE first appears in [CH07] for bidirectional single-hop (ciphertexts can only be re-encrypted once) schemes. This allows the adversary to adaptively corrupt secret keys. A definition of IND-CCA security for unidirectional schemes is given in [LV08].

Honest Re-encryption Attacks. Recently, a stronger notion than PRE-IND-CPA security has been introduced which allows the adversary to receive re-encryptions of non-challenge ciphertexts from uncorrupted to corrupted keys, as long as the ciphertexts were honestly generated. Cohen formalised these as *Honest Re-encryption Attacks (HRA)* [Coh17] but the same idea is also used elsewhere [LT18]. This motivates the concept of a PRE scheme being Indistinguishable against Honest Re-encryption Attacks (IND-HRA) [Coh17]. We base our formulation on IND-ENC-security [LT18], HRA-security [Coh17] and IND-CPA-security [PRSV17].

Definition 7. *A PRE scheme $\mathcal{PRE}$ is said to be* (selectively) ϵ-Indistinguishable against Honest Re-encryption Attacks (ϵ-IND-HRA-secure) *if for all* PPT *adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$:*

$$\left|\Pr\left[\mathsf{IND\text{-}HRA}_{\mathcal{A}}^{0}(1^\lambda) = 1\right] - \Pr\left[\mathsf{IND\text{-}HRA}_{\mathcal{A}}^{1}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

where $\mathsf{IND\text{-}HRA}_{\mathcal{A}}^{b,\mathcal{PRE}}$ is defined in Figure 5.

If ϵ is negligible as parameterised by the security parameter, then we say the scheme is (selectively) Indistinguishable against Honest Re-encryption Attacks (IND-HRA-secure).

```
$\mathsf{IND\text{-}HRA}^{b,\mathcal{PRE}}_{\mathcal{A}}(1^\lambda)$
$\mathcal{K}_{\mathsf{chal}}, \mathcal{K}_{\mathsf{corrupted}}, \mathcal{C}_{\mathsf{honest}}, \mathcal{C}_{\mathsf{chal}}, \mathcal{T}_{\mathsf{honest}}, \mathcal{DRG} = \emptyset$
$\kappa = 0, \mathsf{called} = \mathsf{false}$
$state \overset{(\$)}{\leftarrow} \mathcal{A}_0^{\mathsf{O_{KeyGen}}, \mathsf{O_{Corrupt}}}(1^\lambda)$
$b' \overset{(\$)}{\leftarrow} \mathcal{A}_1^{\mathsf{O_{Enc}}, \mathsf{O_{ReKeyGen}}, \mathsf{O^{IND\text{-}HRA}_{ReEnc}}, \mathsf{O^{IND\text{-}HRA}_{challenge}}}(1^\lambda, state)$
$\mathcal{K}_{\mathsf{chal}} \leftarrow \mathsf{UpdateChallengeKeys}(\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG})$
if $\mathcal{K}_{\mathsf{chal}} \cap \mathcal{K}_{\mathsf{corrupted}} \neq \emptyset$ : return $0$
return $b'$
```

```
$\mathsf{O^{IND\text{-}HRA}_{ReEnc}}(C, i, j, [\Delta_{i,j}])$
if $(i, C) \notin \mathcal{C}_{\mathsf{honest}}$ :
  return $\perp$
if $\Delta_{i,j}$ given AND $(i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}}$ :
  return $\perp$
if $\Delta_{i,j}$ not given :
  $\Delta_{i,j} \overset{(\$)}{\leftarrow} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
$C' \overset{(\$)}{\leftarrow} \mathsf{ReEnc}(\Delta_{i,j}, C)$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{j, C'\}$
if $(i, C) \in \mathcal{C}_{\mathsf{chal}}$ :
  $\mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{j, C'\}, \mathcal{DRG}.\mathsf{add}\{\overrightarrow{e}_{i,j}\}$
return $C'$
```

```
$\mathsf{O^{IND\text{-}HRA}_{challenge}}(i, m_0, m_1)$
if $|m_0| \neq |m_1|$ OR $\mathsf{called} = \mathsf{true}$ :
  return $\perp$
$C \overset{\$}{\leftarrow} \mathsf{Enc}(\mathsf{pk}_i, m_b)$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{i, C\}$
$\mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{i, C\}$   $\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\mathsf{sk}_i\}$
$\mathsf{called} \leftarrow \mathsf{true}$
return $C$
```

Fig. 5: The IND-HRA game [Coh17], which allows re-encryptions of non-challenge ciphertexts from uncorrupted to corrupted keys using $\mathsf{O_{ReEnc}}$.

We discuss security with respect to adaptive key corruptions in Section 6.

Theorem 1. *IND-HRA $\implies$ PRE-IND-CPA $\implies$ PKE-IND-CPA.*

As each game builds directly on the last but giving the adversary access to more information, the proof of this theorem follows trivially.

3.2 Re-encryption Simulatability and Source-hiding

There are a few properties that a PRE scheme can have that lift the scheme from being CPA secure to being HRA secure. The goal is to demonstrate that it is possible to replace re-encryptions from uncorrupted keys to corrupted ones with a value that is created independently of the old secret key. This enables an adversary $\mathcal{A}^{\mathsf{cpa}}$ in the CPA game to simulate the HRA game, meaning they can run an adversary $\mathcal{A}^{\mathsf{hra}}$ in the HRA game and return the same output to win the CPA game with the same advantage. One such property that was defined by Cohen is *re-encryption simulatability.*

Definition 8. *A PRE scheme $\mathcal{PRE}$ is* re-encryption simulatable *if there exists a probabilistic, polynomial-time algorithm* ReEncSim *such that with high probability over* aux*, for all $m \in \mathcal{M}$:*

$$(\mathsf{ReEncSim}(\mathsf{pk}_i, \mathsf{pk}_j, \mathsf{sk}_j, C, m), \mathsf{aux}) \approx_s (\mathsf{ReEnc}(\Delta_{i,j}, C), \mathsf{aux})$$

where $\approx_s$ denotes statistical indistinguishability, and values are sampled according to

$$\begin{aligned}
&(\mathsf{pk}_i, \mathsf{sk}_i) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)\\
&(\mathsf{pk}_j, \mathsf{sk}_j) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)\\
&\Delta_{i,j} \xleftarrow{(\$)} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)\\
&C \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, m)\\
&\mathsf{aux} = (\mathsf{pk}_i, \mathsf{pk}_j, \mathsf{sk}_j, C, m).
\end{aligned}$$

Cohen demonstrates that PRE-IND-CPA-secure schemes which have this property are IND-HRA-secure [Coh17, Theorem 5].

Fuchsbauer et al. [FKKP18] expand on Cohen's work in defining *source-hiding*. Informally, in a source-hiding scheme a fresh encryption of a message is indistinguishable from a re-encrypted ciphertext that is an encryption of the same message even given both key pairs, the old ciphertext and the update token. This means re-encrypted ciphertexts reveal no history as to the keys they were previously encrypted under, or similarities between components of previous ciphertexts. We give a formal description of the game defining the source-hiding property in Figure 6.

We note that we have changed this definition from the one given our original publication, which was a multi-key, multi-challenge version of the definition of source-hiding presented in [FKKP18]. We now use the same definition as in [FKKP18] to avoid confusion and enable easier comparison. Proofs have been updated accordingly.

```
$\mathsf{SH}_{\mathcal{A}}^{b,\mathcal{PRE}}(1^\lambda)$
$(\mathsf{pk}_0, \mathsf{sk}_0) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$(\mathsf{pk}_1, \mathsf{sk}_1) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$\Delta_{0,1} \xleftarrow{(\$)} \mathsf{ReKeyGen}(\mathsf{sk}_0, \mathsf{pk}_1)$
$b' \xleftarrow{(\$)} \mathcal{A}^{\mathsf{O}^{\mathsf{SH}}_{\mathsf{challenge}}}(1^\lambda, (\mathsf{pk}_0, \mathsf{sk}_0), (\mathsf{pk}_1, \mathsf{sk}_1), \Delta_{0,1})$
return $b'$
```

```
$\mathsf{O}^{\mathsf{SH}}_{\mathsf{challenge}}(m^*, \ell^*)$
if $\ell^* + 1 > L]$ :
  return $\perp$
$C^* \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_0, m^*, \ell^*)$
$C'^{(0)} \xleftarrow{(\$)} \mathsf{ReEnc}(\Delta_{0,1}, C^*)$
$C'^{(1)} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_1, m^*, \ell^* + 1)$
return $(C^*, C'^{(b)})$
```

Fig. 6: Experiments for the source-hiding property. Here, ℓ denotes a *level* for the ciphertext to be encrypted at – essentially the number of times C has been re-encrypted. This is important for noisy PRE schemes, but ignored for PRE schemes without levelling. L is the number of times a ciphertext can be re-encrypted without breaking the correctness conditions (the correctness bound).

Definition 9. *A PRE scheme $\mathcal{PRE}$ is said to be* ϵ-source-hiding *if for all* PPT *adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$:*

$$\left|\Pr\left[\mathsf{SH}_{\mathcal{A}}^{0,\mathcal{PRE}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{SH}_{\mathcal{A}}^{1,\mathcal{PRE}}(1^\lambda) = 1\right]\right| \leq \epsilon$$

where $\mathsf{SH}_{\mathcal{A}}^{b,\mathcal{PRE}}$ is defined in Figure 6.

If ϵ is negligible as parameterised by the security parameter, then we say the scheme is source-hiding (SH).

The purpose of source-hiding in [FKKP18] is to relate selective and adaptive security. Fuchsbauer et al. also define *weak key privacy* and show how this together with source-hiding can reduce security against adaptive HRAs to security against CPA, with sub-exponential loss for some re-encryption graphs. In particular, they show quasi-polynomial loss in security when lifting to adaptive corruption for trees and chains, and exponential loss otherwise. We discuss adaptive key corruptions more fully in Section 6, but note that the reason for the exponential security loss in [FKKP18] is because the definition of source-hiding is defined for only two key pairs, a single challenge, and challenges output both the original ciphertext and the potential re-encryption. This means the simulator in their proof needs to guess in advance which keys the adversary will ask for an honest re-encryption of and which ciphertext will be re-encrypted so that they can integrate the source-hiding challenge in the right place. This only works if they guess correctly, which leads to the loss in security. We shall both use and discuss this technique in Section 4.4.

3.3 Ciphertext re-randomisation

Thus far we have not considered key revocation explicitly. In this case, stronger definitions requiring re-encryption to re-randomise the ciphertext are required. To see this, consider the key encapsulation approach to PRE:

– Encrypt the message m using a symmetric encryption algorithm with key k ($C_1 \overset{\$}{\leftarrow} \mathsf{symEnc}(k, m)$).
– Encrypt k with a public key of a PRE scheme pk_i and prepend this to make a header for the encrypted message ($C_0 \overset{\$}{\leftarrow} \mathcal{PRE}.\mathsf{Enc}(\mathsf{pk}_i, k)$).
– To decrypt, first decrypt the header to retrieve k', then use k' to retrieve m' ($k' \leftarrow \mathcal{PRE}.\mathsf{Dec}(\mathsf{sk}_i, C_0), m' \leftarrow \mathsf{symDec}(k', C_1)$).
– The update token is the token from the PRE scheme ($\Delta_{i,j} \overset{(\$)}{\leftarrow} \mathcal{PRE}.\mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$).
– To re-encrypt, re-encrypt the header of the ciphertext ($C_0' \overset{(\$)}{\leftarrow} \mathcal{PRE}.\mathsf{ReKeyGen}(\Delta_{i,j}, C_0)$). C_0' is an encryption of k under pk_j.

Whilst this method grants the benefits of hybrid encryption, key-scraping attacks are possible: a malicious user simply retains the message encryption key k and can derive the message regardless of how many times the ciphertext is re-encrypted. It may be unrealistic for a malicious revoked user to download all the plaintexts due to storage constraints, as is the case for subscriber-based streaming platforms. However, as symmetric keys are typically much shorter than the plaintexts, it is more realistic that a malicious subscriber could retain the message key, k. Although constructions based on this model can be shown to meet the typical confidentiality definitions for PRE shown in Section 3.1, they are not appropriate for PCS.

There is little work on requiring re-encryption to also re-randomise. In [CH07], Canetti and Hohenberger discuss a notion for re-randomising ciphertexts (which they call *unlinkability*) in the context of creating a CCA definition. Other notions of re-randomisation exist in the related areas of *key rotation* [EPRS17] and *updatable encryption* [LT18]. These differ from proxy re-encryption in that they are symmetric schemes, updates happen sequentially from k_i to k_{i+1}, and they name full re-randomisation as a necessary security goal. The fact the schemes are symmetric gives the impression that they must be faster than asymmetric schemes. However, both constructions rely on homomorphic properties of field groups used in public-key cryptography, and to date all such schemes rely on these. It is difficult to see how to construct a scheme that re-encrypts a ciphertext that doesn't rely on such public-key-like primitives. It would therefore be wrong to assume that symmetric re-encryption primitives must be faster. Our public-key constructions should therefore not necessarily be assumed inefficient in comparison to all symmetric constructions.

[EPRS17] defines UP-REENC security for key rotation / updatable encryption schemes, which does not give the adversary the token used to create the challenge re-encryption. Their construction, $\mathsf{ReCrypt}$ allows an adversary who has learned the update token to reverse the re-encryption of the challenge ciphertext and therefore does not cover full compromise of the user who generated the update token. Other related work models PCS by giving a bound on the amount of information the adversary retains about a the ciphertext prior to re-encryption [Lee17, MS17]. Such definitions do not account for the possibility of revoked users storing parts of the original ciphertexts and colluding, and lead to more complicated, less intuitive proofs than our approach. We create a stronger definition in which schemes must be unidirectional, and the adversary knows the update token used to create the challenge re-encryption.

A particular work of note is [LT18], where Lehmann and Tackmann define PCS for *updatable encryption* schemes as IND-UPD. We give an asymmetric version of IND-UPD for comparison with our work in Appendix B, but state the main points briefly here. In the $\mathsf{pkIND\text{-}UPD}$ game, key updates happen sequentially. The challenge oracle outputs a re-encrypted ciphertext and the adversary must guess which ciphertext it is a re-encryption of. Challenge ciphertexts are updated whenever a new key is generated, but only given to the adversary if the oracle $\mathsf{O}^{\mathsf{pkIU}}_{\mathsf{LearnChal}}$ is called under a specific key (or *epoch*, in their terminology). An updatable encryption scheme can be bidirectional and still be IND-UPD, as bidirectionality invokes the additional winning condition that the adversary cannot have learned any update tokens used to create a challenge ciphertext. Another of the winning conditions given in $\mathsf{O}^{\mathsf{pkIU}}_{\mathsf{ReEnc}}$ is that when ReEnc is deterministic, the adversary cannot have re-encrypted either of the potential challenge ciphertexts $\bar{C}_0, \bar{C}_1$. Another notable condition is that the adversary cannot learn the update token going towards the key that the challenge is given under, which is enforced as a condition in $\mathsf{O}^{\mathsf{pkIU}}_{\mathsf{LearnTok}}$. We will address these points when we build our own definition in Section 4. We provide separating examples with respect to $\mathsf{pkIND\text{-}UPD}$ in Section 4.3 to demonstrate that our notion is stronger.

4 Strong PCS for PRE

In this section, we create a definition for strong PCS for public-key PRE schemes. We begin by justifying our motivation, explaining why we believe existing definitions are insufficient for PCS.

We have two main motivations for creating a new definition for PCS in the context of PRE. The first is that there is currently no definition that implies unidirectionality and ciphertext re-randomisation. We believe that the distinction whether a PRE scheme is unidirectional or bidirectional is vital in the post-compromise scenario to model the corruption of used update tokens, and hence we define PCS to explicitly mean that schemes meeting the definition must be unidirectional. The second motivation is a matter of existing definitions inherently assuming which of the algorithms in the PRE scheme are probabilistic. We explain how this introduces problems when considering a post-compromise scenario. We now go into both of these motivations in more detail.

Explicit unidirectionality. We use the most relevant definition in the existing literature to make our case. IND-UPD [LT18], defined for updatable encryption, places restrictions based on inferable information, as defined by the [LT18] notions of directionality:

– When sk_j can be derived from sk_i and $\Delta_{i,j}$ (LT-bidirectional).

– When sk_j cannot be derived from sk_i and $\Delta_{i,j}$ (LT-unidirectional)[4]

In the LT-unidirectional case, the adversary can acquire re-encryption tokens from a corrupted key to a challenge key, but not the other way around. In the LT-bidirectional case, the adversary is additionally prevented, by definition, from learning tokens from challenge keys to corrupted keys or vice versa. This means that for bidirectional schemes, the adversary queries tokens in such a way that the resulting re-encryption graphs form disjoint sub-graphs – one containing corrupted keys and the other containing challenge keys. Proving security is therefore reduced to proving that unrelated, randomly-generated keys cannot be used to infer information about an encrypted message. We consider applying the same restrictions for a definition of PCS for PRE to be too restrictive, as they go against our intuition of PCS that only the new secret key must remain secret.

Since the derivability of the new secret key is implicit information to the game, meeting the definition itself says nothing about directionality and therefore the extent to which update token compromise is of concern. We believe that it is better for directionality to be explicit as it allows practitioners to understand the security of a scheme more clearly.

Assuming probabilistic algorithms. There appear to be only two existing security definitions which explicitly consider re-randomisation [EPRS17, LT18]. The [EPRS17] definition of a *key rotation scheme* assumes that ReKeyGen is randomised but ReEnc is deterministic. This leads to a necessary condition that the update token used to create the challenge re-encryption cannot be learned by the adversary, otherwise the adversary could use it to re-encrypt the input ciphertexts locally and compare this to the challenge to win the game. The adversary is allowed to learn other tokens going from corrupted to challenge keys using an oracle $\mathsf{O}_{\mathsf{ReKeyGen}}$, but not the exact token used to perform the re-encryption. This means UP-REENC [EPRS17] does not model compromise of the update token used. For this reason, it is important that new randomness is also introduced in ReEnc to prevent trivial downgrading of the challenge ciphertext if the adversary compromises the update token used.

In the [LT18] definition of an updatable encryption scheme, the opposite assumption is made – that ReEnc is randomised and ReKeyGen is deterministic. This means that for keys $\mathsf{sk}_i, \mathsf{pk}_j$, there is only one update token $\Delta_{i,j}$. This is reflected in their IND-UPD security game (and pkIND-UPD) by having an update token $\Delta_{i-1,i}$ created when k_i is generated and later having oracles reveal tokens to the adversary. This is less fitting for schemes where ReKeyGen is randomised and there are multiple tokens per key pair[5]. More importantly, such an assumption implicitly rules out the possibility that additional randomness incorporated into the update token masks the secret keys, which is important for security as the secret key sk_i should not be derivable from the update token $\Delta_{i,j}$ and the public key pk_j. BV-PRE [PRSV17] is an example of this, where knowledge of the 'public' key 'pk_j' together with $\Delta_{i,j}$ can be used to derive sk_i. Another example is ElGamal-based symmetric PRE schemes (e.g. [BBS98, LT18]) where update tokens have the form $\Delta_{i,j} = \mathsf{sk}_j/\mathsf{sk}_i$. Clearly, given the update token, compromise of the old key leads to compromise of the new key. Introducing additional randomness into the update token not only gives a means of masking the new key, but also means that the client does not need to fully rely on the proxy to be assured of new randomness, which is more appropriate for some trust scenarios.

[4]The general understanding of unidirectionality is not so strong - the new key does not necessarily have to be derivable, but the token and old key should lead to the message being learned.

[5] Interestingly, other works such as [FKKP18] take a similar approach to the adversary learning update tokens, despite their assuming randomised tokens.

For constructions where randomness is added in both $\mathsf{ReKeyGen}$ and ReEnc, neither definition is suitable. It is therefore of interest to create a security notion for PCS which factors in the possibility that both the $\mathsf{ReKeyGen}$ and ReEnc algorithms are probabilistic.

4.1 Post-Compromise Security

We model Post-Compromise Security (PCS) using an adversary $\mathcal{A}$ who chooses two ciphertexts (whose decryption key can be known) and a re-encryption token, and receives a challenge ciphertext which is a re-encryption of one of the chosen ciphertexts created using the specified token. $\mathcal{A}$ attempts to distinguish which ciphertext was re-encrypted. This models the compromise of all key-related material prior to the challenge re-encryption.

Unlike some previous definitions, we allow $\mathcal{A}$ to corrupt the secret key with which the update token was generated. As in IND-HRA security, we also allow $\mathcal{A}$ to re-encrypt honestly (oracle)-generated non-challenge ciphertexts to corrupted keys. Challenges are obtained via the challenge oracle $\mathsf{O}^{\mathsf{PC}}_{\mathsf{challenge}}$ which will only accept as input honestly-generated ciphertexts of the same length, and honestly-generated update tokens. The challenger maintains lists to enforce this: $\mathcal{C}_{\mathsf{honest}}$ and $\mathcal{T}_{\mathsf{honest}}$ respectively.

Here we present a formal definition of PCS for general PRE. As for PRE-IND-CPA and IND-HRA security, the first stage adversary $\mathcal{A}_0$ can access key generation and key corruption oracles $\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Corrupt}}$ and the second stage adversary $\mathcal{A}_1$ can access the encryption, token generation, re-encryption and challenge oracles $\mathsf{O}_{\mathsf{Enc}}, \mathsf{O}_{\mathsf{ReKeyGen}}, \mathsf{O}^{\mathsf{PC}}_{\mathsf{ReEnc}}, \mathsf{O}^{\mathsf{PC}}_{\mathsf{challenge}}$. The challenger maintains a list of corrupted keys $\mathcal{K}_{\mathsf{corrupted}}$, honest ciphertexts $\mathcal{C}_{\mathsf{honest}}$ and challenge ciphertexts $\mathcal{C}_{\mathsf{chal}}$, and enforces the trivial win condition using a re-encryption graph $\mathcal{DRG}$ to verify that no challenge keys have been corrupted.

Definition 10. *A PRE scheme $\mathcal{PRE}$ is said to have* (selective) ϵ-Post-Compromise Security (ϵ-PCS) *if for all* PPT *adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$:*

$$\left|\Pr\left[\mathsf{PostComp}^0_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{PostComp}^1_{\mathcal{A}}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

where $\mathsf{PostComp}^{b,\mathcal{PRE}}_{\mathcal{A}}$ is defined in Figure 7.

If ϵ is negligible as parameterised by the security parameter, then we say the scheme has (selective) Post-Compromise Security (PCS)*.*

We give a definition of PCS for symmetric PRE schemes in Appendix A.

4.2 Basic observations

In Lemma 1, we show that it is necessary for PCS to have randomness incorporated into re-encryption.

Lemma 1. *No PRE scheme where* ReEnc *is deterministic has PCS.*

Proof. If ReEnc is deterministic then $\mathcal{A}$ can submit $(C_0, C_1, i, j, \Delta_{i,j})$ to $\mathsf{O}_{\mathsf{challenge}}$ to learn challenge C'^*_b. Then $\mathcal{A}$ can locally compute $C'_0 \leftarrow \mathsf{ReEnc}(\Delta_{i,j}, C_0)$ and compare this with C'^*_b – if they match then output $b' = 0$, otherwise output $b' = 1$.

Lemma 2. *PCS $\Longrightarrow$ unidirectional.*

```
$\mathsf{PostComp}^{b,\mathcal{PRE}}_{\mathcal{A}}(1^\lambda)$
$\mathcal{K}_{\mathsf{chal}}, \mathcal{K}_{\mathsf{corrupted}}, \mathcal{C}_{\mathsf{honest}}, \mathcal{C}_{\mathsf{chal}}, \mathcal{C}_{\mathsf{msg}}, \mathcal{T}_{\mathsf{honest}}, \mathcal{DRG} = \emptyset$
$\kappa = 0, \mathsf{called} = \mathsf{false}$
$state \overset{(\$)}{\leftarrow} \mathcal{A}_0^{\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Corrupt}}}(1^\lambda)$
$b' \overset{(\$)}{\leftarrow} \mathcal{A}_1^{\mathsf{O}_{\mathsf{ReKeyGen}}, \mathsf{O}_{\mathsf{Enc}}, \mathsf{O}^{\mathsf{PC}}_{\mathsf{ReEnc}}, \mathsf{O}^{\mathsf{PC}}_{\mathsf{challenge}}}(1^\lambda, state)$
$\mathcal{K}_{\mathsf{chal}} \leftarrow \mathsf{UpdateChallengeKeys}(\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG})$
if $\mathcal{K}_{\mathsf{chal}} \cap \mathcal{K}_{\mathsf{corrupted}} \neq \emptyset$ : return $0$
return $b'$
```

```
$\mathsf{O}^{\mathsf{PC}}_{\mathsf{ReEnc}}(C, i, j, [\Delta_{i,j}])$
if $\Delta_{i,j}$ given :
  if $(i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}}$ : return $\perp$
else : $\Delta_{i,j} \leftarrow \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
if $(i, C) \notin \mathcal{C}_{\mathsf{honest}}$ : return $\perp$
$C' \overset{(\$)}{\leftarrow} \mathsf{ReEnc}(\Delta_{i,j}, C)$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{j, C'\}$
$\mathcal{C}_{\mathsf{msg}}[(j, C')] = \mathcal{C}_{\mathsf{msg}}[(i, C)]$
if $(i, C) \in \mathcal{C}_{\mathsf{chal}}$ :
  $\mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{j, C'\}, \mathcal{DRG}.\mathsf{add}\{\vec{e}_{i,j}\}$
return $C'$
```

```
$\mathsf{O}^{\mathsf{PC}}_{\mathsf{challenge}}(C_0, C_1, i, j, \Delta_{i,j})$
if $|C_0| \neq |C_1|$ OR $\mathsf{called} = \mathsf{true}$ : return $\perp$
if $(i, C_0), (i, C_1) \notin \mathcal{C}_{\mathsf{honest}}$ OR $(i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}}$ : return $\perp$
$C_0'^{*} \overset{(\$)}{\leftarrow} \mathsf{ReEnc}(\Delta_{i,j}, C_0)$
$C_1'^{*} \overset{(\$)}{\leftarrow} \mathsf{ReEnc}(\Delta_{i,j}, C_1)$
$\mathcal{C}_{\mathsf{msg}}[(j, C_b'^{*})] = \mathcal{C}_{\mathsf{msg}}[(i, C_b)]$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{j, C_b'^{*}\}, \mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{j, C_b'^{*}\}, \mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\mathsf{sk}_j\}$
$\mathsf{called} \leftarrow \mathsf{true}$
return $C_b'^{*}$
```

Fig. 7: The PostComp game. This reflects full compromise of the old secret key and update token used to perform the re-encryption.

Proof. We show that if a scheme is bidirectional then it cannot have PCS. Bidirectionality implies that an update token $\Delta_{i,j}$, can be used to derive $\Delta_{j,i}$. The adversary $\mathcal{A}$ proceeds as follows: $\mathcal{A}$ first corrupts a key sk_i, then creates two ciphertexts $C_0 \overset{\$}{\leftarrow} \mathsf{O}_{\mathsf{Enc}}(i, m_0), C_1 \overset{\$}{\leftarrow} \mathsf{O}_{\mathsf{Enc}}(i, m_1)$ and update token $\Delta_{i,j} \overset{(\$)}{\leftarrow} \mathsf{O}_{\mathsf{ReKeyGen}}(i, j)$, before submitting $(C_0, C_1, i, j, \Delta_{i,j})$ to the challenge oracle. $\mathcal{A}$ then computes $\Delta_{j,i}$ from $\Delta_{i,j}$, and uses this to compute $C'' \overset{(\$)}{\leftarrow} \mathsf{ReEnc}(\Delta_{j,i}, C')$, where C' is the received challenge re-encryption, to obtain the challenge ciphertext under key pk_i, before decrypting using sk_i to win the game.

This means an easy way to disprove PCS for existing schemes is to show bidirectionality.

4.3 Separating examples

We now demonstrate the relationship between PCS and existing security notions and constructions by means of a number of separating examples.

Lemma 3. *pkIND-UPD-security $\centernot\implies$ PCS.*

Proof. Let $\mathcal{PRE}$ be a pkIND-UPD-secure PRE scheme where ReEnc is deterministic. By Lemma 1, this scheme does not have PCS.

The same can be said for IND-UPD security and PCS for symmetric re-encryption schemes as defined in Appendix A.

Lemma 4. *Let $\mathcal{PRE}$ be a PRE scheme where* $\mathsf{ReKeyGen}$ *is deterministic. If $\mathcal{PRE}$ has PCS, then it is pkIND-UPD-secure.*

We note that Lemma 4 is not an exact comparison, as pkIND-UPD-security assumes that ciphertexts are re-encrypted through every key (as is the case in the key rotation application) whereas the same is not true in security PRE[6]. However include a proof sketch to demonstrate how the two may relate.

Proof sketch. The $\mathsf{PostComp}$ adversary $\mathcal{A}$ can simulate the $\mathsf{pkIND\text{-}UPD}$ game. Before the challenge is issued, $\mathsf{O}_{\mathsf{Next}}$ can be easily simulated by generating a new key pair, corrupting the old secret key and creating an update token between the old key and the new. The adversary replaces the challenge ciphertext with the output from $\mathsf{O}^{\mathsf{PC}}_{\mathsf{challenge}}(C_0, C_1, \tilde{e}-1, \tilde{e})$. The $\mathsf{PostComp}$ adversary $\mathcal{A}_0$ must guess the remaining keys which the $\mathsf{pkIND\text{-}UPD}$ will corrupt, which will result in a sub-exponential loss of security as the challenge graph will be a chain. The simulator can update both challenge and honest ciphertexts using $\mathsf{O}_{\mathsf{ReEnc}}$, and corrupting tokens can be simulated with calls to $\mathsf{O}_{\mathsf{ReKeyGen}}$. Re-encrypting a challenge ciphertext directly to the requested key as opposed to going through all previous keys in the chain first will go unnoticed, as if the number of times a ciphertext has been re-encrypted could be detected then this could be used to win the $\mathsf{PostComp}$ game.

Lemma 5. *IND-HRA-security $\not\Longrightarrow$ PCS.*

Proof. Let $\overline{\mathcal{PRE}} = (\overline{\mathsf{KeyGen}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}}, \overline{\mathsf{ReKeyGen}}, \overline{\mathsf{ReEnc}})$ be a IND-HRA-secure PRE scheme, and let $(\mathsf{symKeyGen}, \mathsf{symEnc}, \mathsf{symDec})$ be an IND-CPA-secure symmetric encryption scheme. We now define a PRE scheme using the key encapsulation method as follows:

- $\mathsf{KeyGen}(1^\lambda) \overset{\$}{\to} (\mathsf{pk}, \mathsf{sk}) : (\mathsf{pk}, \mathsf{sk}) \overset{\$}{\leftarrow} \overline{\mathsf{KeyGen}}(1^\lambda)$
- $\mathsf{Enc}(\mathsf{pk}, m) \overset{\$}{\to} (C_0, C_1) : k \overset{\$}{\leftarrow} \mathsf{symKeyGen}(1^\lambda), C_0 \overset{\$}{\leftarrow} \overline{\mathsf{Enc}}(\mathsf{pk}, k), C_1 \overset{\$}{\leftarrow} \mathsf{symEnc}(k, m)$
- $\mathsf{Dec}(\mathsf{sk}, C) \to m' : k' \leftarrow \mathsf{Dec}(\mathsf{sk}, C_0), m' \leftarrow \mathsf{symDec}(k', C_1)$
- $\mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j) \overset{(\$)}{\to} \Delta_{i,j} : \Delta_{i,j} \overset{(\$)}{\leftarrow} \overline{\mathsf{ReKeyGen}}(\mathsf{sk}_i, \mathsf{pk}_j)$
- $\mathsf{ReEnc}(\Delta_{i,j}, C) \overset{(\$)}{\to} (C'_0, C'_1) : C'_0 \overset{(\$)}{\leftarrow} \overline{\mathsf{ReEnc}}(\Delta_{i,j}, C_0), C'_1 = C_1$

This scheme is also $\mathsf{IND\text{-}HRA}$, but is not PCS. An adversary $\mathcal{A}$ can submit two ciphertexts to the challenge oracle, and compare the second part of the challenge re-encryption with the submitted ciphertexts to win the game.

Lemma 6. *PCS $\not\Longrightarrow$ PRE-IND-CPA-security.*

Proof. Let $\overline{\mathcal{PRE}} = (\overline{\mathsf{KeyGen}}, \overline{\mathsf{Enc}}, \overline{\mathsf{Dec}}, \overline{\mathsf{ReKeyGen}}, \overline{\mathsf{ReEnc}})$ be a PRE scheme that is IND-HRA-secure and has PCS. We use it to construct the following PRE scheme:

- $\mathsf{KeyGen}(1^\lambda) \overset{\$}{\to} (\mathsf{pk}, \mathsf{sk}) : (\mathsf{pk}, \mathsf{sk}) \overset{\$}{\leftarrow} \overline{\mathsf{KeyGen}}(1^\lambda)$
- $\mathsf{Enc}(\mathsf{pk}, m) \overset{\$}{\to} (C_0, C_1) : C \overset{\$}{\leftarrow} (m, \overline{\mathsf{Enc}}(\mathsf{pk}, m))$
- $\mathsf{Dec}(\mathsf{sk}, C) \to m' : m' \leftarrow \overline{\mathsf{Dec}}(\mathsf{sk}, C_1)$
- $\mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j) \overset{(\$)}{\to} \Delta_{i,j} : \Delta_{i,j} \overset{(\$)}{\leftarrow} \overline{\mathsf{ReKeyGen}}(\mathsf{sk}_i, \mathsf{pk}_j)$
- $\mathsf{ReEnc}(\Delta_{i,j}, C) \overset{(\$)}{\to} (C'_0, C'_1) : C'_0 \overset{\$}{\leftarrow} \overline{\mathsf{Enc}}(\mathsf{pk}_j, 0), C'_1 \overset{(\$)}{\leftarrow} \overline{\mathsf{ReEnc}}(\Delta_{i,j}, C_1)$

[6]However, it appears that if a re-encryption scheme has transparency, then it is much easier to relate security for updatable encryption schemes with security for PRE schemes meaning that the relation applies exactly for such schemes.

Clearly this scheme is not PRE-IND-CPA secure, as fresh ciphertexts contain the plaintext. However the scheme has PCS, as re-encryptions C_1' will be unrelated to C_1, since $\overline{\mathcal{PRE}}$ has PCS, and the creation of C_0' is independent of both C_0 and $\Delta_{i,j}$.

Since PCS does not imply any security notion concerning confidentiality of the message, confidentiality definitions must be proven separately in order to demonstrate that a PRE scheme is useful in practice.

4.4 PCS via source-hiding

This section has been updated, partly due to errors and partly due to the change in our given definition of source-hiding in Definition 9 as we no longer use a multi-key, multi-challenge version. The main impact is that the proven advantages are now higher, but still negligible.

In this section we show that a PRE scheme that is both source-hiding and PRE-IND-CPA-secure also has PCS.

Theorem 2 (main). *Let $\mathcal{PRE}$ be a PRE scheme with transparency which is both PRE-IND-CPA-secure and source-hiding. Then $\mathcal{PRE}$ also has PCS, limited to the case where re-encryption graphs are acyclic.*

More specifically, if $\mathcal{PRE}$ is ϵ_{CPA}-PRE-IND-CPA-secure and ϵ_{SH}-source-hiding, then the advantage ϵ of an adversary $\mathcal{A}$ in winning $\mathsf{PostComp}_{\mathcal{A}}^{\mathcal{PRE}}$ is

$$\epsilon \leq \kappa(\kappa-1)(Q_E + Q_{HRE} + 1)(Q_{HRE} + 1) \cdot \epsilon_{\mathsf{SH}} + \epsilon_{\mathsf{CPA}} < \mathsf{negl}(\lambda)$$

for some negligible function $\mathsf{negl}(\lambda)$, where Q_E is the number of queries $\mathcal{A}$ makes to $\mathsf{O}_{\mathsf{Enc}}$, and Q_{HRE} is the number of re-encryption queries for non-challenge ciphertexts that $\mathcal{A}$ makes to $\mathsf{O}_{\mathsf{ReEnc}}$.

The intuition behind this proof is that as source-hiding implies that re-encrypted ciphertexts appear to be independent of the original ciphertext (and the old secret key), the challenge re-encryption in $\mathsf{PostComp}_{\mathcal{A}}^{\mathcal{PRE}}$ can be replaced with a fresh encryption of m_b at the appropriate level. As long as the ciphertext does not reveal anything about the underlying message, $\mathcal{PRE}$ is PCS. We give a reduction-based proof that a PRE scheme that is both source-hiding and PRE-IND-CPA-secure is PCS.

Proof. We prove this theorem using a sequence of game hops, breaking the proof down into a number of lemmas. Let $\mathsf{PostCompSH}$ be a variant of the $\mathsf{PostComp}$ game where all re-encryptions of non-challenge ciphertexts and the re-encryption made by $\mathsf{O}_{\mathsf{challenge}}^{\mathsf{PostComp}}$ are replaced with fresh encryptions at the appropriate level. Let Game^b denote the experiment Game where choice of b is made explicit. We first demonstrate that $\mathsf{PostComp}$ and $\mathsf{PostCompSH}$ are computationally indistinguishable, before showing how PRE-IND-CPA-security implies that $\mathsf{PostCompSH}^0$ and $\mathsf{PostCompSH}^1$ are computationally indistinguishable.

Lemma 7. *Let ϵ_0 be the advantage in distinguishing between $\mathsf{PostCompSH}$ and $\mathsf{PostComp}$. Then*

$$\epsilon_0 \leq \kappa(\kappa-1)(Q_E + Q_{HRE} + 1)(Q_{HRE} + 1) \cdot \epsilon_{\mathsf{SH}}.$$

Proof. This proof is based on proof of [FKKP18, Lemma 7], which gives a reduction based on source-hiding. We use a hybrid argument where re-encryptions are replaced one at a time. Let $\mathsf{PostCompSH}_0 := \mathsf{PostComp}$, and let $\mathsf{PostCompSH}_k$ be identical to $\mathsf{PostCompSH}_{k-1}$ except that the k^{th} re-encryption query is replaced with a fresh encryption. We see that $\mathsf{PostCompSH}_{Q_{HRE}+1} = \mathsf{PostCompSH}$.

We describe how an adversary $\mathcal{B}$ trying to win SH^d can simulate either $\mathsf{PostCompSH}_{k-1}$ or $\mathsf{PostCompSH}_k$, depending on the bit d. Therefore, if there exists a PPT distinguisher $\mathcal{D}$ that, with advantage δ, outputs $d'_{\mathcal{D}} = 0$ to indicate that it is playing $\mathsf{PostCompSH}_{k-1}$ and $d'_{\mathcal{D}} = 1$ to indicate that it is playing $\mathsf{PostCompSH}_k$, then $\mathcal{B}$ can run $\mathcal{D}$ as a subroutine and set $d' = d'_{\mathcal{D}}$ to win SH with the same advantage.

$\mathcal{B}$ first sets $b \overset{\$}{\leftarrow} \{0,1\}$ for PostComp. Note that SH only has two keypairs $(\mathsf{pk}_0^{\mathsf{SH}}, \mathsf{sk}_0^{\mathsf{SH}}), (\mathsf{pk}_1^{\mathsf{SH}}, \mathsf{sk}_1^{\mathsf{SH}})$, and that the challenge $\mathsf{O}^{\mathsf{SH}}_{\mathsf{challenge}}(m, \ell)$ returns both the fresh encryption C^* and the challenge $C^{(b)}$. Therefore, $\mathcal{B}$ must first guess which ciphertext C and key indexes i^*, j^*will be input for the k^{th} replacement. $\mathcal{B}$ responds to the k^{th} re-encryption query as follows:

– It takes the two keypairs from the source-hiding game $(\mathsf{pk}_0^{\mathsf{SH}}, \mathsf{sk}_0^{\mathsf{SH}}), (\mathsf{pk}_1^{\mathsf{SH}}, \mathsf{sk}_1^{\mathsf{SH}})$, sets these as $(\mathsf{pk}_{i^*}, \mathsf{sk}_{i^*}), (\mathsf{pk}_{j^*}, \mathsf{sk}_{j^*})$ respectively, and then self-generates the remaining $\kappa - 2$ key pairs for PostComp.
– When the adversary makes the query that leads to the creation of C under pk_i (which could be either through $\mathsf{O}_{\mathsf{Enc}}$ or $\mathsf{O}_{\mathsf{ReEnc}}$)), if $i \neq i^*$ then the reduction aborts. Otherwise the reduction takes the underlying message and uses it to query $\mathsf{O}^{\mathsf{SH}}_{\mathsf{challenge}}(m, \ell) \overset{\$}{\rightarrow} (C^*, C'^{(b)})$ and returns C^* to the adversary. Note that C^* is created via oracles, either by $O_{\mathsf{Enc}}(i, m)$ or by calling $\mathsf{O}_{\mathsf{ReEnc}}$ on an oracle-created ciphertext. This means that for a pair (i, C) where C is an oracle-created ciphertext under pk_i reduction will know the underling message m and the number of times ℓ that the ciphertext has been (or appears to have been) re-encrypted.
– Let the k^{th} re-encryption is the adversary queries for an honest ciphertext C be $\mathsf{O}_{\mathsf{ReEnc}}(i^*, j, (\Delta_{i^*,j}), C)$. If $C \neq C^*$ or $j \neq j^*$, the reduction aborts. Otherwise, it returns the challenge $C'^{(b)}$ it received from the source-hiding game.

Note that for the challenge ciphertext, $\mathcal{B}$ has chosen b and therefore can easily simulate the challenge by re-encrypting as described above.

Because $\mathcal{PRE}$ is ϵ_{SH}-source-hiding by assumption, we conclude that the advantage δ that $\mathcal{D}$ has in distinguishing between $\mathsf{PostCompSH}_{k-1}$ and $\mathsf{PostCompSH}_k$ cannot be greater than $\kappa(\kappa-1)(Q_E + Q_{HRE} + 1) \cdot \epsilon_{\mathsf{SH}}$. Because $Q_{HRE} + 1$ replacements need to be made in total, we get that the advantage of distinguishing between PostComp and PostCompSH is bounded by $\epsilon_0 \leq \delta \cdot (Q_{HRE} + 1) \cdot \epsilon_{\mathsf{SH}}$, as required.

Lemma 8. *If $\mathcal{PRE}$ is ϵ_{CPA}-secure, then the advantage ϵ_1 in distinguishing between* $\mathsf{PostCompSH}^{0,\mathcal{PRE}}$ *and* $\mathsf{PostCompSH}^{0,\mathcal{PRE}}$ *is*

$$\epsilon_1 \leq \epsilon_{\mathsf{CPA}}.$$

Proof. We show how an adversary $\mathcal{A}$ in $\mathsf{PRE\text{-}IND\text{-}CPA}^b$ can simulate $\mathsf{PostCompSH}^{b,\mathcal{PRE}}$. Thus, if there exists a PPT adversary $\mathcal{B}$ that wins PostCompSH (returning $b'_{\mathcal{B}}$) with advantage ϵ', then $\mathcal{A}$ can win $\mathsf{PRE\text{-}IND\text{-}CPA}^{b,\mathcal{PRE}}$ with the same advantage by returning $b' = b'_{\mathcal{B}}$. $\mathcal{A}$ simulates $\mathsf{PostComp}^{b,\mathcal{PRE}}$ by responding to oracle queries made by $\mathcal{B}$ as follows:

– $\mathcal{A}$ simulates $\mathsf{O}_{\mathsf{Enc}}$ as described in $\mathsf{PostComp}$, also setting lookup tables $\mathcal{C}_{\mathsf{msg}}[(i, C)] := m$ for the underlying message m and $\mathcal{C}_{\mathsf{lev}}[(i, C)] := 0$ for the ciphertext level.
– $\mathcal{A}$ forwards calls to $\mathsf{O}_{\mathsf{ReKeyGen}}$ to the same oracle in its own game.
– To simulate $\mathsf{O}^{\mathsf{PC}}_{\mathsf{challenge}}(i^*, j^*, [\Delta_{i^*,j^*}], C_0^*, C_1^*)$, $\mathcal{A}$ first retrieves $m_0 \leftarrow \mathcal{C}_{\mathsf{msg}}[(C_0^*, i^*)]$, $\ell_0 \leftarrow \mathcal{C}_{\mathsf{lev}}[(C_0^*)]$, $m_1 \leftarrow \mathcal{C}_{\mathsf{msg}}[(C_1^*)]$, $\ell_1 \leftarrow \mathcal{C}_{\mathsf{lev}}[(C_1^*, i^*)]$. Then $\mathcal{A}$ queries $\mathsf{O}^{\mathsf{CPA}}_{\mathsf{challenge}}(j, m_0, m_1) \rightarrow C'^{(b)}$.
– $\mathcal{A}$ simulates $\mathsf{O}_{\mathsf{ReEnc}}(i, j, C)$ for non-challenge ciphertexts by retrieving $m \leftarrow \mathcal{C}_{\mathsf{msg}}[(i, C)]$ and $\ell \leftarrow \mathcal{C}_{\mathsf{lev}}[(i, C)]$ and returning $C' \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_j, m, \ell + 1)$. $\mathcal{A}$ sets $\mathcal{C}_{\mathsf{msg}}[(j, C')] := m, \mathcal{C}_{\mathsf{lev}}[(j, C')] := \ell + 1$.
For challenge ciphertexts, if $\mathcal{B}$ has set $\Delta_{i,j} = \perp$ then $\mathcal{A}$ retrieved $\Delta_{i,j} \leftarrow \mathsf{O}_{\mathsf{ReKeyGen}}(i, j)$ and returns $C' \leftarrow \mathsf{ReEnc}(\Delta_{i,j}, C)$.

We note that, as $\mathcal{PRE}$ has transparency, the level of the challenge re-encryption will be hidden meaning that $\mathcal{B}$ cannot submit ciphertexts at different levels and use this to distinguish $C'^{(b)}$.

The resulting simulation is identical to $\mathsf{PostCompSH}^b$. Therefore, if there exists an adversary $\mathcal{B}$ that wins $\mathsf{PRE\text{-}IND\text{-}CPA}^b$ with advantage ϵ, then $\mathcal{A}$ can win $\mathsf{PostCompSH}$ with the same advantage. We conclude that $\epsilon_1 \leq \epsilon_{\mathsf{CPA}}$, as required.

By Lemmas 7 and 8 and the triangle inequality, we see that

$$\epsilon \leq \kappa(\kappa - 1)(Q_E + Q_{HRE} + 1)(Q_{HRE} + 1) \cdot \epsilon_{\mathsf{SH}} + \epsilon_{\mathsf{CPA}},$$

as required. □

Remarks. *These remarks address the benefits or our original approach where we used a multi-key, multi-challenge variant of source-hiding.*

We note that a multi-key, multi-challenge variant of source-hiding will result in a simpler reduction, as the simulator will no longer need to guess in advance which keys and ciphertexts will be used in replacing the re-encrypted ciphertext with a fresh encryption. We conjecture that proving that a PRE scheme meets multi-key, multi-challenge variant of source-hiding should not incur a significant loss, proving source-hiding requires a statistical argument that ciphertexts are identically distributed, which should apply to multiple keys and ciphertexts as long as they are honestly-generated. Indeed, it is possible that the overall loss from using a multi-key, multi-challenge notion of source-hiding will mean the overall loss of security is less severe than it is in the current approach.

Finally, we conjecture that there is the potential for a tighter relation between IND-HRA-security and PCS, as at a high level the only re-encryption that needs replacing here is the one performed by $\mathsf{O}^{\mathsf{PC}}_{\mathsf{challenge}}$. Because the single-challenge definition of source-hiding has limited application in that it only says that re-encryptions $C' \xleftarrow{(\$)} \mathsf{ReEnc}(\Delta_{i,j}, C)$ of fresh ciphertexts $C \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, m, [\ell])$ can be replaced by fresh encryptions, whereas we need this property to hold even when C is itself a re-encrypted ciphertext.

It should be possible that such a replacement can be made without needing to replace all previous re-encryptions. Informally, this follows trivially from an iterative argument, or from the statistical argument outlined above.

4.5 Existing PRE schemes that satisfy PCS

This subsection has also been updated to discuss the possibility that additional schemes have PCS.

In [FKKP18, Construction 7.b], Fuchsbauer et al. present a PRE scheme which is PRE-IND-CPA-secure and source-hiding. This construction is based on the hardness of the decision learning with errors problem with sub-exponential noise-to-modulus ratio. Using the result of Theorem 2, see that this scheme is PCS. However, current known techniques that make a PRE scheme provably source-hiding result in a scheme that is both less efficient, and relies on stronger security assumptions. We shall discuss this in detail in Section 6.3.

We therefore aim to demonstrate that source-hiding is not necessary for proving PCS, which we do with our construction in Section 5. We note that, subject to the remarks given in Section 4.4, it is possible that [FKKP18, Construction 2, Construction 4] also have PCS, as, whilst these constructions are not source-hiding, they do re-randomise ciphertexts upon re-encryption in a way that is necessary for PCS.

5 An Efficient Construction from Lattices

We introduce a natural construction with PCS, based on $\mathsf{BV\text{-}PRE}$ – the ring-LWE (RLWE) construction presented in [PRSV17]. Whilst Theorem 2 shows that source-hiding can lead to PCS, the existing constructions with this property [FKKP18] make sub-optimal parameter choices that significantly impact the schemes' practicality. Our construction has PCS but is not source-hiding, implying that source-hiding is not necessary for PCS. This means that our construction can make much better parameter choices in terms of efficiency. Our construction also does not rely on strong assumptions or heavy techniques such as obfuscation [HRSV07, CCV12]. We also achieve transparency, meaning the cost of decryption does not grow for repeatedly re-encrypted ciphertexts. If extra computation is needed to decrypt a re-encrypted ciphertext, then this may outweigh the benefits of outsourcing re-encryption. Heavier computation may go against the reasons for outsourcing re-encryption to begin with.

Our construction makes some adaptations to $\mathsf{BV\text{-}PRE}$ to fit the workflow of PRE; making use of the key resampling technique of [BV11] to re-randomise the ciphertext. Any scheme that permits similar re-randomisation can be proven secure using related methods. We begin this section by covering some necessary preliminaries for lattices.

5.1 Lattice preliminaries

We let $q \in \mathbb{Z}$ denote some modulus and n denote a ring dimension. We represent the set of integers modulo q as $\mathbb{Z}_q = \{\lfloor -q/2 \rfloor, \ldots, 0, \ldots, \lfloor q/2 \rfloor\}$. We will be working over power-of-two cyclotomic rings of the form $\mathcal{R}_q = \mathbb{Z}_q[x]/\langle x^n + \rangle$ where n is a power of two. We use Φ to denote the polynomial such that $\mathcal{R}_q = \mathbb{Z}_q[x]/\Phi$. Next we discuss the various distributions that will be used. We use the notation $s \xleftarrow{\$} D$ to denote that the element s is sampled according to distribution D. If D is a set, then we assume $s \xleftarrow{\$} D$ means that s is sampled uniformly from the set D. We denote the discrete Gaussian distribution over $\mathbb{Z}_q$ as χ_σ. The distribution χ_σ has its support restricted to $\mathbb{Z}_q$ and a probability mass function proportional to that of a Gaussian distribution with variance σ^2. We sometimes write χ_e where we use a subscript e to denote an "error" distribution, but the underlying variance is still denoted by σ. Slightly abusing notation, we can sample a polynomial $s \xleftarrow{\$} \chi_e$ by sampling each of the coefficients of s according to the distribution χ_e. We say a distribution D is (B, δ)-bounded if $\Pr\left(|x| > B : x \xleftarrow{\$} D\right) \leq \delta$. We denote the uniform distribution over $\mathcal{R}_q$ as $\mathcal{U}_q$, and we use the shorthand $(a, b) \xleftarrow{\$} \mathcal{U}_q^2$ to mean that $a, b \xleftarrow{\$} \mathcal{U}_q$.

ring-LWE (RLWE) assumption: Let s be some secret polynomial in $\mathcal{R}_q$. Samples from the $\mathrm{RLWE}_{n,q,\chi_e}(s)$ distribution take the form $(a, b = as + e) \in \mathcal{R}_q \times \mathcal{R}_q$ where $a \overset{\$}{\leftarrow} \mathcal{U}_q$, $e \overset{\$}{\leftarrow} \chi_e$. The *(normal form) $RLWE_{n,q,\chi_e}$ problem* is to distinguish between an oracle that outputs samples from $\mathrm{RLWE}_{n,q,\chi_e}(s)$ where $s \overset{\$}{\leftarrow} \chi_e$ and an oracle that outputs uniform elements in $\mathcal{R}_q^2$. The *$RLWE_{n,q,\chi_e}$ assumption* states that no PPT algorithm can solve the $\mathrm{RLWE}_{n,q,\chi_e}$ problem with a non-negligible advantage. Note that if we take $\sigma \geq \omega(\log n)$ and $\sigma/q = 1/\mathrm{poly}(n)$, the $\mathrm{RLWE}_{n,q,\chi_e}$ problem is at least as hard as solving standard worst-case lattice problems over ideal lattices up to polynomial approximation factors using quantum algorithms [LPR10].

5.2 Adapting $\mathsf{BV\text{-}PRE}$ for PCS

The underlying scheme, $\mathsf{BV\text{-}PRE}$ [PRSV17], is based on the BV encryption scheme [BV11], which is based on RLWE. This scheme is parameterised by ciphertext modulus q, plaintext modulus $p \geq 2$, ring dimension n, polynomial ring $\mathcal{R}_q = \mathbb{Z}_q[n]/\langle x^n + 1\rangle$ and relinearisation window r. $\mathsf{BV\text{-}PRE}$ uses a relinearisation technique to reduce the size of the error that is added to a ciphertext upon re-encryption. During relinearisation, we break down a ciphertext into parts as determined by the relinearisation window:

$$C = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} C_i \cdot (2^r)^i. \qquad (1)$$

Note $\mathsf{BV\text{-}PRE}$ is not fully public-key, relying on an additional 'public'[7] key 'pk'$_B$ for the target key sk_B to generate update tokens. These 'public' keys are also RLWE samples with the same underlying secret, s_B as the public key pk_B. However, and these 'public' keys are assumed to be unknown to the adversary when proving security. Indeed any entity in possession of both the 'public' key and an update token created from this key can derive the old secret key, sk_A. A description of the full $\mathsf{BV\text{-}PRE}$ scheme can be found in Appendix C.

We get around the need for additional 'public' keys using the key resampling technique $\mathsf{ReSample}$ [BV11] which takes a public key pk_B and outputs a fresh public key pk'_B with the same underlying secret. We also use same relinearisation technique as [PRSV17] to reduce error growth. We describe the re-sampling algorithm $\mathsf{ReSample}$ in Appendix D, but note that it can also be written as $(\bar{(a)}, \bar{(b)}) \overset{\$}{\leftarrow} \mathsf{pcBV\text{-}PRE.Enc}((a, b), 0)$.

We give our construction, $\mathsf{pcBV\text{-}PRE}$, in Figure 8. It builds on $\mathsf{BV\text{-}PRE}$ by having the proxy add further randomness in the ReEnc operation. Recall that this is necessary for a scheme to have PCS, as otherwise an adversary could re-encrypt locally to obtain the same re-encryption. This additional randomness has minor implications for how this affects the correctness requirement for multiple re-encryptions over that presented in [PRSV17].

5.3 Security proofs

For brevity, we defer the correctness analysis of $\mathsf{pcBV\text{-}PRE}$ to Appendix E.

Post-Compromise Security. *The proof in this updated version is new, as it addresses errors in the previous proof regarding how key corruption is handled, and how re-encryptions of non-challenge ciphertexts are handled.*

[7] Polyakov et al. also refer to these keys as "'public'".

```
KeyGen(1^λ)
    $a \xleftarrow{\$} \mathcal{U}_q$,
    $s, e \xleftarrow{\$} \chi_e$
    $b = a \cdot s + pe$
    $\mathsf{sk} = s$
    $\mathsf{pk} = (a, b)$
    return $(\mathsf{pk}, \mathsf{sk})$

Enc(pk, m) $\xrightarrow{\$} C$
    $(a, b) \leftarrow \mathsf{pk}$
    $v, e_0, e_1 \xleftarrow{\$} \chi_e$
    $C_0 = b \cdot v + pe_0 + m$
    $C_1 = a \cdot v + pe_1$
    return $c = (C_0, C_1)$

Dec(sk, C) $\to m$
    $s \leftarrow \mathsf{sk}$
    $m' = C_0 - s \cdot C_1 \mod p$
    return $m'$

ReKeyGen(sk_A, pk_B) $\xrightarrow{\$} \Delta_{A,B}$
    for $i \in \{0, 1, \ldots, \lfloor \log_2(q)/r \rfloor\}$ :
        $(\beta_i, \theta_i) \xleftarrow{\$} \mathsf{ReSample}(\mathsf{pk}_B)$
        $\gamma_i = \theta_i - \mathsf{sk}_A \cdot (2^r)^i$
    $\Delta_{A \to B} = \left(\{(\beta_i, \gamma_i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor}, \mathsf{pk}_B\right)$
    return $\Delta_{A \to B}$

ReEnc(Δ_{A→B}, C) $\xrightarrow{\$} C'$
    $\left(\{(\beta_i, \gamma_i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor}, \mathsf{pk}_B\right) \leftarrow \Delta_{A \to B}$
    $(C_0, C_1) \leftarrow c$
    $(\beta^{\text{proxy}}, \theta^{\text{proxy}}) \xleftarrow{\$} \mathsf{ReSample}(\mathsf{pk}_B)$
    $C'_0 = C_0 + \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (C_1^{(i)} \cdot \gamma_i) + \theta^{\text{proxy}}$
    $C'_1 = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (C_1^{(i)} \cdot \beta_i) + \beta^{\text{proxy}}$
    return $C' = (C'_0, C'_1)$
```

Fig. 8: pcBV-PRE. This adapts BV-PRE [PRSV17] to be fully public-key, and minimises computation and bandwidth for the user for the re-encryption process. The key resampling algorithm ReSample is described in Appendix D.

We firstly show that pcBV-PRE is satisfies PCS with a direct proof. In other words, we do not leverage the proof of Theorem 2 to prove PCS via source-hiding security. This is because pcBV-PRE as written does not satisfy source-hiding security, see Section 6.3 for more details.

Theorem 3. pcBV-PRE *has Post-Compromise Security. In other words, for any adversary $\mathcal{A}$ to the* PostComp *game,*

$$\left|\Pr\left[\mathsf{PostComp}_{\mathcal{A}}^{0,\mathcal{PRE}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{PostComp}_{\mathcal{A}}^{1,\mathsf{pcBV\text{-}PRE}}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

for some $\epsilon = \mathsf{negl}(\lambda)$ under the $RLWE_{n,q,\chi_e}$ assumption.

Proof Overview. We only give a brief informal argument here and refer the reader to Appendix E for a full proof. We first demonstrate that the scheme is secure for a variant of the game where re-encryptions of non-challenge ciphertexts from uncorrupted to corrupted keys are forbidden. We call this variant CPA-PostComp.

This proof is adapted from the PRE-IND-CPA-security (IND-CPA-security in their terminology) proof of the BV-PRE in [PRSV17]. It follows a sequence of game hops beginning with the $\mathsf{CPA\text{-}PostComp}^{b,\mathsf{pcBV\text{-}PRE}}$ security game where $b \xleftarrow{\$} \{0, 1\}$. In this game, the adversary is challenged to guess the bit b. Then we use a hybrid argument with a series of game hops where each game hop replaces:

1. resamples of the public key of a single honest entity with uniform random values, and
2. the update tokens created using the honest entity's public key with uniform random values.

Finally, the challenge re-encryption given to the adversary is a uniformly sampled value and thus the adversary has no advantage in this game. This implies that $\mathsf{CPA\text{-}PostComp}^{0,\mathsf{pcBV\text{-}PRE}}$ and $\mathsf{CPA\text{-}PostComp}^{1,\mathsf{pcBV\text{-}PRE}}$ are indistinguishable.

To prove security for the full $\mathsf{PostComp}$ game including honest re-encryptions, we use a similar idea to re-encryption simulatability to demonstrate how re-encryptions from uncorrupted to corrupted keys can be simulated without the update token or knowledge of the old secret key. We demonstrate how this simulated game is indistinguishable from $\mathsf{PostComp}$, meaning that $\mathsf{pcBV\text{-}PRE}$ has post-compromise security. The full proof is given in Appendix E.

PRE-IND-CPA security. We secondly show that $\mathsf{pcBV\text{-}PRE}$ is a PRE-IND-CPA-secure PRE scheme. This is a much simpler proof since it follows a similar argument to that of Theorem 3, and the fact that the same security notion was shown in [PRSV17, Theorem 2].

Theorem 4. $\mathsf{pcBV\text{-}PRE}$ *is IND-HRA secure.*

Proof. The proof of this property follows an identical similar argument to that of Theorem 3, except that the challenge is now replacing a fresh encryption as opposed to a re-encrypted ciphertext. □

6 From selective to adaptive security

Thus far, we have discussed selective security, where the adversary first corrupts keys, then learns challenges, in two distinct stages. In the adaptive model, a single-stage adversary can corrupt secret keys at any point and therefore choose which keys to corrupt as a result of received challenges. As long as the trivial win condition holds (that no challenge key has been corrupted), the adversary can adaptively decide which keys should be corrupted depending on the output of other oracle queries.

To lift to this stronger model, we need notions of key privacy. The intuition behind this is that we can replace some challenge-related queries with counterparts under different keys and relate this to key privacy.

6.1 Weak Key Privacy

In this section, we present the formal definitions for the related notions of key privacy. *key privacy* (sometimes called *key anonymity*), was first introduced for PKE in [BBDP01]. Informally it means that an adversary is unable to determine which public key was used to encrypt a ciphertext. A strong version for PRE schemes for update tokens appears in [ABH09] which is referred to as *strong key privacy* in [FKKP18], and states that an adversary is unable to distinguish between an update token between two uncorrupted keys and a random element of the token space. Note that this is strictly stronger than key inference where the hidden secret keys must be computed, as is is the concern in *collusion attacks*. We refer the interested reader to [ABH09] for further details.

In their work relating selective and adaptive security, Fuchsbauer et al. [FKKP18] define *weak key privacy* and show that this is sufficient for adaptive security. We now define this formally.

Definition 11. *A PRE scheme $\mathcal{PRE}$ is said to have* ϵ-weak key privacy *if for all* PPT *adversaries $\mathcal{A}$:*

$$\left|\Pr\left[\mathsf{weakKP}^0_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{weakKP}^1_{\mathcal{A}}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

where $\mathsf{weakKP}^{b,\mathcal{PRE}}_{\mathcal{A}}$ *is defined in Figure 9.*

```
$\mathsf{weakKP}_{\mathcal{A}}^{b,\mathcal{PRE}}(1^\lambda)$
$\kappa = 0$
$b' \overset{(\$)}{\leftarrow} \mathcal{A}^{\mathsf{O}_{\mathsf{KeyGen}},\mathsf{O}_{\mathsf{challenge}}}(1^\lambda)$
return $b'$
```

```
$\mathsf{O}_{\mathsf{KeyGen}}()$
$\kappa \leftarrow \kappa + 1$
$(\mathsf{pk}_\kappa, \mathsf{sk}_\kappa) \overset{\$}{\leftarrow} \mathsf{KeyGen}(1^\lambda)$
return $\mathsf{pk}_\kappa$
```

```
$\mathsf{O}_{\mathsf{challenge}}(i, j)$
$\Delta^0 \overset{(\$)}{\leftarrow} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
$(\tilde{\mathsf{sk}}, \tilde{\mathsf{pk}}) \overset{\$}{\leftarrow} \mathsf{KeyGen}(1^\lambda)$
$\Delta^1 \overset{(\$)}{\leftarrow} \mathsf{ReKeyGen}(\tilde{\mathsf{sk}}, \mathsf{pk}_j)$
return $\Delta^b$
```

Fig. 9: The weak key privacy game [FKKP18, Definion 8]. Given an update token, an adversary cannot distinguish the source key.

6.2 Adaptive Post-Compromise Security

In this section, we discuss PCS with adaptive key corruptions. We give the full adaptive game for adaptive PCS in Appendix G.

In general, selective security can be shown to imply adaptive security at an exponential loss, where the adversary in the selective game must first guess which keys the adaptive adversary will corrupt. This is known as *complexity leveraging*. A general framework for showing relations between selective and adaptive security for general definitions is given in [JKK$^+$17]. Jafargholi et al. achieve this by constructing a variant of the selective game, then using pebbling games that dictate what replacements can be made in a series of hybrids, leading to the adaptive game, demonstrating the loss of security between each hybrid. Jafargholi et al. demonstrate how using pebbling games can give a smaller loss of security. We refer the interested reader to [JKK$^+$17,FKKP18] for a full description of pebbling games and their usage. A similar approach specific to PRE is used in [FKKP18], limited to when the adversary creates directed re-encryption graphs which are acyclic and have only one source node. An even tighter reduction is given which results in quasi-polynomial loss between selective and adaptive HRA security for PRE schemes with ciphertext indistinguishability and weak key privacy, limited to some types of directed re-encryption graph, namely trees and chains.

Selective PRE-IND-CPA security combined with weak key privacy gives adaptive PRE-IND-CPA security [FKKP18, Theorem 1, Theorem 5]. The loss of security compared to the selective setting depends on the number of keys κ generated during the game, and the space and time complexities of the $\mathcal{DRG}$. More precisely, the loss of security is approximately $\tau \cdot \kappa^\sigma$, where σ is the maximum number of pebbles and τ is the maximum number of moves for a valid pebbling strategy for a considered class of re-encryption graphs $\mathcal{DRG}$. This means that the loss is exponential for general re-encryption graphs, but quasi-polynomial for trees and chains. We therefore obtain the following theorem, whose proof follows from Theorem 2 and uses the same techniques as those used to prove [FKKP18, Theorem 5]:

Theorem 5. *Let $\mathcal{PRE}$ be a PRE scheme which is ϵ_{CPA}-selectively PRE-IND-CPA secure, ϵ_{SH}-source-hiding and ϵ_{wKP}-weakly key private. for some negligible function* $\mathsf{negl}(\lambda)$*, where Q_E is the number of queries $\mathcal{A}$ makes to $\mathsf{O}_{\mathsf{Enc}}$, and Q_{HRE} is the number of re-encryption queries for non-challenge ciphertexts that $\mathcal{A}$ makes to $\mathsf{O}_{\mathsf{ReEnc}}$. Let Q_E be the number of encryption oracle queries made by the adversary, and Q_{HRE} be the number of re-encryption queries for non-challenge ciphertexts that the adversary makes to $\mathsf{O}_{\mathsf{ReEnc}}$.*

Then $\mathcal{PRE}$ is ϵ-adaptively PCS secure with a security loss of $\approx \tau \cdot \kappa^\sigma$ for directed re-encryption graphs $\mathcal{DRG}$ with degree κ, space complexity σ and time complexity τ, where

$$\epsilon \leq 2\kappa(\kappa-1)(Q_E + Q_{HRE} + 1)(Q_{HRE} + 1) \cdot \epsilon_{\mathsf{SH}} + \kappa^{\sigma+\delta+1}(\epsilon_{\mathsf{CPA}} + 2\tau \cdot \epsilon_{\mathsf{wKP}}), \qquad (2)$$

restricted to graphs $\mathcal{DRG}$ which are acyclic, have at most κ nodes, degree δ and depth d.

6.3 Achieving adaptive security of pcBV-PRE

Our construction pcBV-PRE is not source-hiding. This is because fresh encryptions will have different noise magnitudes compared with re-encryptions. One method of overcoming this is using a noise 'blurring' approach [CCL$^+$14], as was noted by [FKKP18, Construction 7.b]. This would involve 'blurring' or 'drowning' the noise in fresh encryptions so that it was distributed in the same way as re-encryptions. To do this, we would modify the ciphertext pairs in the Enc and ReEnc algorithms so that we would add fresh noise to them. This fresh noise would be taken from an error distribution such that it 'blurred' out all the old noise that was introduced from the encryption and re-encryption operations. We present a variant of pcBV-PRE that is provably source-hiding using this blurring approach in Appendix E.3.

The advantage of doing this, is that on decryption the noise will not reveal anything about the method of encryption/re-encryption that was used, or the keys the ciphertext was previously encrypted under. This is vital since in the source-hiding security property, the adversary receives all of the available decryption keys and thus can decrypt any ciphertext that they want.

With this in mind, there are a number of reasons why proving source-hiding is actually a hindrance. Using the same approach as [FKKP18, construction 7b] requires at least a sub-exponential noise-to-modulus ratio which considerably harms performance and security. This is because the LWE assumption (and respectively RLWE in our case) that is used becomes much stronger, since the approximation factors of the related ideal lattice problems become sub-exponential rather than polynomial in n. In particular, there are quantum polynomial time algorithms solving ideal lattice problems with approximation factor $\exp(\tilde{O}(\sqrt{n}))$ [BS16, CDPR16, CDW17] providing evidence that we cannot simply use an arbitrary noise-to-modulus ratio while retaining the same security guarantees. Moreover, the increase in modulus that is required makes standard operations much slower. As a by-product, the scheme of [FKKP18] allows only a constant number of re-encryptions.

The intention with our construction was to give a practical PRE scheme with PCS with minimal restrictions and from weaker assumptions. Since pcBV-PRE is very close to the original (BV-PRE) [PRSV17] which is comparatively fast (see [PRSV17] for exact figures) and our construction only adds extra sampling, loss of efficiency is minimal. Incorporating source-hiding, results in a scheme that is impractical, based on much stronger assumptions and also heavily restricted in the number of re-encryptions that can occur.

We prove that pcBV-PRE has weak key privacy in Appendix H. Therefore, it should be noted that a source-hiding version of pcBV-PRE would achieve adaptive security (for the restricted graphs of [FKKP18]) by the proof of Theorem 2 and the fact that pcBV-PRE is PRE-IND-CPA secure (Theorems 1 and 4).

7 Conclusions and Future work

In this paper, we have presented the strongest notion of Post-Compromise Security for PRE to date. By strongest, we mean that existing Post-Compromise Security notions are implied by our notion of security, and we have presented separating examples showing that the opposite implication does not hold. We have also shown that PCS can be achieved via a number of existing PRE security notions which immediately shows that at least one existing PRE scheme satisfies PCS [FKKP18]. We presented an efficient construction of a PRE scheme with PCS based on lattices that is transparent. We have also discussed the possibility of achieving adaptive security for restricted re-encryption graphs. We leave as future work the possibility of proving tighter bounds between security notions, and further investigating the relationship between selective and adaptive security for more generic re-encryption graphs.

Acknowledgements

Special thanks to Katriel Cohn-Gordon for his help in motivating this work and providing the context for using PCS PRE to compliment the PCS messages in transit. We also thank Rachel Player for her assistance in proving Theorem 7.

References

ABH09. Giuseppe Ateniese, Karyn Benson, and Susan Hohenberger. Key-private proxy re-encryption. In Marc Fischlin, editor, *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*, volume 5473 of *Lecture Notes in Computer Science*, pages 279–294. Springer, 2009.

AFGH06. Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur.*, 9(1):1–30, 2006.

BBB$^+$12. Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid. Recommendation for key management part 1: General (revision 3). *NIST special publication*, 800(57):1–147, 2012.

BBDP01. Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.

BBS98. Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 127–144. Springer, 1998.

BLMR13. Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic prfs and their applications. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 410–428. Springer, 2013.

BS16. Jean-François Biasse and Fang Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 893–902. SIAM, 2016.

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 505–524. Springer, 2011.

CCG16. Katriel Cohn-Gordon, Cas J. F. Cremers, and Luke Garratt. On post-compromise security. In *IEEE 29th Computer Security Foundations Symposium, CSF 2016, Lisbon, Portugal, June 27 - July 1, 2016*, pages 164–178. IEEE Computer Society, 2016.

CCL$^+$14. Nishanth Chandran, Melissa Chase, Feng-Hao Liu, Ryo Nishimaki, and Keita Xagawa. Re-encryption, functional re-encryption, and multi-hop re-encryption: A framework for achieving obfuscation-based security and instantiations from lattices. In Hugo Krawczyk, editor, *Public-Key Cryptography - PKC 2014 - 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, March 26-28, 2014. Proceedings*, volume 8383 of *Lecture Notes in Computer Science*, pages 95–112. Springer, 2014.

CCV12. Nishanth Chandran, Melissa Chase, and Vinod Vaikuntanathan. Functional re-encryption and collusion-resistant obfuscation. In Ronald Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 404–421. Springer, 2012.

CDPR16. Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. Recovering short generators of principal ideals in cyclotomic rings. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 559–585. Springer, 2016.

CDW17. Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. Short stickelberger class relations and application to ideal-svp. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*, volume 10210 of *Lecture Notes in Computer Science*, pages 324–348, 2017.

CH07. Ran Canetti and Susan Hohenberger. Chosen-ciphertext secure proxy re-encryption. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*, pages 185–194. ACM, 2007.

Coh17. Aloni Cohen. What about bob? the inadequacy of CPA security for proxy reencryption. *IACR Cryptology ePrint Archive*, 2017:785, 2017.

EPRS17. Adam Everspaugh, Kenneth G. Paterson, Thomas Ristenpart, and Samuel Scott. Key rotation for authenticated encryption. *IACR Cryptology ePrint Archive*, 2017:527, 2017.

FKKP18. Georg Fuchsbauer, Chethan Kamath, Karen Klein, and Krzysztof Pietrzak. Adaptively secure proxy re-encryption. *IACR Cryptology ePrint Archive*, 2018:426, 2018.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178. ACM, 2009.

HRSV07. Susan Hohenberger, Guy N. Rothblum, Abhi Shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. In Salil P. Vadhan, editor, *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, volume 4392 of *Lecture Notes in Computer Science*, pages 233–252. Springer, 2007.

ID03. Anca-Andreea Ivan and Yevgeniy Dodis. Proxy cryptography revisited. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2003, San Diego, California, USA*. The Internet Society, 2003.

JKK$^+$17. Zahra Jafargholi, Chethan Kamath, Karen Klein, Ilan Komargodski, Krzysztof Pietrzak, and Daniel Wichs. Be adaptive, avoid overcommitting. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 133–163. Springer, 2017.

LATV13. Adriana Lopez-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. Cryptology ePrint Archive, Report 2013/094, 2013. `https://eprint.iacr.org/2013/094`.

Lee17. Ela Lee. Improved security notions for proxy re-encryption to enforce access control. Cryptology ePrint Archive, Report 2017/824, 2017. `http://eprint.iacr.org/2017/824`.

LPR10. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.

LT18. Anja Lehmann and Björn Tackmann. Updatable encryption with post-compromise security. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 685–716. Springer, 2018.

LV08. Benoît Libert and Damien Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. In Ronald Cramer, editor, *Public Key Cryptography - PKC 2008, 11th International Workshop on Practice and Theory in Public-Key Cryptography, Barcelona, Spain, March 9-12, 2008. Proceedings*, volume 4939 of *Lecture Notes in Computer Science*, pages 360–379. Springer, 2008.

MR07. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

MS17. Steven Myers and Adam Shull. Efficient hybrid proxy re-encryption for practical revocation and key rotation. *IACR Cryptology ePrint Archive*, 2017:833, 2017.

OWA18. OWASP. Cryptographic storage cheat sheet. `https://www.owasp.org/index.php/Cryptographic_Storage_Cheat_Sheet`, 2018. Accessed 9th October 2018.

PCI18. PCI Security Standards Council 2018. Payment Card Industry (PCI) data security standard (version 3.2.1). 2018.

PRSV17. Yuriy Polyakov, Kurt Rohloff, Gyana Sahu, and Vinod Vaikuntanathan. Fast proxy re-encryption for publish/subscribe systems. *ACM Trans. Priv. Secur.*, 20(4):14:1–14:31, 2017.

A Definitions for symmetric PRE

Here we give definitions for the symmetric setting for easier comparison with other re-encryption schemes defined with symmetric keys. We leave out some security games where the change is analogous, but explicitly give those which contains lists to clarify how those lists are updated.

Definition 12. *A* symmetric Proxy Re-Encryption (symPRE) *scheme $s\mathcal{PRE}$ consists of the following algorithms:*

- $s\mathcal{PRE}.\mathsf{Setup}(1^\lambda) \to params$*: Outputs a set of public parameters, including the message space and ciphertext space. Note that params is input to every subsequent algorithm, but we leave it out for compactness of notation. We often omit the* Setup *algorithm for the same reason.*
- $s\mathcal{PRE}.\mathsf{KeyGen}(1^\lambda) \xrightarrow{\$} k$*: Generates a secret key*
- $s\mathcal{PRE}.\mathsf{Enc}(k, m, \ell) \xrightarrow{\$} C$*: Encrypts a message m using a key k, producing a ciphertext at level ℓ. We often leave out ℓ for compactness.*
- $s\mathcal{PRE}.\mathsf{Dec}(k, C) \to m' \cup \perp$*: Decrypts a ciphertext C to produce either an element of the message space m' or the error symbol*
- $s\mathcal{PRE}.\mathsf{ReKeyGen}(k_i, k_j) \xrightarrow{(\$)} \Delta_{i,j} \cup \perp$*: Takes current key k_i and next key k_j and outputs an* update token *$\Delta_{i,j}$, or $\perp$ when $i = j$. This last condition is often left out of constructions for compactness.*
- $s\mathcal{PRE}.\mathsf{ReEnc}(\Delta_{i,j}, C) \xrightarrow{(\$)} C'$*: Takes a ciphertext C under k_i and outputs a new ciphertext C' under k_j.*

A symmetric PRE scheme is correct *if for all* $m \in \mathcal{M}, k \xleftarrow{\$} s\mathcal{PRE}.\mathsf{KeyGen}(1^\lambda)$*:*

$$s\mathcal{PRE}.\mathsf{Dec}(k, s\mathcal{PRE}\mathsf{Enc}(k, m)) \to m$$

and if for all $C \in \mathcal{C}$ *such that* $s\mathcal{PRE}.\mathsf{Dec}(k_i, C) \to m$*:*

$$s\mathcal{PRE}.\mathsf{Dec}(k_j, s\mathcal{PRE}.\mathsf{ReEnc}(\Delta_{i,j}, C)) \to m$$

where $k_i, k_j, \xleftarrow{\$} s\mathcal{PRE}.\mathsf{KeyGen}(1^\lambda)$ *and* $\Delta_{i,j} \leftarrow s\mathcal{PRE}.\mathsf{ReKeyGen}(k_i, k_j)$*.*

```
$\mathsf{SE\text{-}IND\text{-}CPA}^{b,\mathcal{SE}}_{\mathcal{A}}(1^\lambda)$
$\kappa = 0$
$b' \xleftarrow{(\$)} \mathcal{A}^{\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Enc}}, \mathsf{O}_{\mathsf{challenge}}}(1^\lambda)$
return $b' = b$

$\mathsf{O}_{\mathsf{challenge}}(i, m_0, m_1)$
if $|m_0| \neq |m_1|$ :
    return $\perp$
$C \xleftarrow{\$} \mathsf{Enc}(k_i, m_b)$
return $C$
```

Fig. 10: SE-IND-CPA, the symmetric variant of the PKE-IND-CPA game. Again, we give a multi-key, multi-challenge variant.

We present oracles commonly used in games for symmetric PRE schemes in Figure 11.

```
$\mathsf{O}_{\mathsf{KeyGen}}(1^\lambda)$
$\kappa = \kappa + 1$
$k_\kappa \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$\mathcal{DRG}.\mathsf{add}\{v_\kappa\}$

$\mathsf{O}_{\mathsf{Corrupt}}(i)$
$\mathcal{K}_{\mathsf{corrupted}}.\mathsf{add}\{k_i\}$
return $k_i$

$\mathsf{O}_{\mathsf{Enc}}(i, m)$
$C \xleftarrow{\$} \mathsf{Enc}(k_i, m)$
$\boxed{\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{(i, C)\}}$
return $C$

$\mathsf{O}_{\mathsf{ReKeyGen}}(i, j)$
$\Delta_{i,j} \xleftarrow{(\$)} \mathsf{ReKeyGen}(k_i, k_j)$
$\mathcal{T}_{\mathsf{honest}}.\mathsf{add}\{(i, j, \Delta_{i,j})\}$
$\mathcal{DRG}.\mathsf{add}\{\vec{e}_{i,j}\}$
return $\Delta_{i,j}$
```

Fig. 11: Common oracles used in security games for symmetric PRE.

Definition 13. *An encryption scheme* $\mathcal{SE}$ *is* ϵ-Indistinguishable against Chosen Plaintext Attacks (ϵ-SE-IND-CPA-secure) *if for all* PPT *adversaries* $\mathcal{A}$*:*

$$\left|\Pr\left[\mathsf{SE\text{-}IND\text{-}CPA}^{0,\mathcal{SE}}_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{SE\text{-}IND\text{-}CPA}^{1,\mathcal{SE}}_{\mathcal{A}}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

where $\mathsf{SE\text{-}IND\text{-}CPA}^{b,\mathcal{SE}}_{\mathcal{A}}(1^\lambda)$ *is defined in Figure 10. If* ϵ *is negligible as parameterised by the security parameter* λ*, then we say the scheme is*Indistinguishable against Chosen Plaintext Attacks (sym-CPA-secure)*.*
A symmetric PRE scheme $s\mathcal{PRE}$ *is* (ϵ-sym-CPA-secure) *if the encryption scheme given by* $s\mathcal{PRE} = \{s\mathcal{PRE}.\mathsf{KeyGen}, s\mathcal{PRE}.\mathsf{Enc}, s\mathcal{PRE}.\mathsf{Dec}\}$ *is* ϵ*-SE-IND-CPA-secure.*

Definition 14. *A symmetric PRE scheme* $s\mathcal{PRE}$ *is* ϵ-source-hiding *if for all* PPT *adversaries* $\mathcal{A}$*:*

$$\left|\Pr\left[\mathsf{symSH}^{0,s\mathcal{PRE}}_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{symSH}^{1,s\mathcal{PRE}}_{\mathcal{A}}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

where symSH *is defined in Figure 12. If* ϵ *is negligible as parameterised by the security parameter* λ*, then we say* $s\mathcal{PRE}$ *is* source-hiding*.*

```
$\mathsf{symSH}_{\mathcal{A}}^{b,s\mathcal{PRE}}(1^\lambda)$
$k_0 \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda), k_1 \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$\Delta_{0,1} \xleftarrow{\$} \mathsf{ReKeyGen}(k_0, k_1)$
$b' \xleftarrow{(\$)} \mathcal{A}^{\mathsf{O}_{\mathsf{challenge}}}(1^\lambda, k_0, k_1, \Delta_{0,1})$
$\mathbf{return}\ b' = b$
```

```
$\mathsf{O}_{\mathsf{challenge}}(m^*, \ell^*)$
$\mathbf{if}\ \ell^* > L - 1 : \mathbf{return}\ \perp$
$C \leftarrow \mathsf{Enc}(k_0, m^*, \ell^*)$
${C'}^{(0)} \leftarrow \mathsf{ReEnc}(\Delta_{0,1}, C)$
${C'}^{(1)} \xleftarrow{\$} \mathsf{Enc}(k_1, m^*, \ell^* + 1)$
$\mathbf{return}\ (C, {C'}^{(b)})$
```

Fig. 12: Experiments for the symmetric source-hiding property. L is the number of times a ciphertext can be re-encrypted without breaking the correctness conditions.

```
$\mathsf{SymPostComp}_{\mathcal{A}}^{b,s\mathcal{PRE}}(1^\lambda)$
$\mathcal{K}_{\mathsf{chal}}, \mathcal{K}_{\mathsf{corrupted}}, \mathcal{C}_{\mathsf{honest}}, \mathcal{C}_{\mathsf{chal}}, \mathcal{T}_{\mathsf{honest}}, \mathcal{DRG} = \emptyset$
$\kappa = 0, \mathsf{called} = \mathsf{false}$
$state \xleftarrow{(\$)} \mathcal{A}_0^{\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Corrupt}}}(1^\lambda)$
$b' \xleftarrow{(\$)} \mathcal{A}_1^{\mathsf{O}_{\mathsf{Enc}}, \mathsf{O}_{\mathsf{ReKeyGen}}, \mathsf{O}_{\mathsf{ReEnc}}, \mathsf{O}_{\mathsf{challenge}}}(1^\lambda, state)$
$\mathcal{K}_{\mathsf{chal}} \leftarrow \mathsf{UpdateChallengeKeys}(\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG})$
$\mathbf{if}\ \mathcal{K}_{\mathsf{chal}} \cap \mathcal{K}_{\mathsf{corrupted}} \neq \emptyset : \mathbf{return}\ \perp$
$\mathbf{return}\ b' = b$
```

```
$\mathsf{O}_{\mathsf{ReEnc}}(C, i, j, [\Delta_{i,j}])$
$\mathbf{if}\ \Delta_{i,j}$ given :
    $\mathbf{if}\ (i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}} : \mathbf{return}\ \perp$
$\mathbf{else}\ : \Delta_{i,j} \leftarrow \mathsf{ReKeyGen}(k_i, k_j)$
$\mathbf{if}\ (i, C) \notin \mathcal{C}_{\mathsf{honest}} : \mathbf{return}\ \perp$
$C' \xleftarrow{\$} \mathsf{ReEnc}(\Delta_{i,j}, C)$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{(j, C')\}$
$\mathbf{if}\ (i, C) \in \mathcal{C}_{\mathsf{chal}}$ :
    $\mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{(j, C')\}, \mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{k_j\}$
$\mathbf{return}\ C'$
```

```
$\mathsf{O}_{\mathsf{challenge}}(C_0, C_1, i, j, \Delta_{i,j})$
$\mathbf{if}\ |C_0| \neq |C_1|\ \mathbf{OR}\ \mathsf{called} = \mathsf{true} : \mathbf{return}\ \perp$
$\mathbf{if}\ (i, C_0), (i, C_1) \notin \mathcal{C}_{\mathsf{honest}}\ \mathbf{OR}\ (i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}} : \mathbf{return}\ \perp$
$C' \xleftarrow{\$} \mathsf{ReEnc}(\Delta_{i,j}, C_b)$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{(j, C')\}, \mathcal{C}_{\mathsf{chal}}.\mathcal{A}(j, C'), \mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{k_j\}$
$\mathsf{called} \leftarrow \mathsf{true}$
$\mathbf{return}\ C'$
```

Fig. 13: The symmetric variant of PostComp game.

Definition 15. *A symmetric PRE scheme $s\mathcal{PRE}$ is said to be* ϵ-post-compromise secure (ϵ-PCS) *if for all* PPT *adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$:*

$$\left| \Pr\left[\mathsf{SymPostComp}_{\mathcal{A}}^{0,s\mathcal{PRE}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{SymPostComp}_{\mathcal{A}}^{1,s\mathcal{PRE}}(1^\lambda) = 1\right] \right| \leq \epsilon,$$

where $\mathsf{SymPostComp}_{\mathcal{A}}^{b,s\mathcal{PRE}}(1^\lambda)$ *is defined in Figure 13.*

If ϵ is negligible as parameterised by the security parameter, then we say the scheme is Post-Compromise Security (PCS)*.*

We give the equivalent lemma to Lemma 5 for symmetic PRE.

Lemma 9. *IND-UPD $\not\Longrightarrow$ symmetric PCS.*

Proof. Because RISE is not unidirectional, it is not compromise secure by Lemma 2. We present the following full counterexample to demonstrate this. RISE [LT18] is proven to be IND-UPD. Here we adapt RISE for general proxy re-encryption as this fits more with our notation, but we observe that the original construction defined as an updatable encryption scheme is also sufficient for the proof.

- $\mathsf{RISE}'.\mathsf{KeyGen}(1^\lambda) : x \xleftarrow{\$} \mathbb{Z}_q^*, (\mathsf{pk}, \mathsf{sk}) =, (x, g^x)$
- $\mathsf{RISE}'.\mathsf{Enc}(\mathsf{pk}, m) : r \xleftarrow{\$} \mathbb{Z}_q^*, C \leftarrow (\mathsf{pk}^r, g^r \cdot m)$
- $\mathsf{RISE}'.\mathsf{Dec}(\mathsf{sk}, m) : m' \leftarrow C_1 \cdot C_0^{-1/\mathsf{sk}}$
- $\mathsf{RISE}'.\mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{sk}_j) : \Delta_{i,j} = (\mathsf{sk}_j/\mathsf{sk}_i, \mathsf{pk}_j) = (x_j/x_i, g^{x_j})$
- $\mathsf{RISE}'.\mathsf{ReEnc}(\Delta_{i,j}, C) : (\Delta, y') = \Delta_{i,j}, r' \xleftarrow{\$} \mathbb{Z}_q^*, C' \leftarrow (C_0^\Delta \cdot {y'}^{r'}, C_1 \cdot g^{r'})$

Given $\Delta_{i,j} = (x_j/x_i, g^{x_j})$, and x_i, $\mathcal{A}$ can compute x_j and use this to decrypt the challenge ciphertext.

B Asymmetric IND-UPD

IND-UPD [LT18, definition 3] is a post-compromise security notion for updatable encryption schemes, which are a variant of symmetric PCS schemes. The main difference in updatable encryption is that key updates happen sequentially from k_i to k_{i+1}, meaning the re-encryption graph is always a chain. Another notable difference between updatable encryption schemes and PRE schemes is that they contain an algorithm Next, which generates a new key and an update token from the old key to the new one as opposed to defining these functions separately.

Here we adapt IND-UPD for the public-key setting.

```
$\mathsf{pkIND\text{-}UPD}^{b,\mathcal{PRE}}_{\mathcal{A}}(1^\lambda)$
$e = 0, \mathcal{DRG} = (v_0)$
$(\mathsf{sk}_0, \mathsf{pk}_0) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$\mathcal{K}_{\mathsf{corrupted}}, \mathcal{C}_{\mathsf{honest}}, \mathcal{K}_{\mathsf{chal}} = \emptyset$
$\mathsf{called} = 0$
$state \xleftarrow{(\$)} \mathcal{A}_0^{\mathsf{O_{Next}}, \mathsf{O_{Enc}}, \mathsf{O_{Corrupt}}, \mathsf{O^{pkIU}_{LearnTok}}, \mathsf{O^{pkIU}_{ReEnc}}}(1^\lambda, \mathsf{pk}_0)$
$b' \xleftarrow{(\$)} \mathcal{A}_1^{\mathsf{O_{Next}}, \mathsf{O_{Enc}}, \mathsf{O_{Corrupt}}, \mathsf{O^{pkIU}_{LearnTok}}, \mathsf{O^{pkIU}_{LearnChal}}, \mathsf{O^{pkIU}_{ReEnc}}, \mathsf{O^{pkIU}_{challenge}}}(1^\lambda, state)$
$\mathcal{K}_{\mathsf{chal}} \leftarrow \mathsf{UpdateChallengeKeys}(\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG})$
**if** $\mathcal{K}_{\mathsf{chal}} \cap \mathcal{K}_{\mathsf{corrupted}} \neq \emptyset$ : **return** $0$
**return** $b'$
```

```
$\mathsf{O_{Next}}()$
$e \leftarrow e + 1$
$\mathcal{DRG}.\mathsf{add}\{v_e\}$
$(\mathsf{pk}_e, \mathsf{sk}_e) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$\Delta_{e-1,e} \xleftarrow{(\$)} \mathsf{ReKeyGen}(\mathsf{sk}_{e-1}, \mathsf{pk}_e, )$
**if** $\mathsf{called} = 1$ :
    $C^*_e \xleftarrow{(\$)} \mathsf{ReEnc}(\Delta_{e-1,e}, C^*_{e-1})$
**return** $\mathsf{pk}_e$
```

```
$\mathsf{O^{pkIU}_{LearnTok}}(i)$
**if** $i = \bar{e}$ : **return** $\perp$
**if** $\mathcal{PRE}$ is unidirectional :
    $\mathcal{DRG}.\mathsf{add}\{\overrightarrow{e}_{i-1,i}\}$ // *directed edge*
**elseif** $\mathcal{PRE}$ is bidirectional :
    $\mathcal{DRG}.\mathsf{add}\{e_{i,i-1}\}$ // *undirected edge*
**return** $\Delta_{i-1,i}$
```

```
$\mathsf{O^{pkIU}_{LearnChal}}()$
$\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\mathsf{sk}_e\}$
**return** $C^*_e$
```

```
$\mathsf{O^{pkIU}_{ReEnc}}(C)$
**if** $(e-1, C) \notin \mathcal{C}_{\mathsf{honest}}$ : **return** $\perp$
**if** $\mathsf{ReEnc}$ is deterministic, $\mathsf{called} = 1$ **AND** $e = \bar{e}$ :
    **if** $C = \bar{C}_0$ **OR** $C = \bar{C}_1$ : **return** $\perp$
$C' \xleftarrow{(\$)} \mathsf{ReEnc}(\Delta_{e-1,e}, C)$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{e, C'\}$
**return** $C'$
```

```
$\mathsf{O^{pkIU}_{challenge}}(C_0, C_1)$
**if** $\mathsf{called} = 1$ **OR** $|C_0| \neq |C_1|$ : **return** $\perp$
**if** $(e, C_0), (e, C_1) \notin \mathcal{C}_{\mathsf{honest}}$ : **return** $\perp$
$e \leftarrow= e + 1$
$\mathcal{DRG}.\mathsf{add}\{v_e\}$
$(\mathsf{pk}_e, \mathsf{sk}_e) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$\bar{e} = e, \bar{C}_0 = C_0, \bar{C}_1 = C_1$
$\Delta_{e-1,e} \xleftarrow{(\$)} \mathsf{ReKeyGen}(\mathsf{sk}_{e-1}, \mathsf{pk}_e)$
$C^*_e \xleftarrow{(\$)} \mathsf{ReEnc}(\Delta_{e-1,e}, C_b)$
$\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\mathsf{sk}_e\}$
**return** $(\mathsf{pk}_e, C^*_e)$
```

Fig. 14: The $\mathsf{pkIND\text{-}UPD}$ game, based on IND-UPD [LT18] adapted to the public-key setting.

Definition 16. *A PRE scheme $\mathcal{PRE}$ is said to be* (selectively) ϵ-pkIND-UPD-secure *if for all* PPT *adversaries* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$*:*

$$\left|\Pr\left[\mathsf{pkIND\text{-}UPD}_{\mathcal{A}}^{0,\mathcal{PRE}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{pkIND\text{-}UPD}_{\mathcal{A}}^{1,\mathcal{PRE}}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

where $\mathsf{pkIND\text{-}UPD}_{\mathcal{A}}^{b,\mathcal{PRE}}(1^\lambda)$ *is defined in Figure 14.*

If ϵ *is negligible as parameterised by the security parameter, then we say the scheme is* (selectively) pkIND-UPD-secure.

C BV-PRE

We describe BV-PRE [PRSV17] in Figure 15. Note that BV-PRE is not fully public-key as it uses an additional 'public' key `pk'_B for the target key, which cannot be made public without exposing sk_A to anyone with the update token.

```
Setup(1^λ)
    Choose positive integers $q, n$
    Choose plaintext modulus $p$
    Choose key switching window $r$
    return $pp = (q, n, p.r)$
```

```
KeyGen(1^λ)
    $a \xleftarrow{\$} \mathcal{U}_q$
    $s, e \xleftarrow{\$} \chi_e$
    $b = a \cdot s + pe$
    $\mathsf{sk} = s, \mathsf{pk} = (a, b)$
    return $(\mathsf{sk}, \mathsf{pk})$
```

```
Enc(pk, m)
    $(a, b) \leftarrow \mathsf{pk}$
    $v, e_0, e_1 \xleftarrow{\$} \chi_e$
    $C_0 = b \cdot v + pe_0 + m$
    $C_1 = a \cdot v + pe_1$
    return $C = (C_0, C_1)$
```

```
Dec(sk, c)
    $s \leftarrow \mathsf{sk}, (C_0, C_1) \leftarrow C$
    $m' = C_0 - s \cdot C_1 \mod p$
    return $m'$
```

```
Preprocess(1^λ, sk_B)
    $s_b \leftarrow \mathsf{sk}_b$
    for $i \in \{0, 1, \ldots, \lfloor \log_2(q)/r \rfloor\}$ :
        $\beta_i \xleftarrow{\$} \mathcal{U}_q$
        $e_i \xleftarrow{\$} \chi_e$
        $\theta_i = \beta_i \cdot s_B + pe_i$
    return $\text{`pk'}_B = \{(\beta_i, \theta_i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor}$
```

```
ReKeyGen(sk_A, 'pk'_B)
    $s_A \leftarrow \mathsf{sk}_A$
    $\{(\beta_i, \theta_i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor} \leftarrow \text{`pk'}_B$
    for $i \in \{0, 1, \ldots, \lfloor \log_2(q)/r \rfloor\}$ :
        $\gamma_i = \theta_i - s_A \cdot (2^r)^i$
    $\Delta_{A \to B} = \{(\beta_i, \gamma_i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor}$
    return $\Delta_{A,B}$
```

```
ReEnc(Δ_{A,B}, C)
    $\{(\beta_i, \gamma_i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor} \leftarrow \Delta_{A,B}$
    $(C_0, C_1) \leftarrow C$
    $C'_0 = C_0 + \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (C_1^{(i)} \cdot \gamma_i)$
    $C'_1 = \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (C_1^{(i)} \cdot \beta_i)$
    return $C' = (C'_0, C'_1)$
```

Fig. 15: BV-PRE [PRSV17]. $\mathcal{U}_q$ is the discrete uniform distribution over R_q and χ_e is a B_e-bounded discrete Gaussian error distribution.

D Key Resampling and its Implications

To prevent the need for a 'public' key 'pk'$_B$ to create a secure PRE scheme as in BV-PRE, we a key resampling technique. This means our scheme is public-key in the traditional sense where no extra keys are required for re-encryption.

D.1 Key Resampling

The key resampling technique [BV11], can be found in Figure 16. We use this in our construction in Figure 8 to obtain PCS.

```
ReSample(pk)
$(a, b) \leftarrow \mathsf{pk}$
$v, e' \xleftarrow{\$} \chi_e$
$e'' \xleftarrow{\$} \chi_e$
$\bar{a} = av + pe'$
$\bar{b} = bv + pe''$
return $(\bar{a}, \bar{b})$
```

Fig. 16: Key resampling technique [BV11].

Observe that $\bar{b} = \bar{a}s + p(ev + e'' - e's)$ and therefore $(\bar{a}, \bar{b})$ is also suitable for encryption. $(\bar{a}, \bar{b})$ is computationally indistinguishable from a freshly generated public key for s. This leads us to define the a new security game for resampled keys in Figure 17, where given a public key (a, b), the challenger either returns a resampled key or a uniformly sampled pair. It is similar to RLWE challenges for fresh keys.

D.2 Implications of resampling

```
ReSample-ncRLWE$^d_{\mathcal{A}}(1^\lambda)$
$\kappa = 0$
$d' \leftarrow \mathcal{A}^{\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Corrupt}}}(1^\lambda)$
return $(d' = d)$

$\mathsf{O}_{\mathsf{KeyGen}}()$
$\kappa \leftarrow \kappa + 1$
$a \xleftarrow{\$} \mathcal{U}_q$
$s, e \xleftarrow{\$} \chi_e$
$b = a \cdot s + pe$
$\mathsf{pk}_\kappa = (a, b), \mathsf{sk}_\kappa = s$
return $\mathsf{pk}_\kappa$

$\mathsf{O}_{\mathsf{challenge}}(i)$
$(a, b) \leftarrow \mathsf{pk}_i$
$v, e' \xleftarrow{\$} \chi_e, e'' \xleftarrow{\$} \chi_e$
$(\bar{a}_0, \bar{b}_0) = (av + pe', bv + pe'')$
$(\bar{a}_1, \bar{b}_1) \xleftarrow{\$} \mathcal{U}_q^2$
return $(\bar{a}_d, \bar{b}_d)$
```

Fig. 17: A variant of RLWE for resampling. Note that the challenge oracle can be queried more than once.

Lemma 10. *Let* ReSample-ncRLWE$^b_{\mathcal{A}}(1^\lambda)$ *be defined as in Figure 17 and let* κ *be the number of keys generated in* ReSample-ncRLWE$^b_{\mathcal{A}}(1^\lambda)$*. Then for all* PPT *adversaries* $\mathcal{A}$ *there exists a RLWE distinguisher* $\mathcal{D}$ *such that:*

$$\left|\Pr\left[\mathsf{ReSample\text{-}ncRLWE}^0_{\mathcal{A}} = 1\right] - \Pr\left[\mathsf{ReSample\text{-}ncRLWE}^1_{\mathcal{A}} = 1\right]\right| \leq (\kappa + Q) \cdot \mathbf{Adv}RLWE_{\phi,q,\chi_e}(\mathcal{D}),$$

where Q is the number of calls that $\mathcal{A}$ makes to $\mathsf{O}_{\mathsf{challenge}}$.

Proof. Let $\mathcal{D}$ receive an RLWE challenge, where either the sample is genuine $d = 0$ or it is random $d = 1$. We demonstrate via a hybrid argument that $\mathcal{D}$ can simulate ReSample-RLWE.

Let $\mathsf{H}_0 :=$ ReSample-RLWE, and let H_i be identical to H_{i-1} except that the first i public keys have been replaced with uniform random values. Suppose there exists an adversary $\mathcal{B}$ that can distinguish between H_{i-1} and H_i with advantage ϵ. We demonstrate how an RLWE distinguisher $\mathcal{D}$ that can break RLWE with the same advantage by simulating H_i and running $\mathcal{B}$ as a subroutine. $\mathcal{D}$ simulates H_{i-d} by responding to calls $\mathcal{B}$ makes to $\mathsf{O}_{\mathsf{KeyGen}}$ as follows:

- if $k < i$ then $\mathcal{D}$ returns $(u_0, u_1) \xleftarrow{\$} \mathcal{U}_q^2$,
- if $k = i$ $\mathcal{D}$ returns an RLWE challenge (a, b),
- if $k > i$ then $\mathcal{D}$ returns a genuine RLWE sample.

Then $\mathcal{D}$ returns $d' = d'_{\mathcal{A}}$. The maximal advantage in distinguishing between H_i and H_{i-1} is therefore $\mathbf{Adv}RLWE_{\phi,q,\chi_e}(\mathcal{D})$.

Now consider a further hybrid argument $\bar{\mathsf{H}}_0 \ldots \bar{\mathsf{H}}_Q$ where $\bar{\mathsf{H}}_0$ is identical to H_κ and $\bar{\mathsf{H}}_i$ is identical to $\bar{\mathsf{H}}_{i-1}$ except that the ith output of $\mathsf{O}_{\mathsf{challenge}}$ is replaced with $(\bar{a}, \bar{b}) \xleftarrow{\$} \mathcal{U}_q^2$.

Because all public keys are now random elements of $\mathcal{U}_q$, we note that we can replace $\bar{a}, \bar{b}$ with random elements of $\mathcal{U}_q$, by the RLWE assumption. Alternatively, this can be seen in the same way as the IND-CPA security of the underlying encryption scheme [PRSV17, proof of Theorem 5.1], as ReSample is the same as encrypting 0. This means that the maximal advantage in distinguishing between $\bar{\mathsf{H}}_i$ and $\bar{\mathsf{H}}_{i-1}$ is $\mathbf{Adv}RLWE_{\phi,q,\chi_e}(\mathcal{D})$

Overall, we get that the number of times we rely on RLWE is $\kappa + Q$. This completes the proof. □

To get to selective key corruption models, we need to consider a variant of ReSample-ncRLWE where $\mathcal{A}$ is also given access to a key corruption oracle. We describe this variant, which we call ReSample-RLWE, in Figure 18.

```
ReSample-RLWE$^d_{\mathcal{A}}(1^\lambda)$
$\kappa = 0$
$\mathcal{K}_{\mathsf{corrupted}}, \mathcal{K}_{\mathsf{chal}} = \emptyset$
$d' \leftarrow \mathcal{A}^{\mathsf{O}_{\mathsf{KeyGen}}, \mathsf{O}_{\mathsf{Corrupt}}, \mathsf{O}_{\mathsf{challenge}}}(1^\lambda)$
return $(d' = d)$
```

```
$\mathsf{O}_{\mathsf{KeyGen}}()$
$\kappa \leftarrow \kappa + 1$
$a \xleftarrow{\$} \mathcal{U}_q$
$s, e \xleftarrow{\$} \chi_e$
$b = a \cdot s + pe$
$\mathsf{pk}_\kappa = (a, b)$
$\mathsf{sk}_\kappa = s$
return $\mathsf{pk}_\kappa$
```

```
$\mathsf{O}_{\mathsf{Corrupt}}(i)$
if $i \in \mathcal{K}_{\mathsf{chal}}$ :
    return $\perp$
$\mathcal{K}_{\mathsf{corrupted}}.\mathsf{add}\{()i)$
return $\mathsf{sk}_i$
```

```
$\mathsf{O}_{\mathsf{challenge}}(i)$
if $i \in \mathcal{K}_{\mathsf{corrupted}}$ :
    return $\perp$
$(a, b) \leftarrow \mathsf{pk}_i$
$v, e' \xleftarrow{\$} \chi_e, e'' \xleftarrow{\$} \chi_e$
$(\bar{a}_0, \bar{b}_0) = (av + pe', bv + pe'')$
$(\bar{a}_1, \bar{b}_1) \xleftarrow{\$} \mathcal{U}_q^2$
$\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{()i)$
return $(\bar{a}_d, \bar{b}_d)$
```

Fig. 18: A variant of RLWE for resampling with key corruptions. As in ReSample-ncRLWE the challenge oracle can be queried more than once.

To prove ReSample-RLWE$^d_{\mathcal{A}}(1^\lambda)$, we revisit an observation made in [FKKP18], where Fuchsbauer et al. note that when proving the security of PRE schemes, it is sufficient to only need to replace values connected to the *challenge graph* – the subgraph of the re-encryption graph which is reachable from the challenge node. We believe this is logical, as the corruption of independently generated keys with no links to challenge keys should not affect the security of the challenge keys. We generalise this observation assumption as follows: if $\{q\}$ is the set of queries made in a security game Game that are independent of any challenge queries, then $\{q\}$ can give no advantage to an adversary hoping to win Game. We call this the *challenge independence assumption.* This is relevant for key corruptions as, if the adversary corrupts keys which are never related to any challenge keys, the challenge independence assumption means these keys give the adversary no additional advantage. This principle is similar to the intuition of how re-encryption simulatability and source-hiding imply IND-HRA, as replacing re-encryptions with ciphertexts independent of the challenge secret key cannot give an advantage in learning challenge keys.

Lemma 11. *Let* $\mathsf{ReSample\text{-}RLWE}^b_{\mathcal{A}}(1^\lambda)$ *be defined as in Figure 18 and let* κ_{chal} *be the number of keys on which challenges are called by* $\mathcal{A}$[8]*. Then an adversary who wins* $\mathsf{ReSample\text{-}ncRLWE}$ *when* κ_{chal} *keys are generated can win* $\mathsf{ReSample\text{-}RLWE}$ *with the same advantage by the challenge independence assumption.*

Proof. We note that all keys are generated independently, and that in $\mathsf{ReSample\text{-}RLWE}$ keys either become challenge keys or corrupted keys. Therefore corrupted keys are independent of the challenge keys, so the challenge independence assumption applies, meaning key corruption gives no advantage in winning the original game. We conclude that $\mathsf{ReSample\text{-}ncRLWE}$ implies $\mathsf{ReSample\text{-}RLWE}$, and so we can therefore replace all RLWE samples with uniform random values.

Combining Lemma 10 and Lemma 11 gives the following result.

Lemma 12. *Then for all* PPT *adversaries* $\mathcal{A}$*:*

$$\left|\Pr\left[\mathsf{ReSample\text{-}RLWE}^0_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ReSample\text{-}RLWE}^1_{\mathcal{A}}(1^\lambda) = 1\right]\right| \\ \leq (\kappa + Q) \cdot \mathbf{Adv} RLWE_{\phi,q,\chi_e}(\mathcal{D}), \quad (3)$$

where κ *is the number of key pairs generated in* $\mathsf{ReSample\text{-}ncRLWE}^d_{\mathcal{A}}(1^\lambda)$ *and* Q *is the number of calls that* $\mathcal{A}$ *makes to* $\mathsf{O}_{\mathsf{challenge}}$*.*

E Analysis of pcBV-PRE

In this section, we present proofs concerning pcBV-PRE. We first demonstrate correctness and calculate the correctness bound, before giving a full proof of Theorem 3 that pcBV-PRE has PCS.

E.1 Correctness of pcBV-PRE

Much of the analysis used to argue correctness from [PRSV17] holds for our modified scheme in Figure 8. Therefore, we keep the discussion fairly brief and refer to [PRSV17] for full details.

Recall that the decryption algorithm computes $C_0 - C_1 s$ and then performs a reduction modulo p. In the case that (C_0, C_1) is a ciphertext produced by the Enc algorithm, we have that

$$C_0 - C_1 s = p(ev + e_0 + e_1 s) + m, \quad (4)$$

[8] Without loss of generality, we assume that by the end of the game, all keys are either corrupted or challenged.

so decryption is successful when $p(ev + e_0 + e_1 s) + m$ (i.e. the error plus the message) does not wrap around modulo q. However, the correctness condition of decryption is different when considering ciphertexts output by the ReEnc algorithm. To see how, let $(C_0', C_1') \xleftarrow{\$} \mathsf{ReEnc}(\varDelta_{A\to B}, (C_0, C_1))$. We also note that for $(\bar{a}, \bar{b}) \leftarrow \mathsf{ReSample}(i, j)$ where $b = as + pe$, we get

$$\bar{b} - s\bar{a} = p(ev + e'' - e's). \tag{5}$$

Using notation consistent with Figure 16, we have that

$$\begin{aligned} C_0' - C_1' s_B &= C_0 + \sum_{i=0}^{\lfloor \log(q)/r \rfloor} C_1^{(i)} \left(\theta_i - s_A (2^r)^i - s_B \beta_i\right) + (\theta^{\text{proxy}} - \beta^{\text{proxy}} s_B) \\ &= C_0 - C_1 s_A + \sum_{i=0}^{\lfloor \log(q)/r \rfloor} C_1^{(i)} \left(\theta_i - s_B \beta_i\right) + (\theta^{\text{proxy}} - \beta^{\text{proxy}} s_B) \\ &\overset{(5)}{=} C_0 - C_1 s_A + \sum_{i=0}^{\lfloor \log(q)/r \rfloor} C_1^{(i)} \cdot p(ev_i + e_i' s_B + e_i'') \\ &\qquad\qquad + p(ev^{\text{proxy}} + e'^{\text{proxy}} s_B + e''^{\text{proxy}}), \end{aligned}$$

where all arithmetic is done over the integers modulo q. This shows that the error grows by an additive term of pE on each re-encryption where

$$E = \sum_{i=0}^{\lfloor \log(q)/r \rfloor} C_1^{(i)} \cdot (ev_i + e_i' s_B + e_i'') + (ev^{\text{proxy}} + e'^{\text{proxy}} s_B + e''^{\text{proxy}}).$$

Assume that χ_e is (B, δ)-bounded for some small δ (we quantify these values later). Then we have that the noise grows by at most

$$p||E||_\infty \leq pn(2^r - 1)(\lfloor \log(q)/r \rfloor + 1)(2B^2 n + B) + p(2B^2 n + B) =: G(n, q, r, B) \tag{6}$$

on each re-encryption taking into consideration the multiplication of degree $n-1$ polynomials. Therefore, after ℓ re-encryptions, the noise has grown by an additive term of size ℓG. Therefore, the condition for successful decryption after ℓ re-encryptions is

$$p(2B^n + B) + \ell G(n, q, r, B) + p/2 \leq q/2 \tag{7}$$

where G is defined in Equation (6). Note that if our error distribution has $\sigma > \omega(\log n)$, then we can set $B = \sigma\sqrt{n}$ and $\delta = 2^{-n+1}$ [MR07, LATV13]. In order to choose parameters of the system, one needs to ensure that Equation (7) is satisfied. Note that this analysis is a worst-case one. It is shown in [PRSV17] that the central limit theorem can be used to essentially alter the form of B and change all occurrences of n into occurrences of $\sqrt{n}$ in Equation (7) by allowing for a tunable failure probability. We omit this more practical method of correctness analysis for brevity.

E.2 Full proof of Theorem 3

As conveyed in Section 5.3, we separate the proof of Theorem 3 into two theorems, the first of which proves a variant of the game that does not consider honest re-encryptions between uncorrupted and corrupted keys, and the second which does.

Theorem 6. *Let* CPA-PostComp *be a variant of the* PostComp *game where no re-encryptions from uncorrupted to corrupted keys are permitted whatsoever. Then for all* PPT *adversaries $\mathcal{A}$, there exists an RLWE distinguisher $\mathcal{D}$ such that:*

$$\left|\Pr\left[\mathsf{CPA\text{-}PostComp}_{\mathcal{A}}^{0,\mathsf{pcBV\text{-}PRE}}(1^\lambda) = 1\right] = \Pr\left[\mathsf{CPA\text{-}PostComp}_{\mathcal{A}}^{1,\mathsf{pcBV\text{-}PRE}}(1^\lambda) = 1\right]\right|$$
$$\leq (\kappa + (Q_{RK} + Q_{RE\perp} + 1)(\lfloor \log_2(q)/r \rfloor + 1) + 1) \cdot \mathbf{Adv}RLWE_{\phi,q,\chi_e}(\mathcal{D}), \quad (8)$$

where Q_{RK} is the number of queries $\mathcal{A}$ makes to $\mathsf{O}_{\mathsf{ReKeyGen}}(i,j)$ *where sk_i and* sk_i *are uncorrupted, and $Q_{RE\perp}$ is the number of queries $\mathcal{A}$ makes to* $\mathsf{O}_{\mathsf{ReKeyGen}}(C,i,j,\Delta_{i,j})$ *where* sk_i *and* sk_j *are uncorrupted and $\Delta_{i,j} = \perp$.*
Because $\mathcal{A}$ is PPT*, we note that Q_{RK} and $Q_{RE\perp}$ are both polynomial, meaning that the overall advantage in distinguishing b is negligible by the RLWE assumption.*

Proof. We prove this theorem using a sequence of game hops as described in Lemmas 13 to 15.

Lemma 13. *Let* $\mathsf{Game}_0^{b,\mathsf{pcBV\text{-}PRE}}$ *be the same as* $\mathsf{CPA\text{-}PostComp}^{b,\mathsf{pcBV\text{-}PRE}}$. *Let* Game_1 *is identical to* Game_0 *except that all update tokens between uncorrupted keys* $\left(\{(\beta_i, \gamma_i = \theta_i - \mathsf{sk}_A \cdot (2^r)^i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor}, \mathsf{pk}_B\right)$ *are replaced with* $\left(\{(\beta_i, \theta_i)\}_{i=0}^{\lfloor \log_2(q)/r \rfloor}, \mathsf{pk}_B\right)$.
Then for $b \in \{0,1\}$ and for all PPT *adversaries $\mathcal{A}$:*

$$\left|\Pr\left[1 \leftarrow \mathsf{Game}_0^{b,\mathsf{pcBV\text{-}PRE}}(\mathcal{A})\right] - \Pr\left[1 \leftarrow \mathsf{Game}_1^{b,\mathsf{pcBV\text{-}PRE}}(\mathcal{A})\right]\right|$$
$$\leq (\kappa + (Q_{RK} + Q_{RE\perp})(\lfloor \log_2(q)/r \rfloor + 1)) \cdot \mathbf{Adv}RLWE_{\phi,q,\chi_e}(\mathcal{D}).$$

Proof. Note that if $x = r + u \in \mathcal{R}_q^2$, then if $r \xleftarrow{\$} \mathcal{U}_q^2$, then x is indistinguishable from $x' \xleftarrow{\$} \mathcal{U}_q^2$ even if u is known, and vice versa. This means we can replace $\{(\beta_\iota, \gamma_i = \theta_\iota - \mathsf{sk}_i)_{\iota=0}^{\lfloor \log_2(q)/r \rfloor}\}$ where $\{(\beta_\iota, \theta_\iota) \xleftarrow{\$} \mathsf{ReSample}(\mathsf{pk}_j)\}_{\iota=0}^{\lfloor \log_2(q)/r \rfloor}$ with uniform random values if and only if we can replace calls to $\mathsf{ReSample}(\mathsf{pk}_j)$ with uniform random values. This means it will be possible to simulate update tokens $\Delta_{i,j}$ when sk_i and sk_j are unknown.

Let $d \in \{0,1\}$. We demonstrate how an adversary $\mathcal{B}$ for $\mathsf{ReSample\text{-}RLWE}_{\mathcal{B}}^d(1^\lambda)$ can simulate $\mathsf{Game}_d^{b,\mathsf{pcBV\text{-}PRE}}$:

- $\mathcal{B}$ samples $b \xleftarrow{\$} \{0,1\}$
- When $\mathcal{A}$ calls $\mathsf{O}_{\mathsf{KeyGen}}^{\mathsf{Game}_d}$ and $\mathsf{O}_{\mathsf{Corrupt}}^{\mathsf{Game}_d}$, $\mathcal{B}$ forwards these queries the equivalent oracles in ReSample-RLWE.
- When $\mathcal{A}$ calls $\mathsf{O}_{\mathsf{ReKeyGen}}^{\mathsf{Game}_d}(i,j)$:
  - If sk_i is not corrupted and sk_j is corrupted, $\mathcal{B}$ returns $\perp$.
  - If sk_i is corrupted, $\mathcal{B}$ returns $\Delta_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$.
  - If sk_i and sk_j are not corrupted, $\mathcal{B}$ uses the ReSample-RLWE challenger to obtain $\{(\beta_\iota, \theta_\iota) \xleftarrow{\$} \mathsf{O}_{\mathsf{challenge}}^{\mathsf{rrlwe}}(j)\}_{\iota=0}^{\lfloor \log_2(q)/r \rfloor}$ and sets $\Delta \leftarrow \left(\{(\beta_\iota, \theta_\iota)\}_{\iota=0}^{\lfloor \log_2(q)/r \rfloor}, \mathsf{pk}_j\right)$.
- When $\mathcal{A}$ calls $\mathsf{O}_{\mathsf{ReEnc}}^{\mathsf{Game}_d}(i,j,\Delta_{i,j},C)$, $\mathcal{B}$ makes the checks described in PostComp, using the same method as for $\mathsf{O}_{\mathsf{ReKeyGen}}$ if $\Delta_{i,j} = \perp$.
- When $\mathcal{A}$ calls $O_{\mathsf{challenge}}^{\mathsf{Game}_d}(i,j,\Delta_{i,j},C_0,C_1)$, $\mathcal{B}$ responds as it would for $\mathsf{O}_{\mathsf{ReEnc}}^{\mathsf{Game}_d}(i,j,\Delta_{i,j},C_b)$.

If $d = 0$ then resamples are genuine so this perfectly simulates $\mathsf{Game}_0^{b,\mathsf{pcBV\text{-}PRE}}$, and if $d = 1$ then this simulates $\mathsf{Game}_1^{b,\mathsf{pcBV\text{-}PRE}}$. Therefore, for an adversary $\mathcal{A}$ that outputs $d'_{\mathcal{A}} = d$ with advantage δ, $\mathcal{B}$ can set $d' = d'_{\mathcal{A}}$ and win $\mathsf{ReSample\text{-}RLWE}_{\mathcal{B}}^{d}(1^\lambda)$ with the same advantage. Overall, $\mathcal{B}$ calls $\mathsf{O}_{\mathsf{challenge}}^{\mathsf{rrlwe}}$ $\lfloor \log_2(q)/r \rfloor + 1$ times for each call $\mathcal{A}$ makes to $\mathsf{O}_{\mathsf{ReEnc}}$, and a further $\lfloor \log_2(q)/r \rfloor + 1$ times whenever $\mathcal{A}$ calls $\mathsf{O}_{\mathsf{ReEnc}}(C, i, j, \Delta_{i,j} = \perp)$ for some C, i, j. Overall, $\mathsf{O}_{\mathsf{challenge}}^{\mathsf{rrlwe}}$ was called at most $(Q_{RK} + Q_{RE\perp})(\lfloor \log_2(q)/r \rfloor + 1)$ times. Lemma 12 gives us the result.

Lemma 14. *Let* $\mathsf{Game}_2^{b,\mathsf{pcBV\text{-}PRE}}$ *be identical to* $\mathsf{Game}_1^{b,\mathsf{pcBV\text{-}PRE}}$*, except that when* $\mathsf{O}_{\mathsf{challenge}}(C_0, C_1, i, j, \Delta_{i,j})$ *is queried, if* $\Delta_{i,j} = \perp$*, the challenger generates an update token as described for calls to* $\mathsf{O}_{\mathsf{ReKeyGen}}$ *in proof of Lemma 13, and the output of the* $\mathsf{ReSample}(\mathsf{pk}_j)$ *used in* $\mathsf{O}_{\mathsf{challenge}}(C_0, C_1, i, j, \Delta_{i,j})$ *is replaced with uniform random values. Then for* $b \in \{0, 1\}$ *and all* PPT *adversaries* $\mathcal{A}$*, there exists an RLWE distinguisher* $\mathcal{D}$ *such that*

$$\left|\Pr\left[1 \xleftarrow{\$} \mathsf{Game}_1^{b,\mathsf{pcBV\text{-}PRE}}(\mathcal{A})\right] - \Pr\left[1 \xleftarrow{\$} \mathsf{Game}_2^{b,\mathsf{pcBV\text{-}PRE}}(\mathcal{A})\right]\right| \\ \leq (\lfloor \log_2(q)/r \rfloor + 2) \cdot \mathbf{Adv} RLWE_{\phi,q,\chi_e}(\mathcal{D}).$$

Proof. First, we note that for the challenge query $\mathsf{O}_{\mathsf{challenge}}(C_0, C_1, i, j, \Delta_{i,j})$, if $\Delta_{i,j} = \perp$ and sk_i is uncorrupted, then Δ cna be replaced with uniform random values as described in proof of Lemma 13. This incurs a security loss of $(\lfloor \log_2(q)/r \rfloor + 1) \cdot \mathbf{Adv} RLWE_{\phi,q,\chi_e}(\mathcal{D})$.

For the remaining step we demonstrate that if there exists an adversary $\mathcal{A}$ that outputs $d'_{\mathcal{A}} = d$ for Game_{d+1}^{b} (where $d \in \{0, 1\}$) with advantage δ, then there exists an adversary $\mathcal{B}$ who can win $\mathsf{ReSample\text{-}RLWE}_{\mathcal{B}}^{d,\mathsf{pcBV\text{-}PRE}}(1^\lambda)$ with the same advantage. As before, we show how $\mathcal{B}$ can use $\mathsf{ReSample\text{-}RLWE}_{\mathcal{B}}^{d}(1^\lambda)$ to simulate Game_{d+1}^{b}. $\mathcal{B}$ first samples $b \xleftarrow{\$} \{0, 1\}$, and responds to all oracles except the challenge as described in proof of Lemma 13. When $\mathcal{A}$ calls $\mathsf{O}_{\mathsf{challenge}}^{\mathsf{Game}_d}(C_0, C_1, i, j, \Delta_{i,j})$, $\mathcal{B}$ performs the appropriate checks, then follows the procedure for $\mathsf{pcBV\text{-}PRE.ReEnc}(\Delta_{i,j}, C_b)$ except that it replaces $\mathsf{ReSample}(\mathsf{pk}_j)$ with a call to $\mathsf{O}_{\mathsf{challenge}}^{\mathsf{rrlwe}}(j)$.

For $\mathsf{ReSample\text{-}RLWE}^{0,\mathsf{pcBV\text{-}PRE}}$, this simulates Game_1 and for $\mathsf{ReSample\text{-}RLWE}^{1,\mathsf{pcBV\text{-}PRE}}$, this simulates Game_2. Therefore, $\mathcal{B}$ can run $\mathcal{A}$ on the simulation and return $d' = d'_{\mathcal{A}}$ to win with the same advantage.

Since $\mathsf{O}_{\mathsf{challenge}}^{\mathsf{rrlwe}}(j)$ is called once for the resample and at most $\lfloor \log_2(q)/r \rfloor + 1$ times for the update token, we leverage RLWE at most $\lfloor \log_2(q)/r \rfloor + 2$ times. □

Lemma 15. *Let* Game_3 *be identical to* Game_2*, except that* $\mathsf{O}_{\mathsf{challenge}}$ *returns uniform random values. For all adversaries* $\mathcal{A}$*:*

$$\Pr\left[1 \xleftarrow{\$} \mathsf{Game}_2^{b,\mathsf{pcBV\text{-}PRE}}(\mathcal{A})\right] = \Pr\left[1 \xleftarrow{\$} \mathsf{Game}_3^{b,\mathsf{pcBV\text{-}PRE}}(\mathcal{A})\right]. \tag{9}$$

Proof. In both Game_2 and Game_3, the adversary submits two ciphertexts $C_0 = (C_{0,0}, C_{0,1})$ and $C_1 = (C_{1,0}, C_{1,1})$ when it queries the challenge oracle. In Game_3, the challenger responds by setting $C_0 = C_{0(b)}$ and $C_1 = C_{1(b)}$ for uniformly chosen $b \xleftarrow{\$} \{0, 1\}$ and returning the challenge ciphertext given by

$$C'_0 = C_0 + \theta^* + \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (C_1^{(i)} \cdot \gamma_i), \quad C'_1 = \beta^* + \sum_{i=0}^{\lfloor \log_2(q)/r \rfloor} (C_1^{(i)} \cdot \beta_i),$$

using the relinearisation breakdown described in Equation (1). In particular, β^* and θ^* are the result of calling $\mathsf{ReSample}(\mathsf{pk}_j)$ in Game_2 on an honest public key and are therefore uniform random values that are used once and never revealed to $\mathcal{A}$. Therefore, C_0' and C_1' are independent uniform random values in Game_2 which is the exact case in Game_3. This argument holds for each of $\mathcal{A}$'s challenge oracle queries. This concludes the proof of this lemma. □

Because the challenge in Lemma 15 can be replaced with random values for both $b = 0$ and $b = 1$, we can use Lemmas 13 to 15 together with the triangle inequality to get:

$$\left|\Pr\left[\mathsf{CPA\text{-}PostComp}_{\mathcal{A}}^{0,\mathsf{pcBV\text{-}PRE}}(1^\lambda) = 1\right] = \Pr\left[\mathsf{CPA\text{-}PostComp}_{\mathcal{A}}^{1,\mathsf{pcBV\text{-}PRE}}(1^\lambda) = 1\right]\right|$$
$$\leq (\kappa + (Q_{RK} + Q_{RE\perp} + 1)(\lfloor \log_2(q)/r \rfloor + 1) + 1) \cdot \mathbf{Adv} RLWE_{\phi,q,\chi_e}(\mathcal{D}).$$

This concludes the proof of Theorem 6. □

Theorem 7. *If there exists a* PPT *adversary $\mathcal{A}$ that can win* $\mathsf{PostComp}_{\mathcal{A}}^{b,\mathsf{pcBV\text{-}PRE}}(1^\lambda)$ *with non-negligible probability, then there exists an adversary $\mathcal{B}$ that can win* $\mathsf{CPA\text{-}PostComp}_{\mathcal{B}}^{b,\mathsf{pcBV\text{-}PRE}}(1^\lambda)$ *with non-negligible probability, by the RLWE assumption.*

Proof. This is similar to the proof outline of [Coh17, Theorem 5]. We demonstrate how an adversary $\mathcal{B}$ for $\mathsf{CPA\text{-}PostComp}_{\mathcal{B}}^{b,\mathsf{pcBV\text{-}PRE}}$ can simulate $\mathsf{PostComp}_{\mathcal{A}}^{b,\mathsf{pcBV\text{-}PRE}}$. $\mathcal{B}$ replaces calls to $\mathsf{O}_{\mathsf{ReEnc}}(C, i, j[, \Delta_{i,j}])$ where $\mathsf{sk}_i \notin \mathcal{K}_{\mathsf{corrupted}}$ $\mathsf{sk}_j \in \mathcal{K}_{\mathsf{corrupted}}$ and $(i, C) \in \mathcal{C}_{\mathsf{honest}}$ but $(i, C) \notin \mathcal{C}_{\mathsf{chal}}$, with a fake re-encryption. If this replacement is not detectable, then $\mathcal{B}$ can call $\mathcal{A}$ as a subroutine, output the same guess and win with the same advantage. We demonstrate how $\mathcal{B}$ can create these fake re-encryptions of ciphertext C, when $\mathsf{pk}_i, \mathsf{pk}_j$ are known but $\Delta_{i,j}$ is unknown, without detection. Without loss of generality, we give the proof for the simplified construction given in Figure 19.

Note that for $b_i = a_i s_i + p e_i$,

$$b_i - a_i s_i = p e_i. \tag{10}$$

```
$C \leftarrow \mathsf{Enc}(\mathsf{pk}_i, m)$
$\tilde{v}, \tilde{e_0}, \tilde{e_1} \xleftarrow{\$} \chi_e$
$C_0 = b_i \tilde{v} + p\tilde{e_0} + m$
$C_1 = a_i \tilde{v} + p\tilde{e_1}$
```

```
$m' \leftarrow \mathsf{Dec}(\mathsf{sk}_i = s_i, C)$
$(C_0, C_1) \leftarrow c$
$m' = C_0 - C_1 s_i \mod p$
```

```
$\Delta_{i,j} \leftarrow \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
$(\bar{a}, \bar{b}) \leftarrow \mathsf{ReSample}(\mathsf{pk}_j)$
$\gamma = \bar{b} - s_i$
$\bar{a} = a_j \bar{v} + p\bar{e_1}, \bar{b} = b_j \bar{v} + p\bar{e_0}$
$\Delta_{i,j} = (\bar{a}, \gamma)$
```

```
$C' \leftarrow \mathsf{ReEnc}(\Delta_{i,j}, C)$
$(\bar{a}, \gamma) \leftarrow \Delta_{i,j}$
$(\hat{a}, \hat{b}) \leftarrow \mathsf{ReSample}(\mathsf{pk}_j)$
$\hat{a} = a_j \hat{v} + p\hat{e_1}, \hat{b} = b_j \hat{v} + p\hat{e_0}$
$C_0' = C_0 + C_1 \gamma + \hat{b}$
$C_1' = C_1 \bar{a} + \hat{a}$
```

Fig. 19: Simplified construction

Let $(\mathsf{pk}_i, \mathsf{sk}_i) = ((a_i, b_i = a_i s_i + p e_i), s_i)$, $(\mathsf{pk}_j, \mathsf{sk}_j) = ((a_j, b_j = a_j s_j + p e_j), s_j)$ and let $C, \Delta_{i,j}, C'$ be as described in Figure 19. We first note that by Equation (4), a genuine re-encryption will result in a ciphertext $C' = (C_0', C_1')$, where

$$\begin{aligned}
C_0' &= C_0 + C_1\gamma + \hat{b} \\
&= C_0 + C_1(\bar{b} - s_i) + \hat{b} \\
&= C_0 - s_iC_1 + C_1\bar{b} + \hat{b} \\
&\overset{4}{=} p(\tilde{e_0}s_i - \tilde{e_1} + e_i\tilde{v}) + m + C_1(b_j\bar{v} + p\bar{e_0}) + b_j\hat{v} + p\hat{e_0} \\
&= b_j(C_1\bar{v} + \hat{v}) + p(C_1\bar{e_0} + \hat{e_0} + \tilde{e_0}s_i - \tilde{e_1} + e_i\tilde{v}) + m, \\
C_1' &= C_1\bar{a} + \hat{a} \\
&= C_1(a_j\bar{v} + p\bar{e_1}) + a_j\hat{v} + p\hat{e_1} \\
&= a_j(C_1\bar{v} + \hat{v}) + p(C_1\bar{e_1} + \hat{e_1}).
\end{aligned}$$

If we let

$$\mathbf{v} = C_1\bar{v} + \hat{v}, \qquad \mathbf{e_0} = C_1\bar{e_0} + \hat{e_0} + \tilde{e_0}s_i - \tilde{e_1} + e_i\tilde{v}, \qquad \mathbf{e_1} = C_1\bar{e_1} + \hat{e_1}, \tag{11}$$

we get

$$\begin{aligned}
C_0' &= b_j\mathbf{v} + p\mathbf{e_0} + m, \\
C_1' &= a_j\mathbf{v} + p\mathbf{e_1}.
\end{aligned} \tag{12}$$

Therefore, if the secret key s_i (and therefore e_i), message m and random values $\tilde{v}, \tilde{e_0}, \tilde{e_1}$ were known, an alternative way to compute a genuine re-encryption would be to sample $\bar{v}, \bar{e_0}, \bar{e_1}, \hat{v}, \hat{e_0}, \hat{e_1} \overset{\$}{\leftarrow} \chi_e$ and compute C' as described in eqs. (11) and (12).

We note that, as $\bar{v}, \bar{e_0}, \bar{e_1}, \hat{v}, \hat{e_0}, \hat{e_1}$ are freshly sampled for one-time use, $\mathcal{B}$ can easily sample its own values from χ_e and use these to calculate $\mathbf{v}, \mathbf{e_1}$. However, since the simulator does not know s_i, e_i, they cannot compute $\mathbf{e_0}$. Instead $\mathcal{B}$ can sample its own $s_i', e_i' \overset{\$}{\leftarrow} \chi_e$ and use these to create

$$\mathbf{e_0'} = C_1\bar{e_0} + \hat{e_0} + \tilde{e_0}s_i' - \tilde{e_1} + e_i'\tilde{v},$$

and create a fake re-encryption

$$C'^* = (b_j\mathbf{v} + p\mathbf{e_0'} + m, a_j\mathbf{v} + p\mathbf{e_1}). \tag{13}$$

We need to demonstrate that this replacement will not be detected.

We note that, since pk_i is uncorrupted, the distinguisher does not know $\mathsf{sk}_i = s_i$, and therefore cannot use C to learn the ciphertext error $E_C = \tilde{e_0}s_i - \tilde{e_1} + e_i\tilde{v}$, otherwise they could derive m from C without knowing the secret key, thereby breaking RLWE. However, as they do know $\mathsf{sk}_j = s_j$, they can compute the error term of a re-encrypted ciphertext. Since

$$\begin{aligned}
C_0' - C_1's_j &= (C_0 + C_1\gamma + \hat{b}) - (C_i\bar{a} + \hat{a})s_j \\
&= C_1(\gamma - \bar{a}s_j) + C_0 + \hat{b} - \hat{a}s_j \\
&= C_1(\bar{b} - s_i - \bar{a}s_j) + C_0 + \hat{b} - \hat{a}s_j \\
&= C_0 + C_1(\bar{b} - \bar{a}s_j - s_i) + (b_j\hat{v} + p\hat{e_0}) - (a_j\hat{v} + p\hat{e_1})s_j \\
&= C_0 - C_1s_i + C_1(\bar{b} - \bar{a}s_j) + (b_j - a_js_j)\hat{v} + p(\hat{e_0} - \hat{e_1}s_j) \\
&\overset{10}{=} m + pE_c + C_1p(e_j\bar{v} + \bar{e_0} - \bar{e_1}s_j) + pe_j\hat{v} + p(\hat{e_0} - \hat{e_1}s_j) \\
&= m + p(E_c + \hat{e_0} - \hat{e_1}s_j + e_j\hat{v} + C_1(e_j\bar{v} + \bar{e_0} - \bar{e_1}s_j)),
\end{aligned}$$

the error on a genuine re-encryption C' is

$$E_{C'} = \tilde{e_0} s_i - \tilde{e_1} + e_i \tilde{v} + \hat{e_0} - \hat{e_1} \underline{s_j} + \underline{e_j} \hat{v} + \underline{C_1}(\underline{e_j} \bar{v} + \bar{e_0} - \bar{e_1} \underline{s_j}),$$

and for a fake re-encryption C'^* is

$$E_{C'^*} = \tilde{e_0} {s_i}' - \tilde{e_1} + {e_i}' \tilde{v} + \hat{e_0} - \hat{e_1} \underline{s_j} + \underline{e_j} \hat{v} + \underline{C_1}(\underline{e_j} \bar{v} + \bar{e_0} - \bar{e_1} \underline{s_j}),$$

where $\underline{\text{underlined}}$ values are known to the adversary.

As $C_1 = a_i \tilde{v} + p\tilde{e_1}$, and a_i is public, (a_i, C_1) is an RLWE sample with secret $\tilde{v}$ and error $p\tilde{e_1}$. Because $\tilde{v}$ and $\tilde{e_1}$ are chosen independently of s_i and e_i, it follows that even if s_i and e_i are later corrupted, $\tilde{v}$ and $\tilde{e_1}$ should remain unknown. As E_c is the only other value derived using $\tilde{v}, \tilde{e_1}$ and it is also unknown to the adversary, we can therefore argue that the C_1 is indistinguishable from a value chosen uniformly at random, by the RLWE assumption.

Rearranging, we need to demonstrate that

$$E_{C'} = \underline{C_1} \bar{e_0} + \hat{e_0} - \hat{e_1} \underline{s_j} + \underline{e_j} \hat{v} + \underline{C_1}(\underline{e_j} \bar{v} - \bar{e_1} \underline{s_j}) + E_c \tag{14}$$

and

$$E_{C'^*} = \underline{C_1} \bar{e_0} + \hat{e_0} - \hat{e_1} \underline{s_j} + \underline{e_j} \hat{v} + \underline{C_1}(\underline{e_j} \bar{v} - \bar{e_1} \underline{s_j}) + E_C^* \tag{15}$$

where $E_C^* = \tilde{e_0} {s_i}' - \tilde{e_1} + {e_i}' \tilde{v}$, appear to have the same distribution to any entity that only knows m, C, s_i and $C'^{(*)}$.

Because C_1 appears to be a uniform element of $\mathcal{R}_q$, $C_1 \bar{e_0} + \hat{e_0}$ is also distributed like an RLWE sample with secret $\bar{e_0}$ and error $\hat{e_0}$. The distributions of Equations (14) and (15) are therefore indistinguishable, under the RLWE assumption, from the distributions of a modified version of Equations (14) and (15) where $C_1 \bar{e_0} + \hat{e_0}$ is replaced with $u \xleftarrow{\$} \mathcal{U}_q$.

If w is an element of $\mathcal{R}_q$ and $u \xleftarrow{\$} \mathcal{U}_q$, then $u + w$ is also distributed as a uniform element of $\mathcal{R}_q$. We can therefore replace the entirety of both E'_C and E'^*_C with $u' \xleftarrow{\$} \mathcal{U}_q$. We conclude that $\mathbf{e_0}$ and $\mathbf{e'_0}$ are indistinguishable.

Thus, $\mathcal{B}$ can produce indistinguishable fake re-encryptions and thus can simulate $\mathsf{PostComp}_{\mathcal{A}}^{\mathsf{pcBV\text{-}PRE}}(1^\lambda)$ and gain the same advantage as $\mathcal{A}$. This completes the proof.

Proof of Theorem 3 – that pcBV-PRE is PCS – follows from Theorem 6 and Theorem 7.

E.3 Source-hiding adaptation of pcBV-PRE

We now describe an adaptation of pcBV-PRE which is provably source-hiding. By the BV-PRE adaptation, we mean pcBV-PRE apart from the fact that there are blurring widths E_ℓ associated to level ℓ encryption. In particular, $\mathsf{Enc}(\mathsf{pk}, m; \ell) = (c_0, c_1) = (bv + pe_0 + m + pf_\ell, av + pe_1)$ where $f_\ell \xleftarrow{\$} [-E_\ell, E_\ell]^n$. In addition, add f_ℓ to c'_0 when re-encrypting to obtain a level ℓ ciphertext too.

Lemma 16. *The adapted version of* BV-PRE *is computationally source-hiding provided that* $E_\ell \gg E_{\ell-1}$ *and* $E_0 \gg (\sigma n)^2$.

Proof. For correctly sampled $\mathsf{aux} = \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{sk}_0, \mathsf{sk}_1, \mathsf{rk}_{0\to1}$, we will show that the distributions of

$$(\mathsf{aux}, \mathsf{Enc}(\mathsf{pk}_0, m; \ell), \mathsf{Enc}(\mathsf{pk}_1, m; \ell + 1)) \tag{16}$$

and

$$(\mathsf{aux}, c := \mathsf{Enc}(\mathsf{pk}_0, m; \ell), \mathsf{ReEnc}(\mathsf{rk}_{0\to 1}, \mathsf{pk}_1, c; \ell + 1)) \tag{17}$$

are computationally indistinguishable for any valid choice of m and ℓ.

Firstly, after writing $\mathsf{pk}_1 = (a_1, b_1 := a_1 \cdot s_1 + e_1)$, for $v, e', e'' \overset{\$}{\leftarrow} \chi_\sigma$ and $f_{\ell+1} \overset{\$}{\leftarrow} [-E_{\ell+1}, E_{\ell+1}]^n$, we have that $(b_1 \cdot v + p(e' + f_{\ell+1}) + m, a_1 \cdot v + pe'')$ or equivalently

$$((a_1 \cdot v) \cdot s_1 + p(e' + e_1 v + f_{\ell+1}) + m, a_1 \cdot v + pe'') \tag{18}$$

is the form of a valid level $\ell + 1$ encryption of m. Let $f''_{\ell+1} = f'_{\ell+1} + (e'' s_1 - e' - e_1 v)$ where $f'_{\ell+1} \overset{\$}{\leftarrow} [-E_{\ell+1}, E_{\ell+1}]$. If $E_{\ell+1} \gg \sigma^2 n^2$ (which is greater than the largest coefficient of $e' - e'' s_1 + e_1 v$ with all but negligible probability), then the distribution of $f_{\ell+1}$ is statistically indistinguishable from the distribution of $f''_{\ell+1}$. Therefore, the distribution of Equation (18) is statistically indistinguishable from the distribution of

$$\big((a_1 \cdot v + pe'') \cdot s_1 + pf'_{\ell+1} + m, a_1 \cdot v + pe''\big)\,.$$

In turn, the above is computationally indistinguishable via a RLWE assumption to

$$\big(u \cdot s_1 + pf'_{\ell+1} + m, u\big)$$

where $u \overset{\$}{\leftarrow} R_q$. Since, the arguments hold for any correctly distributed aux, it follows that the distribution of (16) is computationally indistinguishable from

$$\big(\mathsf{aux}, \mathsf{Enc}(\mathsf{pk}_0, m; \ell), (u_1 \cdot s_0 + pf'_{\ell+1} + m, u_1)\big) \tag{19}$$

where $u_1 \overset{\$}{\leftarrow} R_q$, $f'_\ell \overset{\$}{\leftarrow} [-E_{\ell+1}, E_{\ell+1}]$.

To complete the proof, we will show that the distribution of (17) is also computationally indistinguishable to that the distribution of (19) via a similar argument. Let $\mathsf{pk}_0 = (a_0, b_0 := a_0 \cdot s_0 + e_0)$ and fix a valid ciphertext $(c_0, c_1) = ((a_0 \cdot s_0 + e_0)\bar{v} + p(\bar{e} + f_\ell) + m, a_0 \cdot \bar{v} + p\bar{\bar{e}})$ where $\bar{v}, \bar{e}, \bar{\bar{e}} \overset{\$}{\leftarrow} \chi_\sigma$. Writing $\mathsf{rk}_{0\to 1} = \{\beta_i, \theta_i - s_0 \cdot (2^r)^i\}_{i=0}^{\lfloor \log(q)/r \rfloor}$, we have that $\beta_i \cdot s_1 = \theta_i - p(e_1 \bar{v}_i + \bar{e}_i - p\bar{\bar{e}}_i s_1)$ for some $\bar{v}_i, \bar{e}_i, \bar{\bar{e}}_i$ sampled from χ_σ. In addition, we have the re-randomisation term $(\theta^{\text{proxy}}, \beta^{\text{proxy}})$ satisfying $\beta^{\text{proxy}} \cdot s_1 = \theta^{\text{proxy}} - p(e_1 \bar{v}^{\text{proxy}} + \bar{e}^{\text{proxy}} - p\bar{\bar{e}}^{\text{proxy}} s_1)$ where $\bar{v}^{\text{proxy}}, \bar{e}^{\text{proxy}}, \bar{\bar{e}}^{\text{proxy}}$ are sampled from χ_σ. Let $c'_1 = \sum_{i=0}^{\lfloor \log(q)/r \rfloor} c_1^{(i)} \cdot \beta_i + \beta^{\text{proxy}}$ and $e_c = \sum c_1^{(i)} (e_1 \bar{v}_i + \bar{e}_i - p\bar{e}_i s_1) + e_1 \bar{v}^{\text{proxy}} + \bar{e}^{\text{proxy}} - p\bar{e}^{\text{proxy}} s_1$. We can write a re-encryption as

$$(c'_1 \cdot s_1 + p\,(e_c + e_0 \bar{v} + \bar{e} - \bar{\bar{e}} s + f_\ell + f_{\ell+1}) + m, c'_1)\,. \tag{20}$$

where $f_{\ell+1} \overset{\$}{\leftarrow} [-E_{\ell+1}, E_{\ell+1}]$. Now let $f''_{\ell+1} = f'_{\ell+1} - (e_c + e_0 \bar{v} + \bar{e} - \bar{\bar{e}} s + f_\ell)$ where $f'_{\ell+1} \overset{\$}{\leftarrow} [-E_{\ell+1}, E_{\ell+1}]$. If $E_{\ell+1} \gg \max\{E_\ell, (\log(q)/r) \cdot (n\sigma)^2\}$, then the distribution of $f''_{\ell+1}$ and $f_{\ell+1}$ are statistically indistinguishable. Therefore (20) is distributed statistically close to the distribution of

$$\big(c'_1 \cdot s_1 + pf'_{\ell+1} + m, c'_1\big)\,. \tag{21}$$

Finally, using a RLWE assumption on β^{proxy}, the term c'_1 is statistically indistinguishable from uniform, meaning that the distribution of the above is computationally indistinguishable from $(u \cdot s_1 + pf'_{\ell+1}, u)$ where $u \overset{\$}{\leftarrow} R_q$. Since these arguments work for any valid aux and ciphertext (c_0, c_1), it follows that the distributions of (17) and (19) are computationally indistinguishable.

```
$\mathsf{adIND\text{-}HRA}_{\mathcal{A}}^{b,\mathcal{PRE}}(1^\lambda)$
$\mathcal{K}_{\mathsf{corrupted}}, \mathcal{K}_{\mathsf{chal}}, \mathcal{C}_{\mathsf{chal}}, \mathcal{T}_{\mathsf{honest}}, \mathcal{DRG} = \emptyset$
$\kappa = 0, \mathsf{called} \leftarrow \mathsf{false}$
$b \xleftarrow{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathsf{O_{KeyGen}}, \mathsf{O_{Enc}}, \mathsf{O_{Corrupt}}, \mathsf{O_{ReEnc}}, \mathsf{O_{ReEnc}}, \mathsf{O_{challenge}}}(1^\lambda)$
$\mathcal{K}_{\mathsf{chal}} \leftarrow \mathsf{UpdateChallengeKeys}(\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG})$
if $\mathcal{K}_{\mathsf{corrupted}} \cap \mathcal{K}_{\mathsf{chal}} \neq \emptyset$ :
  return $0$
else return $b'$
```

```
$\mathsf{O_{ReEnc}}(C, i, j, [\Delta_{i,j}])$
if $\Delta_{i,j}$ given AND $(i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}}$ :
  return $\perp$
if $\Delta_{i,j}$ not given :
  $\Delta_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
$C' \xleftarrow{\$} \mathsf{ReEnc}(\Delta_{i,j}, C)$
if $(i, C) \in \mathcal{C}_{\mathsf{chal}}$ :
  $\mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{(\}j, C'), \mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\}\mathsf{sk}_j$
return $C'$
```

```
$\mathsf{O_{challenge}}(m_0, m_1, i)$
if $|m_0| \neq |m_1|$ OR $\mathsf{called} = \mathsf{true}$ :
  return $\perp$
$C \xleftarrow{\$} \mathsf{ReEnc}(\mathsf{pk}_i, m_b)$
$\mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{(\}i, C)$
$\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{\mathsf{sk}\}_i$
$\mathsf{called} \leftarrow \mathsf{true}$
return $C$
```

Fig. 20: The adIND-HRA game. Like the HRA model [Coh17], it allows re-encryptions of non-challenge ciphertexts to compromised keys using $\mathsf{O_{ReEnc}}$. It allows adaptive corruption of keys subject to the trivial win condition that no key which a challenge ciphertext has been learned under can be corrupted.

F Adaptive HRA security

Here we describe an extension of selective IND-HRA security as described in Section 3.1, which allows the adversary to adaptively corrupt keys in response after receiving challenges.

Definition 17. *A PRE scheme $\mathcal{PRE}$ is said to be* ϵ-Adaptively Indistinguishable Honest Re-encryption Plaintext Attacks-secure (ϵ-adIND-HRA-secure) *if for all* PPT *adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$:*

$$\left|\Pr\left[\mathsf{adIND\text{-}HRA}_{\mathcal{A}}^{0,\mathcal{PRE}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{adIND\text{-}HRA}_{\mathcal{A}}^{1,\mathcal{PRE}}(1^\lambda) = 1\right]\right| \leq \epsilon, \qquad (22)$$

where $\mathsf{adIND\text{-}HRA}_{\mathcal{A}}^{\mathcal{PRE}}$ *is given in Figure 20. If ϵ is negligible as parameterised by the security parameter, then we say the scheme is* Adaptively Indistinguishable Honest Re-encryption Plaintext Attacks-secure (adIND-HRA-secure)*.*

In the adIND-HRA game (Figure 20), $\mathcal{A}$ can adaptively corrupt keys subject to the trivial win condition that they cannot have corrupted a key with which a challenge ciphertext can be decrypted. In order to keep track of the status of keys, the lists $\mathcal{C}_{\mathsf{chal}}, \mathcal{K}_{\mathsf{chal}}$ and $\mathcal{K}_{\mathsf{corrupted}}$ are used as well as a directed re-encryption graph $\mathcal{DRG}$ which tracks update token queries. $\mathcal{C}_{\mathsf{chal}}$ contains outputs of the challenge oracle $\mathsf{O_{challenge}}$ as well as outputs of $\mathsf{O_{ReEnc}}$ when given a challenge as input. $\mathcal{K}_{\mathsf{corrupted}}$ contains the outputs of $\mathsf{O_{Corrupt}}$ queries. $\mathcal{DRG}$ consists of nodes v_i that represent key pairs, and edges $\overrightarrow{e}_{i,j}$ which are added when $\mathsf{O_{ReKeyGen}}(i, j)$ is queried. Using update tokens, the adversary can locally re-encrypt challenge ciphertexts. Therefore, if a challenge ciphertext is and encryption under pk_i, and there exists a sequence of tokens going from i to j, then both sk_i and sk_j are considered challenge keys. Represented using the graph $\mathcal{DRG}$, if there is a path from v_i to v_j and sk_i is a challenge key, then so is sk_j. At the end of the game, $\mathcal{DRG}$ is used to update the list of challenge keys. Then the trivial winning condition translates to $\mathcal{K}_{\mathsf{chal}} \cap \mathcal{K}_{\mathsf{corrupted}}$ being empty.

G Adaptive Post-Compromise Security

Here we give the explicit definition for adaptive PCS.

```
$\mathsf{ad\text{-}PostComp}_{\mathcal{A}}^{b,\mathcal{PRE}}(1^\lambda)$
-----------------------------------------
$\mathcal{K}_{\mathsf{corrupted}}, \mathcal{K}_{\mathsf{chal}}, \mathcal{C}_{\mathsf{honest}}, \mathcal{C}_{\mathsf{chal}}, \mathcal{T}_{\mathsf{honest}}, \mathcal{DRG} = \emptyset$
$\kappa = 0, \mathsf{called} = \mathsf{false}$
$b' \leftarrow \mathcal{A}^{\mathsf{O_{KeyGen}}, \mathsf{O_{Enc}}, \mathsf{O_{Corrupt}}, \mathsf{O_{ReKeyGen}} \mathsf{O}^{\mathsf{aPC}}_{\mathsf{ReEnc}}, \mathsf{O}^{\mathsf{aPC}}_{\mathsf{challenge}}}(1^\lambda)$
$\mathcal{K}_{\mathsf{chal}} \leftarrow \mathsf{UpdateChallengeKeys}(\mathcal{K}_{\mathsf{chal}}, \mathcal{DRG})$
if $\mathcal{K}_{\mathsf{corrupted}} \cap \mathcal{K}_{\mathsf{chal}} \neq \emptyset$ :
  return $\perp$
else return $b'$
```

```
$\mathsf{O}^{\mathsf{aPC}}_{\mathsf{ReEnc}}(C, i, j, [\Delta_{i,j}])$
-----------------------------------------
if $|C_0| \neq |C_1|$ OR $\mathsf{called} = \mathsf{true}$ : return $\perp$
if $(i, C_0), (i, C_1) \notin \mathcal{C}_{\mathsf{honest}}$ : return $\perp$
if $(i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}}$ : return $\perp$
if not given : $\Delta_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_j)$
$C' \xleftarrow{\$} \mathsf{ReEnc}(\Delta_{i,j}, C)$
if $(i, C) \in \mathcal{C}_{\mathsf{honest}}$ : $\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{(\}j, C')$
if $(i, C) \in \mathcal{C}_{\mathsf{chal}}$ :
  $\mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{(\}j, C')$
  $\mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{(\}\mathsf{sk}_j)$
return $C'$
```

```
$\mathsf{O}^{\mathsf{aPC}}_{\mathsf{challenge}}(C_0, C_1, i, j, \Delta_{i,j})$
-----------------------------------------
if $|C_0| \neq |C_1|$ : return $\perp$
if $(i, C_0), (i, C_1) \notin \mathcal{C}_{\mathsf{honest}}$ : return $\perp$
if $(i, j, \Delta_{i,j}) \notin \mathcal{T}_{\mathsf{honest}}$ : return $\perp$
$C' \xleftarrow{\$} \mathsf{ReEnc}(\Delta_{i,j}, C_b)$
$\mathcal{C}_{\mathsf{honest}}.\mathsf{add}\{(\}j, C'), \mathcal{C}_{\mathsf{chal}}.\mathsf{add}\{(\}j, C'), \mathcal{K}_{\mathsf{chal}}.\mathsf{add}\{(\}\mathsf{sk}_j)$
$\mathsf{called} \leftarrow \mathsf{false}$
return $C'$
```

Fig. 21: The adaptive post compromise game PostComp. This is stronger than the selective version as the adversary can choose to corrupt keys as a result of challenge queries, subject to the trivial win condition.

Definition 18. *A PRE scheme $\mathcal{PRE}$ is said to have* ϵ-Adaptive Post-Compromise Security (ϵ-adPCS) *if for all* PPT *adversaries $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$:*

$$\left|\Pr\left[\mathsf{ad\text{-}PostComp}^0_{\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{ad\text{-}PostComp}^1_{\mathcal{A}}(1^\lambda) = 1\right]\right| \leq \epsilon,$$

where $\mathsf{ad\text{-}PostComp}\mathcal{A}^{b,\mathcal{PRE}}$ *is defined in Figure 21. If ϵ is negligible as parameterised by the security parameter, then we say the scheme has* Adaptive Post-Compromise Security (adPCS).

H Weak key privacy of our construction

We now demonstrates that pcBV-PRE has weak key privacy (**??**). Recall that this property is necessary in order to use the results of [FKKP18].

Theorem 8. $\mathsf{pcBV\text{-}PRE}$ *has weak key privacy.*

Proof. Once again, the proof replaces the public key and $\mathsf{ReSample}$ outputs associated to identity labels $i = 1, \ldots, \kappa$ by uniform values one by one. This is permissible due to the RLWE assumption. Define the following sequence of games:

- Game_0: The $\mathsf{weakKP}_{\mathcal{A}}^{b,\mathcal{PRE}}(1^\lambda, 1^\kappa)$ game.
- Game'_i for $i = 1, \ldots, \kappa$: The same as Game_{i-1} apart from the fact that pk_i is replaced with a uniform value.
- Game_i for $i = 1, \ldots, \kappa$: The same as Game'_i except that the output of calls to $\mathsf{ReKeyGen}(\cdot, \mathsf{pk}_i)$ are replaced by a uniform value.

The transitions or game hops occur in the order $\mathsf{Game}_0 \to \mathsf{Game}'_1 \to \mathsf{Game}_1 \to \cdots \to \mathsf{Game}'_\kappa \to \mathsf{Game}_\kappa$. We first note that the secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_\kappa$ are never used in any of the games so we ignore these throughout this proof. Game_{i-1} can be seen to be indistinguishable from Game'_i by acknowledging that the difference between these games is that a single RLWE public key is replaced by a uniform value. Next we need to argue that Game'_i is indistinguishable from Game_i. To do so, we consider a sequence of hybrids between Game'_i and Game_i denoted as $\mathsf{Game}'_{i,j}$ where $j = 0 \ldots \lceil \log_2(q)/r \rceil$. Essentially, in $\mathsf{Game}'_{i,j}$, the first j calls of the form $\mathsf{ReSample}(pk_i)$ are replaced by uniform random values. Recall that for both these games, the public key $\mathsf{pk}_i = (a_1, a_2)$ is a uniform random value itself. Therefore, the difference between $\mathsf{Game}'_{i,j}$ and $\mathsf{Game}'_{i,j+1}$ is that the $(j+1)^{th}$ call to $\mathsf{ReSample}$ denoted by (b_1, b_2) either has the form $(a_1 v + e', a_2 v + e'')$ or takes the form of a uniform random value. Considering the pairs (a_1, b_1) and (a_2, b_2), an adversary cannot distinguish between $\mathsf{Game}'_{i,j}$ and $\mathsf{Game}'_{i,j+1}$ with non-negligible advantage according to the RLWE assumption. This holds for $j = 0, \ldots, \lceil \log_2(q)/r \rceil - 1$. Since we have $\mathsf{Game}'_{i,0} = \mathsf{Game}'_i$ and $\mathsf{Game}'_{i,\lceil \log_2(q)/r \rceil} = \mathsf{Game}_i$, we can conclude that Game'_i is indistinguishable from Game_i by using the RLWE assumption multiple times. Iterating through $i = 0$ to κ completes the proof. $\square$